gclib 2.0.9

C API for Galil controllers and PLCs

Galil Motion Control

Mon Dec 16 2024

Contents

Chapter 1

Introduction

The Galil Communication Library (gclib) is a communication library for Galil motion controllers and PLCs. A number of programming languages, operating systems, and hardware platforms are supported.

The library consists of a basic set of function calls (gclib.h), and an open-source extension library (gclibo.h). A number of examples are provided to demonstrate how to use the library with various languages.

gclib will import virtually anywhere a dll/so/dylib can be imported. Please contact support@galil.com if the language or platform required is not listed.

Installation

- Windows
- Linux

License

gclib binaries are covered under the Galil Closed Source License.

The open source portion (gclibo), examples, and wrappers are covered under the Galil Open Source License.

gclib and gcaps use OpenSSL, which is licensed under the Apache 2.0 license.

Getting Started

- Language Support
- Using gclib
- Example Projects
- **List of all functions**

Release Notes

See the update history of gclib in the release notes.

Galil maintains an RSS page to notify users of updates.

See the update history of gcaps in the release notes.

Technical Support

For help please email support@galil.com, or call Galil Applications.

1.1 Language Support

Below are a number of examples demonstrating how to use the library with various languages and on various platforms.

- C/C++
- Python
- .Net
- Java
- LabVIEW

Can't find what you need? Please email support@galil.com, or call Galil Applications.

1.1.1 C/C++

- Microsoft Visual Studio 2019 (16.0)
- Microsoft Visual Studio 2017 (15.0)
- Microsoft Visual Studio 2015 (14.0)
- Microsoft Visual Studio 2013 (12.0)
- MinGW
- Borland C++
- gcc (Linux)
- clang (OS X)

1.1.1.1 Microsoft Visual Studio 2019 (16.0)

For brevity, these instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib**.

x_simple.c from *VS2019 x64 Native Tools Command Prompt*

Open *x64 Native Tools Command Prompt for VS 2019*.

Copy files

Navigate to a convenient, empty, writable location.

Set an environment variable for the base path.

```
>set base=C:\Program Files (x86)\Galil\gclib
```

Copy simple example

```
>copy "%base%\examples\cpp\x_simple.c" .
```

Edit GOpen() call as necessary

In a text editor, open *x_simple.c*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.

Compile

```
>cl x_simple.c "%base%\lib\dynamic\x64\*.lib" -I "%base%\include"
```

Set Path to DLL

```
>set PATH=%base%\dll\x64\;%PATH%
```

Execute

```
>x_simple.exe
version: 211.211.211 1.0.0.128
info: 192.168.0.42, DMCC640 Rev 1.0g, 9999
response:  3757802.0000
:
```

Using the pre-configured MSVC project (x_examples.cpp)

The directory *gclib\examples\msvc* has fully functional MSVC examples. These instructions detail how to use the 2019 version.

- Copy *gclib\examples\msvc\2019_16.0\gclib_example* to a convenient, writable location.
- Run *gclib_example\gclib_example\copy_source.bat* to copy the files.
- Open *gclib_example\gclib_example.sln* in Visual Studio 2019.
- In the *Solution Explorer*, expand the *gclib_example* and expand *Source Files* to show a listing of source.
- Open *x_examples.cpp*.
- Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.
- Find the `#if 0` preprocessor block enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.
- Hit *F5* to build and run the example.

Create Project with MSVC 2019 (x_examples.cpp)

The instructions below allow building a project from scratch.

The following instructions were performed on *Visual Studio Professional 2019* and can be extended to other Visual Studio versions. For brevity, the instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib** and a build type of **x64**.

- Launch *Visual Studio 2019*.

- At the initial start window, Choose *Create a new project*.
- In the *Create a new project* window, choose *Empty Project* and click *Next*.
- Choose a Name, e.g. **gclib_example**.
- Choose a Location, e.g. *C:\Users\user\Desktop*.
- Uncheck *Place solution and project in the same directory*.
- Click *Create*.
- In the *Solution Explorer*, right-click on *Source Files* and choose *Add->Existing Item*.
  - Navigate to the gclib installation directory, then to *examples\cpp* in the installation directory.
  - In *File Name* type **x_*.cpp** and click *Add*, this will filter out the files needed
  - Select all files in the file chooser and click *Add*.
- In the *Solution Explorer* right-click on *gclib_example*, choose *Properties*.
  - Click the *Configuration Manager...* button.
    - Under *Active solution platform:* choose *x64*.
    - Click *Close*.
  - Highlight *Configuration Properties* in the side bar, and set the following project properties.
    - At the top of the window, change *Configuration:* to *All Configurations* and ensure *Platform* lists *Active(x64)*.
    - *Configuration Properties -> C/C++ -> General -> Additional Include Directories* add **C:\Program Files (x86)\Galil\gclib\include**
    - *Configuration Properties -> Linker -> General -> Additional Library Directories* add **C:\Program Files (x86)\Galil\gclib\lib\dynamic\x64**
    - *Configuration Properties -> Linker -> Input -> Additional Dependencies* add **gclib.lib;gclibo.↩ lib;**`{rest of text}` where `{rest of text}` is the original string that was in the cell. Note the semicolons between library files.
    - *Configuration Properties -> Debugging -> Environment* add **PATH=C:\Program Files (x86)\Galil\gclib\dll\x64;%PATH%**
    - Click *OK*.
- In the *Solution Explorer* open *x_examples.cpp*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.
- Find the `#if 0` preprocessor blocks enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.
- Hit *F5* to build and run the example.

1.1.1.2 Microsoft Visual Studio 2017 (15.0)

For brevity, these instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib**.

x_simple.c from *VS2017 x64 Native Tools Command Prompt*

Open *x64 Native Tools Command Prompt for VS 2017*.

Copy files

Navigate to a convenient, empty, writable location.

Set an environment variable for the base path.

```
>set base=C:\Program Files (x86)\Galil\gclib
```

Copy simple example

```
>copy "%base%\examples\cpp\x_simple.c" .
```

Edit GOpen() call as necessary

In a text editor, open *x_simple.c*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.

Compile

```
>cl x_simple.c "%base%\lib\dynamic\x64\*.lib" -I "%base%\include"
```

Set Path to DLL

```
>set PATH=%base%\dll\x64\;%PATH%
```

Execute

```
>x_simple.exe
version: 211.211.211 1.0.0.128
info: 192.168.0.42, DMCC640 Rev 1.0g, 9999
response:  3757802.0000
:
```

Using the pre-configured MSVC project (x_examples.cpp)

The directory *gclib\examples\msvc* has fully functional MSVC examples. These instructions detail how to use the 2017 version.

- Copy *gclib\examples\msvc\2017_15.0\gclib_example* to a convenient, writable location.
- Run *gclib_example\gclib_example\copy_source.bat* to copy the files.
- Open *gclib_example\gclib_example.sln* in Visual Studio 2017.
- In the *Solution Explorer*, expand the *gclib_example* and expand *Source Files* to show a listing of source.
- Open *x_examples.cpp*.
- Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.
- Find the `#if 0` preprocessor block enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.
- Hit *F5* to build and run the example.

Create Project with MSVC 2017 (x_examples.cpp)

The instructions below allow building a project from scratch.

The following instructions were performed on *Visual Studio Professional 2017* and can be extended to other Visual Studio versions. For brevity, the instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib** and a build type of **x64**.

- Launch *Visual Studio 2017*.
- Choose *File->New->Project*.
- In the *New Project* dialog, choose *Visual C++->Empty Project*.
- Choose a Name, e.g. **gclib_example**.
- Choose a Location, e.g. *C:\Users\user\Desktop*.
- Check *Create directory for solution*.
- Click *OK*.
- In the *Solution Explorer*, right-click on *Source Files* and choose *Add->Existing Item*.
  - Navigate to the gclib installation directory, then to *examples\cpp* in the installation directory.
  - In *File Name* type **x_*.cpp** and click *Add*, this will filter out the files needed
  - Select all files in the file chooser and click *Add*.
- In the *Solution Explorer* right-click on *gclib_example*, choose *Properties*.
  - Click the *Configuration Manager...* button.
    - Under *Active solution platform:* choose *x64*.
    - Click *Close*.
  - Highlight *Configuration Properties* in the side bar, and set the following project properties.
    - At the top of the window, change *Configuration:* to *All Configurations* and ensure *Platform* lists *Active(x64)*.
    - *Configuration Properties -> C/C++ -> General -> Additional Include Directories* add **C:\Program Files (x86)\Galil\gclib\include**
    - *Configuration Properties -> Linker -> General -> Additional Library Directories* add **C:\Program Files (x86)\Galil\gclib\lib\dynamic\x64**
    - *Configuration Properties -> C/C++ -> Code Generation -> Spectre Mitigation* set to **Disabled**. If your application will cross trust boundaries, consider Spectre and Meltdown vulnerabilities before deploying.
    - *Configuration Properties -> Linker -> Input -> Additional Dependencies* add **gclib.lib;gclibo.↩ lib;**`{rest of text}` where `{rest of text}` is the original string that was in the cell. Note the semicolons between library files.
    - *Configuration Properties -> Debugging -> Environment* add **PATH=C:\Program Files (x86)\Galil\gclib\dll\x64;%PATH%**
    - Click *OK*.
- In the *Solution Explorer* open *x_examples.cpp*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.
- Find the `#if 0` preprocessor blocks enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.
- Hit *F5* to build and run the example.

1.1.1.3 Microsoft Visual Studio 2015 (14.0)

For brevity, these instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib**.

x_simple.c from *VS2015 x64 Native Tools Command Prompt*

Open *VS2015 x64 Native Tools Command Prompt*.

Copy files

Navigate to a convenient, empty, writable location.

Set an environment variable for the base path.

```
>set base=C:\Program Files (x86)\Galil\gclib
```

Copy simple example

```
>copy "%base%\examples\cpp\x_simple.c" .
```

Edit GOpen() call as necessary

In a text editor, open *x_simple.c*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.

Compile

```
>cl x_simple.c "%base%\lib\dynamic\x64\*.lib" -I "%base%\include"
```

Set Path to DLL

```
>set PATH=%base%\dll\x64\;%PATH%
```

Execute

```
>x_simple.exe
version: 211.211.211 1.0.0.128
info: 192.168.0.42, DMCC640 Rev 1.0g, 9999
response:  3757802.0000
:
```

Using the pre-configured MSVC project (x_examples.cpp)

The directory *gclib\examples\msvc* has fully functional MSVC examples. These instructions detail how to use the 2015 version.

- Copy *gclib\examples\msvc\2015_14.0\gclib_example* to a convenient, writable location.
- Run *gclib_example\gclib_example\copy_source.bat* to copy the files.
- Open *gclib_example\gclib_example.sln* in Visual Studio 2015.
- In the *Solution Explorer*, expand the *gclib_example* and expand *Source Files* to show a listing of source.

- Open *x_examples.cpp*.
- Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.
- Find the `#if 0` preprocessor block enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.
- Hit *F5* to build and run the example.

Create Project with MSVC 2015 (x_examples.cpp)

The instructions below allow building a project from scratch.

The following instructions were performed on *Visual Studio Professional 2015* and can be extended to other Visual Studio versions. For brevity, the instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib** and a build type of **x64**.

- Launch *Visual Studio 2015*.
- Choose *File->New->Project*.
- In the *New Project* dialog, choose *Visual C++->Empty Project*.
- Choose a Name, e.g. **gclib_example**.
- Choose a Location, e.g. *C:\Users\user\Desktop*.
- Check *Create directory for solution*.
- Click *OK*.
- In the *Solution Explorer*, right-click on *Source Files* and choose *Add->Existing Item*.
  - Navigate to the gclib installation directory, then to *examples\cpp* in the installation directory.
  - In *File Name* type **x_*.cpp** and click *Add*, this will filter out the files needed
  - Select all files in the file chooser and click *Add*.
- In the *Solution Explorer* right-click on *gclib_example*, choose *Properties*.
  - Click the *Configuration Manager...* button.
    - Under *Active solution platform:* choose *x64*.
    - Click *Close*.
  - Highlight *Configuration Properties* in the side bar, and set the following project properties.
    - At the top of the window, change *Configuration:* to *All Configurations* and ensure *Platform* lists *Active(x64)*.
    - *Configuration Properties -> C/C++ -> Additional Include Directories* add **C:\Program Files (x86)\Galil\gclib\include**
    - *Configuration Properties -> Linker -> General -> Additional Library Directories* add **C:\Program Files (x86)\Galil\gclib\lib\dynamic\x64**
    - *Configuration Properties -> Linker -> Input -> Additional Dependencies* add **gclib.lib;gclibo.lib;**`{rest of text}` where `{rest of text}` is the original string that was in the cell. Note the semicolons between library files.
    - *Configuration Properties -> Debugging -> Environment* add **PATH=C:\Program Files (x86)\Galil\gclib\dll\x64;%PATH%**
    - Click *OK*.
- In the *Solution Explorer* open *x_examples.cpp*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.

- Find the `#if 0` preprocessor blocks enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.
- Hit *F5* to build and run the example.

1.1.1.4 Microsoft Visual Studio 2013 (12.0)

For brevity, these instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib**.

x_simple.c from *VS2013 x64 Native Tools Command Prompt*

Open *VS2013 x64 Native Tools Command Prompt*.

Copy files

Navigate to a convenient, empty, writable location, e.g. *C:\temp*.

Set an environment variable for the base path.

```
C:\temp>set base=C:\Program Files (x86)\Galil\gclib
```

Copy simple example

```
C:\temp>copy "%base%\examples\cpp\x_simple.c" .
```

Edit GOpen() call as necessary

In a text editor, open *x_simple.c*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.

Compile

```
C:\temp>cl x_simple.c "%base%\lib\dynamic\x64\*.lib" -I "%base%\include"
```

Set Path to DLL

```
C:\temp>set PATH=%base%\dll\x64\;%PATH%
```

Execute

```
C:\temp>x_simple.exe
rc: 0
version: 85.60.138
rc: 0
rc: 0
info: 10.1.3.17, DMC4020 Rev 1.2b, 291
rc: 0
response:  357247808.0000
:
```

Using the pre-configured MSVC project (x_examples.cpp)

The directory *gclib\examples\msvc* has fully functional MSVC examples. These instructions detail how to use the 2013 version.

- Copy *gclib\examples\msvc\2013_12.0\gclib_example* to a convenient, writable location, e.g. *C:\temp*.
- Run *gclib_example\gclib_example\copy_source.bat* to copy the files.
- Open *gclib_example\gclib_example.sln* in Visual Studio 2013.
- In the *Solution Explorer*, expand the *gclib_example* and expand *Source Files* to show a listing of source.
- Open *x_examples.cpp*
- Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.
- Find the `#if 0` preprocessor block enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.
- Hit *F5* to build and run the example.

Create Project with MSVC 2013 (x_examples.cpp)

The instructions below allow building a project from scratch.

The following instructions were performed on *Visual Studio Professional 2013* and can be extended to other Visual Studio versions. For brevity, the instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib** and a build type of **x86 (win32)**.

- Launch *Visual Studio 2013*
- Choose *File->New->Project*
- In the *New Project* dialog, choose *Visual C++->Empty Project*
- Choose a Name, e.g. **gclib_example**
- Choose a Location, e.g. *C:\Users\user\Desktop*
- Check *Create directory for solution*
- Click *OK*
- In the *Solution Explorer*, right-click on *Source Files* and choose *Add->Existing Item*
  - Navigate to the gclib installation directory, then to *examples\cpp* in the installation directory
  - In *File Name* type **x_*.cpp** and click *Add*, this will filter out the files needed
  - Select all files in the file chooser and click *Add*
- In the *Solution Explorer* right-click on *gclib_example*, choose *Properties*, highlight *Configuration Properties*, and set the following project properties
  - At the top of the window, change *Configuration:* to *All Configurations* and ensure *Platform* lists *Active(↩ Win32)*
  - *Configuration Properties -> C/C++ -> Additional Include Directories* add **C:\Program Files (x86)\Galil\gclib\include**
  - *Configuration Properties -> Linker -> General -> Additional Library Directories* add **C:\Program Files (x86)\Galil\gclib\lib\dynamic\x86**

  - *Configuration Properties -> Linker -> Input -> Additional Dependencies* add **gclib.lib;gclibo.lib;**`{rest of text}` where `{rest of text}` is the original string that was in the cell. Note the semicolons between library files.
  - *Configuration Properties -> Debugging -> Environment* add **PATH=C:\Program Files (x86)\Galil\gclib\dll\x86;%PATH%**

- In the *Solution Explorer* open *x_examples.cpp*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.
- Find the `#if 0` preprocessor block enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.
- Hit *F5* to build and run the example.

1.1.1.5 MinGW

The following instructions were performed with x86 Minimalist GNU for Windows (MinGW) installed from http://mingw-w64.sourceforge.net/download.php#mingw-builds

For brevity, these instructions assume the default installation location of "C:\Program Files (x86)\Galil\gclib".

Copy Files

Copy "gclib\examples\mingw" to a convenient, writable location, e.g. "C:\temp". Run `C:\temp\mingw\copy_source.bat` to copy all files.

x_simple.c

Edit GOpen() call as necessary

In a text editor, open *x_simple.c*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.

Compile

- Launch the MinGW terminal, e.g. *Start -> All Programs -> MinGW-W64 project -> i686-4.9.1-posix-dwarf-rt_v3-rev3 -> Run Terminal*.
- Navigate to the directory with the files above.
- Compile the code.

```
C:\temp\mingw\>gcc x_simple.c -L. -lgclibo -lgclib -o simple.exe
```

Execute

```
C:\temp\mingw>simple.exe
rc: 0
version: 85.60.138
rc: 0
rc: 0
info: 10.1.3.17, DMC4020 Rev 1.2b, 291
rc: 0
response:  1584328.0000
:
```

x_examples.cpp

Review and Modify source

- In a text editor, open *x_examples.cpp*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.
- Find the `#if 0` preprocessor block enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.

Compile

- Launch the MinGW terminal, e.g. *Start -> All Programs -> MinGW-W64 project -> i686-4.9.1-posix-dwarf-rt_v3-rev3 -> Run Terminal*.
- Navigate to the directory with the files above.
- Compile the code.

```
C:\temp\mingw>g++ *.cpp -L. -lgclibo -lgclib -o examples.exe
```

Execute

```
C:\temp\mingw>examples.exe
Library version: 41.35.34

192.168.0.43, DMC4020 Rev 1.2b, 291


************************************************************************
Example GRead() and GWrite() usage
************************************************************************

Read 155 QR bytes.

************************************************************************
Example GCommand() usage
************************************************************************
Revision report, ^R^V
DMC4020 Rev 1.2b
:

Command Values
val is 10
val is 11
val is 3.1415
val is 9.869

Command Trimming
> 95653016.0000
:<
> 95653016.0000<
>95653016.0000<

Receiving Binary Data
QR read 155 bytes

Error handling
QD correctly trapped, not allowed, try GArrayDownload()
DL correctly trapped, not allowed, try GProgramDownload()

Modifying timeout
Burning program...OK

************************************************************************
Example GProgramDownload() and GProgramUpload() usage
```

```
************************************************************************
GProgramDownload() correctly errored. Can't fit with level 3 compression
Program Downloaded with compression level 4
Uploading program:
#A;i=0;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i
i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;EN

Program executed as expected
************************************************************************
Example GArrayDownload() and GArrayUpload() usage
************************************************************************
2.0000, 4.0000, 6.0000, 8.0000, 10.0000, 12.0000, 14.0000, 16.0000, 18.000
0000

2.0000, 1.0000, 3.0000, 5.0000, 10.0000, 12.0000, 14.0000, 16.0000, 18.000
0000

3.0000, 5.0000, 10.0000

************************************************************************
Example GRecord() usage
************************************************************************

QR-based data record
38564
393216000

DR-based data record
38670
38772
38874
38976
39078
39180
39282
39384
39486
39588
39690

QR-based data record with offsets
39692
39692

************************************************************************
Example GMessage() usage
************************************************************************
 0.0000
 1.0000
 2.0000
 3.0000
 4.0000
 5.0000
 6.0000
 7.0000
 8.0000
 9.0000

************************************************************************
Example GInterrupt() usage
************************************************************************
"UI 8" executed.

************************************************************************
Example GMotionComplete() usage
************************************************************************

Position: 0, 0
Beginning independent motion... Motion Complete on A
Position: 8000, 0

Position: 0, 0
Beginning vector motion... Motion Complete on vector plane S
```

```
Position: 6000, 0


examples.cpp executed OK
main() is finished. Press Enter to exit:
```

1.1.1.6 Borland C++

The following instructions were performed on:

```
Embarcadero C++ 7.10 for Win32 Copyright (c) 1993-2015 Embarcadero Technologies, Inc.
```

For brevity, these instructions assume the default installation location of "C:\Program Files (x86)\Galil\gclib".

Copy Files

Copy "gclib\examples\borland" to a convenient, writable location, e.g. "C:\temp". Run `C:\temp\borland\copy_source.bat` to copy all files.

```
C:\temp>cd borland

C:\temp\borland>copy_source.bat
\Program Files (x86)\Galil\gclib\examples\cpp\x_arrays.cpp
\Program Files (x86)\Galil\gclib\examples\cpp\x_examples.cpp
\Program Files (x86)\Galil\gclib\examples\cpp\x_examples.h
\Program Files (x86)\Galil\gclib\examples\cpp\x_gcommand.cpp
\Program Files (x86)\Galil\gclib\examples\cpp\x_ginterrupt.cpp
\Program Files (x86)\Galil\gclib\examples\cpp\x_gmessage.cpp
\Program Files (x86)\Galil\gclib\examples\cpp\x_gmotioncomplete.cpp
\Program Files (x86)\Galil\gclib\examples\cpp\x_gread_gwrite.cpp
\Program Files (x86)\Galil\gclib\examples\cpp\x_grecord.cpp
\Program Files (x86)\Galil\gclib\examples\cpp\x_nonblocking.cpp
\Program Files (x86)\Galil\gclib\examples\cpp\x_programs.cpp
\Program Files (x86)\Galil\gclib\examples\cpp\x_simple.c
       12 file(s) copied.
\Program Files (x86)\Galil\gclib\include\gclib.h
\Program Files (x86)\Galil\gclib\include\gclibo.h
\Program Files (x86)\Galil\gclib\include\gclib_errors.h
\Program Files (x86)\Galil\gclib\include\gclib_record.h
        4 file(s) copied.
\Program Files (x86)\Galil\gclib\lib\dynamic\x86\gclib.lib
\Program Files (x86)\Galil\gclib\lib\dynamic\x86\gclibo.lib
        2 file(s) copied.
\Program Files (x86)\Galil\gclib\dll\x86\gclib.dll
\Program Files (x86)\Galil\gclib\dll\x86\gclibo.dll
        2 file(s) copied.

C:\temp\borland>
```

Modify Path

- Add Borland's compiler to the `PATH` variable.

```
C:\temp\borland>set PATH=c:\Program Files (x86)\Embarcadero\Studio\17.0\bin;%PATH%
```

Convert lib files

```
C:\temp\borland>move gclib.lib _gclib.lib
    1 file(s) moved.

C:\temp\borland>move gclibo.lib _gclibo.lib
    1 file(s) moved.
```

```
C:\temp\borland>coff2omf.exe _gclib.lib gclib.lib
COFF to OMF Converter Version 1.2.0 Copyright (c) 1999-2009 Embarcadero Technologies, Inc.
All rights reserved.

C:\temp\borland>coff2omf.exe _gclibo.lib gclibo.lib
COFF to OMF Converter Version 1.2.0 Copyright (c) 1999-2009 Embarcadero Technologies, Inc.
All rights reserved.
```

x_simple.c

Edit GOpen() call as necessary

In a text editor, open *x_simple.c*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.

Compile

```
C:\temp\borland>bcc32 gclib.lib gclibo.lib x_simple.c
Embarcadero C++ 7.10 for Win32 Copyright (c) 1993-2015 Embarcadero Technologies, Inc.
x_simple.c:
Turbo Incremental Link 6.72 Copyright (c) 1997-2015 Embarcadero Technologies, Inc.
```

Execute

```
C:\temp\borland>x_simple.exe
version: 130.115.279
info: 192.168.0.43, DMC4143 Rev 1.2b, 9998
response:  61016.0000
:
```

x_examples.cpp

Review and Modify source

- In a text editor, open *x_examples.cpp*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.
- Find the `#if 0` preprocessor block enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.

Compile

```
C:\temp\borland>bcc32 -c  *.cpp
```

Link

```
C:\temp\borland>bcc32 -o examples.exe *.obj gclib.lib gclibo.lib
```

Execute

```
C:\temp\borland>examples.exe
Library version: 130.115.279

192.168.0.43, DMC4020 Rev 1.2b, 291


**************************************************************************
```

```
Example GRead() and GWrite() usage
************************************************************************

Read 155 QR bytes.

************************************************************************
Example GCommand() usage
************************************************************************
Revision report, ^R^V
DMC4020 Rev 1.2b
:

Command Values
val is 10
val is 11
val is 3.1415
val is 9.869

Command Trimming
> 95653016.0000
:<
> 95653016.0000<
>95653016.0000<

Receiving Binary Data
QR read 155 bytes

Error handling
QD correctly trapped, not allowed, try GArrayDownload()
DL correctly trapped, not allowed, try GProgramDownload()

Modifying timeout
Burning program...OK

************************************************************************
Example GProgramDownload() and GProgramUpload() usage
************************************************************************
GProgramDownload() correctly errored. Can't fit with level 3 compression
Program Downloaded with compression level 4
Uploading program:
#A;i=0;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i
i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;EN

Program executed as expected
************************************************************************
Example GArrayDownload() and GArrayUpload() usage
************************************************************************
2.0000, 4.0000, 6.0000, 8.0000, 10.0000, 12.0000, 14.0000, 16.0000, 18.000
0000

2.0000, 1.0000, 3.0000, 5.0000, 10.0000, 12.0000, 14.0000, 16.0000, 18.000
0000

3.0000, 5.0000, 10.0000

************************************************************************
Example GRecord() usage
************************************************************************

QR-based data record
38564
393216000

DR-based data record
38670
38772
38874
38976
39078
39180
39282
39384
39486
```

```
39588
39690

QR-based data record with offsets
39692
39692

************************************************************************
Example GMessage() usage
************************************************************************
 0.0000
 1.0000
 2.0000
 3.0000
 4.0000
 5.0000
 6.0000
 7.0000
 8.0000
 9.0000

************************************************************************
Example GInterrupt() usage
************************************************************************
"UI 8" executed.

************************************************************************
Example GMotionComplete() usage
************************************************************************

Position: 0, 0
Beginning independent motion... Motion Complete on A
Position: 8000, 0

Position: 0, 0
Beginning vector motion... Motion Complete on vector plane S
Position: 6000, 0


examples.cpp executed OK
main() is finished. Press Enter to exit:
```

1.1.1.7 gcc (Linux)

The following instructions were performed on

```
$ uname -a
Linux localhost.localdomain 3.17.4-301.fc21.x86_64 #1 SMP Thu Nov 27 19:09:10 UTC 2014 x86_64 x86_64 x86_64 GN
$ g++ --version
g++ (GCC) 4.9.2 20150212 (Red Hat 4.9.2-6)
```

Copy Files

```
$ cp -r /usr/share/gclib/doc/examples/cpp/x_examples .
$ cd x_examples
$ ls
x_arrays.cpp    x_gcommand.cpp    x_gmotioncomplete.cpp  x_programs.cpp
x_examples.cpp  x_ginterrupt.cpp  x_gread_gwrite.cpp     x_simple.c
x_examples.h    x_gmessage.cpp    x_grecord.cpp
```

x_simple.c

- In a text editor, open *x_simple.c*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.

Compile

```
$ gcc -Wall -Werror x_simple.c -lgclib -lgclibo -o simple
```

Run

```
$ ./simple
rc: 0
version: 85.60.131
rc: 0
rc: 0
info: 10.1.3.17, DMC4020 Rev 1.2b, 291
rc: 0
response:  179340166.0000
:
```

x_examples.cpp

- In a text editor, open *x_examples.cpp*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options. Don't forget `-s ALL` if data records, interrupts, and messages are to be tested.
- Find the `#if 0` preprocessor block enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.

Compile

```
$ g++ x_*.cpp -lgclib -lgclibo -o example
```

Run

$./example Library version: 85.60.131

```
10.1.3.17, DMC4020 Rev 1.2b, 291

************************************************************************
Example GRead() and GWrite() usage
************************************************************************

Read 155 QR bytes.

************************************************************************
Example GCommand() usage
************************************************************************
Revision report, ^R^V
DMC4020 Rev 1.2b
:

Command Values
val is 10
val is 11
val is 3.1415
val is 9.869

Command Trimming
> 179798738.0000
:<
> 179798738.0000<
>179798738.0000<

Receiving Binary Data
QR read 155 bytes
```

```
Error handling
QD correctly trapped, not allowed, try GArrayDownload()
DL correctly trapped, not allowed, try GProgramDownload()

Modifying timeout
Burning program...OK

************************************************************************
Example GProgramDownload() and GProgramUpload() usage
************************************************************************
GProgramDownload() correctly errored. Can't fit with level 3 compression
Program Downloaded with compression level 4
Uploading program:
#A;i=0;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1
i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;EN

Program executed as expected
************************************************************************
Example GArrayDownload(), GArrayUploadFile()
GArrayDownloadFile(), and GArrayUpload usage
************************************************************************
2.0000, 4.0000, 6.0000, 8.0000, 10.0000, 12.0000, 14.0000, 16.0000, 18.0000, 20.0000

2.0000, 1.0000, 3.0000, 5.0000, 10.0000, 12.0000, 14.0000, 16.0000, 18.0000, 20.0000

3.0000, 5.0000, 10.0000
2.0000, 1.0000, 3.0000, 5.0000, 10.0000, 12.0000, 14.0000, 16.0000, 18.0000, 20.0000


************************************************************************
Example GRecord() usage
************************************************************************

QR-based data record
36100
6000

DR-based data record
36204
36306
36408
36510
36612
36714
36816
36918
37020
37122
37224

QR-based data record with offsets
37224
37224

************************************************************************
Example GMessage() usage
************************************************************************
 0.0000
 1.0000
 2.0000
 3.0000
 4.0000
 5.0000
 6.0000
 7.0000
 8.0000
 9.0000

************************************************************************
Example GInterrupt() usage
************************************************************************
"UI 8" executed.
```

```
************************************************************************
Example GMotionComplete() usage
************************************************************************

Position: 0, 0
Beginning independent motion... Motion Complete on A
Position: 8000, 0

Position: 0, 0
Beginning vector motion... Motion Complete on vector plane S
Position: 6000, 0


examples.cpp executed OK
main() is finished. Press Enter to exit:
```

1.1.1.8 clang (OS X)

The following instructions were performed on

```
$ sw_vers
ProductName:  Mac OS X
ProductVersion: 10.10.5
BuildVersion: 14F27
$ gcc --version
Configured with: --prefix=/Library/Developer/CommandLineTools/usr --with-gxx-include-dir=/usr/include/c++/4.2.
Apple LLVM version 6.1.0 (clang-602.0.53) (based on LLVM 3.6.0svn)
Target: x86_64-apple-darwin14.5.0
Thread model: posix
```

Copy Files

```
$ cd ~
$ mkdir test
$ cd test
$ tar -xzf /Applications/gclib/examples/gclib_examples.tar.gz
$ cp /Applications/gclib/include/* .
$ cp /Applications/gclib/dylib/* .
$ ls
gclib.0.dylib   x_arrays.cpp    x_gmotioncomplete.cpp
gclib.h     x_examples.cpp    x_gread_gwrite.cpp
gclib_errors.h    x_examples.h    x_grecord.cpp
gclib_record.h    x_gcommand.cpp    x_nonblocking.cpp
gclibo.0.dylib    x_ginterrupt.cpp  x_programs.cpp
gclibo.h    x_gmessage.cpp    x_simple.c
```

x_simple.c

- In a text editor, open *x_simple.c*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options.

Compile

```
$ gcc -Wall -Werror x_simple.c gclib.0.dylib gclibo.0.dylib -o simple
```

Run

```
$ ./simple
rc: 0
version: 126.108.229
rc: 0
```

```
rc: 0
info: 10.1.3.142, DMC4020 Rev 1.2a-BH, 291
rc: 0
response:  206676.0000
:
```

x_examples.cpp

- In a text editor, open *x_examples.cpp*. Find the GOpen() call and update the address to match the desired hardware. See the documentation for GOpen() for address formatting options. Don't forget `-s ALL` if data records, interrupts, and messages are to be tested.
- Find the `#if 0` preprocessor block enclosing the example calls. Change to `#if 1` to run the examples. Comment out the function calls to be avoided. Note some calls attempt to move motors and not all functions are compatible with all Galil products.

Compile

```
$ g++ x_*.cpp gclib.0.dylib gclibo.0.dylib -o example
```

Run

```
$ ./example
Library version: 126.108.229

10.1.3.142, DMC4020 Rev 1.2a-BH, 291

************************************************************************
Example GRead() and GWrite() usage
************************************************************************

Read 1 byte(s)
:
Program test OK.

************************************************************************
Example GCommand() usage
************************************************************************
Revision report, ^R^V
DMC4020 Rev 1.2a-BH
:

Command Values
val is 10
val is 11
val is 3.1415
val is 9.869

Command Trimming
> 408978.0000
:<
> 408978.0000<
>408978.0000<

Receiving Binary Data
QR read 155 bytes

Error handling
QD correctly trapped, not allowed, try GArrayDownload()
DL correctly trapped, not allowed, try GProgramDownload()

Modifying timeout
Burning program...OK

************************************************************************
Example GProgramDownload() and GProgramUpload() usage
************************************************************************
```

```
GProgramDownload() correctly errored. Can't fit with level 3 compression
Program Downloaded with compression level 4
Uploading program:
#A;i=0;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1
i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;EN

Program executed as expected
************************************************************************
Example GArrayDownload(), GArrayUploadFile()
GArrayDownloadFile(), and GArrayUpload usage
************************************************************************
2.0000, 4.0000, 6.0000, 8.0000, 10.0000, 12.0000, 14.0000, 16.0000, 18.0000, 20.0000

2.0000, 1.0000, 3.0000, 5.0000, 10.0000, 12.0000, 14.0000, 16.0000, 18.0000, 20.0000

3.0000, 5.0000, 10.0000
2.0000, 1.0000, 3.0000, 5.0000, 10.0000, 12.0000, 14.0000, 16.0000, 18.0000, 20.0000


************************************************************************
Example GRecord() usage
************************************************************************

QR-based data record
18358
0

DR-based data record
18462
18564
18666
18768
18870
18972
19074
19176
19278
19380
19482

QR-based data record with offsets
19482
19482

************************************************************************
Example GMessage() usage
************************************************************************
 0.0000
 1.0000
 2.0000
 3.0000
 4.0000
 5.0000
 6.0000
 7.0000
 8.0000
 9.0000

************************************************************************
Example GInterrupt() usage
************************************************************************
"UI 8" executed.

************************************************************************
Example GMotionComplete() usage
************************************************************************

Position: 0, 0
Beginning independent motion... Motion Complete on A
Position: 8000, 0

Position: 0, 0
Beginning vector motion... Motion Complete on vector plane S
```

```
Position: 6000, 0

************************************************************************
Example GMessage non-blocking usage
************************************************************************
 422902.0000


************************************************************************
Example GInterrupt non-blocking usage
************************************************************************
 F1

************************************************************************
Example GRecord non-blocking usage
************************************************************************
 33786


examples.cpp executed OK
main() is finished. Press Enter to exit:
```

1.1.2 Python

The gclib Python wrapper assumes the default gclib installation location.

Install Python

- See https://www.python.org/ if Python is not already installed on the system. The gclib Python wrapper supports Python versions 2 and 3.
- On Windows, choose to add Python to the environment variable during installation. This allows Python to be invoked from the command line.

Install the gclib Python module

Windows

- Type the following commands into a command prompt.

```
>cd %temp%
>mkdir py
>cd py
>copy "c:\Program Files (x86)\Galil\gclib\source\wrappers\python\*" .
c:\Program Files (x86)\Galil\gclib\source\wrappers\python\gclib.py
c:\Program Files (x86)\Galil\gclib\source\wrappers\python\setup.py
    2 file(s) copied.
>copy "c:\Program Files (x86)\Galil\gclib\examples\python\*" .
c:\Program Files (x86)\Galil\gclib\examples\python\example.py
    1 file(s) copied.
>python setup.py install
running install
running build
running build_py
creating build
creating build\lib
copying gclib.py -> build\lib
running install_lib
running install_egg_info
Removing C:\Users\user\AppData\Local\Programs\Python\Python37-32\Lib\site-packages\gclib-1.0-py3.7.egg-info
Writing C:\Users\user\AppData\Local\Programs\Python\Python37-32\Lib\site-packages\gclib-1.0-py3.7.egg-info
```

- The gclib Python wrapper is now installed. Go to the next section, **Using gclib from the Python Interpreter**.

Linux

- Type the following commands into a terminal prompt.

```
$ mkdir py
$ cd py
$ cp /usr/share/gclib/src/wrappers/python/* .
$ cp /usr/share/gclib/doc/examples/python/* .
$ sudo python setup.py install
[sudo] password for user:
running install
running build
running build_py
creating build
creating build/lib
copying gclib.py -> build/lib
running install_lib
copying build/lib/gclib.py -> /usr/lib/python2.7/site-packages
byte-compiling /usr/lib/python2.7/site-packages/gclib.py to gclib.pyc
running install_egg_info
Writing /usr/lib/python2.7/site-packages/gclib-1.0-py2.7.egg-info
```

- The gclib Python wrapper is now installed. Go to the next section, **Using gclib from the Python Interpreter**.

OS X

- Be sure that the *Create Environment Variable* step has been followed in the OS X installation instructions.
- Type the following commands into a Terminal prompt.

```
$ mkdir ~/python_temp
$ cd ~/python_temp/
$ tar -xvf /Applications/gclib/source/gclib_python.tar.gz
x gclib.py
x setup.py
$ tar -xvf /Applications/gclib/examples/gclib_python_examples.tar.gz
x example.py
$ sudo python setup.py install
running install
running build
running build_py
creating build
creating build/lib
copying gclib.py -> build/lib
running install_lib
copying build/lib/gclib.py -> /Library/Python/2.7/site-packages
byte-compiling /Library/Python/2.7/site-packages/gclib.py to gclib.pyc
running install_egg_info
Writing /Library/Python/2.7/site-packages/gclib-1.0-py2.7.egg-info
```

- The gclib Python wrapper is now installed. Go to the next section, **Using gclib from the Python Interpreter**.

Using gclib from the Python Interpreter

- Invoke the Python Interpreter.
- Type the following into the Python prompt.

```
>>> import gclib
>>> g = gclib.py()
>>> g.GOpen('192.168.0.42')
>>> print(g.GInfo())
192.168.0.42, DMC4080 Rev 1.2c, 783
```

Running Python scripts

- Navigate the terminal to the location from **Install the gclib Python module** where example.py was copied.
- Open *example.py* in a text editor.
- Set the address in the g.GOpen() call to match an avaiable connection.
- Execute the following command at the Terminal.

```
$ python example.py
gclib version: py.127.110.250
192.168.0.42, DMC4080 Rev 1.2c, 783
```

- Experiment with the example by uncommenting sections, between the triple quotes, '''.

```
$ python example.py
gclib version: py.127.110.250
192.168.0.42, DMC4080 Rev 1.2c, 783
GProgramDownload() correctly errored. Can't fit with level 3 compression
Uploaded program:
#A;i=0;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1
i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;EN
Downloaded program verified
Array element verified
 187942.0000

Starting move...
done.
```

Getting help

```
>>> help(g.GOpen)
Help on method GOpen in module gclib:

GOpen(address) method of gclib.py instance
    Opens a connection a galil controller.
    See the gclib docs for address string formatting.

>>> help(g.GCommand)
Help on method GCommand in module gclib:

GCommand(command) method of gclib.py instance
    Performs a command-and-response transaction on the connection.
    Trims the response.

>>> 'for a full listing, try help(g)'
```

1.1.3 .Net

- VB.NET
- C#.NET

1.1.3.1 VB.NET

gclib ships with *gclib.vb*, a Visual Basic class which exposes the functionality of the gclib. In addition, a VB forms example is included which demonstrates how to use *gclib.vb*. The following instructions were performed on Visual Studio Professional 2013 and can be extended to other Visual Studio versions.

Running the included Visual Basic Example

For brevity, these instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib**.

Copy files

- Navigate to a convenient, empty, writable location, e.g. *C:\temp*.
- Copy the contents of *C:\Program Files (x86)\Galil\gclib\examples\vb\2013_12.0\gclib_example* to this location.

Open in Microsoft Visual Studio 2013

- Open *gclib_example.sln* in Visual Studio. This demo was tested on MSVS 2013.

Add existing item, *gclib.vb*

- In the *Solution Explorer*, right-click on *gclib_example* and choose *Add->Existing Item...*
- Choose *C:\Program Files (x86)\Galil\gclib\source\wrappers\vb\gclib.vb*

Run Demo

- Type *F5* to run the program.
- Type a valid GOpen() address in the text box and click Go.

Create Project from scratch with MSVC 2013

For brevity, these instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib**.

Configure Project

- Launch Visual Studio 2013
- Choose File->New->Project
- In the *New Project* dialog, choose Visual Basic -> Windows Forms Application
- Type *gclib_example* for the Name
- Choose a Location, e.g. C:\Users\user\Desktop
- Check *Create directory for solution*
- Click OK, the project will configure itself
- In the *Solution Explorer*, right click on *Solution 'gclib_example' (1 project)* and choose *Configuration Manager...*
  - In the *gclib_example* project row, click in the *Platform* column and choose <New...>
    - Choose *x86* from *Type or select the new platform:*
    - Choose *Any CPU* from *Copy settings from:*
    - Check *Create new solutions platform*
    - Click OK.
  - If x64 support is also desired, repeat the <New...> procedure for *x64*
  - In the *Active solution platform* combobox at the top of the *Configuration Manager* dialog, choose <Edit...>
    - Select *Any CPU* and click the *Remove* button
    - Click *Close*
  - Close the *Configuration Manager* dialog

- In the *Solution Explorer*, right-click on *gclib_example* and choose Add->Existing Item
  - Navigate to the installation location C:\Program Files (x86)\Galil\gclib\source\wrappers\vb
  - Choose *gclib.vb*
- In the *Solution Explorer* double-click on *gclib.vb*
  - Note that there is a preprocessor definition starting with `#if PLATFORM = "x86" Then` and `#ElseIf PLATFORM = "x64" Then`
  - Note that these sections of code enable/disable with the choice of the *Solution Platform* x86/x64, usually found in the Visual Studio toolbar
  - If a non-default gclib installation location is used, the paths in these sections of code must be updated to reflect the dll locations

Add some simple code

- In the *Solution Explorer* right-click on *Form1.vb* and choose *View Code*
- Replace the text in *Form1.vb* with the following code

```
Public Class Form1
    Dim gclib As New Gclib()
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Me.Text = "gclib simple example"
        Dim tb As New TextBox
        With tb
            .Multiline = True
            .Dock = DockStyle.Fill
            .Parent = Me
            Try
                'calls to gclib should be in a try-catch
                .AppendText("GVersion: " & gclib.GVersion() & vbCrLf)
                gclib.GOpen("192.168.0.42") 'Set an appropriate IP address here
                .AppendText("GInfo: " & gclib.GInfo() & vbCrLf)
                .AppendText("GCommand: " & gclib.GCommand("MG TIME") & vbCrLf)
            Catch ex As Exception
                .AppendText("ERROR: " & ex.Message)
      Finally
        gclib.GClose() ' Don't forget to close!
            End Try

        End With
    End Sub
End Class
```

- In the gclib.GOpen() call, indicate a correct IP address for the controller that is used for this project
- Hit *F5* to run the project

1.1.3.2 C#.NET

gclib ships with *gclib.cs*, a C# class which exposes the functionality of the gclib. In addition, a C# forms example is included which demonstrates how to use *gclib.cs*.

For brevity, these instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib**.

Running the C# Example

Copy files

- Navigate to a convenient, empty, writable location, e.g. *C:\temp*.
- Copy the contents of *C:\Program Files (x86)\Galil\gclib\examples\cs\2013_12.0\gclib_example* to this location.

Open in Microsoft Visual Studio 2013

- Open *gclib_example.sln* in Visual Studio. This demo was tested on MSVS 2013.

Add existing item, *gclib.cs*

- In the *Solution Explorer*, right-click on *gclib_example* and choose *Add->Existing Item...*
- Choose *C:\Program Files (x86)\Galil\gclib\source\wrappers\cs\gclib.cs*

Run Demo

- Type *F5* to run the program.
- Type a valid GOpen() address in the text box and click Go.

Create Project from scratch with MSVC 2013

For brevity, these instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib**.

Configure Project

- Launch Visual Studio 2013
- Choose File->New->Project
- In the *New Project* dialog, choose Visual C# -> Windows Forms Application
- Type *gclib_example* for the Name
- Choose a Location, e.g. C:\Users\user\Desktop
- Check *Create directory for solution*
- Click OK, the project will configure itself
- In the *Solution Explorer*, right click on *Solution 'gclib_example' (1 project)* and choose *Configuration Manager...*
  - In the *gclib_example* project row, click in the *Platform* column and choose <New...>
    - Choose *x86* from *Type or select the new platform:*
    - Choose *Any CPU* from *Copy settings from:*
    - Check *Create new solutions platform*
    - Click OK.
  - If x64 support is also desired, repeat the <New...> procedure for *x64*
  - In the *Active solution platform* combobox at the top of the *Configuration Manager* dialog, choose <Edit...>
    - Select *Any CPU* and click the *Remove* button
    - Click *Close*
  - Close the *Configuration Manager* dialog
- In the *Solution Explorer*, right-click on *gclib_example* and choose *Properties*
  - Choose the *Build* item on the left
    - In the *Configuration:* combobox, choose *All Configurations*
    - Choose *x86* from the *Platform* combobox
    - In *Conditional compilation symbols* type *x86*

  - If x64 is to be used also, add an *x64* token as well to the *x64 Platform*
  - Save and close the *Properties* window
- In the *Solution Explorer*, right-click on *gclib_example* and choose Add->Existing Item
  - Navigate to the installation location C:\Program Files (x86)\Galil\gclib\source\wrappers\cs
  - Choose *gclib.cs*
- In the *Solution Explorer* double-click on *gclib.cs*
  - Note that there is a preprocessor definition starting with `#if x86` and `#elif x64`
  - Note that these sections of code enable/disable with the choice of the *Solution Platform* x86/x64, usually found in the Visual Studio toolbar
  - If a non-default gclib installation location is used, the paths in these sections of code must be updated to reflect the dll locations

Add some simple code

- In the *Solution Explorer* right-click on *Form1.cs* and choose *View Code*
- Replace the text in *Form1.vb* with the following code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace gclib_example
{
    public partial class Form1 : Form
    {
        gclib gclib = new gclib();
        public Form1()
        {

            InitializeComponent();
            this.Text = "gclib simple example";
            TextBox tb = new TextBox();
            tb.Multiline = true;
            tb.Dock = DockStyle.Fill;
            tb.Parent = this;
            try
            {
                //calls to gclib should be in a try-catch
                tb.AppendText("GVersion: " + gclib.GVersion() + "\n");
                gclib.GOpen("192.168.0.42"); //Set an appropriate IP address here
                tb.AppendText("GInfo: " + gclib.GInfo() + "\n");
                tb.AppendText("GCommand: " + gclib.GCommand("MG TIME") + "\n");

            }
            catch(Exception ex)
            {
                tb.AppendText("ERROR: " + ex.Message);

            }
            finally
            {
                gclib.GClose(); //Don't forget to close!
            }

        }
    }
}
```

- In the gclib.GOpen() call, indicate a correct IP address for the controller that is used for this project
- Hit *F5* to run the project

1.1.4 Java

gclib uses the venerable `Java Native Access (JNA)` library to simplify integration into the Java Native Interface (JNI).

Attention

This is the initial version of the the gclib Java wrapper. As such, GclibJava ships as source files, not the compiled jar files. All functions are subject to change in future releases of gclib. Feel free to email `support@galil.com` with recommendations on how to make this library better.

Windows

The following instructions were performed with 64 bit Windows 7 on `Oracle NetBeans IDE 8.2 and Java 1.8.0_131`.

For brevity, these instructions assume the default gclib installation location of "C:\Program Files (x86)\Galil\gclib".

Step-by-Step

1. Install `gclib` with 64 bit binaries (default install).
2. Install 64 bit NetBeans and Java, `jdk-8u131-nb-8_2-windows-x64.exe`.
3. Launch NetBeans.
4. Create a new application.
   (a) File | New Project...
   (b) Under *Categories*, select *Java*.
   (c) Under *Projects*, select *Java Application*.
   (d) Click *Next*.
   (e) Type `GclibTest` for the *Project Name*.
   (f) Note the location of the *Project Folder*.
   (g) Uncheck *Create Main Class*
   (h) Click *Finish*
5. Open the *Project Folder* as noted above.
6. Open the *src* directory in the *Project Folder* location.
7. Copy the whole directory *C:\Program Files (x86)\Galil\gclib\examples\java\gclibtest* to this directory.
8. Copy the whole directory *C:\Program Files (x86)\Galil\gclib\source\wrappers\java\gclibjava* to this directory.
9. Create a directory at `c:\jna\`.
   - Another directory may be chosen. The purpose of this directory is to hold jna's *jar* binary for the Java classpath.
10. Download a copy of *jna.jar* to the new directory.
    - `https://github.com/java-native-access/jna#download`
    - This example uses *jna-4.4.0.jar*.

11. In the NetBeans *Projects* tab, expand *GclibTest*.
12. Right-click on *Libraries* and choose *Add JAR/Folder....*
13. Navigate to the *jna.jar* saved above. Click *Open* to add *jna.jar* to the classpath.
14. In the NetBeans *Projects* tab, right-click on *GclibTest* and choose *Properties*.
15. Choose the *Run* item out of the *Categories* options tree.
16. In the *Main Class* text box, type `gclibtest.GclibTest`. Click *OK*.
17. In the NetBeans *Projects* tab, expand *GclibTest* | *Source Packages* | *gclibtest*.
18. Double click *GclibTest.java*, and find the line containing `gclib.GOpen`.
19. Update the address for the desired hardware.
20. Choose *Run* | *Run Project (GclibTest)* or hit the `F6` key to run the application.
21. The appplication output will print in the NetBeans *Output* window.

Documentation

The GclibJava class has helpful documentation for developing a Java application. Use the following instructions to create the Javadoc.

1. In the NetBeans *Projects* tab, right-click *GclibTest*.
2. Choose *Generate Javadoc* to create the documentation and open it in the system's default browser.

1.1.5 LabVIEW

LabVIEW instructions are currently not available in PDF format. Please visit the web version of this document for LabVIEW instructions.

1.2 Using gclib

1.2.1 gcaps

gcaps (Galil Controller Asynchronous Proxy Server) is a communication server natively supported by gclib to multiplex Galil hardware communication features. It runs in the background on the host computer, as a service or daemon.

gclib & gcaps

gclib will attempt to use gcaps whenever GOpen() is called without the `--direct` or `-d` switch. Other than this small difference, gclib function calls through gcaps operate as if the connection was direct. The first version of gclib supporting gcaps is 299.

Other gcaps Usage

The following functions will attempt to use gcaps first to gather data. If gcaps is not found, the functions will fall back to user space calls to populate information.

gclib Function	Usage	If gcaps unavailable
GVersion()	Provide the version of gclib and gcaps (if available).	No gcaps version.
GIpRequests()	Provide a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.	Must be root.
GAssign()	Assigns an IP address over the Ethernet to a controller at a given MAC address.	Must be root.
GAddresses()	Provides a listing of all available connection addresses.	Must be root, or user must be in device group.

Because gcaps runs as a service on Windows, and as a system daemon on Linux, gcaps runs with root privileges. See *If gcaps unavailable* column in the above table when running without gcaps.

If gcaps is unavailable when these functions are run, a ~1 second delay will be incurred while gclib searches for the absent server. In order to prevent gcaps usage in these functions, comment out the symbol G_USE_GCAPS in gclibo.h and rebuild gclibo. See Rebuilding gclibo.

1.2.2 Program Preprocessor

gclib's program downloader provides a preprocessor for DMC code. The preprocessor modifies the program prior to download providing a number of language features not present in native DMC code.

The preprocessor is invoked in the following two ways.

1. With both GProgramDownload() and GProgramDownloadFile() via the `preprocessor` argument. Downloading code with null for the preprocessor argument uses defaults.
2. From within DMC code via in-band preprocessor directives.

The `preprocessor` argument

GProgramDownload() and GProgramDownloadFile() can be called with a string passed to the `preprocessor` argument. The program will be modified based on this string prior to download. See *Preprocessor Options* below for syntax.

In-band Operation

DMC code can be written with special markup to signal the preprocessor to take actions prior to download.
For example, the following program will invoke the in-band preprocessor. The specifics are described below.

```
## Author: Zaphod Beeblebrox
## Project: Total Perspective Vortex
//the above 4 hashmarks enable the preprocessor
##option "--min 4" //use a minimum of level four compression
REM REM-style comments are supported at all times
PRA=1000
BGA
AMA
EN
```

The REM Comment

Lines beginning with the string REM are removed prior to download. REM comments are always removed regardless of whether the other preprocessor options are enabled or not.

Double Hash

Most preprocessor statements begin with a double hash, ##. When proceeded by a space, the double hash acts like a REM comment.
When proceeded by a character other than space, ## is interpreted as a preprocessor directive. For example, see ##option below.

Note

Double hash lines are removed from the program only when the preprocessor is enabled with a quad hash.

Quad Hash to enable

In order to enable the in-band preprocessor, the first two lines of the DMC program must start with a double hash. This syntax of using two lines with double hashmarks is called a *quad hash*.
Content may follow the hash marks. For example, a good code writing style is to use double hash comments as a comment header showing author, project name, etc.

C-style comments

With the preprocessor enabled, C-style comments may be used with the // prefix. These comments are very similar to REM comments. The primary advantage of using this comment over REM is that // comments may occur anywhere in a line. This is helpful for line comments such as the following.

```
SIA= 1,25,25,0<4>1 //SSI 25 bits total, all single turn, no status
```

Strings containing // are not interpreted as comments.

Note

// comments are removed from the program only when the preprocessor is enabled with a quad hash.

Preprocessor Directives

Note

Directives are only followed when the preprocessor is enabled with a quad hash.

##option

The option directive allows passing switches directly to the preprocessor with the same syntax as the preprocessor argument in GProgramDownload() and GProgramDownloadFile(). The syntax of the option directive is the following.

```
##option "{preprocessor switches}"
```

For example, the following line will disable compression in the program.

```
##option "--max 0"
```

See *Preprocessor Options* below for other switches.

##include

The include directive provides a way to include the contents of another DMC file in the current program. This is useful for reusing code such as automatic subroutines, homing operations, or controller initilization routines.
The contents of the file will be inserted in place of the include line. The insertion occurs prior to code compression.
The syntax of the include directive is the following.

```
##include "{filename}"
```

For example,

```
##include "c:\galil\initialize.dmc"
##include "homing.dmc"
```

To write more portable code, use the include directive with just the file name, no absolute path. The path to find the file on the system is set depending on usage.

1. In the Galil Design Kit, specify the include path in GDK's settings with the --search or -I switch as defined below.
2. When downloading code via GProgramDownload() or GProgramDownloadFile(), use the --search or -I switch in the preprocessor argument.
3. Finally, if the file is in the executable search path, the file will be found. However, one of the previous two options is more reliable.

##gclib

Galil Design Kit uses the ##gclib directive in GDK Macros. gclib ignores this directive.

In-band Support

In addition to gclib, Galil Design Kit supports the preprocessor. Proper preprocessor usage will be colored in the Editor's syntax highlighter. If the quad hash is not present, preprocessor syntax will be colored differently to indicate improper usage.
The preprocessor is not supported in software prior to GDK/gclib. DMC code downloads using the in-band preprocessor in prior generation software (e.g. GalilTools or SmartTerm) will fail with a TC code of 61, *Duplicate or bad label*.

Preprocessor Options

Compression, --min, --max

- Default uses minimum compression needed to fit the program.
- --max *n* provides compression up to and including level *n*. Only the necessary compression will be performed up to level *n*.
- --min *n* will compress at least up to and including *n*. *n* defined as with --max.

Compression Levels, *n*

- Level 0 (mandatory)
  1. Remove lines beginning with REM.
  2. Remove trailing semicolons.
  3. Comment blank lines with '.
  4. Remove white space (space/tab) in front of # (label declarations).
  5. Remove white space after commands.
  6. Line ends changed to carriage return.
  7. Replace leading tabs with double space.
  8. Replace non-leading tabs with single space.
  9. A backslash (\) character on a line other than a preprocessor line will result in an error.
- Level 1
  1. Remove unnecessary spaces. Strings, comments ('), and no-ops (NO) are not changed.
- Level 2
  1. Remove comments (') but not no-ops (NO).
- Level 3
  1. Remove no-ops (NO).
- Level 4
  1. Break apart compound lines that are too long.
  2. Compact lines of code to maximize line usage.
  3. Use backtick to support long lines where applicable.

Code insertion, `--insert`

- Default begins at line zero and overwrites anything present.
- `--insert` *arg* invokes the insert option of the firmware's *DL* command. *arg* can be one of the following.
  1. Line number, e.g. `100`. Program insertion will occur on the line after the line specified.
  2. Variable name, e.g. `myvar`. Program insertion will occur on the line after the line equal to the value of the variable.
  3. Label callout, e.g. `#mylabel`. Program insertion will occur on the line after the label.
  4. A lone `#` symbol. Program insertion will occur on the line after the last line in the program buffer.
- Compression directives `--max` and `--min` are followed.
- All original code following the point of insertion is cleared.
- Not all products support the `--insert` operation, e.g. DMC-30010. See the `DL` command for support.

Warning

It is the user's responsibility to ensure that the code will fit in the inserted location. The preprocessor will not check line numbers when executing the `--insert` option.

Include Search Paths, `--search`, `-I`

- The `##include` directive will attempt to open its string argument directly. The open will succeed if the argument is the absolute path, or if the argument is in the executable's path, e.g. in the same directory.
- `--search` *path* allows the user to specify a directory or directories to be searched for the `include` file in case the first open fails.
  - For historical reasons, `-I` is shorthand for `--search`.
- Multiple directories may be specified with multiple `-I` directives.
- For in-band code, `-I` must be specified prior to the include.
- A common use for `-I` is to specify only the filename in the DMC source code and use the `preprocessor` argument during download to specify the path to the files. This allows the files to be moved without a change to source code.
- Search order
  1. The `##include` argument is checked first as-is.
  2. Then each `-I` argument in the `preprocessor` argument, in the order specified.
  3. Then `##option` directives in the DMC file, in the order specified.
- If the search path contains spaces, enclose the path in double quotes, escaped with a backslash. See example below.

In-band Example

```
##option "-I /code/dmc/homing"
##option "-I \"/code/dmc/other code\""
##include "auto.dmc"
//executable's directory will be checked
//then c:\code\dmc\homing
//then c:\code\dmc\other code
```

Macro Definition, `--define`, `-D`

- `--define` provides a way to substitute one token for another. This is useful for writing code that is generic until program download. Wherever the token is found in code, it is substituted by the replacement. The replacement occurs right before code compression.
- `-D` is shorthand for `--define`.
- The token should consist of a starting backslash character, followed by upper or lower case alphanumeric characters, underscores, and an ending backslash.
- The common usage for this feature is to write code with a token, and then call the program download with the `-D` switch.

In this example, an axis is defined at download time. Specifying the following for the preprocessor argument
```
--define \ax\:A
```
would cause the following code
```
SH\ax\
JG\ax\=1000
BG\ax\
```
to be downloaded as
```
SHA
JGA=1000
BGA
```
This causes the *A* axis to be addressed.

Note

The macro `\pid\` is reserved for exclusive use by GDK.

Conditional Directives, `--ifdef`, `--ifndef`

To specify a preprocessor directive should be executed only if a macro is defined, use `--ifdef`.
```
##option "--ifdef \minify\ --min 4" //maximally compress code if minify macro set
```
To specify a preprocessor directive should be executed only if a macro is NOT defined, use `--ifndef`.
```
##option "--ifndef \axis\ -D \axis\:A" //Default to axis A
```

GDK Support

- See the `preprocessor` text box in the *Editor* settings page to set the desired preprocessor setting for developing in GDK's editor.

1.2.3 Thread Safety

The Basics

- The easiest way to multithread, and/or to use multiple applications to access the same hardware, is to communicate through gcaps.
- Just leave out `-d` and `--direct` in your GOpen() address and gcaps will be used.
- Each thread, and each application, should use their own GCon handle. In the higher-level Language Support, each thread or application should manage their own gclib object. Don't pass the connection handle between threads.

The Formalism

gclib supports multi-threaded operation with the following operational definitions.

gclib is "reentrant"

Reentrant means that a given gclib function call may be invoked in multiple threads when passed distinct arguments. For example, GCommand() may be called simultaneously in different threads so long as the following arguments have unique values, indicating they point to unique memory.

- GCon `g`, the connection must be unique.

- GBufOut `buffer`, the writable buffer must be unique.
- GSize `*bytes_returned`, the writable value must be unique.

gclib is not "thread-safe"

Thread safety would imply that a given gclib function call could be invoked in multiple threads when passed *the same* arguments. This mode of operation **is not** supported by gclib. In other words, it is not safe to call GCommand() simultaneously in different threads if any mutable arguments point to the same memory.
In short, it is **not** safe to call GCommand() in multiple threads to the same physical connection.
If such operation is required, it is the user's responsibility to use a mutual exclusion (mutex) or other mechanism to protect memory.

Multi-threaded access to the same connection with gcaps

gcaps provides a multiplexing capability to Galil hardware. When using gcaps, it is therefore safe to call GCommand() in multiple threads to the *same physical connection* (though not the same GCon value). gclib can connect multiple times to the same Galil connection through gcaps. Because the GCon variable is unique, the reentrant capability of gclib can be used to communicate to the same physical connection through gcaps.

1.2.4 Galil Widgets

Note

gclib provides the communications foundation for the Galil Widgets project.Galill Widgets are a collection of .Net WinForms User Controls that provide quick development of custom graphical user interfaces (GUIs) that communicate with Galil Motion Controllers and PLCs.

Galil Widgets has been designed to support three general user needs

The software novice, or the hurried prototyper

Within minutes, a full UI can be laid out. All controls can be configured with menus and mouse clicks for an absolute minimum requirement for writing code. The quick start guide, and Microsoft Visual Studio Express is all that is needed to make a free application GUI with minimal effort.

The .Net developer, adding to pre-existing code.

In addition to the point-and-click configuration of the tools, each tool has a set of public function calls and properties which allows the C# or VB.Net user the ability to integrate the Galil Widgets into a .Net application with ease.

The power user

The entire Galil Widgets source code is available in the installation package. This allows users to tweak, extend, and add Widgets to the library with ease. The "GalilWidget" interface defines a number of function calls that new Widgets should implement to function correctly.

The following widgets are currently available

- `GWComs`: Communications to Galil hardware including event-driven handling of asynchronous traffic.
- `GWTerm`: A terminal for direct user interaction with the hardware.
- `GWPoll`: A polling tool to display important data on screen.
- `GWSettings` A tool for displaying, editing, backing up, and restoring controller parameters and mission-critical variables. Program backup and loading, and firmware upgrades are also supported.
- `GWDatRec`: A data record visualization tool. Used to display controller status through user-configurable labels, "soft LEDs", and analog sliders.

For more information, get the free Galil Widgets package

See the Galil Widgets release notes for changes.

Screen shots of an example motion controller configuration (left), and a similar RIO configuration (right)

1.2.5 Rebuilding gclibo

gclib ships with a compiled version of the open source portion, *gclibo*. However, if a source modification is desired, the following instructions will help with recompiling this portion of the library.

Windows

For brevity, these instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib** and a build type of **x86 (win32)**. The following instructions were performed on *Visual Studio Professional 2015* and can be extended to other Visual Studio versions.

Preparation

Create a working directory. A convenient, empty, writable location, e.g.

```
C:\>mkdir %homepath%\Desktop\temp
```

Note

In this documentation, a single *greater-than* character (>) will indicate a command prompt at this working directory.

Recompiling gclibo requires the source code for the open source compression library **zlib**. This can be downloaded from the zlib website: **http://zlib.net/zlib1211.zip**.
Extract the downloaded zlib source files to the working directory.
Open *VS2015 x86 Native Tools Command Prompt* and navigate to the working directory.

```
C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC>cd %homepath%\Desktop\temp
C:\Users\user\Desktop\temp>dir /b
zlib-1.2.11
```

Copy files

Set an environment variable for the base path.

```
>set base="C:\Program Files (x86)\Galil\gclib"
```

Set an environment variable for the zlib base path.

```
>set zlib="%CD%\zlib-1.2.11"
```

Copy the gclibo source files.

```
>copy %base%\source\gclibo\*.c .
C:\Program Files (x86)\Galil\gclib\source\gclibo\arrays.c
C:\Program Files (x86)\Galil\gclib\source\gclibo\gclibo.c
        2 file(s) copied.
```

Modify source

Make any necessary changes. For this example, the GInfo() function was changed from

```
GReturn GCALL GInfo(GCon g, GCStringOut info, GSize info_len)
{
  return GUtility(g, G_UTIL_INFO, info, &info_len);
}
```

to

```
GReturn GCALL GInfo(GCon g, GCStringOut info, GSize info_len)
{
  strncpy(info, "My controller", info_len);
  return G_NO_ERROR;
  //return GUtility(g, G_UTIL_INFO, info, &info_len);
}
```

Compile and copy

Compile the source code.

```
>cl -c *.c %zlib%\*.c -I %base%\include -I %zlib% -DBUILDING_GCLIB
```

Link the binaries.

```
>link /DLL *.obj %base%\lib\dynamic\x86\gclib.lib /OUT:gclibo.dll
```

Copy

Copy back to the installation location from the file explorer. This will require administrator privileges.

- Copy gclibo.lib to "C:\Program Files (x86)\Galil\gclib\lib\dynamic\x86"
- Copy gclibo.dll to "C:\Program Files (x86)\Galil\gclib\dll\x86"

Test

Copy simple example

```
>copy %base%\examples\cpp\x_simple.c .
```

Edit GOpen() call as necessary.

Compile

```
>cl x_simple.c %base%\lib\dynamic\x86\*.lib -I %base%\include
```

Set Path to DLL

```
>set PATH=%base%\dll\x86\;%PATH%
```

Execute

```
>x_simple.exe
rc: 0
version: 85.60.138
rc: 0
rc: 0
info: My controller
rc: 0
response:  355000958.0000
:
```

Linux

Copy files

```
$ cp -r /usr/share/gclib/src/gclibo .
$ cd gclibo
$ cp /usr/include/gclib*.h .
```

Modify source

Make any necessary changes. For this example, the GInfo() function was changed from

```
GReturn GCALL GInfo(GCon g, GCStringOut info, GSize info_len)
{
  return GUtility(g, G_UTIL_INFO, info, &info_len);
}
```

to

```
GReturn GCALL GInfo(GCon g, GCStringOut info, GSize info_len)
{
  strncpy(info, "My controller", info_len);
  return G_NO_ERROR;
  //return GUtility(g, G_UTIL_INFO, info, &info_len);
}
```

Make and install

```
# make install -f makefile_gclibo
# make clean -f makefile_gclibo
```

Test

Copy simple example

```
$ cp /usr/share/gclib/doc/examples/cpp/x_examples/x_simple.c .
```

Edit GOpen() call as necessary.

Compile

```
$ gcc x_simple.c -Wall -Werror -lgclib -lgclibo -o simple
```

Execute

```
$ ./simple
rc: 0
version: 85.60.131
rc: 0
rc: 0
info: My controller
rc: 0
response:  182879322.0000
:
```

OS X

Copy files

```
$ mkdir test
$ cd test
$ tar -xvf /Applications/gclib/source/gclibo_src.tar.gz x gclibo.h
x gclibo.c
x arrays.c
x makefile_gclibo
$ cp /Applications/gclib/include/* .
$ cp /Applications/gclib/dylib/gclib.0.dylib .
$ ls
arrays.c  gclib.h   gclib_record.h  gclibo.h
gclib.0.dylib gclib_errors.h  gclibo.c  makefile_gclibo
```

Modify source

Make any necessary changes. For this example, the GInfo() function was changed from

```
GReturn GCALL GInfo(GCon g, GCStringOut info, GSize info_len)
{
  return GUtility(g, G_UTIL_INFO, info, &info_len);
}
```

to

```
GReturn GCALL GInfo(GCon g, GCStringOut info, GSize info_len)
{
  strncpy(info, "My controller", info_len);
  return G_NO_ERROR;
  //return GUtility(g, G_UTIL_INFO, info, &info_len);
}
```

Make and install

```
$ make -f makefile_gclibo
Open source component, gclibo.0.dylib
  Compiling open source component.
gcc -c -Wall -Werror -fPIC -fvisibility=hidden -DBUILDING_GCLIB -DHAVE_VISIBILITY *.c
  Linking open source component into shared library.
gcc -dynamiclib -o gclibo.0.dylib *.o gclib.0.dylib
strip -u -r gclibo.0.dylib
  Cleaning up.
$ make install -f makefile_gclibo
```

```
Installing gclibo.0.dylib
cp gclibo.0.dylib /Applications/gclib/dylib
$ make clean -f makefile_gclibo
Cleaning project...
```

Test

Extract simple example

```
$ tar -xzf /Applications/gclib/examples/gclib_examples.tar.gz x_simple.c
```

Edit GOpen() call as necessary.

Compile

```
$ gcc x_simple.c -Wall -Werror gclib.0.dylib gclibo.0.dylib -o simple
```

Execute

```
$ ./simple
rc: 0
version: 127.110.253
rc: 0
rc: 0
info: My controller
rc: 0
response:  182879322.0000
:
```

1.2.6 Legacy Compatibility

- GalilTools included the GCL (GalilTools Communication Library). gclib ships with an open source wrapper implementation of the GCL.
- DMC32 OSU is intended for existing applications that used software based on the legacy DMCWIN32 library for Windows XP and earlier.

1.2.6.1 GalilTools

To provide maximum compatibility, gclib ships with an open source wrapper implementation of the GCL (GalilTools Communication Library). Users wanting to upgrade to gclib that have source built on Galil.h can use this wrapper to minimize source changes. This wrapper is also indicated for users that want the same function calls as Galil.h, but don't want the usage of QT as in galil1.dll.
This wrapper is intended for existing applications already using the library distributed with GalilTools (galil1.dll) or the previous *STL* library (galil2.dll). New applications should be written with gclib.

Windows

Compile galil2.dll with MSVC 2013

The following instructions were performed on *Visual Studio Professional 2013* and can be extended to other Visual Studio versions. For brevity, the instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib** and a build type of **x86 (win32)**.

Launch the compiler command prompt

- Open *VS2013 x86 Native Tools Command Prompt*.
- Navigate to a convenient, writable location, e.g. *C:\temp*.

Set an environment variable for the base path

```
C:\temp>set base=C:\Program Files (x86)\Galil\gclib
```

Compile the source code

Note the quotes.

```
C:\temp>cl -c "%base%\source\wrappers\gcl\*.cpp" -I "%base%\include" -EHsc -MD
```

Link the source code

Note the quotes.

```
C:\temp>link /DLL gcl_datarecord.obj gcl_galil.obj "%base%\lib\dynamic\x86\gclib.lib" "%base%\lib\dynamic\x86\
```

The output files *galil2.dll* and *galil2.lib* can now be used in a project using the GCL.

Test

Help the loader find the right dlls.

```
C:\temp>set PATH=%PATH%;%BASE%\dll\x86
```

Link the simple example.

```
C:\temp>link gcl_simple.obj "%base%\lib\dynamic\x86\gclib.lib" "%base%\lib\dynamic\x86\gclibo.lib" galil2.lib
```

Run the example.

```
C:\temp>simple.exe
Galil2.dll wrapper, gclib 106.75.180
10.1.3.169, DMC4020 Rev 1.2c, 291
```

Linux

Copy files

```
$ cp -r /usr/share/gclib/src/wrappers/gcl .
$ cd gcl
$ ls
Galil.h             gcl_galil.cpp  gcl_simple.cpp
gcl_datarecord.cpp  gcl_galil.h    makefile
```

Make and install

```
$ make
gcl open source wrapper for gclib
  Compiling wrapper, libgalil.so.2.0
g++ -c -fPIC -std=c++11 gcl_datarecord.cpp  gcl_galil.cpp
  Linking wrapper into shared library.
g++ -shared -o libgalil.so.2.0 *.o -Wl,-soname=libgalil.so.2
strip --strip-unneeded libgalil.so.2.0
  Cleaning up.
$ sudo make install
Installing libgalil.so.2.0
install -m 755 libgalil.so.2.0 /usr/lib
install -m 644 Galil.h /usr/lib
ldconfig
ln -s /usr/lib/libgalil.so.2 /usr/lib/libgalil.so
$ make clean
Cleaning project...
```

Test

```
$ g++ gcl_simple.cpp -lgalil -lgclib -lgclibo -o simple
$ ./simple
Galil2.dll wrapper, gclib 95.71.164
10.1.3.169, DMC4020 Rev 1.2c, 291
```

1.2.6.2 DMC32 OSU

Note

gclib provides the communications foundation for the *DMC32 Operating System Upgrade (OSU)* project.

DMC32 OSU is intended for existing applications that used software based on the legacy DMCWIN32 library for Windows XP and earlier. If such an application must be upgraded to Windows 7 ‡, 8, 8.1, or 10 DMC32 OSU may be used on these O.S. upgrades.
‡ Galil's support for Windows 7 has ended. Please click here for more information.

Galil's Windows XP support statement, http://www.galil.com/about/xp-support

- For more information refer to the documentation, http://www.galil.com/sw/pub/all/doc/dmc32osu/html/i
html
- See the release notes for changes, http://www.galil.com/sw/pub/all/rn/dmc32osu.↩
html
- The installer is available for download from Galil's website, http://www.galil.com/sw/pub/win/dmc32osu/gali
_dmc32_osu_exe.html

1.3 Example Projects

Description

Welcome to gclib Example Projects. The Galil Communication Library (gclib) is a communication library for Galil motion controllers and PLCs. A number of programming languages, operating systems, and hardware platforms are supported. These in-depth examples will demonstrate how to use the basics of gclib such as connecting to the controller and issuing commands, as well as more advanced topics such as assigning a controller an IP Address and monitoring interrupts.

Projects

Example	Description
Commands Example	Demonstrates various uses of GCommand() and GUtility().
Message Example	Demonstrates how to receive messages from the controller and detect differences in Trace and crashed code.
Position Tracking Example	Puts controller into Position Tracking Mode and accepts user-entered positions.
Jog Example	Puts controller into Jog Mode and accepts user input to adjust the speed.
Vector Mode Example	Puts controller into Vector Mode and accepts a file defining vector points.
IP Assigner Example	Assigns controller an IP Adress given a serial number and a 1 byte address.
Motion Complete Example	Uses interrupts to track when the motion of controller is completed.
Record Position Example	Record user's training and saves to a text file.
Contour Example	Record user's training and plays back training through contour mode.
Remote Server Example	Advertise local gcaps server on the network.
Remote Client Example	Discover and connect to other gcaps servers on the network.

Instructions

For build instructions, please select a supported language:

- C/C++
- C#.NET
- VB.NET

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

1.3.1 Commands Example

C++	C#	Visual Basic
Example	Example	Example
Logic	Logic	Logic
Instructions	Instructions	Instructions

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

Concepts

This example demonstrates:

- How to connect to a controller via IP Address.
- How to issue basic commands.
- How to read data back from the controller.
- How to get context on errors that occur in the program.

Prerequisites

A Galil controller is required for this example.

Command Line Arguments

This example requires 1 argument:

- The IP Address of a Galil controller.

```
commands_example.exe 192.168.42.200
```

Example Output

```
`************************************************************************`
`*************************    GCmdT() example    *************************`
`************************************************************************`
GCmdT() will return a trimmed response of GCommand()
The command 'PR ?,?' will return the relative position of the A and B axes
«PR ?,? with GCommand():  0, 10000
:»
«PR ?,? with GCmdT():  0, 10000»


`************************************************************************`
`*************************    GCmdI() example    *************************`
`************************************************************************`
GCmdI() will return the value of GCommand() parsed as an int
The command 'MG _LMS' will return the available space in the vector buffer of the S plane.
MG _LMS with GCmdT():  511.0000
MG _LMS with GCmdI(): 511


`************************************************************************`
`*************************    GCmd() example    *************************`
`************************************************************************`
GCmd() will execute the given command but does not return a value.
GCmd is useful for basic operations such as beginning motion or setting speed
GCmd(g, "BG A");
GCmd(g, "SP 5000");


`************************************************************************`
`*************************    GCmdD() example    *************************`
`************************************************************************`
GCmdD() will return the value of GCommand parsed as a double
The command 'MG @AN[1]' will return the value of Analog Input 1
MG @AN[1] with GCmdD(): 9.7726


`************************************************************************`
```

```
`************************    Galil Double Format   **************************`
`*************************************************************************`
Galil Controllers expect double values to be formatted to 4 decimal places
Unformatted double value: 0.00235
Formatted double value rounded to 4 decimal places: 0.0024


`*************************************************************************`
`******************    G_UTIL_ERROR_CONTEXT example   ********************`
`*************************************************************************`
GUtility() with G_UTIL_ERROR_CONTEXT: Broken Pipe
```

1.3.2 Message Example

C++	C#	Visual Basic
Example	Example	Example
Logic	Logic	Logic
Instructions	Instructions	Instructions

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

Concepts

This example demonstrates:

- How to connect to a controller via IP Address.
- How to issue basic commands.
- How to reconstitute a full message from GMessage().
- How to detect differences in crashed DMC code and Trace.
- How to print messages.
- How to use Keep Alive to maintain connection to gcaps.

Prerequisites

A Galil controller with a motor connected at the A axis is needed for this example.

Command Line Arguments

This example requires 1 argument:

- The IP Address of a Galil controller.

```
message_example.exe 192.168.42.96
```

Example Output

```
`*************************************************************************`
Example GMessage() usage
`*************************************************************************`
<HELLO WORLD
>
Trace Line: 0 i=0
Trace Line: 1 #A
Trace Line: 2 MGi
Standard Line:  0.0000
Trace Line: 3 i=i+1
Trace Line: 4 WT100
Trace Line: 5 JP#A,i<1
Trace Line: 6 i=i/0
Crashed Code: ?6 i=i/0
```

1.3.3 Position Tracking Example

C++	C#	Visual Basic
Example	Example	Example
Logic	Logic	Logic
Instructions	Instructions	Instructions

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

Concepts

This example demonstrates:

- How to connect to a controller via IP Address.
- How to issue basic commands.
- A controller in position tracking mode.

Prerequisites

A Galil controller with a motor connected at the A axis is needed for this example.

Command Line Arguments

This example has 1 required argument and 1 optional argument:

- Required: The IP Address of a Galil controller.
- Optional: The speed of the controller in Position Tracking mode (Default 5000).

```
position_tracking_example.exe 192.168.42.96 4000
```

Example Output

```
Begin Position Tracking with speed 5000. Enter a non-number to exit.
Enter a new position:
4000
Enter a new position:
-8000
Enter a new position:
10000
Enter a new position:
```

1.3.4 Jog Example

C++	C#	Visual Basic
Example	Example	Example
Logic	Logic	Logic
Instructions	Instructions	Instructions

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

Concepts

This example demonstrates:

- How to connect to a controller via IP Address.
- How to issue basic commands.
- A controller in jogging mode.
- How to utilize keyboard input at the console.

Prerequisites

A Galil controller with a motor connected at the A axis is needed for this example.

Note

Linux users will need to install the ncurses library.

Command Line Arguments

This example requires 1 argument:

- The IP Address of a Galil controller.

```
jog_example.exe 192.168.42.96
```

Example Output

```
Enter a character on the keyboard to change the motor's speed:
<q> Quit
<a> -2000 counts/s
<s> -500  counts/s
<d> +500  counts/s
<f> +2000 counts/s
<r> Direction Reversal
Jog Speed: 0
Jog Speed: 2000
Jog Speed: 4000
Jog Speed: 6000
Jog Speed: -6000
Jog Speed: -8000
Jog Speed: -6000
```

1.3.5 Vector Mode Example

C++	C#	Visual Basic
Example	Example	Example
Logic	Logic	Logic
Instructions	Instructions	Instructions

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

Concepts

This example demonstrates:

- How to connect to a controller via IP Address.
- How to issue basic commands.
- A controller in vector mode.
- How to read and maintain the length of the vector buffer.
- How to read in a file of vector points and apply them to the controller.

Prerequisites

A Galil controller with two motors: one connected at the A axis and the other connected at the B axis.

Command Line Arguments

This example requires 2 arguments:

- The IP Address of a Galil controller.

- The path to a file containing vector commands.

```
vector_example.exe 192.168.42.92 vector_points.txt
```

1.3.6 IP Assigner Example

C++	C#	Visual Basic
Example	Example	Example
Logic	Logic	Logic
Instructions	Instructions	Instructions

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

Concepts

This example demonstrates:

- How to issue basic commands.
- How to listen on the network for Galil Controllers requesting an IP Address.
- How to assign a Galil Controller an IP Address.
- How to connect to a controller via IP Address.
- How to get information on a connected controller such as MAC Address and Serial Number.

Prerequisites

A Galil controller connected to the same network as the host computer.

Command Line Arguments

This example requires 2 arguments:

- The serial number of your controller. The value to use is the number after the prefix on the controller's serial number marking. For example, if the serial number is marked as *BV-1234*, the value to use for this argument is *1234*.
- A value between 1-254 that defines the last byte of the newly assigned IP Address. This example will assign an IP address that matches your computer's IP address, with the last byte changed. For example, if your IP address is *192.168.42.92* and *96* is specified, the controller will be assigned *192.168.42.96*. The example will ping the IP address to ensure that the IP address is not already taken.

```
ip_assigner_example.exe 1234 96
```

1.3.7 Motion Complete Example

C++	C#	Visual Basic
Example	Example	Example
Logic	Logic	Logic
Instructions	Instructions	Instructions

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

Concepts

This example demonstrates:

- How to connect to a controller via IP Address.
- How to issue basic commands.
- How to move the controller to a precise position.
- How to monitor the interrupts of the controller.

Prerequisites

A Galil controller with two motors: one connected at the A axis and the other connected at the B axis.

Command Line Arguments

This example requires 1 argument:

- The IP Address of a Galil controller.

```
motion_complete_example.exe 192.168.42.96
```

Example Output

```
`************************************************************`
Example GInterrupt() usage
`************************************************************`

Position: 0, 0, 0, 0, 0, 0, 0, 0
Beginning independent motion...
Motion Complete on A and B
Position: 8000, 10000, 0, 0, 0, 0, 0, 0
```

1.3.8 Record Position Example

C++	C#	Visual Basic
Example	Example	Example
Logic	Logic	Logic
Instructions	Instructions	Instructions

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

Concepts

This example demonstrates:

- How to connect to a controller via IP Address.
- How to issue basic commands.
- How to manage a record array (RC/RD/RA) ring buffer.
- How to record position data and save to a text file.

Prerequisites

A Galil controller with a motor connected at the A axis and B axis is needed for this example.

Command Line Arguments

This example requires 3 arguments:

- The IP Address of a Galil controller.
- The path to a file to save Axis A positional data.
- The path to a file to save Axis B positional data.

```
record_position_example.exe 192.168.42.96 axis_a.csv axis_b.csv
```

1.3.9 Contour Example

C++	C#	Visual Basic
Example	Example	Example
Logic	Logic	Logic
Instructions	Instructions	Instructions

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

Concepts

This example demonstrates:

- How to connect to a controller via IP Address.
- How to issue basic commands.
- How to record position data and save to a text file.
- How to play back recorded data using contour mode.

Prerequisites

A Galil controller with a motor connected at the A axis and the B axis is needed for this example.

Command Line Arguments

This example requires 3 arguments:

- The IP Address of a Galil controller.
- The path to a csv file to store positional data for the A axis.
- The path to a csv file to store positional data for the B axis.

```
contour_example.exe 192.168.42.200 axis_a.csv axis_b.csv
```

1.3.10 Remote Server Example

C++	C#	Visual Basic
Example	Example	Example
Logic	Logic	Logic
Instructions	Instructions	Instructions

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

Concepts

This example demonstrates:

- How to advertise your gcaps server on the network for others to discover

Prerequisites

This example works best in conjunction with the Remote Client Example running on a separate machine on the same network.

Note

Linux users will need to install the ncurses library.

1.3.11 Remote Client Example

C++	C#	Visual Basic
Example	Example	Example
Logic	Logic	Logic
Instructions	Instructions	Instructions

We are always interested in what our customers would like to see! To request any new examples or supported languages, please email support@galil.com.

Concepts

This example demonstrates:

- How to discover other gcaps servers on your local network
- How to connect to other gcaps servers
- How to list available hardware on your connected server

Prerequisites

This example works best in conjunction with the Remote Server Example running on a separate machine on the same network.

Note

Linux users will need to install the ncurses library.

Example Output

Text colored green represents user input.

```
<s> List available servers on the network
<h> List available hardware on currently connected server
<0-9> Enter numbers 0-9 to connect to a server by index
<l> Set active server back to local server
<q> Quit
```

h

```
192.168.42.28, DMC4080 Rev 1.3c, Controller, 192.168.42.1
COM1
COM3
COM4
```

s

```
Available Servers:
<0> Example Server
```

0

```
Server set to: Example Server
```

h

```
/dev/ttyS0
/dev/ttyUSB0
```

l

```
Server set to: Local
```

1.3.12 C/C++

Please choose an operating system to get detailed instructions on how to build the gclib example projects.

- Microsoft Windows
- Linux

1.3.12.1 Microsoft Windows

Copy files

- Navigate to a convenient, empty, writable location, e.g. **C:\Users\{username}\Documents\Galil\cpp_examples**.
- Copy the contents of **C:\Program Files (x86)\Galil\gclib\examples\cpp\examples** to this location.

Open Visual Studio Project

The following instructions were performed on *Visual Studio Professional 2017* and *Visual Studio Professional 2019* and can be extended to other Visual Studio versions. For brevity, the instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib** and a build type of **x64**.

- Launch *Visual Studio 2017* or *Visual Studio 2019*.
- Choose *File->Open->Project/Solution...*.
- Navigate to the examples.vcxproj file in the C:\Users\{username}\Documents\Galil\cpp_examples directory.
- Click *Open*.
- In the *Solution Explorer* right-click on the *examples* project file, choose *Properties*.
  - Click the *Configuration Manager...* button.
    - Under *Active solution platform:* choose *x64*.
    - Click *Close*.
  - Highlight *Configuration Properties* in the side bar, and set the following project properties.
    - *Configuration Properties -> Debugging -> Environment* add **PATH=C:\Program Files (x86)\Galil\gclib\dll\x64;%PATH%**
    - Click *OK*.
- Many of the examples require command line arguments to execute. To enter command line arguments in Visual Studio:
  - In the *Solution Explorer* right-click on the *examples* project file, choose *Properties*.
    - Under *Configuration Properties*, highlight *Debugging* in the side bar. Enter the appropriate arguments in the *Command Arguments* box. Refer to each example's landing page for required command line arguments.
- Ensure the *Solution Configurations* and *Solution Platforms* are set to *Debug* and *x64* respectively.
- Hit *F5* to build and run the example.

Run a Different Example

To run a different example, remove the current example from the solution and add the next example.

- In the *Solution Explorer* right-click on *commands_example.cpp*, choose *Remove*.
  - Click the *Remove* button.
- In the *Solution Explorer* right-click on the *examples* project file, choose *Add->Existing Item*.
  - Navigate to the desired example file and click *Add*.
- Hit *F5* to build and run the example.

1.3.12.2 Linux

Copy examples to a temporary directory

```
cp -r /usr/share/gclib/doc/examples/cpp .
cd cpp
```

Run Make

To build all examples:
```
make
```
To build a single example:
```
make commands.o
```
To run an example:
```
./commands_example.out
```

1.3.13 C#.NET

Open Visual Studio Project

The following instructions were performed on *Visual Studio Professional 2017* and can be extended to other Visual Studio versions. For brevity, the instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib** and a build type of **x64**.

Copy files

- Navigate to a convenient, empty, writable location, e.g. **C:\Users\{username}\Documents\Galil\cs_examples**.
- Copy the contents of **C:\Program Files (x86)\Galil\gclib\examples\cs\examples** to this location.

Configure Project

- Launch *Visual Studio 2017*.
- Choose *File->Open->Project/Solution...*.
- Navigate to the examples.sln file in the **C:\Users\{username}\Documents\Galil\cs_examples** directory.
- Click *Open*.
- In the *Solution Explorer* right-click on the *examples* project file, choose *Properties*.
  - Click the *Build* tab.
    - At the top of the window next to *Platform* choose *x64*.
    - Click *Save*.
    - Close the properties window.
  - In the *Solution Explorer* right-click on the *examples* project file, choose *Add->Existing Item*.
    - Navigate to the gclib C# wrapper at location **C:\Program Files (x86)\Galil\gclib\source\wrappers\cs** and select *gclib.cs*.
    - Click *OK*.
- Many of the examples require command line arguments to execute. To enter command line arguments in Visual Studio:
  - In the *Solution Explorer* right-click on the *examples* project file, choose *Properties*.
    - Under *Debug*, enter the appropriate arguments in the *Command line arguments* box. Refer to each example's landing page for required command line arguments.
- Ensure the *Solution Configurations* and *Solution Platforms* are set to *Debug* and *x64* respectively.
- Hit *F5* to build and run the example.

Run a Different Example

To run a different example, change the *Startup object* to the new example.

- In the *Solution Explorer* right-click on the *examples* project file, choose *Properties*.
  - Click the *Application* tab.
    - Under the *Startup object* dropdown, select a different example.
  - Click the *Debug* tab.
    - Enter the appropriate arguments in the *Command line arguments* box. Refer to each example's landing page for required command line arguments.
- Hit *F5* to build and run the example.

1.3.14 VB.NET

Open Visual Studio Project

The following instructions were performed on *Visual Studio Professional 2017* and can be extended to other Visual Studio versions. For brevity, the instructions assume the default installation location of **C:\Program Files (x86)\Galil\gclib** and a build type of **x64**.

Copy files

- Navigate to a convenient, empty, writable location, e.g. **C:\Users\{username}\Documents\Galil\vb_examples**.
- Copy the contents of **C:\Program Files (x86)\Galil\gclib\examples\vb\examples** to this location.

Configure Project

- Launch *Visual Studio 2017*.
- Choose *File->Open->Project/Solution...*.
- Navigate to the examples.sln file in the **C:\Users\{username}\Documents\Galil\vb_examples** directory.
- Click *Open*.
- In the *Solution Explorer* right-click on the *examples* project file, choose *Properties*.
  - Click the *Compile* tab.
    - At the top of the window next to *Platform* choose *x64*.
    - Close the properties window.
  - In the *Solution Explorer* right-click on the *examples* project file, choose *Add->Existing Item*.
    - Navigate to the gclib VB wrapper at location **C:\Program Files (x86)\Galil\gclib\source\wrappers\vb** and select *gclib.vb*.
    - Click *OK*.
- Many of the examples require command line arguments to execute. To enter command line arguments in Visual Studio:
  - In the *Solution Explorer* right-click on the *examples* project file, choose *Properties*.
    - Under *Debug*, enter the appropriate arguments in the *Command line Arguments* box. Refer to each example's landing page for required command line arguments.
- Ensure the *Solution Configurations* and *Solution Platforms* are set to *Debug* and *x64* respectively.
- Hit *F5* to build and run the example.

Run a Different Example

To run a different example, change the *Startup object* to the new example.

- In the *Solution Explorer* right-click on the *examples* project file, choose *Properties*.
  - Click the *Application* tab.
    - Under the *Startup object* dropdown, select a different example.
  - Click the *Debug* tab.
    - Enter the appropriate arguments in the *Command line arguments* box. Refer to each example's landing page for required command line arguments.
- Hit *F5* to build and run the example.

Chapter 2

Topic Index

2.1 Topics

Here is a list of all topics with brief descriptions:

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 5

Data Structure Index

5.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 6

File Index

6.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 7

Topic Documentation

7.1 API

7.1.1 Description

Language Support	C++	C#	VB.NET	Java	Python
gclib Functions	Yes	Yes	Yes	Yes	Yes
Data Records and Data Structures	Yes	Yes	Yes	No	No
gclib Macros	Yes	No	No	No	No
gclib Typedefs	Yes	No	No	No	No

Files

- file gclib.h
- file gclibo.h
- file gclibo.c
- file gclib_record.h
- file gclib_errors.h

Data Structures

- struct GDataRecord4000

  Data record struct for DMC-4000 controllers, including 4000, 4200, 4103, and 500x0.
- struct GDataRecord52000

  Data record struct for DMC-52000 controller. Same as DMC-4000, with bank indicator added at byte 40.
- struct GDataRecord1806

  Data record struct for DMC-1806 controller.
- struct GDataRecord2103

  Data record struct for DMC-2103 controllers.
- struct GDataRecord1802
- struct GDataRecord30000

  Data record struct for DMC-30010 controllers.
- struct GDataRecord47000_ENC

  Data record struct for RIO-471xx and RIO-472xx PLCs. Includes encoder fields.
- struct GDataRecord47300_ENC

  Data record struct for RIO-47300. Includes encoder fields.
- struct GDataRecord47300_24EX

  Data record struct for RIO-47300 with 24EX I/O daughter board.
- struct GDataRecord47162

  Data record struct for RIO-47162.

- union GDataRecord

  Data record union, containing all structs and a generic byte array accessor.

Macros

- #define GCLIB_DLL_EXPORTED
- #define GCALL __stdcall

  Specify calling convention for Windows.
- #define G_DR 1

  Value for GRecord() `method` variable for acquiring a data record via DR mode.
- #define G_QR 0

  Value for GRecord() `method` variable for acquiring a data record via QR mode.
- #define G_BOUNDS -1

  For functions that take range options, e.g. GArrayUpload(), use this value for full range.
- #define G_CR 0

  For GArrayUpload(), use this value in the delim field to delimit with carriage returns.
- #define G_COMMA 1

  For GArrayUpload(), use this value in the delim field to delimit with commas.
- #define G_PUBLISH_SERVER 1

  For GPublishServer(), use this value to publish server to local network.
- #define G_REMOVE_SERVER 0

  For GPublishServer(), use this value to remove server from local network.
- #define G_UTIL_TIMEOUT 1

  GUtility(), Access to timeout.
- #define G_UTIL_TIMEOUT_OVERRIDE 2

  GUtility(), read/write access to timeout override.
- #define G_USE_INITIAL_TIMEOUT -1

  GUtility(), for timeout override. Set `G_UTIL_TIMEOUT_OVERRIDE` to this value to use initial GOpen() timeout (`--timeout`).
- #define G_UTIL_VERSION 128

  GUtility(), get a library version string.
- #define G_UTIL_INFO 129

  GUtility(), get a connection info string.
- #define G_UTIL_SLEEP 130

  GUtility(), specify an interval to sleep.
- #define G_UTIL_ADDRESSES 131

  GUtility(), get a list of available connections.
- #define G_UTIL_IPREQUEST 132

  GUtility(), get a list of hardware requesting IPs.
- #define G_UTIL_ASSIGN 133

  GUtility(), assign IP addresses over the network.
- #define G_UTIL_DEVICE_INITIALIZE 134

  GUtility(), sends CF, CW, EO etc. to initialize the connection. Useful after RS or other reset.
- #define G_UTIL_PING 135

  GUtility(), uses ICMP ping to determine if an IP address is reachable and assigned.
- #define G_UTIL_ERROR_CONTEXT 136

  GUtility(), provides additional error context, where available.
- #define G_UTIL_GCAPS_HOST 256
- #define G_UTIL_GCAPS_VERSION 257

  GUtility(), get the version of the gcaps server.
- #define G_UTIL_GCAPS_KEEPALIVE 258

GUtility(), Deprecated 20210119. No longer functional.

- #define G_UTIL_GCAPS_ADDRESSES 259

  GUtility(), get a list of available connections from the gcaps server.
- #define G_UTIL_GCAPS_IPREQUEST 260

  GUtility(), get a list of hardware requesting IPs from the gcaps server.
- #define G_UTIL_GCAPS_ASSIGN 261

  GUtility(), assign IP addresses over the network from the gcaps server.
- #define G_UTIL_GCAPS_PING 262

  GUtility(), uses ICMP ping to determine if an IP address is reachable and assigned. Ping sent from the gcaps server.
- #define G_UTIL_GCAPS_LIST_SERVERS 263

  GUtility(), get a list of all available gcaps servers on the local network.
- #define G_UTIL_GCAPS_PUBLISH_SERVER 264

  GUtility(), make local gcaps server discoverable by other gcaps servers on the local network.
- #define G_UTIL_GCAPS_SET_SERVER 265

  GUtility(), set the new active gcaps server.
- #define G_UTIL_GCAPS_SERVER_STATUS 266

  GUtility(), get information on the local server's name and if it is published to the local network.
- #define G_UTIL_GCAPS_REMOTE_CONNECTIONS 267

  GUtility(), get a list of remote addresses connected to local server.
- #define G_UTIL_GCAPS_SERVER_INFO 268
- #define G_UTIL_GCAPS_ADDRESSES_GET_REMEMBERED 269

  GUtility(), returns true if gcaps is remembering ip assignments.
- #define G_UTIL_GCAPS_ADDRESSES_SET_REMEMBERED 270

  GUtility(), sets if gcaps should remember ip assignments.
- #define G_SMALL_BUFFER 1024

  Most reads from Galil are small. This value will easily hold most, e.g. TH, TZ, etc.
- #define G_HUGE_BUFFER 524288

  Most reads from Galil hardware are small. This value will hold the largest array or program upload/download possible.
- #define G_LINE_BUFFER 80

  For writes, via command interpreter, to the Galil.
- #define GCLIB_DLL_EXPORTED
- #define GCALL __stdcall
- #define MALLOCBUF G_HUGE_BUFFER

  Malloc used for large program and array uploads.
- #define MAXPROG MALLOCBUF

  Maximum size for a program.
- #define MAXARRAY MALLOCBUF

  Maximum size for an array table upload.
- #define POLLINGINTERVAL 100

  Interval, in milliseconds, for polling commands, e.g. GWaitForBool().
- #define G_USE_GCAPS

  Use the GCAPS server in GAddresses(), GAssign(), GIpRequests(), and GVersion(). To avoid GCAPS, comment out this line and recompile, http://galil.com/sw/pub/all/doc/gclib/html/gclibo.html.
- #define _CRT_SECURE_NO_WARNINGS
- #define GALILDATARECORDMAXLENGTH 512

  Max size for any Galil data record, equal to dual port ram size of PCI.
- #define G_NO_ERROR 0

  Return value if function succeeded.
- #define G_NO_ERROR_S "no error"
- #define G_GCLIB_ERROR -1

General library error. Indicates internal API caught an unexpected error. Contact Galil support if this error is returned, softwaresupport@galil.com.

- #define G_GCLIB_ERROR_S "gclib unexpected error"
- #define G_GCLIB_UTILITY_ERROR -2

  An invalid request value was specified to GUtility.
- #define G_GCLIB_UTILITY_ERROR_S "invalid request value or bad arguments were specified to GUtility()"
- #define G_GCLIB_UTILITY_IP_TAKEN -3

  The IP cannot be assigned because ping returned a reply.
- #define G_GCLIB_UTILITY_IP_TAKEN_S "ip address is already taken by a device on the network"
- #define G_GCLIB_NON_BLOCKING_READ_EMPTY -4

  GMessage, GInterrupt, and GRecord can be called with a zero timeout. If there wasn't data waiting in memory, this error is returned.
- #define G_GCLIB_NON_BLOCKING_READ_EMPTY_S "data was not waiting for a zero-timeout read"
- #define G_GCLIB_POLLING_FAILED -5

  GWaitForBool out of polling trials.
- #define G_GCLIB_POLLING_FAILED_S "exit condition not met in specified polling period"
- #define G_TIMEOUT -1100

  Operation timed out. Timeout is set by the –timeout option in GOpen() and can be overriden by GSetting().
- #define G_TIMEOUT_S "device timed out"
- #define G_OPEN_ERROR -1101

  Device could not be opened. E.G. Serial port or PCI device already open.
- #define G_OPEN_ERROR_S "device failed to open"
- #define G_ALREADY_OPEN -1111

  Serial or PCI file has a flock placed on it, presumably by another gclib connection.
- #define G_ALREADY_OPEN_S "Serial or PCI port already open"
- #define G_READ_ERROR -1103

  Device read failed. E.G. Socket was closed by remote host. See G_UTIL_GCAPS_KEEPALIVE.
- #define G_READ_ERROR_S "device read error"
- #define G_WRITE_ERROR -1104

  Device write failed. E.G. Socket was closed by remote host. See G_UTIL_GCAPS_KEEPALIVE.
- #define G_WRITE_ERROR_S "device write error"
- #define G_INVALID_PREPROCESSOR_OPTIONS -1204

  GProgramDownload was called with a bad preprocessor directive.
- #define G_INVALID_PREPROCESSOR_OPTIONS_S "preprocessor did not recognize options"
- #define G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND -1106

  GCommand() was called with an illegal command, e.g. ED, DL or QD.
- #define G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND_S "illegal command passed to command call"
- #define G_DATA_RECORD_ERROR -1107

  Data record error, e.g. DR attempted on serial connection.
- #define G_DATA_RECORD_ERROR_S "data record error"
- #define G_UNSUPPORTED_FUNCTION -1109

  Function cannot be called on this bus. E.G. GInterrupt() on serial.
- #define G_UNSUPPORTED_FUNCTION_S "function not supported on this communication bus"
- #define G_FIRMWARE_LOAD_NOT_SUPPORTED -1110

  Firmware is not supported on this bus, e.g. Ethernet for the DMC-21x3 series.
- #define G_FIRMWARE_LOAD_NOT_SUPPORTED_S "firmware cannot be loaded on this communication bus to this hardware"
- #define G_ARRAY_NOT_DIMENSIONED -1200

  Array operation was called on an array that was not in the controller's array table, see LA command.
- #define G_ARRAY_NOT_DIMENSIONED_S "array not dimensioned on controller or wrong size"
- #define G_CONNECTION_NOT_ESTABLISHED -1201

Function was called with no connection.

- #define G_CONNECTION_NOT_ESTABLISHED_S "connection to hardware not established"
- #define G_ILLEGAL_DATA_IN_PROGRAM -1202

  Data to download not valid, e.g. \ in data.
- #define G_ILLEGAL_DATA_IN_PROGRAM_S "illegal ASCII character in program"
- #define G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT -1203

  Program preprocessor could not compress the program within the user's constraints.
- #define G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT_S "program cannot be compressed to fit on the controller"
- #define G_BAD_RESPONSE_QUESTION_MARK -10000

  Operation received a ?, indicating controller has a TC error.
- #define G_BAD_RESPONSE_QUESTION_MARK_S "question mark returned by controller"
- #define G_BAD_VALUE_RANGE -10002

  Bad value or range, e.g. GCon g variable passed to function was bad.
- #define G_BAD_VALUE_RANGE_S "value passed to function was bad or out of range"
- #define G_BAD_FULL_MEMORY -10003

  Not enough memory for an operation, e.g. all connections allowed for a process already taken.
- #define G_BAD_FULL_MEMORY_S "operation could not complete because of a memory error"
- #define G_BAD_LOST_DATA -10004

  Lost data, e.g. GCommand() response buffer was too small for the controller's response.
- #define G_BAD_LOST_DATA_S "data was lost due to buffer or fifo limitations"
- #define G_BAD_FILE -10005

  Bad file path, bad file contents, or bad write.
- #define G_BAD_FILE_S "file was not found, contents are invalid, or write failed"
- #define G_BAD_ADDRESS -10006

  Bad address.
- #define G_BAD_ADDRESS_S "a bad address was specified in open"
- #define G_BAD_FIRMWARE_LOAD -10008

  Bad firmware upgrade.
- #define G_BAD_FIRMWARE_LOAD_S "Firmware upgrade failed"
- #define G_GCAPS_OPEN_ERROR -20000

  gcaps connection couldn't open. Server is not running or is not reachable.
- #define G_GCAPS_OPEN_ERROR_S "gcaps connection could not be opened"
- #define G_GCAPS_SUBSCRIPTION_ERROR -20002

  GMessage(), GRecord(), GInterrupt() called on a connection without –subscribe switch.
- #define G_GCAPS_SUBSCRIPTION_ERROR_S "function requires subscription not specified in GOpen()"

Typedefs

- typedef int GReturn

  Every function returns a value of type GReturn. See gclib_errors.h for possible values.
- typedef void * GCon

  Connection handle. Unique for each connection in process. Assigned a non-zero value in GOpen().
- typedef unsigned int GSize

  Size of buffers, etc.
- typedef int GOption

  Option integer for various formatting, etc.
- typedef char * GCStringOut

  C-string output from the library. Implies null-termination.
- typedef const char * GCStringIn

  C-string input to the library. Implies null-termination.
- typedef char * GBufOut

Data output from the library. No null-termination implied. Returned values may be null-terminated, see function documentation for details.

- typedef const char * GBufIn

  Data input to the library. No null-termination, function will have a GSize to indicate bytes to write .

- typedef unsigned char GStatus

  Interrupt status byte.

- typedef void * GMemory

  Pointer to untyped memory for use in GUtility().

- typedef uint8_t UB
- typedef uint16_t UW
- typedef int16_t SW
- typedef int32_t SL
- typedef uint32_t UL

Functions

- GCLIB_DLL_EXPORTED GReturn GCALL GOpen (GCStringIn address, GCon *g)

  Open a connection to a Galil Controller.

- GCLIB_DLL_EXPORTED GReturn GCALL GClose (GCon g)

  Closes a connection to a Galil Controller.

- GCLIB_DLL_EXPORTED GReturn GCALL GRead (GCon g, GBufOut buffer, GSize buffer_len, GSize *bytes_read)

  Performs a read on the connection.

- GCLIB_DLL_EXPORTED GReturn GCALL GWrite (GCon g, GBufIn buffer, GSize buffer_len)

  Performs a write on the connection.

- GCLIB_DLL_EXPORTED GReturn GCALL GCommand (GCon g, GCStringIn command, GBufOut buffer, GSize buffer_len, GSize *bytes_returned)

  Performs a command-and-response transaction on the connection.

- GCLIB_DLL_EXPORTED GReturn GCALL GProgramDownload (GCon g, GCStringIn program, GCStringIn preprocessor)

  Downloads a program to the controller's program buffer.

- GCLIB_DLL_EXPORTED GReturn GCALL GProgramUpload (GCon g, GBufOut buffer, GSize buffer_len)

  Uploads a program from the controller's program buffer.

- GCLIB_DLL_EXPORTED GReturn GCALL GArrayDownload (GCon g, const GCStringIn array_name, GOption first, GOption last, GCStringIn buffer)

  Downloads array data to a pre-dimensioned array in the controller's array table.

- GCLIB_DLL_EXPORTED GReturn GCALL GArrayUpload (GCon g, const GCStringIn array_name, GOption first, GOption last, GOption delim, GBufOut buffer, GSize buffer_len)

  Uploads array data from the controller's array table.

- GCLIB_DLL_EXPORTED GReturn GCALL GRecord (GCon g, union GDataRecord *record, GOption method)

  Provides a fresh copy of the controller's data record. Data is cast into a union, GDataRecord.

- GCLIB_DLL_EXPORTED GReturn GCALL GMessage (GCon g, GCStringOut buffer, GSize buffer_len)

  Provides access to unsolicited messages from the controller.

- GCLIB_DLL_EXPORTED GReturn GCALL GInterrupt (GCon g, GStatus *status_byte)

  Provides access to PCI and UDP interrupts from the controller.

- GCLIB_DLL_EXPORTED GReturn GCALL GFirmwareDownload (GCon g, GCStringIn filepath)

  Upgrade firmware.

- GCLIB_DLL_EXPORTED GReturn GCALL GUtility (GCon g, GOption request, GMemory memory1, GMemory memory2)

  Provides read/write access to driver settings and convenience features based on the request variable.

- GCLIB_DLL_EXPORTED void GCALL GSleep (unsigned int timeout_ms)

  Uses GUtility() and G_UTIL_SLEEP to provide a blocking sleep call which can be useful for timing-based chores.

- GCLIB_DLL_EXPORTED GReturn GCALL GVersion (GCStringOut ver, GSize ver_len)

  Uses GUtility(), G_UTIL_VERSION and G_UTIL_GCAPS_VERSION to provide the library and gcaps version numbers.
- GCLIB_DLL_EXPORTED GReturn GCALL GAddresses (GCStringOut addresses, GSize addresses_len)

  Uses GUtility(), G_UTIL_GCAPS_ADDRESSES or G_UTIL_ADDRESSES to provide a listing of all available connection addresses.
- GCLIB_DLL_EXPORTED GReturn GCALL GInfo (GCon g, GCStringOut info, GSize info_len)

  Uses GUtility() and G_UTIL_INFO to provide a useful connection string.
- GCLIB_DLL_EXPORTED GReturn GCALL GTimeout (GCon g, short timeout_ms)

  Uses GUtility() and G_UTIL_TIMEOUT_OVERRIDE to set the library timeout.
- GCLIB_DLL_EXPORTED GReturn GCALL GCmd (GCon g, GCStringIn command)

  Wrapper around GCommand for use when the return value is not desired.
- GCLIB_DLL_EXPORTED GReturn GCALL GCmdT (GCon g, GCStringIn command, GCStringOut trimmed_response, GSize response_len, GCStringOut *front)

  Wrapper around GCommand that trims the response.
- GCLIB_DLL_EXPORTED GReturn GCALL GCmdI (GCon g, GCStringIn command, int *value)

  Wrapper around GCommand that provides the return value of a command parsed into an int.
- GCLIB_DLL_EXPORTED GReturn GCALL GCmdD (GCon g, GCStringIn command, double *value)

  Wrapper around GCommand that provides the return value of a command parsed into a double.
- GCLIB_DLL_EXPORTED GReturn GCALL GWaitForBool (GCon g, GCStringIn predicate, int trials)

  Blocking call that returns when the controller evaluates the predicate as true.
- GCLIB_DLL_EXPORTED GReturn GCALL GMotionComplete (GCon g, GCStringIn axes)

  Blocking call that returns once all axes specified have completed their motion.
- GCLIB_DLL_EXPORTED GReturn GCALL GRecordRate (GCon g, double period_ms)

  Sets the asynchronous data record to a user-specified period via `DR`.
- GCLIB_DLL_EXPORTED GReturn GCALL GProgramDownloadFile (GCon g, GCStringIn file_path, GCStringIn preprocessor)

  Program download from file.
- GCLIB_DLL_EXPORTED GReturn GCALL GProgramUploadFile (GCon g, GCStringIn file_path)

  Program upload to file.
- GCLIB_DLL_EXPORTED GReturn GCALL GArrayDownloadFile (GCon g, GCStringIn file_path)

  Array download from file.
- GCLIB_DLL_EXPORTED GReturn GCALL GArrayUploadFile (GCon g, GCStringIn file_path, GCStringIn names)

  Array upload to file.
- GCLIB_DLL_EXPORTED GReturn GCALL GIpRequests (GCStringOut requests, GSize requests_len)

  Uses GUtility(), G_UTIL_GCAPS_IPREQUEST or G_UTIL_IPREQUEST to provide a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.
- GCLIB_DLL_EXPORTED GReturn GCALL GSetServer (GCStringIn server_name)

  Uses GUtility(), G_UTIL_GCAPS_SET_SERVER to set the new active server.
- GCLIB_DLL_EXPORTED GReturn GCALL GListServers (GCStringOut servers, GSize servers_len)

  Uses GUtility(), G_UTIL_GCAPS_LIST_SERVERS to provide a list of all available gcaps services on the local network.
- GCLIB_DLL_EXPORTED GReturn GCALL GPublishServer (GCStringIn name, GOption publish, GOption save)

  Uses GUtility(), G_UTIL_GCAPS_PUBLISH_SERVER to publish local gcaps server to the local network.
- GCLIB_DLL_EXPORTED GReturn GCALL GServerStatus (GCStringOut status, GSize status_len)

  Uses GUtility(), G_UTIL_GCAPS_SERVER_STATUS to get information on the local server name and if it is published to the local network.
- GCLIB_DLL_EXPORTED GReturn GCALL GRemoteConnections (GCStringOut connections, GSize connections_length)

  Uses GUtility(), G_UTIL_GCAPS_REMOTE_CONNECTIONS to get a list of remote addresses connected to the local server.

- GCLIB_DLL_EXPORTED GReturn GCALL GAssign (GCStringIn ip, GCStringIn mac)

  Uses GUtility(), G_UTIL_GCAPS_ASSIGN or G_UTIL_ASSIGN to assign an IP address over the Ethernet to a controller at a given MAC address.
- GCLIB_DLL_EXPORTED void GCALL GError (GReturn rc, GCStringOut error, GSize error_len)

  Provides a human-readable description string for return codes.
- GCLIB_DLL_EXPORTED GReturn GCALL GSetupDownloadFile (GCon g, GCStringIn file_path, GOption options, GCStringOut info, GSize info_len)

  Download a saved controller configuration from a file.

7.1.2 Macro Definition Documentation

7.1.2.1 _CRT_SECURE_NO_WARNINGS

```
#define _CRT_SECURE_NO_WARNINGS
```
Definition at line 14 of file gclibo.c.

7.1.2.2 G_ALREADY_OPEN

```
#define G_ALREADY_OPEN -1111
```
Serial or PCI file has a flock placed on it, presumably by another gclib connection.
Definition at line 37 of file gclib_errors.h.

7.1.2.3 G_ALREADY_OPEN_S

```
#define G_ALREADY_OPEN_S "Serial or PCI port already open"
```
Definition at line 38 of file gclib_errors.h.

7.1.2.4 G_ARRAY_NOT_DIMENSIONED

```
#define G_ARRAY_NOT_DIMENSIONED -1200
```
Array operation was called on an array that was not in the controller's array table, see LA command.
Definition at line 61 of file gclib_errors.h.

7.1.2.5 G_ARRAY_NOT_DIMENSIONED_S

```
#define G_ARRAY_NOT_DIMENSIONED_S "array not dimensioned on controller or wrong size"
```
Definition at line 62 of file gclib_errors.h.

7.1.2.6 G_BAD_ADDRESS

```
#define G_BAD_ADDRESS -10006
```
Bad address.
Definition at line 88 of file gclib_errors.h.

7.1.2.7 G_BAD_ADDRESS_S

```
#define G_BAD_ADDRESS_S "a bad address was specified in open"
```
Definition at line 89 of file gclib_errors.h.

7.1.2.8 G_BAD_FILE

```
#define G_BAD_FILE -10005
```
Bad file path, bad file contents, or bad write.
Definition at line 85 of file gclib_errors.h.

7.1.2.9 G_BAD_FILE_S

```
#define G_BAD_FILE_S "file was not found, contents are invalid, or write failed"
```
Definition at line 86 of file gclib_errors.h.

7.1.2.10 G_BAD_FIRMWARE_LOAD

```
#define G_BAD_FIRMWARE_LOAD -10008
```
Bad firmware upgrade.
Definition at line 91 of file gclib_errors.h.

7.1.2.11 G_BAD_FIRMWARE_LOAD_S

```
#define G_BAD_FIRMWARE_LOAD_S "Firmware upgrade failed"
```
Definition at line 92 of file gclib_errors.h.

7.1.2.12 G_BAD_FULL_MEMORY

```
#define G_BAD_FULL_MEMORY -10003
```
Not enough memory for an operation, e.g. all connections allowed for a process already taken.
Definition at line 79 of file gclib_errors.h.

7.1.2.13 G_BAD_FULL_MEMORY_S

```
#define G_BAD_FULL_MEMORY_S "operation could not complete because of a memory error"
```
Definition at line 80 of file gclib_errors.h.

7.1.2.14 G_BAD_LOST_DATA

```
#define G_BAD_LOST_DATA -10004
```
Lost data, e.g. GCommand() response buffer was too small for the controller's response.
Definition at line 82 of file gclib_errors.h.

7.1.2.15 G_BAD_LOST_DATA_S

```
#define G_BAD_LOST_DATA_S "data was lost due to buffer or fifo limitations"
```
Definition at line 83 of file gclib_errors.h.

7.1.2.16 G_BAD_RESPONSE_QUESTION_MARK

```
#define G_BAD_RESPONSE_QUESTION_MARK -10000
```
Operation received a ?, indicating controller has a TC error.
Definition at line 73 of file gclib_errors.h.

7.1.2.17 G_BAD_RESPONSE_QUESTION_MARK_S

```
#define G_BAD_RESPONSE_QUESTION_MARK_S "question mark returned by controller"
```
Definition at line 74 of file gclib_errors.h.

7.1.2.18 G_BAD_VALUE_RANGE

```
#define G_BAD_VALUE_RANGE -10002
```
Bad value or range, e.g. GCon *g* variable passed to function was bad.
Definition at line 76 of file gclib_errors.h.

7.1.2.19 G_BAD_VALUE_RANGE_S

```
#define G_BAD_VALUE_RANGE_S "value passed to function was bad or out of range"
```
Definition at line 77 of file gclib_errors.h.

7.1.2.20 G_BOUNDS

```
#define G_BOUNDS -1
```
For functions that take range options, e.g. GArrayUpload(), use this value for full range.
Definition at line 52 of file gclib.h.

7.1.2.21 G_COMMA

```
#define G_COMMA 1
```

For GArrayUpload(), use this value in the delim field to delimit with commas.
Definition at line 54 of file gclib.h.

7.1.2.22 G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND

```
#define G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND -1106
```

GCommand() was called with an illegal command, e.g. ED, DL or QD.
Definition at line 49 of file gclib_errors.h.

7.1.2.23 G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND_S

```
#define G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND_S "illegal command passed to command call"
```

Definition at line 50 of file gclib_errors.h.

7.1.2.24 G_CONNECTION_NOT_ESTABLISHED

```
#define G_CONNECTION_NOT_ESTABLISHED -1201
```

Function was called with no connection.
Definition at line 64 of file gclib_errors.h.

7.1.2.25 G_CONNECTION_NOT_ESTABLISHED_S

```
#define G_CONNECTION_NOT_ESTABLISHED_S "connection to hardware not established"
```

Definition at line 65 of file gclib_errors.h.

7.1.2.26 G_CR

```
#define G_CR 0
```

For GArrayUpload(), use this value in the delim field to delimit with carriage returns.
Definition at line 53 of file gclib.h.

7.1.2.27 G_DATA_RECORD_ERROR

```
#define G_DATA_RECORD_ERROR -1107
```

Data record error, e.g. DR attempted on serial connection.
Definition at line 52 of file gclib_errors.h.

7.1.2.28 G_DATA_RECORD_ERROR_S

```
#define G_DATA_RECORD_ERROR_S "data record error"
```

Definition at line 53 of file gclib_errors.h.

7.1.2.29 G_DR

```
#define G_DR 1
```

Value for GRecord() `method` variable for acquiring a data record via DR mode.
Definition at line 50 of file gclib.h.

7.1.2.30 G_FIRMWARE_LOAD_NOT_SUPPORTED

```
#define G_FIRMWARE_LOAD_NOT_SUPPORTED -1110
```

Firmware is not supported on this bus, e.g. Ethernet for the DMC-21x3 series.
Definition at line 58 of file gclib_errors.h.

7.1.2.31 G_FIRMWARE_LOAD_NOT_SUPPORTED_S

```
#define G_FIRMWARE_LOAD_NOT_SUPPORTED_S "firmware cannot be loaded on this communication bus
to this hardware"
```

Definition at line 59 of file gclib_errors.h.

7.1.2.32 G_GCAPS_OPEN_ERROR

```
#define G_GCAPS_OPEN_ERROR -20000
```

gcaps connection couldn't open. Server is not running or is not reachable.
Definition at line 94 of file gclib_errors.h.

7.1.2.33 G_GCAPS_OPEN_ERROR_S

```
#define G_GCAPS_OPEN_ERROR_S "gcaps connection could not be opened"
```

Definition at line 95 of file gclib_errors.h.

7.1.2.34 G_GCAPS_SUBSCRIPTION_ERROR

```
#define G_GCAPS_SUBSCRIPTION_ERROR -20002
```

GMessage(), GRecord(), GInterrupt() called on a connection without –subscribe switch.
Definition at line 97 of file gclib_errors.h.

7.1.2.35 G_GCAPS_SUBSCRIPTION_ERROR_S

```
#define G_GCAPS_SUBSCRIPTION_ERROR_S "function requires subscription not specified in GOpen()"
```

Definition at line 98 of file gclib_errors.h.

7.1.2.36 G_GCLIB_ERROR

```
#define G_GCLIB_ERROR -1
```

General library error. Indicates internal API caught an unexpected error. Contact Galil support if this error is returned, softwaresupport@galil.com.
Definition at line 16 of file gclib_errors.h.

7.1.2.37 G_GCLIB_ERROR_S

```
#define G_GCLIB_ERROR_S "gclib unexpected error"
```

Definition at line 17 of file gclib_errors.h.

7.1.2.38 G_GCLIB_NON_BLOCKING_READ_EMPTY

```
#define G_GCLIB_NON_BLOCKING_READ_EMPTY -4
```

GMessage, GInterrupt, and GRecord can be called with a zero timeout. If there wasn't data waiting in memory, this error is returned.
Definition at line 25 of file gclib_errors.h.

7.1.2.39 G_GCLIB_NON_BLOCKING_READ_EMPTY_S

```
#define G_GCLIB_NON_BLOCKING_READ_EMPTY_S "data was not waiting for a zero-timeout read"
```

Definition at line 26 of file gclib_errors.h.

7.1.2.40 G_GCLIB_POLLING_FAILED

```
#define G_GCLIB_POLLING_FAILED -5
```

GWaitForBool out of polling trials.
Definition at line 28 of file gclib_errors.h.

7.1.2.41 G_GCLIB_POLLING_FAILED_S

```
#define G_GCLIB_POLLING_FAILED_S "exit condition not met in specified polling period"
```
Definition at line 29 of file gclib_errors.h.

7.1.2.42 G_GCLIB_UTILITY_ERROR

```
#define G_GCLIB_UTILITY_ERROR -2
```
An invalid request value was specified to GUtility.
Definition at line 19 of file gclib_errors.h.

7.1.2.43 G_GCLIB_UTILITY_ERROR_S

```
#define G_GCLIB_UTILITY_ERROR_S "invalid request value or bad arguments were specified to
GUtility()"
```
Definition at line 20 of file gclib_errors.h.

7.1.2.44 G_GCLIB_UTILITY_IP_TAKEN

```
#define G_GCLIB_UTILITY_IP_TAKEN -3
```
The IP cannot be assigned because ping returned a reply.
Definition at line 22 of file gclib_errors.h.

7.1.2.45 G_GCLIB_UTILITY_IP_TAKEN_S

```
#define G_GCLIB_UTILITY_IP_TAKEN_S "ip address is already taken by a device on the network"
```
Definition at line 23 of file gclib_errors.h.

7.1.2.46 G_HUGE_BUFFER

```
#define G_HUGE_BUFFER 524288
```
Most reads from Galil hardware are small. This value will hold the largest array or program upload/download possible.
Definition at line 90 of file gclib.h.

7.1.2.47 G_ILLEGAL_DATA_IN_PROGRAM

```
#define G_ILLEGAL_DATA_IN_PROGRAM -1202
```
Data to download not valid, e.g. \ in data.
Definition at line 67 of file gclib_errors.h.

7.1.2.48 G_ILLEGAL_DATA_IN_PROGRAM_S

```
#define G_ILLEGAL_DATA_IN_PROGRAM_S "illegal ASCII character in program"
```
Definition at line 68 of file gclib_errors.h.

7.1.2.49 G_INVALID_PREPROCESSOR_OPTIONS

```
#define G_INVALID_PREPROCESSOR_OPTIONS -1204
```
GProgramDownload was called with a bad preprocessor directive.
Definition at line 46 of file gclib_errors.h.

7.1.2.50 G_INVALID_PREPROCESSOR_OPTIONS_S

```
#define G_INVALID_PREPROCESSOR_OPTIONS_S "preprocessor did not recognize options"
```
Definition at line 47 of file gclib_errors.h.

7.1.2.51 G_LINE_BUFFER

```
#define G_LINE_BUFFER 80
```

For writes, via command interpreter, to the Galil.
Definition at line 91 of file gclib.h.

7.1.2.52 G_NO_ERROR

```
#define G_NO_ERROR 0
```

Return value if function succeeded.
Definition at line 13 of file gclib_errors.h.

7.1.2.53 G_NO_ERROR_S

```
#define G_NO_ERROR_S "no error"
```

Definition at line 14 of file gclib_errors.h.

7.1.2.54 G_OPEN_ERROR

```
#define G_OPEN_ERROR -1101
```

Device could not be opened. E.G. Serial port or PCI device already open.
Definition at line 34 of file gclib_errors.h.

7.1.2.55 G_OPEN_ERROR_S

```
#define G_OPEN_ERROR_S "device failed to open"
```

Definition at line 35 of file gclib_errors.h.

7.1.2.56 G_PUBLISH_SERVER

```
#define G_PUBLISH_SERVER 1
```

For GPublishServer(), use this value to publish server to local network.
Definition at line 55 of file gclib.h.

7.1.2.57 G_QR

```
#define G_QR 0
```

Value for GRecord() `method` variable for acquiring a data record via QR mode.
Definition at line 51 of file gclib.h.

7.1.2.58 G_READ_ERROR

```
#define G_READ_ERROR -1103
```

Device read failed. E.G. Socket was closed by remote host. See G_UTIL_GCAPS_KEEPALIVE.
Definition at line 40 of file gclib_errors.h.

7.1.2.59 G_READ_ERROR_S

```
#define G_READ_ERROR_S "device read error"
```

Definition at line 41 of file gclib_errors.h.

7.1.2.60 G_REMOVE_SERVER

```
#define G_REMOVE_SERVER 0
```

For GPublishServer(), use this value to remove server from local network.
Definition at line 56 of file gclib.h.

7.1.2.61 G_SMALL_BUFFER

```
#define G_SMALL_BUFFER 1024
```

Most reads from Galil are small. This value will easily hold most, e.g. TH, TZ, etc.
Definition at line 89 of file gclib.h.

7.1.2.62 G_TIMEOUT

```
#define G_TIMEOUT -1100
```

Operation timed out. Timeout is set by the –timeout option in GOpen() and can be overriden by GSetting().
Definition at line 31 of file gclib_errors.h.

7.1.2.63 G_TIMEOUT_S

```
#define G_TIMEOUT_S "device timed out"
```

Definition at line 32 of file gclib_errors.h.

7.1.2.64 G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT

```
#define G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT -1203
```

Program preprocessor could not compress the program within the user's constraints.
Definition at line 70 of file gclib_errors.h.

7.1.2.65 G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT_S

```
#define G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT_S "program cannot be compressed to fit on the
controller"
```

Definition at line 71 of file gclib_errors.h.

7.1.2.66 G_UNSUPPORTED_FUNCTION

```
#define G_UNSUPPORTED_FUNCTION -1109
```

Function cannot be called on this bus. E.G. GInterrupt() on serial.
Definition at line 55 of file gclib_errors.h.

7.1.2.67 G_UNSUPPORTED_FUNCTION_S

```
#define G_UNSUPPORTED_FUNCTION_S "function not supported on this communication bus"
```

Definition at line 56 of file gclib_errors.h.

7.1.2.68 G_USE_GCAPS

```
#define G_USE_GCAPS
```

Use the GCAPS server in GAddresses(), GAssign(), GIpRequests(), and GVersion(). To avoid GCAPS, comment out this line and recompile, http://galil.com/sw/pub/all/doc/gclib/html/gclibo.html.
Definition at line 41 of file gclibo.h.

7.1.2.69 G_USE_INITIAL_TIMEOUT

```
#define G_USE_INITIAL_TIMEOUT -1
```

GUtility(), for timeout override. Set G_UTIL_TIMEOUT_OVERRIDE to this value to use initial GOpen() timeout (--timeout).
Definition at line 61 of file gclib.h.

7.1.2.70 G_UTIL_ADDRESSES

```
#define G_UTIL_ADDRESSES 131
```

GUtility(), get a list of available connections.
Definition at line 65 of file gclib.h.

7.1.2.71 G_UTIL_ASSIGN

```
#define G_UTIL_ASSIGN 133
```
GUtility(), assign IP addresses over the network.
Definition at line 67 of file gclib.h.

7.1.2.72 G_UTIL_DEVICE_INITIALIZE

```
#define G_UTIL_DEVICE_INITIALIZE 134
```
GUtility(), sends CF, CW, EO etc. to initialize the connection. Useful after RS or other reset.
Definition at line 68 of file gclib.h.

7.1.2.73 G_UTIL_ERROR_CONTEXT

```
#define G_UTIL_ERROR_CONTEXT 136
```
GUtility(), provides additional error context, where available.
Definition at line 70 of file gclib.h.

7.1.2.74 G_UTIL_GCAPS_ADDRESSES

```
#define G_UTIL_GCAPS_ADDRESSES 259
```
GUtility(), get a list of available connections from the gcaps server.
Definition at line 75 of file gclib.h.

7.1.2.75 G_UTIL_GCAPS_ADDRESSES_GET_REMEMBERED

```
#define G_UTIL_GCAPS_ADDRESSES_GET_REMEMBERED 269
```
GUtility(), returns true if gcaps is remembering ip assignments.
Definition at line 85 of file gclib.h.

7.1.2.76 G_UTIL_GCAPS_ADDRESSES_SET_REMEMBERED

```
#define G_UTIL_GCAPS_ADDRESSES_SET_REMEMBERED 270
```
GUtility(), sets if gcaps should remember ip assignments.
Definition at line 86 of file gclib.h.

7.1.2.77 G_UTIL_GCAPS_ASSIGN

```
#define G_UTIL_GCAPS_ASSIGN 261
```
GUtility(), assign IP addresses over the network from the gcaps server.
Definition at line 77 of file gclib.h.

7.1.2.78 G_UTIL_GCAPS_HOST

```
#define G_UTIL_GCAPS_HOST 256
```
Definition at line 72 of file gclib.h.

7.1.2.79 G_UTIL_GCAPS_IPREQUEST

```
#define G_UTIL_GCAPS_IPREQUEST 260
```
GUtility(), get a list of hardware requesting IPs from the gcaps server.
Definition at line 76 of file gclib.h.

7.1.2.80 G_UTIL_GCAPS_KEEPALIVE

```
#define G_UTIL_GCAPS_KEEPALIVE 258
```
GUtility(), Deprecated 20210119. No longer functional.
Definition at line 74 of file gclib.h.

7.1.2.81 G_UTIL_GCAPS_LIST_SERVERS

```
#define G_UTIL_GCAPS_LIST_SERVERS 263
```

GUtility(), get a list of all available gcaps servers on the local network.
Definition at line 79 of file gclib.h.

7.1.2.82 G_UTIL_GCAPS_PING

```
#define G_UTIL_GCAPS_PING 262
```

GUtility(), uses ICMP ping to determine if an IP address is reachable and assigned. Ping sent from the gcaps server.
Definition at line 78 of file gclib.h.

7.1.2.83 G_UTIL_GCAPS_PUBLISH_SERVER

```
#define G_UTIL_GCAPS_PUBLISH_SERVER 264
```

GUtility(), make local gcaps server discoverable by other gcaps servers on the local network.
Definition at line 80 of file gclib.h.

7.1.2.84 G_UTIL_GCAPS_REMOTE_CONNECTIONS

```
#define G_UTIL_GCAPS_REMOTE_CONNECTIONS 267
```

GUtility(), get a list of remote addresses connected to local server.
Definition at line 83 of file gclib.h.

7.1.2.85 G_UTIL_GCAPS_SERVER_INFO

```
#define G_UTIL_GCAPS_SERVER_INFO 268
```

Definition at line 84 of file gclib.h.

7.1.2.86 G_UTIL_GCAPS_SERVER_STATUS

```
#define G_UTIL_GCAPS_SERVER_STATUS 266
```

GUtility(), get information on the local server's name and if it is published to the local network.
Definition at line 82 of file gclib.h.

7.1.2.87 G_UTIL_GCAPS_SET_SERVER

```
#define G_UTIL_GCAPS_SET_SERVER 265
```

GUtility(), set the new active gcaps server.
Definition at line 81 of file gclib.h.

7.1.2.88 G_UTIL_GCAPS_VERSION

```
#define G_UTIL_GCAPS_VERSION 257
```

GUtility(), get the version of the gcaps server.
Definition at line 73 of file gclib.h.

7.1.2.89 G_UTIL_INFO

```
#define G_UTIL_INFO 129
```

GUtility(), get a connection info string.
Definition at line 63 of file gclib.h.

7.1.2.90 G_UTIL_IPREQUEST

```
#define G_UTIL_IPREQUEST 132
```

GUtility(), get a list of hardware requesting IPs.
Definition at line 66 of file gclib.h.

7.1.2.91 G_UTIL_PING

```
#define G_UTIL_PING 135
```

GUtility(), uses ICMP ping to determine if an IP address is reachable and assigned.
Definition at line 69 of file gclib.h.

7.1.2.92 G_UTIL_SLEEP

```
#define G_UTIL_SLEEP 130
```

GUtility(), specify an interval to sleep.
Definition at line 64 of file gclib.h.

7.1.2.93 G_UTIL_TIMEOUT

```
#define G_UTIL_TIMEOUT 1
```

GUtility(), Access to timeout.
Definition at line 59 of file gclib.h.

7.1.2.94 G_UTIL_TIMEOUT_OVERRIDE

```
#define G_UTIL_TIMEOUT_OVERRIDE 2
```

GUtility(), read/write access to timeout override.
Definition at line 60 of file gclib.h.

7.1.2.95 G_UTIL_VERSION

```
#define G_UTIL_VERSION 128
```

GUtility(), get a library version string.
Definition at line 62 of file gclib.h.

7.1.2.96 G_WRITE_ERROR

```
#define G_WRITE_ERROR -1104
```

Device write failed. E.G. Socket was closed by remote host. See G_UTIL_GCAPS_KEEPALIVE.
Definition at line 43 of file gclib_errors.h.

7.1.2.97 G_WRITE_ERROR_S

```
#define G_WRITE_ERROR_S "device write error"
```

Definition at line 44 of file gclib_errors.h.

7.1.2.98 GALILDATARECORDMAXLENGTH

```
#define GALILDATARECORDMAXLENGTH 512
```

Max size for any Galil data record, equal to dual port ram size of PCI.
Definition at line 32 of file gclib_record.h.

7.1.2.99 GCALL [1/2]

```
#define GCALL __stdcall
```

Specify calling convention for Windows.
Definition at line 38 of file gclib.h.

7.1.2.100 GCALL [2/2]

```
#define GCALL __stdcall
```

Definition at line 28 of file gclibo.h.

7.1.2.101 GCLIB_DLL_EXPORTED [1/2]

```
#define GCLIB_DLL_EXPORTED
```
Definition at line 31 of file gclib.h.

7.1.2.102 GCLIB_DLL_EXPORTED [2/2]

```
#define GCLIB_DLL_EXPORTED
```
Definition at line 22 of file gclibo.h.

7.1.2.103 MALLOCBUF

```
#define MALLOCBUF G_HUGE_BUFFER
```
Malloc used for large program and array uploads.
Definition at line 37 of file gclibo.h.

7.1.2.104 MAXARRAY

```
#define MAXARRAY MALLOCBUF
```
Maximum size for an array table upload.
Definition at line 39 of file gclibo.h.

7.1.2.105 MAXPROG

```
#define MAXPROG MALLOCBUF
```
Maximum size for a program.
Definition at line 38 of file gclibo.h.

7.1.2.106 POLLINGINTERVAL

```
#define POLLINGINTERVAL 100
```
Interval, in milliseconds, for polling commands, e.g. GWaitForBool().
Definition at line 40 of file gclibo.h.

7.1.3 Typedef Documentation

7.1.3.1 GBufIn

```
typedef const char* GBufIn
```
Data input to the library. No null-termination, function will have a GSize to indicate bytes to write .
Definition at line 100 of file gclib.h.

7.1.3.2 GBufOut

```
typedef char* GBufOut
```
Data output from the library. No null-termination implied. Returned values may be null-terminated, see function documentation for details.
Definition at line 99 of file gclib.h.

7.1.3.3 GCon

```
typedef void* GCon
```
Connection handle. Unique for each connection in process. Assigned a non-zero value in GOpen().
Definition at line 94 of file gclib.h.

7.1.3.4 GCStringIn

```
typedef const char* GCStringIn
```
C-string input to the library. Implies null-termination.
Definition at line 98 of file gclib.h.

7.1.3.5 GCStringOut

```
typedef char* GCStringOut
```

C-string output from the library. Implies null-termination.
Definition at line 97 of file gclib.h.

7.1.3.6 GMemory

```
typedef void* GMemory
```

Pointer to untyped memory for use in GUtility().
Definition at line 102 of file gclib.h.

7.1.3.7 GOption

```
typedef int GOption
```

Option integer for various formatting, etc.
Definition at line 96 of file gclib.h.

7.1.3.8 GReturn

```
typedef int GReturn
```

Every function returns a value of type GReturn. See gclib_errors.h for possible values.
Definition at line 93 of file gclib.h.

7.1.3.9 GSize

```
typedef unsigned int GSize
```

Size of buffers, etc.
Definition at line 95 of file gclib.h.

7.1.3.10 GStatus

```
typedef unsigned char GStatus
```

Interrupt status byte.
Definition at line 101 of file gclib.h.

7.1.3.11 SL

```
typedef int32_t SL
```

Definition at line 19 of file gclib_record.h.

7.1.3.12 SW

```
typedef int16_t SW
```

Definition at line 18 of file gclib_record.h.

7.1.3.13 UB

```
typedef uint8_t UB
```

Definition at line 16 of file gclib_record.h.

7.1.3.14 UL

```
typedef uint32_t UL
```

Definition at line 20 of file gclib_record.h.

7.1.3.15 UW

```
typedef uint16_t UW
```

Definition at line 17 of file gclib_record.h.

7.1.4 Function Documentation

7.1.4.1 GAddresses()

```
GReturn GCALL GAddresses (
            GCStringOut addresses,
            GSize addresses_len )
```

Uses GUtility(), G_UTIL_GCAPS_ADDRESSES or G_UTIL_ADDRESSES to provide a listing of all available connection addresses.

Note

Serial ports are listed, e.g. COM1. Upon open, it may be necessary to specify a baud rate for the controller, e.g. `--baud 19200`. Default baud is 115200. See GOpen().

Parameters

addresses	Buffer to hold the output string. Buffer will be null terminated, even if the data must be truncated to do so. See below for more information.
addresses_len	Length of buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

If gcaps is available, the listing will come from the server via G_UTIL_GCAPS_ADDRESSES. In the absence of the server, gclib will use G_UTIL_ADDRESSES to generate the list.

- Ethernet controllers will be listed as *ip_address, revision_report, network_adapter_name, network_adapter_ip_address*. If an IP address is unreachable via ping, the address will be in parentheses.
- PCI controllers will be listed by their identifier, e.g. GALILPCI1.
- Serial ports will be listed by their identifier, e.g. COM1.

```
10.1.3.91, DMC4020 Rev 1.2e, LAN, 10.1.3.10
192.168.0.63, DMC4040 Rev 1.2f, Static, 192.168.0.41
(192.0.0.42), RIO47102 Rev 1.1j, Static, 192.168.0.41
GALILPCI1
COM1
COM2
```

Note

GAddresses() will take up to 1 second to look for gcaps.

See x_examples.cpp for an example.
Definition at line 54 of file gclibo.c.
References G_NO_ERROR, G_UTIL_ADDRESSES, G_UTIL_GCAPS_ADDRESSES, GUtility(), and vector().
Referenced by remote_client().

7.1.4.2 GArrayDownload()

```
GCLIB_DLL_EXPORTED GReturn GCALL GArrayDownload (
            GCon g,
            const GCStringIn array_name,
            GOption first,
            GOption last,
            GCStringIn buffer )
```

Downloads array data to a pre-dimensioned array in the controller's array table.

Warning

The array must already exist on the controller and be sufficient dimension to hold the desired array data, e.g. via DM.

Parameters

g	Connection's handle.
array_name	Null-terminated string containing the name of the array to download. Must match the array name used in DM.
first	The first element of the array for sub-array downloads. G_BOUNDS to omit.
last	The last element of the array for sub-array downloads. G_BOUNDS to omit.
buffer	Buffer containing the null-terminated data to be sent to the controller. The array data may be separated with *carriage return*, *carriage return + line feed*, or a *comma*. No spaces.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_arrays.cpp for an example.
Referenced by H_DownloadArraysFromList(), and x_arrays().

7.1.4.3 GArrayDownloadFile()

```
GCLIB_DLL_EXPORTED GReturn GCALL GArrayDownloadFile (
            GCon g,
            GCStringIn file_path )
```

Array download from file.
Downloads a csv file containing array data at file_path. If the arrays don't exist, they will be dimensioned.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the array file.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_arrays.cpp for an example.
Definition at line 380 of file arrays.c.
References G_BAD_FILE, G_NO_ERROR, H_ArrayDownloadFromMemory(), and vector().
Referenced by x_arrays().

7.1.4.4 GArrayUpload()

```
GCLIB_DLL_EXPORTED GReturn GCALL GArrayUpload (
            GCon g,
            const GCStringIn array_name,
            GOption first,
            GOption last,
            GOption delim,
            GBufOut buffer,
            GSize buffer_len )
```

Uploads array data from the controller's array table.

Parameters

g	Connection's handle.
array_name	Null-terminated string containing the name of the array to upload.
first	The first element of the array for sub-array uploads. G_BOUNDS to omit.
last	The last element of the array for sub-array uploads. G_BOUNDS to omit.

Parameters

delim	Sets the delimeter between array elements in the returned data, `G_CR` specifies carriage return, `G_COMMA` specifies comma.
buffer	Buffer to receive the uploaded data. The data will be null terminated unless function returns `G_BAD_LOST_DATA` due to the buffer being too small to hold the data.
buffer_len	The length of the receive buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_arrays.cpp for an example.
Referenced by H_UploadArrayToList(), write_array_to_file(), and x_arrays().

7.1.4.5 GArrayUploadFile()

```
GCLIB_DLL_EXPORTED GReturn GCALL GArrayUploadFile (
            GCon g,
            GCStringIn file_path,
            GCStringIn names )
```

Array upload to file.
Uploads the entire controller array table or a subset and saves the data as a csv file specified by `file_path`.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the array file, file will be overwritten if it exists.
names	Null-terminated string containing the arrays to upload, delimited with space. "" or null uploads all arrays listed in LA.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_arrays.cpp for an example.
Definition at line 408 of file arrays.c.
References G_NO_ERROR, GCmdT(), H_FreeArrays(), H_InitArrayNode(), H_UploadArrayToList(), H_WriteArrayCsv(), and vector().
Referenced by x_arrays().

7.1.4.6 GAssign()

```
GReturn GCALL GAssign (
            GCStringIn ip,
            GCStringIn mac )
```

Uses GUtility(), G_UTIL_GCAPS_ASSIGN or G_UTIL_ASSIGN to assign an IP address over the Ethernet to a controller at a given MAC address.

Parameters

ip	The null-terminated ip address to assign. The hardware should not yet have an IP address.
mac	The null-terminated MAC address of the hardware.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

On Linux and Mac, the desired IP address will be pinged prior to the assignment. If the ping is returned, GAssign() will return G_GCLIB_UTILITY_IP_TAKEN.
If gcaps is available, the assign will be performed from the server via G_UTIL_GCAPS_ASSIGN. gcaps will remember the assignment and will automatically assign the desired IP address should the controller ever request one again, e.g. after a controller master reset. To clear the remembered IP address from gcaps, call GAssign() with a blank string in place of the ip address. To remove all remembered ip addresses, specfify a blank string for the mac address.
In the absence of the server, gclib will use G_UTIL_ASSIGN to assign. GAssign() will take up to 1 second to look for gcaps. When not using gcaps, Linux/OS X users must be root to use GAssign() and have UDP access to send on port 68.
See x_examples.cpp for an example.
Definition at line 70 of file gclibo.c.
References G_GCLIB_UTILITY_IP_TAKEN, G_NO_ERROR, G_UTIL_ASSIGN, G_UTIL_GCAPS_ASSIGN, G_UTIL_GCAPS_PING, G_UTIL_PING, GUtility(), and vector().
Referenced by ip_assigner().

7.1.4.7 GClose()

```
GCLIB_DLL_EXPORTED GReturn GCALL GClose (
            GCon g )
```

Closes a connection to a Galil Controller.

Attention

gclib requires that `GClose()` be called whenever a program is finished with a controller. This includes when a program closes. A rule of thumb is that for every `GOpen()` call on a given connection, a `GClose()` call should be found on every code path. Failing to call GClose() may cause controller resources to not be released or can hang the process if there are outstanding asynchronous operations. The latter can occur, for example, if a call to GRead() times out and the process exits without calling GClose(). In this case, GRead() still has an outstanding asynchronous read pending. GClose() will terminate this operation allowing the process to exit correctly.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_examples.cpp for an example.

7.1.4.8 GCmd()

```
GReturn GCALL GCmd (
            GCon g,
            GCStringIn command )
```

Wrapper around GCommand for use when the return value is not desired.
The returned data is still checked for error, e.g. `?` or timeout, but is not brought out through the prototype.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 237 of file gclibo.c.
References G_SMALL_BUFFER, GCommand(), and vector().
Referenced by check_interrupts(), commands(), contour(), GRecordRate(), H_DownloadArraysFromList(), H_DownloadData(), ip_assigner(), jog(), load_buf(), load_buffer(), message(), motion_complete(), position_tracking(), record_position(), vector(), x_arrays(), x_dr_motioncomplete(), x_ei_motioncomplete(), x_gcommand(), x_ginterrupt(), x_gmessage(), x_gmotioncomplete(), x_gread_gwrite(), x_grecord(), x_nonblocking(), and x_programs().

7.1.4.9 GCmdD()

```
GReturn GCALL GCmdD (
            GCon g,
            GCStringIn command,
            double * value )
```

Wrapper around GCommand that provides the return value of a command parsed into a double.
Use this function to retrieve the full Galil 4.2 range, e.g. for a variable value with fractional data, or the value of an Analog input or Output.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.
value	Pointer to a double that will be filled with the return value.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 289 of file gclibo.c.
References G_NO_ERROR, G_SMALL_BUFFER, GCommand(), and vector().
Referenced by commands(), GRecordRate(), and x_gcommand().

7.1.4.10 GCmdI()

```
GReturn GCALL GCmdI (
            GCon g,
            GCStringIn command,
            int * value )
```

Wrapper around GCommand that provides the return value of a command parsed into an int.
Use this function to get most values including TP, RP, TE, Digital I/O states, etc.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.
value	Pointer to an int that will be filled with the return value.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 278 of file gclibo.c.

References G_NO_ERROR, G_SMALL_BUFFER, GCommand(), and vector().
Referenced by commands(), contour(), record_position(), vector(), x_dr_motioncomplete(), x_gcommand(), x_gread_gwrite(), x_grecord(), and x_programs().

7.1.4.11 GCmdT()

```
GReturn GCALL GCmdT (
            GCon g,
            GCStringIn command,
            GCStringOut trimmed_response,
            GSize response_len,
            GCStringOut * front )
```

Wrapper around GCommand that trims the response.
For use when the return value is desired, is ASCII (not binary), and the response should be trimmed of trailing colon, whitespace, and optionally leading space.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.
trimmed_response	The trimmed response from the controller. Trailing space is trimmed by null terminating any trailing spaces, carriage returns, or line feeds.
response_len	The length of the trimmed_response buffer.
front	If non-null, upon return ∗front will point to the first non-space character in trimmed_response. This allows trimming the front of the string without modifying the user's buffer pointer, which may be allocated on the heap.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 243 of file gclibo.c.
References G_NO_ERROR, GCommand(), and vector().
Referenced by commands(), GArrayUploadFile(), GRecordRate(), motion_complete(), x_gcommand(), x_ginterrupt(), x_gmotioncomplete(), and x_grecord().

7.1.4.12 GCommand()

```
GCLIB_DLL_EXPORTED GReturn GCALL GCommand (
            GCon g,
            GCStringIn command,
            GBufOut buffer,
            GSize buffer_len,
            GSize * bytes_returned )
```

Performs a *command-and-response* transaction on the connection.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller. The library will append a carriage return to the command string.
buffer	Buffer for the response. Will be filled with the response from the controller. The data will be null terminated unless the function returns G_BAD_LOST_DATA due to the buffer being too small to hold the data.
buffer_len	The size of the response buffer.
bytes_returned	The size of the data returned from the controller. This does not include null termination. This argument may be null if the value is not desired.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Referenced by commands(), error(), GCmd(), GCmdD(), gclib::GCmdD(), GCmdI(), gclib::GCmdI(), GCmdT(), GWaitForBool(), motion_complete(), x_gcommand(), x_ginterrupt(), and x_grecord().

7.1.4.13 GError()

```
void GCALL GError (
            GReturn rc,
            GCStringOut error,
            GSize error_len )
```

Provides a human-readable description string for return codes.

Parameters

rc	The return code to lookup.
error	The buffer to fill with the error text. Buffer will be null terminated, even if the data must be truncated to do so.
error_len	The length of the error buffer.

See x_examples.cpp for an example.
Definition at line 459 of file gclibo.c.
References error(), G_ALREADY_OPEN, G_ARRAY_NOT_DIMENSIONED, G_BAD_ADDRESS, G_BAD_FILE, G_BAD_FIRMWARE_LOAD, G_BAD_FULL_MEMORY, G_BAD_LOST_DATA, G_BAD_RESPONSE_QUESTION_MARK, G_BAD_VALUE_RANGE, G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND, G_CONNECTION_NOT_ESTABLISHED, G_DATA_RECORD_ERROR, G_FIRMWARE_LOAD_NOT_SUPPORTED, G_GCAPS_OPEN_ERROR, G_GCAPS_SUBSCRIPTION, G_GCLIB_ERROR, G_GCLIB_NON_BLOCKING_READ_EMPTY, G_GCLIB_UTILITY_ERROR, G_GCLIB_UTILITY_IP_TAKEN, G_ILLEGAL_DATA_IN_PROGRAM, G_INVALID_PREPROCESSOR_OPTIONS, G_NO_ERROR, G_OPEN_ERROR, G_READ_ERROR, G_TIMEOUT, G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT, G_UNSUPPORTED_FUNCTION, G_WRITE_ERROR, and vector().
Referenced by error(), gclib::GArrayDownload(), gclib::GArrayDownloadFile(), gclib::GArrayUpload(), gclib::GArrayUploadFile(), gclib::GAssign(), gclib::GCommand(), gclib::GFirmwareDownload(), gclib::GListServers(), gclib::GMotionComplete(), gclib::GOpen(), gclib::GProgramDownload(), gclib::GProgramDownloadFile(), gclib::GProgramUpload(), gclib::GProgramUploadFile(), gclib::GPublishServer(), gclib::GRecord< T >(), gclib::GRecordRate(), gclib::GRemoteConnections(), gclib::GServerStatus(), gclib::GSetServer(), gclib::GSetupDownloadFile(), and gclib::GWrite().

7.1.4.14 GFirmwareDownload()

```
GCLIB_DLL_EXPORTED GReturn GCALL GFirmwareDownload (
            GCon g,
            GCStringIn filepath )
```

Upgrade firmware.

Parameters

g	Connection's handle.
filepath	The full file path to the Galil-supplied firmware hex file. See http://www.galil.com/downloads/firmware

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

```
ec(GInfo(g, buf, sizeof(buf))); //get conntroller info
cout « buf « '\n'; //print the info
ec(GFirmwareDownload(g, "F:/1806.dmc/dmc-1806-r11a.hex"));
ec(GInfo(g, buf, sizeof(buf))); //get the info again
cout « buf « '\n';
```

```
// example output:
// GALILPCI1, DMC1846 Rev 1.1a-CM, 4232
// GALILPCI1, DMC1846 Rev 1.1a, 4232
```

7.1.4.15 GInfo()

```
GReturn GCALL GInfo (
            GCon g,
            GCStringOut info,
            GSize info_len )
```

Uses GUtility() and G_UTIL_INFO to provide a useful connection string.

Parameters

g	Connection's handle.
info	Buffer to hold the output string. Buffer will be null terminated, even if the data must be truncated to do so.
info_len	Length of buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

The response is *address, revision_report, serial_number*. For example:

```
COM2, RIO47102 Rev 1.1j, 37290
```

See x_examples.cpp for an example.
Definition at line 49 of file gclibo.c.
References G_UTIL_INFO, GUtility(), and vector().
Referenced by ip_assigner().

7.1.4.16 GInterrupt()

```
GCLIB_DLL_EXPORTED GReturn GCALL GInterrupt (
            GCon g,
            GStatus * status_byte )
```

Provides access to PCI and UDP interrupts from the controller.
Interrupts can be generated automatically by the firmware on important events via `EI` (Enable Interrupt) or by the user in embedded DMC code via `UI` (User Interrupt). To use this function, `-s EI` must be used in the GOpen() address string to subscribe to interrupts.

Parameters

g	Connection's handle.
status_byte	A pointer to a GStatus to receive the status byte.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

GInterrupt() will block until an interrupt is received, or the function times out.

Note

If this function is called with a timeout of zero, a non-blocking read is performed. If interrupt data is waiting in the interrupt queue, the oldest byte will be popped off the queue. If there is no interrupt data queued, but there is data waiting in the socket or PCI FIFO, one read will be performed to process the waiting data. If new data is still not found after these two attempts, G_GCLIB_NON_BLOCKING_READ_EMPTY will be returned.

See x_ginterrupt.cpp for an example. See x_nonblocking.cpp for an example of non-blocking usage.
Referenced by check_interrupts(), motion_complete(), x_ei_motioncomplete(), x_ginterrupt(), and x_nonblocking().

7.1.4.17 GIpRequests()

```
GReturn GCALL GIpRequests (
            GCStringOut requests,
            GSize requests_len )
```

Uses GUtility(), G_UTIL_GCAPS_IPREQUEST or G_UTIL_IPREQUEST to provide a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.

Parameters

requests	The buffer to hold the list of requesting controllers. Data will be null terminated, even if the data must be truncated to do so. See below for more information.
requests_len	The length of the requests buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

GIpRequests() will block up to 5 seconds while listening for requests.
If gcaps is available, the listing will come from the server via G_UTIL_GCAPS_IPREQUEST. In the absence of the server, gclib will use G_UTIL_IPREQUEST to generate the list. GIpRequests() will take up to 1 second to look for gcaps. When not using gcaps, Linux/OS X users must be root to use GIpRequests() and have UDP access to bind and listen on port 67.
Each line of the returned data will be of the form *model, serial_number, MAC_address, network_adapter_name, network_adapter_ip_address, remembered_ip_assignment*. See GAssign() for more infomation about remembered IP assignments. The following is an example output.

```
DMC2000, 34023, 00:50:4C:00:84:E7, enp5s0, 192.168.42.92, 192.168.42.200
DMC2105, 7, 00:50:4C:58:00:07, enp5s0, 192.168.42.92, 0.0.0.0
DMC2105, 13, 00:50:4C:58:00:0D, enp5s0, 192.168.42.92, 0.0.0.0
```

See x_examples.cpp for an example.
Definition at line 106 of file gclibo.c.
References G_NO_ERROR, G_UTIL_GCAPS_IPREQUEST, G_UTIL_IPREQUEST, GSleep(), GUtility(), and vector().
Referenced by ip_assigner().

7.1.4.18 GListServers()

```
GReturn GCALL GListServers (
            GCStringOut servers,
            GSize servers_len )
```

Uses GUtility(), G_UTIL_GCAPS_LIST_SERVERS to provide a list of all available gcaps services on the local network.

Note

This function is only available on Windows 10 and Linux.

Parameters

servers	The buffer to hold the list of available gcaps servers
servers_len	The length of the servers buffer

This function is used to find a list of available gcaps servers that have made themselves "Discoverable".
The list of available servers are separated by a newline '\n' character.

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 169 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_LIST_SERVERS, GUtility(), and vector().
Referenced by remote_client().

7.1.4.19 GMessage()

```
GCLIB_DLL_EXPORTED GReturn GCALL GMessage (
            GCon g,
            GCStringOut buffer,
            GSize buffer_len )
```

Provides access to unsolicited messages from the controller.
To use this function, `-s MG` must be used in the GOpen() `address` string to subscribe to messages. Unsolicited bytes must be flagged by the high-bit setting, `CW 1`. The driver will automatically set this when subscribing to messages. The user should not overwrite this setting.
Unsolicited messages are data generated by the controller that are not in response to a command, a data record, or an interrupt. Examples follow.

1. Data generated by the `MG` command from embedded code. `MG` sent from the host is solicited.
2. Any command in an embedded program that returns data, e.g. `TP`, `RP`, `var=?`
3. A run time error in an embedded program, e.g. `?55 i=var`

Note

Messages are unframed byte streams. There is no guarantee that the user will get complete messages or single messages in a call to GMessage(). If multiple messages have been sent from the controller since the last call to `GMessage()`, they will all be placed in the buffer, separated by newline characters.

Parameters

g	Connection's handle.
buffer	The buffer to write the message data. The buffer will be null terminated.
buffer_len	The length of the user's buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

GMessage() will block until a message is received, or the function times out.

Note

If this function is called with a timeout of zero, a non-blocking read is performed. If message data has been processed since the last time the function was called, this data will be returned. If there is no processed message data, but there is data waiting in the socket or PCI FIFO, one read will be performed to process the waiting data. If new data is still not found after these two attempts, G_GCLIB_NON_BLOCKING_READ_↩ EMPTY will be returned.

Warning

When sending message streams through gcaps, the following non-printable bytes are illegal, `$00-$07` and `$10-$17`. These bytes may be routed to a third party device such as am HMI or display panel. See MG and CF.

See x_gmessage.cpp for an example. See x_nonblocking.cpp for an example of non-blocking usage.
Referenced by message(), x_gmessage(), and x_nonblocking().

7.1.4.20 GMotionComplete()

```
GReturn GCALL GMotionComplete (
            GCon g,
            GCStringIn axes )
```

Blocking call that returns once all axes specified have completed their motion.

Note

This function uses a profiled motion indicator, not the position of the encoder. E.G. see the difference between AM (profiled) and MC (encoder-based).

Although using the _BGm operand is the most generally compatible method, there are higher-performance ways to check for motion complete by using the data record, or interrupts. See examples x_dr_motioncomplete() and x_ei_motioncomplete().

Parameters

g	Connection's handle.
axes	A null-terminated string containing a multiple-axes mask. Every character in the string should be a valid argument to MG_BGm, i.e. XYZWABCDEFGHST.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gmotioncomplete.cpp for an example.
Definition at line 300 of file gclibo.c.
References G_NO_ERROR, GWaitForBool(), and vector().
Referenced by contour(), jog(), position_tracking(), vector(), and x_gmotioncomplete().

7.1.4.21 GOpen()

```
GCLIB_DLL_EXPORTED GReturn GCALL GOpen (
            GCStringIn address,
            GCon * g )
```

Open a connection to a Galil Controller.

Parameters

address	Null-terminated address string. See table below.
g	Pointer to user's `GCon` variable. On success, the library will fill the user's variable with the handle to use for the rest of the connection. A valid `g` value is nonzero.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

address switch	Meaning	Arguments (default), other options	Examples
--address	**Simple address to hardware**	*IP address*, *PCI*, *COM port*	--address COM1
-a	shorthand for --address	See *Address Ranges* below	-a GALILPCI1
{no switch}	--address is implicit for any lone token		192.168.0.42
--baud	**Baud rate**	(115200), *valid baud...*	COM2 --baud 19200
-b	shorthand for --baud		COM3 -b 38400
--command	**Command-and-response socket protocol**	(TCP), UDP	192.168.0.42 --command TCP
-c	shorthand for --command		192.168.0.42 -c UDP
--direct	**Connect directly to hardware instead of via** gcaps		-a GALILPCI2 --direct
-d	shorthand for --direct		GALILPCI2 -d
--subscribe	**Subscribe to messages, data records, and/or interrupts**	(NONE), MG, DR, EI, ALL	192.168.0.42 --subscribe MG
-s	shorthand for --subscribe		192.168.0.42 -s DR -s EI
--timeout	**timeout in ms**	(5000), *0-65535*	192.168.0.42 --timeout 5000
-t	shorthand for --timeout		GALILPCI2 -t 500
--unsolicited	**Unsolicited socket protocol**	(UDP), NONE	192.168.0.42 --unsolicited NONE
-u	shorthand for --unsolicited		192.168.1.42 -u UDP
The following address switches are deprecated and will be unavailable starting July 1st, 2020.			
--p1	**Primary port for command-and-response traffic**	(23), *valid port number*	192.168.0.42 --p1 5000
--p2	**Secondary port for unsolicited traffic**	(60007), *valid port number*	192.168.0.42 --p2 5000

Operating System	Address Range	Notes
Windows	COM1 - COM256	RS232 and USB-to-serial
Linux	/dev/ttyS0 - /dev/ttyS255	RS232
Linux	/dev/ttyUSB0 - /dev/ttyUSB255	USB-to-serial, e.g. DMC-4103
Windows	GALILPCI1 - GALILPCI8	PCI
Linux	/dev/galilpci0 - /dev/galilpci7	PCI

See x_examples.cpp for an example.
When connecting to a network device, if the command-and-response socket is opened successfully but the unsolicited socket fails, GOpen() will still complete successfully. This allows connection to a Galil controller when only one Ethernet handle is available. Unsolicited traffic will not be accessible in this case.
Referenced by ip_assigner(), and main().

7.1.4.22 GProgramDownload()

```
GCLIB_DLL_EXPORTED GReturn GCALL GProgramDownload (
```

```
            GCon g,
            GCStringIn program,
            GCStringIn preprocessor )
```

Downloads a program to the controller's program buffer.

Parameters

g	Connection's handle.
program	Null-terminated program for download.
preprocessor	Options string for preprocessing the program before sending it to the controller. Null allows the library to use defaults for the download. See the Program Preprocessor documentation for options.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_programs.cpp for an example.
Referenced by GProgramDownloadFile(), GSetupDownloadFile(), message(), record_position(), x_ginterrupt(), x_gmessage(), x_nonblocking(), and x_programs().

7.1.4.23 GProgramDownloadFile()

```
GReturn GCALL GProgramDownloadFile (
            GCon g,
            GCStringIn file_path,
            GCStringIn preprocessor )
```

Program download from file.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the program file.
preprocessor	Options string for preprocessing the program before sending it to the controller. See GProgramDownload().

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_programs.cpp for an example.
Definition at line 387 of file gclibo.c.
References G_BAD_FILE, G_BAD_FULL_MEMORY, G_NO_ERROR, GProgramDownload(), and vector().
Referenced by x_programs().

7.1.4.24 GProgramUpload()

```
GCLIB_DLL_EXPORTED GReturn GCALL GProgramUpload (
            GCon g,
            GBufOut buffer,
            GSize buffer_len )
```

Uploads a program from the controller's program buffer.

Parameters

g	Connection's handle.
buffer	Buffer to receive the controller's program. The data will be null terminated unless function returns G_BAD_LOST_DATA due to the buffer being too small to hold the data.

Parameters

buffer_len	The length of the receive buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_programs.cpp for an example.
Referenced by GProgramUploadFile(), and x_programs().

7.1.4.25 GProgramUploadFile()

```
GReturn GCALL GProgramUploadFile (
            GCon g,
            GCStringIn file_path )
```

Program upload to file.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the program file, file will be overwritten if it exists.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_programs.cpp for an example.
Definition at line 430 of file gclibo.c.
References G_BAD_FILE, G_BAD_FULL_MEMORY, G_NO_ERROR, GProgramUpload(), MAXPROG, and vector().
Referenced by x_programs().

7.1.4.26 GPublishServer()

```
GReturn GCALL GPublishServer (
            GCStringIn name,
            GOption publish,
            GOption save )
```

Uses GUtility(), G_UTIL_GCAPS_PUBLISH_SERVER to publish local gcaps server to the local network.

Note

This function is only available on Windows 10 and Linux.

Parameters

name	The name of the server to publish or remove
publish	Option to publish or remove server from network
save	Option to save this configuration for future reboots

This function is used to make your local gcaps server "Discoverable" or "Invisible"
publish Option:
Set to 1 to publish server to the network and make "Discoverable"
Set to 0 to remove server from the network and make "Invisible"

save Option:

Set to 1 to save the configuration for future reboots of the server
Set to 0 to use this configuration once, and not overwrite previous server settings

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 189 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_PUBLISH_SERVER, GUtility(), and vector().
Referenced by remote_server().

7.1.4.27 GRead()

```
GCLIB_DLL_EXPORTED GReturn GCALL GRead (
            GCon g,
            GBufOut buffer,
            GSize buffer_len,
            GSize * bytes_read )
```

Performs a read on the connection.

Parameters

g	Connection's handle.
buffer	The user's read buffer.
buffer_len	The length of the user's read buffer.
bytes_read	Pointer to a GSize which will be filled with the number of bytes read upon return.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Warning

This function is deprecated and will be removed in a future gclib version. Please contact Galil for needs not covered by the other gclib functions.

Unsolicited messages may be returned in the read data. The high bit of each message byte will be set unless the user changes the CW setting. Interrupts and Data Records are always filtered from a read.
See x_gread_gwrite.cpp for an example.
Referenced by x_gread_gwrite().

7.1.4.28 GRecord()

```
GCLIB_DLL_EXPORTED GReturn GCALL GRecord (
            GCon g,
            union GDataRecord * record,
            GOption method )
```

Provides a fresh copy of the controller's data record. Data is cast into a union, GDataRecord.

Parameters

g	Connection's handle.
record	A pointer to the user's DataRecord union to hold the copy.

Parameters

method	Determines the method for acquiring the data. • G_QR: QR is used via command-and-response. • G_DR: DR is used for asynchronous acquisition.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

When using G_DR, the asynchronous data record must already be set up.

- -s DR must be used in the GOpen() address string to subscribe to records. The driver will automatically set the second argument of DR, where applicable.
- GRecordRate() should be issued to set DR to an appropriate interval, n. The interval must be no faster than the rate at which GRecord() is called.
- If GRecord() is called more slowly than the data record rate, stale data will be returned until GRecord() has been called once for each record sent by the controller.

GRecord() will block until the data record is received, or the transaction times out.

Note

If this function is called with a timeout of zero and the G_DR method, a non-blocking read is performed. If a data record has been processed since the last time the function was called, this data will be returned. If there is not a processed data reecord, but there is data waiting in the socket or PCI FIFO, one read will be performed to process the waiting data. If new data is still not found after these two attempts, G_GCLIB_↩ NON_BLOCKING_READ_EMPTY will be returned.

See x_grecord.cpp for an example. See x_nonblocking.cpp for an example of non-blocking usage.
Referenced by x_dr_motioncomplete(), x_grecord(), and x_nonblocking().

7.1.4.29 GRecordRate()

```
GReturn GCALL GRecordRate (
            GCon g,
            double period_ms )
```

Sets the asynchronous data record to a user-specified period via DR.
Takes TM and product type into account and sets the DR period to the period requested by the user, if possible.

Parameters

g	Connection's handle.
period_ms	Period, in milliseconds, to set up for the asynchronous data record.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_grecord.cpp for an example.
Definition at line 342 of file gclibo.c.
References G_NO_ERROR, G_SMALL_BUFFER, GCmd(), GCmdD(), GCmdT(), and vector().
Referenced by x_dr_motioncomplete(), x_grecord(), and x_nonblocking().

7.1.4.30 GRemoteConnections()

```
GReturn GCALL GRemoteConnections (
```

```
            GCStringOut connections,
            GSize connections_length )
```

Uses GUtility(), G_UTIL_GCAPS_REMOTE_CONNECTIONS to get a list of remote addresses connected to the local server.

Note

This function is only available on Windows 10 and Linux.

Parameters

connections	The buffer to hold the list of remote IP addresses currently connected to your hardware
connections_len	The length of the connections buffer

This function is used to find a list of IP Addresses of machines that currently have open connections to your local hardware. If another user sets your local server as their active server, and then opens a connection to your hardware, their IP Address will appear in this list.
The list of IP addresses are separated by a newline '\n' character.

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 217 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_REMOTE_CONNECTIONS, GUtility(), and vector().

7.1.4.31 GServerStatus()

```
GReturn GCALL GServerStatus (
            GCStringOut status,
            GSize status_len )
```

Uses GUtility(), G_UTIL_GCAPS_SERVER_STATUS to get information on the local server name and if it is published to the local network.

Note

This function is only available on Windows 10 and Linux.

Parameters

status	The buffer to hold the status of the local gcaps server
status_len	The length of the status buffer

This function is used to find the status of your local gcaps server. Use this function to determine the name your server is currently using, and whether or not your gcaps server is currently set to "Discoverable" or "Invisible"
The status buffer will be filled in the form of "[Server Name], [Discoverable]"
For example, for a server with the name "Example Server" that is set to "Discoverable", the status buffer would contain "Example Server, true".

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 149 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_SERVER_STATUS, GUtility(), and vector().

7.1.4.32 GSetServer()

```
GReturn GCALL GSetServer (
            GCStringIn server_name )
```

Uses GUtility(), G_UTIL_GCAPS_SET_SERVER to set the new active server.

Note

This function is only available on Windows 10 and Linux.

Parameters

server_name	The name of the server to set as your new active server.

Use this function in conjunction with GListServers(). Choose a name received from GListServers() to set as your new active server.
After setting a new active server, all gclib calls will route through that new active server, unless explicitly noted otherwise.
To set your active server back to your local server, simply pass "Local" to GSetServer():

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 128 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_SET_SERVER, GUtility(), and vector().
Referenced by remote_client().

7.1.4.33 GSetupDownloadFile()

```
GCLIB_DLL_EXPORTED GReturn GCALL GSetupDownloadFile (
            GCon g,
            GCStringIn file_path,
            GOption options,
            GCStringOut info,
            GSize info_len )
```

Download a saved controller configuration from a file.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the gcb file.
options	Bit mask to determine what configuration data to download. See below for all options.
info	Optional pointer to a buffer to store the controller info. If no info is needed, specify as NULL.
info_len	Length of optional info buffer. If no info is needed, specify as NULL.

Returns

The success status or error code of the function. If the options parameter is set to 0, the return value will be a bit mask indicating which sectors in the specified GCB are not empty. Otherwise, see gclib_errors.h for possible error values.

Note

By default, GSetupDownloadFile() will stop immediately if an error is encountered downloading data. This can be overridden in the options parameter. For example, you may want to override the error if you have a backup from an 8-axis controller and want to restore the parameters for the first 4 axes to a 4-axis controller.

If both info and info_len are not NULL, the controller information will be provided regardless of the options parameter. The options parameter is a bit mask. If options is set to 0, GSetupDownloadFile() will return a bit mask indicating which sectors in the specified GCB are not empty. The following contains a list of all currently available options:

Bit	Value	Function	Description
1	0x0002	Restore parameters	**KPA**, **KIA**, **KDA**, etc...
3	0x0008	Restore variables	Variables are listed by the **LV** command
4	0x0010	Restore arrays	Arrays are listed by the **LA** command
5	0x0020	Restore program	The program is listed by the **LS** command
31	0x8000	Ignore errors	Ignore invalid parameter errors and continue restoring data. GSetupDownloadFile() will still stop immediately if a connection issue or other fatal error is encountered

Usage example:

```
GCon g;
GOption opt = 0;

GCStringOut info;
GSize info_len = 4096;

GReturn rc = GOpen("192.168.0.50", &g);
if (rc) return rc;

// Call GSetupDownloadFile() with options set to 0 so we can get the non-empty sector bit mask
opt = GSetupDownloadFile(g, "C:\\path\\to\\gcb\\file.gcb", 0, NULL, NULL);

info = (GCStringOut)malloc(sizeof(GCStringOut) * info_len);

// Call GSetupDownloadFile() with the bit mask returned in the previous function call
rc = GSetupDownloadFile(g, "C:\\path\\to\\gcb\\file.gcb", opt, info, info_len);

printf("Info:\n\n%s", info);

GClose(g);

free(info);
return rc;
```

Definition at line 476 of file arrays.c.
References G_BAD_FILE, G_NO_ERROR, GProgramDownload(), H_ArrayDownloadFromMemory(), H_DownloadData(), H_FindSector(), and vector().

7.1.4.34 GSleep()

```
void GCALL GSleep (
            unsigned int timeout_ms )
```

Uses GUtility() and G_UTIL_SLEEP to provide a blocking sleep call which can be useful for timing-based chores.

Parameters

timeout_ms	The timeout, in milliseconds, to block before returning.

See GWaitForBool() for an example.
Definition at line 24 of file gclibo.c.
References G_UTIL_SLEEP, and GUtility().

Referenced by contour(), GIpRequests(), GWaitForBool(), record_position(), vector(), x_gread_gwrite(), and x_programs().

7.1.4.35 GTimeout()

```
GReturn GCALL GTimeout (
            GCon g,
            short timeout_ms )
```

Uses GUtility() and G_UTIL_TIMEOUT_OVERRIDE to set the library timeout.

Parameters

g	Connection's handle.
timeout_ms	The value to be used for the timeout. Use `G_USE_INITIAL_TIMEOUT` to set the timeout back to the initial GOpen() value, `--timeout`.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp and x_gread_gwrite.cpp for examples.
Definition at line 65 of file gclibo.c.
References G_UTIL_TIMEOUT_OVERRIDE, and GUtility().
Referenced by motion_complete(), x_gcommand(), x_ginterrupt(), x_gmessage(), x_gread_gwrite(), and x_nonblocking().

7.1.4.36 GUtility()

```
GCLIB_DLL_EXPORTED GReturn GCALL GUtility (
            GCon g,
            GOption request,
            GMemory memory1,
            GMemory memory2 )
```

Provides read/write access to driver settings and convenience features based on the request variable.

Note

The open source library, gclibo.h, has wrappers for most of these utilities.

Parameters

g	Connection's handle.
request	Defines the request. Input/Output and type of memory are implicit in the value of request. The following lists the supported request values.

- G_UTIL_TIMEOUT Read initial timeout value, as specified in GOpen() via `--timeout` switch.
  - `memory1` is output and must be an `unsigned short*`.
  - `memory2` is ignored, use null.
- G_UTIL_TIMEOUT_OVERRIDE See GTimeout(). Write/Read override timeout value.
  - `memory1` is input. If nonnull, value must be a `short*` holding the override, in milliseconds, for the timeout. Write `G_USE_INITIAL_TIMEOUT` to use initial timeout. If null, no write occurs.
  - `memory2` is output. If nonnul, value must be a `short*` which will be filled with the current override. `G_USE_INITIAL_TIMEOUT` indicates initial timeout used. If null, no read occurs. `memory2` is processed before 'memory1`.

- G_UTIL_VERSION See GVersion(). Returns the library version. A valid connection (g) is not necessary, i.e. g may be null.
  - memory1 is output, and must be a char*. Data will be null terminated, even if the data must be truncated to do so.
  - memory2 is input and must be an unsigned int* holding the length of the buffer in memory1.
- G_UTIL_INFO See GInfo(). Returns information about the connection.
  - memory1 is output and must be a char*. Data will be null terminated, even if the data must be truncated to do so.
  - memory2 is input and must be an unsigned int* holding the length of the buffer in memory1.
- G_UTIL_SLEEP See GSleep(). Platform-independent, non-busy, sleep. A valid connection (g) is not necessary, i.e. g may be null.
  - memory1 is input and must be an unsigned int*, units are milliseconds.
  - memory2 is ignored, use null.
- G_UTIL_ADDRESSES see GAddresses(). Provides a \n delimited listing of all available IP addresses, PCI addresses, and COM ports. A valid connection (g) is not necessary, i.e. g may be null. The suffix -d will be appended to each address to indicate these addresses are available via direct connection. See G_UTIL_↩ GCAPS_ADDRESSES for addresses through gcaps.
  - memory1 is output and must be a char*. Data will be null terminated, even if the data must be truncated to do so.
  - memory2 is input and must be an unsigned int* holding the length of the buffer in memory1.
- G_UTIL_IPREQUEST see GIpRequests(). Listens and returns a \n delimited listing of Galil MAC addresses sending BOOT-P or DHCP requests. The function will listen, and block, for roughly 5 seconds. A valid connection (g) is not necessary, i.e. g may be null.
  - memory1 is output and must be a char*. Data will be null terminated, even if the data must be truncated to do so.
  - memory2 is input and must be an unsigned int* holding the length of the buffer in memory1.
- G_UTIL_ASSIGN see GAssign(). Provides a method to assign an IP address given a Galil MAC address. A valid connection (g) is not necessary, i.e. g may be null.
  - memory1 is input and must be a char* containing the null terminated address that is to be assigned. e.g. "192.168.0.43".
  - memory2 is input and must be a char* containing the null terminated controller MAC address. e.g. "00:50:4C:20:01:23".
- G_UTIL_DEVICE_INITIALIZE Provides a method to reinitialize a connection after a reset, e.g. an RS command. Depending on the device type, the appropriate commands will be sent to configure the communication bus for optimal performance.
  - memory1 is ignored, use null.
  - memory2 is ignored, use null.
- G_UTIL_PING Uses ICMP ping to determine if an IP address is reachable and assigned. A valid connection (g) is not necessary, i.e. g may be null.
  - memory1 is input and must be a char* containing the null terminated address that is to be pinged. e.g. "192.168.0.43".
  - memory2 is output and must be an int*. The value will be set to zero if the ping times out, and nonzero if a ping reply is returned.
- G_UTIL_ERROR_CONTEXT More error detail for the last error on GCon, where available. The internal error message is cleared upon read.
  - memory1 is output and must be a char*. Data will be null terminated, even if the data must be truncated to do so.
  - memory2 is input and must be an unsigned int* holding the length of the buffer in memory1.

The following request values are for use with a @ref gcaps server.

- G_UTIL_GCAPS_VERSION see GVersion(). Returns the gcaps server version. A valid connection (g) is not necessary, i.e. g may be null. This operation will connect to the server to determine the version.
  - `memory1` is output and must be a `char*`. Data will be null terminated, even if the data must be truncated to do so.
  - `memory2` is input and must be an `unsigned int*` holding the length of the buffer in `memory1`.
- G_UTIL_GCAPS_ADDRESSES see GAddresses(). Provides a \n delimited listing of all available IP addresses, PCI addresses, and COM ports as available from the gcaps server. A valid connection (g) is not necessary, i.e. g may be null.
  - `memory1` is output and must be a `char*`. Data will be null terminated, even if the data must be truncated to do so.
  - `memory2` is input and must be an `unsigned int*` holding the length of the buffer in `memory1`.
- G_UTIL_GCAPS_IPREQUEST see GIpRequests(). Connects to gcaps and returns a \n delimited listing of Galil MAC addresses sending BOOT-P or DHCP requests. A valid connection (g) is not necessary, i.e. g may be null.
  - `memory1` is output and must be a `char*`. Data will be null terminated, even if the data must be truncated to do so.
  - `memory2` is input and must be an `unsigned int*` holding the length of the buffer in `memory1`.
- G_UTIL_GCAPS_ASSIGN see GAssign(). Provides a method to assign an IP address through gcaps given a Galil MAC address. A valid connection (g) is not necessary, i.e. g may be null.
  - `memory1` is input and must be a `char*` containing the null terminated address that is to be assigned. e.g. `"192.168.0.43"`.
  - `memory2` is input and must be a `char*` containing the null terminated controller MAC address. e.g. `"00:50:4C:20:01:23"`.
- G_UTIL_GCAPS_PING Uses ICMP ping to determine if an IP address is reachable and assigned. Ping sent from the gcaps server. A valid connection (g) is not necessary, i.e. g may be null.
  - `memory1` is input and must be a `char*` containing the null terminated address that is to be pinged. e.g. `"192.168.0.43"`.
  - `memory2` is output and must be an `int*`. The value will be set to zero if the ping times out, and nonzero if a ping reply is returned.

Parameters

memory1	An untyped pointer to data required for request. The data type is defined by the request variable.
memory2	An untyped pointer to data required for request. The data type is defined by the request variable.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See the following functions from gclibo, the open source portion, for implementation of several GUtility() requests.:

- GAddresses()
- GAssign()
- GInfo()
- GIpRequests()
- GSleep()

- GTimeout()
- GVersion()

Referenced by commands(), error(), GAddresses(), GAssign(), GInfo(), GIpRequests(), GListServers(), GPublishServer(), GRemoteConnections(), GServerStatus(), GSetServer(), GSleep(), GTimeout(), GVersion(), and message().

7.1.4.37 GVersion()

```
GReturn GCALL GVersion (
            GCStringOut ver,
            GSize ver_len )
```

Uses GUtility(), G_UTIL_VERSION and G_UTIL_GCAPS_VERSION to provide the library and gcaps version numbers.

Parameters

ver	Buffer to hold the output string. Buffer will be null terminated, even if the data must be truncated to do so.
ver_len	Length of buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

The version number of gclib is provided first. If the gcaps server can be found, its version will be provided after a space.
Example with gcaps version.

```
154.190.329 1.0.0.82
```

Example with gclib version only.

```
154.190.329
```

Note

GVersion() will take up to 1 second to look for gcaps.

See x_examples.cpp for an example.
Definition at line 29 of file gclibo.c.
References G_NO_ERROR, G_UTIL_GCAPS_VERSION, G_UTIL_VERSION, GUtility(), and vector().

7.1.4.38 GWaitForBool()

```
GReturn GCALL GWaitForBool (
            GCon g,
            GCStringIn predicate,
            int trials )
```

Blocking call that returns when the controller evaluates the predicate as true.
Polls the message command (MG) to check the value of predicate. Polling will continue until the controller responds with a nonzero value or the number of polling trials is reached.
The amount of time until the function fails with G_GCLIB_POLLING_FAILED is roughly (trials ∗ POLLINGINTERVAL) milliseconds.

Parameters

g	Connection's handle.
predicate	A null-terminated string containing the predicate to be polled. The predicate will be enclosed in parentheses and used in the command `MG (predicate)` to return the value.
trials	The number of polling cycles to perform looking for a nonzero value. Use -1 to poll indefinitely.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See GMotionComplete() for an example.
Definition at line 318 of file gclibo.c.
References G_GCLIB_POLLING_FAILED, G_LINE_BUFFER, G_NO_ERROR, G_SMALL_BUFFER, GCommand(), GSleep(), POLLINGINTERVAL, and vector().
Referenced by GMotionComplete().

7.1.4.39 GWrite()

```
GCLIB_DLL_EXPORTED GReturn GCALL GWrite (
            GCon g,
            GBufIn buffer,
            GSize buffer_len )
```

Performs a write on the connection.

Parameters

g	Connection's handle.
buffer	The user's write buffer. To send a Galil command, a terminating carriage return is usually required.
buffer_len	The length of the data in the buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values. If G_NO_ERROR is returned, all bytes were written.

Warning

This function is deprecated and will be removed in a future gclib version. Please contact Galil for needs not covered by the other gclib functions.

See x_gread_gwrite.cpp for an example.
Referenced by x_gread_gwrite().

7.2 C++ examples

7.2.1 Description

Files included in the C++ examples.

Files

- file commands.cpp
- file commands_example.cpp
- file contour.cpp
- file contour_example.cpp
- file examples.h
- file ip_assigner.cpp
- file ip_assigner_example.cpp
- file jog.cpp
- file jog_example.cpp
- file message.cpp
- file message_example.cpp
- file motion_complete.cpp
- file motion_complete_example.cpp

- file position_tracking.cpp
- file position_tracking_example.cpp
- file record_position.cpp
- file record_position_example.cpp
- file remote_client.cpp
- file remote_client_example.cpp
- file remote_server.cpp
- file remote_server_example.cpp
- file vector.cpp
- file vector_example.cpp

Macros

- #define _CRT_SECURE_NO_WARNINGS
- #define GALIL_EXAMPLE_OK 0
- #define GALIL_EXAMPLE_ERROR -100
- #define G_LASTINDEX 999

Typedefs

- typedef std::vector< string > tokens

Functions

- GReturn commands (GCon g)

  Demonstrates various uses of GCommand() and GUtility().
- int main (int argc, char *argv[])

  Main function for Commands Example.
- bool load_buf (GCon g, const std::vector< int > &positions_A, const std::vector< int > &positions_B, int capacity, int &cmd)

  Loads contour buffer with commands from the given text file.
- std::vector< int > csv_to_vector (ifstream &is)

  Converts a file of comma separated values to a vector.
- GReturn contour (GCon g, char *fileA, char *fileB)

  Record user's training and plays back training through contour mode.
- void e (GReturn rc)

  A trivial, C++ style return code check used in Galil's examples and demos.
- void error (GCon g, GReturn rc)

  An example of error handling and debugging information.
- int pause ()

  Pauses console apps for a user key stroke.
- GReturn position_tracking (GCon g, int speed=5000)

  Puts controller into Position Tracking Mode and accepts user-entered positions.
- GReturn jog (GCon g)

  Puts controller into Jog Mode and accepts user input to adjust the speed.
- GReturn vector (GCon g, char *file)

  Puts controller into Vector Mode and accepts a file defining vector points.
- GReturn ip_assigner (char *serial_num, int address)

  Assigns controller an IP Adress given a serial number and a 1 byte address.
- GReturn motion_complete (GCon g)

  Uses interrupts to track when the motion of controller is completed.
- GReturn message (GCon g)

  Demonstrates how to receive messages from the controller and detect differences in Trace and crashed code.
- GReturn record_position (GCon g, char *fileA, char *fileB)

Record user's training and saves to a text file.

- GReturn remote_server (const char *server_name)

  Publishes local gcaps server to the network.

- GReturn remote_client ()

  Lists available remote servers and allows connection to remote server.

- tokens string_split (const string &str, const string &token)

  Splits a string into a vector based on a token.

- int check_interrupts (GCon g, GCStringIn axes)

  Monitors interrupt status on the given axes and returns when interrupts are fired.

- void write_array_to_file (GCon g, ofstream &os, const char *array_name, int previous_rd, int rd)

  Grabs data from array on controller and writes it to the given text file.

- void print_client_message (const char *message)
- void print_servers_list (const std::vector< std::string > &server_list)
- void servers_to_list (std::vector< std::string > &server_list, std::string servers)
- void print_server_message (const char *message)
- bool load_buffer (GCon g, ifstream &fs, int capacity)

7.2.2 Macro Definition Documentation

7.2.2.1 _CRT_SECURE_NO_WARNINGS

```
#define _CRT_SECURE_NO_WARNINGS
```

Definition at line 16 of file examples.h.

7.2.2.2 G_LASTINDEX

```
#define G_LASTINDEX 999
```

Definition at line 13 of file record_position.cpp.

7.2.2.3 GALIL_EXAMPLE_ERROR

```
#define GALIL_EXAMPLE_ERROR -100
```

Definition at line 25 of file examples.h.

7.2.2.4 GALIL_EXAMPLE_OK

```
#define GALIL_EXAMPLE_OK 0
```

Definition at line 24 of file examples.h.

7.2.3 Typedef Documentation

7.2.3.1 tokens

```
typedef std::vector<string> tokens
```

Definition at line 16 of file ip_assigner.cpp.

7.2.4 Function Documentation

7.2.4.1 check_interrupts()

```
int check_interrupts (
            GCon g,
            GCStringIn axes )
```

Monitors interrupt status on the given axes and returns when interrupts are fired.
Definition at line 77 of file motion_complete.cpp.
References e(), G_NO_ERROR, G_SMALL_BUFFER, GCmd(), GInterrupt(), and vector().
Referenced by motion_complete().

7.2.4.2 commands()

```
GReturn commands (
            GCon g )
```

Demonstrates various uses of GCommand() and GUtility().

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See commands_example.cpp for an example.
Definition at line 16 of file commands.cpp.
References e(), G_SMALL_BUFFER, G_UTIL_ERROR_CONTEXT, GCmd(), GCmdD(), GCmdI(), GCmdT(), GCommand(), and GUtility().
Referenced by main().

7.2.4.3 contour()

```
GReturn contour (
            GCon g,
            char * fileA,
            char * fileB )
```

Record user's training and plays back training through contour mode.

Parameters

g	Connection's handle.
fileA	A Path to a text file where training for Axis A will be recorded.
fileB	A Path to a text file where training for Axis B will be recorded.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See contour_example.cpp for an example.
Definition at line 20 of file contour.cpp.
References csv_to_vector(), e(), G_SMALL_BUFFER, GCmd(), GCmdI(), GMotionComplete(), GSleep(), load_buf(), record_position(), and vector().

7.2.4.4 csv_to_vector()

```
std::vector< int > csv_to_vector (
            ifstream & is )
```

Converts a file of comma separated values to a vector.
Definition at line 102 of file contour.cpp.
References G_SMALL_BUFFER, and vector().
Referenced by contour().

7.2.4.5 e()

```
void e (
            GReturn rc )  [inline]
```

A trivial, C++ style return code check used in Galil's examples and demos.
Throws GReturn if return value is not G_NO_ERROR. See Commands_Example.cpp for example usage and catch() handler.

Definition at line 33 of file examples.h.
References G_NO_ERROR, and vector().
Referenced by check_interrupts(), commands(), contour(), GclibJava::GArrayUpload(), H_ArrayDownloadFromMemory(), ip_assigner(), jog(), load_buf(), load_buffer(), main(), message(), motion_complete(), position_tracking(), record_position(), remote_client(), remote_server(), vector(), and write_array_to_file().

7.2.4.6 error()

```
void error (
            GCon g,
            GReturn rc )  [inline]
```

An example of error handling and debugging information.
Definition at line 40 of file examples.h.
References G_BAD_RESPONSE_QUESTION_MARK, G_NO_ERROR, G_SMALL_BUFFER, G_UTIL_ERROR_CONTEXT, GCommand(), GError(), GUtility(), and vector().
Referenced by GError(), and main().

7.2.4.7 ip_assigner()

```
GReturn ip_assigner (
            char * serial_num,
            int address )
```

Assigns controller an IP Adress given a serial number and a 1 byte address.

Parameters

serial_num	Serial Number of the controller.
address	A 1 byte address that defines the last byte of the IP Address.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See ip_assigner_example.cpp for an example.
This function will listen on the network for controllers requesting an IP Address. If a detected controller matches the serial number provided by the user, a new IP Address will be assigned based on the first 3 bytes of the detected IP Address combined with the user defined 1 byte address.
Definition at line 26 of file ip_assigner.cpp.
References e(), G_SMALL_BUFFER, GAssign(), GCmd(), GInfo(), GIpRequests(), GOpen(), string_split(), and vector().

7.2.4.8 jog()

```
GReturn jog (
            GCon g )
```

Puts controller into Jog Mode and accepts user input to adjust the speed.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See jog_example.cpp for an example.

Key	Usage
q	Quit Jogging

Key	Usage
a	-2000 counts / second
s	-500 counts / second
d	+500 counts / second
f	+2000 counts / second
r	Direction Reversal

Definition at line 29 of file jog.cpp.
References e(), G_CONNECTION_NOT_ESTABLISHED, G_SMALL_BUFFER, GCmd(), GMotionComplete(), and vector().

7.2.4.9 load_buf()

```
bool load_buf (
            GCon g,
            const std::vector< int > & positions_A,
            const std::vector< int > & positions_B,
            int capacity,
            int & cmd )
```

Loads contour buffer with commands from the given text file.
Definition at line 74 of file contour.cpp.
References e(), G_SMALL_BUFFER, GCmd(), and vector().
Referenced by contour().

7.2.4.10 load_buffer()

```
bool load_buffer (
            GCon g,
            ifstream & fs,
            int capacity )
```

Loads vector buffer with commands from the given text file.
Returns false when there are no more lines in the text file
Definition at line 88 of file vector.cpp.
References e(), GCmd(), and vector().
Referenced by vector().

7.2.4.11 main()

```
int main (
            int argc,
            char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.
Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.
contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.

message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

7.2.4.12 message()

```
GReturn message (
            GCon g )
```

Demonstrates how to receive messages from the controller and detect differences in Trace and crashed code.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See message_example.cpp for an example.
Definition at line 14 of file message.cpp.
References e(), G_NO_ERROR, G_SMALL_BUFFER, G_UTIL_GCAPS_KEEPALIVE, GCmd(), GMessage(), GProgramDownload(), GUtility(), message(), and vector().
Referenced by Examples::Message(), and message().

7.2.4.13 motion_complete()

```
GReturn motion_complete (
            GCon g )
```

Uses interrupts to track when the motion of controller is completed.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See motion_complete_example.cpp for an example.
Definition at line 18 of file motion_complete.cpp.
References check_interrupts(), e(), G_NO_ERROR, G_SMALL_BUFFER, G_UNSUPPORTED_FUNCTION, GCmd(), GCmdT(), GCommand(), GInterrupt(), GTimeout(), and vector().

7.2.4.14 pause()

```
int pause ( )  [inline]
```

Pauses console apps for a user key stroke.

Definition at line 62 of file examples.h.
Referenced by main().

7.2.4.15 position_tracking()

```
GReturn position_tracking (
            GCon g,
            int speed = 5000 )
```

Puts controller into Position Tracking Mode and accepts user-entered positions.

Parameters

g	Connection's handle.
speed	Optional speed of the controller in Position Tracking Mode. Default value of 5000.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See position_tracking_example.cpp for an example.
Definition at line 15 of file position_tracking.cpp.
References e(), G_CONNECTION_NOT_ESTABLISHED, G_SMALL_BUFFER, GCmd(), GMotionComplete(), and vector().

7.2.4.16 print_client_message()

```
void print_client_message (
            const char * message )
```

Definition at line 24 of file remote_client.cpp.

7.2.4.17 print_server_message()

```
void print_server_message (
            const char * message )
```

Definition at line 22 of file remote_server.cpp.

7.2.4.18 print_servers_list()

```
void print_servers_list (
            const std::vector< std::string > & server_list )
```

Definition at line 34 of file remote_client.cpp.

7.2.4.19 record_position()

```
GReturn record_position (
            GCon g,
            char * fileA,
            char * fileB )
```

Record user's training and saves to a text file.

Parameters

g	Connection's handle.
fileA	A Path to a text file where training for Axis A will be recorded.
fileB	A Path to a text file where training for Axis B will be recorded.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See record_position_example.cpp for an example.
Definition at line 20 of file record_position.cpp.
References e(), GCmd(), GCmdI(), GProgramDownload(), GSleep(), vector(), and write_array_to_file().
Referenced by contour().

7.2.4.20 remote_client()

```
GReturn remote_client ( )
```

Lists available remote servers and allows connection to remote server.

Returns

The success status or error code of the function.See gclib_errors.h for possible values.

See remote_client_example for an example.

Key	Usage
q	Quit
s	List available servers on then network
h	List available hardware on the current server
0-9	Connect to server instance by number
l	Connect back to local server

Definition at line 89 of file remote_client.cpp.
References e(), G_SMALL_BUFFER, GAddresses(), GListServers(), GSetServer(), and vector().

7.2.4.21 remote_server()

```
GReturn remote_server (
            const char * server_name )
```

Publishes local gcaps server to the network.

Parameters

Name	to publish server under.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See remote_server_example for an example.

Key	Usage
q	Quit
p	Publish this server to the network
r	Remove this server from the network

Definition at line 39 of file remote_server.cpp.
References e(), G_SMALL_BUFFER, GPublishServer(), and vector().

7.2.4.22 servers_to_list()

```
void servers_to_list (
            std::vector< std::string > & server_list,
            std::string servers )
```

Definition at line 54 of file remote_client.cpp.

7.2.4.23 string_split()

```
tokens string_split (
            const string & str,
            const string & token )
```

Splits a string into a vector based on a token.
Definition at line 96 of file ip_assigner.cpp.
References vector().
Referenced by ip_assigner().

7.2.4.24 vector()

```
GReturn vector (
            GCon g,
            char * file )
```

Puts controller into Vector Mode and accepts a file defining vector points.

Parameters

g	Connection's handle.
file	A Path to a file that defines vector commands.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See vector_example.cpp for an example.
Example text file:

```
VP -2219,-2667
VP -2523,-2832
VP 2844,-1425
VP 728,1971
VP 2127,183
VP -997,688
VP 725,-1893
VP 527,2899
VP -37,2523
VP 1277,1425
VP 857,2388
VP 1096,-1694
CR 1000,0,90
```

Definition at line 36 of file vector.cpp.
References e(), G_BAD_FILE, G_CONNECTION_NOT_ESTABLISHED, GCmd(), GCmdI(), GMotionComplete(), GSleep(), load_buffer(), and vector().
Referenced by py::__init__(), AddressConvert(), check_interrupts(), Examples::Commands(), Examples::Contour(), contour(), csv_to_vector(), e(), error(), GclibJava::finalize(), gclib::GAddresses(), GAddresses(), py::GAddresses(), py::GArrayDownload(), gclib::GArrayDownload(), GclibJava::GArrayDownload(), GArrayDownloadFile(), py::GArrayDownloadFile(), GclibJava::GArrayDownloadFile(), gclib::GArrayDownloadFile(), py::GArrayUpload(), gclib::GArrayUpload(), GclibJava::GArrayUpload(), GArrayUploadFile(), py::GArrayUploadFile(), GclibJava::GArrayUploadFile(), GclibJava::GArrayUploadFile(), gclib::GArrayUploadFile(), GAssign(), py::GAssign(), gclib::GAssign(), GclibJava::GAssign(), gclib::gclib(), py::GClose(), GCmd(), GCmdD(), gclib::GCmdD(), GCmdI(), gclib::GCmdI(), GCmdT(), py::GCommand(), GclibJava::GCommand(), gclib::GCommand(), GError(), py::GFirmwareDownload(), gclib::GFirmwareDownload(), GclibJava::GFirmwareDownload(), gclib::GInfo(), GInfo(), py::GInfo(), gclib::GInterrupt(), GclibJava::GInterrupt(), py::GInterrupt(), gclib::GIpRequests(), GIpRequests(), py::GIpRequests(), gclib::GListServers(), GListServers(), gclib::GMessage(), py::GMessage(), GMotionComplete(), py::GMotionComplete(), gclib::GMotionComplete(), py::GOpen(), gclib::GOpen(), GclibJava::GOpen(), py::GProgramDownload(), GclibJava::GProgramDownload(), GclibJava::GProgramDownload(), gclib::GProgramDownload(), GProgramDownloadFile(), py::GProgramDownloadFile(), gclib::GProgramDownloadFile(), GclibJava::GProgramDownloadFile(), GclibJava::GProgramDownloadFile(), gclib::GProgramUpload(), py::GProgramUpload(), GProgramUploadFile(), py::GProgramUploadFile(), gclib::GProgramUploadFile(), GclibJava::GProgramUploadFile(), GPublishServer(), gclib::GPublishServer(), GclibJava::GPublishServer(), gclib::GRead(), gclib::GRecord< T >(), gclib::GRecordRate(), GRecordRate(), gclib::GRemoteConnections(),

GRemoteConnections(), gclib::GServerStatus(), GServerStatus(), GSetServer(), gclib::GSetServer(), GclibJava::GSetServer(), GSetupDownloadFile(), py::GSetupDownloadFile(), gclib::GSetupDownloadFile(), py::GSleep(), py::GTimeout(), gclib::GVersion(), GVersion(), py::GVersion(), GWaitForBool(), gclib::GWrite(), H_AddArray(), H_ArrayAddElement(), H_ArrayDownloadFromMemory(), H_CreateArrayNode(), H_DownloadArraysFromList(), H_DownloadData(), H_FindSector(), H_FreeArrays(), H_InitArrayNode(), H_UploadArrayToList(), H_WriteArrayCsv(), ip_assigner(), Examples::IP_Assigner(), Examples::Jog(), jog(), load_buf(), load_buffer(), Remote_Client_Example::Main(), Commands_Example::Main(), Contour_Example::Main(), IP_Assigner_Example::Main(), Jog_Example::Main(), Message_Example::Main(), Motion_Complete_Example::Main(), Position_Tracking_Example::Main(), Record_Position_Example::Mai Remote_Server_Example::Main(), Vector_Mode_Example::Main(), Examples::Message(), message(), Examples::Motion_Complete() motion_complete(), Examples::Position_Tracking(), position_tracking(), Examples::PrintError(), Examples::Record_Position(), record_position(), remote_client(), Examples::Remote_Client(), remote_server(), Examples::Remote_Server(), string_split(), vector(), Examples::Vector_Mode(), write_array_to_file(), x_arrays(), x_dr_motioncomplete(), x_e(), x_ei_motioncomplete(), x_gcommand(), x_ginterrupt(), x_gmessage(), x_gmotioncomplete(), x_gread_gwrite(), x_grecord(), x_nonblocking(), and x_programs().

7.2.4.25 write_array_to_file()

```
void write_array_to_file (
            GCon g,
            ofstream & os,
            const char * array_name,
            int previous_rd,
            int rd )
```

Grabs data from array on controller and writes it to the given text file.
Definition at line 67 of file record_position.cpp.
References e(), G_COMMA, G_HUGE_BUFFER, GArrayUpload(), and vector().
Referenced by record_position().

7.3 C#/VB examples

7.3.1 Description

Files included in the C# Examples.

Files

- file commands.cs
- file commands_example.cs
- file contour.cs
- file contour_example.cs
- file examples.cs
- file ip_assigner.cs
- file ip_assigner_example.cs
- file jog.cs
- file jog_example.cs
- file message.cs
- file message_example.cs
- file motion_complete.cs
- file motion_complete_example.cs
- file position_tracking.cs
- file position_tracking_example.cs
- file record_position.cs
- file record_position_example.cs
- file Remote_Client.cs
- file remote_client_example.cs
- file Remote_Server.cs
- file remote_server_example.cs

- file vector_mode.cs
- file vector_mode_example.cs

Data Structures

- class Commands_Example

  Demonstrates various uses of GCommand() and basic controller queries.
- class Contour_Example

  Record user's training and plays back training through contour mode.
- class Examples

  Provides a class of shared constants and methods for gclib's example projects.
- class IP_Assigner_Example

  Assigns controller an IP Adress given a serial number and a 1 byte address.
- class Jog_Example

  Accepts user-input at the command line to control the speed of the controller in Jog mode.
- class Message_Example

  Demonstrates how to handle and interpret messages from the controller.
- class Motion_Complete_Example

  Uses controller interrupts to detect when motion is complete.
- class Position_Tracking_Example

  Places controller into position tracking mode. Accepts user-defined positional values at the command line.
- class Record_Position_Example

  Takes two file paths at the command line to hold positional data for Axis A and Axis B. Positional data is saved to the two files until an analog input value changes.
- class Remote_Client_Example

  Demonstrates various uses of GListServers() and GSetServer()
- class Remote_Server_Example

  Demonstrates various uses of GPublishServer()
- class Vector_Mode_Example

  Takes a path to a file at the command line holding vector commands for the controller. The controller is placed into vector mode and commands are read from the file and sent to the controller.

Functions

- static int Commands (gclib gclib)

  Demonstrates various uses of GCommand() and basic controller queries.
- static int Contour (gclib gclib, string fileA, string fileB)

  Record user's training and plays back training through contour mode.
- static int IP_Assigner (gclib gclib, string serial_num, byte address)

  Assigns controller an IP Adress given a serial number and a 1 byte address.
- static int Jog (gclib gclib)

  Puts controller into Jog Mode and accepts user input to adjust the speed.
- static int Message (gclib gclib)

  Demonstrates how to receive messages from the controller and detect differences in Trace and crashed code.
- static int Motion_Complete (gclib gclib)

  Uses interrupts to track when the motion of controller is completed.
- static int Position_Tracking (gclib gclib, int speed)

  Puts controller into Position Tracking Mode and accepts user-entered positions.
- static int Record_Position (gclib gclib, string fileA, string fileB)

  Record user's training and saves to a text file.
- static int Remote_Client ()

  Accepts user input to publish to list and connect to available servers.

- static int Remote_Server (string server_name)

  Accepts user input to publish or remove local gcaps server from the network.
- static int Vector_Mode (gclib gclib, string file)

  Puts controller into Vector Mode and accepts a file defining vector points.

7.3.2 Function Documentation

7.3.2.1 Commands()

```
static int Commands (
            gclib gclib )  [inline], [static]
```

Demonstrates various uses of GCommand() and basic controller queries.

Parameters

gclib	A gclib object with a valid connection.

Returns

The success status or error code of the function.

See commands_example.cs for an example.
For VB.NET, see definition in file commands.vb
Definition at line 28 of file commands.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCmdD(), gclib.GCmdI(), gclib.GCommand(), and vector().
Referenced by Commands_Example.Main().

7.3.2.2 Contour()

```
static int Contour (
            gclib gclib,
            string fileA,
            string fileB )  [inline], [static]
```

Record user's training and plays back training through contour mode.

Parameters

gclib	A gclib object with a valid connection.
fileA	A Path to a text file where training for Axis A will be recorded.
fileB	A Path to a text file where training for Axis B will be recorded.

Returns

The success status or error code of the function.

See contour_example.cs for an example.
For VB.NET, see definition in file contour.vb
Definition at line 32 of file contour.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GCmdI(), gclib.GCommand(), gclib.GMotionComplete(), Examples.Record_Position(), and vector().
Referenced by Contour_Example.Main().

7.3.2.3 IP_Assigner()

```
static int IP_Assigner (
            gclib gclib,
            string serial_num,
            byte address )  [inline], [static]
```

Assigns controller an IP Adress given a serial number and a 1 byte address.

Parameters

gclib	A gclib object.
serial_num	The serial number of a Galil controller.
address	A 1 byte value to be added to the new IP Address.

Returns

The success status or error code of the function.

This function will listen on the network for controllers requesting an IP Address.
If a detected controller matches the serial number provided by the user, a new IP Address will be assigned based on the first 3 bytes of the detected IP Address combined with the user defined 1 byte address.

See ip_assigner_example.cs for an example.
For VB.NET, see definition in file ip_assigner.vb
Definition at line 36 of file ip_assigner.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GAssign(), gclib.GCommand(), gclib.GInfo(), gclib.GIpRequests(), gclib.GOpen(), and vector().
Referenced by IP_Assigner_Example.Main().

7.3.2.4 Jog()

```
static int Jog (
            gclib gclib )  [inline], [static]
```

Puts controller into Jog Mode and accepts user input to adjust the speed.

Parameters

gclib	A gclib object with a valid connection.

Returns

The success status or error code of the function.

Key	Usage
q	Quit Jogging
a	-2000 counts / second
s	-500 counts / second
d	+500 counts / second
f	+2000 counts / second
r	Direction Reversal

See jog_example.cs for an example.
For VB.NET, see definition in file jog.vb
Definition at line 35 of file jog.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCommand(), gclib.GMotionComplete(), and vector().
Referenced by Jog_Example.Main().

7.3.2.5 Message()

```
static int Message (
            gclib gclib )  [inline], [static]
```

Demonstrates how to receive messages from the controller and detect differences in Trace and crashed code.

Parameters

gclib	A gclib object with a valid connection.

Returns

The success status or error code of the function.

See message_example.cs for an example.
For VB.NET, see definition in file message.vb
Definition at line 27 of file message.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCommand(), gclib.GMessage(), gclib.GProgramDownload(), message(), and vector().
Referenced by Message_Example.Main().

7.3.2.6 Motion_Complete()

```
static int Motion_Complete (
            gclib gclib )  [inline], [static]
```

Uses interrupts to track when the motion of controller is completed.

Parameters

gclib	A gclib object with a valid connection.

Returns

The success status or error code of the function.

See motion_complete_example.cs for an example.

For VB.NET, see definition in file motion_complete.vb
Definition at line 26 of file motion_complete.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GCommand(), gclib.GInterrupt(), gclib.GTimeout(), and vector().
Referenced by Motion_Complete_Example.Main().

7.3.2.7 Position_Tracking()

```
static int Position_Tracking (
            gclib gclib,
            int speed )  [inline], [static]
```

Puts controller into Position Tracking Mode and accepts user-entered positions.

Parameters

gclib	A gclib object with a valid connection.
speed	Optional speed of the controller in Position Tracking Mode. Default value of 5000

Returns

The success status or error code of the function.

See position_tracking_example.cs for an example.
For VB.NET, see definition in file position_tracking.vb
Definition at line 28 of file position_tracking.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCommand(), gclib.GMotionComplete(), and vector().
Referenced by Position_Tracking_Example.Main().

7.3.2.8 Record_Position()

```
static int Record_Position (
            gclib gclib,
            string fileA,
            string fileB )  [inline], [static]
```

Record user's training and saves to a text file.

Parameters

gclib	A gclib object with a valid connection.
fileA	A Path to a text file where training for Axis A will be recorded.
fileB	A Path to a text file where training for Axis B will be recorded.

Returns

The success status or error code of the function.

See record_position_example.cs for an example.
For VB.NET, see definition in file record_position.vb
Definition at line 32 of file record_position.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCmdI(), gclib.GCommand(), gclib.GProgramDownload(), and vector().
Referenced by Examples.Contour(), and Record_Position_Example.Main().

7.3.2.9 Remote_Client()

```
static int Remote_Client ( )  [inline], [static]
```

Accepts user input to publish to list and connect to available servers.

Returns

The success status or error code of the function.

Key	Usage
q	Quit
s	List available servers on then network
h	List available hardware on the current server
0-9	Connect to server instance by number
l	Connect back to local server

See remote_client_example.cs for an example.
For VB.NET, see definition in file remote_client.vb
Definition at line 33 of file Remote_Client.cs.
References gclib.GAddresses(), Examples.GALIL_EXAMPLE_OK, gclib.GListServers(), gclib.GSetServer(), and vector().
Referenced by Remote_Client_Example.Main().

7.3.2.10 Remote_Server()

```
static int Remote_Server (
            string server_name )  [inline], [static]
```

Accepts user input to publish or remove local gcaps server from the network.

Parameters

server_name	The name to publish local gcaps server under.

Returns

The success status or error code of the function.

Key	Usage
q	Quit
p	Publish this server to the network
r	Remove this server from the network

See remote_server_example.cs for an example.
For VB.NET, see definition in file remote_server.vb
Definition at line 32 of file Remote_Server.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GPublishServer(), and vector().
Referenced by Remote_Server_Example.Main().

7.3.2.11 Vector_Mode()

```
static int Vector_Mode (
            gclib gclib,
            string file )  [inline], [static]
```

Puts controller into Vector Mode and accepts a file defining vector points.

Parameters

gclib	A gclib object with a valid connection.
file	A path to a file with stored vector commands.

Returns

The success status or error code of the function.

Example text file:

```
VP -2219,-2667
VP -2523,-2832
VP 2844,-1425
VP 728,1971
VP 2127,183
VP -997,688
VP 725,-1893
VP 527,2899
VP -37,2523
VP 1277,1425
VP 857,2388
VP 1096,-1694
CR 1000,0,90
```

See vector_mode_example.cs for an example.
For VB.NET, see definition in file vector_mode.vb
Definition at line 45 of file vector_mode.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCmdI(), gclib.GCommand(), gclib.GMotionComplete(), and vector().
Referenced by Vector_Mode_Example.Main().

7.4 C#/VB API

7.4.1 Description

Files included in the C#/VB API.

Data Structures

- class gclib

Provides a class that binds to gclib's unmanaged dll. Wraps each call and provides a more user-friendly interface for use in C#.

Chapter 8

Namespace Documentation

8.1 examples Namespace Reference

Data Structures

- class Commands_Example

  Demonstrates various uses of GCommand() and basic controller queries.
- class Contour_Example

  Record user's training and plays back training through contour mode.
- class Examples

  Provides a class of shared constants and methods for gclib's example projects.
- class IP_Assigner_Example

  Assigns controller an IP Adress given a serial number and a 1 byte address.
- class Jog_Example

  Accepts user-input at the command line to control the speed of the controller in Jog mode.
- class Message_Example

  Demonstrates how to handle and interpret messages from the controller.
- class Motion_Complete_Example

  Uses controller interrupts to detect when motion is complete.
- class Position_Tracking_Example

  Places controller into position tracking mode. Accepts user-defined positional values at the command line.
- class Record_Position_Example

  Takes two file paths at the command line to hold positional data for Axis A and Axis B. Positional data is saved to the two files until an analog input value changes.
- class Remote_Client_Example

  Demonstrates various uses of GListServers() and GSetServer()
- class Remote_Server_Example

  Demonstrates various uses of GPublishServer()
- class Vector_Mode_Example

  Takes a path to a file at the command line holding vector commands for the controller. The controller is placed into vector mode and commands are read from the file and sent to the controller.

Functions

- int Commands (Gclib gclib)

  Demonstrates various uses of GCommand() And basic controller queries.
- int Main ()

  Main function for the commands example.
- int **Contour** (Gclib gclib, string fileA, string fileB)
- void PrintError (Gclib gclib, Exception ex)

Prints the exception to the console And queries the controller for the most recent error message.

- int **IP_Assigner** (Gclib gclib, string serial_num, byte address)
- int **Jog** (Gclib gclib)
- **Message** (Gclib gclib)
- **Motion_Complete** (Gclib gclib)
- **Position_Tracking** (Gclib gclib, int speed)
- **Record_Position** (Gclib gclib, string fileA, string fileB)
- int Remote_Client ()

  Demonstrates various uses of GListServers() and GSetServer()
- int Remote_Server (string server_name)

  Demonstrates various uses of GPublishServer()
- int **Vector_Mode** (Gclib gclib, string file)

Variables

- partial Module Examples
- positions_A As List< string >
- const int GALIL_EXAMPLE_OK = 0
- const GALIL_EXAMPLE_ERROR = -100

8.1.1 Function Documentation

8.1.1.1 Commands()

```
int Commands (
            Gclib gclib )
```

Demonstrates various uses of GCommand() And basic controller queries.

Parameters

gclib	A gclib object with a valid connection.

Returns

The success status Or error code of the function.

See commands_example.cs for an example.

8.1.1.2 Main()

```
int Main ( )
```

Main function for the commands example.
Main function for the Remote Server example.
Main function for the Remote Client example.
Demonstrates various uses of GCommand() And basic controller queries.
The first argument should be the IP Address of a Galil controller.

Returns

The success status Or error code of the function.

The first argument should be the IP Address of a Galil controller.
Demonstrates various uses of GListServers() and GSetServer().
This example requires no command line arguments.

Returns

The success status Or error code of the function.

This example requires no command line arguments.
Demonstrates various uses of GPublishServer().
The first argument is optional and defines the name to publish the server under.

Returns

The success status Or error code of the function.

The first argument is optional and defines the name to publish the server under.

8.1.1.3 PrintError()

```
void PrintError (
            Gclib gclib,
            Exception ex )
```

Prints the exception to the console And queries the controller for the most recent error message.

Parameters

gclib	The gclib object from where the exception originated.
ex	The exception object caught by the example.

See commands_example.cs for an example.

8.1.1.4 Remote_Client()

```
int Remote_Client ( )
```

Demonstrates various uses of GListServers() and GSetServer()

Returns

The success status Or error code of the function.

See remote_client_example.cs for an example.

8.1.1.5 Remote_Server()

```
int Remote_Server (
            string server_name )
```

Demonstrates various uses of GPublishServer()

Parameters

server_name	The name to publish the server under.

Returns

The success status Or error code of the function.

See remote_server_example.cs for an example.

8.1.2 Variable Documentation

8.1.2.1 Examples

```
partial Module Examples
```

Provides a class of shared constants And methods for gclib's example projects.
Definition at line 4 of file Commands.vb.

8.1.2.2 GALIL_EXAMPLE_ERROR

```
const GALIL_EXAMPLE_ERROR = -100  [static]
```
Definition at line 10 of file Examples.vb.

8.1.2.3 GALIL_EXAMPLE_OK

```
const int GALIL_EXAMPLE_OK = 0  [static]
```
Definition at line 9 of file Examples.vb.

8.1.2.4 List< string >

```
positions_A As positions_B As List< string >
```
Definition at line 6 of file Contour.vb.

8.2 gclib_example Namespace Reference

Data Structures

- class MainForm

  Demonstrates using gclib in a Windows Form, including using a second thread to free the GUI.
- class Program

8.3 Package gclibjava

Data Structures

- class GclibJava
- class GclibJavaException

8.3.1 Detailed Description

This is the initial version of the the gclib Java wrapper. All functions are subject to change in future releases of gclib.
Java hackers with recommendations on how to make this library better are encouraged to email softwaresupport@galil.com. Somebody has to teach those Galil Java noobs what's what.
Some identified "To Do" tasks:

1. Synchronize access to Gclib and Gclibo interfaces.
2. Choose a data structure to return GAddresses and GIpRequests. 3.

8.4 vb Namespace Reference

Data Structures

- class Gclib

  Provides a class that binds to gclib's unmanaged dll. Wraps each call and provides a more user-friendly interface for use in Visual Basic.

Typedefs

- using GReturn = System.Int32
- using GCon = System.IntPtr
- using GSize = System.UInt32
- using GCStringOut = System.Text.StringBuilder
- using GCStringIn = System.String
- using GBufOut = System.Text.StringBuilder
- using GBufIn = System.String
- using GStatus = System.Byte

Variables

- Module LibraryPath
- const string GclibDllPath_ = "C:\Program Files[] \Galil\gclib\dll\x86\gclib.dll"
- const string GcliboDllPath_ = "C:\Program Files[] \Galil\gclib\dll\x86\gclibo.dll"

8.4.1 Typedef Documentation

8.4.1.1 GBufIn

```
typedef System String GBufIn
```
Definition at line 19 of file gclib.vb.

8.4.1.2 GBufOut

```
typedef System Text StringBuilder GBufOut
```
Definition at line 18 of file gclib.vb.

8.4.1.3 GCon

```
typedef System IntPtr GCon
```
Definition at line 14 of file gclib.vb.

8.4.1.4 GCStringIn

```
typedef System String GCStringIn
```
Definition at line 17 of file gclib.vb.

8.4.1.5 GCStringOut

```
typedef System Text StringBuilder GCStringOut
```
Definition at line 16 of file gclib.vb.

8.4.1.6 GReturn

```
typedef System Int32 GReturn
```
Definition at line 13 of file gclib.vb.

8.4.1.7 GSize

```
typedef System UInt32 GSize
```
Definition at line 15 of file gclib.vb.

8.4.1.8 GStatus

```
typedef System Byte GStatus
```
Definition at line 20 of file gclib.vb.

8.4.2 Variable Documentation

8.4.2.1 GclibDllPath_

```
const string GclibDllPath_ = "C:\Program Files[] \Galil\gclib\dll\x86\gclib.dll"  [static]
```
Definition at line 22 of file gclib.vb.

8.4.2.2 GcliboDllPath_

```
const string GcliboDllPath_ = "C:\Program Files[] \Galil\gclib\dll\x86\gclibo.dll"  [static]
```
Definition at line 23 of file gclib.vb.

8.4.2.3 LibraryPath

```
Module LibraryPath
```

Definition at line 21 of file gclib.vb.

Chapter 9

Data Structure Documentation

9.1 Commands_Example Class Reference

Demonstrates various uses of GCommand() and basic controller queries.

Static Public Member Functions

- static int Main (string[] args)

  Main function for the commands example.

9.1.1 Detailed Description

Demonstrates various uses of GCommand() and basic controller queries.
The first argument should be the IP Address of a Galil controller.
For VB.NET, see definition in file commands_example.vb
Definition at line 23 of file commands_example.cs.

9.1.2 Member Function Documentation

9.1.2.1 Main()

```
static int Main (
            string[] args )  [inline], [static]
```

Main function for the commands example.

Parameters

args	An array of command line arguments.

Returns

The success status or error code of the function.

The first argument should be the IP Address of a Galil controller.
Definition at line 31 of file commands_example.cs.
References Examples.Commands(), Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GClose(), gclib.GOpen(), Examples.PrintError(), and vector().
The documentation for this class was generated from the following file:

- commands_example.cs

9.2 Contour_Example Class Reference

Record user's training and plays back training through contour mode.

Static Public Member Functions

- static int Main (string[] args)

  Main function for the contour example.

9.2.1 Detailed Description

Record user's training and plays back training through contour mode.
The first argument should be the IP Address of a Galil controller.
The second argument should be a path to a csv file holding positional data for the A axis.
The third argument should be a path to a csv file holding positional data for the B axis.
For VB.NET, see definition in file contour_example.vb
Definition at line 27 of file contour_example.cs.

9.2.2 Member Function Documentation

9.2.2.1 Main()

```
static int Main (
            string[] args )  [inline], [static]
```

Main function for the contour example.

Parameters

args	An array of command line arguments.

Returns

The success status or error code of the function.

The first argument should be the IP Address of a Galil controller.
The second argument should be a path to a text file where training for Axis A will be recorded.
The third argument should be a path to a text file where training for Axis B will be recorded..
Definition at line 39 of file contour_example.cs.
References Examples.Contour(), Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GClose(), gclib.GOpen(), Examples.PrintError(), and vector().
The documentation for this class was generated from the following file:

- contour_example.cs

9.3 Examples Class Reference

Provides a class of shared constants and methods for gclib's example projects.

Static Public Member Functions

- static int Commands (gclib gclib)

  Demonstrates various uses of GCommand() and basic controller queries.
- static int Contour (gclib gclib, string fileA, string fileB)

  Record user's training and plays back training through contour mode.
- static void PrintError (gclib gclib, Exception ex)

  Prints the exception to the console and queries the controller for the most recent error message.
- static int IP_Assigner (gclib gclib, string serial_num, byte address)

  Assigns controller an IP Adress given a serial number and a 1 byte address.
- static int Jog (gclib gclib)

  Puts controller into Jog Mode and accepts user input to adjust the speed.

- static int Message (gclib gclib)

  Demonstrates how to receive messages from the controller and detect differences in Trace and crashed code.
- static int Motion_Complete (gclib gclib)

  Uses interrupts to track when the motion of controller is completed.
- static int Position_Tracking (gclib gclib, int speed)

  Puts controller into Position Tracking Mode and accepts user-entered positions.
- static int Record_Position (gclib gclib, string fileA, string fileB)

  Record user's training and saves to a text file.
- static int Remote_Client ()

  Accepts user input to publish to list and connect to available servers.
- static int Remote_Server (string server_name)

  Accepts user input to publish or remove local gcaps server from the network.
- static int Vector_Mode (gclib gclib, string file)

  Puts controller into Vector Mode and accepts a file defining vector points.

Static Public Attributes

- const int GALIL_EXAMPLE_OK = 0

  Examples success code.
- const int GALIL_EXAMPLE_ERROR = -100

  Examples error code.

9.3.1 Detailed Description

Provides a class of shared constants and methods for gclib's example projects.
For VB.NET, see definition in file examples.vb
Definition at line 15 of file commands.cs.

9.3.2 Member Function Documentation

9.3.2.1 Commands()

```
static int Commands (
            gclib gclib )  [inline], [static]
```

Demonstrates various uses of GCommand() and basic controller queries.

Parameters

gclib	A gclib object with a valid connection.

Returns

The success status or error code of the function.

See commands_example.cs for an example.
For VB.NET, see definition in file commands.vb
Definition at line 28 of file commands.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCmdD(), gclib.GCmdI(), gclib.GCommand(), and vector().
Referenced by Commands_Example.Main().

9.3.2.2 Contour()

```
static int Contour (
            gclib gclib,
            string fileA,
            string fileB )  [inline], [static]
```

Record user's training and plays back training through contour mode.

Parameters

gclib	A gclib object with a valid connection.
fileA	A Path to a text file where training for Axis A will be recorded.
fileB	A Path to a text file where training for Axis B will be recorded.

Returns

The success status or error code of the function.

See contour_example.cs for an example.
For VB.NET, see definition in file contour.vb
Definition at line 32 of file contour.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GCmdI(), gclib.GCommand(), gclib.GMotionComplete(), Examples.Record_Position(), and vector().
Referenced by Contour_Example.Main().

9.3.2.3 IP_Assigner()

```
static int IP_Assigner (
            gclib gclib,
            string serial_num,
            byte address )  [inline], [static]
```

Assigns controller an IP Adress given a serial number and a 1 byte address.

Parameters

gclib	A gclib object.
serial_num	The serial number of a Galil controller.
address	A 1 byte value to be added to the new IP Address.

Returns

The success status or error code of the function.

This function will listen on the network for controllers requesting an IP Address.
If a detected controller matches the serial number provided by the user, a new IP Address will be assigned based on the first 3 bytes of the detected IP Address combined with the user defined 1 byte address.

See ip_assigner_example.cs for an example.
For VB.NET, see definition in file ip_assigner.vb
Definition at line 36 of file ip_assigner.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GAssign(), gclib.GCommand(), gclib.GInfo(), gclib.GIpRequests(), gclib.GOpen(), and vector().
Referenced by IP_Assigner_Example.Main().

9.3.2.4 Jog()

```
static int Jog (
            gclib gclib )  [inline], [static]
```

Puts controller into Jog Mode and accepts user input to adjust the speed.

Parameters

gclib	A gclib object with a valid connection.

Returns

The success status or error code of the function.

Key	Usage
q	Quit Jogging
a	-2000 counts / second
s	-500 counts / second
d	+500 counts / second
f	+2000 counts / second
r	Direction Reversal

See jog_example.cs for an example.
For VB.NET, see definition in file jog.vb
Definition at line 35 of file jog.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCommand(), gclib.GMotionComplete(), and vector().
Referenced by Jog_Example.Main().

9.3.2.5 Message()

```
static int Message (
            gclib gclib )  [inline], [static]
```

Demonstrates how to receive messages from the controller and detect differences in Trace and crashed code.

Parameters

gclib	A gclib object with a valid connection.

Returns

The success status or error code of the function.

See message_example.cs for an example.
For VB.NET, see definition in file message.vb
Definition at line 27 of file message.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCommand(), gclib.GMessage(), gclib.GProgramDownload(), message(), and vector().
Referenced by Message_Example.Main().

9.3.2.6 Motion_Complete()

```
static int Motion_Complete (
            gclib gclib )  [inline], [static]
```

Uses interrupts to track when the motion of controller is completed.

Parameters

gclib	A gclib object with a valid connection.

Returns

The success status or error code of the function.

See motion_complete_example.cs for an example.

For VB.NET, see definition in file motion_complete.vb
Definition at line 26 of file motion_complete.cs.

References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GCommand(), gclib.GInterrupt(), gclib.GTimeout(), and vector().
Referenced by Motion_Complete_Example.Main().

9.3.2.7 Position_Tracking()

```
static int Position_Tracking (
            gclib gclib,
            int speed )  [inline], [static]
```

Puts controller into Position Tracking Mode and accepts user-entered positions.

Parameters

gclib	A gclib object with a valid connection.
speed	Optional speed of the controller in Position Tracking Mode. Default value of 5000

Returns

The success status or error code of the function.

See position_tracking_example.cs for an example.
For VB.NET, see definition in file position_tracking.vb
Definition at line 28 of file position_tracking.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCommand(), gclib.GMotionComplete(), and vector().
Referenced by Position_Tracking_Example.Main().

9.3.2.8 PrintError()

```
static void PrintError (
            gclib gclib,
            Exception ex )  [inline], [static]
```

Prints the exception to the console and queries the controller for the most recent error message.

Parameters

gclib	The gclib object from where the exception originated.
ex	The exception object caught by the example.

See commands_example.cs for an example.
Definition at line 39 of file examples.cs.
References gclib.GCommand(), and vector().
Referenced by Commands_Example.Main(), Contour_Example.Main(), IP_Assigner_Example.Main(), Jog_Example.Main(), Message_Example.Main(), Motion_Complete_Example.Main(), Position_Tracking_Example.Main(), Record_Position_Example.Main() and Vector_Mode_Example.Main().

9.3.2.9 Record_Position()

```
static int Record_Position (
            gclib gclib,
            string fileA,
            string fileB )  [inline], [static]
```

Record user's training and saves to a text file.

Parameters

gclib	A gclib object with a valid connection.
fileA	A Path to a text file where training for Axis A will be recorded.

Parameters

fileB	A Path to a text file where training for Axis B will be recorded.

Returns

The success status or error code of the function.

See record_position_example.cs for an example.
For VB.NET, see definition in file record_position.vb
Definition at line 32 of file record_position.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCmdI(), gclib.GCommand(), gclib.GProgramDownload(), and vector().
Referenced by Examples.Contour(), and Record_Position_Example.Main().

9.3.2.10 Remote_Client()

```
static int Remote_Client ( )  [inline], [static]
```

Accepts user input to publish to list and connect to available servers.

Returns

The success status or error code of the function.

Key	Usage
q	Quit
s	List available servers on then network
h	List available hardware on the current server
0-9	Connect to server instance by number
l	Connect back to local server

See remote_client_example.cs for an example.
For VB.NET, see definition in file remote_client.vb
Definition at line 33 of file Remote_Client.cs.
References gclib.GAddresses(), Examples.GALIL_EXAMPLE_OK, gclib.GListServers(), gclib.GSetServer(), and vector().
Referenced by Remote_Client_Example.Main().

9.3.2.11 Remote_Server()

```
static int Remote_Server (
            string server_name )  [inline], [static]
```

Accepts user input to publish or remove local gcaps server from the network.

Parameters

server_name	The name to publish local gcaps server under.

Returns

The success status or error code of the function.

Key	Usage
q	Quit
p	Publish this server to the network
r	Remove this server from the network

See remote_server_example.cs for an example.
For VB.NET, see definition in file remote_server.vb
Definition at line 32 of file Remote_Server.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GPublishServer(), and vector().
Referenced by Remote_Server_Example.Main().

9.3.2.12 Vector_Mode()

```
static int Vector_Mode (
            gclib gclib,
            string file )  [inline], [static]
```

Puts controller into Vector Mode and accepts a file defining vector points.

Parameters

gclib	A gclib object with a valid connection.
file	A path to a file with stored vector commands.

Returns

The success status or error code of the function.

Example text file:

```
VP -2219,-2667
VP -2523,-2832
VP 2844,-1425
VP 728,1971
VP 2127,183
VP -997,688
VP 725,-1893
VP 527,2899
VP -37,2523
VP 1277,1425
VP 857,2388
VP 1096,-1694
CR 1000,0,90
```

See vector_mode_example.cs for an example.
For VB.NET, see definition in file vector_mode.vb
Definition at line 45 of file vector_mode.cs.
References Examples.GALIL_EXAMPLE_OK, gclib.GCmdI(), gclib.GCommand(), gclib.GMotionComplete(), and vector().
Referenced by Vector_Mode_Example.Main().

9.3.3 Field Documentation

9.3.3.1 GALIL_EXAMPLE_ERROR

```
const int GALIL_EXAMPLE_ERROR = -100  [static]
```

Examples error code.
Definition at line 30 of file examples.cs.
Referenced by Examples.Contour(), Examples.IP_Assigner(), Remote_Client_Example.Main(), Commands_Example.Main(), Contour_Example.Main(), IP_Assigner_Example.Main(), Jog_Example.Main(), Message_Example.Main(), Motion_Complete_Example.Main(), Position_Tracking_Example.Main(), Record_Position_Example.Main(), Remote_Server_Example.Main(), Vector_Mode_Example.Main(), and Examples.Motion_Complete().

9.3.3.2 GALIL_EXAMPLE_OK

```
const int GALIL_EXAMPLE_OK = 0  [static]
```

Examples success code.
Definition at line 29 of file examples.cs.
Referenced by Examples.Commands(), Examples.Contour(), Examples.IP_Assigner(), Examples.Jog(), Remote_Client_Example.Main(), Commands_Example.Main(), Contour_Example.Main(), IP_Assigner_Example.Main(), Jog_Example.Main(), Message_Example.Main(), Motion_Complete_Example.Main(), Position_Tracking_Example.Main(),

Record_Position_Example.Main(), Remote_Server_Example.Main(), Vector_Mode_Example.Main(), Examples.Message(), Examples.Motion_Complete(), Examples.Position_Tracking(), Examples.Record_Position(), Examples.Remote_Client(), Examples.Remote_Server(), and Examples.Vector_Mode().
The documentation for this class was generated from the following files:

- commands.cs
- examples.cs
- contour.cs
- ip_assigner.cs
- jog.cs
- message.cs
- motion_complete.cs
- position_tracking.cs
- record_position.cs
- Remote_Client.cs
- Remote_Server.cs
- vector_mode.cs

9.4 Galil Class Reference

Public Member Functions

- Galil (std::string address="")
- std::string connection ()
- std::string command (const std::string &command="MG TIME", const std::string &terminator="\r", const std::string &ack=":", bool trim=true)
- double commandValue (const std::string &command="MG TIME")
- std::string message (int timeout_ms=500)
- int interrupt (int timeout_ms=500)
- std::string programUpload ()
- void programDownload (const std::string &program="MG TIME\rEN")
- void programUploadFile (const std::string &file="program.dmc")
- void programDownloadFile (const std::string &file="program.dmc")
- std::vector< double > arrayUpload (const std::string &name="array")
- void arrayDownload (const std::vector< double > &array, const std::string &name="array")
- void arrayUploadFile (const std::string &file="arrays.csv", const std::string &names="")
- void arrayDownloadFile (const std::string &file="arrays.csv")
- void firmwareDownloadFile (const std::string &file="firmware.hex")
- int write (const std::string &bytes="\r")
- std::string read ()
- std::vector< std::string > sources ()
- void recordsStart (double period_ms=-1)
- std::vector< char > record (const std::string &method="QR")
- double sourceValue (const std::vector< char > &record, const std::string &source="TIME")
- std::string source (const std::string &field="Description", const std::string &source="TIME")
- void setSource (const std::string &field="Description", const std::string &source="TIME", const std::string &to="Sample counter")

Static Public Member Functions

- static std::string libraryVersion ()
- static std::vector< std::string > addresses ()

Data Fields

- int timeout_ms

9.4.1 Detailed Description

Definition at line 25 of file Galil.h.

9.4.2 Constructor & Destructor Documentation

9.4.2.1 Galil()

```
Galil (
            std::string address = "" )
```

Definition at line 186 of file gcl_galil.cpp.

9.4.2.2 ~Galil()

```
~Galil ( )
```

Definition at line 195 of file gcl_galil.cpp.

9.4.3 Member Function Documentation

9.4.3.1 addresses()

```
vector< string > addresses ( )  [static]
```

Definition at line 151 of file gcl_galil.cpp.

9.4.3.2 arrayDownload()

```
void arrayDownload (
            const std::vector< double > & array,
            const std::string & name = "array" )
```

Definition at line 327 of file gcl_galil.cpp.

9.4.3.3 arrayDownloadFile()

```
void arrayDownloadFile (
            const std::string & file = "arrays.csv" )
```

Definition at line 355 of file gcl_galil.cpp.

9.4.3.4 arrayUpload()

```
vector< double > arrayUpload (
            const std::string & name = "array" )
```

Definition at line 306 of file gcl_galil.cpp.

9.4.3.5 arrayUploadFile()

```
void arrayUploadFile (
            const std::string & file = "arrays.csv",
            const std::string & names = "" )
```

Definition at line 348 of file gcl_galil.cpp.

9.4.3.6 command()

```
string command (
            const std::string & command = "MG TIME",
            const std::string & terminator = "\r",
            const std::string & ack = ":",
            bool trim = true )
```

Definition at line 207 of file gcl_galil.cpp.

9.4.3.7 commandValue()

```
double commandValue (
            const std::string & command = "MG TIME" )
```
Definition at line 229 of file gcl_galil.cpp.

9.4.3.8 connection()

```
string connection ( )
```
Definition at line 201 of file gcl_galil.cpp.

9.4.3.9 firmwareDownloadFile()

```
void firmwareDownloadFile (
            const std::string & file = "firmware.hex" )
```
Definition at line 362 of file gcl_galil.cpp.

9.4.3.10 interrupt()

```
int interrupt (
            int timeout_ms = 500 )
```
Definition at line 267 of file gcl_galil.cpp.

9.4.3.11 libraryVersion()

```
string libraryVersion ( )  [static]
```
Definition at line 144 of file gcl_galil.cpp.

9.4.3.12 message()

```
string message (
            int timeout_ms = 500 )
```
Definition at line 238 of file gcl_galil.cpp.

9.4.3.13 programDownload()

```
void programDownload (
            const std::string & program = "MG TIME\rEN" )
```
Definition at line 285 of file gcl_galil.cpp.

9.4.3.14 programDownloadFile()

```
void programDownloadFile (
            const std::string & file = "program.dmc" )
```
Definition at line 299 of file gcl_galil.cpp.

9.4.3.15 programUpload()

```
string programUpload ( )
```
Definition at line 277 of file gcl_galil.cpp.

9.4.3.16 programUploadFile()

```
void programUploadFile (
            const std::string & file = "program.dmc" )
```
Definition at line 292 of file gcl_galil.cpp.

9.4.3.17 read()

```
string read ( )
```
Definition at line 377 of file gcl_galil.cpp.

9.4.3.18 record()

```
vector< char > record (
            const std::string & method = "QR" )
```

Definition at line 23 of file gcl_datarecord.cpp.

9.4.3.19 recordsStart()

```
void recordsStart (
            double period_ms = -1 )
```

Definition at line 18 of file gcl_datarecord.cpp.

9.4.3.20 setSource()

```
void setSource (
            const std::string & field = "Description",
            const std::string & source = "TIME",
            const std::string & to = "Sample counter" )
```

Definition at line 100 of file gcl_datarecord.cpp.

9.4.3.21 source()

```
string source (
            const std::string & field = "Description",
            const std::string & source = "TIME" )
```

Definition at line 73 of file gcl_datarecord.cpp.

9.4.3.22 sources()

```
vector< std::string > sources ( )
```

Definition at line 7 of file gcl_datarecord.cpp.

9.4.3.23 sourceValue()

```
double sourceValue (
            const std::vector< char > & record,
            const std::string & source = "TIME" )
```

Definition at line 34 of file gcl_datarecord.cpp.

9.4.3.24 write()

```
int write (
            const std::string & bytes = "\r" )
```

Definition at line 369 of file gcl_galil.cpp.

9.4.4 Field Documentation

9.4.4.1 timeout_ms

```
int timeout_ms
```

Definition at line 36 of file Galil.h.

The documentation for this class was generated from the following files:

- Galil.h
- gcl_datarecord.cpp
- gcl_galil.cpp

9.5 GalilPrivate Class Reference

Public Member Functions

- GalilPrivate (Galil *galil_ptr, GCon gclib_handle)
- void InitializeDataRecord ()

Data Fields

- GCon g
- char tbuf [TRAFFICBUF]
- std::unordered_map< std::string, Source > map

9.5.1 Detailed Description

Definition at line 32 of file gcl_galil.h.

9.5.2 Constructor & Destructor Documentation

9.5.2.1 GalilPrivate()

```
GalilPrivate (
            Galil * galil_ptr,
            GCon gclib_handle )  [inline]
```

Definition at line 35 of file gcl_galil.h.

9.5.3 Member Function Documentation

9.5.3.1 InitializeDataRecord()

```
void InitializeDataRecord ( )
```

Definition at line 138 of file gcl_datarecord.cpp.

9.5.4 Field Documentation

9.5.4.1 g

```
GCon g
```

Definition at line 39 of file gcl_galil.h.

9.5.4.2 map

```
std::unordered_map<std::string, Source> map
```

Definition at line 41 of file gcl_galil.h.

9.5.4.3 tbuf

```
char tbuf[TRAFFICBUF]
```

Definition at line 40 of file gcl_galil.h.

The documentation for this class was generated from the following files:

- gcl_galil.h
- gcl_datarecord.cpp

9.6 gclib Class Reference

Provides a class that binds to gclib's unmanaged dll. Wraps each call and provides a more user-friendly interface for use in C#.

Public Member Functions

- gclib ()

  Constructor of the gclib wrapper class.
- string[] GAddresses ()

  Return a string array of available connection addresses.
- void GArrayDownload (string array_name, ref List< double > data, Int16 first=-1, Int16 last=-1)

  Downloads array data to a pre-dimensioned array in the controller's array table.
- void GArrayDownloadFile (string Path)

  Allows downloading of a program array file to the controller.
- List< double > GArrayUpload (string array_name, Int16 first=-1, Int16 last=-1)

  Uploads array data from the controller's array table.
- void GArrayUploadFile (string Path, string Names)

  Allows uploading of a program array file from the controller to an array CSV file.
- void GAssign (string ip, string mac)

  Assigns IP address over the Ethernet to a controller at a given MAC address.
- void GClose ()

  Used to close a connection to Galil hardware.
- string GCommand (string Command, bool Trim=true)

  Used for command-and-response transactions.
- Int16 GCmdI (string Command)

  Used for command-and-response transactions.
- double GCmdD (string Command)

  Used for command-and-response transactions.
- void GFirmwareDownload (string filepath)

  Upgrade firmware.
- string GInfo ()

  Provides a useful connection string.
- byte GInterrupt ()

  Provides access to PCI and UDP interrupts from the controller.
- string[] GIpRequests ()

  Provides a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.
- string GMessage ()

  Provides access to unsolicited messages.
- void GMotionComplete (string axes)

  Blocking call that returns once all axes specified have completed their motion.
- void GOpen (string address)

  Used to open a connection to Galil hardware.
- void GProgramDownload (string program, string preprocessor="")

  Allows downloading of a DMC program from a string buffer.
- void GProgramDownloadFile (string file_path, string preprocessor="")

  Allows downloading of a DMC program from file.
- string GProgramUpload ()

  Allows uploading of a DMC program to a string.
- void GProgramUploadFile (string file_path)

  Allows uploading of a DMC program to a file.
- byte[] GRead ()

  Performs a read on the connection.
- T GRecord< T > (bool async)

  Used for retrieving data records from the controller.
- void GRecordRate (double period_ms)

Sets the asynchronous data record to a user-specified period via DR.

- void GTimeout (Int16 timeout_ms)

  Set the timeout of communication transactions. Use -1 to set the original timeout from GOpen().

- string GVersion ()

  Used to get the gclib version.

- void GWrite (string buffer)

  Performs a write on the connection.

- string[] GSetupDownloadFile (string path, Int32 options)

  Allows downloading of a Galil compressed backup (gcb) file to the controller.

- void GSetServer (string server_name)

  Connects gclib to a new gcaps server.

- string GServerStatus ()

  Retrieves the name of your local gcaps server and whether or not it is currently published.

- string[] GListServers ()

  Retrieves a list of gcaps servers that are advertising themselves on the local network.

- void GPublishServer (string server_name, bool publish, bool save)

  Publishes or removes local gcaps server from the network.

- string[] GRemoteConnections ()

  Returns a list of IP Addresses that currently have an open connection to your hardware.

Data Fields

- argtypes
- restype

Protected Attributes

- str _gclib_path = r'C:\Program Files (x86)\Galil\gclib\dll\x64\gclib.dll'
- str _gclibo_path = r'C:\Program Files (x86)\Galil\gclib\dll\x64\gclibo.dll'
- _gclib = WinDLL(_gclib_path)
- _gclibo = WinDLL(_gclibo_path)
- _GReturn = c_int
- _GCon = c_void_p
- _GCon_ptr = POINTER(_GCon)
- _GSize = c_ulong
- _GSize_ptr = POINTER(_GSize)
- _GCStringIn = c_char_p
- _GCStringOut = c_char_p
- _GOption = c_int
- _GStatus = c_ubyte
- _GStatus_ptr = POINTER(_GStatus)
- str _enc = "ASCII"
- int _buf_size = 500000
- _error_buf = create_string_buffer(128)

9.6.1 Detailed Description

Provides a class that binds to gclib's unmanaged dll. Wraps each call and provides a more user-friendly interface for use in C#.
The Gclib class assumes the default installation of gclib, "C:\Program Files (x86)\Galil\gclib\". If the dlls are elsewhere, change the path strings GclibDllPath_, and GcliboDllPath_.
Definition at line 67 of file gclib.cs.

9.6.2 Constructor & Destructor Documentation

9.6.2.1 gclib()

```
gclib ( )  [inline]
```

Constructor of the gclib wrapper class.
Checks to ensure gclib dlls are in the correct location.

Exceptions

System.Exception	Will throw an exception if either dll isn't found.

Definition at line 85 of file gclib.cs.
References vector().

9.6.3 Member Function Documentation

9.6.3.1 GAddresses()

```
string[] GAddresses ( )  [inline]
```

Return a string array of available connection addresses.

Returns

String array containing all available Galil Ethernet controllers, PCI controllers, and COM ports.

Wrapper around gclib GAddresses(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a6a6114683ed5749519b64f19512c24d6 An empty array is returned on error.
Definition at line 102 of file gclib.cs.
References G_NO_ERROR, and vector().
Referenced by Examples.Remote_Client().

9.6.3.2 GArrayDownload()

```
void GArrayDownload (
            string array_name,
            ref List< double > data,
            Int16 first = -1,
            Int16 last = -1 )  [inline]
```

Downloads array data to a pre-dimensioned array in the controller's array table.

Parameters

array_name	String containing the name of the array to download. Must match the array name used in DM.
data	A list of doubles, to be downloaded.
first	The first element of the array for sub-array downloads.
last	The last element of the array for sub-array downloads.

Wrapper around gclib GArrayDownload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a6ea5ae6d167675e4c27ccfaf2f240f8a The array must already exist on the controller, see DM and LA.

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 126 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.3 GArrayDownloadFile()

```
void GArrayDownloadFile (
            string Path )  [inline]
```

Allows downloading of a program array file to the controller.

Parameters

Path	The full filepath of the array csv file.

Wrapper around gclib GArrayDownload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a14b448ab8c7e6cf495865af301be398e

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 153 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.4 GArrayUpload()

```
List< double > GArrayUpload (
            string array_name,
            Int16 first = -1,
            Int16 last = -1 )  [inline]
```

Uploads array data from the controller's array table.

Parameters

array_name	String containing the name of the array to upload.
first	The first element of the array for sub-array uploads.
last	The last element of the array for sub-array uploads.

Returns

The desired array as a list of doubles.

Wrapper around gclib GArrayUpload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#af215806ec26ba06ed3f174ebeeafa7a7

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 173 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.5 GArrayUploadFile()

```
void GArrayUploadFile (
            string Path,
            string Names )  [inline]
```

Allows uploading of a program array file from the controller to an array CSV file.

Parameters

Path	The full filepath of the array csv file to save.

Parameters

Names	A space separated list of the array names to upload. A null string uploads all arrays in the array table (LA).

Wrapper around gclib GArrayUpload(). http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#af215806ec26ba06ed3f174ebeeafa7a7

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 206 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.6 GAssign()

```
void GAssign (
            string ip,
            string mac )  [inline]
```

Assigns IP address over the Ethernet to a controller at a given MAC address.

Parameters

ip	The ip address to assign. The hardware should not yet have an IP address.
mac	The MAC address of the hardware.

Wrapper around gclib GAssign(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#acc996b7c22cfed8e5573d096ef1ab759

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 224 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().
Referenced by Examples.IP_Assigner().

9.6.3.7 GClose()

```
void GClose ( )  [inline]
```

Used to close a connection to Galil hardware.
Wrapper around gclib GClose(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a24a437bcde9637b0db4b94176563a052 Be sure to call GClose() whenever a connection is finished.
Definition at line 239 of file gclib.cs.
Referenced by py.__del__(), Commands_Example.Main(), Contour_Example.Main(), IP_Assigner_Example.Main(), Jog_Example.Main(), Message_Example.Main(), Motion_Complete_Example.Main(), Position_Tracking_Example.Main(), Record_Position_Example.Main(), and Vector_Mode_Example.Main().

9.6.3.8 GCmdD()

```
double GCmdD (
            string Command )  [inline]
```

Used for command-and-response transactions.

Parameters

Command	The command to send to the controller. Do not append a carriage return. Use only ASCII-based commmands.

Returns

The command's response parsed as a double.

Wrapper around gclib GCmdD(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5ac031e76efc965affdd73a1bec084a8
Definition at line 301 of file gclib.cs.
References GCommand(), and vector().
Referenced by Examples.Commands().

9.6.3.9 GCmdI()

```
Int16 GCmdI (
            string Command ) [inline]
```

Used for command-and-response transactions.

Parameters

Command	The command to send to the controller. Do not append a carriage return. Use only ASCII-based commmands.

Returns

The command's response parsed as an integer.

Wrapper around gclib GCmdI(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5ac031e76efc965affdd73a1bec084a8
Definition at line 288 of file gclib.cs.
References GCommand(), and vector().
Referenced by Examples.Commands(), Examples.Contour(), Examples.Record_Position(), and Examples.Vector_Mode().

9.6.3.10 GCommand()

```
string GCommand (
            string Command,
            bool Trim = true ) [inline]
```

Used for command-and-response transactions.

Parameters

Command	The command to send to the controller. Do not append a carriage return. Use only ASCII-based commmands.
Trim	If true, the response will be trimmed of the trailing colon and any leading or trailing whitespace.

Returns

The command's response.

Wrapper around gclib GCommand(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5ac031e76efc965affdd73a1bec084a8

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 257 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().
Referenced by Examples.Commands(), Examples.Contour(), Examples.IP_Assigner(), Examples.Jog(), Examples.Message(), Examples.Motion_Complete(), Examples.Position_Tracking(), Examples.PrintError(), Examples.Record_Position(), and Examples.Vector_Mode().

9.6.3.11 GFirmwareDownload()

```
void GFirmwareDownload (
            string filepath )  [inline]
```

Upgrade firmware.

Parameters

filepath	The full filepath of the firmware hex file.

Wrapper around gclib GFirmwareDownload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a1878a2285ff17897fa4fb20182ba6fdf

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 330 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.12 GInfo()

```
string GInfo ( )  [inline]
```

Provides a useful connection string.
Wrapper around gclib GInfo(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a08abfcff8a1a85a01987859473167518

Returns

String containing connection information, e.g. "192.168.0.43, DMC4020 Rev 1.2c, 291". A null string indicates an error was returned from the library.

Definition at line 344 of file gclib.cs.
References G_NO_ERROR, and vector().
Referenced by Examples.IP_Assigner().

9.6.3.13 GInterrupt()

```
byte GInterrupt ( )  [inline]
```

Provides access to PCI and UDP interrupts from the controller.

Returns

The status byte from the controller. Zero will be returned if a status byte is not read.

Wrapper around gclib GInterrupt(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5bcf802404a96343e7593d247b67f132 -s ALL or -s EI must be specified in the address argument of GOpen() to receive interrupts.
Definition at line 364 of file gclib.cs.
References G_NO_ERROR, and vector().
Referenced by Examples.Motion_Complete().

9.6.3.14 GIpRequests()

```
string[] GIpRequests ( )  [inline]
```

Provides a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.

Returns

Each line of the returned data will be of the form "model, serial_number, mac".

Wrapper around gclib GIpRequests(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a0afb4c82642a4ef86f997c39a5518952 An empty array is returned on error. Call will take roughly 5 seconds to return.
Definition at line 386 of file gclib.cs.
References G_NO_ERROR, and vector().
Referenced by Examples.IP_Assigner().

9.6.3.15 GListServers()

```
string[] GListServers ( )  [inline]
```

Retrieves a list of gcaps servers that are advertising themselves on the local network.

Returns

A list of available gcaps server names

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 716 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().
Referenced by Examples.Remote_Client().

9.6.3.16 GMessage()

```
string GMessage ( )  [inline]
```

Provides access to unsolicited messages.

Returns

String containing all messages received by controller.

Wrapper around gclib GMessage(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#aabc5eaa09ddeca55ab8ee048b916cbcd An empty string is returned on error. -s ALL or -s MG must be specified in the address argument of GOpen() to receive messages.
Definition at line 407 of file gclib.cs.
References G_NO_ERROR, and vector().
Referenced by Examples.Message().

9.6.3.17 GMotionComplete()

```
void GMotionComplete (
            string axes )  [inline]
```

Blocking call that returns once all axes specified have completed their motion.

Parameters

axes	A string containing a multiple-axes mask. Every character in the string should be a valid argument to MG_BGm, i.e. XYZWABCDEFGHST.

Wrapper around gclib GMotionComplete(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a19c22087944298797070644419 7f397a

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 428 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().
Referenced by Examples.Contour(), Examples.Jog(), Examples.Position_Tracking(), and Examples.Vector_Mode().

9.6.3.18 GOpen()

```
void GOpen (
            string address )  [inline]
```

Used to open a connection to Galil hardware.

Parameters

address	Address string including any connection switches. See gclib documentation for GOpen().

Wrapper around gclib GOpen(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#aef4aec8a85630eed029b7a46aea7db54

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 445 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().
Referenced by Examples.IP_Assigner(), Commands_Example.Main(), Contour_Example.Main(), Jog_Example.Main(), Message_Example.Main(), Motion_Complete_Example.Main(), Position_Tracking_Example.Main(), Record_Position_Example.Main() and Vector_Mode_Example.Main().

9.6.3.19 GProgramDownload()

```
void GProgramDownload (
            string program,
            string preprocessor = "" )  [inline]
```

Allows downloading of a DMC program from a string buffer.

Parameters

program	The program to download.
preprocessor	Preprocessor directives. Use nullstring for none.

Wrapper around gclib GProgramDownload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#acafe19b2dd0537ff458e3c8afe3acfeb

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 465 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().
Referenced by Examples.Message(), and Examples.Record_Position().

9.6.3.20 GProgramDownloadFile()

```
void GProgramDownloadFile (
            string file_path,
            string preprocessor = "" )  [inline]
```

Allows downloading of a DMC program from file.

Parameters

file_path	The full filepath of the DMC file.
preprocessor	Preprocessor directives. Use nullstring for none.

Wrapper around gclib GProgramDownloadFile(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a8e44e2e321df9e7b8c538bf2d640633f

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 483 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.21 GProgramUpload()

```
string GProgramUpload ( )  [inline]
```

Allows uploading of a DMC program to a string.
Wrapper around gclib GProgramUpload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a80a653ce387a2bd16bde2793c6de77e9

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 499 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.22 GProgramUploadFile()

```
void GProgramUploadFile (
            string file_path )  [inline]
```

Allows uploading of a DMC program to a file.

Parameters

file_path	The full filepath of the DMC file to save.

Wrapper around gclib GProgramUploadFile(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a38c5565afc11762fa19d37fbaa3c9aa3

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 520 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.23 GPublishServer()

```
void GPublishServer (
            string server_name,
            bool publish,
            bool save )  [inline]
```

Publishes or removes local gcaps server from the network.

Parameters

server_name	Name to publish server under.
publish	True=publish server, False=remove server.
save	Save this configuration for future server reboots.

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 738 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().
Referenced by Examples.Remote_Server().

9.6.3.24 GRead()

```
byte[] GRead ( )  [inline]
```

Performs a read on the connection.

Returns

String containing the read data, or a nullstring if nothing was read or an error occured.

Wrapper around gclib GRead(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#adab6ec79b7e1bc7f0266684dd3434923
Definition at line 536 of file gclib.cs.
References G_NO_ERROR, and vector().

9.6.3.25 GRecord< T >()

```
T GRecord< T > (
            bool async )  [inline]
```

Used for retrieving data records from the controller.

Returns

A struct containing the information from the retrieved data record.

Parameters

async	False to user QR, True to use DR.

Wrapper around gclib GRecord(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a1f39cd57dcfa55d065c972a020b1f8ee To use async, -s ALL or -s DR must be specified in the address argument of GOpen(), and the records must be started via DR or RecordRate().

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Type Constraints

T* : *struct

T* : *GDataRecord

Definition at line 566 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.26 GRecordRate()

```
void GRecordRate (
            double period_ms )  [inline]
```

Sets the asynchronous data record to a user-specified period via DR.

Parameters

period_ms	Period, in milliseconds, to set up for the asynchronous data record.

Wrapper around gclib GRecordRate(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#ada86dc9d33ac961412583881963a1b8a Takes TM and product type into account and sets the DR period to the period requested by the user, if possible.

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 588 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.27 GRemoteConnections()

```
string[] GRemoteConnections ( )  [inline]
```

Returns a list of IP Addresses that currently have an open connection to your hardware.

Returns

Returns a list of IP Addresses that currently have an open connection to your hardware.

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 751 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.28 GServerStatus()

```
string GServerStatus ( )  [inline]
```

Retrieves the name of your local gcaps server and whether or not it is currently published.

Returns

A string in the form "<server_name>, <isPublished>"

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 701 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.29 GSetServer()

```
void GSetServer (
            string server_name )  [inline]
```

Connects gclib to a new gcaps server.

Parameters

server_name	Name of the server to connect.

Wrapper around gclib GSetServer(), Call GSetServer("Local") to connect gclib back to local gcaps server

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 686 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().
Referenced by Examples.Remote_Client().

9.6.3.30 GSetupDownloadFile()

```
string[] GSetupDownloadFile (
            string path,
            Int32 options )  [inline]
```

Allows downloading of a Galil compressed backup (gcb) file to the controller.

Parameters

path	The full filepath of the gcb file.
options	A bit mask indicating which sectors of the gcb file to restore to the controller.

Returns

The controller information stored in the gcb file.

Wrapper around gclib GSetupDownloadFile(),
If options is specified as 0, the return string will have a number appended corresponding to a bit mask of the available gcb sectors

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 655 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.3.31 GTimeout()

```
void GTimeout (
            Int16 timeout_ms )  [inline]
```

Set the timeout of communication transactions. Use -1 to set the original timeout from GOpen().

Parameters

timeout_ms	New timeout in miliseconds.

Wrapper around gclib GTimeout(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a179aa2d1b8e2227944cc06a7ceaf5640
Definition at line 604 of file gclib.cs.
Referenced by Examples.Motion_Complete(), and py.timeout().

9.6.3.32 GVersion()

```
string GVersion ( )  [inline]
```
Used to get the gclib version.

Returns

The library version, e.g. "104.73.179". A null string indicates an error was returned from the library.

Wrapper around gclib GVersion(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a1784b39416b77af20efc98a05f8ce475
Definition at line 614 of file gclib.cs.
References G_NO_ERROR, and vector().

9.6.3.33 GWrite()

```
void GWrite (
            string buffer )  [inline]
```
Performs a write on the connection.

Parameters

buffer	The user's write buffer. To send a Galil command, a terminating carriage return is usually required.

Wrapper around gclib GWrite(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#abe28ebaecd5b3940adf4e145d40e5456

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

Definition at line 635 of file gclib.cs.
References G_NO_ERROR, GError(), and vector().

9.6.4 Field Documentation

9.6.4.1 _buf_size

```
int _buf_size = 500000  [protected]
```
Definition at line 127 of file gclib.py.

9.6.4.2 _enc

```
str _enc = "ASCII"  [protected]
```
Definition at line 126 of file gclib.py.

9.6.4.3 _error_buf

```
_error_buf = create_string_buffer(128)  [protected]
```
Definition at line 128 of file gclib.py.

9.6.4.4 _gclib

```
_gclib = WinDLL(_gclib_path)  [protected]
```
Definition at line 20 of file gclib.py.

9.6.4.5 _gclib_path

```
str _gclib_path = r'C:\Program Files (x86)\Galil\gclib\dll\x64\gclib.dll'  [protected]
```
Definition at line 18 of file gclib.py.

9.6.4.6 _gclibo

```
_gclibo = WinDLL(_gclibo_path)  [protected]
```
Definition at line 21 of file gclib.py.

9.6.4.7 _gclibo_path

```
str _gclibo_path = r'C:\Program Files (x86)\Galil\gclib\dll\x64\gclibo.dll'  [protected]
```
Definition at line 19 of file gclib.py.

9.6.4.8 _GCon

```
_GCon = c_void_p  [protected]
```
Definition at line 80 of file gclib.py.

9.6.4.9 _GCon_ptr

```
_GCon_ptr = POINTER(_GCon)  [protected]
```
Definition at line 81 of file gclib.py.

9.6.4.10 _GCStringIn

```
_GCStringIn = c_char_p  [protected]
```
Definition at line 84 of file gclib.py.

9.6.4.11 _GCStringOut

```
_GCStringOut = c_char_p  [protected]
```
Definition at line 85 of file gclib.py.

9.6.4.12 _GOption

```
_GOption = c_int  [protected]
```
Definition at line 86 of file gclib.py.

9.6.4.13 _GReturn

```
_GReturn = c_int  [protected]
```
Definition at line 79 of file gclib.py.

9.6.4.14 _GSize

```
_GSize = c_ulong  [protected]
```
Definition at line 82 of file gclib.py.

9.6.4.15 _GSize_ptr

```
_GSize_ptr = POINTER(_GSize)  [protected]
```
Definition at line 83 of file gclib.py.

9.6.4.16 _GStatus

```
_GStatus = c_ubyte  [protected]
```
Definition at line 87 of file gclib.py.

9.6.4.17 _GStatus_ptr

```
_GStatus_ptr = POINTER(_GStatus)  [protected]
```
Definition at line 88 of file gclib.py.

9.6.4.18 argtypes

`argtypes`
Definition at line 92 of file gclib.py.

9.6.4.19 restype

`restype`
Definition at line 108 of file gclib.py.
The documentation for this class was generated from the following files:

- gclib.cs
- gclib.py

9.7 Gclib Class Reference

Provides a class that binds to gclib's unmanaged dll. Wraps each call and provides a more user-friendly interface for use in Visual Basic.

Data Structures

- interface GDataRecord
- struct GDataRecord1802
- struct GDataRecord1806
- struct GDataRecord2103
- struct GDataRecord30000
- struct GDataRecord4000
- struct GDataRecord47000_ENC
- struct GDataRecord47162
- struct GDataRecord47300_24EX
- struct GDataRecord47300_ENC
- struct GDataRecord52000

Public Member Functions

- void New ()

  Constructor of the gclib wrapper class.
- GAddresses ()

  Return a string array of available connection addresses.
- void GArrayDownload (string array_name, ref List< double > data, Int16 first=-1, Int16 last=-1)

  Downloads array data to a pre-dimensioned array in the controller's array table.
- void GArrayDownloadFile (string Path)

  Allows downloading of a program array file to the controller.
- List< double > GArrayUpload (string array_name, Int16 first=-1, Int16 last=-1)

  Uploads array data from the controller's array table.
- void GArrayUploadFile (string Path, string Names)

  Allows uploading of a program array file from the controller to an array CSV file.
- void GAssign (string ip, string mac)

  Assigns IP address over the Ethernet to a controller at a given MAC address.
- void GClose ()

  Used to close a connection to Galil hardware.
- string GCommand (string Command, bool Trim=True)

  Used for command-and-response transactions.
- Int16 GCmdI (string Command)

  Used for command-And-response transactions.

- double GCmdD (string Command)

  Used for command-And-response transactions.
- void GFirmwareDownload (string filepath)

  Upgrade firmware.
- string GInfo ()

  Provides a useful connection string.
- byte GInterrupt ()

  Provides access to PCI and UDP interrupts from the controller.
- GIpRequests ()

  Provides a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.
- string GMessage ()

  Provides access to unsolicited messages.
- void GMotionComplete (string axes)

  Blocking call that returns once all axes specified have completed their motion.
- void GOpen (string address)

  Used to open a connection to Galil hardware.
- void GProgramDownload (ref string program, string preprocessor="")

  Allows downloading of a DMC program from a string buffer.
- void GProgramDownloadFile (string file_path, string preprocessor="")

  Allows downloading of a DMC program from file.
- string GProgramUpload ()

  Allows uploading of a DMC program to a string.
- void GProgramUploadFile (string file_path)

  Allows uploading of a DMC program to a file.
- GRead ()

  Performs a read on the connection.
- T GRecord (Of GDataRecord)(async T bool)

  Used for retrieving data records from the controller.
- void GRecordRate (double period_ms)

  Sets the asynchronous data record to a user-specified period via DR.
- void GTimeout (Int16 timeout_ms)

  Set the timeout of communication transactions. Use -1 to set the original timeout from GOpen().
- string GVersion ()

  Used to get the gclib version.
- void GWrite (ref string buffer)

  Performs a write on the connection.
- string[] GSetupDownloadFile (string Path, Int32 Options)

  Allows downloading of a Galil compressed backup (gcb) file to the controller.
- void GSetServer (string server_name)

  Connects gclib to a New gcaps server.
- string GServerStatus ()

  Retrieves the name of your local gcaps server And whether Or Not it Is currently published.
- GListServers ()

  Retrieves a list of gcaps servers that are advertising themselves on the local network.
- void GPublishServer (string server_name, bool publish, bool save)

  Publishes Or removes local gcaps server from the network.
- GRemoteConnections ()

  Returns a list of IP Addresses that currently have an open connection to your hardware.

9.7.1 Detailed Description

Provides a class that binds to gclib's unmanaged dll. Wraps each call and provides a more user-friendly interface for use in Visual Basic.
The Gclib class assumes the default installation of gclib, "C:\Program Files (x86)\Galil\gclib\". If the dlls are elsewhere, change the path strings GclibDllPath_, and GcliboDllPath_.
Definition at line 44 of file gclib.vb.

9.7.2 Member Function Documentation

9.7.2.1 GAddresses()

```
GAddresses ( )
```

Return a string array of available connection addresses.

Returns

String array containing all available Galil Ethernet controllers, PCI controllers, and COM ports.

Wrapper around gclib GAddresses(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a6a6114683ed5749519b64f19512c24d6 An empty array is returned on error.

9.7.2.2 GArrayDownload()

```
void GArrayDownload (
            string array_name,
            ref List< double > data,
            Int16 first = -1,
            Int16 last = -1 )
```

Downloads array data to a pre-dimensioned array in the controller's array table.

Parameters

array_name	String containing the name of the array to download. Must match the array name used in DM.
data	A list of doubles, to be downloaded.
first	The first element of the array for sub-array downloads.
last	The last element of the array for sub-array downloads.

Wrapper around gclib GArrayDownload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a6ea5ae6d167675e4c27ccfaf2f240f8a The array must already exist on the controller, see DM and LA.

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.3 GArrayDownloadFile()

```
void GArrayDownloadFile (
            string Path )
```

Allows downloading of a program array file to the controller.

Parameters

Path	The full filepath of the array csv file.

Wrapper around gclib GArrayDownload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a14b448ab8c7e6cf495865af301be398e

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.4 GArrayUpload()

```
List< double > GArrayUpload (
            string array_name,
            Int16 first = -1,
            Int16 last = -1 )
```

Uploads array data from the controller's array table.

Parameters

array_name	String containing the name of the array to upload.
first	The first element of the array for sub-array uploads.
last	The last element of the array for sub-array uploads.

Returns

The desired array as a list of doubles.

Wrapper around gclib GArrayUpload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#af215806ec26ba06ed3f174ebeeafa7a7

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.5 GArrayUploadFile()

```
void GArrayUploadFile (
            string Path,
            string Names )
```

Allows uploading of a program array file from the controller to an array CSV file.

Parameters

Path	The full filepath of the array csv file to save.
Names	A space separated list of the array names to upload. A null string uploads all arrays in the array table (LA).

Wrapper around gclib GArrayUpload(). http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#af215806ec26ba06ed3f174ebeeafa7a7

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.6 GAssign()

```
void GAssign (
            string ip,
```

```
            string mac )
```

Assigns IP address over the Ethernet to a controller at a given MAC address.

Parameters

ip	The ip address to assign. The hardware should not yet have an IP address.
mac	The MAC address of the hardware.

Wrapper around gclib GAssign(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#acc996b7c22cfed8e5573d096ef1ab759

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.7 GClose()

```
void GClose ( )
```

Used to close a connection to Galil hardware.
Wrapper around gclib GClose(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a24a437bcde9637b0db4b94176563a052 Be sure to call GClose() whenever a connection is finished.
Referenced by py.__del__().

9.7.2.8 GCmdD()

```
double GCmdD (
            string Command )
```

Used for command-And-response transactions.

Parameters

Command	The command to send to the controller. Do Not append a carriage return. Use only ASCII-based commmands.

Returns

The command's response parsed as a double.

Wrapper around gclib GCmdD(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5ac031e76efc965affdd73a1bec084a8

9.7.2.9 GCmdI()

```
Int16 GCmdI (
            string Command )
```

Used for command-And-response transactions.

Parameters

Command	The command to send to the controller. Do Not append a carriage return. Use only ASCII-based commmands.

Returns

The command's response parsed as an integer.

Wrapper around gclib GCmdI(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5ac031e76efc965affdd73a1bec084a8

9.7.2.10 GCommand()

```
string GCommand (
            string Command,
            bool Trim = True )
```

Used for command-and-response transactions.

Parameters

Command	The command to send to the controller. Do not append a carriage return. Use only ASCII-based commmands.
Trim	If true, the response will be trimmed of the trailing colon and any leading or trailing whitespace.

Returns

The command's response.

Wrapper around gclib GCommand(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5ac031e76efc965affdd73a1bec084a8

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.11 GFirmwareDownload()

```
void GFirmwareDownload (
            string filepath )
```

Upgrade firmware.

Parameters

filepath	The full filepath of the firmware hex file.

Wrapper around gclib GFirmwareDownload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a1878a2285ff17897fa4fb20182ba6fdf

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.12 GInfo()

```
string GInfo ( )
```

Provides a useful connection string.
Wrapper around gclib GInfo(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a08abfcff8a1a85a01987859473167518

Returns

String containing connection information, e.g. "192.168.0.43, DMC4020 Rev 1.2c, 291". A null string indicates an error was returned from the library.

9.7.2.13 GInterrupt()

`byte GInterrupt ( )`
Provides access to PCI and UDP interrupts from the controller.

Returns

The status byte from the controller. Zero will be returned if a status byte is not read.

Wrapper around gclib GInterrupt(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5bcf802404a96343e7593d247b67f132 -s ALL or -s EI must be specified in the address argument of GOpen() to receive interrupts.

9.7.2.14 GIpRequests()

`GIpRequests ( )`
Provides a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.

Returns

Each line of the returned data will be of the form "model, serial_number, mac".

Wrapper around gclib GIpRequests(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a0afb4c82642a4ef86f997c39a5518952 An empty array is returned on error. Call will take roughly 5 seconds to return.

9.7.2.15 GListServers()

`GListServers ( )`
Retrieves a list of gcaps servers that are advertising themselves on the local network.

Returns

A list of available gcaps server names

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR Is received from gclib.

9.7.2.16 GMessage()

`string GMessage ( )`
Provides access to unsolicited messages.

Returns

String containing all messages received by controller.

Wrapper around gclib GMessage(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#aabc5eaa09ddeca55ab8ee048b916cbcd An empty string is returned on error. -s ALL or -s MG must be specified in the address argument of GOpen() to receive messages.

9.7.2.17 GMotionComplete()

```
void GMotionComplete (
            string axes )
```

Blocking call that returns once all axes specified have completed their motion.

Parameters

axes	A string containing a multiple-axes mask. Every character in the string should be a valid argument to MG_BGm, i.e. XYZWABCDEFGHST.

Wrapper around gclib GMotionComplete(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a19c22087944298797070644197f397a

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.18 GOpen()

```
void GOpen (
            string address )
```

Used to open a connection to Galil hardware.

Parameters

address	Address string including any connection switches. See gclib documentation for GOpen().

Wrapper around gclib GOpen(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#aef4aec8a85630eed029b7a46aea7db54

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.19 GProgramDownload()

```
void GProgramDownload (
            ref string program,
            string preprocessor = "" )
```

Allows downloading of a DMC program from a string buffer.

Parameters

program	The program to download.
preprocessor	Preprocessor directives. Use nullstring for none.

Wrapper around gclib GProgramDownload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#acafe19b2dd0537ff458e3c8afe3acfeb

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.20 GProgramDownloadFile()

```
void GProgramDownloadFile (
            string file_path,
            string preprocessor = "" )
```

Allows downloading of a DMC program from file.

Parameters

file_path	The full filepath of the DMC file.
preprocessor	Preprocessor directives. Use nullstring for none.

Wrapper around gclib GProgramDownloadFile(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a8e44e2e321df9e7b8c538bf2d640633f

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.21 GProgramUpload()

```
string GProgramUpload ( )
```

Allows uploading of a DMC program to a string.
Wrapper around gclib GProgramUpload(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a80a653ce387a2bd16bde2793c6de77e9

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.22 GProgramUploadFile()

```
void GProgramUploadFile (
            string file_path )
```

Allows uploading of a DMC program to a file.

Parameters

file_path	The full filepath of the DMC file to save.

Wrapper around gclib GProgramUploadFile(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a38c5565afc11762fa19d37fbaa3c9aa3

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.23 GPublishServer()

```
void GPublishServer (
            string server_name,
            bool publish,
            bool save )
```

Publishes Or removes local gcaps server from the network.

Parameters

server_name	Name to publish server under.
publish	True=publish server, False=remove server.

Parameters

save	Save this configuration for future server reboots.

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR Is received from gclib.

9.7.2.24 GRead()

```
GRead ( )
```

Performs a read on the connection.

Returns

String containing the read data, or a nullstring if nothing was read or an error occured.

Wrapper around gclib GRead(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#adab6ec79b7e1bc7f0266684dd3434923

9.7.2.25 GRecord()

```
T GRecord (
            Of GDataRecord )
```

Used for retrieving data records from the controller.

Returns

A struct containing the information of the retrieved data record.

Parameters

async	False to user QR, True to use DR.

Wrapper around gclib GRecord(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a1f39cd57dcfa55d065c972a020b1f8ee To use async, -s ALL or -s DR must be specified in the address argument of GOpen(), and the records must be started via DR or RecordRate().

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.26 GRecordRate()

```
void GRecordRate (
            double period_ms )
```

Sets the asynchronous data record to a user-specified period via DR.

Parameters

period_ms	Period, in milliseconds, to set up for the asynchronous data record.

Wrapper around gclib GRecordRate(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#ada86dc9d33ac961412583881963a1b8a Takes TM and product type into account and sets the DR period to the period requested by the user, if possible.

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.27 GRemoteConnections()

```
GRemoteConnections ( )
```

Returns a list of IP Addresses that currently have an open connection to your hardware.

Returns

Returns a list of IP Addresses that currently have an open connection to your hardware.

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR Is received from gclib.

9.7.2.28 GServerStatus()

```
string GServerStatus ( )
```

Retrieves the name of your local gcaps server And whether Or Not it Is currently published.

Returns

A string in the form "<server_name>, <isPublished>"

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR Is received from gclib.

9.7.2.29 GSetServer()

```
void GSetServer (
            string server_name )
```

Connects gclib to a New gcaps server.

Parameters

server_name	Name of the server to connect.

Wrapper around gclib GSetServer(), Call GSetServer("Local") to connect gclib back to local gcaps server

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR Is received from gclib.

9.7.2.30 GSetupDownloadFile()

```
string[] GSetupDownloadFile (
            string Path,
            Int32 Options )
```

Allows downloading of a Galil compressed backup (gcb) file to the controller.

Parameters

Path	The full filepath of the gcb file.
Options	A bit mask indicating which sectors of the gcb file to restore to the controller.

Returns

The controller information stored in the gcb file.

Wrapper around gclib GSetupDownloadFile(),
If options is specified as 0, the return string will have a number appended corresponding to a bit mask of the available gcb sectors

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.31 GTimeout()

```
void GTimeout (
            Int16 timeout_ms )
```

Set the timeout of communication transactions. Use -1 to set the original timeout from GOpen().

Parameters

timeout_ms	New timeout in miliseconds.

Wrapper around gclib GTimeout(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a179aa2d1b8e2227944cc06a7ceaf5640
Referenced by py.timeout().

9.7.2.32 GVersion()

```
string GVersion ( )
```

Used to get the gclib version.

Returns

The library version, e.g. "104.73.179". A null string indicates an error was returned from the library.

Wrapper around gclib GVersion(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a1784b39416b77af20efc98a05f8ce475

9.7.2.33 GWrite()

```
void GWrite (
            ref string buffer )
```

Performs a write on the connection.

Parameters

buffer	The user's write buffer. To send a Galil command, a terminating carriage return is usually required.

Wrapper around gclib GWrite(), http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#abe28ebaecd5b3940adf4e145d40e5456

Exceptions

System.Exception	Will throw an exception if anything other than G_NO_ERROR is received from gclib.

9.7.2.34 New()

```
void New ( )
```

Constructor of the gclib wrapper class.
Checks to ensure gclib dlls are in the correct location.

Exceptions

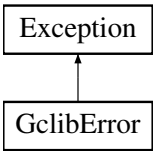
System.Exception	Will throw an exception if either dll isn't found.

The documentation for this class was generated from the following file:

- gclib.vb

9.8 GclibError Class Reference

Inheritance diagram for GclibError:



9.8.1 Detailed Description

```
Error class for non-zero gclib return codes.
```

Definition at line 137 of file gclib.py.
The documentation for this class was generated from the following file:

- gclib.py

9.9 GclibJava Class Reference

Data Structures

- interface **Gclib**
- interface **Gclibo**

Public Member Functions

- GclibJava ()
- void GArrayDownload (String arrayName, List< Double > data) throws GclibJavaException
- List< Double > GArrayUpload (String arrayName) throws GclibJavaException
- void GClose ()
- String GCommand (String command) throws GclibJavaException
- void GFirmwareDownload (String filePath) throws GclibJavaException
- byte GInterrupt () throws GclibJavaException
- String GMessage () throws GclibJavaException
- void GOpen (String address) throws GclibJavaException

- void GProgramDownload (String program, String preprocessor) throws GclibJavaException
- void GProgramDownload (String program) throws GclibJavaException
- String GProgramUpload () throws GclibJavaException
- String GAddresses () throws GclibJavaException
- void GArrayDownloadFile (String filePath) throws GclibJavaException
- void GArrayUploadFile (String filePath, String names) throws GclibJavaException
- void GArrayUploadFile (String filePath) throws GclibJavaException
- void GAssign (String ipAddress, String macAddress) throws GclibJavaException
- String GInfo () throws GclibJavaException
- String GIpRequests () throws GclibJavaException
- void GProgramDownloadFile (String filePath, String preprocessor) throws GclibJavaException
- void GProgramDownloadFile (String filePath) throws GclibJavaException
- void GProgramUploadFile (String filePath) throws GclibJavaException
- void GSleep (int timeout_ms)
- void GTimeout (short timeout_ms) throws GclibJavaException
- String GVersion () throws GclibJavaException
- void GSetServer (String server_name) throws GclibJavaException
- String GServerStatus () throws GclibJavaException
- String GListServers () throws GclibJavaException
- void GPublishServer (String server_name, int publish, int save) throws GclibJavaException
- String GRemoteConnections () throws GclibJavaException

Protected Member Functions

- void finalize () throws Throwable

9.9.1 Detailed Description

GclibJava uses Java Native Access (JNA) internally to wrap the gclib functions in a Java-callable class.
Definition at line 36 of file GclibJava.java.

9.9.2 Constructor & Destructor Documentation

9.9.2.1 GclibJava()

`GclibJava ( ) [inline]`
Constructor adds gclib to JNA's path.
Definition at line 45 of file GclibJava.java.

9.9.3 Member Function Documentation

9.9.3.1 finalize()

`void finalize ( ) throws Throwable [inline], [protected]`
The last line of defense to close connection. Do NOT rely on finalize(), call GClose() explicitly.

Exceptions

Throwable	super can throw.

Definition at line 57 of file GclibJava.java.
References GclibJava.GClose(), and vector().

9.9.3.2 GAddresses()

`String GAddresses ( ) throws GclibJavaException [inline]`
Uses GUtility(), G_UTIL_GCAPS_ADDRESSES or G_UTIL_ADDRESSES to provide a listing of all available connection addresses.

Returns

String containing the available addresses.

10.1.3.91, DMC4020 Rev 1.2e, LAN, 10.1.3.10 192.168.0.63, DMC4040 Rev 1.2f, Static, 192.168.0.41 (192.0.0.↩ 42), RIO47102 Rev 1.1j, Static, 192.168.0.41 10.1., RIO47102 Rev 1.1j, Static, 192.168.0.41 GALILPCI1 COM1 COM2

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 359 of file GclibJava.java.

9.9.3.3 GArrayDownload()

```
void GArrayDownload (
            String arrayName,
            List< Double > data ) throws GclibJavaException  [inline]
```

Downloads array data to a pre-dimensioned array in the controller's array table.

Parameters

arrayName	String containing the name of the array to download. Must match the array name used in DM.
data	List containing the array data. The length of data may not be larger than the array dimensioned.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 111 of file GclibJava.java.
References vector().

9.9.3.4 GArrayDownloadFile()

```
void GArrayDownloadFile (
            String filePath ) throws GclibJavaException  [inline]
```

Array download from file. Downloads a csv file containing array data at file_path. If the arrays don't exist, they will be dimensioned.

Parameters

filePath	String containing the path to the array file.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 373 of file GclibJava.java.
References vector().

9.9.3.5 GArrayUpload()

```
List< Double > GArrayUpload (
            String arrayName ) throws GclibJavaException  [inline]
```

Uploads array data from the controller's array table.

Parameters

arrayName	String containing the name of the array to upload.

Returns

A List of Doubles, containing the array data.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 129 of file GclibJava.java.
References e(), and vector().

9.9.3.6 GArrayUploadFile() [1/2]

```
void GArrayUploadFile (
            String filePath ) throws GclibJavaException  [inline]
```

Overload of GArrayUploadFile to upload all arrays.

Parameters

filePath	String containing the path to the array file. File will be overwritten if it exists.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 404 of file GclibJava.java.
References GclibJava.GArrayUploadFile(), and vector().

9.9.3.7 GArrayUploadFile() [2/2]

```
void GArrayUploadFile (
            String filePath,
            String names ) throws GclibJavaException  [inline]
```

Array upload to file. Uploads the entire controller array table or a subset and saves the data as a csv file specified by file_path.

Parameters

filePath	String containing the path to the array file. File will be overwritten if it exists.
names	String containing the arrays to upload, delimited with space. "" uploads all arrays listed in LA.

Exceptions

gclibjava.GclibJavaException	If an error is generated by gclib.

Definition at line 391 of file GclibJava.java.
References vector().
Referenced by GclibJava.GArrayUploadFile().

9.9.3.8 GAssign()

```
void GAssign (
            String ipAddress,
            String macAddress ) throws GclibJavaException  [inline]
```

Uses GUtility(), G_UTIL_GCAPS_ASSIGN or G_UTIL_ASSIGN to assign an IP address over the Ethernet to a controller at a given MAC address.

Parameters

ipAddress	The IP address to assign.
macAddress	The MAC address of the hardware.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 419 of file GclibJava.java.
References vector().

9.9.3.9 GClose()

```
void GClose ( )  [inline]
```

Closes a connection to a Galil Controller.
Definition at line 151 of file GclibJava.java.
Referenced by py.__del__(), GclibJava.finalize(), and GclibJava.GOpen().

9.9.3.10 GCommand()

```
String GCommand (
            String command ) throws GclibJavaException  [inline]
```

Performs a command-and-response transaction on the connection.

Parameters

command	command string to send to the controller. The library will append a carriage return to the command string.

Returns

The response from the controller.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 167 of file GclibJava.java.
References vector().

9.9.3.11 GFirmwareDownload()

```
void GFirmwareDownload (
            String filePath ) throws GclibJavaException  [inline]
```

Upgrade firmware.

Parameters

filePath	The full file path to the Galil-supplied firmware hex file. See http://www.galil.com/downloads/firmware

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 190 of file GclibJava.java.
References vector().

9.9.3.12 GInfo()

```
String GInfo ( ) throws GclibJavaException  [inline]
```

Uses GUtility() and G_UTIL_INFO to provide a useful connection string.

Returns

A String containing the info, e.g. 192.168.0.42, DMC30010 Rev 1.2i, 6969

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 432 of file GclibJava.java.

9.9.3.13 GInterrupt()

```
byte GInterrupt ( ) throws GclibJavaException  [inline]
```

Provides access to PCI and UDP interrupts from the controller.
Interrupts can be generated automatically by the firmware on important events via EI (Enable Interrupt) or by the user in embedded DMC code via UI (User Interrupt). To use this function, -s EI must be used in the GOpen() address string to subscribe to interrupts.

Returns

The status byte of the interrupt.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 207 of file GclibJava.java.
References vector().

9.9.3.14 GIpRequests()

```
String GIpRequests ( ) throws GclibJavaException  [inline]
```

Uses GUtility(), G_UTIL_GCAPS_IPREQUEST or G_UTIL_IPREQUEST to provide a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.

Returns

String containing hardware requesting IP addresses.

DMC4000, 291, 00:50:4C:20:01:23, LAN, 10.1.3.10 RIO47000, 37290, 00:50:4C:28:91:AA, Static, 192.168.0.41

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 449 of file GclibJava.java.

9.9.3.15 GListServers()

```
String GListServers ( ) throws GclibJavaException  [inline]
```

Retrieves a list of gcaps servers that are advertising themselves on the local network.

Returns

A list of available gcaps server names.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 568 of file GclibJava.java.

9.9.3.16 GMessage()

```
String GMessage ( ) throws GclibJavaException  [inline]
```

Provides access to unsolicited messages from the controller.
To use this function, -s MG must be used in the GOpen() address string to subscribe to messages. Unsolicited bytes must be flagged by the high-bit setting, CW 1. The driver will automatically set this when subscribing to messages. The user should not overwrite this setting.
Unsolicited messages are data generated by the controller that are not in response to a command, a data record, or an interrupt.
GMessage() will block until a message is received, or the function times out.
Messages are unframed byte streams. There is no guarantee that the user will get complete messages or single messages in a call to GMessage().

Returns

the message received.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 235 of file GclibJava.java.

9.9.3.17 GOpen()

```
void GOpen (
            String address ) throws GclibJavaException  [inline]
```

Open a connection to a Galil Controller.

Parameters

address	address string. See `gclib GOpen()`

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 250 of file GclibJava.java.
References GclibJava.GClose(), and vector().

9.9.3.18 GProgramDownload() [1/2]

```
void GProgramDownload (
            String program ) throws GclibJavaException  [inline]
```

Overload of GProgramDownload to use default preprocessor options.

Parameters

program	Program for download.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 282 of file GclibJava.java.
References GclibJava.GProgramDownload(), and vector().

9.9.3.19 GProgramDownload() [2/2]

```
void GProgramDownload (
            String program,
            String preprocessor ) throws GclibJavaException  [inline]
```

Downloads a program to the controller's program buffer.

Parameters

program	Program for download.
preprocessor	Options string for preprocessing the program before sending it to the controller.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 270 of file GclibJava.java.
References vector().
Referenced by GclibJava.GProgramDownload().

9.9.3.20 GProgramDownloadFile() [1/2]

```
void GProgramDownloadFile (
            String filePath ) throws GclibJavaException  [inline]
```

Overload of GProgramDownloadFile to use default preprocessor options.

Parameters

filePath	String containing the path to the program file.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 476 of file GclibJava.java.
References GclibJava.GProgramDownloadFile(), and vector().

9.9.3.21 GProgramDownloadFile() [2/2]

```
void GProgramDownloadFile (
            String filePath,
            String preprocessor ) throws GclibJavaException  [inline]
```

Program download from file.

Parameters

filePath	String containing the path to the program file.
preprocessor	Options string for preprocessing the program before sending it to the controller.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 464 of file GclibJava.java.
References vector().
Referenced by GclibJava.GProgramDownloadFile().

9.9.3.22 GProgramUpload()

```
String GProgramUpload ( ) throws GclibJavaException  [inline]
```

Uploads a program from the controller's program buffer.

Returns

The uploaded program.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 294 of file GclibJava.java.

9.9.3.23 GProgramUploadFile()

```
void GProgramUploadFile (
            String filePath ) throws GclibJavaException  [inline]
```

Program upload to file.

Parameters

filePath	String containing the path to the program file, file will be overwritten if it exists.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 489 of file GclibJava.java.
References vector().

9.9.3.24 GPublishServer()

```
void GPublishServer (
            String server_name,
            int publish,
            int save ) throws GclibJavaException  [inline]
```

Publishes or removes local gcaps server from the network

Parameters

server_name	Name to publish server under.
publish	True=publish server, False=remove server.
save	Save this configuration for future server reboots.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 583 of file GclibJava.java.
References vector().

9.9.3.25 GRemoteConnections()

```
String GRemoteConnections ( ) throws GclibJavaException  [inline]
```

Returns a list of IP Addresses that currently have an open connection to your hardware.

Returns

a list of IP Addresses that currently have an open connection to your hardware.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 595 of file GclibJava.java.

9.9.3.26 GServerStatus()

```
String GServerStatus ( ) throws GclibJavaException  [inline]
```

Retrieves the name of your local gcaps server and whether or not it is currently published Retrieves a list of gcaps servers that are advertising themselves on the local network.

Returns

A string in the form "<server_name>, <isPublished>"

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 555 of file GclibJava.java.

9.9.3.27 GSetServer()

```
void GSetServer (
            String server_name ) throws GclibJavaException  [inline]
```

Connects gclib to a new gcaps server

Parameters

server_name	Name to publish server under.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 542 of file GclibJava.java.
References vector().

9.9.3.28 GSleep()

```
void GSleep (
            int timeout_ms ) [inline]
```

Uses GUtility() and G_UTIL_SLEEP to provide a blocking sleep call which can be useful for timing-based chores. In GclibJava, this is primarily a debugging call.

Parameters

timeout_ms	Sleep time in milliseconds.

Definition at line 502 of file GclibJava.java.

9.9.3.29 GTimeout()

```
void GTimeout (
            short timeout_ms ) throws GclibJavaException [inline]
```

Uses GUtility() and G_UTIL_TIMEOUT_OVERRIDE to set the library timeout.

Parameters

timeout_ms	The value to be used for the timeout. Use -1 to set the timeout back to the initial GOpen() value, –timeout.

Exceptions

GclibJavaException	If an error is generated by gclib.

Definition at line 515 of file GclibJava.java.
Referenced by py.timeout().

9.9.3.30 GVersion()

```
String GVersion ( ) throws GclibJavaException [inline]
```

Uses GUtility(), G_UTIL_VERSION and G_UTIL_GCAPS_VERSION to provide the library and gcaps version numbers.

Returns

A String containing the version, e.g. 189.224.370 1.0.0.125

Exceptions

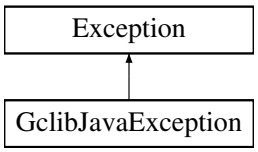
GclibJavaException	If an error is generated by gclib.

Definition at line 529 of file GclibJava.java.
The documentation for this class was generated from the following file:

- GclibJava.java

9.10 GclibJavaException Class Reference

Inheritance diagram for GclibJavaException:



Public Member Functions

- GclibJavaException (int errorCode, String message)
- int getErrorCode ()

9.10.1 Detailed Description

Definition at line 6 of file GclibJavaException.java.

9.10.2 Constructor & Destructor Documentation

9.10.2.1 GclibJavaException()

```
GclibJavaException (
            int errorCode,
            String message )  [inline]
```

Definition at line 8 of file GclibJavaException.java.

9.10.3 Member Function Documentation

9.10.3.1 getErrorCode()

```
int getErrorCode ( )  [inline]
```

Definition at line 12 of file GclibJavaException.java.
The documentation for this class was generated from the following file:

- GclibJavaException.java

9.11 GclibTest Class Reference

Static Public Member Functions

- static void main (String[] args)

9.11.1 Detailed Description

A test of the Java gclib wrapper.
Definition at line 13 of file GclibTest.java.

9.11.2 Member Function Documentation

9.11.2.1 main()

```
static void main (
            String[] args )  [inline], [static]
```

Parameters

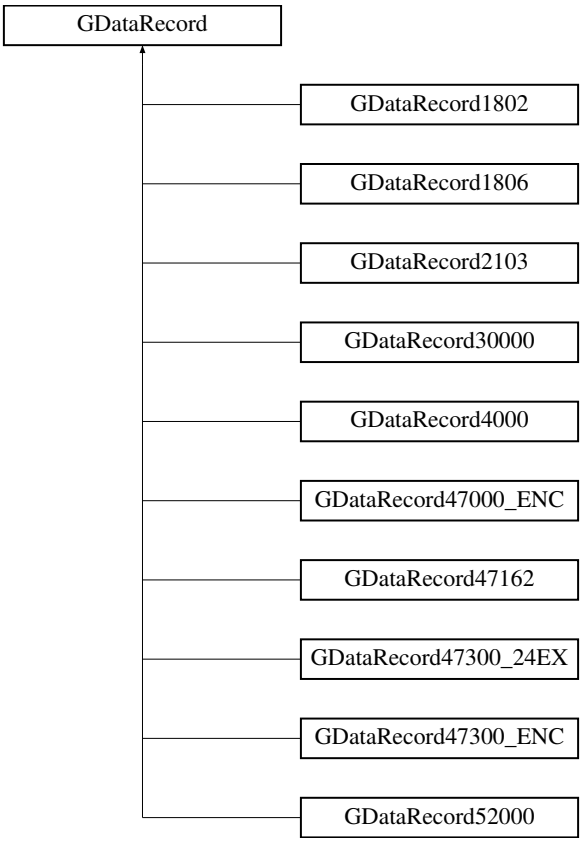
args	the command line arguments

Definition at line 18 of file GclibTest.java.
The documentation for this class was generated from the following file:

- GclibTest.java

9.12 GDataRecord Interface Reference

Inheritance diagram for GDataRecord:



Public Member Functions

- byte[] byte_array ()

  Returns the data record as a byte array and allows for access to individual bytes.

9.12.1 Detailed Description

Definition at line 897 of file gclib.cs.

9.12.2 Member Function Documentation

9.12.2.1 byte_array()

```
byte[] byte_array ( )
```

Returns the data record as a byte array and allows for access to individual bytes.
Implemented in GDataRecord4000, GDataRecord52000, GDataRecord1806, GDataRecord2103, GDataRecord1802, GDataRecord30000, GDataRecord47000_ENC, GDataRecord47300_ENC, GDataRecord47300_24EX, and GDataRecord47162.
The documentation for this interface was generated from the following file:

- gclib.cs

9.13 GDataRecord Union Reference

Data record union, containing all structs and a generic byte array accessor.
`#include <gclib_record.h>`

Data Fields

- struct GDataRecord4000 dmc4000

  The DMC-4000 data record.
- struct GDataRecord4000 dmc4103

  The DMC-4103 data record.
- struct GDataRecord4000 dmc50000

  The DMC-50000 data record.
- struct GDataRecord52000 dmc52000

  The DMC-52000 data record.
- struct GDataRecord30000 dmc30000

  The DMC-30000 data record.
- struct GDataRecord2103 dmc2103

  The DMC-21x3 data record.
- struct GDataRecord1806 dmc1806

  The DMC-1806 data record.
- struct GDataRecord1802 dmc1802

  The DMC-1802 data record.
- struct GDataRecord47000_ENC rio47000

  The RIO-471xx & 472xx data record, including encoder support.
- struct GDataRecord47300_ENC rio47300

  The RIO 473xx data record, including encoder support.
- struct GDataRecord47300_24EX rio47300_24ex

  The RIO 473xx data record, with 24EXOUT/24EXIN support.
- struct GDataRecord47162 rio47162

  The RIO 47162 data record.
- unsigned char byte_array [GALILDATARECORDMAXLENGTH]

  Generic byte array for offsets.

9.13.1 Detailed Description

Data record union, containing all structs and a generic byte array accessor.
Named structs can be used to access typed data by name. Offsets into the data record can also be used by referencing the member `byte_array`.

```
//Getting the sample counter for the DMC-4000.
cout « data_record->dmc4000.sample_number « '\n'; //access by 4000 product
cout « * ((unsigned short *) (data_record->byte_array + 4)) « '\n'; //access by pointer arithmetic
```

Definition at line 1082 of file gclib_record.h.

9.13.2 Field Documentation

9.13.2.1 byte_array

```
unsigned char byte_array[GALILDATARECORDMAXLENGTH]
```

Generic byte array for offsets.
Definition at line 1103 of file gclib_record.h.

9.13.2.2 dmc1802

```
struct GDataRecord1802 dmc1802
```

The DMC-1802 data record.
Definition at line 1096 of file gclib_record.h.

9.13.2.3 dmc1806

```
struct GDataRecord1806 dmc1806
```

The DMC-1806 data record.
Definition at line 1094 of file gclib_record.h.

9.13.2.4 dmc2103

```
struct GDataRecord2103 dmc2103
```

The DMC-21x3 data record.
Definition at line 1092 of file gclib_record.h.

9.13.2.5 dmc30000

```
struct GDataRecord30000 dmc30000
```

The DMC-30000 data record.
Definition at line 1090 of file gclib_record.h.

9.13.2.6 dmc4000

```
struct GDataRecord4000 dmc4000
```

The DMC-4000 data record.
Definition at line 1084 of file gclib_record.h.

9.13.2.7 dmc4103

```
struct GDataRecord4000 dmc4103
```

The DMC-4103 data record.
Definition at line 1085 of file gclib_record.h.

9.13.2.8 dmc50000

```
struct GDataRecord4000 dmc50000
```

The DMC-50000 data record.
Definition at line 1086 of file gclib_record.h.

9.13.2.9 dmc52000

```
struct GDataRecord52000 dmc52000
```

The DMC-52000 data record.
Definition at line 1088 of file gclib_record.h.

9.13.2.10 rio47000

```
struct GDataRecord47000_ENC rio47000
```

The RIO-471xx & 472xx data record, including encoder support.
Definition at line 1098 of file gclib_record.h.

9.13.2.11 rio47162

`struct GDataRecord47162 rio47162`
The RIO 47162 data record.
Definition at line 1101 of file gclib_record.h.

9.13.2.12 rio47300

`struct GDataRecord47300_ENC rio47300`
The RIO 473xx data record, including encoder support.
Definition at line 1099 of file gclib_record.h.

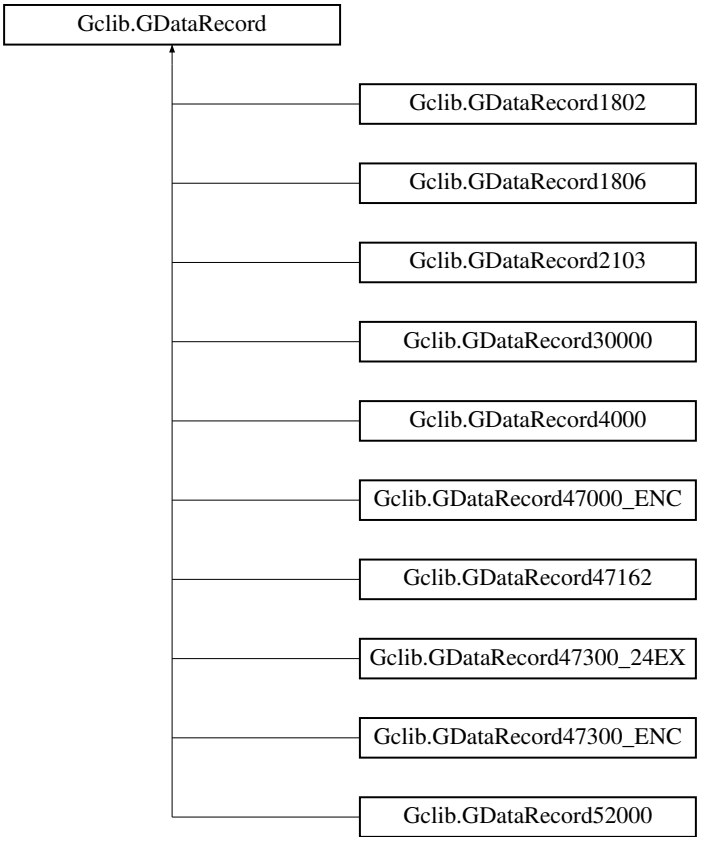
9.13.2.13 rio47300_24ex

`struct GDataRecord47300_24EX rio47300_24ex`
The RIO 473xx data record, with 24EXOUT/24EXIN support.
Definition at line 1100 of file gclib_record.h.
The documentation for this union was generated from the following file:

- gclib_record.h

9.14 Gclib.GDataRecord Interface Reference

Inheritance diagram for Gclib.GDataRecord:



Public Member Functions

- byte_array ()

9.14.1 Detailed Description

Definition at line 467 of file gclib.vb.

9.14.2 Member Function Documentation

9.14.2.1 byte_array()

```
byte_array ( )
```

Implemented in Gclib.GDataRecord4000, Gclib.GDataRecord52000, Gclib.GDataRecord1806, Gclib.GDataRecord2103, Gclib.GDataRecord1802, Gclib.GDataRecord30000, Gclib.GDataRecord47000_ENC, Gclib.GDataRecord47300_ENC, Gclib.GDataRecord47300_24EX, and Gclib.GDataRecord47162.

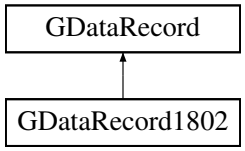
The documentation for this interface was generated from the following file:

- gclib.vb

9.15 GDataRecord1802 Struct Reference

Data record struct for DMC-1802 controllers.

Inheritance diagram for GDataRecord1802:



Public Member Functions

- byte[] byte_array ()

  Returns the data record as a byte array and allows for access to individual bytes.

Data Fields

- UW sample_number

  sample number.
- UB input_bank_0

  general input bank 0 (inputs 1-8).
- UB input_bank_1

  general input bank 1 (inputs 9-16).
- UB input_bank_2

  general input bank 2 (inputs 17-24).
- UB input_bank_3

  general input bank 3 (inputs 25-32).
- UB input_bank_4

  general input bank 4 (inputs 33-40).
- UB input_bank_5

  general input bank 5 (inputs 41-48).
- UB input_bank_6

  general input bank 6 (inputs 49-56).
- UB input_bank_7

  general input bank 7 (inputs 57-64).
- UB input_bank_8

  general input bank 8 (inputs 65-72).
- UB input_bank_9

general input bank 9 (inputs 73-80).

- UB output_bank_0

  general output bank 0 (outputs 1-8).
- UB output_bank_1

  general output bank 1 (outputs 9-16).
- UB output_bank_2

  general output bank 2 (outputs 17-24).
- UB output_bank_3

  general output bank 3 (outputs 25-32).
- UB output_bank_4

  general output bank 4 (outputs 33-40).
- UB output_bank_5

  general output bank 5 (outputs 41-48).
- UB output_bank_6

  general output bank 6 (outputs 49-56).
- UB output_bank_7

  general output bank 7 (outputs 57-64).
- UB output_bank_8

  general output bank 8 (outputs 65-72).
- UB output_bank_9

  general output bank 9 (outputs 73-80).
- UB error_code

  error code.
- UB general_status

  general status
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

  coordinated move status for S plane.
- SL s_distance

  distance traveled in coordinated move for S plane.
- UW t_plane_segment_count

  segment count of coordinated move for T plane.
- UW t_plane_move_status

  Coordinated move status for T plane.
- SL t_distance

  distance traveled in coordinated move for T plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

  A axis stop code.
- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

  A axis auxiliary position.

- SL axis_a_velocity

  A axis velocity.
- SW axis_a_torque

  A axis torque.
- UB axis_a_reserved_0

  Reserved.
- UB axis_a_reserved_1

  Reserved.
- UW axis_b_status

  B axis status.
- UB axis_b_switches

  B axis switches.
- UB axis_b_stop_code

  B axis stop code.
- SL axis_b_reference_position

  B axis reference position.
- SL axis_b_motor_position

  B axis motor position.
- SL axis_b_position_error

  B axis position error.
- SL axis_b_aux_position

  B axis auxiliary position.
- SL axis_b_velocity

  B axis velocity.
- SW axis_b_torque

  B axis torque.
- UB axis_b_reserved_0

  Reserved.
- UB axis_b_reserved_1

  Reserved.
- UW axis_c_status

  C axis status.
- UB axis_c_switches

  C axis switches.
- UB axis_c_stop_code

  C axis stop code.
- SL axis_c_reference_position

  C axis reference position.
- SL axis_c_motor_position

  C axis motor position.
- SL axis_c_position_error

  C axis position error.
- SL axis_c_aux_position

  C axis auxiliary position.
- SL axis_c_velocity

  C axis velocity.
- SW axis_c_torque

  C axis torque.
- UB axis_c_reserved_0

  Reserved.
- UB axis_c_reserved_1

Reserved.

- UW axis_d_status

  D axis status.
- UB axis_d_switches

  D axis switches.
- UB axis_d_stop_code

  D axis stop code.
- SL axis_d_reference_position

  D axis reference position.
- SL axis_d_motor_position

  D axis motor position.
- SL axis_d_position_error

  D axis position error.
- SL axis_d_aux_position

  D axis auxiliary position.
- SL axis_d_velocity

  D axis velocity.
- SW axis_d_torque

  D axis torque.
- UB axis_d_reserved_0

  Reserved.
- UB axis_d_reserved_1

  Reserved.

9.15.1 Detailed Description

Data record struct for DMC-1802 controllers.
The 18x2 Data record is the Same as 2103 except the following.

1. No header bytes. Software removes it from QR.
2. No analog in axis data.

Definition at line 1615 of file gclib.cs.

9.15.2 Member Function Documentation

9.15.2.1 byte_array()

```
byte[] byte_array ( )  [inline]
```

Returns the data record as a byte array and allows for access to individual bytes.
Implements GDataRecord.
Definition at line 1617 of file gclib.cs.

9.15.3 Field Documentation

9.15.3.1 axis_a_aux_position

```
SL axis_a_aux_position
```

A axis auxiliary position.
Definition at line 1662 of file gclib.cs.

9.15.3.2 axis_a_motor_position

```
SL axis_a_motor_position
```

A axis motor position.
Definition at line 1660 of file gclib.cs.

9.15.3.3 axis_a_position_error

`SL axis_a_position_error`
A axis position error.
Definition at line 1661 of file gclib.cs.

9.15.3.4 axis_a_reference_position

`SL axis_a_reference_position`
A axis reference position.
Definition at line 1659 of file gclib.cs.

9.15.3.5 axis_a_reserved_0

`UB axis_a_reserved_0`
Reserved.
Definition at line 1665 of file gclib.cs.

9.15.3.6 axis_a_reserved_1

`UB axis_a_reserved_1`
Reserved.
Definition at line 1666 of file gclib.cs.

9.15.3.7 axis_a_status

`UW axis_a_status`
A axis status.
Definition at line 1656 of file gclib.cs.

9.15.3.8 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 1658 of file gclib.cs.

9.15.3.9 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 1657 of file gclib.cs.

9.15.3.10 axis_a_torque

`SW axis_a_torque`
A axis torque.
Definition at line 1664 of file gclib.cs.

9.15.3.11 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 1663 of file gclib.cs.

9.15.3.12 axis_b_aux_position

`SL axis_b_aux_position`
B axis auxiliary position.
Definition at line 1674 of file gclib.cs.

9.15.3.13 axis_b_motor_position

`SL axis_b_motor_position`
B axis motor position.
Definition at line 1672 of file gclib.cs.

9.15.3.14 axis_b_position_error

`SL axis_b_position_error`
B axis position error.
Definition at line 1673 of file gclib.cs.

9.15.3.15 axis_b_reference_position

`SL axis_b_reference_position`
B axis reference position.
Definition at line 1671 of file gclib.cs.

9.15.3.16 axis_b_reserved_0

`UB axis_b_reserved_0`
Reserved.
Definition at line 1677 of file gclib.cs.

9.15.3.17 axis_b_reserved_1

`UB axis_b_reserved_1`
Reserved.
Definition at line 1678 of file gclib.cs.

9.15.3.18 axis_b_status

`UW axis_b_status`
B axis status.
Definition at line 1668 of file gclib.cs.

9.15.3.19 axis_b_stop_code

`UB axis_b_stop_code`
B axis stop code.
Definition at line 1670 of file gclib.cs.

9.15.3.20 axis_b_switches

`UB axis_b_switches`
B axis switches.
Definition at line 1669 of file gclib.cs.

9.15.3.21 axis_b_torque

`SW axis_b_torque`
B axis torque.
Definition at line 1676 of file gclib.cs.

9.15.3.22 axis_b_velocity

`SL axis_b_velocity`
B axis velocity.
Definition at line 1675 of file gclib.cs.

9.15.3.23 axis_c_aux_position

`SL axis_c_aux_position`
C axis auxiliary position.
Definition at line 1686 of file gclib.cs.

9.15.3.24 axis_c_motor_position

`SL axis_c_motor_position`
C axis motor position.
Definition at line 1684 of file gclib.cs.

9.15.3.25 axis_c_position_error

`SL axis_c_position_error`
C axis position error.
Definition at line 1685 of file gclib.cs.

9.15.3.26 axis_c_reference_position

`SL axis_c_reference_position`
C axis reference position.
Definition at line 1683 of file gclib.cs.

9.15.3.27 axis_c_reserved_0

`UB axis_c_reserved_0`
Reserved.
Definition at line 1689 of file gclib.cs.

9.15.3.28 axis_c_reserved_1

`UB axis_c_reserved_1`
Reserved.
Definition at line 1690 of file gclib.cs.

9.15.3.29 axis_c_status

`UW axis_c_status`
C axis status.
Definition at line 1680 of file gclib.cs.

9.15.3.30 axis_c_stop_code

`UB axis_c_stop_code`
C axis stop code.
Definition at line 1682 of file gclib.cs.

9.15.3.31 axis_c_switches

`UB axis_c_switches`
C axis switches.
Definition at line 1681 of file gclib.cs.

9.15.3.32 axis_c_torque

`SW axis_c_torque`
C axis torque.
Definition at line 1688 of file gclib.cs.

9.15.3.33 axis_c_velocity

`SL axis_c_velocity`
C axis velocity.
Definition at line 1687 of file gclib.cs.

9.15.3.34 axis_d_aux_position

`SL axis_d_aux_position`
D axis auxiliary position.
Definition at line 1698 of file gclib.cs.

9.15.3.35 axis_d_motor_position

`SL axis_d_motor_position`
D axis motor position.
Definition at line 1696 of file gclib.cs.

9.15.3.36 axis_d_position_error

`SL axis_d_position_error`
D axis position error.
Definition at line 1697 of file gclib.cs.

9.15.3.37 axis_d_reference_position

`SL axis_d_reference_position`
D axis reference position.
Definition at line 1695 of file gclib.cs.

9.15.3.38 axis_d_reserved_0

`UB axis_d_reserved_0`
Reserved.
Definition at line 1701 of file gclib.cs.

9.15.3.39 axis_d_reserved_1

`UB axis_d_reserved_1`
Reserved.
Definition at line 1702 of file gclib.cs.

9.15.3.40 axis_d_status

`UW axis_d_status`
D axis status.
Definition at line 1692 of file gclib.cs.

9.15.3.41 axis_d_stop_code

`UB axis_d_stop_code`
D axis stop code.
Definition at line 1694 of file gclib.cs.

9.15.3.42 axis_d_switches

`UB axis_d_switches`
D axis switches.
Definition at line 1693 of file gclib.cs.

9.15.3.43 axis_d_torque

`SW axis_d_torque`
D axis torque.
Definition at line 1700 of file gclib.cs.

9.15.3.44 axis_d_velocity

`SL axis_d_velocity`
D axis velocity.
Definition at line 1699 of file gclib.cs.

9.15.3.45 error_code

`UB error_code`
error code.
Definition at line 1645 of file gclib.cs.

9.15.3.46 general_status

`UB general_status`
general status
Definition at line 1646 of file gclib.cs.

9.15.3.47 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 1623 of file gclib.cs.

9.15.3.48 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 1624 of file gclib.cs.

9.15.3.49 input_bank_2

`UB input_bank_2`
general input bank 2 (inputs 17-24).
Definition at line 1625 of file gclib.cs.

9.15.3.50 input_bank_3

`UB input_bank_3`
general input bank 3 (inputs 25-32).
Definition at line 1626 of file gclib.cs.

9.15.3.51 input_bank_4

`UB input_bank_4`
general input bank 4 (inputs 33-40).
Definition at line 1627 of file gclib.cs.

9.15.3.52 input_bank_5

`UB input_bank_5`
general input bank 5 (inputs 41-48).
Definition at line 1628 of file gclib.cs.

9.15.3.53 input_bank_6

`UB input_bank_6`
general input bank 6 (inputs 49-56).
Definition at line 1629 of file gclib.cs.

9.15.3.54 input_bank_7

`UB input_bank_7`
general input bank 7 (inputs 57-64).
Definition at line 1630 of file gclib.cs.

9.15.3.55 input_bank_8

`UB input_bank_8`
general input bank 8 (inputs 65-72).
Definition at line 1631 of file gclib.cs.

9.15.3.56 input_bank_9

`UB input_bank_9`
general input bank 9 (inputs 73-80).
Definition at line 1632 of file gclib.cs.

9.15.3.57 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 1634 of file gclib.cs.

9.15.3.58 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 1635 of file gclib.cs.

9.15.3.59 output_bank_2

`UB output_bank_2`
general output bank 2 (outputs 17-24).
Definition at line 1636 of file gclib.cs.

9.15.3.60 output_bank_3

`UB output_bank_3`
general output bank 3 (outputs 25-32).
Definition at line 1637 of file gclib.cs.

9.15.3.61 output_bank_4

`UB output_bank_4`
general output bank 4 (outputs 33-40).
Definition at line 1638 of file gclib.cs.

9.15.3.62 output_bank_5

`UB output_bank_5`
general output bank 5 (outputs 41-48).
Definition at line 1639 of file gclib.cs.

9.15.3.63 output_bank_6

`UB output_bank_6`
general output bank 6 (outputs 49-56).
Definition at line 1640 of file gclib.cs.

9.15.3.64 output_bank_7

`UB output_bank_7`
general output bank 7 (outputs 57-64).
Definition at line 1641 of file gclib.cs.

9.15.3.65 output_bank_8

`UB output_bank_8`
general output bank 8 (outputs 65-72).
Definition at line 1642 of file gclib.cs.

9.15.3.66 output_bank_9

`UB output_bank_9`
general output bank 9 (outputs 73-80).
Definition at line 1643 of file gclib.cs.

9.15.3.67 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 1650 of file gclib.cs.

9.15.3.68 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 1649 of file gclib.cs.

9.15.3.69 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 1648 of file gclib.cs.

9.15.3.70 sample_number

`UW sample_number`
sample number.
Definition at line 1621 of file gclib.cs.

9.15.3.71 t_distance

`SL t_distance`
distance traveled in coordinated move for T plane.
Definition at line 1654 of file gclib.cs.

9.15.3.72 t_plane_move_status

`UW t_plane_move_status`
Coordinated move status for T plane.
Definition at line 1653 of file gclib.cs.

9.15.3.73 t_plane_segment_count

`UW t_plane_segment_count`
segment count of coordinated move for T plane.
Definition at line 1652 of file gclib.cs.
The documentation for this struct was generated from the following file:

- gclib.cs

9.16 GDataRecord1802 Struct Reference

`#include <gclib_record.h>`

Data Fields

- UW sample_number
  sample number.
- UB input_bank_0
  general input bank 0 (inputs 1-8).
- UB input_bank_1
  general input bank 1 (inputs 9-16).
- UB input_bank_2
  general input bank 2 (inputs 17-24).
- UB input_bank_3
  general input bank 3 (inputs 25-32).
- UB input_bank_4
  general input bank 4 (inputs 33-40).
- UB input_bank_5
  general input bank 5 (inputs 41-48).
- UB input_bank_6
  general input bank 6 (inputs 49-56).
- UB input_bank_7
  general input bank 7 (inputs 57-64).
- UB input_bank_8
  general input bank 8 (inputs 65-72).
- UB input_bank_9
  general input bank 9 (inputs 73-80).
- UB output_bank_0
  general output bank 0 (outputs 1-8).
- UB output_bank_1
  general output bank 1 (outputs 9-16).
- UB output_bank_2
  general output bank 2 (outputs 17-24).
- UB output_bank_3
  general output bank 3 (outputs 25-32).
- UB output_bank_4
  general output bank 4 (outputs 33-40).
- UB output_bank_5
  general output bank 5 (outputs 41-48).
- UB output_bank_6
  general output bank 6 (outputs 49-56).
- UB output_bank_7
  general output bank 7 (outputs 57-64).

- UB output_bank_8

  general output bank 8 (outputs 65-72).
- UB output_bank_9

  general output bank 9 (outputs 73-80).
- UB error_code

  error code.
- UB general_status

  general status
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

  coordinated move status for S plane.
- SL s_distance

  distance traveled in coordinated move for S plane.
- UW t_plane_segment_count

  segment count of coordinated move for T plane.
- UW t_plane_move_status

  Coordinated move status for T plane.
- SL t_distance

  distance traveled in coordinated move for T plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

  A axis stop code.
- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

  A axis auxiliary position.
- SL axis_a_velocity

  A axis velocity.
- SW axis_a_torque

  A axis torque.
- UB axis_a_reserved_0

  Reserved.
- UB axis_a_reserved_1

  Reserved.
- UW axis_b_status

  B axis status.
- UB axis_b_switches

  B axis switches.
- UB axis_b_stop_code

  B axis stop code.
- SL axis_b_reference_position

  B axis reference position.
- SL axis_b_motor_position

B axis motor position.

- SL axis_b_position_error

  B axis position error.
- SL axis_b_aux_position

  B axis auxiliary position.
- SL axis_b_velocity

  B axis velocity.
- SW axis_b_torque

  B axis torque.
- UB axis_b_reserved_0

  Reserved.
- UB axis_b_reserved_1

  Reserved.
- UW axis_c_status

  C axis status.
- UB axis_c_switches

  C axis switches.
- UB axis_c_stop_code

  C axis stop code.
- SL axis_c_reference_position

  C axis reference position.
- SL axis_c_motor_position

  C axis motor position.
- SL axis_c_position_error

  C axis position error.
- SL axis_c_aux_position

  C axis auxiliary position.
- SL axis_c_velocity

  C axis velocity.
- SW axis_c_torque

  C axis torque.
- UB axis_c_reserved_0

  Reserved.
- UB axis_c_reserved_1

  Reserved.
- UW axis_d_status

  D axis status.
- UB axis_d_switches

  D axis switches.
- UB axis_d_stop_code

  D axis stop code.
- SL axis_d_reference_position

  D axis reference position.
- SL axis_d_motor_position

  D axis motor position.
- SL axis_d_position_error

  D axis position error.
- SL axis_d_aux_position

  D axis auxiliary position.
- SL axis_d_velocity

  D axis velocity.

- SW axis_d_torque
  D axis torque.
- UB axis_d_reserved_0
  Reserved.
- UB axis_d_reserved_1
  Reserved.

9.16.1 Detailed Description

Data record struct for DMC-1802 controllers.
The 18x2 Data record is the Same as 2103 except the following.

1. No header bytes. Software removes it from QR.
2. No analog in axis data.

Definition at line 727 of file gclib_record.h.

9.16.2 Field Documentation

9.16.2.1 axis_a_aux_position

`SL axis_a_aux_position`
A axis auxiliary position.
Definition at line 773 of file gclib_record.h.

9.16.2.2 axis_a_motor_position

`SL axis_a_motor_position`
A axis motor position.
Definition at line 771 of file gclib_record.h.

9.16.2.3 axis_a_position_error

`SL axis_a_position_error`
A axis position error.
Definition at line 772 of file gclib_record.h.

9.16.2.4 axis_a_reference_position

`SL axis_a_reference_position`
A axis reference position.
Definition at line 770 of file gclib_record.h.

9.16.2.5 axis_a_reserved_0

`UB axis_a_reserved_0`
Reserved.
Definition at line 776 of file gclib_record.h.

9.16.2.6 axis_a_reserved_1

`UB axis_a_reserved_1`
Reserved.
Definition at line 777 of file gclib_record.h.

9.16.2.7 axis_a_status

`UW axis_a_status`
A axis status.
Definition at line 767 of file gclib_record.h.

9.16.2.8 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 769 of file gclib_record.h.

9.16.2.9 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 768 of file gclib_record.h.

9.16.2.10 axis_a_torque

`SW axis_a_torque`
A axis torque.
Definition at line 775 of file gclib_record.h.

9.16.2.11 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 774 of file gclib_record.h.

9.16.2.12 axis_b_aux_position

`SL axis_b_aux_position`
B axis auxiliary position.
Definition at line 785 of file gclib_record.h.

9.16.2.13 axis_b_motor_position

`SL axis_b_motor_position`
B axis motor position.
Definition at line 783 of file gclib_record.h.

9.16.2.14 axis_b_position_error

`SL axis_b_position_error`
B axis position error.
Definition at line 784 of file gclib_record.h.

9.16.2.15 axis_b_reference_position

`SL axis_b_reference_position`
B axis reference position.
Definition at line 782 of file gclib_record.h.

9.16.2.16 axis_b_reserved_0

`UB axis_b_reserved_0`
Reserved.
Definition at line 788 of file gclib_record.h.

9.16.2.17 axis_b_reserved_1

`UB axis_b_reserved_1`
Reserved.
Definition at line 789 of file gclib_record.h.

9.16.2.18 axis_b_status

`UW axis_b_status`
B axis status.
Definition at line 779 of file gclib_record.h.

9.16.2.19 axis_b_stop_code

`UB axis_b_stop_code`
B axis stop code.
Definition at line 781 of file gclib_record.h.

9.16.2.20 axis_b_switches

`UB axis_b_switches`
B axis switches.
Definition at line 780 of file gclib_record.h.

9.16.2.21 axis_b_torque

`SW axis_b_torque`
B axis torque.
Definition at line 787 of file gclib_record.h.

9.16.2.22 axis_b_velocity

`SL axis_b_velocity`
B axis velocity.
Definition at line 786 of file gclib_record.h.

9.16.2.23 axis_c_aux_position

`SL axis_c_aux_position`
C axis auxiliary position.
Definition at line 797 of file gclib_record.h.

9.16.2.24 axis_c_motor_position

`SL axis_c_motor_position`
C axis motor position.
Definition at line 795 of file gclib_record.h.

9.16.2.25 axis_c_position_error

`SL axis_c_position_error`
C axis position error.
Definition at line 796 of file gclib_record.h.

9.16.2.26 axis_c_reference_position

`SL axis_c_reference_position`
C axis reference position.
Definition at line 794 of file gclib_record.h.

9.16.2.27 axis_c_reserved_0

`UB axis_c_reserved_0`
Reserved.
Definition at line 800 of file gclib_record.h.

9.16.2.28 axis_c_reserved_1

`UB axis_c_reserved_1`
Reserved.
Definition at line 801 of file gclib_record.h.

9.16.2.29 axis_c_status

`UW axis_c_status`
C axis status.
Definition at line 791 of file gclib_record.h.

9.16.2.30 axis_c_stop_code

`UB axis_c_stop_code`
C axis stop code.
Definition at line 793 of file gclib_record.h.

9.16.2.31 axis_c_switches

`UB axis_c_switches`
C axis switches.
Definition at line 792 of file gclib_record.h.

9.16.2.32 axis_c_torque

`SW axis_c_torque`
C axis torque.
Definition at line 799 of file gclib_record.h.

9.16.2.33 axis_c_velocity

`SL axis_c_velocity`
C axis velocity.
Definition at line 798 of file gclib_record.h.

9.16.2.34 axis_d_aux_position

`SL axis_d_aux_position`
D axis auxiliary position.
Definition at line 809 of file gclib_record.h.

9.16.2.35 axis_d_motor_position

`SL axis_d_motor_position`
D axis motor position.
Definition at line 807 of file gclib_record.h.

9.16.2.36 axis_d_position_error

`SL axis_d_position_error`
D axis position error.
Definition at line 808 of file gclib_record.h.

9.16.2.37 axis_d_reference_position

`SL axis_d_reference_position`
D axis reference position.
Definition at line 806 of file gclib_record.h.

9.16.2.38 axis_d_reserved_0

`UB axis_d_reserved_0`
Reserved.
Definition at line 812 of file gclib_record.h.

9.16.2.39 axis_d_reserved_1

`UB axis_d_reserved_1`
Reserved.
Definition at line 813 of file gclib_record.h.

9.16.2.40 axis_d_status

`UW axis_d_status`
D axis status.
Definition at line 803 of file gclib_record.h.

9.16.2.41 axis_d_stop_code

`UB axis_d_stop_code`
D axis stop code.
Definition at line 805 of file gclib_record.h.

9.16.2.42 axis_d_switches

`UB axis_d_switches`
D axis switches.
Definition at line 804 of file gclib_record.h.

9.16.2.43 axis_d_torque

`SW axis_d_torque`
D axis torque.
Definition at line 811 of file gclib_record.h.

9.16.2.44 axis_d_velocity

`SL axis_d_velocity`
D axis velocity.
Definition at line 810 of file gclib_record.h.

9.16.2.45 error_code

`UB error_code`
error code.
Definition at line 756 of file gclib_record.h.

9.16.2.46 general_status

`UB general_status`
general status
Definition at line 757 of file gclib_record.h.

9.16.2.47 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 734 of file gclib_record.h.

9.16.2.48 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 735 of file gclib_record.h.

9.16.2.49 input_bank_2

`UB input_bank_2`
general input bank 2 (inputs 17-24).
Definition at line 736 of file gclib_record.h.

9.16.2.50 input_bank_3

`UB input_bank_3`
general input bank 3 (inputs 25-32).
Definition at line 737 of file gclib_record.h.

9.16.2.51 input_bank_4

`UB input_bank_4`
general input bank 4 (inputs 33-40).
Definition at line 738 of file gclib_record.h.

9.16.2.52 input_bank_5

`UB input_bank_5`
general input bank 5 (inputs 41-48).
Definition at line 739 of file gclib_record.h.

9.16.2.53 input_bank_6

`UB input_bank_6`
general input bank 6 (inputs 49-56).
Definition at line 740 of file gclib_record.h.

9.16.2.54 input_bank_7

`UB input_bank_7`
general input bank 7 (inputs 57-64).
Definition at line 741 of file gclib_record.h.

9.16.2.55 input_bank_8

`UB input_bank_8`
general input bank 8 (inputs 65-72).
Definition at line 742 of file gclib_record.h.

9.16.2.56 input_bank_9

`UB input_bank_9`
general input bank 9 (inputs 73-80).
Definition at line 743 of file gclib_record.h.

9.16.2.57 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 745 of file gclib_record.h.

9.16.2.58 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 746 of file gclib_record.h.

9.16.2.59 output_bank_2

`UB output_bank_2`
general output bank 2 (outputs 17-24).
Definition at line 747 of file gclib_record.h.

9.16.2.60 output_bank_3

`UB output_bank_3`
general output bank 3 (outputs 25-32).
Definition at line 748 of file gclib_record.h.

9.16.2.61 output_bank_4

`UB output_bank_4`
general output bank 4 (outputs 33-40).
Definition at line 749 of file gclib_record.h.

9.16.2.62 output_bank_5

`UB output_bank_5`
general output bank 5 (outputs 41-48).
Definition at line 750 of file gclib_record.h.

9.16.2.63 output_bank_6

`UB output_bank_6`
general output bank 6 (outputs 49-56).
Definition at line 751 of file gclib_record.h.

9.16.2.64 output_bank_7

`UB output_bank_7`
general output bank 7 (outputs 57-64).
Definition at line 752 of file gclib_record.h.

9.16.2.65 output_bank_8

`UB output_bank_8`
general output bank 8 (outputs 65-72).
Definition at line 753 of file gclib_record.h.

9.16.2.66 output_bank_9

`UB output_bank_9`
general output bank 9 (outputs 73-80).
Definition at line 754 of file gclib_record.h.

9.16.2.67 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 761 of file gclib_record.h.

9.16.2.68 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 760 of file gclib_record.h.

9.16.2.69 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 759 of file gclib_record.h.

9.16.2.70 sample_number

`UW sample_number`
sample number.
Definition at line 732 of file gclib_record.h.

9.16.2.71 t_distance

`SL t_distance`
distance traveled in coordinated move for T plane.
Definition at line 765 of file gclib_record.h.

9.16.2.72 t_plane_move_status

`UW t_plane_move_status`
Coordinated move status for T plane.
Definition at line 764 of file gclib_record.h.

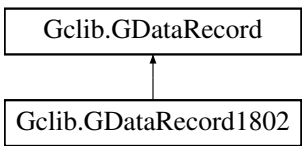
9.16.2.73 t_plane_segment_count

`UW t_plane_segment_count`
segment count of coordinated move for T plane.
Definition at line 763 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

9.17 Gclib.GDataRecord1802 Struct Reference

Inheritance diagram for Gclib.GDataRecord1802:



Public Member Functions

- general input **bank** (inputs 1-8). ∗/public UB input_bank_1

  /∗03
- general input **bank** (inputs 9-16). ∗/public UB input_bank_2

  /∗04
- general input **bank** (inputs 17-24). ∗/public UB input_bank_3

  /∗05

- general input **bank** (inputs 25-32). */public UB input_bank_4
  /*06
- general input **bank** (inputs 33-40). */public UB input_bank_5
  /*07
- general input **bank** (inputs 41-48). */public UB input_bank_6
  /*08
- general input **bank** (inputs 49-56). */public UB input_bank_7
  /*09
- general input **bank** (inputs 57-64). */public UB input_bank_8
  /*10
- general input **bank** (inputs 65-72). */public UB input_bank_9
  /*11
- general input **bank** (inputs 73-80). */public UB output_bank_0
  /*12
- general output **bank** (outputs 1-8). */public UB output_bank_1
  /*13
- general output **bank** (outputs 9-16). */public UB output_bank_2
  /*14
- general output **bank** (outputs 17-24). */public UB output_bank_3
  /*15
- general output **bank** (outputs 25-32). */public UB output_bank_4
  /*16
- general output **bank** (outputs 33-40). */public UB output_bank_5
  /*17
- general output **bank** (outputs 41-48). */public UB output_bank_6
  /*18
- general output **bank** (outputs 49-56). */public UB output_bank_7
  /*19
- general output **bank** (outputs 57-64). */public UB output_bank_8
  /*20
- general output **bank** (outputs 65-72). */public UB output_bank_9
  /*21
- general output **bank** (outputs 73-80). */public UB error_code
  /*22
- Reserved.* byte_array ()

Data Fields

- UW sample_number
  /*00-01
- sample number.*UB input_bank_0
  /*02
- error code.*UB general_status
  /*23
- general status *UW s_plane_segment_count
  /*24-25
- segment count of coordinated move for S plane.*UW s_plane_move_status
  /*26-27
- coordinated move status for S plane.*SL s_distance
  /*28-31
- distance traveled in coordinated move for S plane.*UW t_plane_segment_count

/*32-33
- segment count of coordinated move for T plane.*UW t_plane_move_status
/*34-35
- Coordinated move status for T plane.*SL t_distance
/*36-39
- distance traveled in coordinated move for T plane.*UW axis_a_status
/*40-41
- A axis status.*UB axis_a_switches
/*42
- A axis switches.*UB axis_a_stop_code
/*43
- A axis stop code.*SL axis_a_reference_position
/*44-47
- A axis reference position.*SL axis_a_motor_position
/*48-51
- A axis motor position.*SL axis_a_position_error
/*52-55
- A axis position error.*SL axis_a_aux_position
/*56-59
- A axis auxiliary position.*SL axis_a_velocity
/*60-63
- A axis velocity.*SW axis_a_torque
/*64-65
- A axis torque.*UB axis_a_reserved_0
/*66
- Reserved.*UB axis_a_reserved_1
/*67
- Reserved.*UW axis_b_status
/*68-69
- B axis status.*UB axis_b_switches
/*70
- B axis switches.*UB axis_b_stop_code
/*71
- B axis stop code.*SL axis_b_reference_position
/*72-75
- B axis reference position.*SL axis_b_motor_position
/*76-79
- B axis motor position.*SL axis_b_position_error
/*80-83
- B axis position error.*SL axis_b_aux_position
/*84-87
- B axis auxiliary position.*SL axis_b_velocity
/*88-91
- B axis velocity.*SW axis_b_torque
/*92-93
- B axis torque.*UB axis_b_reserved_0
/*94
- Reserved.*UB axis_b_reserved_1
/*95
- Reserved.*UW axis_c_status
/*96-97

- C axis status.∗UB axis_c_switches
  /∗98
- C axis switches.∗UB axis_c_stop_code
  /∗99
- C axis stop code.∗SL axis_c_reference_position
  /∗100-103
- C axis reference position.∗SL axis_c_motor_position
  /∗104-107
- C axis motor position.∗SL axis_c_position_error
  /∗108-111
- C axis position error.∗SL axis_c_aux_position
  /∗112-115
- C axis auxiliary position.∗SL axis_c_velocity
  /∗116-119
- C axis velocity.∗SW axis_c_torque
  /∗120-121
- C axis torque.∗UB axis_c_reserved_0
  /∗122
- Reserved.∗UB axis_c_reserved_1
  /∗123
- Reserved.∗UW axis_d_status
  /∗124-125
- D axis status.∗UB axis_d_switches
  /∗126
- D axis switches.∗UB axis_d_stop_code
  /∗127
- D axis stop code.∗SL axis_d_reference_position
  /∗128-131
- D axis reference position.∗SL axis_d_motor_position
  /∗132-135
- D axis motor position.∗SL axis_d_position_error
  /∗136-139
- D axis position error.∗SL axis_d_aux_position
  /∗140-143
- D axis auxiliary position.∗SL axis_d_velocity
  /∗144-147
- D axis velocity.∗SW axis_d_torque
  /∗148-149
- D axis torque.∗UB axis_d_reserved_0
  /∗150
- Reserved.∗UB axis_d_reserved_1
  /∗151

9.17.1 Detailed Description

Definition at line 1076 of file gclib.vb.

9.17.2 Member Function Documentation

9.17.2.1 byte_array()

```
Reserved.* byte_array ( )
```

Implements GDataRecord.byte_array
Implements Gclib.GDataRecord.

9.17.3 Field Documentation

9.17.3.1 axis_a_aux_position

```
A axis position error.* SL axis_a_aux_position
```
/*56-59
Definition at line 1114 of file gclib.vb.

9.17.3.2 axis_a_motor_position

```
A axis reference position.* SL axis_a_motor_position
```
/*48-51
Definition at line 1112 of file gclib.vb.

9.17.3.3 axis_a_position_error

```
A axis motor position.* SL axis_a_position_error
```
/*52-55
Definition at line 1113 of file gclib.vb.

9.17.3.4 axis_a_reference_position

```
A axis stop code.* SL axis_a_reference_position
```
/*44-47
Definition at line 1111 of file gclib.vb.

9.17.3.5 axis_a_reserved_0

```
A axis torque.* UB axis_a_reserved_0
```
/*66
Definition at line 1117 of file gclib.vb.

9.17.3.6 axis_a_reserved_1

```
Reserved.* UB axis_a_reserved_1
```
/*67
Definition at line 1118 of file gclib.vb.

9.17.3.7 axis_a_status

```
distance traveled in coordinated move for T plane.* UW axis_a_status
```
/*40-41
Definition at line 1108 of file gclib.vb.

9.17.3.8 axis_a_stop_code

```
A axis switches.* UB axis_a_stop_code
```
/*43
Definition at line 1110 of file gclib.vb.

9.17.3.9 axis_a_switches

```
A axis status.* UB axis_a_switches
```
/*42
Definition at line 1109 of file gclib.vb.

9.17.3.10 axis_a_torque

```
A axis velocity.* SW axis_a_torque
```
/*64-65
Definition at line 1116 of file gclib.vb.

9.17.3.11 axis_a_velocity

```
A axis auxiliary position.* SL axis_a_velocity
```
/*60-63
Definition at line 1115 of file gclib.vb.

9.17.3.12 axis_b_aux_position

```
B axis position error.* SL axis_b_aux_position
```
/*84-87
Definition at line 1125 of file gclib.vb.

9.17.3.13 axis_b_motor_position

```
B axis reference position.* SL axis_b_motor_position
```
/*76-79
Definition at line 1123 of file gclib.vb.

9.17.3.14 axis_b_position_error

```
B axis motor position.* SL axis_b_position_error
```
/*80-83
Definition at line 1124 of file gclib.vb.

9.17.3.15 axis_b_reference_position

```
B axis stop code.* SL axis_b_reference_position
```
/*72-75
Definition at line 1122 of file gclib.vb.

9.17.3.16 axis_b_reserved_0

```
B axis torque.* UB axis_b_reserved_0
```
/*94
Definition at line 1128 of file gclib.vb.

9.17.3.17 axis_b_reserved_1

```
Reserved.* UB axis_b_reserved_1
```
/*95
Definition at line 1129 of file gclib.vb.

9.17.3.18 axis_b_status

```
Reserved.* UW axis_b_status
```
/*68-69
Definition at line 1119 of file gclib.vb.

9.17.3.19 axis_b_stop_code

```
B axis switches.* UB axis_b_stop_code
```
/*71
Definition at line 1121 of file gclib.vb.

9.17.3.20 axis_b_switches

```
B axis status.* UB axis_b_switches
```
/*70
Definition at line 1120 of file gclib.vb.

9.17.3.21 axis_b_torque

```
B axis velocity.* SW axis_b_torque
```
/*92-93
Definition at line 1127 of file gclib.vb.

9.17.3.22 axis_b_velocity

```
B axis auxiliary position.* SL axis_b_velocity
```
/*88-91
Definition at line 1126 of file gclib.vb.

9.17.3.23 axis_c_aux_position

```
C axis position error.* SL axis_c_aux_position
```
/*112-115
Definition at line 1136 of file gclib.vb.

9.17.3.24 axis_c_motor_position

```
C axis reference position.* SL axis_c_motor_position
```
/*104-107
Definition at line 1134 of file gclib.vb.

9.17.3.25 axis_c_position_error

```
C axis motor position.* SL axis_c_position_error
```
/*108-111
Definition at line 1135 of file gclib.vb.

9.17.3.26 axis_c_reference_position

```
C axis stop code.* SL axis_c_reference_position
```
/*100-103
Definition at line 1133 of file gclib.vb.

9.17.3.27 axis_c_reserved_0

```
C axis torque.* UB axis_c_reserved_0
```
/*122
Definition at line 1139 of file gclib.vb.

9.17.3.28 axis_c_reserved_1

```
Reserved.* UB axis_c_reserved_1
```
/*123
Definition at line 1140 of file gclib.vb.

9.17.3.29 axis_c_status

```
Reserved.* UW axis_c_status
```
/*96-97
Definition at line 1130 of file gclib.vb.

9.17.3.30 axis_c_stop_code

```
C axis switches.* UB axis_c_stop_code
```
/*99
Definition at line 1132 of file gclib.vb.

9.17.3.31 axis_c_switches

```
C axis status.* UB axis_c_switches
```
/*98
Definition at line 1131 of file gclib.vb.

9.17.3.32 axis_c_torque

```
C axis velocity.* SW axis_c_torque
```
/*120-121
Definition at line 1138 of file gclib.vb.

9.17.3.33 axis_c_velocity

```
C axis auxiliary position.* SL axis_c_velocity
```
/*116-119
Definition at line 1137 of file gclib.vb.

9.17.3.34 axis_d_aux_position

```
D axis position error.* SL axis_d_aux_position
```
/*140-143
Definition at line 1147 of file gclib.vb.

9.17.3.35 axis_d_motor_position

```
D axis reference position.* SL axis_d_motor_position
```
/*132-135
Definition at line 1145 of file gclib.vb.

9.17.3.36 axis_d_position_error

```
D axis motor position.* SL axis_d_position_error
```
/*136-139
Definition at line 1146 of file gclib.vb.

9.17.3.37 axis_d_reference_position

```
D axis stop code.* SL axis_d_reference_position
```
/*128-131
Definition at line 1144 of file gclib.vb.

9.17.3.38 axis_d_reserved_0

```
D axis torque.* UB axis_d_reserved_0
```
/*150
Definition at line 1150 of file gclib.vb.

9.17.3.39 axis_d_reserved_1

```
Reserved.* UB axis_d_reserved_1
```
/*151
Definition at line 1151 of file gclib.vb.

9.17.3.40 axis_d_status

```
Reserved.* UW axis_d_status
```
/*124-125
Definition at line 1141 of file gclib.vb.

9.17.3.41 axis_d_stop_code

```
D axis switches.* UB axis_d_stop_code
```
/*127
Definition at line 1143 of file gclib.vb.

9.17.3.42 axis_d_switches

```
D axis status.* UB axis_d_switches
```
/*126
Definition at line 1142 of file gclib.vb.

9.17.3.43 axis_d_torque

```
D axis velocity.* SW axis_d_torque
```
/*148-149
Definition at line 1149 of file gclib.vb.

9.17.3.44 axis_d_velocity

```
D axis auxiliary position.* SL axis_d_velocity
```
/*144-147
Definition at line 1148 of file gclib.vb.

9.17.3.45 general_status

```
error code.* UB general_status
```
/*23
Definition at line 1101 of file gclib.vb.

9.17.3.46 input_bank_0

```
sample number.* UB input_bank_0
```
/*02
Definition at line 1080 of file gclib.vb.

9.17.3.47 s_distance

```
coordinated move status for S plane.* SL s_distance
```
/*28-31
Definition at line 1104 of file gclib.vb.

9.17.3.48 s_plane_move_status

```
segment count of coordinated move for S plane.* UW s_plane_move_status
```
/*26-27
Definition at line 1103 of file gclib.vb.

9.17.3.49 s_plane_segment_count

```
general status* UW s_plane_segment_count
```
/*24-25
Definition at line 1102 of file gclib.vb.

9.17.3.50 sample_number

```
UW sample_number
```
/*00-01
Definition at line 1079 of file gclib.vb.

9.17.3.51 t_distance

```
Coordinated move status for T plane.* SL t_distance
```

/*36-39
Definition at line 1107 of file gclib.vb.

9.17.3.52 t_plane_move_status

```
segment count of coordinated move for T plane.* UW t_plane_move_status
```

/*34-35
Definition at line 1106 of file gclib.vb.

9.17.3.53 t_plane_segment_count

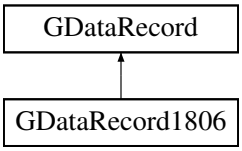
```
distance traveled in coordinated move for S plane.* UW t_plane_segment_count
```

/*32-33
Definition at line 1105 of file gclib.vb.
The documentation for this struct was generated from the following file:

- gclib.vb

9.18 GDataRecord1806 Struct Reference

Data record struct for DMC-1806 controller.
Inheritance diagram for GDataRecord1806:



Public Member Functions

- byte[] byte_array ()

  Returns the data record as a byte array and allows for access to individual bytes.

Data Fields

- UW sample_number

  sample number.
- UB input_bank_0

  general input bank 0 (inputs 1-8).
- UB input_bank_1

  general input bank 1 (inputs 9-16).
- UB input_bank_2

  general input bank 2 (inputs 17-24).
- UB input_bank_3

  general input bank 3 (inputs 25-32).
- UB input_bank_4

  general input bank 4 (inputs 33-40).
- UB input_bank_5

  general input bank 5 (inputs 41-48).
- UB input_bank_6

  general input bank 6 (inputs 49-56).

- UB input_bank_7

  general input bank 7 (inputs 57-64).
- UB input_bank_8

  general input bank 8 (inputs 65-72).
- UB input_bank_9

  general input bank 9 (inputs 73-80).
- UB output_bank_0

  general output bank 0 (outputs 1-8).
- UB output_bank_1

  general output bank 1 (outputs 9-16).
- UB output_bank_2

  general output bank 2 (outputs 17-24).
- UB output_bank_3

  general output bank 3 (outputs 25-32).
- UB output_bank_4

  general output bank 4 (outputs 33-40).
- UB output_bank_5

  general output bank 5 (outputs 41-48).
- UB output_bank_6

  general output bank 6 (outputs 49-56).
- UB output_bank_7

  general output bank 7 (outputs 57-64).
- UB output_bank_8

  general output bank 8 (outputs 65-72).
- UB output_bank_9

  general output bank 9 (outputs 73-80).
- SW reserved_0

  Reserved.
- SW reserved_2

  Reserved.
- SW reserved_4

  Reserved.
- SW reserved_6

  Reserved.
- SW reserved_8

  Reserved.
- SW reserved_10

  Reserved.
- SW reserved_12

  Reserved.
- SW reserved_14

  Reserved.
- UB reserved_16

  Reserved.
- UB reserved_17

  Reserved.
- UB reserved_18

  Reserved.
- UB reserved_19

  Reserved.
- UB reserved_20

Reserved.

- UB reserved_21

  Reserved.
- UB reserved_22

  Reserved.
- UB reserved_23

  Reserved.
- UB error_code

  error code.
- UB thread_status

  thread status.
- UL reserved_24

  Reserved.
- UL contour_segment_count

  Segment Count for Contour Mode.
- UW contour_buffer_available

  Buffer space remaining, Contour Mode.
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

  coordinated move status for S plane.
- SL s_distance

  distance traveled in coordinated move for S plane.
- UW s_plane_buffer_available

  Buffer space remaining, S Plane.
- UW t_plane_segment_count

  segment count of coordinated move for T plane.
- UW t_plane_move_status

  Coordinated move status for T plane.
- SL t_distance

  distance traveled in coordinated move for T plane.
- UW t_plane_buffer_available

  Buffer space remaining, T Plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

  A axis stop code.
- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

  A axis auxiliary position.
- SL axis_a_velocity

  A axis velocity.
- SL axis_a_torque

  A axis torque.

- UW axis_a_analog_in

  A axis analog input.
- UB axis_a_reserved_0

  Reserved.
- UB axis_a_reserved_1

  Reserved.
- SL axis_a_variable

  A User-defined variable (ZA).
- UW axis_b_status

  B axis status.
- UB axis_b_switches

  B axis switches.
- UB axis_b_stop_code

  B axis stop code.
- SL axis_b_reference_position

  B axis reference position.
- SL axis_b_motor_position

  B axis motor position.
- SL axis_b_position_error

  B axis position error.
- SL axis_b_aux_position

  B axis auxiliary position.
- SL axis_b_velocity

  B axis velocity.
- SL axis_b_torque

  B axis torque.
- UW axis_b_analog_in

  B axis analog input.
- UB axis_b_reserved_0

  Reserved.
- UB axis_b_reserved_1

  Reserved.
- SL axis_b_variable

  B User-defined variable (ZA).
- UW axis_c_status

  C axis status.
- UB axis_c_switches

  C axis switches.
- UB axis_c_stop_code

  C axis stop code.
- SL axis_c_reference_position

  C axis reference position.
- SL axis_c_motor_position

  C axis motor position.
- SL axis_c_position_error

  C axis position error.
- SL axis_c_aux_position

  C axis auxiliary position.
- SL axis_c_velocity

  C axis velocity.
- SL axis_c_torque

C axis torque.

- UW axis_c_analog_in

  C axis analog input.
- UB axis_c_reserved_0

  Reserved.
- UB axis_c_reserved_1

  Reserved.
- SL axis_c_variable

  C User-defined variable (ZA).
- UW axis_d_status

  D axis status.
- UB axis_d_switches

  D axis switches.
- UB axis_d_stop_code

  D axis stop code.
- SL axis_d_reference_position

  D axis reference position.
- SL axis_d_motor_position

  D axis motor position.
- SL axis_d_position_error

  D axis position error.
- SL axis_d_aux_position

  D axis auxiliary position.
- SL axis_d_velocity

  D axis velocity.
- SL axis_d_torque

  D axis torque.
- UW axis_d_analog_in

  D axis analog input.
- UB axis_d_reserved_0

  Reserved.
- UB axis_d_reserved_1

  Reserved.
- SL axis_d_variable

  D User-defined variable (ZA).
- UW axis_e_status

  E axis status.
- UB axis_e_switches

  E axis switches.
- UB axis_e_stop_code

  E axis stop code.
- SL axis_e_reference_position

  E axis reference position.
- SL axis_e_motor_position

  E axis motor position.
- SL axis_e_position_error

  E axis position error.
- SL axis_e_aux_position

  E axis auxiliary position.
- SL axis_e_velocity

  E axis velocity.

- SL axis_e_torque

  E axis torque.
- UW axis_e_analog_in

  E axis analog input.
- UB axis_e_reserved_0

  Reserved.
- UB axis_e_reserved_1

  Reserved.
- SL axis_e_variable

  E User-defined variable (ZA).
- UW axis_f_status

  F axis status.
- UB axis_f_switches

  F axis switches.
- UB axis_f_stop_code

  F axis stop code.
- SL axis_f_reference_position

  F axis reference position.
- SL axis_f_motor_position

  F axis motor position.
- SL axis_f_position_error

  F axis position error.
- SL axis_f_aux_position

  F axis auxiliary position.
- SL axis_f_velocity

  F axis velocity.
- SL axis_f_torque

  F axis torque.
- UW axis_f_analog_in

  F axis analog input.
- UB axis_f_reserved_0

  Reserved.
- UB axis_f_reserved_1

  Reserved.
- SL axis_f_variable

  F User-defined variable (ZA).
- UW axis_g_status

  G axis status.
- UB axis_g_switches

  G axis switches.
- UB axis_g_stop_code

  G axis stop code.
- SL axis_g_reference_position

  G axis reference position.
- SL axis_g_motor_position

  G axis motor position.
- SL axis_g_position_error

  G axis position error.
- SL axis_g_aux_position

  G axis auxiliary position.
- SL axis_g_velocity

G axis velocity.

- SL axis_g_torque

  G axis torque.

- UW axis_g_analog_in

  G axis analog input.

- UB axis_g_reserved_0

  Reserved.

- UB axis_g_reserved_1

  Reserved.

- SL axis_g_variable

  G User-defined variable (ZA).

- UW axis_h_status

  H axis status.

- UB axis_h_switches

  H axis switches.

- UB axis_h_stop_code

  H axis stop code.

- SL axis_h_reference_position

  H axis reference position.

- SL axis_h_motor_position

  H axis motor position.

- SL axis_h_position_error

  H axis position error.

- SL axis_h_aux_position

  H axis auxiliary position.

- SL axis_h_velocity

  H axis velocity.

- SL axis_h_torque

  H axis torque.

- UW axis_h_analog_in

  H axis analog input.

- UB axis_h_reserved_0

  Reserved.

- UB axis_h_reserved_1

  Reserved.

- SL axis_h_variable

  H User-defined variable (ZA).

9.18.1 Detailed Description

Data record struct for DMC-1806 controller.
The 18x6 Data record is the same as 4000 except the following.

1. No header bytes. Firmware strips it in DR. Software removes it from QR.
2. No Ethernet status (bytes 42-49).
3. No amplfifier status (bytes 52-55).
4. No axis-specific hall input status.

Definition at line 1294 of file gclib.cs.

9.18.2 Member Function Documentation

9.18.2.1 byte_array()

```
byte[] byte_array ( )  [inline]
```

Returns the data record as a byte array and allows for access to individual bytes.
Implements GDataRecord.
Definition at line 1296 of file gclib.cs.

9.18.3 Field Documentation

9.18.3.1 axis_a_analog_in

```
UW axis_a_analog_in
```

A axis analog input.
Definition at line 1367 of file gclib.cs.

9.18.3.2 axis_a_aux_position

```
SL axis_a_aux_position
```

A axis auxiliary position.
Definition at line 1364 of file gclib.cs.

9.18.3.3 axis_a_motor_position

```
SL axis_a_motor_position
```

A axis motor position.
Definition at line 1362 of file gclib.cs.

9.18.3.4 axis_a_position_error

```
SL axis_a_position_error
```

A axis position error.
Definition at line 1363 of file gclib.cs.

9.18.3.5 axis_a_reference_position

```
SL axis_a_reference_position
```

A axis reference position.
Definition at line 1361 of file gclib.cs.

9.18.3.6 axis_a_reserved_0

```
UB axis_a_reserved_0
```

Reserved.
Definition at line 1368 of file gclib.cs.

9.18.3.7 axis_a_reserved_1

```
UB axis_a_reserved_1
```

Reserved.
Definition at line 1369 of file gclib.cs.

9.18.3.8 axis_a_status

```
UW axis_a_status
```

A axis status.
Definition at line 1358 of file gclib.cs.

9.18.3.9 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 1360 of file gclib.cs.

9.18.3.10 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 1359 of file gclib.cs.

9.18.3.11 axis_a_torque

`SL axis_a_torque`
A axis torque.
Definition at line 1366 of file gclib.cs.

9.18.3.12 axis_a_variable

`SL axis_a_variable`
A User-defined variable (ZA).
Definition at line 1370 of file gclib.cs.

9.18.3.13 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 1365 of file gclib.cs.

9.18.3.14 axis_b_analog_in

`UW axis_b_analog_in`
B axis analog input.
Definition at line 1381 of file gclib.cs.

9.18.3.15 axis_b_aux_position

`SL axis_b_aux_position`
B axis auxiliary position.
Definition at line 1378 of file gclib.cs.

9.18.3.16 axis_b_motor_position

`SL axis_b_motor_position`
B axis motor position.
Definition at line 1376 of file gclib.cs.

9.18.3.17 axis_b_position_error

`SL axis_b_position_error`
B axis position error.
Definition at line 1377 of file gclib.cs.

9.18.3.18 axis_b_reference_position

`SL axis_b_reference_position`
B axis reference position.
Definition at line 1375 of file gclib.cs.

9.18.3.19 axis_b_reserved_0

`UB axis_b_reserved_0`
Reserved.
Definition at line 1382 of file gclib.cs.

9.18.3.20 axis_b_reserved_1

`UB axis_b_reserved_1`
Reserved.
Definition at line 1383 of file gclib.cs.

9.18.3.21 axis_b_status

`UW axis_b_status`
B axis status.
Definition at line 1372 of file gclib.cs.

9.18.3.22 axis_b_stop_code

`UB axis_b_stop_code`
B axis stop code.
Definition at line 1374 of file gclib.cs.

9.18.3.23 axis_b_switches

`UB axis_b_switches`
B axis switches.
Definition at line 1373 of file gclib.cs.

9.18.3.24 axis_b_torque

`SL axis_b_torque`
B axis torque.
Definition at line 1380 of file gclib.cs.

9.18.3.25 axis_b_variable

`SL axis_b_variable`
B User-defined variable (ZA).
Definition at line 1384 of file gclib.cs.

9.18.3.26 axis_b_velocity

`SL axis_b_velocity`
B axis velocity.
Definition at line 1379 of file gclib.cs.

9.18.3.27 axis_c_analog_in

`UW axis_c_analog_in`
C axis analog input.
Definition at line 1395 of file gclib.cs.

9.18.3.28 axis_c_aux_position

`SL axis_c_aux_position`
C axis auxiliary position.
Definition at line 1392 of file gclib.cs.

9.18.3.29 axis_c_motor_position

`SL axis_c_motor_position`
C axis motor position.
Definition at line 1390 of file gclib.cs.

9.18.3.30 axis_c_position_error

`SL axis_c_position_error`
C axis position error.
Definition at line 1391 of file gclib.cs.

9.18.3.31 axis_c_reference_position

`SL axis_c_reference_position`
C axis reference position.
Definition at line 1389 of file gclib.cs.

9.18.3.32 axis_c_reserved_0

`UB axis_c_reserved_0`
Reserved.
Definition at line 1396 of file gclib.cs.

9.18.3.33 axis_c_reserved_1

`UB axis_c_reserved_1`
Reserved.
Definition at line 1397 of file gclib.cs.

9.18.3.34 axis_c_status

`UW axis_c_status`
C axis status.
Definition at line 1386 of file gclib.cs.

9.18.3.35 axis_c_stop_code

`UB axis_c_stop_code`
C axis stop code.
Definition at line 1388 of file gclib.cs.

9.18.3.36 axis_c_switches

`UB axis_c_switches`
C axis switches.
Definition at line 1387 of file gclib.cs.

9.18.3.37 axis_c_torque

`SL axis_c_torque`
C axis torque.
Definition at line 1394 of file gclib.cs.

9.18.3.38 axis_c_variable

`SL axis_c_variable`
C User-defined variable (ZA).
Definition at line 1398 of file gclib.cs.

9.18.3.39 axis_c_velocity

`SL axis_c_velocity`
C axis velocity.
Definition at line 1393 of file gclib.cs.

9.18.3.40 axis_d_analog_in

`UW axis_d_analog_in`
D axis analog input.
Definition at line 1409 of file gclib.cs.

9.18.3.41 axis_d_aux_position

`SL axis_d_aux_position`
D axis auxiliary position.
Definition at line 1406 of file gclib.cs.

9.18.3.42 axis_d_motor_position

`SL axis_d_motor_position`
D axis motor position.
Definition at line 1404 of file gclib.cs.

9.18.3.43 axis_d_position_error

`SL axis_d_position_error`
D axis position error.
Definition at line 1405 of file gclib.cs.

9.18.3.44 axis_d_reference_position

`SL axis_d_reference_position`
D axis reference position.
Definition at line 1403 of file gclib.cs.

9.18.3.45 axis_d_reserved_0

`UB axis_d_reserved_0`
Reserved.
Definition at line 1410 of file gclib.cs.

9.18.3.46 axis_d_reserved_1

`UB axis_d_reserved_1`
Reserved.
Definition at line 1411 of file gclib.cs.

9.18.3.47 axis_d_status

`UW axis_d_status`
D axis status.
Definition at line 1400 of file gclib.cs.

9.18.3.48 axis_d_stop_code

`UB axis_d_stop_code`
D axis stop code.
Definition at line 1402 of file gclib.cs.

9.18.3.49 axis_d_switches

`UB axis_d_switches`
D axis switches.
Definition at line 1401 of file gclib.cs.

9.18.3.50 axis_d_torque

`SL axis_d_torque`
D axis torque.
Definition at line 1408 of file gclib.cs.

9.18.3.51 axis_d_variable

`SL axis_d_variable`
D User-defined variable (ZA).
Definition at line 1412 of file gclib.cs.

9.18.3.52 axis_d_velocity

`SL axis_d_velocity`
D axis velocity.
Definition at line 1407 of file gclib.cs.

9.18.3.53 axis_e_analog_in

`UW axis_e_analog_in`
E axis analog input.
Definition at line 1423 of file gclib.cs.

9.18.3.54 axis_e_aux_position

`SL axis_e_aux_position`
E axis auxiliary position.
Definition at line 1420 of file gclib.cs.

9.18.3.55 axis_e_motor_position

`SL axis_e_motor_position`
E axis motor position.
Definition at line 1418 of file gclib.cs.

9.18.3.56 axis_e_position_error

`SL axis_e_position_error`
E axis position error.
Definition at line 1419 of file gclib.cs.

9.18.3.57 axis_e_reference_position

`SL axis_e_reference_position`
E axis reference position.
Definition at line 1417 of file gclib.cs.

9.18.3.58 axis_e_reserved_0

`UB axis_e_reserved_0`
Reserved.
Definition at line 1424 of file gclib.cs.

9.18.3.59 axis_e_reserved_1

`UB axis_e_reserved_1`
Reserved.
Definition at line 1425 of file gclib.cs.

9.18.3.60 axis_e_status

`UW axis_e_status`
E axis status.
Definition at line 1414 of file gclib.cs.

9.18.3.61 axis_e_stop_code

`UB axis_e_stop_code`
E axis stop code.
Definition at line 1416 of file gclib.cs.

9.18.3.62 axis_e_switches

`UB axis_e_switches`
E axis switches.
Definition at line 1415 of file gclib.cs.

9.18.3.63 axis_e_torque

`SL axis_e_torque`
E axis torque.
Definition at line 1422 of file gclib.cs.

9.18.3.64 axis_e_variable

`SL axis_e_variable`
E User-defined variable (ZA).
Definition at line 1426 of file gclib.cs.

9.18.3.65 axis_e_velocity

`SL axis_e_velocity`
E axis velocity.
Definition at line 1421 of file gclib.cs.

9.18.3.66 axis_f_analog_in

`UW axis_f_analog_in`
F axis analog input.
Definition at line 1437 of file gclib.cs.

9.18.3.67 axis_f_aux_position

`SL axis_f_aux_position`
F axis auxiliary position.
Definition at line 1434 of file gclib.cs.

9.18.3.68 axis_f_motor_position

`SL axis_f_motor_position`
F axis motor position.
Definition at line 1432 of file gclib.cs.

9.18.3.69 axis_f_position_error

`SL axis_f_position_error`
F axis position error.
Definition at line 1433 of file gclib.cs.

9.18.3.70 axis_f_reference_position

`SL axis_f_reference_position`
F axis reference position.
Definition at line 1431 of file gclib.cs.

9.18.3.71 axis_f_reserved_0

`UB axis_f_reserved_0`
Reserved.
Definition at line 1438 of file gclib.cs.

9.18.3.72 axis_f_reserved_1

`UB axis_f_reserved_1`
Reserved.
Definition at line 1439 of file gclib.cs.

9.18.3.73 axis_f_status

`UW axis_f_status`
F axis status.
Definition at line 1428 of file gclib.cs.

9.18.3.74 axis_f_stop_code

`UB axis_f_stop_code`
F axis stop code.
Definition at line 1430 of file gclib.cs.

9.18.3.75 axis_f_switches

`UB axis_f_switches`
F axis switches.
Definition at line 1429 of file gclib.cs.

9.18.3.76 axis_f_torque

`SL axis_f_torque`
F axis torque.
Definition at line 1436 of file gclib.cs.

9.18.3.77 axis_f_variable

`SL axis_f_variable`
F User-defined variable (ZA).
Definition at line 1440 of file gclib.cs.

9.18.3.78 axis_f_velocity

`SL axis_f_velocity`
F axis velocity.
Definition at line 1435 of file gclib.cs.

9.18.3.79 axis_g_analog_in

`UW axis_g_analog_in`
G axis analog input.
Definition at line 1451 of file gclib.cs.

9.18.3.80 axis_g_aux_position

`SL axis_g_aux_position`
G axis auxiliary position.
Definition at line 1448 of file gclib.cs.

9.18.3.81 axis_g_motor_position

`SL axis_g_motor_position`
G axis motor position.
Definition at line 1446 of file gclib.cs.

9.18.3.82 axis_g_position_error

`SL axis_g_position_error`
G axis position error.
Definition at line 1447 of file gclib.cs.

9.18.3.83 axis_g_reference_position

`SL axis_g_reference_position`
G axis reference position.
Definition at line 1445 of file gclib.cs.

9.18.3.84 axis_g_reserved_0

`UB axis_g_reserved_0`
Reserved.
Definition at line 1452 of file gclib.cs.

9.18.3.85 axis_g_reserved_1

`UB axis_g_reserved_1`
Reserved.
Definition at line 1453 of file gclib.cs.

9.18.3.86 axis_g_status

`UW axis_g_status`
G axis status.
Definition at line 1442 of file gclib.cs.

9.18.3.87 axis_g_stop_code

`UB axis_g_stop_code`
G axis stop code.
Definition at line 1444 of file gclib.cs.

9.18.3.88 axis_g_switches

`UB axis_g_switches`
G axis switches.
Definition at line 1443 of file gclib.cs.

9.18.3.89 axis_g_torque

`SL axis_g_torque`
G axis torque.
Definition at line 1450 of file gclib.cs.

9.18.3.90 axis_g_variable

`SL axis_g_variable`
G User-defined variable (ZA).
Definition at line 1454 of file gclib.cs.

9.18.3.91 axis_g_velocity

`SL axis_g_velocity`
G axis velocity.
Definition at line 1449 of file gclib.cs.

9.18.3.92 axis_h_analog_in

`UW axis_h_analog_in`
H axis analog input.
Definition at line 1465 of file gclib.cs.

9.18.3.93 axis_h_aux_position

`SL axis_h_aux_position`
H axis auxiliary position.
Definition at line 1462 of file gclib.cs.

9.18.3.94 axis_h_motor_position

`SL axis_h_motor_position`
H axis motor position.
Definition at line 1460 of file gclib.cs.

9.18.3.95 axis_h_position_error

`SL axis_h_position_error`
H axis position error.
Definition at line 1461 of file gclib.cs.

9.18.3.96 axis_h_reference_position

`SL axis_h_reference_position`
H axis reference position.
Definition at line 1459 of file gclib.cs.

9.18.3.97 axis_h_reserved_0

`UB axis_h_reserved_0`
Reserved.
Definition at line 1466 of file gclib.cs.

9.18.3.98 axis_h_reserved_1

`UB axis_h_reserved_1`
Reserved.
Definition at line 1467 of file gclib.cs.

9.18.3.99 axis_h_status

`UW axis_h_status`
H axis status.
Definition at line 1456 of file gclib.cs.

9.18.3.100 axis_h_stop_code

`UB axis_h_stop_code`
H axis stop code.
Definition at line 1458 of file gclib.cs.

9.18.3.101 axis_h_switches

`UB axis_h_switches`
H axis switches.
Definition at line 1457 of file gclib.cs.

9.18.3.102 axis_h_torque

`SL axis_h_torque`
H axis torque.
Definition at line 1464 of file gclib.cs.

9.18.3.103 axis_h_variable

`SL axis_h_variable`
H User-defined variable (ZA).
Definition at line 1468 of file gclib.cs.

9.18.3.104 axis_h_velocity

`SL axis_h_velocity`
H axis velocity.
Definition at line 1463 of file gclib.cs.

9.18.3.105 contour_buffer_available

`UW contour_buffer_available`
Buffer space remaining, Contour Mode.
Definition at line 1346 of file gclib.cs.

9.18.3.106 contour_segment_count

`UL contour_segment_count`
Segment Count for Contour Mode.
Definition at line 1345 of file gclib.cs.

9.18.3.107 error_code

`UB error_code`
error code.
Definition at line 1341 of file gclib.cs.

9.18.3.108 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 1301 of file gclib.cs.

9.18.3.109 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 1302 of file gclib.cs.

9.18.3.110 input_bank_2

`UB input_bank_2`
general input bank 2 (inputs 17-24).
Definition at line 1303 of file gclib.cs.

9.18.3.111 input_bank_3

`UB input_bank_3`
general input bank 3 (inputs 25-32).
Definition at line 1304 of file gclib.cs.

9.18.3.112 input_bank_4

`UB input_bank_4`
general input bank 4 (inputs 33-40).
Definition at line 1305 of file gclib.cs.

9.18.3.113 input_bank_5

`UB input_bank_5`
general input bank 5 (inputs 41-48).
Definition at line 1306 of file gclib.cs.

9.18.3.114 input_bank_6

`UB input_bank_6`
general input bank 6 (inputs 49-56).
Definition at line 1307 of file gclib.cs.

9.18.3.115 input_bank_7

`UB input_bank_7`
general input bank 7 (inputs 57-64).
Definition at line 1308 of file gclib.cs.

9.18.3.116 input_bank_8

`UB input_bank_8`
general input bank 8 (inputs 65-72).
Definition at line 1309 of file gclib.cs.

9.18.3.117 input_bank_9

`UB input_bank_9`
general input bank 9 (inputs 73-80).
Definition at line 1310 of file gclib.cs.

9.18.3.118 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 1312 of file gclib.cs.

9.18.3.119 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 1313 of file gclib.cs.

9.18.3.120 output_bank_2

`UB output_bank_2`
general output bank 2 (outputs 17-24).
Definition at line 1314 of file gclib.cs.

9.18.3.121 output_bank_3

`UB output_bank_3`
general output bank 3 (outputs 25-32).
Definition at line 1315 of file gclib.cs.

9.18.3.122 output_bank_4

`UB output_bank_4`
general output bank 4 (outputs 33-40).
Definition at line 1316 of file gclib.cs.

9.18.3.123 output_bank_5

`UB output_bank_5`
general output bank 5 (outputs 41-48).
Definition at line 1317 of file gclib.cs.

9.18.3.124 output_bank_6

`UB output_bank_6`
general output bank 6 (outputs 49-56).
Definition at line 1318 of file gclib.cs.

9.18.3.125 output_bank_7

`UB output_bank_7`
general output bank 7 (outputs 57-64).
Definition at line 1319 of file gclib.cs.

9.18.3.126 output_bank_8

`UB output_bank_8`
general output bank 8 (outputs 65-72).
Definition at line 1320 of file gclib.cs.

9.18.3.127 output_bank_9

`UB output_bank_9`
general output bank 9 (outputs 73-80).
Definition at line 1321 of file gclib.cs.

9.18.3.128 reserved_0

`SW reserved_0`
Reserved.
Definition at line 1323 of file gclib.cs.

9.18.3.129 reserved_10

`SW reserved_10`
Reserved.
Definition at line 1328 of file gclib.cs.

9.18.3.130 reserved_12

`SW reserved_12`
Reserved.
Definition at line 1329 of file gclib.cs.

9.18.3.131 reserved_14

`SW reserved_14`
Reserved.
Definition at line 1330 of file gclib.cs.

9.18.3.132 reserved_16

`UB reserved_16`
Reserved.
Definition at line 1332 of file gclib.cs.

9.18.3.133 reserved_17

`UB reserved_17`
Reserved.
Definition at line 1333 of file gclib.cs.

9.18.3.134 reserved_18

`UB reserved_18`
Reserved.
Definition at line 1334 of file gclib.cs.

9.18.3.135 reserved_19

`UB reserved_19`
Reserved.
Definition at line 1335 of file gclib.cs.

9.18.3.136 reserved_2

`SW reserved_2`
Reserved.
Definition at line 1324 of file gclib.cs.

9.18.3.137 reserved_20

`UB reserved_20`
Reserved.
Definition at line 1336 of file gclib.cs.

9.18.3.138 reserved_21

`UB reserved_21`
Reserved.
Definition at line 1337 of file gclib.cs.

9.18.3.139 reserved_22

`UB reserved_22`
Reserved.
Definition at line 1338 of file gclib.cs.

9.18.3.140 reserved_23

`UB reserved_23`
Reserved.
Definition at line 1339 of file gclib.cs.

9.18.3.141 reserved_24

`UL reserved_24`
Reserved.
Definition at line 1343 of file gclib.cs.

9.18.3.142 reserved_4

`SW reserved_4`
Reserved.
Definition at line 1325 of file gclib.cs.

9.18.3.143 reserved_6

`SW reserved_6`
Reserved.
Definition at line 1326 of file gclib.cs.

9.18.3.144 reserved_8

`SW reserved_8`
Reserved.
Definition at line 1327 of file gclib.cs.

9.18.3.145 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 1350 of file gclib.cs.

9.18.3.146 s_plane_buffer_available

`UW s_plane_buffer_available`
Buffer space remaining, S Plane.
Definition at line 1351 of file gclib.cs.

9.18.3.147 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 1349 of file gclib.cs.

9.18.3.148 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 1348 of file gclib.cs.

9.18.3.149 sample_number

`UW sample_number`
sample number.
Definition at line 1299 of file gclib.cs.

9.18.3.150 t_distance

`SL t_distance`
distance traveled in coordinated move for T plane.
Definition at line 1355 of file gclib.cs.

9.18.3.151 t_plane_buffer_available

`UW t_plane_buffer_available`
Buffer space remaining, T Plane.
Definition at line 1356 of file gclib.cs.

9.18.3.152 t_plane_move_status

`UW t_plane_move_status`
Coordinated move status for T plane.
Definition at line 1354 of file gclib.cs.

9.18.3.153 t_plane_segment_count

`UW t_plane_segment_count`
segment count of coordinated move for T plane.
Definition at line 1353 of file gclib.cs.

9.18.3.154 thread_status

`UB thread_status`
thread status.
Definition at line 1342 of file gclib.cs.
The documentation for this struct was generated from the following file:

- gclib.cs

9.19 GDataRecord1806 Struct Reference

Data record struct for DMC-1806 controller.
`#include <gclib_record.h>`

Data Fields

- UW sample_number
  sample number.
- UB input_bank_0
  general input bank 0 (inputs 1-8).
- UB input_bank_1
  general input bank 1 (inputs 9-16).
- UB input_bank_2
  general input bank 2 (inputs 17-24).
- UB input_bank_3
  general input bank 3 (inputs 25-32).
- UB input_bank_4

general input bank 4 (inputs 33-40).

- UB input_bank_5

  general input bank 5 (inputs 41-48).
- UB input_bank_6

  general input bank 6 (inputs 49-56).
- UB input_bank_7

  general input bank 7 (inputs 57-64).
- UB input_bank_8

  general input bank 8 (inputs 65-72).
- UB input_bank_9

  general input bank 9 (inputs 73-80).
- UB output_bank_0

  general output bank 0 (outputs 1-8).
- UB output_bank_1

  general output bank 1 (outputs 9-16).
- UB output_bank_2

  general output bank 2 (outputs 17-24).
- UB output_bank_3

  general output bank 3 (outputs 25-32).
- UB output_bank_4

  general output bank 4 (outputs 33-40).
- UB output_bank_5

  general output bank 5 (outputs 41-48).
- UB output_bank_6

  general output bank 6 (outputs 49-56).
- UB output_bank_7

  general output bank 7 (outputs 57-64).
- UB output_bank_8

  general output bank 8 (outputs 65-72).
- UB output_bank_9

  general output bank 9 (outputs 73-80).
- SW reserved_0

  Reserved.
- SW reserved_2

  Reserved.
- SW reserved_4

  Reserved.
- SW reserved_6

  Reserved.
- SW reserved_8

  Reserved.
- SW reserved_10

  Reserved.
- SW reserved_12

  Reserved.
- SW reserved_14

  Reserved.
- UB reserved_16

  Reserved.
- UB reserved_17

  Reserved.

- UB reserved_18

  Reserved.
- UB reserved_19

  Reserved.
- UB reserved_20

  Reserved.
- UB reserved_21

  Reserved.
- UB reserved_22

  Reserved.
- UB reserved_23

  Reserved.
- UB error_code

  error code.
- UB thread_status

  thread status.
- UL reserved_24

  Reserved.
- UL contour_segment_count

  Segment Count for Contour Mode.
- UW contour_buffer_available

  Buffer space remaining, Contour Mode.
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

  coordinated move status for S plane.
- SL s_distance

  distance traveled in coordinated move for S plane.
- UW s_plane_buffer_available

  Buffer space remaining, S Plane.
- UW t_plane_segment_count

  segment count of coordinated move for T plane.
- UW t_plane_move_status

  Coordinated move status for T plane.
- SL t_distance

  distance traveled in coordinated move for T plane.
- UW t_plane_buffer_available

  Buffer space remaining, T Plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

  A axis stop code.
- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

A axis auxiliary position.

- SL axis_a_velocity

  A axis velocity.
- SL axis_a_torque

  A axis torque.
- UW axis_a_analog_in

  A axis analog input.
- UB axis_a_reserved_0

  Reserved.
- UB axis_a_reserved_1

  Reserved.
- SL axis_a_variable

  A User-defined variable (ZA).
- UW axis_b_status

  B axis status.
- UB axis_b_switches

  B axis switches.
- UB axis_b_stop_code

  B axis stop code.
- SL axis_b_reference_position

  B axis reference position.
- SL axis_b_motor_position

  B axis motor position.
- SL axis_b_position_error

  B axis position error.
- SL axis_b_aux_position

  B axis auxiliary position.
- SL axis_b_velocity

  B axis velocity.
- SL axis_b_torque

  B axis torque.
- UW axis_b_analog_in

  B axis analog input.
- UB axis_b_reserved_0

  Reserved.
- UB axis_b_reserved_1

  Reserved.
- SL axis_b_variable

  B User-defined variable (ZA).
- UW axis_c_status

  C axis status.
- UB axis_c_switches

  C axis switches.
- UB axis_c_stop_code

  C axis stop code.
- SL axis_c_reference_position

  C axis reference position.
- SL axis_c_motor_position

  C axis motor position.
- SL axis_c_position_error

  C axis position error.

- SL axis_c_aux_position

  C axis auxiliary position.
- SL axis_c_velocity

  C axis velocity.
- SL axis_c_torque

  C axis torque.
- UW axis_c_analog_in

  C axis analog input.
- UB axis_c_reserved_0

  Reserved.
- UB axis_c_reserved_1

  Reserved.
- SL axis_c_variable

  C User-defined variable (ZA).
- UW axis_d_status

  D axis status.
- UB axis_d_switches

  D axis switches.
- UB axis_d_stop_code

  D axis stop code.
- SL axis_d_reference_position

  D axis reference position.
- SL axis_d_motor_position

  D axis motor position.
- SL axis_d_position_error

  D axis position error.
- SL axis_d_aux_position

  D axis auxiliary position.
- SL axis_d_velocity

  D axis velocity.
- SL axis_d_torque

  D axis torque.
- UW axis_d_analog_in

  D axis analog input.
- UB axis_d_reserved_0

  Reserved.
- UB axis_d_reserved_1

  Reserved.
- SL axis_d_variable

  D User-defined variable (ZA).
- UW axis_e_status

  E axis status.
- UB axis_e_switches

  E axis switches.
- UB axis_e_stop_code

  E axis stop code.
- SL axis_e_reference_position

  E axis reference position.
- SL axis_e_motor_position

  E axis motor position.
- SL axis_e_position_error

E axis position error.

- SL axis_e_aux_position

  E axis auxiliary position.
- SL axis_e_velocity

  E axis velocity.
- SL axis_e_torque

  E axis torque.
- UW axis_e_analog_in

  E axis analog input.
- UB axis_e_reserved_0

  Reserved.
- UB axis_e_reserved_1

  Reserved.
- SL axis_e_variable

  E User-defined variable (ZA).
- UW axis_f_status

  F axis status.
- UB axis_f_switches

  F axis switches.
- UB axis_f_stop_code

  F axis stop code.
- SL axis_f_reference_position

  F axis reference position.
- SL axis_f_motor_position

  F axis motor position.
- SL axis_f_position_error

  F axis position error.
- SL axis_f_aux_position

  F axis auxiliary position.
- SL axis_f_velocity

  F axis velocity.
- SL axis_f_torque

  F axis torque.
- UW axis_f_analog_in

  F axis analog input.
- UB axis_f_reserved_0

  Reserved.
- UB axis_f_reserved_1

  Reserved.
- SL axis_f_variable

  F User-defined variable (ZA).
- UW axis_g_status

  G axis status.
- UB axis_g_switches

  G axis switches.
- UB axis_g_stop_code

  G axis stop code.
- SL axis_g_reference_position

  G axis reference position.
- SL axis_g_motor_position

  G axis motor position.

- SL axis_g_position_error
  G axis position error.
- SL axis_g_aux_position
  G axis auxiliary position.
- SL axis_g_velocity
  G axis velocity.
- SL axis_g_torque
  G axis torque.
- UW axis_g_analog_in
  G axis analog input.
- UB axis_g_reserved_0
  Reserved.
- UB axis_g_reserved_1
  Reserved.
- SL axis_g_variable
  G User-defined variable (ZA).
- UW axis_h_status
  H axis status.
- UB axis_h_switches
  H axis switches.
- UB axis_h_stop_code
  H axis stop code.
- SL axis_h_reference_position
  H axis reference position.
- SL axis_h_motor_position
  H axis motor position.
- SL axis_h_position_error
  H axis position error.
- SL axis_h_aux_position
  H axis auxiliary position.
- SL axis_h_velocity
  H axis velocity.
- SL axis_h_torque
  H axis torque.
- UW axis_h_analog_in
  H axis analog input.
- UB axis_h_reserved_0
  Reserved.
- UB axis_h_reserved_1
  Reserved.
- SL axis_h_variable
  H User-defined variable (ZA).

9.19.1 Detailed Description

Data record struct for DMC-1806 controller.
The 18x6 Data record is the same as 4000 except the following.

1. No header bytes. Firmware strips it in DR. Software removes it from QR.
2. No Ethernet status (bytes 42-49).
3. No amplfifier status (bytes 52-55).
4. No axis-specific hall input status.

Definition at line 409 of file gclib_record.h.

9.19.2 Field Documentation

9.19.2.1 axis_a_analog_in

`UW axis_a_analog_in`
A axis analog input.
Definition at line 481 of file gclib_record.h.

9.19.2.2 axis_a_aux_position

`SL axis_a_aux_position`
A axis auxiliary position.
Definition at line 478 of file gclib_record.h.

9.19.2.3 axis_a_motor_position

`SL axis_a_motor_position`
A axis motor position.
Definition at line 476 of file gclib_record.h.

9.19.2.4 axis_a_position_error

`SL axis_a_position_error`
A axis position error.
Definition at line 477 of file gclib_record.h.

9.19.2.5 axis_a_reference_position

`SL axis_a_reference_position`
A axis reference position.
Definition at line 475 of file gclib_record.h.

9.19.2.6 axis_a_reserved_0

`UB axis_a_reserved_0`
Reserved.
Definition at line 482 of file gclib_record.h.

9.19.2.7 axis_a_reserved_1

`UB axis_a_reserved_1`
Reserved.
Definition at line 483 of file gclib_record.h.

9.19.2.8 axis_a_status

`UW axis_a_status`
A axis status.
Definition at line 472 of file gclib_record.h.

9.19.2.9 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 474 of file gclib_record.h.

9.19.2.10 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 473 of file gclib_record.h.

9.19.2.11 axis_a_torque

`SL axis_a_torque`
A axis torque.
Definition at line 480 of file gclib_record.h.

9.19.2.12 axis_a_variable

`SL axis_a_variable`
A User-defined variable (ZA).
Definition at line 484 of file gclib_record.h.

9.19.2.13 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 479 of file gclib_record.h.

9.19.2.14 axis_b_analog_in

`UW axis_b_analog_in`
B axis analog input.
Definition at line 495 of file gclib_record.h.

9.19.2.15 axis_b_aux_position

`SL axis_b_aux_position`
B axis auxiliary position.
Definition at line 492 of file gclib_record.h.

9.19.2.16 axis_b_motor_position

`SL axis_b_motor_position`
B axis motor position.
Definition at line 490 of file gclib_record.h.

9.19.2.17 axis_b_position_error

`SL axis_b_position_error`
B axis position error.
Definition at line 491 of file gclib_record.h.

9.19.2.18 axis_b_reference_position

`SL axis_b_reference_position`
B axis reference position.
Definition at line 489 of file gclib_record.h.

9.19.2.19 axis_b_reserved_0

`UB axis_b_reserved_0`
Reserved.
Definition at line 496 of file gclib_record.h.

9.19.2.20 axis_b_reserved_1

`UB axis_b_reserved_1`
Reserved.
Definition at line 497 of file gclib_record.h.

9.19.2.21 axis_b_status

`UW axis_b_status`
B axis status.
Definition at line 486 of file gclib_record.h.

9.19.2.22 axis_b_stop_code

`UB axis_b_stop_code`
B axis stop code.
Definition at line 488 of file gclib_record.h.

9.19.2.23 axis_b_switches

`UB axis_b_switches`
B axis switches.
Definition at line 487 of file gclib_record.h.

9.19.2.24 axis_b_torque

`SL axis_b_torque`
B axis torque.
Definition at line 494 of file gclib_record.h.

9.19.2.25 axis_b_variable

`SL axis_b_variable`
B User-defined variable (ZA).
Definition at line 498 of file gclib_record.h.

9.19.2.26 axis_b_velocity

`SL axis_b_velocity`
B axis velocity.
Definition at line 493 of file gclib_record.h.

9.19.2.27 axis_c_analog_in

`UW axis_c_analog_in`
C axis analog input.
Definition at line 509 of file gclib_record.h.

9.19.2.28 axis_c_aux_position

`SL axis_c_aux_position`
C axis auxiliary position.
Definition at line 506 of file gclib_record.h.

9.19.2.29 axis_c_motor_position

`SL axis_c_motor_position`
C axis motor position.
Definition at line 504 of file gclib_record.h.

9.19.2.30 axis_c_position_error

`SL axis_c_position_error`
C axis position error.
Definition at line 505 of file gclib_record.h.

9.19.2.31 axis_c_reference_position

`SL axis_c_reference_position`
C axis reference position.
Definition at line 503 of file gclib_record.h.

9.19.2.32 axis_c_reserved_0

`UB axis_c_reserved_0`
Reserved.
Definition at line 510 of file gclib_record.h.

9.19.2.33 axis_c_reserved_1

`UB axis_c_reserved_1`
Reserved.
Definition at line 511 of file gclib_record.h.

9.19.2.34 axis_c_status

`UW axis_c_status`
C axis status.
Definition at line 500 of file gclib_record.h.

9.19.2.35 axis_c_stop_code

`UB axis_c_stop_code`
C axis stop code.
Definition at line 502 of file gclib_record.h.

9.19.2.36 axis_c_switches

`UB axis_c_switches`
C axis switches.
Definition at line 501 of file gclib_record.h.

9.19.2.37 axis_c_torque

`SL axis_c_torque`
C axis torque.
Definition at line 508 of file gclib_record.h.

9.19.2.38 axis_c_variable

`SL axis_c_variable`
C User-defined variable (ZA).
Definition at line 512 of file gclib_record.h.

9.19.2.39 axis_c_velocity

`SL axis_c_velocity`
C axis velocity.
Definition at line 507 of file gclib_record.h.

9.19.2.40 axis_d_analog_in

`UW axis_d_analog_in`
D axis analog input.
Definition at line 523 of file gclib_record.h.

9.19.2.41 axis_d_aux_position

`SL axis_d_aux_position`
D axis auxiliary position.
Definition at line 520 of file gclib_record.h.

9.19.2.42 axis_d_motor_position

`SL axis_d_motor_position`
D axis motor position.
Definition at line 518 of file gclib_record.h.

9.19.2.43 axis_d_position_error

`SL axis_d_position_error`
D axis position error.
Definition at line 519 of file gclib_record.h.

9.19.2.44 axis_d_reference_position

`SL axis_d_reference_position`
D axis reference position.
Definition at line 517 of file gclib_record.h.

9.19.2.45 axis_d_reserved_0

`UB axis_d_reserved_0`
Reserved.
Definition at line 524 of file gclib_record.h.

9.19.2.46 axis_d_reserved_1

`UB axis_d_reserved_1`
Reserved.
Definition at line 525 of file gclib_record.h.

9.19.2.47 axis_d_status

`UW axis_d_status`
D axis status.
Definition at line 514 of file gclib_record.h.

9.19.2.48 axis_d_stop_code

`UB axis_d_stop_code`
D axis stop code.
Definition at line 516 of file gclib_record.h.

9.19.2.49 axis_d_switches

`UB axis_d_switches`
D axis switches.
Definition at line 515 of file gclib_record.h.

9.19.2.50 axis_d_torque

`SL axis_d_torque`
D axis torque.
Definition at line 522 of file gclib_record.h.

9.19.2.51 axis_d_variable

`SL axis_d_variable`
D User-defined variable (ZA).
Definition at line 526 of file gclib_record.h.

9.19.2.52 axis_d_velocity

`SL axis_d_velocity`
D axis velocity.
Definition at line 521 of file gclib_record.h.

9.19.2.53 axis_e_analog_in

`UW axis_e_analog_in`
E axis analog input.
Definition at line 537 of file gclib_record.h.

9.19.2.54 axis_e_aux_position

`SL axis_e_aux_position`
E axis auxiliary position.
Definition at line 534 of file gclib_record.h.

9.19.2.55 axis_e_motor_position

`SL axis_e_motor_position`
E axis motor position.
Definition at line 532 of file gclib_record.h.

9.19.2.56 axis_e_position_error

`SL axis_e_position_error`
E axis position error.
Definition at line 533 of file gclib_record.h.

9.19.2.57 axis_e_reference_position

`SL axis_e_reference_position`
E axis reference position.
Definition at line 531 of file gclib_record.h.

9.19.2.58 axis_e_reserved_0

`UB axis_e_reserved_0`
Reserved.
Definition at line 538 of file gclib_record.h.

9.19.2.59 axis_e_reserved_1

`UB axis_e_reserved_1`
Reserved.
Definition at line 539 of file gclib_record.h.

9.19.2.60 axis_e_status

`UW axis_e_status`
E axis status.
Definition at line 528 of file gclib_record.h.

9.19.2.61 axis_e_stop_code

`UB axis_e_stop_code`
E axis stop code.
Definition at line 530 of file gclib_record.h.

9.19.2.62 axis_e_switches

`UB axis_e_switches`
E axis switches.
Definition at line 529 of file gclib_record.h.

9.19.2.63 axis_e_torque

`SL axis_e_torque`
E axis torque.
Definition at line 536 of file gclib_record.h.

9.19.2.64 axis_e_variable

`SL axis_e_variable`
E User-defined variable (ZA).
Definition at line 540 of file gclib_record.h.

9.19.2.65 axis_e_velocity

`SL axis_e_velocity`
E axis velocity.
Definition at line 535 of file gclib_record.h.

9.19.2.66 axis_f_analog_in

`UW axis_f_analog_in`
F axis analog input.
Definition at line 551 of file gclib_record.h.

9.19.2.67 axis_f_aux_position

`SL axis_f_aux_position`
F axis auxiliary position.
Definition at line 548 of file gclib_record.h.

9.19.2.68 axis_f_motor_position

`SL axis_f_motor_position`
F axis motor position.
Definition at line 546 of file gclib_record.h.

9.19.2.69 axis_f_position_error

`SL axis_f_position_error`
F axis position error.
Definition at line 547 of file gclib_record.h.

9.19.2.70 axis_f_reference_position

`SL axis_f_reference_position`
F axis reference position.
Definition at line 545 of file gclib_record.h.

9.19.2.71 axis_f_reserved_0

`UB axis_f_reserved_0`
Reserved.
Definition at line 552 of file gclib_record.h.

9.19.2.72 axis_f_reserved_1

`UB axis_f_reserved_1`
Reserved.
Definition at line 553 of file gclib_record.h.

9.19.2.73 axis_f_status

`UW axis_f_status`
F axis status.
Definition at line 542 of file gclib_record.h.

9.19.2.74 axis_f_stop_code

`UB axis_f_stop_code`
F axis stop code.
Definition at line 544 of file gclib_record.h.

9.19.2.75 axis_f_switches

`UB axis_f_switches`
F axis switches.
Definition at line 543 of file gclib_record.h.

9.19.2.76 axis_f_torque

`SL axis_f_torque`
F axis torque.
Definition at line 550 of file gclib_record.h.

9.19.2.77 axis_f_variable

`SL axis_f_variable`
F User-defined variable (ZA).
Definition at line 554 of file gclib_record.h.

9.19.2.78 axis_f_velocity

`SL axis_f_velocity`
F axis velocity.
Definition at line 549 of file gclib_record.h.

9.19.2.79 axis_g_analog_in

`UW axis_g_analog_in`
G axis analog input.
Definition at line 565 of file gclib_record.h.

9.19.2.80 axis_g_aux_position

`SL axis_g_aux_position`
G axis auxiliary position.
Definition at line 562 of file gclib_record.h.

9.19.2.81 axis_g_motor_position

`SL axis_g_motor_position`
G axis motor position.
Definition at line 560 of file gclib_record.h.

9.19.2.82 axis_g_position_error

`SL axis_g_position_error`
G axis position error.
Definition at line 561 of file gclib_record.h.

9.19.2.83 axis_g_reference_position

`SL axis_g_reference_position`
G axis reference position.
Definition at line 559 of file gclib_record.h.

9.19.2.84 axis_g_reserved_0

`UB axis_g_reserved_0`
Reserved.
Definition at line 566 of file gclib_record.h.

9.19.2.85 axis_g_reserved_1

`UB axis_g_reserved_1`
Reserved.
Definition at line 567 of file gclib_record.h.

9.19.2.86 axis_g_status

`UW axis_g_status`
G axis status.
Definition at line 556 of file gclib_record.h.

9.19.2.87 axis_g_stop_code

`UB axis_g_stop_code`
G axis stop code.
Definition at line 558 of file gclib_record.h.

9.19.2.88 axis_g_switches

`UB axis_g_switches`
G axis switches.
Definition at line 557 of file gclib_record.h.

9.19.2.89 axis_g_torque

`SL axis_g_torque`
G axis torque.
Definition at line 564 of file gclib_record.h.

9.19.2.90 axis_g_variable

`SL axis_g_variable`
G User-defined variable (ZA).
Definition at line 568 of file gclib_record.h.

9.19.2.91 axis_g_velocity

`SL axis_g_velocity`
G axis velocity.
Definition at line 563 of file gclib_record.h.

9.19.2.92 axis_h_analog_in

`UW axis_h_analog_in`
H axis analog input.
Definition at line 579 of file gclib_record.h.

9.19.2.93 axis_h_aux_position

`SL axis_h_aux_position`
H axis auxiliary position.
Definition at line 576 of file gclib_record.h.

9.19.2.94 axis_h_motor_position

`SL axis_h_motor_position`
H axis motor position.
Definition at line 574 of file gclib_record.h.

9.19.2.95 axis_h_position_error

`SL axis_h_position_error`
H axis position error.
Definition at line 575 of file gclib_record.h.

9.19.2.96 axis_h_reference_position

`SL axis_h_reference_position`
H axis reference position.
Definition at line 573 of file gclib_record.h.

9.19.2.97 axis_h_reserved_0

`UB axis_h_reserved_0`
Reserved.
Definition at line 580 of file gclib_record.h.

9.19.2.98 axis_h_reserved_1

`UB axis_h_reserved_1`
Reserved.
Definition at line 581 of file gclib_record.h.

9.19.2.99 axis_h_status

`UW axis_h_status`
H axis status.
Definition at line 570 of file gclib_record.h.

9.19.2.100 axis_h_stop_code

`UB axis_h_stop_code`
H axis stop code.
Definition at line 572 of file gclib_record.h.

9.19.2.101 axis_h_switches

`UB axis_h_switches`
H axis switches.
Definition at line 571 of file gclib_record.h.

9.19.2.102 axis_h_torque

`SL axis_h_torque`
H axis torque.
Definition at line 578 of file gclib_record.h.

9.19.2.103 axis_h_variable

`SL axis_h_variable`
H User-defined variable (ZA).
Definition at line 582 of file gclib_record.h.

9.19.2.104 axis_h_velocity

`SL axis_h_velocity`
H axis velocity.
Definition at line 577 of file gclib_record.h.

9.19.2.105 contour_buffer_available

`UW contour_buffer_available`
Buffer space remaining, Contour Mode.
Definition at line 460 of file gclib_record.h.

9.19.2.106 contour_segment_count

`UL contour_segment_count`
Segment Count for Contour Mode.
Definition at line 459 of file gclib_record.h.

9.19.2.107 error_code

`UB error_code`
error code.
Definition at line 455 of file gclib_record.h.

9.19.2.108 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 415 of file gclib_record.h.

9.19.2.109 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 416 of file gclib_record.h.

9.19.2.110 input_bank_2

`UB input_bank_2`
general input bank 2 (inputs 17-24).
Definition at line 417 of file gclib_record.h.

9.19.2.111 input_bank_3

`UB input_bank_3`
general input bank 3 (inputs 25-32).
Definition at line 418 of file gclib_record.h.

9.19.2.112 input_bank_4

`UB input_bank_4`
general input bank 4 (inputs 33-40).
Definition at line 419 of file gclib_record.h.

9.19.2.113 input_bank_5

`UB input_bank_5`
general input bank 5 (inputs 41-48).
Definition at line 420 of file gclib_record.h.

9.19.2.114 input_bank_6

`UB input_bank_6`
general input bank 6 (inputs 49-56).
Definition at line 421 of file gclib_record.h.

9.19.2.115 input_bank_7

`UB input_bank_7`
general input bank 7 (inputs 57-64).
Definition at line 422 of file gclib_record.h.

9.19.2.116 input_bank_8

`UB input_bank_8`
general input bank 8 (inputs 65-72).
Definition at line 423 of file gclib_record.h.

9.19.2.117 input_bank_9

`UB input_bank_9`
general input bank 9 (inputs 73-80).
Definition at line 424 of file gclib_record.h.

9.19.2.118 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 426 of file gclib_record.h.

9.19.2.119 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 427 of file gclib_record.h.

9.19.2.120 output_bank_2

`UB output_bank_2`
general output bank 2 (outputs 17-24).
Definition at line 428 of file gclib_record.h.

9.19.2.121 output_bank_3

`UB output_bank_3`
general output bank 3 (outputs 25-32).
Definition at line 429 of file gclib_record.h.

9.19.2.122 output_bank_4

`UB output_bank_4`
general output bank 4 (outputs 33-40).
Definition at line 430 of file gclib_record.h.

9.19.2.123 output_bank_5

`UB output_bank_5`
general output bank 5 (outputs 41-48).
Definition at line 431 of file gclib_record.h.

9.19.2.124 output_bank_6

`UB output_bank_6`
general output bank 6 (outputs 49-56).
Definition at line 432 of file gclib_record.h.

9.19.2.125 output_bank_7

`UB output_bank_7`
general output bank 7 (outputs 57-64).
Definition at line 433 of file gclib_record.h.

9.19.2.126 output_bank_8

`UB output_bank_8`
general output bank 8 (outputs 65-72).
Definition at line 434 of file gclib_record.h.

9.19.2.127 output_bank_9

`UB output_bank_9`
general output bank 9 (outputs 73-80).
Definition at line 435 of file gclib_record.h.

9.19.2.128 reserved_0

`SW reserved_0`
Reserved.
Definition at line 437 of file gclib_record.h.

9.19.2.129 reserved_10

`SW reserved_10`
Reserved.
Definition at line 442 of file gclib_record.h.

9.19.2.130 reserved_12

`SW reserved_12`
Reserved.
Definition at line 443 of file gclib_record.h.

9.19.2.131 reserved_14

`SW reserved_14`
Reserved.
Definition at line 444 of file gclib_record.h.

9.19.2.132 reserved_16

`UB reserved_16`
Reserved.
Definition at line 446 of file gclib_record.h.

9.19.2.133 reserved_17

`UB reserved_17`
Reserved.
Definition at line 447 of file gclib_record.h.

9.19.2.134 reserved_18

`UB reserved_18`
Reserved.
Definition at line 448 of file gclib_record.h.

9.19.2.135 reserved_19

`UB reserved_19`
Reserved.
Definition at line 449 of file gclib_record.h.

9.19.2.136 reserved_2

`SW reserved_2`
Reserved.
Definition at line 438 of file gclib_record.h.

9.19.2.137 reserved_20

`UB reserved_20`
Reserved.
Definition at line 450 of file gclib_record.h.

9.19.2.138 reserved_21

`UB reserved_21`
Reserved.
Definition at line 451 of file gclib_record.h.

9.19.2.139 reserved_22

`UB reserved_22`
Reserved.
Definition at line 452 of file gclib_record.h.

9.19.2.140 reserved_23

`UB reserved_23`
Reserved.
Definition at line 453 of file gclib_record.h.

9.19.2.141 reserved_24

`UL reserved_24`
Reserved.
Definition at line 457 of file gclib_record.h.

9.19.2.142 reserved_4

`SW reserved_4`
Reserved.
Definition at line 439 of file gclib_record.h.

9.19.2.143 reserved_6

`SW reserved_6`
Reserved.
Definition at line 440 of file gclib_record.h.

9.19.2.144 reserved_8

`SW reserved_8`
Reserved.
Definition at line 441 of file gclib_record.h.

9.19.2.145 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 464 of file gclib_record.h.

9.19.2.146 s_plane_buffer_available

`UW s_plane_buffer_available`
Buffer space remaining, S Plane.
Definition at line 465 of file gclib_record.h.

9.19.2.147 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 463 of file gclib_record.h.

9.19.2.148 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 462 of file gclib_record.h.

9.19.2.149 sample_number

`UW sample_number`
sample number.
Definition at line 413 of file gclib_record.h.

9.19.2.150 t_distance

`SL t_distance`
distance traveled in coordinated move for T plane.
Definition at line 469 of file gclib_record.h.

9.19.2.151 t_plane_buffer_available

`UW t_plane_buffer_available`
Buffer space remaining, T Plane.
Definition at line 470 of file gclib_record.h.

9.19.2.152 t_plane_move_status

`UW t_plane_move_status`
Coordinated move status for T plane.
Definition at line 468 of file gclib_record.h.

9.19.2.153 t_plane_segment_count

`UW t_plane_segment_count`
segment count of coordinated move for T plane.
Definition at line 467 of file gclib_record.h.

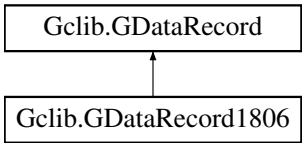
9.19.2.154 thread_status

`UB thread_status`
thread status.
Definition at line 456 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

9.20 Gclib.GDataRecord1806 Struct Reference

Inheritance diagram for Gclib.GDataRecord1806:



Public Member Functions

- general input **bank** (inputs 1-8). ∗/public UB input_bank_1
  /∗03
- general input **bank** (inputs 9-16). ∗/public UB input_bank_2
  /∗04
- general input **bank** (inputs 17-24). ∗/public UB input_bank_3
  /∗05
- general input **bank** (inputs 25-32). ∗/public UB input_bank_4
  /∗06
- general input **bank** (inputs 33-40). ∗/public UB input_bank_5
  /∗07
- general input **bank** (inputs 41-48). ∗/public UB input_bank_6
  /∗08
- general input **bank** (inputs 49-56). ∗/public UB input_bank_7
  /∗09
- general input **bank** (inputs 57-64). ∗/public UB input_bank_8
  /∗10
- general input **bank** (inputs 65-72). ∗/public UB input_bank_9

/*11

- general input **bank** (inputs 73-80). */public UB output_bank_0

  /*12
- general output **bank** (outputs 1-8). */public UB output_bank_1

  /*13
- general output **bank** (outputs 9-16). */public UB output_bank_2

  /*14
- general output **bank** (outputs 17-24). */public UB output_bank_3

  /*15
- general output **bank** (outputs 25-32). */public UB output_bank_4

  /*16
- general output **bank** (outputs 33-40). */public UB output_bank_5

  /*17
- general output **bank** (outputs 41-48). */public UB output_bank_6

  /*18
- general output **bank** (outputs 49-56). */public UB output_bank_7

  /*19
- general output **bank** (outputs 57-64). */public UB output_bank_8

  /*20
- general output **bank** (outputs 65-72). */public UB output_bank_9

  /*21
- general output **bank** (outputs 73-80). */public SW reserved_0

  /*22-23
- H User defined variable.*[] byte_array ()

Data Fields

- UW sample_number

  /*00-01
- sample number.*UB input_bank_0

  /*02
- Reserved.*SW reserved_2

  /*24-25
- Reserved.*SW reserved_4

  /*26-27
- Reserved.*SW reserved_6

  /*28-29
- Reserved.*SW reserved_8

  /*30-31
- Reserved.*SW reserved_10

  /*32-33
- Reserved.*SW reserved_12

  /*34-35
- Reserved.*SW reserved_14

  /*36-37
- Reserved.*UB reserved_16

  /*38
- Reserved.*UB reserved_17

  /*39
- Reserved.*UB reserved_18

  /*40

- Reserved.∗UB reserved_19
  /∗41
- Reserved.∗UB reserved_20
  /∗42
- Reserved.∗UB reserved_21
  /∗43
- Reserved.∗UB reserved_22
  /∗44
- Reserved.∗UB reserved_23
  /∗45
- Reserved.∗UB error_code
  /∗46
- error code.∗UB thread_status
  /∗47
- thread status.∗UL reserved_24
  /∗48-51
- Reserved.∗UL contour_segment_count
  /∗52-55
- Segment Count for Contour Mode.∗UW contour_buffer_available
  /∗56-57
- Buffer space remaining
- Buffer space Contour Mode.∗UW s_plane_segment_count
  /∗58-59
- segment count of coordinated move for S plane.∗UW s_plane_move_status
  /∗60-61
- coordinated move status for S plane.∗SL s_distance
  /∗62-65
- distance traveled in coordinated move for S plane.∗UW s_plane_buffer_available
  /∗66-67
- Buffer space S Plane.∗UW t_plane_segment_count
  /∗68-69
- segment count of coordinated move for T plane.∗UW t_plane_move_status
  /∗70-71
- Coordinated move status for T plane.∗SL t_distance
  /∗72-75
- distance traveled in coordinated move for T plane.∗UW t_plane_buffer_available
  /∗76-77
- Buffer space T Plane.∗UW axis_a_status
  /∗78-79
- A axis status.∗UB axis_a_switches
  /∗80
- A axis switches.∗UB axis_a_stop_code
  /∗81
- A axis stop code.∗SL axis_a_reference_position
  /∗82-85
- A axis reference position.∗SL axis_a_motor_position
  /∗86-89
- A axis motor position.∗SL axis_a_position_error
  /∗90-93
- A axis position error.∗SL axis_a_aux_position
  /∗94-97

- A axis auxiliary position.∗SL axis_a_velocity
  /∗98-101
- A axis velocity.∗SL axis_a_torque
  /∗102-105
- A axis torque.∗UW axis_a_analog_in
  /∗106-107
- A axis analog input.∗UB axis_a_reserved_0
  /∗108
- Reserved.∗UB axis_a_reserved_1
  /∗109
- Reserved.∗SL axis_a_variable
  /∗110-113
- A User defined variable.∗[] UW axis_b_status
  /∗114-115
- B axis status.∗UB axis_b_switches
  /∗116
- B axis switches.∗UB axis_b_stop_code
  /∗117
- B axis stop code.∗SL axis_b_reference_position
  /∗118-121
- B axis reference position.∗SL axis_b_motor_position
  /∗122-125
- B axis motor position.∗SL axis_b_position_error
  /∗126-129
- B axis position error.∗SL axis_b_aux_position
  /∗130-133
- B axis auxiliary position.∗SL axis_b_velocity
  /∗134-137
- B axis velocity.∗SL axis_b_torque
  /∗138-141
- B axis torque.∗UW axis_b_analog_in
  /∗142-143
- B axis analog input.∗UB axis_b_reserved_0
  /∗144
- Reserved.∗UB axis_b_reserved_1
  /∗145
- Reserved.∗SL axis_b_variable
  /∗146-149
- B User defined variable.∗[] UW axis_c_status
  /∗150-151
- C axis status.∗UB axis_c_switches
  /∗152
- C axis switches.∗UB axis_c_stop_code
  /∗153
- C axis stop code.∗SL axis_c_reference_position
  /∗154-157
- C axis reference position.∗SL axis_c_motor_position
  /∗158-161
- C axis motor position.∗SL axis_c_position_error
  /∗162-165
- C axis position error.∗SL axis_c_aux_position

/*166-169

- C axis auxiliary position.*SL axis_c_velocity

  /*170-173
- C axis velocity.*SL axis_c_torque

  /*174-177
- C axis torque.*UW axis_c_analog_in

  /*178-179
- C axis analog input.*UB axis_c_reserved_0

  /*180
- Reserved.*UB axis_c_reserved_1

  /*181
- Reserved.*SL axis_c_variable

  /*182-185
- C User defined variable.*[] UW axis_d_status

  /*186-187
- D axis status.*UB axis_d_switches

  /*188
- D axis switches.*UB axis_d_stop_code

  /*189
- D axis stop code.*SL axis_d_reference_position

  /*190-193
- D axis reference position.*SL axis_d_motor_position

  /*194-197
- D axis motor position.*SL axis_d_position_error

  /*198-201
- D axis position error.*SL axis_d_aux_position

  /*202-205
- D axis auxiliary position.*SL axis_d_velocity

  /*206-209
- D axis velocity.*SL axis_d_torque

  /*210-213
- D axis torque.*UW axis_d_analog_in

  /*214-215
- D axis analog input.*UB axis_d_reserved_0

  /*216
- Reserved.*UB axis_d_reserved_1

  /*217
- Reserved.*SL axis_d_variable

  /*218-221
- D User defined variable.*[] UW axis_e_status

  /*222-223
- E axis status.*UB axis_e_switches

  /*224
- E axis switches.*UB axis_e_stop_code

  /*225
- E axis stop code.*SL axis_e_reference_position

  /*226-229
- E axis reference position.*SL axis_e_motor_position

  /*230-233
- E axis motor position.*SL axis_e_position_error

  /*234-237

- E axis position error.∗SL axis_e_aux_position
  /∗238-241
- E axis auxiliary position.∗SL axis_e_velocity
  /∗242-245
- E axis velocity.∗SL axis_e_torque
  /∗256-249
- E axis torque.∗UW axis_e_analog_in
  /∗250-251
- E axis analog input.∗UB axis_e_reserved_0
  /∗252
- Reserved.∗UB axis_e_reserved_1
  /∗253
- Reserved.∗SL axis_e_variable
  /∗254-257
- E User defined variable.∗[] UW axis_f_status
  /∗258-259
- F axis status.∗UB axis_f_switches
  /∗260
- F axis switches.∗UB axis_f_stop_code
  /∗261
- F axis stop code.∗SL axis_f_reference_position
  /∗262-265
- F axis reference position.∗SL axis_f_motor_position
  /∗266-269
- F axis motor position.∗SL axis_f_position_error
  /∗270-273
- F axis position error.∗SL axis_f_aux_position
  /∗274-277
- F axis auxiliary position.∗SL axis_f_velocity
  /∗278-281
- F axis velocity.∗SL axis_f_torque
  /∗282-285
- F axis torque.∗UW axis_f_analog_in
  /∗286-287
- F axis analog input.∗UB axis_f_reserved_0
  /∗288
- Reserved.∗UB axis_f_reserved_1
  /∗289
- Reserved.∗SL axis_f_variable
  /∗290-293
- F User defined variable.∗[] UW axis_g_status
  /∗294-295
- G axis status.∗UB axis_g_switches
  /∗296
- G axis switches.∗UB axis_g_stop_code
  /∗297
- G axis stop code.∗SL axis_g_reference_position
  /∗298-301
- G axis reference position.∗SL axis_g_motor_position
  /∗302-305
- G axis motor position.∗SL axis_g_position_error

/*306-309
- G axis position error.*SL axis_g_aux_position
  /*310-313
- G axis auxiliary position.*SL axis_g_velocity
  /*314-317
- G axis velocity.*SL axis_g_torque
  /*318-321
- G axis torque.*UW axis_g_analog_in
  /*322-323
- G axis analog input.*UB axis_g_reserved_0
  /*324
- Reserved.*UB axis_g_reserved_1
  /*325
- Reserved.*SL axis_g_variable
  /*326-329
- G User defined variable.*[] UW axis_h_status
  /*330-331
- H axis status.*UB axis_h_switches
  /*332
- H axis switches.*UB axis_h_stop_code
  /*333
- H axis stop code.*SL axis_h_reference_position
  /*334-337
- H axis reference position.*SL axis_h_motor_position
  /*338-341
- H axis motor position.*SL axis_h_position_error
  /*342-345
- H axis position error.*SL axis_h_aux_position
  /*346-349
- H axis auxiliary position.*SL axis_h_velocity
  /*350-353
- H axis velocity.*SL axis_h_torque
  /*354-357
- H axis torque.*UW axis_h_analog_in
  /*358-359
- H axis analog input.*UB axis_h_reserved_0
  /*360
- Reserved.*UB axis_h_reserved_1
  /*361
- Reserved.*SL axis_h_variable
  /*362-365

9.20.1 Detailed Description

Definition at line 799 of file gclib.vb.

9.20.2 Member Function Documentation

9.20.2.1 byte_array()

```
H User defined variable.*[] byte_array ( )
```

Implements GDataRecord.byte_array
Implements Gclib.GDataRecord.

9.20.3 Field Documentation

9.20.3.1 axis_a_analog_in

```
A axis torque.* UW axis_a_analog_in
```
/*106-107
Definition at line 861 of file gclib.vb.

9.20.3.2 axis_a_aux_position

```
A axis position error.* SL axis_a_aux_position
```
/*94-97
Definition at line 858 of file gclib.vb.

9.20.3.3 axis_a_motor_position

```
A axis reference position.* SL axis_a_motor_position
```
/*86-89
Definition at line 856 of file gclib.vb.

9.20.3.4 axis_a_position_error

```
A axis motor position.* SL axis_a_position_error
```
/*90-93
Definition at line 857 of file gclib.vb.

9.20.3.5 axis_a_reference_position

```
A axis stop code.* SL axis_a_reference_position
```
/*82-85
Definition at line 855 of file gclib.vb.

9.20.3.6 axis_a_reserved_0

```
A axis analog input.* UB axis_a_reserved_0
```
/*108
Definition at line 862 of file gclib.vb.

9.20.3.7 axis_a_reserved_1

```
Reserved.* UB axis_a_reserved_1
```
/*109
Definition at line 863 of file gclib.vb.

9.20.3.8 axis_a_status

```
Buffer space T Plane.* UW axis_a_status
```
/*78-79
Definition at line 852 of file gclib.vb.

9.20.3.9 axis_a_stop_code

```
A axis switches.* UB axis_a_stop_code
```
/*81
Definition at line 854 of file gclib.vb.

9.20.3.10 axis_a_switches

```
A axis status.* UB axis_a_switches
```
/*80
Definition at line 853 of file gclib.vb.

9.20.3.11 axis_a_torque

A axis velocity.* SL axis_a_torque
/*102-105
Definition at line 860 of file gclib.vb.

9.20.3.12 axis_a_variable

Reserved.* SL axis_a_variable
/*110-113
Definition at line 864 of file gclib.vb.

9.20.3.13 axis_a_velocity

A axis auxiliary position.* SL axis_a_velocity
/*98-101
Definition at line 859 of file gclib.vb.

9.20.3.14 axis_b_analog_in

B axis torque.* UW axis_b_analog_in
/*142-143
Definition at line 874 of file gclib.vb.

9.20.3.15 axis_b_aux_position

B axis position error.* SL axis_b_aux_position
/*130-133
Definition at line 871 of file gclib.vb.

9.20.3.16 axis_b_motor_position

B axis reference position.* SL axis_b_motor_position
/*122-125
Definition at line 869 of file gclib.vb.

9.20.3.17 axis_b_position_error

B axis motor position.* SL axis_b_position_error
/*126-129
Definition at line 870 of file gclib.vb.

9.20.3.18 axis_b_reference_position

B axis stop code.* SL axis_b_reference_position
/*118-121
Definition at line 868 of file gclib.vb.

9.20.3.19 axis_b_reserved_0

B axis analog input.* UB axis_b_reserved_0
/*144
Definition at line 875 of file gclib.vb.

9.20.3.20 axis_b_reserved_1

Reserved.* UB axis_b_reserved_1
/*145
Definition at line 876 of file gclib.vb.

9.20.3.21 axis_b_status

```
A User defined variable.* [] UW axis_b_status
```
/*114-115
Definition at line 865 of file gclib.vb.

9.20.3.22 axis_b_stop_code

```
B axis switches.* UB axis_b_stop_code
```
/*117
Definition at line 867 of file gclib.vb.

9.20.3.23 axis_b_switches

```
B axis status.* UB axis_b_switches
```
/*116
Definition at line 866 of file gclib.vb.

9.20.3.24 axis_b_torque

```
B axis velocity.* SL axis_b_torque
```
/*138-141
Definition at line 873 of file gclib.vb.

9.20.3.25 axis_b_variable

```
Reserved.* SL axis_b_variable
```
/*146-149
Definition at line 877 of file gclib.vb.

9.20.3.26 axis_b_velocity

```
B axis auxiliary position.* SL axis_b_velocity
```
/*134-137
Definition at line 872 of file gclib.vb.

9.20.3.27 axis_c_analog_in

```
C axis torque.* UW axis_c_analog_in
```
/*178-179
Definition at line 887 of file gclib.vb.

9.20.3.28 axis_c_aux_position

```
C axis position error.* SL axis_c_aux_position
```
/*166-169
Definition at line 884 of file gclib.vb.

9.20.3.29 axis_c_motor_position

```
C axis reference position.* SL axis_c_motor_position
```
/*158-161
Definition at line 882 of file gclib.vb.

9.20.3.30 axis_c_position_error

```
C axis motor position.* SL axis_c_position_error
```
/*162-165
Definition at line 883 of file gclib.vb.

9.20.3.31 axis_c_reference_position

```
C axis stop code.* SL axis_c_reference_position
```
/*154-157
Definition at line 881 of file gclib.vb.

9.20.3.32 axis_c_reserved_0

```
C axis analog input.* UB axis_c_reserved_0
```
/*180
Definition at line 888 of file gclib.vb.

9.20.3.33 axis_c_reserved_1

```
Reserved.* UB axis_c_reserved_1
```
/*181
Definition at line 889 of file gclib.vb.

9.20.3.34 axis_c_status

```
B User defined variable.* [] UW axis_c_status
```
/*150-151
Definition at line 878 of file gclib.vb.

9.20.3.35 axis_c_stop_code

```
C axis switches.* UB axis_c_stop_code
```
/*153
Definition at line 880 of file gclib.vb.

9.20.3.36 axis_c_switches

```
C axis status.* UB axis_c_switches
```
/*152
Definition at line 879 of file gclib.vb.

9.20.3.37 axis_c_torque

```
C axis velocity.* SL axis_c_torque
```
/*174-177
Definition at line 886 of file gclib.vb.

9.20.3.38 axis_c_variable

```
Reserved.* SL axis_c_variable
```
/*182-185
Definition at line 890 of file gclib.vb.

9.20.3.39 axis_c_velocity

```
C axis auxiliary position.* SL axis_c_velocity
```
/*170-173
Definition at line 885 of file gclib.vb.

9.20.3.40 axis_d_analog_in

```
D axis torque.* UW axis_d_analog_in
```
/*214-215
Definition at line 900 of file gclib.vb.

9.20.3.41 axis_d_aux_position

```
D axis position error.* SL axis_d_aux_position
```
/*202-205
Definition at line 897 of file gclib.vb.

9.20.3.42 axis_d_motor_position

```
D axis reference position.* SL axis_d_motor_position
```
/*194-197
Definition at line 895 of file gclib.vb.

9.20.3.43 axis_d_position_error

```
D axis motor position.* SL axis_d_position_error
```
/*198-201
Definition at line 896 of file gclib.vb.

9.20.3.44 axis_d_reference_position

```
D axis stop code.* SL axis_d_reference_position
```
/*190-193
Definition at line 894 of file gclib.vb.

9.20.3.45 axis_d_reserved_0

```
D axis analog input.* UB axis_d_reserved_0
```
/*216
Definition at line 901 of file gclib.vb.

9.20.3.46 axis_d_reserved_1

```
Reserved.* UB axis_d_reserved_1
```
/*217
Definition at line 902 of file gclib.vb.

9.20.3.47 axis_d_status

```
C User defined variable.* [] UW axis_d_status
```
/*186-187
Definition at line 891 of file gclib.vb.

9.20.3.48 axis_d_stop_code

```
D axis switches.* UB axis_d_stop_code
```
/*189
Definition at line 893 of file gclib.vb.

9.20.3.49 axis_d_switches

```
D axis status.* UB axis_d_switches
```
/*188
Definition at line 892 of file gclib.vb.

9.20.3.50 axis_d_torque

```
D axis velocity.* SL axis_d_torque
```
/*210-213
Definition at line 899 of file gclib.vb.

9.20.3.51 axis_d_variable

```
Reserved.* SL axis_d_variable
```
/*218-221
Definition at line 903 of file gclib.vb.

9.20.3.52 axis_d_velocity

```
D axis auxiliary position.* SL axis_d_velocity
```
/*206-209
Definition at line 898 of file gclib.vb.

9.20.3.53 axis_e_analog_in

```
E axis torque.* UW axis_e_analog_in
```
/*250-251
Definition at line 913 of file gclib.vb.

9.20.3.54 axis_e_aux_position

```
E axis position error.* SL axis_e_aux_position
```
/*238-241
Definition at line 910 of file gclib.vb.

9.20.3.55 axis_e_motor_position

```
E axis reference position.* SL axis_e_motor_position
```
/*230-233
Definition at line 908 of file gclib.vb.

9.20.3.56 axis_e_position_error

```
E axis motor position.* SL axis_e_position_error
```
/*234-237
Definition at line 909 of file gclib.vb.

9.20.3.57 axis_e_reference_position

```
E axis stop code.* SL axis_e_reference_position
```
/*226-229
Definition at line 907 of file gclib.vb.

9.20.3.58 axis_e_reserved_0

```
E axis analog input.* UB axis_e_reserved_0
```
/*252
Definition at line 914 of file gclib.vb.

9.20.3.59 axis_e_reserved_1

```
Reserved.* UB axis_e_reserved_1
```
/*253
Definition at line 915 of file gclib.vb.

9.20.3.60 axis_e_status

```
D User defined variable.* [] UW axis_e_status
```
/*222-223
Definition at line 904 of file gclib.vb.

9.20.3.61 axis_e_stop_code

```
E axis switches.* UB axis_e_stop_code
```
/*225
Definition at line 906 of file gclib.vb.

9.20.3.62 axis_e_switches

```
E axis status.* UB axis_e_switches
```
/*224
Definition at line 905 of file gclib.vb.

9.20.3.63 axis_e_torque

```
E axis velocity.* SL axis_e_torque
```
/*256-249
Definition at line 912 of file gclib.vb.

9.20.3.64 axis_e_variable

```
Reserved.* SL axis_e_variable
```
/*254-257
Definition at line 916 of file gclib.vb.

9.20.3.65 axis_e_velocity

```
E axis auxiliary position.* SL axis_e_velocity
```
/*242-245
Definition at line 911 of file gclib.vb.

9.20.3.66 axis_f_analog_in

```
F axis torque.* UW axis_f_analog_in
```
/*286-287
Definition at line 926 of file gclib.vb.

9.20.3.67 axis_f_aux_position

```
F axis position error.* SL axis_f_aux_position
```
/*274-277
Definition at line 923 of file gclib.vb.

9.20.3.68 axis_f_motor_position

```
F axis reference position.* SL axis_f_motor_position
```
/*266-269
Definition at line 921 of file gclib.vb.

9.20.3.69 axis_f_position_error

```
F axis motor position.* SL axis_f_position_error
```
/*270-273
Definition at line 922 of file gclib.vb.

9.20.3.70 axis_f_reference_position

```
F axis stop code.* SL axis_f_reference_position
```
/*262-265
Definition at line 920 of file gclib.vb.

9.20.3.71 axis_f_reserved_0

```
F axis analog input.* UB axis_f_reserved_0
```
/*288
Definition at line 927 of file gclib.vb.

9.20.3.72 axis_f_reserved_1

```
Reserved.* UB axis_f_reserved_1
```
/*289
Definition at line 928 of file gclib.vb.

9.20.3.73 axis_f_status

```
E User defined variable.* [] UW axis_f_status
```
/*258-259
Definition at line 917 of file gclib.vb.

9.20.3.74 axis_f_stop_code

```
F axis switches.* UB axis_f_stop_code
```
/*261
Definition at line 919 of file gclib.vb.

9.20.3.75 axis_f_switches

```
F axis status.* UB axis_f_switches
```
/*260
Definition at line 918 of file gclib.vb.

9.20.3.76 axis_f_torque

```
F axis velocity.* SL axis_f_torque
```
/*282-285
Definition at line 925 of file gclib.vb.

9.20.3.77 axis_f_variable

```
Reserved.* SL axis_f_variable
```
/*290-293
Definition at line 929 of file gclib.vb.

9.20.3.78 axis_f_velocity

```
F axis auxiliary position.* SL axis_f_velocity
```
/*278-281
Definition at line 924 of file gclib.vb.

9.20.3.79 axis_g_analog_in

```
G axis torque.* UW axis_g_analog_in
```
/*322-323
Definition at line 939 of file gclib.vb.

9.20.3.80 axis_g_aux_position

```
G axis position error.* SL axis_g_aux_position
```
/*310-313
Definition at line 936 of file gclib.vb.

9.20.3.81 axis_g_motor_position

```
G axis reference position.* SL axis_g_motor_position
```
/*302-305
Definition at line 934 of file gclib.vb.

9.20.3.82 axis_g_position_error

```
G axis motor position.* SL axis_g_position_error
```
/*306-309
Definition at line 935 of file gclib.vb.

9.20.3.83 axis_g_reference_position

```
G axis stop code.* SL axis_g_reference_position
```
/*298-301
Definition at line 933 of file gclib.vb.

9.20.3.84 axis_g_reserved_0

```
G axis analog input.* UB axis_g_reserved_0
```
/*324
Definition at line 940 of file gclib.vb.

9.20.3.85 axis_g_reserved_1

```
Reserved.* UB axis_g_reserved_1
```
/*325
Definition at line 941 of file gclib.vb.

9.20.3.86 axis_g_status

```
F User defined variable.* [] UW axis_g_status
```
/*294-295
Definition at line 930 of file gclib.vb.

9.20.3.87 axis_g_stop_code

```
G axis switches.* UB axis_g_stop_code
```
/*297
Definition at line 932 of file gclib.vb.

9.20.3.88 axis_g_switches

```
G axis status.* UB axis_g_switches
```
/*296
Definition at line 931 of file gclib.vb.

9.20.3.89 axis_g_torque

```
G axis velocity.* SL axis_g_torque
```
/*318-321
Definition at line 938 of file gclib.vb.

9.20.3.90 axis_g_variable

```
Reserved.* SL axis_g_variable
```
/*326-329
Definition at line 942 of file gclib.vb.

9.20.3.91 axis_g_velocity

```
G axis auxiliary position.* SL axis_g_velocity
```
/*314-317
Definition at line 937 of file gclib.vb.

9.20.3.92 axis_h_analog_in

```
H axis torque.* UW axis_h_analog_in
```
/*358-359
Definition at line 952 of file gclib.vb.

9.20.3.93 axis_h_aux_position

```
H axis position error.* SL axis_h_aux_position
```
/*346-349
Definition at line 949 of file gclib.vb.

9.20.3.94 axis_h_motor_position

```
H axis reference position.* SL axis_h_motor_position
```
/*338-341
Definition at line 947 of file gclib.vb.

9.20.3.95 axis_h_position_error

```
H axis motor position.* SL axis_h_position_error
```
/*342-345
Definition at line 948 of file gclib.vb.

9.20.3.96 axis_h_reference_position

```
H axis stop code.* SL axis_h_reference_position
```
/*334-337
Definition at line 946 of file gclib.vb.

9.20.3.97 axis_h_reserved_0

```
H axis analog input.* UB axis_h_reserved_0
```
/*360
Definition at line 953 of file gclib.vb.

9.20.3.98 axis_h_reserved_1

```
Reserved.* UB axis_h_reserved_1
```
/*361
Definition at line 954 of file gclib.vb.

9.20.3.99 axis_h_status

```
G User defined variable.* [] UW axis_h_status
```
/*330-331
Definition at line 943 of file gclib.vb.

9.20.3.100 axis_h_stop_code

```
H axis switches.* UB axis_h_stop_code
```
/*333
Definition at line 945 of file gclib.vb.

9.20.3.101 axis_h_switches

```
H axis status.* UB axis_h_switches
```
/*332
Definition at line 944 of file gclib.vb.

9.20.3.102 axis_h_torque

```
H axis velocity.* SL axis_h_torque
```
/*354-357
Definition at line 951 of file gclib.vb.

9.20.3.103 axis_h_variable

```
Reserved.* SL axis_h_variable
```
/*362-365
Definition at line 955 of file gclib.vb.

9.20.3.104 axis_h_velocity

```
H axis auxiliary position.* SL axis_h_velocity
```
/*350-353
Definition at line 950 of file gclib.vb.

9.20.3.105 contour_buffer_available

```
Segment Count for Contour Mode.* UW contour_buffer_available
```
/*56-57
Definition at line 843 of file gclib.vb.

9.20.3.106 contour_segment_count

```
Reserved.* UL contour_segment_count
```
/*52-55
Definition at line 842 of file gclib.vb.

9.20.3.107 error_code

```
Reserved.* UB error_code
```
/*46
Definition at line 839 of file gclib.vb.

9.20.3.108 input_bank_0

```
sample number.* UB input_bank_0
```
/*02
Definition at line 803 of file gclib.vb.

9.20.3.109 remaining

```
Buffer space remaining
```
Definition at line 843 of file gclib.vb.

9.20.3.110 reserved_10

```
Reserved.* SW reserved_10
```
/*32-33
Definition at line 828 of file gclib.vb.

9.20.3.111 reserved_12

```
Reserved.* SW reserved_12
```
/*34-35
Definition at line 829 of file gclib.vb.

9.20.3.112 reserved_14

```
Reserved.* SW reserved_14
```
/*36-37
Definition at line 830 of file gclib.vb.

9.20.3.113 reserved_16

```
Reserved.* UB reserved_16
```
/*38
Definition at line 831 of file gclib.vb.

9.20.3.114 reserved_17

```
Reserved.* UB reserved_17
```
/*39
Definition at line 832 of file gclib.vb.

9.20.3.115 reserved_18

```
Reserved.* UB reserved_18
```
/*40
Definition at line 833 of file gclib.vb.

9.20.3.116 reserved_19

```
Reserved.* UB reserved_19
```
/*41
Definition at line 834 of file gclib.vb.

9.20.3.117 reserved_2

```
Reserved.* SW reserved_2
```
/*24-25
Definition at line 824 of file gclib.vb.

9.20.3.118 reserved_20

```
Reserved.* UB reserved_20
```
/*42
Definition at line 835 of file gclib.vb.

9.20.3.119 reserved_21

```
Reserved.* UB reserved_21
```
/*43
Definition at line 836 of file gclib.vb.

9.20.3.120 reserved_22

```
Reserved.* UB reserved_22
```
/*44
Definition at line 837 of file gclib.vb.

9.20.3.121 reserved_23

```
Reserved.* UB reserved_23
```
/*45
Definition at line 838 of file gclib.vb.

9.20.3.122 reserved_24

```
thread status.* UL reserved_24
```
/*48-51
Definition at line 841 of file gclib.vb.

9.20.3.123 reserved_4

```
Reserved.* SW reserved_4
```
/*26-27
Definition at line 825 of file gclib.vb.

9.20.3.124 reserved_6

```
Reserved.* SW reserved_6
```
/*28-29
Definition at line 826 of file gclib.vb.

9.20.3.125 reserved_8

```
Reserved.* SW reserved_8
```
/*30-31
Definition at line 827 of file gclib.vb.

9.20.3.126 s_distance

```
coordinated move status for S plane.* SL s_distance
```
/*62-65
Definition at line 846 of file gclib.vb.

9.20.3.127 s_plane_buffer_available

```
distance traveled in coordinated move for S plane.* UW s_plane_buffer_available
```
/*66-67
Definition at line 847 of file gclib.vb.

9.20.3.128 s_plane_move_status

```
segment count of coordinated move for S plane.* UW s_plane_move_status
```
/*60-61
Definition at line 845 of file gclib.vb.

9.20.3.129 s_plane_segment_count

```
Buffer space Contour Mode.* UW s_plane_segment_count
```
/*58-59
Definition at line 844 of file gclib.vb.

9.20.3.130 sample_number

```
UW sample_number
```
/*00-01
Definition at line 802 of file gclib.vb.

9.20.3.131 t_distance

```
Coordinated move status for T plane.* SL t_distance
```
/*72-75
Definition at line 850 of file gclib.vb.

9.20.3.132 t_plane_buffer_available

```
distance traveled in coordinated move for T plane.* UW t_plane_buffer_available
```
/*76-77
Definition at line 851 of file gclib.vb.

9.20.3.133 t_plane_move_status

```
segment count of coordinated move for T plane.* UW t_plane_move_status
```
/*70-71
Definition at line 849 of file gclib.vb.

9.20.3.134 t_plane_segment_count

```
Buffer space S Plane.* UW t_plane_segment_count
```
/*68-69
Definition at line 848 of file gclib.vb.

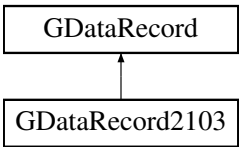
9.20.3.135 thread_status

```
error code.* UB thread_status
```
/*47
Definition at line 840 of file gclib.vb.
The documentation for this struct was generated from the following file:

- gclib.vb

9.21 GDataRecord2103 Struct Reference

Data record struct for DMC-2103 controllers.
Inheritance diagram for GDataRecord2103:



Public Member Functions

- byte[] byte_array ()

  Returns the data record as a byte array and allows for access to individual bytes.

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.

- UB header_3

  4th Byte of Header.
- UW sample_number

  sample number.
- UB input_bank_0

  general input bank 0 (inputs 1-8).
- UB input_bank_1

  general input bank 1 (inputs 9-16).
- UB input_bank_2

  general input bank 2 (inputs 17-24).
- UB input_bank_3

  general input bank 3 (inputs 25-32).
- UB input_bank_4

  general input bank 4 (inputs 33-40).
- UB input_bank_5

  general input bank 5 (inputs 41-48).
- UB input_bank_6

  general input bank 6 (inputs 49-56).
- UB input_bank_7

  general input bank 7 (inputs 57-64).
- UB input_bank_8

  general input bank 8 (inputs 65-72).
- UB input_bank_9

  general input bank 9 (inputs 73-80).
- UB output_bank_0

  general output bank 0 (outputs 1-8).
- UB output_bank_1

  general output bank 1 (outputs 9-16).
- UB output_bank_2

  general output bank 2 (outputs 17-24).
- UB output_bank_3

  general output bank 3 (outputs 25-32).
- UB output_bank_4

  general output bank 4 (outputs 33-40).
- UB output_bank_5

  general output bank 5 (outputs 41-48).
- UB output_bank_6

  general output bank 6 (outputs 49-56).
- UB output_bank_7

  general output bank 7 (outputs 57-64).
- UB output_bank_8

  general output bank 8 (outputs 65-72).
- UB output_bank_9

  general output bank 9 (outputs 73-80).
- UB error_code

  error code.
- UB general_status

  general status
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

coordinated move status for S plane.

- SL s_distance

  distance traveled in coordinated move for S plane.
- UW t_plane_segment_count

  segment count of coordinated move for T plane.
- UW t_plane_move_status

  Coordinated move status for T plane.
- SL t_distance

  distance traveled in coordinated move for T plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

  A axis stop code.
- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

  A axis auxiliary position.
- SL axis_a_velocity

  A axis velocity.
- SW axis_a_torque

  A axis torque.
- UW axis_a_analog_in

  A axis analog input.
- UW axis_b_status

  B axis status.
- UB axis_b_switches

  B axis switches.
- UB axis_b_stop_code

  B axis stop code.
- SL axis_b_reference_position

  B axis reference position.
- SL axis_b_motor_position

  B axis motor position.
- SL axis_b_position_error

  B axis position error.
- SL axis_b_aux_position

  B axis auxiliary position.
- SL axis_b_velocity

  B axis velocity.
- SW axis_b_torque

  B axis torque.
- UW axis_b_analog_in

  B axis analog input.
- UW axis_c_status

  C axis status.

- UB axis_c_switches
  C axis switches.
- UB axis_c_stop_code
  C axis stop code.
- SL axis_c_reference_position
  C axis reference position.
- SL axis_c_motor_position
  C axis motor position.
- SL axis_c_position_error
  C axis position error.
- SL axis_c_aux_position
  C axis auxiliary position.
- SL axis_c_velocity
  C axis velocity.
- SW axis_c_torque
  C axis torque.
- UW axis_c_analog_in
  C axis analog input.
- UW axis_d_status
  D axis status.
- UB axis_d_switches
  D axis switches.
- UB axis_d_stop_code
  D axis stop code.
- SL axis_d_reference_position
  D axis reference position.
- SL axis_d_motor_position
  D axis motor position.
- SL axis_d_position_error
  D axis position error.
- SL axis_d_aux_position
  D axis auxiliary position.
- SL axis_d_velocity
  D axis velocity.
- SW axis_d_torque
  D axis torque.
- UW axis_d_analog_in
  D axis analog input.
- UW axis_e_status
  E axis status.
- UB axis_e_switches
  E axis switches.
- UB axis_e_stop_code
  E axis stop code.
- SL axis_e_reference_position
  E axis reference position.
- SL axis_e_motor_position
  E axis motor position.
- SL axis_e_position_error
  E axis position error.
- SL axis_e_aux_position

E axis auxiliary position.

- SL axis_e_velocity

  E axis velocity.
- SW axis_e_torque

  E axis torque.
- UW axis_e_analog_in

  E axis analog input.
- UW axis_f_status

  F axis status.
- UB axis_f_switches

  F axis switches.
- UB axis_f_stop_code

  F axis stop code.
- SL axis_f_reference_position

  F axis reference position.
- SL axis_f_motor_position

  F axis motor position.
- SL axis_f_position_error

  F axis position error.
- SL axis_f_aux_position

  F axis auxiliary position.
- SL axis_f_velocity

  F axis velocity.
- SW axis_f_torque

  F axis torque.
- UW axis_f_analog_in

  F axis analog input.
- UW axis_g_status

  G axis status.
- UB axis_g_switches

  G axis switches.
- UB axis_g_stop_code

  G axis stop code.
- SL axis_g_reference_position

  G axis reference position.
- SL axis_g_motor_position

  G axis motor position.
- SL axis_g_position_error

  G axis position error.
- SL axis_g_aux_position

  G axis auxiliary position.
- SL axis_g_velocity

  G axis velocity.
- SW axis_g_torque

  G axis torque.
- UW axis_g_analog_in

  G axis analog input.
- UW axis_h_status

  H axis status.
- UB axis_h_switches

  H axis switches.

- UB axis_h_stop_code

  H axis stop code.
- SL axis_h_reference_position

  H axis reference position.
- SL axis_h_motor_position

  H axis motor position.
- SL axis_h_position_error

  H axis position error.
- SL axis_h_aux_position

  H axis auxiliary position.
- SL axis_h_velocity

  H axis velocity.
- SW axis_h_torque

  H axis torque.
- UW axis_h_analog_in

  H axis analog input.

9.21.1 Detailed Description

Data record struct for DMC-2103 controllers.
Definition at line 1473 of file gclib.cs.

9.21.2 Member Function Documentation

9.21.2.1 byte_array()

```
byte[] byte_array ( )  [inline]
```

Returns the data record as a byte array and allows for access to individual bytes.
Implements GDataRecord.
Definition at line 1475 of file gclib.cs.

9.21.3 Field Documentation

9.21.3.1 axis_a_analog_in

```
UW axis_a_analog_in
```

A axis analog input.
Definition at line 1528 of file gclib.cs.

9.21.3.2 axis_a_aux_position

```
SL axis_a_aux_position
```

A axis auxiliary position.
Definition at line 1525 of file gclib.cs.

9.21.3.3 axis_a_motor_position

```
SL axis_a_motor_position
```

A axis motor position.
Definition at line 1523 of file gclib.cs.

9.21.3.4 axis_a_position_error

```
SL axis_a_position_error
```

A axis position error.
Definition at line 1524 of file gclib.cs.

9.21.3.5 axis_a_reference_position

`SL axis_a_reference_position`
A axis reference position.
Definition at line 1522 of file gclib.cs.

9.21.3.6 axis_a_status

`UW axis_a_status`
A axis status.
Definition at line 1519 of file gclib.cs.

9.21.3.7 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 1521 of file gclib.cs.

9.21.3.8 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 1520 of file gclib.cs.

9.21.3.9 axis_a_torque

`SW axis_a_torque`
A axis torque.
Definition at line 1527 of file gclib.cs.

9.21.3.10 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 1526 of file gclib.cs.

9.21.3.11 axis_b_analog_in

`UW axis_b_analog_in`
B axis analog input.
Definition at line 1539 of file gclib.cs.

9.21.3.12 axis_b_aux_position

`SL axis_b_aux_position`
B axis auxiliary position.
Definition at line 1536 of file gclib.cs.

9.21.3.13 axis_b_motor_position

`SL axis_b_motor_position`
B axis motor position.
Definition at line 1534 of file gclib.cs.

9.21.3.14 axis_b_position_error

`SL axis_b_position_error`
B axis position error.
Definition at line 1535 of file gclib.cs.

9.21.3.15 axis_b_reference_position

`SL axis_b_reference_position`
B axis reference position.
Definition at line 1533 of file gclib.cs.

9.21.3.16 axis_b_status

`UW axis_b_status`
B axis status.
Definition at line 1530 of file gclib.cs.

9.21.3.17 axis_b_stop_code

`UB axis_b_stop_code`
B axis stop code.
Definition at line 1532 of file gclib.cs.

9.21.3.18 axis_b_switches

`UB axis_b_switches`
B axis switches.
Definition at line 1531 of file gclib.cs.

9.21.3.19 axis_b_torque

`SW axis_b_torque`
B axis torque.
Definition at line 1538 of file gclib.cs.

9.21.3.20 axis_b_velocity

`SL axis_b_velocity`
B axis velocity.
Definition at line 1537 of file gclib.cs.

9.21.3.21 axis_c_analog_in

`UW axis_c_analog_in`
C axis analog input.
Definition at line 1550 of file gclib.cs.

9.21.3.22 axis_c_aux_position

`SL axis_c_aux_position`
C axis auxiliary position.
Definition at line 1547 of file gclib.cs.

9.21.3.23 axis_c_motor_position

`SL axis_c_motor_position`
C axis motor position.
Definition at line 1545 of file gclib.cs.

9.21.3.24 axis_c_position_error

`SL axis_c_position_error`
C axis position error.
Definition at line 1546 of file gclib.cs.

9.21.3.25 axis_c_reference_position

`SL axis_c_reference_position`
C axis reference position.
Definition at line 1544 of file gclib.cs.

9.21.3.26 axis_c_status

`UW axis_c_status`
C axis status.
Definition at line 1541 of file gclib.cs.

9.21.3.27 axis_c_stop_code

`UB axis_c_stop_code`
C axis stop code.
Definition at line 1543 of file gclib.cs.

9.21.3.28 axis_c_switches

`UB axis_c_switches`
C axis switches.
Definition at line 1542 of file gclib.cs.

9.21.3.29 axis_c_torque

`SW axis_c_torque`
C axis torque.
Definition at line 1549 of file gclib.cs.

9.21.3.30 axis_c_velocity

`SL axis_c_velocity`
C axis velocity.
Definition at line 1548 of file gclib.cs.

9.21.3.31 axis_d_analog_in

`UW axis_d_analog_in`
D axis analog input.
Definition at line 1561 of file gclib.cs.

9.21.3.32 axis_d_aux_position

`SL axis_d_aux_position`
D axis auxiliary position.
Definition at line 1558 of file gclib.cs.

9.21.3.33 axis_d_motor_position

`SL axis_d_motor_position`
D axis motor position.
Definition at line 1556 of file gclib.cs.

9.21.3.34 axis_d_position_error

`SL axis_d_position_error`
D axis position error.
Definition at line 1557 of file gclib.cs.

9.21.3.35 axis_d_reference_position

`SL axis_d_reference_position`
D axis reference position.
Definition at line 1555 of file gclib.cs.

9.21.3.36 axis_d_status

`UW axis_d_status`
D axis status.
Definition at line 1552 of file gclib.cs.

9.21.3.37 axis_d_stop_code

`UB axis_d_stop_code`
D axis stop code.
Definition at line 1554 of file gclib.cs.

9.21.3.38 axis_d_switches

`UB axis_d_switches`
D axis switches.
Definition at line 1553 of file gclib.cs.

9.21.3.39 axis_d_torque

`SW axis_d_torque`
D axis torque.
Definition at line 1560 of file gclib.cs.

9.21.3.40 axis_d_velocity

`SL axis_d_velocity`
D axis velocity.
Definition at line 1559 of file gclib.cs.

9.21.3.41 axis_e_analog_in

`UW axis_e_analog_in`
E axis analog input.
Definition at line 1572 of file gclib.cs.

9.21.3.42 axis_e_aux_position

`SL axis_e_aux_position`
E axis auxiliary position.
Definition at line 1569 of file gclib.cs.

9.21.3.43 axis_e_motor_position

`SL axis_e_motor_position`
E axis motor position.
Definition at line 1567 of file gclib.cs.

9.21.3.44 axis_e_position_error

`SL axis_e_position_error`
E axis position error.
Definition at line 1568 of file gclib.cs.

9.21.3.45 axis_e_reference_position

`SL axis_e_reference_position`
E axis reference position.
Definition at line 1566 of file gclib.cs.

9.21.3.46 axis_e_status

`UW axis_e_status`
E axis status.
Definition at line 1563 of file gclib.cs.

9.21.3.47 axis_e_stop_code

`UB axis_e_stop_code`
E axis stop code.
Definition at line 1565 of file gclib.cs.

9.21.3.48 axis_e_switches

`UB axis_e_switches`
E axis switches.
Definition at line 1564 of file gclib.cs.

9.21.3.49 axis_e_torque

`SW axis_e_torque`
E axis torque.
Definition at line 1571 of file gclib.cs.

9.21.3.50 axis_e_velocity

`SL axis_e_velocity`
E axis velocity.
Definition at line 1570 of file gclib.cs.

9.21.3.51 axis_f_analog_in

`UW axis_f_analog_in`
F axis analog input.
Definition at line 1583 of file gclib.cs.

9.21.3.52 axis_f_aux_position

`SL axis_f_aux_position`
F axis auxiliary position.
Definition at line 1580 of file gclib.cs.

9.21.3.53 axis_f_motor_position

`SL axis_f_motor_position`
F axis motor position.
Definition at line 1578 of file gclib.cs.

9.21.3.54 axis_f_position_error

`SL axis_f_position_error`
F axis position error.
Definition at line 1579 of file gclib.cs.

9.21.3.55 axis_f_reference_position

`SL axis_f_reference_position`
F axis reference position.
Definition at line 1577 of file gclib.cs.

9.21.3.56 axis_f_status

`UW axis_f_status`
F axis status.
Definition at line 1574 of file gclib.cs.

9.21.3.57 axis_f_stop_code

`UB axis_f_stop_code`
F axis stop code.
Definition at line 1576 of file gclib.cs.

9.21.3.58 axis_f_switches

`UB axis_f_switches`
F axis switches.
Definition at line 1575 of file gclib.cs.

9.21.3.59 axis_f_torque

`SW axis_f_torque`
F axis torque.
Definition at line 1582 of file gclib.cs.

9.21.3.60 axis_f_velocity

`SL axis_f_velocity`
F axis velocity.
Definition at line 1581 of file gclib.cs.

9.21.3.61 axis_g_analog_in

`UW axis_g_analog_in`
G axis analog input.
Definition at line 1594 of file gclib.cs.

9.21.3.62 axis_g_aux_position

`SL axis_g_aux_position`
G axis auxiliary position.
Definition at line 1591 of file gclib.cs.

9.21.3.63 axis_g_motor_position

`SL axis_g_motor_position`
G axis motor position.
Definition at line 1589 of file gclib.cs.

9.21.3.64 axis_g_position_error

`SL axis_g_position_error`
G axis position error.
Definition at line 1590 of file gclib.cs.

9.21.3.65 axis_g_reference_position

`SL axis_g_reference_position`
G axis reference position.
Definition at line 1588 of file gclib.cs.

9.21.3.66 axis_g_status

`UW axis_g_status`
G axis status.
Definition at line 1585 of file gclib.cs.

9.21.3.67 axis_g_stop_code

`UB axis_g_stop_code`
G axis stop code.
Definition at line 1587 of file gclib.cs.

9.21.3.68 axis_g_switches

`UB axis_g_switches`
G axis switches.
Definition at line 1586 of file gclib.cs.

9.21.3.69 axis_g_torque

`SW axis_g_torque`
G axis torque.
Definition at line 1593 of file gclib.cs.

9.21.3.70 axis_g_velocity

`SL axis_g_velocity`
G axis velocity.
Definition at line 1592 of file gclib.cs.

9.21.3.71 axis_h_analog_in

`UW axis_h_analog_in`
H axis analog input.
Definition at line 1605 of file gclib.cs.

9.21.3.72 axis_h_aux_position

`SL axis_h_aux_position`
H axis auxiliary position.
Definition at line 1602 of file gclib.cs.

9.21.3.73 axis_h_motor_position

`SL axis_h_motor_position`
H axis motor position.
Definition at line 1600 of file gclib.cs.

9.21.3.74 axis_h_position_error

`SL axis_h_position_error`
H axis position error.
Definition at line 1601 of file gclib.cs.

9.21.3.75 axis_h_reference_position

`SL axis_h_reference_position`
H axis reference position.
Definition at line 1599 of file gclib.cs.

9.21.3.76 axis_h_status

`UW axis_h_status`
H axis status.
Definition at line 1596 of file gclib.cs.

9.21.3.77 axis_h_stop_code

`UB axis_h_stop_code`
H axis stop code.
Definition at line 1598 of file gclib.cs.

9.21.3.78 axis_h_switches

`UB axis_h_switches`
H axis switches.
Definition at line 1597 of file gclib.cs.

9.21.3.79 axis_h_torque

`SW axis_h_torque`
H axis torque.
Definition at line 1604 of file gclib.cs.

9.21.3.80 axis_h_velocity

`SL axis_h_velocity`
H axis velocity.
Definition at line 1603 of file gclib.cs.

9.21.3.81 error_code

`UB error_code`
error code.
Definition at line 1508 of file gclib.cs.

9.21.3.82 general_status

`UB general_status`
general status
Definition at line 1509 of file gclib.cs.

9.21.3.83 header_0

`UB header_0`
1st Byte of Header.
Definition at line 1479 of file gclib.cs.

9.21.3.84 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 1480 of file gclib.cs.

9.21.3.85 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 1481 of file gclib.cs.

9.21.3.86 header_3

`UB header_3`
4th Byte of Header.
Definition at line 1482 of file gclib.cs.

9.21.3.87 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 1486 of file gclib.cs.

9.21.3.88 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 1487 of file gclib.cs.

9.21.3.89 input_bank_2

`UB input_bank_2`
general input bank 2 (inputs 17-24).
Definition at line 1488 of file gclib.cs.

9.21.3.90 input_bank_3

`UB input_bank_3`
general input bank 3 (inputs 25-32).
Definition at line 1489 of file gclib.cs.

9.21.3.91 input_bank_4

`UB input_bank_4`
general input bank 4 (inputs 33-40).
Definition at line 1490 of file gclib.cs.

9.21.3.92 input_bank_5

`UB input_bank_5`
general input bank 5 (inputs 41-48).
Definition at line 1491 of file gclib.cs.

9.21.3.93 input_bank_6

`UB input_bank_6`
general input bank 6 (inputs 49-56).
Definition at line 1492 of file gclib.cs.

9.21.3.94 input_bank_7

`UB input_bank_7`
general input bank 7 (inputs 57-64).
Definition at line 1493 of file gclib.cs.

9.21.3.95 input_bank_8

`UB input_bank_8`
general input bank 8 (inputs 65-72).
Definition at line 1494 of file gclib.cs.

9.21.3.96 input_bank_9

`UB input_bank_9`
general input bank 9 (inputs 73-80).
Definition at line 1495 of file gclib.cs.

9.21.3.97 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 1497 of file gclib.cs.

9.21.3.98 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 1498 of file gclib.cs.

9.21.3.99 output_bank_2

`UB output_bank_2`
general output bank 2 (outputs 17-24).
Definition at line 1499 of file gclib.cs.

9.21.3.100 output_bank_3

`UB output_bank_3`
general output bank 3 (outputs 25-32).
Definition at line 1500 of file gclib.cs.

9.21.3.101 output_bank_4

`UB output_bank_4`
general output bank 4 (outputs 33-40).
Definition at line 1501 of file gclib.cs.

9.21.3.102 output_bank_5

`UB output_bank_5`
general output bank 5 (outputs 41-48).
Definition at line 1502 of file gclib.cs.

9.21.3.103 output_bank_6

`UB output_bank_6`
general output bank 6 (outputs 49-56).
Definition at line 1503 of file gclib.cs.

9.21.3.104 output_bank_7

`UB output_bank_7`
general output bank 7 (outputs 57-64).
Definition at line 1504 of file gclib.cs.

9.21.3.105 output_bank_8

`UB output_bank_8`
general output bank 8 (outputs 65-72).
Definition at line 1505 of file gclib.cs.

9.21.3.106 output_bank_9

`UB output_bank_9`
general output bank 9 (outputs 73-80).
Definition at line 1506 of file gclib.cs.

9.21.3.107 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 1513 of file gclib.cs.

9.21.3.108 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 1512 of file gclib.cs.

9.21.3.109 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 1511 of file gclib.cs.

9.21.3.110 sample_number

`UW sample_number`
sample number.
Definition at line 1484 of file gclib.cs.

9.21.3.111 t_distance

`SL t_distance`
distance traveled in coordinated move for T plane.
Definition at line 1517 of file gclib.cs.

9.21.3.112 t_plane_move_status

`UW t_plane_move_status`
Coordinated move status for T plane.
Definition at line 1516 of file gclib.cs.

9.21.3.113 t_plane_segment_count

`UW t_plane_segment_count`
segment count of coordinated move for T plane.
Definition at line 1515 of file gclib.cs.
The documentation for this struct was generated from the following file:

- gclib.cs

9.22 GDataRecord2103 Struct Reference

Data record struct for DMC-2103 controllers.
`#include <gclib_record.h>`

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.
- UB header_3

  4th Byte of Header.
- UW sample_number

  sample number.
- UB input_bank_0

  general input bank 0 (inputs 1-8).
- UB input_bank_1

  general input bank 1 (inputs 9-16).
- UB input_bank_2

  general input bank 2 (inputs 17-24).
- UB input_bank_3

  general input bank 3 (inputs 25-32).
- UB input_bank_4

  general input bank 4 (inputs 33-40).
- UB input_bank_5

  general input bank 5 (inputs 41-48).
- UB input_bank_6

  general input bank 6 (inputs 49-56).
- UB input_bank_7

  general input bank 7 (inputs 57-64).
- UB input_bank_8

  general input bank 8 (inputs 65-72).
- UB input_bank_9

  general input bank 9 (inputs 73-80).
- UB output_bank_0

  general output bank 0 (outputs 1-8).
- UB output_bank_1

  general output bank 1 (outputs 9-16).
- UB output_bank_2

  general output bank 2 (outputs 17-24).
- UB output_bank_3

  general output bank 3 (outputs 25-32).
- UB output_bank_4

  general output bank 4 (outputs 33-40).
- UB output_bank_5

  general output bank 5 (outputs 41-48).
- UB output_bank_6

  general output bank 6 (outputs 49-56).
- UB output_bank_7

general output bank 7 (outputs 57-64).

- UB output_bank_8

  general output bank 8 (outputs 65-72).
- UB output_bank_9

  general output bank 9 (outputs 73-80).
- UB error_code

  error code.
- UB general_status

  general status
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

  coordinated move status for S plane.
- SL s_distance

  distance traveled in coordinated move for S plane.
- UW t_plane_segment_count

  segment count of coordinated move for T plane.
- UW t_plane_move_status

  Coordinated move status for T plane.
- SL t_distance

  distance traveled in coordinated move for T plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

  A axis stop code.
- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

  A axis auxiliary position.
- SL axis_a_velocity

  A axis velocity.
- SW axis_a_torque

  A axis torque.
- UW axis_a_analog_in

  A axis analog input.
- UW axis_b_status

  B axis status.
- UB axis_b_switches

  B axis switches.
- UB axis_b_stop_code

  B axis stop code.
- SL axis_b_reference_position

  B axis reference position.
- SL axis_b_motor_position

  B axis motor position.

- SL axis_b_position_error

  B axis position error.
- SL axis_b_aux_position

  B axis auxiliary position.
- SL axis_b_velocity

  B axis velocity.
- SW axis_b_torque

  B axis torque.
- UW axis_b_analog_in

  B axis analog input.
- UW axis_c_status

  C axis status.
- UB axis_c_switches

  C axis switches.
- UB axis_c_stop_code

  C axis stop code.
- SL axis_c_reference_position

  C axis reference position.
- SL axis_c_motor_position

  C axis motor position.
- SL axis_c_position_error

  C axis position error.
- SL axis_c_aux_position

  C axis auxiliary position.
- SL axis_c_velocity

  C axis velocity.
- SW axis_c_torque

  C axis torque.
- UW axis_c_analog_in

  C axis analog input.
- UW axis_d_status

  D axis status.
- UB axis_d_switches

  D axis switches.
- UB axis_d_stop_code

  D axis stop code.
- SL axis_d_reference_position

  D axis reference position.
- SL axis_d_motor_position

  D axis motor position.
- SL axis_d_position_error

  D axis position error.
- SL axis_d_aux_position

  D axis auxiliary position.
- SL axis_d_velocity

  D axis velocity.
- SW axis_d_torque

  D axis torque.
- UW axis_d_analog_in

  D axis analog input.
- UW axis_e_status

E axis status.

- UB axis_e_switches

  E axis switches.
- UB axis_e_stop_code

  E axis stop code.
- SL axis_e_reference_position

  E axis reference position.
- SL axis_e_motor_position

  E axis motor position.
- SL axis_e_position_error

  E axis position error.
- SL axis_e_aux_position

  E axis auxiliary position.
- SL axis_e_velocity

  E axis velocity.
- SW axis_e_torque

  E axis torque.
- UW axis_e_analog_in

  E axis analog input.
- UW axis_f_status

  F axis status.
- UB axis_f_switches

  F axis switches.
- UB axis_f_stop_code

  F axis stop code.
- SL axis_f_reference_position

  F axis reference position.
- SL axis_f_motor_position

  F axis motor position.
- SL axis_f_position_error

  F axis position error.
- SL axis_f_aux_position

  F axis auxiliary position.
- SL axis_f_velocity

  F axis velocity.
- SW axis_f_torque

  F axis torque.
- UW axis_f_analog_in

  F axis analog input.
- UW axis_g_status

  G axis status.
- UB axis_g_switches

  G axis switches.
- UB axis_g_stop_code

  G axis stop code.
- SL axis_g_reference_position

  G axis reference position.
- SL axis_g_motor_position

  G axis motor position.
- SL axis_g_position_error

  G axis position error.

- SL axis_g_aux_position

  G axis auxiliary position.
- SL axis_g_velocity

  G axis velocity.
- SW axis_g_torque

  G axis torque.
- UW axis_g_analog_in

  G axis analog input.
- UW axis_h_status

  H axis status.
- UB axis_h_switches

  H axis switches.
- UB axis_h_stop_code

  H axis stop code.
- SL axis_h_reference_position

  H axis reference position.
- SL axis_h_motor_position

  H axis motor position.
- SL axis_h_position_error

  H axis position error.
- SL axis_h_aux_position

  H axis auxiliary position.
- SL axis_h_velocity

  H axis velocity.
- SW axis_h_torque

  H axis torque.
- UW axis_h_analog_in

  H axis analog input.

9.22.1 Detailed Description

Data record struct for DMC-2103 controllers.
Definition at line 586 of file gclib_record.h.

9.22.2 Field Documentation

9.22.2.1 axis_a_analog_in

```
UW axis_a_analog_in
```

A axis analog input.
Definition at line 640 of file gclib_record.h.

9.22.2.2 axis_a_aux_position

```
SL axis_a_aux_position
```

A axis auxiliary position.
Definition at line 637 of file gclib_record.h.

9.22.2.3 axis_a_motor_position

```
SL axis_a_motor_position
```

A axis motor position.
Definition at line 635 of file gclib_record.h.

9.22.2.4 axis_a_position_error

`SL axis_a_position_error`
A axis position error.
Definition at line 636 of file gclib_record.h.

9.22.2.5 axis_a_reference_position

`SL axis_a_reference_position`
A axis reference position.
Definition at line 634 of file gclib_record.h.

9.22.2.6 axis_a_status

`UW axis_a_status`
A axis status.
Definition at line 631 of file gclib_record.h.

9.22.2.7 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 633 of file gclib_record.h.

9.22.2.8 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 632 of file gclib_record.h.

9.22.2.9 axis_a_torque

`SW axis_a_torque`
A axis torque.
Definition at line 639 of file gclib_record.h.

9.22.2.10 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 638 of file gclib_record.h.

9.22.2.11 axis_b_analog_in

`UW axis_b_analog_in`
B axis analog input.
Definition at line 651 of file gclib_record.h.

9.22.2.12 axis_b_aux_position

`SL axis_b_aux_position`
B axis auxiliary position.
Definition at line 648 of file gclib_record.h.

9.22.2.13 axis_b_motor_position

`SL axis_b_motor_position`
B axis motor position.
Definition at line 646 of file gclib_record.h.

9.22.2.14 axis_b_position_error

`SL axis_b_position_error`
B axis position error.
Definition at line 647 of file gclib_record.h.

9.22.2.15 axis_b_reference_position

`SL axis_b_reference_position`
B axis reference position.
Definition at line 645 of file gclib_record.h.

9.22.2.16 axis_b_status

`UW axis_b_status`
B axis status.
Definition at line 642 of file gclib_record.h.

9.22.2.17 axis_b_stop_code

`UB axis_b_stop_code`
B axis stop code.
Definition at line 644 of file gclib_record.h.

9.22.2.18 axis_b_switches

`UB axis_b_switches`
B axis switches.
Definition at line 643 of file gclib_record.h.

9.22.2.19 axis_b_torque

`SW axis_b_torque`
B axis torque.
Definition at line 650 of file gclib_record.h.

9.22.2.20 axis_b_velocity

`SL axis_b_velocity`
B axis velocity.
Definition at line 649 of file gclib_record.h.

9.22.2.21 axis_c_analog_in

`UW axis_c_analog_in`
C axis analog input.
Definition at line 662 of file gclib_record.h.

9.22.2.22 axis_c_aux_position

`SL axis_c_aux_position`
C axis auxiliary position.
Definition at line 659 of file gclib_record.h.

9.22.2.23 axis_c_motor_position

`SL axis_c_motor_position`
C axis motor position.
Definition at line 657 of file gclib_record.h.

9.22.2.24 axis_c_position_error

`SL axis_c_position_error`
C axis position error.
Definition at line 658 of file gclib_record.h.

9.22.2.25 axis_c_reference_position

`SL axis_c_reference_position`
C axis reference position.
Definition at line 656 of file gclib_record.h.

9.22.2.26 axis_c_status

`UW axis_c_status`
C axis status.
Definition at line 653 of file gclib_record.h.

9.22.2.27 axis_c_stop_code

`UB axis_c_stop_code`
C axis stop code.
Definition at line 655 of file gclib_record.h.

9.22.2.28 axis_c_switches

`UB axis_c_switches`
C axis switches.
Definition at line 654 of file gclib_record.h.

9.22.2.29 axis_c_torque

`SW axis_c_torque`
C axis torque.
Definition at line 661 of file gclib_record.h.

9.22.2.30 axis_c_velocity

`SL axis_c_velocity`
C axis velocity.
Definition at line 660 of file gclib_record.h.

9.22.2.31 axis_d_analog_in

`UW axis_d_analog_in`
D axis analog input.
Definition at line 673 of file gclib_record.h.

9.22.2.32 axis_d_aux_position

`SL axis_d_aux_position`
D axis auxiliary position.
Definition at line 670 of file gclib_record.h.

9.22.2.33 axis_d_motor_position

`SL axis_d_motor_position`
D axis motor position.
Definition at line 668 of file gclib_record.h.

9.22.2.34 axis_d_position_error

`SL axis_d_position_error`
D axis position error.
Definition at line 669 of file gclib_record.h.

9.22.2.35 axis_d_reference_position

`SL axis_d_reference_position`
D axis reference position.
Definition at line 667 of file gclib_record.h.

9.22.2.36 axis_d_status

`UW axis_d_status`
D axis status.
Definition at line 664 of file gclib_record.h.

9.22.2.37 axis_d_stop_code

`UB axis_d_stop_code`
D axis stop code.
Definition at line 666 of file gclib_record.h.

9.22.2.38 axis_d_switches

`UB axis_d_switches`
D axis switches.
Definition at line 665 of file gclib_record.h.

9.22.2.39 axis_d_torque

`SW axis_d_torque`
D axis torque.
Definition at line 672 of file gclib_record.h.

9.22.2.40 axis_d_velocity

`SL axis_d_velocity`
D axis velocity.
Definition at line 671 of file gclib_record.h.

9.22.2.41 axis_e_analog_in

`UW axis_e_analog_in`
E axis analog input.
Definition at line 684 of file gclib_record.h.

9.22.2.42 axis_e_aux_position

`SL axis_e_aux_position`
E axis auxiliary position.
Definition at line 681 of file gclib_record.h.

9.22.2.43 axis_e_motor_position

`SL axis_e_motor_position`
E axis motor position.
Definition at line 679 of file gclib_record.h.

9.22.2.44 axis_e_position_error

`SL axis_e_position_error`
E axis position error.
Definition at line 680 of file gclib_record.h.

9.22.2.45 axis_e_reference_position

`SL axis_e_reference_position`
E axis reference position.
Definition at line 678 of file gclib_record.h.

9.22.2.46 axis_e_status

`UW axis_e_status`
E axis status.
Definition at line 675 of file gclib_record.h.

9.22.2.47 axis_e_stop_code

`UB axis_e_stop_code`
E axis stop code.
Definition at line 677 of file gclib_record.h.

9.22.2.48 axis_e_switches

`UB axis_e_switches`
E axis switches.
Definition at line 676 of file gclib_record.h.

9.22.2.49 axis_e_torque

`SW axis_e_torque`
E axis torque.
Definition at line 683 of file gclib_record.h.

9.22.2.50 axis_e_velocity

`SL axis_e_velocity`
E axis velocity.
Definition at line 682 of file gclib_record.h.

9.22.2.51 axis_f_analog_in

`UW axis_f_analog_in`
F axis analog input.
Definition at line 695 of file gclib_record.h.

9.22.2.52 axis_f_aux_position

`SL axis_f_aux_position`
F axis auxiliary position.
Definition at line 692 of file gclib_record.h.

9.22.2.53 axis_f_motor_position

`SL axis_f_motor_position`
F axis motor position.
Definition at line 690 of file gclib_record.h.

9.22.2.54 axis_f_position_error

`SL axis_f_position_error`
F axis position error.
Definition at line 691 of file gclib_record.h.

9.22.2.55 axis_f_reference_position

`SL axis_f_reference_position`
F axis reference position.
Definition at line 689 of file gclib_record.h.

9.22.2.56 axis_f_status

`UW axis_f_status`
F axis status.
Definition at line 686 of file gclib_record.h.

9.22.2.57 axis_f_stop_code

`UB axis_f_stop_code`
F axis stop code.
Definition at line 688 of file gclib_record.h.

9.22.2.58 axis_f_switches

`UB axis_f_switches`
F axis switches.
Definition at line 687 of file gclib_record.h.

9.22.2.59 axis_f_torque

`SW axis_f_torque`
F axis torque.
Definition at line 694 of file gclib_record.h.

9.22.2.60 axis_f_velocity

`SL axis_f_velocity`
F axis velocity.
Definition at line 693 of file gclib_record.h.

9.22.2.61 axis_g_analog_in

`UW axis_g_analog_in`
G axis analog input.
Definition at line 706 of file gclib_record.h.

9.22.2.62 axis_g_aux_position

`SL axis_g_aux_position`
G axis auxiliary position.
Definition at line 703 of file gclib_record.h.

9.22.2.63 axis_g_motor_position

`SL axis_g_motor_position`
G axis motor position.
Definition at line 701 of file gclib_record.h.

9.22.2.64 axis_g_position_error

`SL axis_g_position_error`
G axis position error.
Definition at line 702 of file gclib_record.h.

9.22.2.65 axis_g_reference_position

`SL axis_g_reference_position`
G axis reference position.
Definition at line 700 of file gclib_record.h.

9.22.2.66 axis_g_status

`UW axis_g_status`
G axis status.
Definition at line 697 of file gclib_record.h.

9.22.2.67 axis_g_stop_code

`UB axis_g_stop_code`
G axis stop code.
Definition at line 699 of file gclib_record.h.

9.22.2.68 axis_g_switches

`UB axis_g_switches`
G axis switches.
Definition at line 698 of file gclib_record.h.

9.22.2.69 axis_g_torque

`SW axis_g_torque`
G axis torque.
Definition at line 705 of file gclib_record.h.

9.22.2.70 axis_g_velocity

`SL axis_g_velocity`
G axis velocity.
Definition at line 704 of file gclib_record.h.

9.22.2.71 axis_h_analog_in

`UW axis_h_analog_in`
H axis analog input.
Definition at line 717 of file gclib_record.h.

9.22.2.72 axis_h_aux_position

`SL axis_h_aux_position`
H axis auxiliary position.
Definition at line 714 of file gclib_record.h.

9.22.2.73 axis_h_motor_position

`SL axis_h_motor_position`
H axis motor position.
Definition at line 712 of file gclib_record.h.

9.22.2.74 axis_h_position_error

`SL axis_h_position_error`
H axis position error.
Definition at line 713 of file gclib_record.h.

9.22.2.75 axis_h_reference_position

`SL axis_h_reference_position`
H axis reference position.
Definition at line 711 of file gclib_record.h.

9.22.2.76 axis_h_status

`UW axis_h_status`
H axis status.
Definition at line 708 of file gclib_record.h.

9.22.2.77 axis_h_stop_code

`UB axis_h_stop_code`
H axis stop code.
Definition at line 710 of file gclib_record.h.

9.22.2.78 axis_h_switches

`UB axis_h_switches`
H axis switches.
Definition at line 709 of file gclib_record.h.

9.22.2.79 axis_h_torque

`SW axis_h_torque`
H axis torque.
Definition at line 716 of file gclib_record.h.

9.22.2.80 axis_h_velocity

`SL axis_h_velocity`
H axis velocity.
Definition at line 715 of file gclib_record.h.

9.22.2.81 error_code

`UB error_code`
error code.
Definition at line 620 of file gclib_record.h.

9.22.2.82 general_status

`UB general_status`
general status
Definition at line 621 of file gclib_record.h.

9.22.2.83 header_0

`UB header_0`
1st Byte of Header.
Definition at line 591 of file gclib_record.h.

9.22.2.84 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 592 of file gclib_record.h.

9.22.2.85 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 593 of file gclib_record.h.

9.22.2.86 header_3

`UB header_3`
4th Byte of Header.
Definition at line 594 of file gclib_record.h.

9.22.2.87 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 598 of file gclib_record.h.

9.22.2.88 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 599 of file gclib_record.h.

9.22.2.89 input_bank_2

`UB input_bank_2`
general input bank 2 (inputs 17-24).
Definition at line 600 of file gclib_record.h.

9.22.2.90 input_bank_3

`UB input_bank_3`
general input bank 3 (inputs 25-32).
Definition at line 601 of file gclib_record.h.

9.22.2.91 input_bank_4

`UB input_bank_4`
general input bank 4 (inputs 33-40).
Definition at line 602 of file gclib_record.h.

9.22.2.92 input_bank_5

`UB input_bank_5`
general input bank 5 (inputs 41-48).
Definition at line 603 of file gclib_record.h.

9.22.2.93 input_bank_6

`UB input_bank_6`
general input bank 6 (inputs 49-56).
Definition at line 604 of file gclib_record.h.

9.22.2.94 input_bank_7

`UB input_bank_7`
general input bank 7 (inputs 57-64).
Definition at line 605 of file gclib_record.h.

9.22.2.95 input_bank_8

`UB input_bank_8`
general input bank 8 (inputs 65-72).
Definition at line 606 of file gclib_record.h.

9.22.2.96 input_bank_9

`UB input_bank_9`
general input bank 9 (inputs 73-80).
Definition at line 607 of file gclib_record.h.

9.22.2.97 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 609 of file gclib_record.h.

9.22.2.98 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 610 of file gclib_record.h.

9.22.2.99 output_bank_2

`UB output_bank_2`
general output bank 2 (outputs 17-24).
Definition at line 611 of file gclib_record.h.

9.22.2.100 output_bank_3

`UB output_bank_3`
general output bank 3 (outputs 25-32).
Definition at line 612 of file gclib_record.h.

9.22.2.101 output_bank_4

`UB output_bank_4`
general output bank 4 (outputs 33-40).
Definition at line 613 of file gclib_record.h.

9.22.2.102 output_bank_5

`UB output_bank_5`
general output bank 5 (outputs 41-48).
Definition at line 614 of file gclib_record.h.

9.22.2.103 output_bank_6

`UB output_bank_6`
general output bank 6 (outputs 49-56).
Definition at line 615 of file gclib_record.h.

9.22.2.104 output_bank_7

`UB output_bank_7`
general output bank 7 (outputs 57-64).
Definition at line 616 of file gclib_record.h.

9.22.2.105 output_bank_8

`UB output_bank_8`
general output bank 8 (outputs 65-72).
Definition at line 617 of file gclib_record.h.

9.22.2.106 output_bank_9

`UB output_bank_9`
general output bank 9 (outputs 73-80).
Definition at line 618 of file gclib_record.h.

9.22.2.107 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 625 of file gclib_record.h.

9.22.2.108 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 624 of file gclib_record.h.

9.22.2.109 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 623 of file gclib_record.h.

9.22.2.110 sample_number

`UW sample_number`
sample number.
Definition at line 596 of file gclib_record.h.

9.22.2.111 t_distance

`SL t_distance`
distance traveled in coordinated move for T plane.
Definition at line 629 of file gclib_record.h.

9.22.2.112 t_plane_move_status

`UW t_plane_move_status`
Coordinated move status for T plane.
Definition at line 628 of file gclib_record.h.

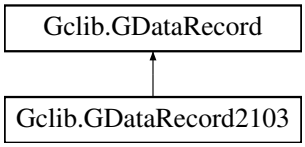
9.22.2.113 t_plane_segment_count

`UW t_plane_segment_count`
segment count of coordinated move for T plane.
Definition at line 627 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

9.23 Gclib.GDataRecord2103 Struct Reference

Inheritance diagram for Gclib.GDataRecord2103:



Public Member Functions

- general input **bank** (inputs 1-8). */public UB input_bank_1
  /07
- general input **bank** (inputs 9-16). */public UB input_bank_2
  /08
- general input **bank** (inputs 17-24). */public UB input_bank_3
  /09
- general input **bank** (inputs 25-32). */public UB input_bank_4
  /10
- general input **bank** (inputs 33-40). */public UB input_bank_5
  /11
- general input **bank** (inputs 41-48). */public UB input_bank_6
  /12
- general input **bank** (inputs 49-56). */public UB input_bank_7
  /13
- general input **bank** (inputs 57-64). */public UB input_bank_8
  /14
- general input **bank** (inputs 65-72). */public UB input_bank_9
  /15
- general input **bank** (inputs 73-80). */public UB output_bank_0
  /16
- general output **bank** (outputs 1-8). */public UB output_bank_1
  /17
- general output **bank** (outputs 9-16). */public UB output_bank_2
  /18
- general output **bank** (outputs 17-24). */public UB output_bank_3
  /19
- general output **bank** (outputs 25-32). */public UB output_bank_4
  /20
- general output **bank** (outputs 33-40). */public UB output_bank_5
  /21
- general output **bank** (outputs 41-48). */public UB output_bank_6
  /22
- general output **bank** (outputs 49-56). */public UB output_bank_7
  /23
- general output **bank** (outputs 57-64). */public UB output_bank_8
  /24
- general output **bank** (outputs 65-72). */public UB output_bank_9

/*25

- general output **bank** (outputs 73-80). */public UB error_code

  /*26

- H axis analog input.* byte_array ()

Data Fields

- UB header_0

  /*00

- byte of Header.*UB header_1

  /*01

- byte of Header.*UB header_2

  /*02

- byte of Header.*UB header_3

  /*03

- byte of Header.*UW sample_number

  /*04-05

- sample number.*UB input_bank_0

  /*06

- error code.*UB general_status

  /*27

- general status *UW s_plane_segment_count

  /*28-29

- segment count of coordinated move for S plane.*UW s_plane_move_status

  /*30-31

- coordinated move status for S plane.*SL s_distance

  /*32-35

- distance traveled in coordinated move for S plane.*UW t_plane_segment_count

  /*36-37

- segment count of coordinated move for T plane.*UW t_plane_move_status

  /*38-39

- Coordinated move status for T plane.*SL t_distance

  /*40-43

- distance traveled in coordinated move for T plane.*UW axis_a_status

  /*44-45

- A axis status.*UB axis_a_switches

  /*46

- A axis switches.*UB axis_a_stop_code

  /*47

- A axis stop code.*SL axis_a_reference_position

  /*48-51

- A axis reference position.*SL axis_a_motor_position

  /*52-55

- A axis motor position.*SL axis_a_position_error

  /*56-59

- A axis position error.*SL axis_a_aux_position

  /*60-63

- A axis auxiliary position.*SL axis_a_velocity

  /*64-67

- A axis velocity.*SW axis_a_torque

  /*68-69

- A axis torque.∗UW axis_a_analog_in
  /∗70-71
- A axis analog input.∗UW axis_b_status
  /∗72-73
- B axis status.∗UB axis_b_switches
  /∗74
- B axis switches.∗UB axis_b_stop_code
  /∗75
- B axis stop code.∗SL axis_b_reference_position
  /∗76-79
- B axis reference position.∗SL axis_b_motor_position
  /∗80-83
- B axis motor position.∗SL axis_b_position_error
  /∗84-87
- B axis position error.∗SL axis_b_aux_position
  /∗88-91
- B axis auxiliary position.∗SL axis_b_velocity
  /∗92-95
- B axis velocity.∗SW axis_b_torque
  /∗96-97
- B axis torque.∗UW axis_b_analog_in
  /∗98-99
- B axis analog input.∗UW axis_c_status
  /∗100-101
- C axis status.∗UB axis_c_switches
  /∗102
- C axis switches.∗UB axis_c_stop_code
  /∗103
- C axis stop code.∗SL axis_c_reference_position
  /∗104-107
- C axis reference position.∗SL axis_c_motor_position
  /∗108-111
- C axis motor position.∗SL axis_c_position_error
  /∗112-115
- C axis position error.∗SL axis_c_aux_position
  /∗116-119
- C axis auxiliary position.∗SL axis_c_velocity
  /∗120-123
- C axis velocity.∗SW axis_c_torque
  /∗124-125
- C axis torque.∗UW axis_c_analog_in
  /∗126-127
- C axis analog input.∗UW axis_d_status
  /∗128-129
- D axis status.∗UB axis_d_switches
  /∗130
- D axis switches.∗UB axis_d_stop_code
  /∗131
- D axis stop code.∗SL axis_d_reference_position
  /∗132-135
- D axis reference position.∗SL axis_d_motor_position

/*136-139

- D axis motor position.*SL axis_d_position_error

  /*140-143
- D axis position error.*SL axis_d_aux_position

  /*144-147
- D axis auxiliary position.*SL axis_d_velocity

  /*148-151
- D axis velocity.*SW axis_d_torque

  /*152-153
- D axis torque.*UW axis_d_analog_in

  /*154-155
- D axis analog input.*UW axis_e_status

  /*156-157
- E axis status.*UB axis_e_switches

  /*158
- E axis switches.*UB axis_e_stop_code

  /*159
- E axis stop code.*SL axis_e_reference_position

  /*160-163
- E axis reference position.*SL axis_e_motor_position

  /*164-167
- E axis motor position.*SL axis_e_position_error

  /*168-171
- E axis position error.*SL axis_e_aux_position

  /*172-175
- E axis auxiliary position.*SL axis_e_velocity

  /*176-179
- E axis velocity.*SW axis_e_torque

  /*180-181
- E axis torque.*UW axis_e_analog_in

  /*182-183
- E axis analog input.*UW axis_f_status

  /*184-185
- F axis status.*UB axis_f_switches

  /*186
- F axis switches.*UB axis_f_stop_code

  /*187
- F axis stop code.*SL axis_f_reference_position

  /*188-191
- F axis reference position.*SL axis_f_motor_position

  /*192-195
- F axis motor position.*SL axis_f_position_error

  /*196-199
- F axis position error.*SL axis_f_aux_position

  /*200-203
- F axis auxiliary position.*SL axis_f_velocity

  /*204-207
- F axis velocity.*SW axis_f_torque

  /*208-209
- F axis torque.*UW axis_f_analog_in

  /*210-211

- F axis analog input.∗UW axis_g_status
  /∗212-213
- G axis status.∗UB axis_g_switches
  /∗214
- G axis switches.∗UB axis_g_stop_code
  /∗215
- G axis stop code.∗SL axis_g_reference_position
  /∗216-219
- G axis reference position.∗SL axis_g_motor_position
  /∗220-223
- G axis motor position.∗SL axis_g_position_error
  /∗224-227
- G axis position error.∗SL axis_g_aux_position
  /∗228-231
- G axis auxiliary position.∗SL axis_g_velocity
  /∗232-235
- G axis velocity.∗SW axis_g_torque
  /∗236-237
- G axis torque.∗UW axis_g_analog_in
  /∗238-239
- G axis analog input.∗UW axis_h_status
  /∗240-241
- H axis status.∗UB axis_h_switches
  /∗242
- H axis switches.∗UB axis_h_stop_code
  /∗243
- H axis stop code.∗SL axis_h_reference_position
  /∗244-247
- H axis reference position.∗SL axis_h_motor_position
  /∗248-251
- H axis motor position.∗SL axis_h_position_error
  /∗252-255
- H axis position error.∗SL axis_h_aux_position
  /∗256-259
- H axis auxiliary position.∗SL axis_h_velocity
  /∗260-263
- H axis velocity.∗SW axis_h_torque
  /∗264-265
- H axis torque.∗UW axis_h_analog_in
  /∗266-267

9.23.1 Detailed Description

Definition at line 958 of file gclib.vb.

9.23.2 Member Function Documentation

9.23.2.1 byte_array()

```
H axis analog input.* byte_array ( )
```

Implements GDataRecord.byte_array
Implements Gclib.GDataRecord.

9.23.3 Field Documentation

9.23.3.1 axis_a_analog_in

```
A axis torque.* UW axis_a_analog_in
```

/*70-71
Definition at line 1003 of file gclib.vb.

9.23.3.2 axis_a_aux_position

```
A axis position error.* SL axis_a_aux_position
```

/*60-63
Definition at line 1000 of file gclib.vb.

9.23.3.3 axis_a_motor_position

```
A axis reference position.* SL axis_a_motor_position
```

/*52-55
Definition at line 998 of file gclib.vb.

9.23.3.4 axis_a_position_error

```
A axis motor position.* SL axis_a_position_error
```

/*56-59
Definition at line 999 of file gclib.vb.

9.23.3.5 axis_a_reference_position

```
A axis stop code.* SL axis_a_reference_position
```

/*48-51
Definition at line 997 of file gclib.vb.

9.23.3.6 axis_a_status

```
distance traveled in coordinated move for T plane.* UW axis_a_status
```

/*44-45
Definition at line 994 of file gclib.vb.

9.23.3.7 axis_a_stop_code

```
A axis switches.* UB axis_a_stop_code
```

/*47
Definition at line 996 of file gclib.vb.

9.23.3.8 axis_a_switches

```
A axis status.* UB axis_a_switches
```

/*46
Definition at line 995 of file gclib.vb.

9.23.3.9 axis_a_torque

```
A axis velocity.* SW axis_a_torque
```

/*68-69
Definition at line 1002 of file gclib.vb.

9.23.3.10 axis_a_velocity

```
A axis auxiliary position.* SL axis_a_velocity
```

/*64-67
Definition at line 1001 of file gclib.vb.

9.23.3.11 axis_b_analog_in

```
B axis torque.* UW axis_b_analog_in
```
/*98-99
Definition at line 1013 of file gclib.vb.

9.23.3.12 axis_b_aux_position

```
B axis position error.* SL axis_b_aux_position
```
/*88-91
Definition at line 1010 of file gclib.vb.

9.23.3.13 axis_b_motor_position

```
B axis reference position.* SL axis_b_motor_position
```
/*80-83
Definition at line 1008 of file gclib.vb.

9.23.3.14 axis_b_position_error

```
B axis motor position.* SL axis_b_position_error
```
/*84-87
Definition at line 1009 of file gclib.vb.

9.23.3.15 axis_b_reference_position

```
B axis stop code.* SL axis_b_reference_position
```
/*76-79
Definition at line 1007 of file gclib.vb.

9.23.3.16 axis_b_status

```
A axis analog input.* UW axis_b_status
```
/*72-73
Definition at line 1004 of file gclib.vb.

9.23.3.17 axis_b_stop_code

```
B axis switches.* UB axis_b_stop_code
```
/*75
Definition at line 1006 of file gclib.vb.

9.23.3.18 axis_b_switches

```
B axis status.* UB axis_b_switches
```
/*74
Definition at line 1005 of file gclib.vb.

9.23.3.19 axis_b_torque

```
B axis velocity.* SW axis_b_torque
```
/*96-97
Definition at line 1012 of file gclib.vb.

9.23.3.20 axis_b_velocity

```
B axis auxiliary position.* SL axis_b_velocity
```
/*92-95
Definition at line 1011 of file gclib.vb.

9.23.3.21 axis_c_analog_in

```
C axis torque.* UW axis_c_analog_in
```
/*126-127
Definition at line 1023 of file gclib.vb.

9.23.3.22 axis_c_aux_position

```
C axis position error.* SL axis_c_aux_position
```
/*116-119
Definition at line 1020 of file gclib.vb.

9.23.3.23 axis_c_motor_position

```
C axis reference position.* SL axis_c_motor_position
```
/*108-111
Definition at line 1018 of file gclib.vb.

9.23.3.24 axis_c_position_error

```
C axis motor position.* SL axis_c_position_error
```
/*112-115
Definition at line 1019 of file gclib.vb.

9.23.3.25 axis_c_reference_position

```
C axis stop code.* SL axis_c_reference_position
```
/*104-107
Definition at line 1017 of file gclib.vb.

9.23.3.26 axis_c_status

```
B axis analog input.* UW axis_c_status
```
/*100-101
Definition at line 1014 of file gclib.vb.

9.23.3.27 axis_c_stop_code

```
C axis switches.* UB axis_c_stop_code
```
/*103
Definition at line 1016 of file gclib.vb.

9.23.3.28 axis_c_switches

```
C axis status.* UB axis_c_switches
```
/*102
Definition at line 1015 of file gclib.vb.

9.23.3.29 axis_c_torque

```
C axis velocity.* SW axis_c_torque
```
/*124-125
Definition at line 1022 of file gclib.vb.

9.23.3.30 axis_c_velocity

```
C axis auxiliary position.* SL axis_c_velocity
```
/*120-123
Definition at line 1021 of file gclib.vb.

9.23.3.31 axis_d_analog_in

```
D axis torque.* UW axis_d_analog_in
```
/*154-155
Definition at line 1033 of file gclib.vb.

9.23.3.32 axis_d_aux_position

```
D axis position error.* SL axis_d_aux_position
```
/*144-147
Definition at line 1030 of file gclib.vb.

9.23.3.33 axis_d_motor_position

```
D axis reference position.* SL axis_d_motor_position
```
/*136-139
Definition at line 1028 of file gclib.vb.

9.23.3.34 axis_d_position_error

```
D axis motor position.* SL axis_d_position_error
```
/*140-143
Definition at line 1029 of file gclib.vb.

9.23.3.35 axis_d_reference_position

```
D axis stop code.* SL axis_d_reference_position
```
/*132-135
Definition at line 1027 of file gclib.vb.

9.23.3.36 axis_d_status

```
C axis analog input.* UW axis_d_status
```
/*128-129
Definition at line 1024 of file gclib.vb.

9.23.3.37 axis_d_stop_code

```
D axis switches.* UB axis_d_stop_code
```
/*131
Definition at line 1026 of file gclib.vb.

9.23.3.38 axis_d_switches

```
D axis status.* UB axis_d_switches
```
/*130
Definition at line 1025 of file gclib.vb.

9.23.3.39 axis_d_torque

```
D axis velocity.* SW axis_d_torque
```
/*152-153
Definition at line 1032 of file gclib.vb.

9.23.3.40 axis_d_velocity

```
D axis auxiliary position.* SL axis_d_velocity
```
/*148-151
Definition at line 1031 of file gclib.vb.

9.23.3.41 axis_e_analog_in

```
E axis torque.* UW axis_e_analog_in
```
/*182-183
Definition at line 1043 of file gclib.vb.

9.23.3.42 axis_e_aux_position

```
E axis position error.* SL axis_e_aux_position
```
/*172-175
Definition at line 1040 of file gclib.vb.

9.23.3.43 axis_e_motor_position

```
E axis reference position.* SL axis_e_motor_position
```
/*164-167
Definition at line 1038 of file gclib.vb.

9.23.3.44 axis_e_position_error

```
E axis motor position.* SL axis_e_position_error
```
/*168-171
Definition at line 1039 of file gclib.vb.

9.23.3.45 axis_e_reference_position

```
E axis stop code.* SL axis_e_reference_position
```
/*160-163
Definition at line 1037 of file gclib.vb.

9.23.3.46 axis_e_status

```
D axis analog input.* UW axis_e_status
```
/*156-157
Definition at line 1034 of file gclib.vb.

9.23.3.47 axis_e_stop_code

```
E axis switches.* UB axis_e_stop_code
```
/*159
Definition at line 1036 of file gclib.vb.

9.23.3.48 axis_e_switches

```
E axis status.* UB axis_e_switches
```
/*158
Definition at line 1035 of file gclib.vb.

9.23.3.49 axis_e_torque

```
E axis velocity.* SW axis_e_torque
```
/*180-181
Definition at line 1042 of file gclib.vb.

9.23.3.50 axis_e_velocity

```
E axis auxiliary position.* SL axis_e_velocity
```
/*176-179
Definition at line 1041 of file gclib.vb.

9.23.3.51 axis_f_analog_in

```
F axis torque.* UW axis_f_analog_in
```
/*210-211
Definition at line 1053 of file gclib.vb.

9.23.3.52 axis_f_aux_position

```
F axis position error.* SL axis_f_aux_position
```
/*200-203
Definition at line 1050 of file gclib.vb.

9.23.3.53 axis_f_motor_position

```
F axis reference position.* SL axis_f_motor_position
```
/*192-195
Definition at line 1048 of file gclib.vb.

9.23.3.54 axis_f_position_error

```
F axis motor position.* SL axis_f_position_error
```
/*196-199
Definition at line 1049 of file gclib.vb.

9.23.3.55 axis_f_reference_position

```
F axis stop code.* SL axis_f_reference_position
```
/*188-191
Definition at line 1047 of file gclib.vb.

9.23.3.56 axis_f_status

```
E axis analog input.* UW axis_f_status
```
/*184-185
Definition at line 1044 of file gclib.vb.

9.23.3.57 axis_f_stop_code

```
F axis switches.* UB axis_f_stop_code
```
/*187
Definition at line 1046 of file gclib.vb.

9.23.3.58 axis_f_switches

```
F axis status.* UB axis_f_switches
```
/*186
Definition at line 1045 of file gclib.vb.

9.23.3.59 axis_f_torque

```
F axis velocity.* SW axis_f_torque
```
/*208-209
Definition at line 1052 of file gclib.vb.

9.23.3.60 axis_f_velocity

```
F axis auxiliary position.* SL axis_f_velocity
```
/*204-207
Definition at line 1051 of file gclib.vb.

9.23.3.61 axis_g_analog_in

```
G axis torque.* UW axis_g_analog_in
```

/*238-239
Definition at line 1063 of file gclib.vb.

9.23.3.62 axis_g_aux_position

```
G axis position error.* SL axis_g_aux_position
```

/*228-231
Definition at line 1060 of file gclib.vb.

9.23.3.63 axis_g_motor_position

```
G axis reference position.* SL axis_g_motor_position
```

/*220-223
Definition at line 1058 of file gclib.vb.

9.23.3.64 axis_g_position_error

```
G axis motor position.* SL axis_g_position_error
```

/*224-227
Definition at line 1059 of file gclib.vb.

9.23.3.65 axis_g_reference_position

```
G axis stop code.* SL axis_g_reference_position
```

/*216-219
Definition at line 1057 of file gclib.vb.

9.23.3.66 axis_g_status

```
F axis analog input.* UW axis_g_status
```

/*212-213
Definition at line 1054 of file gclib.vb.

9.23.3.67 axis_g_stop_code

```
G axis switches.* UB axis_g_stop_code
```

/*215
Definition at line 1056 of file gclib.vb.

9.23.3.68 axis_g_switches

```
G axis status.* UB axis_g_switches
```

/*214
Definition at line 1055 of file gclib.vb.

9.23.3.69 axis_g_torque

```
G axis velocity.* SW axis_g_torque
```

/*236-237
Definition at line 1062 of file gclib.vb.

9.23.3.70 axis_g_velocity

```
G axis auxiliary position.* SL axis_g_velocity
```

/*232-235
Definition at line 1061 of file gclib.vb.

9.23.3.71 axis_h_analog_in

```
H axis torque.* UW axis_h_analog_in
```
/*266-267
Definition at line 1073 of file gclib.vb.

9.23.3.72 axis_h_aux_position

```
H axis position error.* SL axis_h_aux_position
```
/*256-259
Definition at line 1070 of file gclib.vb.

9.23.3.73 axis_h_motor_position

```
H axis reference position.* SL axis_h_motor_position
```
/*248-251
Definition at line 1068 of file gclib.vb.

9.23.3.74 axis_h_position_error

```
H axis motor position.* SL axis_h_position_error
```
/*252-255
Definition at line 1069 of file gclib.vb.

9.23.3.75 axis_h_reference_position

```
H axis stop code.* SL axis_h_reference_position
```
/*244-247
Definition at line 1067 of file gclib.vb.

9.23.3.76 axis_h_status

```
G axis analog input.* UW axis_h_status
```
/*240-241
Definition at line 1064 of file gclib.vb.

9.23.3.77 axis_h_stop_code

```
H axis switches.* UB axis_h_stop_code
```
/*243
Definition at line 1066 of file gclib.vb.

9.23.3.78 axis_h_switches

```
H axis status.* UB axis_h_switches
```
/*242
Definition at line 1065 of file gclib.vb.

9.23.3.79 axis_h_torque

```
H axis velocity.* SW axis_h_torque
```
/*264-265
Definition at line 1072 of file gclib.vb.

9.23.3.80 axis_h_velocity

```
H axis auxiliary position.* SL axis_h_velocity
```
/*260-263
Definition at line 1071 of file gclib.vb.

9.23.3.81 general_status

```
error code.* UB general_status
```
/*27
Definition at line 987 of file gclib.vb.

9.23.3.82 header_0

```
UB header_0
```
/*00
Definition at line 961 of file gclib.vb.

9.23.3.83 header_1

```
byte of Header.* UB header_1
```
/*01
Definition at line 962 of file gclib.vb.

9.23.3.84 header_2

```
byte of Header.* UB header_2
```
/*02
Definition at line 963 of file gclib.vb.

9.23.3.85 header_3

```
byte of Header.* UB header_3
```
/*03
Definition at line 964 of file gclib.vb.

9.23.3.86 input_bank_0

```
sample number.* UB input_bank_0
```
/*06
Definition at line 966 of file gclib.vb.

9.23.3.87 s_distance

```
coordinated move status for S plane.* SL s_distance
```
/*32-35
Definition at line 990 of file gclib.vb.

9.23.3.88 s_plane_move_status

```
segment count of coordinated move for S plane.* UW s_plane_move_status
```
/*30-31
Definition at line 989 of file gclib.vb.

9.23.3.89 s_plane_segment_count

```
general status* UW s_plane_segment_count
```
/*28-29
Definition at line 988 of file gclib.vb.

9.23.3.90 sample_number

```
byte of Header.* UW sample_number
```
/*04-05
Definition at line 965 of file gclib.vb.

9.23.3.91 t_distance

```
Coordinated move status for T plane.* SL t_distance
```
/*40-43
Definition at line 993 of file gclib.vb.

9.23.3.92 t_plane_move_status

```
segment count of coordinated move for T plane.* UW t_plane_move_status
```
/*38-39
Definition at line 992 of file gclib.vb.

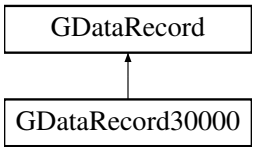
9.23.3.93 t_plane_segment_count

```
distance traveled in coordinated move for S plane.* UW t_plane_segment_count
```
/*36-37
Definition at line 991 of file gclib.vb.
The documentation for this struct was generated from the following file:

- gclib.vb

9.24 GDataRecord30000 Struct Reference

Data record struct for DMC-30010 controllers.
Inheritance diagram for GDataRecord30000:



Public Member Functions

- byte[] byte_array ()

  Returns the data record as a byte array and allows for access to individual bytes.

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.
- UB header_3

  4th Byte of Header.
- UW sample_number

  sample number.
- UB input_bank_0

  general input bank 0 (inputs 1-8).
- UB input_bank_1

  general input bank 1 (inputs 9-16).
- UB output_bank_0

  general output bank 0 (outputs 1-8).

- UB output_bank_1

  general output bank 1 (outputs 9-16).
- UB error_code

  error code.
- UB thread_status

  thread status.
- UW input_analog_2

  Analog input 2. 1 is in axis data, see axis_a_analog_in.
- UW output_analog_1

  Analog output 1.
- UW output_analog_2

  Analog output 2.
- UL amplifier_status

  Amplifier Status.
- UL contour_segment_count

  Segment Count for Contour Mode.
- UW contour_buffer_available

  Buffer space remaining, Contour Mode.
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

  coordinated move status for S plane.
- SL s_distance

  distance traveled in coordinated move for S plane.
- UW s_plane_buffer_available

  Buffer space remaining, S Plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

  A axis stop code.
- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

  A axis auxiliary position.
- SL axis_a_velocity

  A axis velocity.
- SL axis_a_torque

  A axis torque.
- UW axis_a_analog_in

  A axis analog input.
- UB axis_a_halls

  A Hall Input Status.
- UB axis_a_reserved

  Reserved.
- SL axis_a_variable

  A User-defined variable (ZA).

9.24.1 Detailed Description

Data record struct for DMC-30010 controllers.
Definition at line 1708 of file gclib.cs.

9.24.2 Member Function Documentation

9.24.2.1 byte_array()

```
byte[] byte_array ( )  [inline]
```

Returns the data record as a byte array and allows for access to individual bytes.
Implements GDataRecord.
Definition at line 1710 of file gclib.cs.

9.24.3 Field Documentation

9.24.3.1 amplifier_status

```
UL amplifier_status
```

Amplifier Status.
Definition at line 1735 of file gclib.cs.

9.24.3.2 axis_a_analog_in

```
UW axis_a_analog_in
```

A axis analog input.
Definition at line 1754 of file gclib.cs.

9.24.3.3 axis_a_aux_position

```
SL axis_a_aux_position
```

A axis auxiliary position.
Definition at line 1751 of file gclib.cs.

9.24.3.4 axis_a_halls

```
UB axis_a_halls
```

A Hall Input Status.
Definition at line 1755 of file gclib.cs.

9.24.3.5 axis_a_motor_position

```
SL axis_a_motor_position
```

A axis motor position.
Definition at line 1749 of file gclib.cs.

9.24.3.6 axis_a_position_error

```
SL axis_a_position_error
```

A axis position error.
Definition at line 1750 of file gclib.cs.

9.24.3.7 axis_a_reference_position

```
SL axis_a_reference_position
```

A axis reference position.
Definition at line 1748 of file gclib.cs.

9.24.3.8 axis_a_reserved

`UB axis_a_reserved`
Reserved.
Definition at line 1756 of file gclib.cs.

9.24.3.9 axis_a_status

`UW axis_a_status`
A axis status.
Definition at line 1745 of file gclib.cs.

9.24.3.10 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 1747 of file gclib.cs.

9.24.3.11 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 1746 of file gclib.cs.

9.24.3.12 axis_a_torque

`SL axis_a_torque`
A axis torque.
Definition at line 1753 of file gclib.cs.

9.24.3.13 axis_a_variable

`SL axis_a_variable`
A User-defined variable (ZA).
Definition at line 1757 of file gclib.cs.

9.24.3.14 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 1752 of file gclib.cs.

9.24.3.15 contour_buffer_available

`UW contour_buffer_available`
Buffer space remaining, Contour Mode.
Definition at line 1738 of file gclib.cs.

9.24.3.16 contour_segment_count

`UL contour_segment_count`
Segment Count for Contour Mode.
Definition at line 1737 of file gclib.cs.

9.24.3.17 error_code

`UB error_code`
error code.
Definition at line 1727 of file gclib.cs.

9.24.3.18 header_0

`UB header_0`
1st Byte of Header.
Definition at line 1714 of file gclib.cs.

9.24.3.19 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 1715 of file gclib.cs.

9.24.3.20 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 1716 of file gclib.cs.

9.24.3.21 header_3

`UB header_3`
4th Byte of Header.
Definition at line 1717 of file gclib.cs.

9.24.3.22 input_analog_2

`UW input_analog_2`
Analog input 2. 1 is in axis data, see axis_a_analog_in.
Definition at line 1730 of file gclib.cs.

9.24.3.23 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 1721 of file gclib.cs.

9.24.3.24 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 1722 of file gclib.cs.

9.24.3.25 output_analog_1

`UW output_analog_1`
Analog output 1.
Definition at line 1732 of file gclib.cs.

9.24.3.26 output_analog_2

`UW output_analog_2`
Analog output 2.
Definition at line 1733 of file gclib.cs.

9.24.3.27 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 1724 of file gclib.cs.

9.24.3.28 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 1725 of file gclib.cs.

9.24.3.29 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 1742 of file gclib.cs.

9.24.3.30 s_plane_buffer_available

`UW s_plane_buffer_available`
Buffer space remaining, S Plane.
Definition at line 1743 of file gclib.cs.

9.24.3.31 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 1741 of file gclib.cs.

9.24.3.32 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 1740 of file gclib.cs.

9.24.3.33 sample_number

`UW sample_number`
sample number.
Definition at line 1719 of file gclib.cs.

9.24.3.34 thread_status

`UB thread_status`
thread status.
Definition at line 1728 of file gclib.cs.
The documentation for this struct was generated from the following file:

- gclib.cs

9.25 GDataRecord30000 Struct Reference

Data record struct for DMC-30010 controllers.
`#include <gclib_record.h>`

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.

- UB header_3

  4th Byte of Header.
- UW sample_number

  sample number.
- UB input_bank_0

  general input bank 0 (inputs 1-8).
- UB input_bank_1

  general input bank 1 (inputs 9-16).
- UB output_bank_0

  general output bank 0 (outputs 1-8).
- UB output_bank_1

  general output bank 1 (outputs 9-16).
- UB error_code

  error code.
- UB thread_status

  thread status.
- UW input_analog_2

  Analog input 2. 1 is in axis data, see axis_a_analog_in.
- UW output_analog_1

  Analog output 1.
- UW output_analog_2

  Analog output 2.
- UL amplifier_status

  Amplifier Status.
- UL contour_segment_count

  Segment Count for Contour Mode.
- UW contour_buffer_available

  Buffer space remaining, Contour Mode.
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

  coordinated move status for S plane.
- SL s_distance

  distance traveled in coordinated move for S plane.
- UW s_plane_buffer_available

  Buffer space remaining, S Plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

  A axis stop code.
- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

  A axis auxiliary position.
- SL axis_a_velocity

A axis velocity.

- SL axis_a_torque

  A axis torque.

- UW axis_a_analog_in

  A axis analog input.

- UB axis_a_halls

  A Hall Input Status.

- UB axis_a_reserved

  Reserved.

- SL axis_a_variable

  A User-defined variable (ZA).

9.25.1 Detailed Description

Data record struct for DMC-30010 controllers.
Definition at line 818 of file gclib_record.h.

9.25.2 Field Documentation

9.25.2.1 amplifier_status

`UL amplifier_status`
Amplifier Status.
Definition at line 844 of file gclib_record.h.

9.25.2.2 axis_a_analog_in

`UW axis_a_analog_in`
A axis analog input.
Definition at line 863 of file gclib_record.h.

9.25.2.3 axis_a_aux_position

`SL axis_a_aux_position`
A axis auxiliary position.
Definition at line 860 of file gclib_record.h.

9.25.2.4 axis_a_halls

`UB axis_a_halls`
A Hall Input Status.
Definition at line 864 of file gclib_record.h.

9.25.2.5 axis_a_motor_position

`SL axis_a_motor_position`
A axis motor position.
Definition at line 858 of file gclib_record.h.

9.25.2.6 axis_a_position_error

`SL axis_a_position_error`
A axis position error.
Definition at line 859 of file gclib_record.h.

9.25.2.7 axis_a_reference_position

`SL axis_a_reference_position`
A axis reference position.
Definition at line 857 of file gclib_record.h.

9.25.2.8 axis_a_reserved

`UB axis_a_reserved`
Reserved.
Definition at line 865 of file gclib_record.h.

9.25.2.9 axis_a_status

`UW axis_a_status`
A axis status.
Definition at line 854 of file gclib_record.h.

9.25.2.10 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 856 of file gclib_record.h.

9.25.2.11 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 855 of file gclib_record.h.

9.25.2.12 axis_a_torque

`SL axis_a_torque`
A axis torque.
Definition at line 862 of file gclib_record.h.

9.25.2.13 axis_a_variable

`SL axis_a_variable`
A User-defined variable (ZA).
Definition at line 866 of file gclib_record.h.

9.25.2.14 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 861 of file gclib_record.h.

9.25.2.15 contour_buffer_available

`UW contour_buffer_available`
Buffer space remaining, Contour Mode.
Definition at line 847 of file gclib_record.h.

9.25.2.16 contour_segment_count

`UL contour_segment_count`
Segment Count for Contour Mode.
Definition at line 846 of file gclib_record.h.

9.25.2.17 error_code

`UB error_code`
error code.
Definition at line 836 of file gclib_record.h.

9.25.2.18 header_0

`UB header_0`
1st Byte of Header.
Definition at line 823 of file gclib_record.h.

9.25.2.19 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 824 of file gclib_record.h.

9.25.2.20 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 825 of file gclib_record.h.

9.25.2.21 header_3

`UB header_3`
4th Byte of Header.
Definition at line 826 of file gclib_record.h.

9.25.2.22 input_analog_2

`UW input_analog_2`
Analog input 2. 1 is in axis data, see axis_a_analog_in.
Definition at line 839 of file gclib_record.h.

9.25.2.23 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 830 of file gclib_record.h.

9.25.2.24 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 831 of file gclib_record.h.

9.25.2.25 output_analog_1

`UW output_analog_1`
Analog output 1.
Definition at line 841 of file gclib_record.h.

9.25.2.26 output_analog_2

`UW output_analog_2`
Analog output 2.
Definition at line 842 of file gclib_record.h.

9.25.2.27 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 833 of file gclib_record.h.

9.25.2.28 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 834 of file gclib_record.h.

9.25.2.29 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 851 of file gclib_record.h.

9.25.2.30 s_plane_buffer_available

`UW s_plane_buffer_available`
Buffer space remaining, S Plane.
Definition at line 852 of file gclib_record.h.

9.25.2.31 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 850 of file gclib_record.h.

9.25.2.32 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 849 of file gclib_record.h.

9.25.2.33 sample_number

`UW sample_number`
sample number.
Definition at line 828 of file gclib_record.h.

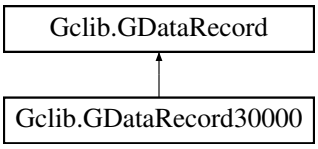
9.25.2.34 thread_status

`UB thread_status`
thread status.
Definition at line 837 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

9.26 Gclib.GDataRecord30000 Struct Reference

Inheritance diagram for Gclib.GDataRecord30000:



Public Member Functions

- general input **bank** (inputs 1-8). ∗/public UB input_bank_1
  /∗07
- general input **bank** (inputs 9-16). ∗/public UB output_bank_0
  /∗08
- general output **bank** (outputs 1-8). ∗/public UB output_bank_1
  /∗09
- general output **bank** (outputs 9-16). ∗/public UB error_code
  /∗10
- A User defined variable.∗[] byte_array ()

Data Fields

- UB header_0
  /∗00
- byte of Header.∗UB header_1
  /∗01
- byte of Header.∗UB header_2
  /∗02
- byte of Header.∗UB header_3
  /∗03
- byte of Header.∗UW sample_number
  /∗04-05
- sample number.∗UB input_bank_0
  /∗06
- error code.∗UB thread_status
  /∗11
- thread status.∗UW input_analog_2
  /∗12-13
- Analog input.is in axis data
- Analog input.is in axis see axis_a_analog_in.∗UW output_analog_1
  /∗14-15
- Analog output.∗UW output_analog_2
  /∗16-17
- Analog output.∗UL amplifier_status
  /∗18-21
- Amplifier Status.∗UL contour_segment_count
  /∗22-25
- Segment Count for Contour Mode.∗UW contour_buffer_available
  /∗26-27
- Buffer space remaining
- Buffer space Contour Mode.∗UW s_plane_segment_count
  /∗28-29
- segment count of coordinated move for S plane.∗UW s_plane_move_status
  /∗30-31

- coordinated move status for S plane.∗SL s_distance
  /∗32-35
- distance traveled in coordinated move for S plane.∗UW s_plane_buffer_available
  /∗36-37
- Buffer space S Plane.∗UW axis_a_status
  /∗38-39
- A axis status.∗UB axis_a_switches
  /∗40
- A axis switches.∗UB axis_a_stop_code
  /∗41
- A axis stop code.∗SL axis_a_reference_position
  /∗42-45
- A axis reference position.∗SL axis_a_motor_position
  /∗46-49
- A axis motor position.∗SL axis_a_position_error
  /∗50-53
- A axis position error.∗SL axis_a_aux_position
  /∗54-57
- A axis auxiliary position.∗SL axis_a_velocity
  /∗58-61
- A axis velocity.∗SL axis_a_torque
  /∗62-65
- A axis torque.∗UW axis_a_analog_in
  /∗66-67
- A axis analog input.∗UB axis_a_halls
  /∗68
- A Hall Input Status.∗UB axis_a_reserved
  /∗69
- Reserved.∗SL axis_a_variable
  /∗70-73

9.26.1 Detailed Description

Definition at line 1154 of file gclib.vb.

9.26.2 Member Function Documentation

9.26.2.1 byte_array()

```
A User defined variable.∗[] byte_array ( )
```

Implements GDataRecord.byte_array
Implements Gclib.GDataRecord.

9.26.3 Field Documentation

9.26.3.1 amplifier_status

```
Analog output.∗ UL amplifier_status
```

/∗18-21
Definition at line 1171 of file gclib.vb.

9.26.3.2 axis_a_analog_in

```
A axis torque.∗ UW axis_a_analog_in
```

/∗66-67
Definition at line 1187 of file gclib.vb.

9.26.3.3 axis_a_aux_position

A axis position error.* SL axis_a_aux_position
/*54-57
Definition at line 1184 of file gclib.vb.

9.26.3.4 axis_a_halls

A axis analog input.* UB axis_a_halls
/*68
Definition at line 1188 of file gclib.vb.

9.26.3.5 axis_a_motor_position

A axis reference position.* SL axis_a_motor_position
/*46-49
Definition at line 1182 of file gclib.vb.

9.26.3.6 axis_a_position_error

A axis motor position.* SL axis_a_position_error
/*50-53
Definition at line 1183 of file gclib.vb.

9.26.3.7 axis_a_reference_position

A axis stop code.* SL axis_a_reference_position
/*42-45
Definition at line 1181 of file gclib.vb.

9.26.3.8 axis_a_reserved

A Hall Input Status.* UB axis_a_reserved
/*69
Definition at line 1189 of file gclib.vb.

9.26.3.9 axis_a_status

Buffer space S Plane.* UW axis_a_status
/*38-39
Definition at line 1178 of file gclib.vb.

9.26.3.10 axis_a_stop_code

A axis switches.* UB axis_a_stop_code
/*41
Definition at line 1180 of file gclib.vb.

9.26.3.11 axis_a_switches

A axis status.* UB axis_a_switches
/*40
Definition at line 1179 of file gclib.vb.

9.26.3.12 axis_a_torque

A axis velocity.* SL axis_a_torque
/*62-65
Definition at line 1186 of file gclib.vb.

9.26.3.13 axis_a_variable

```
Reserved.* SL axis_a_variable
```
/*70-73
Definition at line 1190 of file gclib.vb.

9.26.3.14 axis_a_velocity

```
A axis auxiliary position.* SL axis_a_velocity
```
/*58-61
Definition at line 1185 of file gclib.vb.

9.26.3.15 contour_buffer_available

```
Segment Count for Contour Mode.* UW contour_buffer_available
```
/*26-27
Definition at line 1173 of file gclib.vb.

9.26.3.16 contour_segment_count

```
Amplifier Status.* UL contour_segment_count
```
/*22-25
Definition at line 1172 of file gclib.vb.

9.26.3.17 data

```
Analog input.is in axis data
```
Definition at line 1168 of file gclib.vb.

9.26.3.18 header_0

```
UB header_0
```
/*00
Definition at line 1157 of file gclib.vb.

9.26.3.19 header_1

```
byte of Header.* UB header_1
```
/*01
Definition at line 1158 of file gclib.vb.

9.26.3.20 header_2

```
byte of Header.* UB header_2
```
/*02
Definition at line 1159 of file gclib.vb.

9.26.3.21 header_3

```
byte of Header.* UB header_3
```
/*03
Definition at line 1160 of file gclib.vb.

9.26.3.22 input_analog_2

```
thread status.* UW input_analog_2
```
/*12-13
Definition at line 1168 of file gclib.vb.

9.26.3.23 input_bank_0

```
sample number.* UB input_bank_0
```
/*06
Definition at line 1162 of file gclib.vb.

9.26.3.24 output_analog_1

```
Analog input.is in axis see axis_a_analog_in.* UW output_analog_1
```
/*14-15
Definition at line 1169 of file gclib.vb.

9.26.3.25 output_analog_2

```
Analog output.* UW output_analog_2
```
/*16-17
Definition at line 1170 of file gclib.vb.

9.26.3.26 remaining

```
Buffer space remaining
```
Definition at line 1173 of file gclib.vb.

9.26.3.27 s_distance

```
coordinated move status for S plane.* SL s_distance
```
/*32-35
Definition at line 1176 of file gclib.vb.

9.26.3.28 s_plane_buffer_available

```
distance traveled in coordinated move for S plane.* UW s_plane_buffer_available
```
/*36-37
Definition at line 1177 of file gclib.vb.

9.26.3.29 s_plane_move_status

```
segment count of coordinated move for S plane.* UW s_plane_move_status
```
/*30-31
Definition at line 1175 of file gclib.vb.

9.26.3.30 s_plane_segment_count

```
Buffer space Contour Mode.* UW s_plane_segment_count
```
/*28-29
Definition at line 1174 of file gclib.vb.

9.26.3.31 sample_number

```
byte of Header.* UW sample_number
```
/*04-05
Definition at line 1161 of file gclib.vb.

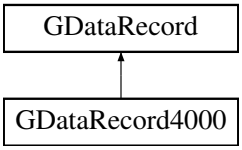
9.26.3.32 thread_status

```
error code.* UB thread_status
```
/*11
Definition at line 1167 of file gclib.vb.
The documentation for this struct was generated from the following file:

- gclib.vb

9.27 GDataRecord4000 Struct Reference

Data record struct for DMC-4000 controllers, including 4000, 4200, 4103, and 500x0.
Inheritance diagram for GDataRecord4000:



Public Member Functions

- byte[] byte_array ()

  Returns the data record as a byte array and allows for access to individual bytes.

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.
- UB header_3

  4th Byte of Header.
- UW sample_number

  sample number.
- UB input_bank_0

  general input bank 0 (inputs 1-8).
- UB input_bank_1

  general input bank 1 (inputs 9-16).
- UB input_bank_2

  general input bank 2 (inputs 17-24).
- UB input_bank_3

  general input bank 3 (inputs 25-32).
- UB input_bank_4

  general input bank 4 (inputs 33-40).
- UB input_bank_5

  general input bank 5 (inputs 41-48).
- UB input_bank_6

  general input bank 6 (inputs 49-56).
- UB input_bank_7

  general input bank 7 (inputs 57-64).
- UB input_bank_8

  general input bank 8 (inputs 65-72).
- UB input_bank_9

  general input bank 9 (inputs 73-80).
- UB output_bank_0

  general output bank 0 (outputs 1-8).
- UB output_bank_1

  general output bank 1 (outputs 9-16).

- UB output_bank_2

  general output bank 2 (outputs 17-24).
- UB output_bank_3

  general output bank 3 (outputs 25-32).
- UB output_bank_4

  general output bank 4 (outputs 33-40).
- UB output_bank_5

  general output bank 5 (outputs 41-48).
- UB output_bank_6

  general output bank 6 (outputs 49-56).
- UB output_bank_7

  general output bank 7 (outputs 57-64).
- UB output_bank_8

  general output bank 8 (outputs 65-72).
- UB output_bank_9

  general output bank 9 (outputs 73-80).
- SW reserved_0

  Reserved.
- SW reserved_2

  Reserved.
- SW reserved_4

  Reserved.
- SW reserved_6

  Reserved.
- SW reserved_8

  Reserved.
- SW reserved_10

  Reserved.
- SW reserved_12

  Reserved.
- SW reserved_14

  Reserved.
- UB ethernet_status_a

  Ethernet Handle A Status.
- UB ethernet_status_b

  Ethernet Handle B Status.
- UB ethernet_status_c

  Ethernet Handle C Status.
- UB ethernet_status_d

  Ethernet Handle D Status.
- UB ethernet_status_e

  Ethernet Handle E Status.
- UB ethernet_status_f

  Ethernet Handle F Status.
- UB ethernet_status_g

  Ethernet Handle G Status.
- UB ethernet_status_h

  Ethernet Handle H Status.
- UB error_code

  error code.
- UB thread_status

thread status

- UL amplifier_status

  Amplifier Status.
- UL contour_segment_count

  Segment Count for Contour Mode.
- UW contour_buffer_available

  Buffer space remaining, Contour Mode.
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

  coordinated move status for S plane.
- SL s_distance

  distance traveled in coordinated move for S plane.
- UW s_plane_buffer_available

  Buffer space remaining, S Plane.
- UW t_plane_segment_count

  segment count of coordinated move for T plane.
- UW t_plane_move_status

  Coordinated move status for T plane.
- SL t_distance

  distance traveled in coordinated move for T plane.
- UW t_plane_buffer_available

  Buffer space remaining, T Plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

  A axis stop code.
- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

  A axis auxiliary position.
- SL axis_a_velocity

  A axis velocity.
- SL axis_a_torque

  A axis torque.
- UW axis_a_analog_in

  A axis analog input.
- UB axis_a_halls

  A Hall Input Status.
- UB axis_a_reserved

  Reserved.
- SL axis_a_variable

  A User-defined variable (ZA).
- UW axis_b_status

  B axis status.

- UB axis_b_switches

  B axis switches.
- UB axis_b_stop_code

  B axis stop code.
- SL axis_b_reference_position

  B axis reference position.
- SL axis_b_motor_position

  B axis motor position.
- SL axis_b_position_error

  B axis position error.
- SL axis_b_aux_position

  B axis auxiliary position.
- SL axis_b_velocity

  B axis velocity.
- SL axis_b_torque

  B axis torque.
- UW axis_b_analog_in

  B axis analog input.
- UB axis_b_halls

  B Hall Input Status.
- UB axis_b_reserved

  Reserved.
- SL axis_b_variable

  B User-defined variable (ZA).
- UW axis_c_status

  C axis status.
- UB axis_c_switches

  C axis switches.
- UB axis_c_stop_code

  C axis stop code.
- SL axis_c_reference_position

  C axis reference position.
- SL axis_c_motor_position

  C axis motor position.
- SL axis_c_position_error

  C axis position error.
- SL axis_c_aux_position

  C axis auxiliary position.
- SL axis_c_velocity

  C axis velocity.
- SL axis_c_torque

  C axis torque.
- UW axis_c_analog_in

  C axis analog input.
- UB axis_c_halls

  C Hall Input Status.
- UB axis_c_reserved

  Reserved.
- SL axis_c_variable

  C User-defined variable (ZA).
- UW axis_d_status

D axis status.

- UB axis_d_switches

  D axis switches.
- UB axis_d_stop_code

  D axis stop code.
- SL axis_d_reference_position

  D axis reference position.
- SL axis_d_motor_position

  D axis motor position.
- SL axis_d_position_error

  D axis position error.
- SL axis_d_aux_position

  D axis auxiliary position.
- SL axis_d_velocity

  D axis velocity.
- SL axis_d_torque

  D axis torque.
- UW axis_d_analog_in

  D axis analog input.
- UB axis_d_halls

  D Hall Input Status.
- UB axis_d_reserved

  Reserved.
- SL axis_d_variable

  D User-defined variable (ZA).
- UW axis_e_status

  E axis status.
- UB axis_e_switches

  E axis switches.
- UB axis_e_stop_code

  E axis stop code.
- SL axis_e_reference_position

  E axis reference position.
- SL axis_e_motor_position

  E axis motor position.
- SL axis_e_position_error

  E axis position error.
- SL axis_e_aux_position

  E axis auxiliary position.
- SL axis_e_velocity

  E axis velocity.
- SL axis_e_torque

  E axis torque.
- UW axis_e_analog_in

  E axis analog input.
- UB axis_e_halls

  E Hall Input Status.
- UB axis_e_reserved

  Reserved.
- SL axis_e_variable

  E User-defined variable (ZA).

- UW axis_f_status

  F axis status.
- UB axis_f_switches

  F axis switches.
- UB axis_f_stop_code

  F axis stop code.
- SL axis_f_reference_position

  F axis reference position.
- SL axis_f_motor_position

  F axis motor position.
- SL axis_f_position_error

  F axis position error.
- SL axis_f_aux_position

  F axis auxiliary position.
- SL axis_f_velocity

  F axis velocity.
- SL axis_f_torque

  F axis torque.
- UW axis_f_analog_in

  F axis analog input.
- UB axis_f_halls

  F Hall Input Status.
- UB axis_f_reserved

  Reserved.
- SL axis_f_variable

  F User-defined variable (ZA).
- UW axis_g_status

  G axis status.
- UB axis_g_switches

  G axis switches.
- UB axis_g_stop_code

  G axis stop code.
- SL axis_g_reference_position

  G axis reference position.
- SL axis_g_motor_position

  G axis motor position.
- SL axis_g_position_error

  G axis position error.
- SL axis_g_aux_position

  G axis auxiliary position.
- SL axis_g_velocity

  G axis velocity.
- SL axis_g_torque

  G axis torque.
- UW axis_g_analog_in

  G axis analog input.
- UB axis_g_halls

  G Hall Input Status.
- UB axis_g_reserved

  Reserved.
- SL axis_g_variable

G User-defined variable (ZA).

- UW axis_h_status

  H axis status.
- UB axis_h_switches

  H axis switches.
- UB axis_h_stop_code

  H axis stop code.
- SL axis_h_reference_position

  H axis reference position.
- SL axis_h_motor_position

  H axis motor position.
- SL axis_h_position_error

  H axis position error.
- SL axis_h_aux_position

  H axis auxiliary position.
- SL axis_h_velocity

  H axis velocity.
- SL axis_h_torque

  H axis torque.
- UW axis_h_analog_in

  H axis analog input.
- UB axis_h_halls

  H Hall Input Status.
- UB axis_h_reserved

  Reserved.
- SL axis_h_variable

  H User-defined variable (ZA).

9.27.1 Detailed Description

Data record struct for DMC-4000 controllers, including 4000, 4200, 4103, and 500x0.
Definition at line 918 of file gclib.cs.

9.27.2 Member Function Documentation

9.27.2.1 byte_array()

```
byte[] byte_array ( )  [inline]
```

Returns the data record as a byte array and allows for access to individual bytes.
Implements GDataRecord.
Definition at line 920 of file gclib.cs.

9.27.3 Field Documentation

9.27.3.1 amplifier_status

```
UL amplifier_status
```

Amplifier Status.
Definition at line 972 of file gclib.cs.

9.27.3.2 axis_a_analog_in

```
UW axis_a_analog_in
```

A axis analog input.
Definition at line 996 of file gclib.cs.

9.27.3.3 axis_a_aux_position

`SL axis_a_aux_position`
A axis auxiliary position.
Definition at line 993 of file gclib.cs.

9.27.3.4 axis_a_halls

`UB axis_a_halls`
A Hall Input Status.
Definition at line 997 of file gclib.cs.

9.27.3.5 axis_a_motor_position

`SL axis_a_motor_position`
A axis motor position.
Definition at line 991 of file gclib.cs.

9.27.3.6 axis_a_position_error

`SL axis_a_position_error`
A axis position error.
Definition at line 992 of file gclib.cs.

9.27.3.7 axis_a_reference_position

`SL axis_a_reference_position`
A axis reference position.
Definition at line 990 of file gclib.cs.

9.27.3.8 axis_a_reserved

`UB axis_a_reserved`
Reserved.
Definition at line 998 of file gclib.cs.

9.27.3.9 axis_a_status

`UW axis_a_status`
A axis status.
Definition at line 987 of file gclib.cs.

9.27.3.10 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 989 of file gclib.cs.

9.27.3.11 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 988 of file gclib.cs.

9.27.3.12 axis_a_torque

`SL axis_a_torque`
A axis torque.
Definition at line 995 of file gclib.cs.

9.27.3.13 axis_a_variable

`SL axis_a_variable`
A User-defined variable (ZA).
Definition at line 999 of file gclib.cs.

9.27.3.14 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 994 of file gclib.cs.

9.27.3.15 axis_b_analog_in

`UW axis_b_analog_in`
B axis analog input.
Definition at line 1010 of file gclib.cs.

9.27.3.16 axis_b_aux_position

`SL axis_b_aux_position`
B axis auxiliary position.
Definition at line 1007 of file gclib.cs.

9.27.3.17 axis_b_halls

`UB axis_b_halls`
B Hall Input Status.
Definition at line 1011 of file gclib.cs.

9.27.3.18 axis_b_motor_position

`SL axis_b_motor_position`
B axis motor position.
Definition at line 1005 of file gclib.cs.

9.27.3.19 axis_b_position_error

`SL axis_b_position_error`
B axis position error.
Definition at line 1006 of file gclib.cs.

9.27.3.20 axis_b_reference_position

`SL axis_b_reference_position`
B axis reference position.
Definition at line 1004 of file gclib.cs.

9.27.3.21 axis_b_reserved

`UB axis_b_reserved`
Reserved.
Definition at line 1012 of file gclib.cs.

9.27.3.22 axis_b_status

`UW axis_b_status`
B axis status.
Definition at line 1001 of file gclib.cs.

9.27.3.23 axis_b_stop_code

`UB axis_b_stop_code`
B axis stop code.
Definition at line 1003 of file gclib.cs.

9.27.3.24 axis_b_switches

`UB axis_b_switches`
B axis switches.
Definition at line 1002 of file gclib.cs.

9.27.3.25 axis_b_torque

`SL axis_b_torque`
B axis torque.
Definition at line 1009 of file gclib.cs.

9.27.3.26 axis_b_variable

`SL axis_b_variable`
B User-defined variable (ZA).
Definition at line 1013 of file gclib.cs.

9.27.3.27 axis_b_velocity

`SL axis_b_velocity`
B axis velocity.
Definition at line 1008 of file gclib.cs.

9.27.3.28 axis_c_analog_in

`UW axis_c_analog_in`
C axis analog input.
Definition at line 1024 of file gclib.cs.

9.27.3.29 axis_c_aux_position

`SL axis_c_aux_position`
C axis auxiliary position.
Definition at line 1021 of file gclib.cs.

9.27.3.30 axis_c_halls

`UB axis_c_halls`
C Hall Input Status.
Definition at line 1025 of file gclib.cs.

9.27.3.31 axis_c_motor_position

`SL axis_c_motor_position`
C axis motor position.
Definition at line 1019 of file gclib.cs.

9.27.3.32 axis_c_position_error

`SL axis_c_position_error`
C axis position error.
Definition at line 1020 of file gclib.cs.

9.27.3.33 axis_c_reference_position

`SL axis_c_reference_position`
C axis reference position.
Definition at line 1018 of file gclib.cs.

9.27.3.34 axis_c_reserved

`UB axis_c_reserved`
Reserved.
Definition at line 1026 of file gclib.cs.

9.27.3.35 axis_c_status

`UW axis_c_status`
C axis status.
Definition at line 1015 of file gclib.cs.

9.27.3.36 axis_c_stop_code

`UB axis_c_stop_code`
C axis stop code.
Definition at line 1017 of file gclib.cs.

9.27.3.37 axis_c_switches

`UB axis_c_switches`
C axis switches.
Definition at line 1016 of file gclib.cs.

9.27.3.38 axis_c_torque

`SL axis_c_torque`
C axis torque.
Definition at line 1023 of file gclib.cs.

9.27.3.39 axis_c_variable

`SL axis_c_variable`
C User-defined variable (ZA).
Definition at line 1027 of file gclib.cs.

9.27.3.40 axis_c_velocity

`SL axis_c_velocity`
C axis velocity.
Definition at line 1022 of file gclib.cs.

9.27.3.41 axis_d_analog_in

`UW axis_d_analog_in`
D axis analog input.
Definition at line 1038 of file gclib.cs.

9.27.3.42 axis_d_aux_position

`SL axis_d_aux_position`
D axis auxiliary position.
Definition at line 1035 of file gclib.cs.

9.27.3.43 axis_d_halls

`UB axis_d_halls`
D Hall Input Status.
Definition at line 1039 of file gclib.cs.

9.27.3.44 axis_d_motor_position

`SL axis_d_motor_position`
D axis motor position.
Definition at line 1033 of file gclib.cs.

9.27.3.45 axis_d_position_error

`SL axis_d_position_error`
D axis position error.
Definition at line 1034 of file gclib.cs.

9.27.3.46 axis_d_reference_position

`SL axis_d_reference_position`
D axis reference position.
Definition at line 1032 of file gclib.cs.

9.27.3.47 axis_d_reserved

`UB axis_d_reserved`
Reserved.
Definition at line 1040 of file gclib.cs.

9.27.3.48 axis_d_status

`UW axis_d_status`
D axis status.
Definition at line 1029 of file gclib.cs.

9.27.3.49 axis_d_stop_code

`UB axis_d_stop_code`
D axis stop code.
Definition at line 1031 of file gclib.cs.

9.27.3.50 axis_d_switches

`UB axis_d_switches`
D axis switches.
Definition at line 1030 of file gclib.cs.

9.27.3.51 axis_d_torque

`SL axis_d_torque`
D axis torque.
Definition at line 1037 of file gclib.cs.

9.27.3.52 axis_d_variable

`SL axis_d_variable`
D User-defined variable (ZA).
Definition at line 1041 of file gclib.cs.

9.27.3.53 axis_d_velocity

`SL axis_d_velocity`
D axis velocity.
Definition at line 1036 of file gclib.cs.

9.27.3.54 axis_e_analog_in

`UW axis_e_analog_in`
E axis analog input.
Definition at line 1052 of file gclib.cs.

9.27.3.55 axis_e_aux_position

`SL axis_e_aux_position`
E axis auxiliary position.
Definition at line 1049 of file gclib.cs.

9.27.3.56 axis_e_halls

`UB axis_e_halls`
E Hall Input Status.
Definition at line 1053 of file gclib.cs.

9.27.3.57 axis_e_motor_position

`SL axis_e_motor_position`
E axis motor position.
Definition at line 1047 of file gclib.cs.

9.27.3.58 axis_e_position_error

`SL axis_e_position_error`
E axis position error.
Definition at line 1048 of file gclib.cs.

9.27.3.59 axis_e_reference_position

`SL axis_e_reference_position`
E axis reference position.
Definition at line 1046 of file gclib.cs.

9.27.3.60 axis_e_reserved

`UB axis_e_reserved`
Reserved.
Definition at line 1054 of file gclib.cs.

9.27.3.61 axis_e_status

`UW axis_e_status`
E axis status.
Definition at line 1043 of file gclib.cs.

9.27.3.62 axis_e_stop_code

`UB axis_e_stop_code`
E axis stop code.
Definition at line 1045 of file gclib.cs.

9.27.3.63 axis_e_switches

`UB axis_e_switches`
E axis switches.
Definition at line 1044 of file gclib.cs.

9.27.3.64 axis_e_torque

`SL axis_e_torque`
E axis torque.
Definition at line 1051 of file gclib.cs.

9.27.3.65 axis_e_variable

`SL axis_e_variable`
E User-defined variable (ZA).
Definition at line 1055 of file gclib.cs.

9.27.3.66 axis_e_velocity

`SL axis_e_velocity`
E axis velocity.
Definition at line 1050 of file gclib.cs.

9.27.3.67 axis_f_analog_in

`UW axis_f_analog_in`
F axis analog input.
Definition at line 1066 of file gclib.cs.

9.27.3.68 axis_f_aux_position

`SL axis_f_aux_position`
F axis auxiliary position.
Definition at line 1063 of file gclib.cs.

9.27.3.69 axis_f_halls

`UB axis_f_halls`
F Hall Input Status.
Definition at line 1067 of file gclib.cs.

9.27.3.70 axis_f_motor_position

`SL axis_f_motor_position`
F axis motor position.
Definition at line 1061 of file gclib.cs.

9.27.3.71 axis_f_position_error

`SL axis_f_position_error`
F axis position error.
Definition at line 1062 of file gclib.cs.

9.27.3.72 axis_f_reference_position

`SL axis_f_reference_position`
F axis reference position.
Definition at line 1060 of file gclib.cs.

9.27.3.73 axis_f_reserved

`UB axis_f_reserved`
Reserved.
Definition at line 1068 of file gclib.cs.

9.27.3.74 axis_f_status

`UW axis_f_status`
F axis status.
Definition at line 1057 of file gclib.cs.

9.27.3.75 axis_f_stop_code

`UB axis_f_stop_code`
F axis stop code.
Definition at line 1059 of file gclib.cs.

9.27.3.76 axis_f_switches

`UB axis_f_switches`
F axis switches.
Definition at line 1058 of file gclib.cs.

9.27.3.77 axis_f_torque

`SL axis_f_torque`
F axis torque.
Definition at line 1065 of file gclib.cs.

9.27.3.78 axis_f_variable

`SL axis_f_variable`
F User-defined variable (ZA).
Definition at line 1069 of file gclib.cs.

9.27.3.79 axis_f_velocity

`SL axis_f_velocity`
F axis velocity.
Definition at line 1064 of file gclib.cs.

9.27.3.80 axis_g_analog_in

`UW axis_g_analog_in`
G axis analog input.
Definition at line 1080 of file gclib.cs.

9.27.3.81 axis_g_aux_position

`SL axis_g_aux_position`
G axis auxiliary position.
Definition at line 1077 of file gclib.cs.

9.27.3.82 axis_g_halls

`UB axis_g_halls`
G Hall Input Status.
Definition at line 1081 of file gclib.cs.

9.27.3.83 axis_g_motor_position

`SL axis_g_motor_position`
G axis motor position.
Definition at line 1075 of file gclib.cs.

9.27.3.84 axis_g_position_error

`SL axis_g_position_error`
G axis position error.
Definition at line 1076 of file gclib.cs.

9.27.3.85 axis_g_reference_position

`SL axis_g_reference_position`
G axis reference position.
Definition at line 1074 of file gclib.cs.

9.27.3.86 axis_g_reserved

`UB axis_g_reserved`
Reserved.
Definition at line 1082 of file gclib.cs.

9.27.3.87 axis_g_status

`UW axis_g_status`
G axis status.
Definition at line 1071 of file gclib.cs.

9.27.3.88 axis_g_stop_code

`UB axis_g_stop_code`
G axis stop code.
Definition at line 1073 of file gclib.cs.

9.27.3.89 axis_g_switches

`UB axis_g_switches`
G axis switches.
Definition at line 1072 of file gclib.cs.

9.27.3.90 axis_g_torque

`SL axis_g_torque`
G axis torque.
Definition at line 1079 of file gclib.cs.

9.27.3.91 axis_g_variable

`SL axis_g_variable`
G User-defined variable (ZA).
Definition at line 1083 of file gclib.cs.

9.27.3.92 axis_g_velocity

`SL axis_g_velocity`
G axis velocity.
Definition at line 1078 of file gclib.cs.

9.27.3.93 axis_h_analog_in

`UW axis_h_analog_in`
H axis analog input.
Definition at line 1094 of file gclib.cs.

9.27.3.94 axis_h_aux_position

`SL axis_h_aux_position`
H axis auxiliary position.
Definition at line 1091 of file gclib.cs.

9.27.3.95 axis_h_halls

`UB axis_h_halls`
H Hall Input Status.
Definition at line 1095 of file gclib.cs.

9.27.3.96 axis_h_motor_position

`SL axis_h_motor_position`
H axis motor position.
Definition at line 1089 of file gclib.cs.

9.27.3.97 axis_h_position_error

`SL axis_h_position_error`
H axis position error.
Definition at line 1090 of file gclib.cs.

9.27.3.98 axis_h_reference_position

`SL axis_h_reference_position`
H axis reference position.
Definition at line 1088 of file gclib.cs.

9.27.3.99 axis_h_reserved

`UB axis_h_reserved`
Reserved.
Definition at line 1096 of file gclib.cs.

9.27.3.100 axis_h_status

`UW axis_h_status`
H axis status.
Definition at line 1085 of file gclib.cs.

9.27.3.101 axis_h_stop_code

`UB axis_h_stop_code`
H axis stop code.
Definition at line 1087 of file gclib.cs.

9.27.3.102 axis_h_switches

`UB axis_h_switches`
H axis switches.
Definition at line 1086 of file gclib.cs.

9.27.3.103 axis_h_torque

`SL axis_h_torque`
H axis torque.
Definition at line 1093 of file gclib.cs.

9.27.3.104 axis_h_variable

`SL axis_h_variable`
H User-defined variable (ZA).
Definition at line 1097 of file gclib.cs.

9.27.3.105 axis_h_velocity

`SL axis_h_velocity`
H axis velocity.
Definition at line 1092 of file gclib.cs.

9.27.3.106 contour_buffer_available

`UW contour_buffer_available`
Buffer space remaining, Contour Mode.
Definition at line 975 of file gclib.cs.

9.27.3.107 contour_segment_count

`UL contour_segment_count`
Segment Count for Contour Mode.
Definition at line 974 of file gclib.cs.

9.27.3.108 error_code

`UB error_code`
error code.
Definition at line 970 of file gclib.cs.

9.27.3.109 ethernet_status_a

`UB ethernet_status_a`
Ethernet Handle A Status.
Definition at line 961 of file gclib.cs.

9.27.3.110 ethernet_status_b

`UB ethernet_status_b`
Ethernet Handle B Status.
Definition at line 962 of file gclib.cs.

9.27.3.111 ethernet_status_c

`UB ethernet_status_c`
Ethernet Handle C Status.
Definition at line 963 of file gclib.cs.

9.27.3.112 ethernet_status_d

`UB ethernet_status_d`
Ethernet Handle D Status.
Definition at line 964 of file gclib.cs.

9.27.3.113 ethernet_status_e

`UB ethernet_status_e`
Ethernet Handle E Status.
Definition at line 965 of file gclib.cs.

9.27.3.114 ethernet_status_f

`UB ethernet_status_f`
Ethernet Handle F Status.
Definition at line 966 of file gclib.cs.

9.27.3.115 ethernet_status_g

`UB ethernet_status_g`
Ethernet Handle G Status.
Definition at line 967 of file gclib.cs.

9.27.3.116 ethernet_status_h

`UB ethernet_status_h`
Ethernet Handle H Status.
Definition at line 968 of file gclib.cs.

9.27.3.117 header_0

`UB header_0`
1st Byte of Header.
Definition at line 923 of file gclib.cs.

9.27.3.118 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 924 of file gclib.cs.

9.27.3.119 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 925 of file gclib.cs.

9.27.3.120 header_3

`UB header_3`
4th Byte of Header.
Definition at line 926 of file gclib.cs.

9.27.3.121 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 930 of file gclib.cs.

9.27.3.122 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 931 of file gclib.cs.

9.27.3.123 input_bank_2

`UB input_bank_2`
general input bank 2 (inputs 17-24).
Definition at line 932 of file gclib.cs.

9.27.3.124 input_bank_3

`UB input_bank_3`
general input bank 3 (inputs 25-32).
Definition at line 933 of file gclib.cs.

9.27.3.125 input_bank_4

`UB input_bank_4`
general input bank 4 (inputs 33-40).
Definition at line 934 of file gclib.cs.

9.27.3.126 input_bank_5

`UB input_bank_5`
general input bank 5 (inputs 41-48).
Definition at line 935 of file gclib.cs.

9.27.3.127 input_bank_6

`UB input_bank_6`
general input bank 6 (inputs 49-56).
Definition at line 936 of file gclib.cs.

9.27.3.128 input_bank_7

`UB input_bank_7`
general input bank 7 (inputs 57-64).
Definition at line 937 of file gclib.cs.

9.27.3.129 input_bank_8

`UB input_bank_8`
general input bank 8 (inputs 65-72).
Definition at line 938 of file gclib.cs.

9.27.3.130 input_bank_9

`UB input_bank_9`
general input bank 9 (inputs 73-80).
Definition at line 939 of file gclib.cs.

9.27.3.131 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 941 of file gclib.cs.

9.27.3.132 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 942 of file gclib.cs.

9.27.3.133 output_bank_2

`UB output_bank_2`
general output bank 2 (outputs 17-24).
Definition at line 943 of file gclib.cs.

9.27.3.134 output_bank_3

`UB output_bank_3`
general output bank 3 (outputs 25-32).
Definition at line 944 of file gclib.cs.

9.27.3.135 output_bank_4

`UB output_bank_4`
general output bank 4 (outputs 33-40).
Definition at line 945 of file gclib.cs.

9.27.3.136 output_bank_5

`UB output_bank_5`
general output bank 5 (outputs 41-48).
Definition at line 946 of file gclib.cs.

9.27.3.137 output_bank_6

`UB output_bank_6`
general output bank 6 (outputs 49-56).
Definition at line 947 of file gclib.cs.

9.27.3.138 output_bank_7

`UB output_bank_7`
general output bank 7 (outputs 57-64).
Definition at line 948 of file gclib.cs.

9.27.3.139 output_bank_8

`UB output_bank_8`
general output bank 8 (outputs 65-72).
Definition at line 949 of file gclib.cs.

9.27.3.140 output_bank_9

`UB output_bank_9`
general output bank 9 (outputs 73-80).
Definition at line 950 of file gclib.cs.

9.27.3.141 reserved_0

`SW reserved_0`
Reserved.
Definition at line 952 of file gclib.cs.

9.27.3.142 reserved_10

`SW reserved_10`
Reserved.
Definition at line 957 of file gclib.cs.

9.27.3.143 reserved_12

`SW reserved_12`
Reserved.
Definition at line 958 of file gclib.cs.

9.27.3.144 reserved_14

`SW reserved_14`
Reserved.
Definition at line 959 of file gclib.cs.

9.27.3.145 reserved_2

`SW reserved_2`
Reserved.
Definition at line 953 of file gclib.cs.

9.27.3.146 reserved_4

`SW reserved_4`
Reserved.
Definition at line 954 of file gclib.cs.

9.27.3.147 reserved_6

`SW reserved_6`
Reserved.
Definition at line 955 of file gclib.cs.

9.27.3.148 reserved_8

`SW reserved_8`
Reserved.
Definition at line 956 of file gclib.cs.

9.27.3.149 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 979 of file gclib.cs.

9.27.3.150 s_plane_buffer_available

`UW s_plane_buffer_available`
Buffer space remaining, S Plane.
Definition at line 980 of file gclib.cs.

9.27.3.151 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 978 of file gclib.cs.

9.27.3.152 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 977 of file gclib.cs.

9.27.3.153 sample_number

`UW sample_number`
sample number.
Definition at line 928 of file gclib.cs.

9.27.3.154 t_distance

`SL t_distance`
distance traveled in coordinated move for T plane.
Definition at line 984 of file gclib.cs.

9.27.3.155 t_plane_buffer_available

`UW t_plane_buffer_available`
Buffer space remaining, T Plane.
Definition at line 985 of file gclib.cs.

9.27.3.156 t_plane_move_status

`UW t_plane_move_status`
Coordinated move status for T plane.
Definition at line 983 of file gclib.cs.

9.27.3.157 t_plane_segment_count

`UW t_plane_segment_count`
segment count of coordinated move for T plane.
Definition at line 982 of file gclib.cs.

9.27.3.158 thread_status

`UB thread_status`
thread status
Definition at line 971 of file gclib.cs.
The documentation for this struct was generated from the following file:

- gclib.cs

9.28 GDataRecord4000 Struct Reference

Data record struct for DMC-4000 controllers, including 4000, 4200, 4103, and 500x0.
`#include <gclib_record.h>`

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.
- UB header_3

  4th Byte of Header.
- UW sample_number

  sample number.
- UB input_bank_0

general input bank 0 (inputs 1-8).

- UB input_bank_1

  general input bank 1 (inputs 9-16).
- UB input_bank_2

  general input bank 2 (inputs 17-24).
- UB input_bank_3

  general input bank 3 (inputs 25-32).
- UB input_bank_4

  general input bank 4 (inputs 33-40).
- UB input_bank_5

  general input bank 5 (inputs 41-48).
- UB input_bank_6

  general input bank 6 (inputs 49-56).
- UB input_bank_7

  general input bank 7 (inputs 57-64).
- UB input_bank_8

  general input bank 8 (inputs 65-72).
- UB input_bank_9

  general input bank 9 (inputs 73-80).
- UB output_bank_0

  general output bank 0 (outputs 1-8).
- UB output_bank_1

  general output bank 1 (outputs 9-16).
- UB output_bank_2

  general output bank 2 (outputs 17-24).
- UB output_bank_3

  general output bank 3 (outputs 25-32).
- UB output_bank_4

  general output bank 4 (outputs 33-40).
- UB output_bank_5

  general output bank 5 (outputs 41-48).
- UB output_bank_6

  general output bank 6 (outputs 49-56).
- UB output_bank_7

  general output bank 7 (outputs 57-64).
- UB output_bank_8

  general output bank 8 (outputs 65-72).
- UB output_bank_9

  general output bank 9 (outputs 73-80).
- SW reserved_0

  Reserved.
- SW reserved_2

  Reserved.
- SW reserved_4

  Reserved.
- SW reserved_6

  Reserved.
- SW reserved_8

  Reserved.
- SW reserved_10

  Reserved.

- SW reserved_12

  Reserved.
- SW reserved_14

  Reserved.
- UB ethernet_status_a

  Ethernet Handle A Status.
- UB ethernet_status_b

  Ethernet Handle B Status.
- UB ethernet_status_c

  Ethernet Handle C Status.
- UB ethernet_status_d

  Ethernet Handle D Status.
- UB ethernet_status_e

  Ethernet Handle E Status.
- UB ethernet_status_f

  Ethernet Handle F Status.
- UB ethernet_status_g

  Ethernet Handle G Status.
- UB ethernet_status_h

  Ethernet Handle H Status.
- UB error_code

  error code.
- UB thread_status

  thread status
- UL amplifier_status

  Amplifier Status.
- UL contour_segment_count

  Segment Count for Contour Mode.
- UW contour_buffer_available

  Buffer space remaining, Contour Mode.
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

  coordinated move status for S plane.
- SL s_distance

  distance traveled in coordinated move for S plane.
- UW s_plane_buffer_available

  Buffer space remaining, S Plane.
- UW t_plane_segment_count

  segment count of coordinated move for T plane.
- UW t_plane_move_status

  Coordinated move status for T plane.
- SL t_distance

  distance traveled in coordinated move for T plane.
- UW t_plane_buffer_available

  Buffer space remaining, T Plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

A axis stop code.

- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

  A axis auxiliary position.
- SL axis_a_velocity

  A axis velocity.
- SL axis_a_torque

  A axis torque.
- UW axis_a_analog_in

  A axis analog input.
- UB axis_a_halls

  A Hall Input Status.
- UB axis_a_reserved

  Reserved.
- SL axis_a_variable

  A User-defined variable (ZA).
- UW axis_b_status

  B axis status.
- UB axis_b_switches

  B axis switches.
- UB axis_b_stop_code

  B axis stop code.
- SL axis_b_reference_position

  B axis reference position.
- SL axis_b_motor_position

  B axis motor position.
- SL axis_b_position_error

  B axis position error.
- SL axis_b_aux_position

  B axis auxiliary position.
- SL axis_b_velocity

  B axis velocity.
- SL axis_b_torque

  B axis torque.
- UW axis_b_analog_in

  B axis analog input.
- UB axis_b_halls

  B Hall Input Status.
- UB axis_b_reserved

  Reserved.
- SL axis_b_variable

  B User-defined variable (ZA).
- UW axis_c_status

  C axis status.
- UB axis_c_switches

  C axis switches.

- UB axis_c_stop_code

  C axis stop code.
- SL axis_c_reference_position

  C axis reference position.
- SL axis_c_motor_position

  C axis motor position.
- SL axis_c_position_error

  C axis position error.
- SL axis_c_aux_position

  C axis auxiliary position.
- SL axis_c_velocity

  C axis velocity.
- SL axis_c_torque

  C axis torque.
- UW axis_c_analog_in

  C axis analog input.
- UB axis_c_halls

  C Hall Input Status.
- UB axis_c_reserved

  Reserved.
- SL axis_c_variable

  C User-defined variable (ZA).
- UW axis_d_status

  D axis status.
- UB axis_d_switches

  D axis switches.
- UB axis_d_stop_code

  D axis stop code.
- SL axis_d_reference_position

  D axis reference position.
- SL axis_d_motor_position

  D axis motor position.
- SL axis_d_position_error

  D axis position error.
- SL axis_d_aux_position

  D axis auxiliary position.
- SL axis_d_velocity

  D axis velocity.
- SL axis_d_torque

  D axis torque.
- UW axis_d_analog_in

  D axis analog input.
- UB axis_d_halls

  D Hall Input Status.
- UB axis_d_reserved

  Reserved.
- SL axis_d_variable

  D User-defined variable (ZA).
- UW axis_e_status

  E axis status.
- UB axis_e_switches

E axis switches.

- UB axis_e_stop_code

  E axis stop code.
- SL axis_e_reference_position

  E axis reference position.
- SL axis_e_motor_position

  E axis motor position.
- SL axis_e_position_error

  E axis position error.
- SL axis_e_aux_position

  E axis auxiliary position.
- SL axis_e_velocity

  E axis velocity.
- SL axis_e_torque

  E axis torque.
- UW axis_e_analog_in

  E axis analog input.
- UB axis_e_halls

  E Hall Input Status.
- UB axis_e_reserved

  Reserved.
- SL axis_e_variable

  E User-defined variable (ZA).
- UW axis_f_status

  F axis status.
- UB axis_f_switches

  F axis switches.
- UB axis_f_stop_code

  F axis stop code.
- SL axis_f_reference_position

  F axis reference position.
- SL axis_f_motor_position

  F axis motor position.
- SL axis_f_position_error

  F axis position error.
- SL axis_f_aux_position

  F axis auxiliary position.
- SL axis_f_velocity

  F axis velocity.
- SL axis_f_torque

  F axis torque.
- UW axis_f_analog_in

  F axis analog input.
- UB axis_f_halls

  F Hall Input Status.
- UB axis_f_reserved

  Reserved.
- SL axis_f_variable

  F User-defined variable (ZA).
- UW axis_g_status

  G axis status.

- UB axis_g_switches

  G axis switches.
- UB axis_g_stop_code

  G axis stop code.
- SL axis_g_reference_position

  G axis reference position.
- SL axis_g_motor_position

  G axis motor position.
- SL axis_g_position_error

  G axis position error.
- SL axis_g_aux_position

  G axis auxiliary position.
- SL axis_g_velocity

  G axis velocity.
- SL axis_g_torque

  G axis torque.
- UW axis_g_analog_in

  G axis analog input.
- UB axis_g_halls

  G Hall Input Status.
- UB axis_g_reserved

  Reserved.
- SL axis_g_variable

  G User-defined variable (ZA).
- UW axis_h_status

  H axis status.
- UB axis_h_switches

  H axis switches.
- UB axis_h_stop_code

  H axis stop code.
- SL axis_h_reference_position

  H axis reference position.
- SL axis_h_motor_position

  H axis motor position.
- SL axis_h_position_error

  H axis position error.
- SL axis_h_aux_position

  H axis auxiliary position.
- SL axis_h_velocity

  H axis velocity.
- SL axis_h_torque

  H axis torque.
- UW axis_h_analog_in

  H axis analog input.
- UB axis_h_halls

  H Hall Input Status.
- UB axis_h_reserved

  Reserved.
- SL axis_h_variable

  H User-defined variable (ZA).

9.28.1 Detailed Description

Data record struct for DMC-4000 controllers, including 4000, 4200, 4103, and 500x0.
Definition at line 35 of file gclib_record.h.

9.28.2 Field Documentation

9.28.2.1 amplifier_status

`UL amplifier_status`
Amplifier Status.
Definition at line 89 of file gclib_record.h.

9.28.2.2 axis_a_analog_in

`UW axis_a_analog_in`
A axis analog input.
Definition at line 113 of file gclib_record.h.

9.28.2.3 axis_a_aux_position

`SL axis_a_aux_position`
A axis auxiliary position.
Definition at line 110 of file gclib_record.h.

9.28.2.4 axis_a_halls

`UB axis_a_halls`
A Hall Input Status.
Definition at line 114 of file gclib_record.h.

9.28.2.5 axis_a_motor_position

`SL axis_a_motor_position`
A axis motor position.
Definition at line 108 of file gclib_record.h.

9.28.2.6 axis_a_position_error

`SL axis_a_position_error`
A axis position error.
Definition at line 109 of file gclib_record.h.

9.28.2.7 axis_a_reference_position

`SL axis_a_reference_position`
A axis reference position.
Definition at line 107 of file gclib_record.h.

9.28.2.8 axis_a_reserved

`UB axis_a_reserved`
Reserved.
Definition at line 115 of file gclib_record.h.

9.28.2.9 axis_a_status

`UW axis_a_status`
A axis status.
Definition at line 104 of file gclib_record.h.

9.28.2.10 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 106 of file gclib_record.h.

9.28.2.11 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 105 of file gclib_record.h.

9.28.2.12 axis_a_torque

`SL axis_a_torque`
A axis torque.
Definition at line 112 of file gclib_record.h.

9.28.2.13 axis_a_variable

`SL axis_a_variable`
A User-defined variable (ZA).
Definition at line 116 of file gclib_record.h.

9.28.2.14 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 111 of file gclib_record.h.

9.28.2.15 axis_b_analog_in

`UW axis_b_analog_in`
B axis analog input.
Definition at line 127 of file gclib_record.h.

9.28.2.16 axis_b_aux_position

`SL axis_b_aux_position`
B axis auxiliary position.
Definition at line 124 of file gclib_record.h.

9.28.2.17 axis_b_halls

`UB axis_b_halls`
B Hall Input Status.
Definition at line 128 of file gclib_record.h.

9.28.2.18 axis_b_motor_position

`SL axis_b_motor_position`
B axis motor position.
Definition at line 122 of file gclib_record.h.

9.28.2.19 axis_b_position_error

`SL axis_b_position_error`
B axis position error.
Definition at line 123 of file gclib_record.h.

9.28.2.20 axis_b_reference_position

`SL axis_b_reference_position`
B axis reference position.
Definition at line 121 of file gclib_record.h.

9.28.2.21 axis_b_reserved

`UB axis_b_reserved`
Reserved.
Definition at line 129 of file gclib_record.h.

9.28.2.22 axis_b_status

`UW axis_b_status`
B axis status.
Definition at line 118 of file gclib_record.h.

9.28.2.23 axis_b_stop_code

`UB axis_b_stop_code`
B axis stop code.
Definition at line 120 of file gclib_record.h.

9.28.2.24 axis_b_switches

`UB axis_b_switches`
B axis switches.
Definition at line 119 of file gclib_record.h.

9.28.2.25 axis_b_torque

`SL axis_b_torque`
B axis torque.
Definition at line 126 of file gclib_record.h.

9.28.2.26 axis_b_variable

`SL axis_b_variable`
B User-defined variable (ZA).
Definition at line 130 of file gclib_record.h.

9.28.2.27 axis_b_velocity

`SL axis_b_velocity`
B axis velocity.
Definition at line 125 of file gclib_record.h.

9.28.2.28 axis_c_analog_in

`UW axis_c_analog_in`
C axis analog input.
Definition at line 141 of file gclib_record.h.

9.28.2.29 axis_c_aux_position

`SL axis_c_aux_position`
C axis auxiliary position.
Definition at line 138 of file gclib_record.h.

9.28.2.30 axis_c_halls

`UB axis_c_halls`
C Hall Input Status.
Definition at line 142 of file gclib_record.h.

9.28.2.31 axis_c_motor_position

`SL axis_c_motor_position`
C axis motor position.
Definition at line 136 of file gclib_record.h.

9.28.2.32 axis_c_position_error

`SL axis_c_position_error`
C axis position error.
Definition at line 137 of file gclib_record.h.

9.28.2.33 axis_c_reference_position

`SL axis_c_reference_position`
C axis reference position.
Definition at line 135 of file gclib_record.h.

9.28.2.34 axis_c_reserved

`UB axis_c_reserved`
Reserved.
Definition at line 143 of file gclib_record.h.

9.28.2.35 axis_c_status

`UW axis_c_status`
C axis status.
Definition at line 132 of file gclib_record.h.

9.28.2.36 axis_c_stop_code

`UB axis_c_stop_code`
C axis stop code.
Definition at line 134 of file gclib_record.h.

9.28.2.37 axis_c_switches

`UB axis_c_switches`
C axis switches.
Definition at line 133 of file gclib_record.h.

9.28.2.38 axis_c_torque

`SL axis_c_torque`
C axis torque.
Definition at line 140 of file gclib_record.h.

9.28.2.39 axis_c_variable

`SL axis_c_variable`
C User-defined variable (ZA).
Definition at line 144 of file gclib_record.h.

9.28.2.40 axis_c_velocity

`SL axis_c_velocity`
C axis velocity.
Definition at line 139 of file gclib_record.h.

9.28.2.41 axis_d_analog_in

`UW axis_d_analog_in`
D axis analog input.
Definition at line 155 of file gclib_record.h.

9.28.2.42 axis_d_aux_position

`SL axis_d_aux_position`
D axis auxiliary position.
Definition at line 152 of file gclib_record.h.

9.28.2.43 axis_d_halls

`UB axis_d_halls`
D Hall Input Status.
Definition at line 156 of file gclib_record.h.

9.28.2.44 axis_d_motor_position

`SL axis_d_motor_position`
D axis motor position.
Definition at line 150 of file gclib_record.h.

9.28.2.45 axis_d_position_error

`SL axis_d_position_error`
D axis position error.
Definition at line 151 of file gclib_record.h.

9.28.2.46 axis_d_reference_position

`SL axis_d_reference_position`
D axis reference position.
Definition at line 149 of file gclib_record.h.

9.28.2.47 axis_d_reserved

`UB axis_d_reserved`
Reserved.
Definition at line 157 of file gclib_record.h.

9.28.2.48 axis_d_status

`UW axis_d_status`
D axis status.
Definition at line 146 of file gclib_record.h.

9.28.2.49 axis_d_stop_code

`UB axis_d_stop_code`
D axis stop code.
Definition at line 148 of file gclib_record.h.

9.28.2.50 axis_d_switches

`UB axis_d_switches`
D axis switches.
Definition at line 147 of file gclib_record.h.

9.28.2.51 axis_d_torque

`SL axis_d_torque`
D axis torque.
Definition at line 154 of file gclib_record.h.

9.28.2.52 axis_d_variable

`SL axis_d_variable`
D User-defined variable (ZA).
Definition at line 158 of file gclib_record.h.

9.28.2.53 axis_d_velocity

`SL axis_d_velocity`
D axis velocity.
Definition at line 153 of file gclib_record.h.

9.28.2.54 axis_e_analog_in

`UW axis_e_analog_in`
E axis analog input.
Definition at line 169 of file gclib_record.h.

9.28.2.55 axis_e_aux_position

`SL axis_e_aux_position`
E axis auxiliary position.
Definition at line 166 of file gclib_record.h.

9.28.2.56 axis_e_halls

`UB axis_e_halls`
E Hall Input Status.
Definition at line 170 of file gclib_record.h.

9.28.2.57 axis_e_motor_position

`SL axis_e_motor_position`
E axis motor position.
Definition at line 164 of file gclib_record.h.

9.28.2.58 axis_e_position_error

`SL axis_e_position_error`
E axis position error.
Definition at line 165 of file gclib_record.h.

9.28.2.59 axis_e_reference_position

`SL axis_e_reference_position`
E axis reference position.
Definition at line 163 of file gclib_record.h.

9.28.2.60 axis_e_reserved

`UB axis_e_reserved`
Reserved.
Definition at line 171 of file gclib_record.h.

9.28.2.61 axis_e_status

`UW axis_e_status`
E axis status.
Definition at line 160 of file gclib_record.h.

9.28.2.62 axis_e_stop_code

`UB axis_e_stop_code`
E axis stop code.
Definition at line 162 of file gclib_record.h.

9.28.2.63 axis_e_switches

`UB axis_e_switches`
E axis switches.
Definition at line 161 of file gclib_record.h.

9.28.2.64 axis_e_torque

`SL axis_e_torque`
E axis torque.
Definition at line 168 of file gclib_record.h.

9.28.2.65 axis_e_variable

`SL axis_e_variable`
E User-defined variable (ZA).
Definition at line 172 of file gclib_record.h.

9.28.2.66 axis_e_velocity

`SL axis_e_velocity`
E axis velocity.
Definition at line 167 of file gclib_record.h.

9.28.2.67 axis_f_analog_in

`UW axis_f_analog_in`
F axis analog input.
Definition at line 183 of file gclib_record.h.

9.28.2.68 axis_f_aux_position

`SL axis_f_aux_position`
F axis auxiliary position.
Definition at line 180 of file gclib_record.h.

9.28.2.69 axis_f_halls

`UB axis_f_halls`
F Hall Input Status.
Definition at line 184 of file gclib_record.h.

9.28.2.70 axis_f_motor_position

`SL axis_f_motor_position`
F axis motor position.
Definition at line 178 of file gclib_record.h.

9.28.2.71 axis_f_position_error

`SL axis_f_position_error`
F axis position error.
Definition at line 179 of file gclib_record.h.

9.28.2.72 axis_f_reference_position

`SL axis_f_reference_position`
F axis reference position.
Definition at line 177 of file gclib_record.h.

9.28.2.73 axis_f_reserved

`UB axis_f_reserved`
Reserved.
Definition at line 185 of file gclib_record.h.

9.28.2.74 axis_f_status

`UW axis_f_status`
F axis status.
Definition at line 174 of file gclib_record.h.

9.28.2.75 axis_f_stop_code

`UB axis_f_stop_code`
F axis stop code.
Definition at line 176 of file gclib_record.h.

9.28.2.76 axis_f_switches

`UB axis_f_switches`
F axis switches.
Definition at line 175 of file gclib_record.h.

9.28.2.77 axis_f_torque

`SL axis_f_torque`
F axis torque.
Definition at line 182 of file gclib_record.h.

9.28.2.78 axis_f_variable

`SL axis_f_variable`
F User-defined variable (ZA).
Definition at line 186 of file gclib_record.h.

9.28.2.79 axis_f_velocity

`SL axis_f_velocity`
F axis velocity.
Definition at line 181 of file gclib_record.h.

9.28.2.80 axis_g_analog_in

`UW axis_g_analog_in`
G axis analog input.
Definition at line 197 of file gclib_record.h.

9.28.2.81 axis_g_aux_position

`SL axis_g_aux_position`
G axis auxiliary position.
Definition at line 194 of file gclib_record.h.

9.28.2.82 axis_g_halls

`UB axis_g_halls`
G Hall Input Status.
Definition at line 198 of file gclib_record.h.

9.28.2.83 axis_g_motor_position

`SL axis_g_motor_position`
G axis motor position.
Definition at line 192 of file gclib_record.h.

9.28.2.84 axis_g_position_error

`SL axis_g_position_error`
G axis position error.
Definition at line 193 of file gclib_record.h.

9.28.2.85 axis_g_reference_position

`SL axis_g_reference_position`
G axis reference position.
Definition at line 191 of file gclib_record.h.

9.28.2.86 axis_g_reserved

`UB axis_g_reserved`
Reserved.
Definition at line 199 of file gclib_record.h.

9.28.2.87 axis_g_status

`UW axis_g_status`
G axis status.
Definition at line 188 of file gclib_record.h.

9.28.2.88 axis_g_stop_code

`UB axis_g_stop_code`
G axis stop code.
Definition at line 190 of file gclib_record.h.

9.28.2.89 axis_g_switches

`UB axis_g_switches`
G axis switches.
Definition at line 189 of file gclib_record.h.

9.28.2.90 axis_g_torque

`SL axis_g_torque`
G axis torque.
Definition at line 196 of file gclib_record.h.

9.28.2.91 axis_g_variable

`SL axis_g_variable`
G User-defined variable (ZA).
Definition at line 200 of file gclib_record.h.

9.28.2.92 axis_g_velocity

`SL axis_g_velocity`
G axis velocity.
Definition at line 195 of file gclib_record.h.

9.28.2.93 axis_h_analog_in

`UW axis_h_analog_in`
H axis analog input.
Definition at line 211 of file gclib_record.h.

9.28.2.94 axis_h_aux_position

`SL axis_h_aux_position`
H axis auxiliary position.
Definition at line 208 of file gclib_record.h.

9.28.2.95 axis_h_halls

`UB axis_h_halls`
H Hall Input Status.
Definition at line 212 of file gclib_record.h.

9.28.2.96 axis_h_motor_position

`SL axis_h_motor_position`
H axis motor position.
Definition at line 206 of file gclib_record.h.

9.28.2.97 axis_h_position_error

`SL axis_h_position_error`
H axis position error.
Definition at line 207 of file gclib_record.h.

9.28.2.98 axis_h_reference_position

`SL axis_h_reference_position`
H axis reference position.
Definition at line 205 of file gclib_record.h.

9.28.2.99 axis_h_reserved

`UB axis_h_reserved`
Reserved.
Definition at line 213 of file gclib_record.h.

9.28.2.100 axis_h_status

`UW axis_h_status`
H axis status.
Definition at line 202 of file gclib_record.h.

9.28.2.101 axis_h_stop_code

`UB axis_h_stop_code`
H axis stop code.
Definition at line 204 of file gclib_record.h.

9.28.2.102 axis_h_switches

`UB axis_h_switches`
H axis switches.
Definition at line 203 of file gclib_record.h.

9.28.2.103 axis_h_torque

`SL axis_h_torque`
H axis torque.
Definition at line 210 of file gclib_record.h.

9.28.2.104 axis_h_variable

`SL axis_h_variable`
H User-defined variable (ZA).
Definition at line 214 of file gclib_record.h.

9.28.2.105 axis_h_velocity

`SL axis_h_velocity`
H axis velocity.
Definition at line 209 of file gclib_record.h.

9.28.2.106 contour_buffer_available

`UW contour_buffer_available`
Buffer space remaining, Contour Mode.
Definition at line 92 of file gclib_record.h.

9.28.2.107 contour_segment_count

`UL contour_segment_count`
Segment Count for Contour Mode.
Definition at line 91 of file gclib_record.h.

9.28.2.108 error_code

`UB error_code`
error code.
Definition at line 87 of file gclib_record.h.

9.28.2.109 ethernet_status_a

`UB ethernet_status_a`
Ethernet Handle A Status.
Definition at line 78 of file gclib_record.h.

9.28.2.110 ethernet_status_b

`UB ethernet_status_b`
Ethernet Handle B Status.
Definition at line 79 of file gclib_record.h.

9.28.2.111 ethernet_status_c

`UB ethernet_status_c`
Ethernet Handle C Status.
Definition at line 80 of file gclib_record.h.

9.28.2.112 ethernet_status_d

`UB ethernet_status_d`
Ethernet Handle D Status.
Definition at line 81 of file gclib_record.h.

9.28.2.113 ethernet_status_e

`UB ethernet_status_e`
Ethernet Handle E Status.
Definition at line 82 of file gclib_record.h.

9.28.2.114 ethernet_status_f

`UB ethernet_status_f`
Ethernet Handle F Status.
Definition at line 83 of file gclib_record.h.

9.28.2.115 ethernet_status_g

`UB ethernet_status_g`
Ethernet Handle G Status.
Definition at line 84 of file gclib_record.h.

9.28.2.116 ethernet_status_h

`UB ethernet_status_h`
Ethernet Handle H Status.
Definition at line 85 of file gclib_record.h.

9.28.2.117 header_0

`UB header_0`
1st Byte of Header.
Definition at line 40 of file gclib_record.h.

9.28.2.118 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 41 of file gclib_record.h.

9.28.2.119 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 42 of file gclib_record.h.

9.28.2.120 header_3

`UB header_3`
4th Byte of Header.
Definition at line 43 of file gclib_record.h.

9.28.2.121 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 47 of file gclib_record.h.

9.28.2.122 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 48 of file gclib_record.h.

9.28.2.123 input_bank_2

`UB input_bank_2`
general input bank 2 (inputs 17-24).
Definition at line 49 of file gclib_record.h.

9.28.2.124 input_bank_3

`UB input_bank_3`
general input bank 3 (inputs 25-32).
Definition at line 50 of file gclib_record.h.

9.28.2.125 input_bank_4

`UB input_bank_4`
general input bank 4 (inputs 33-40).
Definition at line 51 of file gclib_record.h.

9.28.2.126 input_bank_5

`UB input_bank_5`
general input bank 5 (inputs 41-48).
Definition at line 52 of file gclib_record.h.

9.28.2.127 input_bank_6

`UB input_bank_6`
general input bank 6 (inputs 49-56).
Definition at line 53 of file gclib_record.h.

9.28.2.128 input_bank_7

`UB input_bank_7`
general input bank 7 (inputs 57-64).
Definition at line 54 of file gclib_record.h.

9.28.2.129 input_bank_8

`UB input_bank_8`
general input bank 8 (inputs 65-72).
Definition at line 55 of file gclib_record.h.

9.28.2.130 input_bank_9

`UB input_bank_9`
general input bank 9 (inputs 73-80).
Definition at line 56 of file gclib_record.h.

9.28.2.131 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 58 of file gclib_record.h.

9.28.2.132 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 59 of file gclib_record.h.

9.28.2.133 output_bank_2

`UB output_bank_2`
general output bank 2 (outputs 17-24).
Definition at line 60 of file gclib_record.h.

9.28.2.134 output_bank_3

`UB output_bank_3`
general output bank 3 (outputs 25-32).
Definition at line 61 of file gclib_record.h.

9.28.2.135 output_bank_4

`UB output_bank_4`
general output bank 4 (outputs 33-40).
Definition at line 62 of file gclib_record.h.

9.28.2.136 output_bank_5

`UB output_bank_5`
general output bank 5 (outputs 41-48).
Definition at line 63 of file gclib_record.h.

9.28.2.137 output_bank_6

`UB output_bank_6`
general output bank 6 (outputs 49-56).
Definition at line 64 of file gclib_record.h.

9.28.2.138 output_bank_7

`UB output_bank_7`
general output bank 7 (outputs 57-64).
Definition at line 65 of file gclib_record.h.

9.28.2.139 output_bank_8

`UB output_bank_8`
general output bank 8 (outputs 65-72).
Definition at line 66 of file gclib_record.h.

9.28.2.140 output_bank_9

`UB output_bank_9`
general output bank 9 (outputs 73-80).
Definition at line 67 of file gclib_record.h.

9.28.2.141 reserved_0

`SW reserved_0`
Reserved.
Definition at line 69 of file gclib_record.h.

9.28.2.142 reserved_10

`SW reserved_10`
Reserved.
Definition at line 74 of file gclib_record.h.

9.28.2.143 reserved_12

`SW reserved_12`
Reserved.
Definition at line 75 of file gclib_record.h.

9.28.2.144 reserved_14

`SW reserved_14`
Reserved.
Definition at line 76 of file gclib_record.h.

9.28.2.145 reserved_2

`SW reserved_2`
Reserved.
Definition at line 70 of file gclib_record.h.

9.28.2.146 reserved_4

`SW reserved_4`
Reserved.
Definition at line 71 of file gclib_record.h.

9.28.2.147 reserved_6

`SW reserved_6`
Reserved.
Definition at line 72 of file gclib_record.h.

9.28.2.148 reserved_8

`SW reserved_8`
Reserved.
Definition at line 73 of file gclib_record.h.

9.28.2.149 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 96 of file gclib_record.h.

9.28.2.150 s_plane_buffer_available

`UW s_plane_buffer_available`
Buffer space remaining, S Plane.
Definition at line 97 of file gclib_record.h.

9.28.2.151 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 95 of file gclib_record.h.

9.28.2.152 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 94 of file gclib_record.h.

9.28.2.153 sample_number

`UW sample_number`
sample number.
Definition at line 45 of file gclib_record.h.

9.28.2.154 t_distance

`SL t_distance`
distance traveled in coordinated move for T plane.
Definition at line 101 of file gclib_record.h.

9.28.2.155 t_plane_buffer_available

`UW t_plane_buffer_available`
Buffer space remaining, T Plane.
Definition at line 102 of file gclib_record.h.

9.28.2.156 t_plane_move_status

`UW t_plane_move_status`
Coordinated move status for T plane.
Definition at line 100 of file gclib_record.h.

9.28.2.157 t_plane_segment_count

`UW t_plane_segment_count`
segment count of coordinated move for T plane.
Definition at line 99 of file gclib_record.h.

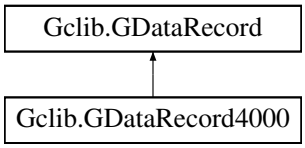
9.28.2.158 thread_status

`UB thread_status`
thread status
Definition at line 88 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

9.29 Gclib.GDataRecord4000 Struct Reference

Inheritance diagram for Gclib.GDataRecord4000:



Public Member Functions

- general input **bank** (inputs 1-8). ∗/public UB input_bank_1
  /∗07
- general input **bank** (inputs 9-16). ∗/public UB input_bank_2
  /∗08
- general input **bank** (inputs 17-24). ∗/public UB input_bank_3
  /∗09
- general input **bank** (inputs 25-32). ∗/public UB input_bank_4
  /∗10
- general input **bank** (inputs 33-40). ∗/public UB input_bank_5
  /∗11
- general input **bank** (inputs 41-48). ∗/public UB input_bank_6
  /∗12
- general input **bank** (inputs 49-56). ∗/public UB input_bank_7
  /∗13
- general input **bank** (inputs 57-64). ∗/public UB input_bank_8
  /∗14
- general input **bank** (inputs 65-72). ∗/public UB input_bank_9
  /∗15
- general input **bank** (inputs 73-80). ∗/public UB output_bank_0
  /∗16
- general output **bank** (outputs 1-8). ∗/public UB output_bank_1
  /∗17
- general output **bank** (outputs 9-16). ∗/public UB output_bank_2
  /∗18
- general output **bank** (outputs 17-24). ∗/public UB output_bank_3
  /∗19
- general output **bank** (outputs 25-32). ∗/public UB output_bank_4
  /∗20
- general output **bank** (outputs 33-40). ∗/public UB output_bank_5
  /∗21
- general output **bank** (outputs 41-48). ∗/public UB output_bank_6
  /∗22
- general output **bank** (outputs 49-56). ∗/public UB output_bank_7
  /∗23
- general output **bank** (outputs 57-64). ∗/public UB output_bank_8
  /∗24
- general output **bank** (outputs 65-72). ∗/public UB output_bank_9
  /∗25
- general output **bank** (outputs 73-80). ∗/public SW reserved_0
  /∗26-27
- H User defined variable.∗[] byte_array ()

Data Fields

- UB header_0
  /*00
- byte of Header.*UB header_1
  /*01
- byte of Header.*UB header_2
  /*02
- byte of Header.*UB header_3
  /*03
- byte of Header.*UW sample_number
  /*04-05
- sample number.*UB input_bank_0
  /*06
- Reserved.*SW reserved_2
  /*28-29
- Reserved.*SW reserved_4
  /*30-31
- Reserved.*SW reserved_6
  /*32-33
- Reserved.*SW reserved_8
  /*34-35
- Reserved.*SW reserved_10
  /*36-37
- Reserved.*SW reserved_12
  /*38-39
- Reserved.*SW reserved_14
  /*40-41
- Reserved.*UB ethernet_status_a
  /*42
- Ethernet Handle A Status.*UB ethernet_status_b
  /*43
- Ethernet Handle B Status.*UB ethernet_status_c
  /*44
- Ethernet Handle C Status.*UB ethernet_status_d
  /*45
- Ethernet Handle D Status.*UB ethernet_status_e
  /*46
- Ethernet Handle E Status.*UB ethernet_status_f
  /*47
- Ethernet Handle F Status.*UB ethernet_status_g
  /*48
- Ethernet Handle G Status.*UB ethernet_status_h
  /*49
- Ethernet Handle H Status.*UB error_code
  /*50
- error code.*UB thread_status
  /*51
- thread status *UL amplifier_status
  /*52-55
- Amplifier Status.*UL contour_segment_count

/*56-59

- Segment Count for Contour Mode.*UW contour_buffer_available

  /*60-61
- Buffer space remaining
- Buffer space Contour Mode.*UW s_plane_segment_count

  /*62-63
- segment count of coordinated move for S plane.*UW s_plane_move_status

  /*64-65
- coordinated move status for S plane.*SL s_distance

  /*66-69
- distance traveled in coordinated move for S plane.*UW s_plane_buffer_available

  /*70-71
- Buffer space S Plane.*UW t_plane_segment_count

  /*72-73
- segment count of coordinated move for T plane.*UW t_plane_move_status

  /*74-75
- Coordinated move status for T plane.*SL t_distance

  /*76-79
- distance traveled in coordinated move for T plane.*UW t_plane_buffer_available

  /*80-81
- Buffer space T Plane.*UW axis_a_status

  /*82-83
- A axis status.*UB axis_a_switches

  /*84
- A axis switches.*UB axis_a_stop_code

  /*85
- A axis stop code.*SL axis_a_reference_position

  /*86-89
- A axis reference position.*SL axis_a_motor_position

  /*90-93
- A axis motor position.*SL axis_a_position_error

  /*94-97
- A axis position error.*SL axis_a_aux_position

  /*98-101
- A axis auxiliary position.*SL axis_a_velocity

  /*102-105
- A axis velocity.*SL axis_a_torque

  /*106-109
- A axis torque.*UW axis_a_analog_in

  /*110-111
- A axis analog input.*UB axis_a_halls

  /*112
- A Hall Input Status.*UB axis_a_reserved

  /*113
- Reserved.*SL axis_a_variable

  /*114-117
- A User defined variable.*[] UW axis_b_status

  /*118-119
- B axis status.*UB axis_b_switches

  /*120
- B axis switches.*UB axis_b_stop_code

/*121

- B axis stop code.*SL axis_b_reference_position

  /*122-125

- B axis reference position.*SL axis_b_motor_position

  /*126-129

- B axis motor position.*SL axis_b_position_error

  /*130-133

- B axis position error.*SL axis_b_aux_position

  /*134-137

- B axis auxiliary position.*SL axis_b_velocity

  /*138-141

- B axis velocity.*SL axis_b_torque

  /*142-145

- B axis torque.*UW axis_b_analog_in

  /*146-147

- B axis analog input.*UB axis_b_halls

  /*148

- B Hall Input Status.*UB axis_b_reserved

  /*149

- Reserved.*SL axis_b_variable

  /*150-153

- B User defined variable.*[] UW axis_c_status

  /*154-155

- C axis status.*UB axis_c_switches

  /*156

- C axis switches.*UB axis_c_stop_code

  /*157

- C axis stop code.*SL axis_c_reference_position

  /*158-161

- C axis reference position.*SL axis_c_motor_position

  /*162-165

- C axis motor position.*SL axis_c_position_error

  /*166-169

- C axis position error.*SL axis_c_aux_position

  /*170-173

- C axis auxiliary position.*SL axis_c_velocity

  /*174-177

- C axis velocity.*SL axis_c_torque

  /*178-181

- C axis torque.*UW axis_c_analog_in

  /*182-183

- C axis analog input.*UB axis_c_halls

  /*184

- C Hall Input Status.*UB axis_c_reserved

  /*185

- Reserved.*SL axis_c_variable

  /*186-189

- C User defined variable.*[] UW axis_d_status

  /*190-191

- D axis status.*UB axis_d_switches

  /*192

- D axis switches.∗UB axis_d_stop_code
  /∗193
- D axis stop code.∗SL axis_d_reference_position
  /∗194-197
- D axis reference position.∗SL axis_d_motor_position
  /∗198-201
- D axis motor position.∗SL axis_d_position_error
  /∗202-205
- D axis position error.∗SL axis_d_aux_position
  /∗206-209
- D axis auxiliary position.∗SL axis_d_velocity
  /∗210-213
- D axis velocity.∗SL axis_d_torque
  /∗214-217
- D axis torque.∗UW axis_d_analog_in
  /∗218-219
- D axis analog input.∗UB axis_d_halls
  /∗220
- D Hall Input Status.∗UB axis_d_reserved
  /∗221
- Reserved.∗SL axis_d_variable
  /∗222-225
- D User defined variable.∗[] UW axis_e_status
  /∗226-227
- E axis status.∗UB axis_e_switches
  /∗228
- E axis switches.∗UB axis_e_stop_code
  /∗229
- E axis stop code.∗SL axis_e_reference_position
  /∗230-233
- E axis reference position.∗SL axis_e_motor_position
  /∗234-237
- E axis motor position.∗SL axis_e_position_error
  /∗238-241
- E axis position error.∗SL axis_e_aux_position
  /∗242-245
- E axis auxiliary position.∗SL axis_e_velocity
  /∗246-249
- E axis velocity.∗SL axis_e_torque
  /∗250-253
- E axis torque.∗UW axis_e_analog_in
  /∗254-255
- E axis analog input.∗UB axis_e_halls
  /∗256
- E Hall Input Status.∗UB axis_e_reserved
  /∗257
- Reserved.∗SL axis_e_variable
  /∗258-261
- E User defined variable.∗[] UW axis_f_status
  /∗262-263
- F axis status.∗UB axis_f_switches

/*264

- F axis switches.*UB axis_f_stop_code

  /*265

- F axis stop code.*SL axis_f_reference_position

  /*266-269

- F axis reference position.*SL axis_f_motor_position

  /*270-273

- F axis motor position.*SL axis_f_position_error

  /*274-277

- F axis position error.*SL axis_f_aux_position

  /*278-281

- F axis auxiliary position.*SL axis_f_velocity

  /*282-285

- F axis velocity.*SL axis_f_torque

  /*286-289

- F axis torque.*UW axis_f_analog_in

  /*290-291

- F axis analog input.*UB axis_f_halls

  /*292

- F Hall Input Status.*UB axis_f_reserved

  /*293

- Reserved.*SL axis_f_variable

  /*294-297

- F User defined variable.*[] UW axis_g_status

  /*298-299

- G axis status.*UB axis_g_switches

  /*300

- G axis switches.*UB axis_g_stop_code

  /*301

- G axis stop code.*SL axis_g_reference_position

  /*302-305

- G axis reference position.*SL axis_g_motor_position

  /*306-309

- G axis motor position.*SL axis_g_position_error

  /*310-313

- G axis position error.*SL axis_g_aux_position

  /*314-317

- G axis auxiliary position.*SL axis_g_velocity

  /*318-321

- G axis velocity.*SL axis_g_torque

  /*322-325

- G axis torque.*UW axis_g_analog_in

  /*326-327

- G axis analog input.*UB axis_g_halls

  /*328

- G Hall Input Status.*UB axis_g_reserved

  /*329

- Reserved.*SL axis_g_variable

  /*330-333

- G User defined variable.*[] UW axis_h_status

  /*334-335

- H axis status.∗UB axis_h_switches
  /∗336
- H axis switches.∗UB axis_h_stop_code
  /∗337
- H axis stop code.∗SL axis_h_reference_position
  /∗338-341
- H axis reference position.∗SL axis_h_motor_position
  /∗342-345
- H axis motor position.∗SL axis_h_position_error
  /∗346-349
- H axis position error.∗SL axis_h_aux_position
  /∗350-353
- H axis auxiliary position.∗SL axis_h_velocity
  /∗354-357
- H axis velocity.∗SL axis_h_torque
  /∗358-361
- H axis torque.∗UW axis_h_analog_in
  /∗362-363
- H axis analog input.∗UB axis_h_halls
  /∗364
- H Hall Input Status.∗UB axis_h_reserved
  /∗365
- Reserved.∗SL axis_h_variable
  /∗366-369

9.29.1 Detailed Description

Definition at line 472 of file gclib.vb.

9.29.2 Member Function Documentation

9.29.2.1 byte_array()

```
H User defined variable.∗[] byte_array ( )
```

Implements GDataRecord.byte_array
Implements Gclib.GDataRecord.

9.29.3 Field Documentation

9.29.3.1 amplifier_status

```
thread status∗ UL amplifier_status
```

/∗52-55
Definition at line 518 of file gclib.vb.

9.29.3.2 axis_a_analog_in

```
A axis torque.∗ UW axis_a_analog_in
```

/∗110-111
Definition at line 538 of file gclib.vb.

9.29.3.3 axis_a_aux_position

```
A axis position error.∗ SL axis_a_aux_position
```

/∗98-101
Definition at line 535 of file gclib.vb.

9.29.3.4 axis_a_halls

```
A axis analog input.* UB axis_a_halls
```
/*112
Definition at line 539 of file gclib.vb.

9.29.3.5 axis_a_motor_position

```
A axis reference position.* SL axis_a_motor_position
```
/*90-93
Definition at line 533 of file gclib.vb.

9.29.3.6 axis_a_position_error

```
A axis motor position.* SL axis_a_position_error
```
/*94-97
Definition at line 534 of file gclib.vb.

9.29.3.7 axis_a_reference_position

```
A axis stop code.* SL axis_a_reference_position
```
/*86-89
Definition at line 532 of file gclib.vb.

9.29.3.8 axis_a_reserved

```
A Hall Input Status.* UB axis_a_reserved
```
/*113
Definition at line 540 of file gclib.vb.

9.29.3.9 axis_a_status

```
Buffer space T Plane.* UW axis_a_status
```
/*82-83
Definition at line 529 of file gclib.vb.

9.29.3.10 axis_a_stop_code

```
A axis switches.* UB axis_a_stop_code
```
/*85
Definition at line 531 of file gclib.vb.

9.29.3.11 axis_a_switches

```
A axis status.* UB axis_a_switches
```
/*84
Definition at line 530 of file gclib.vb.

9.29.3.12 axis_a_torque

```
A axis velocity.* SL axis_a_torque
```
/*106-109
Definition at line 537 of file gclib.vb.

9.29.3.13 axis_a_variable

```
Reserved.* SL axis_a_variable
```
/*114-117
Definition at line 541 of file gclib.vb.

9.29.3.14 axis_a_velocity

```
A axis auxiliary position.* SL axis_a_velocity
```
/*102-105
Definition at line 536 of file gclib.vb.

9.29.3.15 axis_b_analog_in

```
B axis torque.* UW axis_b_analog_in
```
/*146-147
Definition at line 551 of file gclib.vb.

9.29.3.16 axis_b_aux_position

```
B axis position error.* SL axis_b_aux_position
```
/*134-137
Definition at line 548 of file gclib.vb.

9.29.3.17 axis_b_halls

```
B axis analog input.* UB axis_b_halls
```
/*148
Definition at line 552 of file gclib.vb.

9.29.3.18 axis_b_motor_position

```
B axis reference position.* SL axis_b_motor_position
```
/*126-129
Definition at line 546 of file gclib.vb.

9.29.3.19 axis_b_position_error

```
B axis motor position.* SL axis_b_position_error
```
/*130-133
Definition at line 547 of file gclib.vb.

9.29.3.20 axis_b_reference_position

```
B axis stop code.* SL axis_b_reference_position
```
/*122-125
Definition at line 545 of file gclib.vb.

9.29.3.21 axis_b_reserved

```
B Hall Input Status.* UB axis_b_reserved
```
/*149
Definition at line 553 of file gclib.vb.

9.29.3.22 axis_b_status

```
A User defined variable.* [] UW axis_b_status
```
/*118-119
Definition at line 542 of file gclib.vb.

9.29.3.23 axis_b_stop_code

```
B axis switches.* UB axis_b_stop_code
```
/*121
Definition at line 544 of file gclib.vb.

9.29.3.24 axis_b_switches

```
B axis status.* UB axis_b_switches
```
/*120
Definition at line 543 of file gclib.vb.

9.29.3.25 axis_b_torque

```
B axis velocity.* SL axis_b_torque
```
/*142-145
Definition at line 550 of file gclib.vb.

9.29.3.26 axis_b_variable

```
Reserved.* SL axis_b_variable
```
/*150-153
Definition at line 554 of file gclib.vb.

9.29.3.27 axis_b_velocity

```
B axis auxiliary position.* SL axis_b_velocity
```
/*138-141
Definition at line 549 of file gclib.vb.

9.29.3.28 axis_c_analog_in

```
C axis torque.* UW axis_c_analog_in
```
/*182-183
Definition at line 564 of file gclib.vb.

9.29.3.29 axis_c_aux_position

```
C axis position error.* SL axis_c_aux_position
```
/*170-173
Definition at line 561 of file gclib.vb.

9.29.3.30 axis_c_halls

```
C axis analog input.* UB axis_c_halls
```
/*184
Definition at line 565 of file gclib.vb.

9.29.3.31 axis_c_motor_position

```
C axis reference position.* SL axis_c_motor_position
```
/*162-165
Definition at line 559 of file gclib.vb.

9.29.3.32 axis_c_position_error

```
C axis motor position.* SL axis_c_position_error
```
/*166-169
Definition at line 560 of file gclib.vb.

9.29.3.33 axis_c_reference_position

```
C axis stop code.* SL axis_c_reference_position
```
/*158-161
Definition at line 558 of file gclib.vb.

9.29.3.34 axis_c_reserved

```
C Hall Input Status.* UB axis_c_reserved
```
/*185
Definition at line 566 of file gclib.vb.

9.29.3.35 axis_c_status

```
B User defined variable.* [] UW axis_c_status
```
/*154-155
Definition at line 555 of file gclib.vb.

9.29.3.36 axis_c_stop_code

```
C axis switches.* UB axis_c_stop_code
```
/*157
Definition at line 557 of file gclib.vb.

9.29.3.37 axis_c_switches

```
C axis status.* UB axis_c_switches
```
/*156
Definition at line 556 of file gclib.vb.

9.29.3.38 axis_c_torque

```
C axis velocity.* SL axis_c_torque
```
/*178-181
Definition at line 563 of file gclib.vb.

9.29.3.39 axis_c_variable

```
Reserved.* SL axis_c_variable
```
/*186-189
Definition at line 567 of file gclib.vb.

9.29.3.40 axis_c_velocity

```
C axis auxiliary position.* SL axis_c_velocity
```
/*174-177
Definition at line 562 of file gclib.vb.

9.29.3.41 axis_d_analog_in

```
D axis torque.* UW axis_d_analog_in
```
/*218-219
Definition at line 577 of file gclib.vb.

9.29.3.42 axis_d_aux_position

```
D axis position error.* SL axis_d_aux_position
```
/*206-209
Definition at line 574 of file gclib.vb.

9.29.3.43 axis_d_halls

```
D axis analog input.* UB axis_d_halls
```
/*220
Definition at line 578 of file gclib.vb.

9.29.3.44 axis_d_motor_position

```
D axis reference position.* SL axis_d_motor_position
```
/*198-201
Definition at line 572 of file gclib.vb.

9.29.3.45 axis_d_position_error

```
D axis motor position.* SL axis_d_position_error
```
/*202-205
Definition at line 573 of file gclib.vb.

9.29.3.46 axis_d_reference_position

```
D axis stop code.* SL axis_d_reference_position
```
/*194-197
Definition at line 571 of file gclib.vb.

9.29.3.47 axis_d_reserved

```
D Hall Input Status.* UB axis_d_reserved
```
/*221
Definition at line 579 of file gclib.vb.

9.29.3.48 axis_d_status

```
C User defined variable.* [] UW axis_d_status
```
/*190-191
Definition at line 568 of file gclib.vb.

9.29.3.49 axis_d_stop_code

```
D axis switches.* UB axis_d_stop_code
```
/*193
Definition at line 570 of file gclib.vb.

9.29.3.50 axis_d_switches

```
D axis status.* UB axis_d_switches
```
/*192
Definition at line 569 of file gclib.vb.

9.29.3.51 axis_d_torque

```
D axis velocity.* SL axis_d_torque
```
/*214-217
Definition at line 576 of file gclib.vb.

9.29.3.52 axis_d_variable

```
Reserved.* SL axis_d_variable
```
/*222-225
Definition at line 580 of file gclib.vb.

9.29.3.53 axis_d_velocity

```
D axis auxiliary position.* SL axis_d_velocity
```
/*210-213
Definition at line 575 of file gclib.vb.

9.29.3.54 axis_e_analog_in

```
E axis torque.* UW axis_e_analog_in
```
/*254-255
Definition at line 590 of file gclib.vb.

9.29.3.55 axis_e_aux_position

```
E axis position error.* SL axis_e_aux_position
```
/*242-245
Definition at line 587 of file gclib.vb.

9.29.3.56 axis_e_halls

```
E axis analog input.* UB axis_e_halls
```
/*256
Definition at line 591 of file gclib.vb.

9.29.3.57 axis_e_motor_position

```
E axis reference position.* SL axis_e_motor_position
```
/*234-237
Definition at line 585 of file gclib.vb.

9.29.3.58 axis_e_position_error

```
E axis motor position.* SL axis_e_position_error
```
/*238-241
Definition at line 586 of file gclib.vb.

9.29.3.59 axis_e_reference_position

```
E axis stop code.* SL axis_e_reference_position
```
/*230-233
Definition at line 584 of file gclib.vb.

9.29.3.60 axis_e_reserved

```
E Hall Input Status.* UB axis_e_reserved
```
/*257
Definition at line 592 of file gclib.vb.

9.29.3.61 axis_e_status

```
D User defined variable.* [] UW axis_e_status
```
/*226-227
Definition at line 581 of file gclib.vb.

9.29.3.62 axis_e_stop_code

```
E axis switches.* UB axis_e_stop_code
```
/*229
Definition at line 583 of file gclib.vb.

9.29.3.63 axis_e_switches

```
E axis status.* UB axis_e_switches
```
/*228
Definition at line 582 of file gclib.vb.

9.29.3.64 axis_e_torque

```
E axis velocity.* SL axis_e_torque
```
/*250-253
Definition at line 589 of file gclib.vb.

9.29.3.65 axis_e_variable

```
Reserved.* SL axis_e_variable
```
/*258-261
Definition at line 593 of file gclib.vb.

9.29.3.66 axis_e_velocity

```
E axis auxiliary position.* SL axis_e_velocity
```
/*246-249
Definition at line 588 of file gclib.vb.

9.29.3.67 axis_f_analog_in

```
F axis torque.* UW axis_f_analog_in
```
/*290-291
Definition at line 603 of file gclib.vb.

9.29.3.68 axis_f_aux_position

```
F axis position error.* SL axis_f_aux_position
```
/*278-281
Definition at line 600 of file gclib.vb.

9.29.3.69 axis_f_halls

```
F axis analog input.* UB axis_f_halls
```
/*292
Definition at line 604 of file gclib.vb.

9.29.3.70 axis_f_motor_position

```
F axis reference position.* SL axis_f_motor_position
```
/*270-273
Definition at line 598 of file gclib.vb.

9.29.3.71 axis_f_position_error

```
F axis motor position.* SL axis_f_position_error
```
/*274-277
Definition at line 599 of file gclib.vb.

9.29.3.72 axis_f_reference_position

```
F axis stop code.* SL axis_f_reference_position
```
/*266-269
Definition at line 597 of file gclib.vb.

9.29.3.73 axis_f_reserved

```
F Hall Input Status.* UB axis_f_reserved
```
/*293
Definition at line 605 of file gclib.vb.

9.29.3.74 axis_f_status

```
E User defined variable.* [] UW axis_f_status
```
/*262-263
Definition at line 594 of file gclib.vb.

9.29.3.75 axis_f_stop_code

```
F axis switches.* UB axis_f_stop_code
```
/*265
Definition at line 596 of file gclib.vb.

9.29.3.76 axis_f_switches

```
F axis status.* UB axis_f_switches
```
/*264
Definition at line 595 of file gclib.vb.

9.29.3.77 axis_f_torque

```
F axis velocity.* SL axis_f_torque
```
/*286-289
Definition at line 602 of file gclib.vb.

9.29.3.78 axis_f_variable

```
Reserved.* SL axis_f_variable
```
/*294-297
Definition at line 606 of file gclib.vb.

9.29.3.79 axis_f_velocity

```
F axis auxiliary position.* SL axis_f_velocity
```
/*282-285
Definition at line 601 of file gclib.vb.

9.29.3.80 axis_g_analog_in

```
G axis torque.* UW axis_g_analog_in
```
/*326-327
Definition at line 616 of file gclib.vb.

9.29.3.81 axis_g_aux_position

```
G axis position error.* SL axis_g_aux_position
```
/*314-317
Definition at line 613 of file gclib.vb.

9.29.3.82 axis_g_halls

```
G axis analog input.* UB axis_g_halls
```
/*328
Definition at line 617 of file gclib.vb.

9.29.3.83 axis_g_motor_position

```
G axis reference position.* SL axis_g_motor_position
```
/*306-309
Definition at line 611 of file gclib.vb.

9.29.3.84 axis_g_position_error

```
G axis motor position.* SL axis_g_position_error
```
/*310-313
Definition at line 612 of file gclib.vb.

9.29.3.85 axis_g_reference_position

```
G axis stop code.* SL axis_g_reference_position
```
/*302-305
Definition at line 610 of file gclib.vb.

9.29.3.86 axis_g_reserved

```
G Hall Input Status.* UB axis_g_reserved
```
/*329
Definition at line 618 of file gclib.vb.

9.29.3.87 axis_g_status

```
F User defined variable.* [] UW axis_g_status
```
/*298-299
Definition at line 607 of file gclib.vb.

9.29.3.88 axis_g_stop_code

```
G axis switches.* UB axis_g_stop_code
```
/*301
Definition at line 609 of file gclib.vb.

9.29.3.89 axis_g_switches

```
G axis status.* UB axis_g_switches
```
/*300
Definition at line 608 of file gclib.vb.

9.29.3.90 axis_g_torque

```
G axis velocity.* SL axis_g_torque
```
/*322-325
Definition at line 615 of file gclib.vb.

9.29.3.91 axis_g_variable

```
Reserved.* SL axis_g_variable
```
/*330-333
Definition at line 619 of file gclib.vb.

9.29.3.92 axis_g_velocity

```
G axis auxiliary position.* SL axis_g_velocity
```
/*318-321
Definition at line 614 of file gclib.vb.

9.29.3.93 axis_h_analog_in

```
H axis torque.* UW axis_h_analog_in
```
/*362-363
Definition at line 629 of file gclib.vb.

9.29.3.94 axis_h_aux_position

```
H axis position error.* SL axis_h_aux_position
```
/*350-353
Definition at line 626 of file gclib.vb.

9.29.3.95 axis_h_halls

```
H axis analog input.* UB axis_h_halls
```
/*364
Definition at line 630 of file gclib.vb.

9.29.3.96 axis_h_motor_position

```
H axis reference position.* SL axis_h_motor_position
```
/*342-345
Definition at line 624 of file gclib.vb.

9.29.3.97 axis_h_position_error

```
H axis motor position.* SL axis_h_position_error
```
/*346-349
Definition at line 625 of file gclib.vb.

9.29.3.98 axis_h_reference_position

```
H axis stop code.* SL axis_h_reference_position
```
/*338-341
Definition at line 623 of file gclib.vb.

9.29.3.99 axis_h_reserved

```
H Hall Input Status.* UB axis_h_reserved
```
/*365
Definition at line 631 of file gclib.vb.

9.29.3.100 axis_h_status

```
G User defined variable.* [] UW axis_h_status
```
/*334-335
Definition at line 620 of file gclib.vb.

9.29.3.101 axis_h_stop_code

```
H axis switches.* UB axis_h_stop_code
```
/*337
Definition at line 622 of file gclib.vb.

9.29.3.102 axis_h_switches

```
H axis status.* UB axis_h_switches
```
/*336
Definition at line 621 of file gclib.vb.

9.29.3.103 axis_h_torque

```
H axis velocity.* SL axis_h_torque
```
/*358-361
Definition at line 628 of file gclib.vb.

9.29.3.104 axis_h_variable

```
Reserved.* SL axis_h_variable
```
/*366-369
Definition at line 632 of file gclib.vb.

9.29.3.105 axis_h_velocity

```
H axis auxiliary position.* SL axis_h_velocity
```
/*354-357
Definition at line 627 of file gclib.vb.

9.29.3.106 contour_buffer_available

```
Segment Count for Contour Mode.* UW contour_buffer_available
```
/*60-61
Definition at line 520 of file gclib.vb.

9.29.3.107 contour_segment_count

```
Amplifier Status.* UL contour_segment_count
```
/*56-59
Definition at line 519 of file gclib.vb.

9.29.3.108 error_code

```
Ethernet Handle H Status.* UB error_code
```
/*50
Definition at line 516 of file gclib.vb.

9.29.3.109 ethernet_status_a

```
Reserved.* UB ethernet_status_a
```
/*42
Definition at line 508 of file gclib.vb.

9.29.3.110 ethernet_status_b

```
Ethernet Handle A Status.* UB ethernet_status_b
```
/*43
Definition at line 509 of file gclib.vb.

9.29.3.111 ethernet_status_c

```
Ethernet Handle B Status.* UB ethernet_status_c
```
/*44
Definition at line 510 of file gclib.vb.

9.29.3.112 ethernet_status_d

```
Ethernet Handle C Status.* UB ethernet_status_d
```
/*45
Definition at line 511 of file gclib.vb.

9.29.3.113 ethernet_status_e

```
Ethernet Handle D Status.* UB ethernet_status_e
```
/*46
Definition at line 512 of file gclib.vb.

9.29.3.114 ethernet_status_f

```
Ethernet Handle E Status.* UB ethernet_status_f
```

/*47
Definition at line 513 of file gclib.vb.

9.29.3.115 ethernet_status_g

```
Ethernet Handle F Status.* UB ethernet_status_g
```

/*48
Definition at line 514 of file gclib.vb.

9.29.3.116 ethernet_status_h

```
Ethernet Handle G Status.* UB ethernet_status_h
```

/*49
Definition at line 515 of file gclib.vb.

9.29.3.117 header_0

```
UB header_0
```

/*00
Definition at line 475 of file gclib.vb.

9.29.3.118 header_1

```
byte of Header.* UB header_1
```

/*01
Definition at line 476 of file gclib.vb.

9.29.3.119 header_2

```
byte of Header.* UB header_2
```

/*02
Definition at line 477 of file gclib.vb.

9.29.3.120 header_3

```
byte of Header.* UB header_3
```

/*03
Definition at line 478 of file gclib.vb.

9.29.3.121 input_bank_0

```
sample number.* UB input_bank_0
```

/*06
Definition at line 480 of file gclib.vb.

9.29.3.122 remaining

```
Buffer space remaining
```

Definition at line 520 of file gclib.vb.

9.29.3.123 reserved_10

```
Reserved.* SW reserved_10
```

/*36-37
Definition at line 505 of file gclib.vb.

9.29.3.124 reserved_12

```
Reserved.* SW reserved_12
```
/*38-39
Definition at line 506 of file gclib.vb.

9.29.3.125 reserved_14

```
Reserved.* SW reserved_14
```
/*40-41
Definition at line 507 of file gclib.vb.

9.29.3.126 reserved_2

```
Reserved.* SW reserved_2
```
/*28-29
Definition at line 501 of file gclib.vb.

9.29.3.127 reserved_4

```
Reserved.* SW reserved_4
```
/*30-31
Definition at line 502 of file gclib.vb.

9.29.3.128 reserved_6

```
Reserved.* SW reserved_6
```
/*32-33
Definition at line 503 of file gclib.vb.

9.29.3.129 reserved_8

```
Reserved.* SW reserved_8
```
/*34-35
Definition at line 504 of file gclib.vb.

9.29.3.130 s_distance

```
coordinated move status for S plane.* SL s_distance
```
/*66-69
Definition at line 523 of file gclib.vb.

9.29.3.131 s_plane_buffer_available

```
distance traveled in coordinated move for S plane.* UW s_plane_buffer_available
```
/*70-71
Definition at line 524 of file gclib.vb.

9.29.3.132 s_plane_move_status

```
segment count of coordinated move for S plane.* UW s_plane_move_status
```
/*64-65
Definition at line 522 of file gclib.vb.

9.29.3.133 s_plane_segment_count

```
Buffer space Contour Mode.* UW s_plane_segment_count
```
/*62-63
Definition at line 521 of file gclib.vb.

9.29.3.134 sample_number

```
byte of Header.* UW sample_number
```
/*04-05
Definition at line 479 of file gclib.vb.

9.29.3.135 t_distance

```
Coordinated move status for T plane.* SL t_distance
```
/*76-79
Definition at line 527 of file gclib.vb.

9.29.3.136 t_plane_buffer_available

```
distance traveled in coordinated move for T plane.* UW t_plane_buffer_available
```
/*80-81
Definition at line 528 of file gclib.vb.

9.29.3.137 t_plane_move_status

```
segment count of coordinated move for T plane.* UW t_plane_move_status
```
/*74-75
Definition at line 526 of file gclib.vb.

9.29.3.138 t_plane_segment_count

```
Buffer space S Plane.* UW t_plane_segment_count
```
/*72-73
Definition at line 525 of file gclib.vb.

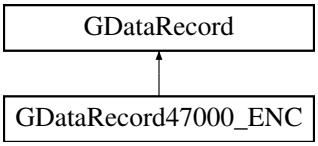
9.29.3.139 thread_status

```
error code.* UB thread_status
```
/*51
Definition at line 517 of file gclib.vb.
The documentation for this struct was generated from the following file:

- gclib.vb

9.30 GDataRecord47000_ENC Struct Reference

Data record struct for RIO-471xx and RIO-472xx PLCs. Includes encoder fields.
Inheritance diagram for GDataRecord47000_ENC:



Public Member Functions

- byte[] byte_array ()

  Returns the data record as a byte array and allows for access to individual bytes.

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.
- UB header_3

  4th Byte of Header.
- UW sample_number

  Sample number.
- UB error_code

  Error code.
- UB general_status

  General status.
- UW output_analog_0

  Analog output 0.
- UW output_analog_1

  Analog output 1.
- UW output_analog_2

  Analog output 2.
- UW output_analog_3

  Analog output 3.
- UW output_analog_4

  Analog output 4.
- UW output_analog_5

  Analog output 5.
- UW output_analog_6

  Analog output 6.
- UW output_analog_7

  Analog output 7.
- UW input_analog_0

  Analog input 0.
- UW input_analog_1

  Analog input 1.
- UW input_analog_2

  Analog input 2.
- UW input_analog_3

  Analog input 3.
- UW input_analog_4

  Analog input 4.
- UW input_analog_5

  Analog input 5.
- UW input_analog_6

  Analog input 6.
- UW input_analog_7

  Analog input 7.
- UW output_bank_0

  Digital outputs 0-15;.
- UW input_bank_0

Digital inputs 0-15;.

- UL pulse_count_0

  Pulse counter (see PC).

- SL zc_variable

  ZC User-defined variable (see ZC).

- SL zd_variable

  ZD User-defined variable (see ZD).

- SL encoder_0

  Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.

- SL encoder_1

  Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.

- SL encoder_2

  Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.

- SL encoder_3

  Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.

9.30.1 Detailed Description

Data record struct for RIO-471xx and RIO-472xx PLCs. Includes encoder fields.
Definition at line 1762 of file gclib.cs.

9.30.2 Member Function Documentation

9.30.2.1 byte_array()

```
byte[] byte_array ( )  [inline]
```

Returns the data record as a byte array and allows for access to individual bytes.
Implements GDataRecord.
Definition at line 1764 of file gclib.cs.

9.30.3 Field Documentation

9.30.3.1 encoder_0

```
SL encoder_0
```

Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1803 of file gclib.cs.

9.30.3.2 encoder_1

```
SL encoder_1
```

Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1804 of file gclib.cs.

9.30.3.3 encoder_2

```
SL encoder_2
```

Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1805 of file gclib.cs.

9.30.3.4 encoder_3

```
SL encoder_3
```

Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1806 of file gclib.cs.

9.30.3.5 error_code

`UB error_code`
Error code.
Definition at line 1774 of file gclib.cs.

9.30.3.6 general_status

`UB general_status`
General status.
Definition at line 1775 of file gclib.cs.

9.30.3.7 header_0

`UB header_0`
1st Byte of Header.
Definition at line 1768 of file gclib.cs.

9.30.3.8 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 1769 of file gclib.cs.

9.30.3.9 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 1770 of file gclib.cs.

9.30.3.10 header_3

`UB header_3`
4th Byte of Header.
Definition at line 1771 of file gclib.cs.

9.30.3.11 input_analog_0

`UW input_analog_0`
Analog input 0.
Definition at line 1786 of file gclib.cs.

9.30.3.12 input_analog_1

`UW input_analog_1`
Analog input 1.
Definition at line 1787 of file gclib.cs.

9.30.3.13 input_analog_2

`UW input_analog_2`
Analog input 2.
Definition at line 1788 of file gclib.cs.

9.30.3.14 input_analog_3

`UW input_analog_3`
Analog input 3.
Definition at line 1789 of file gclib.cs.

9.30.3.15 input_analog_4

`UW input_analog_4`
Analog input 4.
Definition at line 1790 of file gclib.cs.

9.30.3.16 input_analog_5

`UW input_analog_5`
Analog input 5.
Definition at line 1791 of file gclib.cs.

9.30.3.17 input_analog_6

`UW input_analog_6`
Analog input 6.
Definition at line 1792 of file gclib.cs.

9.30.3.18 input_analog_7

`UW input_analog_7`
Analog input 7.
Definition at line 1793 of file gclib.cs.

9.30.3.19 input_bank_0

`UW input_bank_0`
Digital inputs 0-15;.
Definition at line 1797 of file gclib.cs.

9.30.3.20 output_analog_0

`UW output_analog_0`
Analog output 0.
Definition at line 1777 of file gclib.cs.

9.30.3.21 output_analog_1

`UW output_analog_1`
Analog output 1.
Definition at line 1778 of file gclib.cs.

9.30.3.22 output_analog_2

`UW output_analog_2`
Analog output 2.
Definition at line 1779 of file gclib.cs.

9.30.3.23 output_analog_3

`UW output_analog_3`
Analog output 3.
Definition at line 1780 of file gclib.cs.

9.30.3.24 output_analog_4

`UW output_analog_4`
Analog output 4.
Definition at line 1781 of file gclib.cs.

9.30.3.25 output_analog_5

`UW output_analog_5`
Analog output 5.
Definition at line 1782 of file gclib.cs.

9.30.3.26 output_analog_6

`UW output_analog_6`
Analog output 6.
Definition at line 1783 of file gclib.cs.

9.30.3.27 output_analog_7

`UW output_analog_7`
Analog output 7.
Definition at line 1784 of file gclib.cs.

9.30.3.28 output_bank_0

`UW output_bank_0`
Digital outputs 0-15;.
Definition at line 1795 of file gclib.cs.

9.30.3.29 pulse_count_0

`UL pulse_count_0`
Pulse counter (see PC).
Definition at line 1799 of file gclib.cs.

9.30.3.30 sample_number

`UW sample_number`
Sample number.
Definition at line 1773 of file gclib.cs.

9.30.3.31 zc_variable

`SL zc_variable`
ZC User-defined variable (see ZC).
Definition at line 1800 of file gclib.cs.

9.30.3.32 zd_variable

`SL zd_variable`
ZD User-defined variable (see ZD).
Definition at line 1801 of file gclib.cs.
The documentation for this struct was generated from the following file:

- gclib.cs

9.31 GDataRecord47000_ENC Struct Reference

Data record struct for RIO-471xx and RIO-472xx PLCs. Includes encoder fields.
`#include <gclib_record.h>`

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.
- UB header_3

  4th Byte of Header.
- UW sample_number

  Sample number.
- UB error_code

  Error code.
- UB general_status

  General status.
- UW output_analog_0

  Analog output 0.
- UW output_analog_1

  Analog output 1.
- UW output_analog_2

  Analog output 2.
- UW output_analog_3

  Analog output 3.
- UW output_analog_4

  Analog output 4.
- UW output_analog_5

  Analog output 5.
- UW output_analog_6

  Analog output 6.
- UW output_analog_7

  Analog output 7.
- UW input_analog_0

  Analog input 0.
- UW input_analog_1

  Analog input 1.
- UW input_analog_2

  Analog input 2.
- UW input_analog_3

  Analog input 3.
- UW input_analog_4

  Analog input 4.
- UW input_analog_5

  Analog input 5.
- UW input_analog_6

  Analog input 6.
- UW input_analog_7

  Analog input 7.
- UW output_bank_0

  Digital outputs 0-15;.
- UW input_bank_0

Digital inputs 0-15;.

- UL pulse_count_0

  Pulse counter (see PC).
- SL zc_variable

  ZC User-defined variable (see ZC).
- SL zd_variable

  ZD User-defined variable (see ZD).
- SL encoder_0

  Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.
- SL encoder_1

  Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.
- SL encoder_2

  Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.
- SL encoder_3

  Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.

9.31.1 Detailed Description

Data record struct for RIO-471xx and RIO-472xx PLCs. Includes encoder fields.
Definition at line 870 of file gclib_record.h.

9.31.2 Field Documentation

9.31.2.1 encoder_0

`SL encoder_0`
Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 910 of file gclib_record.h.

9.31.2.2 encoder_1

`SL encoder_1`
Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 911 of file gclib_record.h.

9.31.2.3 encoder_2

`SL encoder_2`
Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 912 of file gclib_record.h.

9.31.2.4 encoder_3

`SL encoder_3`
Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 913 of file gclib_record.h.

9.31.2.5 error_code

`UB error_code`
Error code.
Definition at line 881 of file gclib_record.h.

9.31.2.6 general_status

`UB general_status`
General status.
Definition at line 882 of file gclib_record.h.

9.31.2.7 header_0

`UB header_0`
1st Byte of Header.
Definition at line 875 of file gclib_record.h.

9.31.2.8 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 876 of file gclib_record.h.

9.31.2.9 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 877 of file gclib_record.h.

9.31.2.10 header_3

`UB header_3`
4th Byte of Header.
Definition at line 878 of file gclib_record.h.

9.31.2.11 input_analog_0

`UW input_analog_0`
Analog input 0.
Definition at line 893 of file gclib_record.h.

9.31.2.12 input_analog_1

`UW input_analog_1`
Analog input 1.
Definition at line 894 of file gclib_record.h.

9.31.2.13 input_analog_2

`UW input_analog_2`
Analog input 2.
Definition at line 895 of file gclib_record.h.

9.31.2.14 input_analog_3

`UW input_analog_3`
Analog input 3.
Definition at line 896 of file gclib_record.h.

9.31.2.15 input_analog_4

`UW input_analog_4`
Analog input 4.
Definition at line 897 of file gclib_record.h.

9.31.2.16 input_analog_5

`UW input_analog_5`
Analog input 5.
Definition at line 898 of file gclib_record.h.

9.31.2.17 input_analog_6

`UW input_analog_6`
Analog input 6.
Definition at line 899 of file gclib_record.h.

9.31.2.18 input_analog_7

`UW input_analog_7`
Analog input 7.
Definition at line 900 of file gclib_record.h.

9.31.2.19 input_bank_0

`UW input_bank_0`
Digital inputs 0-15;.
Definition at line 904 of file gclib_record.h.

9.31.2.20 output_analog_0

`UW output_analog_0`
Analog output 0.
Definition at line 884 of file gclib_record.h.

9.31.2.21 output_analog_1

`UW output_analog_1`
Analog output 1.
Definition at line 885 of file gclib_record.h.

9.31.2.22 output_analog_2

`UW output_analog_2`
Analog output 2.
Definition at line 886 of file gclib_record.h.

9.31.2.23 output_analog_3

`UW output_analog_3`
Analog output 3.
Definition at line 887 of file gclib_record.h.

9.31.2.24 output_analog_4

`UW output_analog_4`
Analog output 4.
Definition at line 888 of file gclib_record.h.

9.31.2.25 output_analog_5

`UW output_analog_5`
Analog output 5.
Definition at line 889 of file gclib_record.h.

9.31.2.26 output_analog_6

`UW output_analog_6`
Analog output 6.
Definition at line 890 of file gclib_record.h.

9.31.2.27 output_analog_7

`UW output_analog_7`
Analog output 7.
Definition at line 891 of file gclib_record.h.

9.31.2.28 output_bank_0

`UW output_bank_0`
Digital outputs 0-15;.
Definition at line 902 of file gclib_record.h.

9.31.2.29 pulse_count_0

`UL pulse_count_0`
Pulse counter (see PC).
Definition at line 906 of file gclib_record.h.

9.31.2.30 sample_number

`UW sample_number`
Sample number.
Definition at line 880 of file gclib_record.h.

9.31.2.31 zc_variable

`SL zc_variable`
ZC User-defined variable (see ZC).
Definition at line 907 of file gclib_record.h.

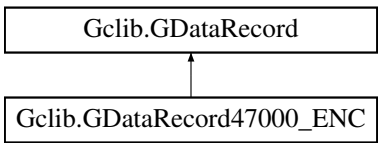
9.31.2.32 zd_variable

`SL zd_variable`
ZD User-defined variable (see ZD).
Definition at line 908 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

9.32 Gclib.GDataRecord47000_ENC Struct Reference

Inheritance diagram for Gclib.GDataRecord47000_ENC:



Public Member Functions

- Pulse **counter** (see PC). ∗/public SL zc_variable
  /∗48-51
- ZC User defined **variable** (see ZC). ∗/public SL zd_variable
  /∗52-55
- ZD User defined **variable** (see ZD). ∗/public SL encoder_0
  /∗56-59
- Encoder channel.Data only valid for parts with or SSI.∗ byte_array ()

Data Fields

- UB header_0
  /*00
- byte of Header.*UB header_1
  /*01
- byte of Header.*UB header_2
  /*02
- byte of Header.*UB header_3
  /*03
- byte of Header.*UW sample_number
  /*04-05
- Sample number.*UB error_code
  /*06
- Error code.*UB general_status
  /*07
- General status.*UW output_analog_0
  /*08-09
- Analog output.*UW output_analog_1
  /*10-11
- Analog output.*UW output_analog_2
  /*12-13
- Analog output.*UW output_analog_3
  /*14-15
- Analog output.*UW output_analog_4
  /*16-17
- Analog output.*UW output_analog_5
  /*18-19
- Analog output.*UW output_analog_6
  /*20-21
- Analog output.*UW output_analog_7
  /*22-23
- Analog output.*UW input_analog_0
  /*24-25
- Analog input.*UW input_analog_1
  /*26-27
- Analog input.*UW input_analog_2
  /*28-29
- Analog input.*UW input_analog_3
  /*30-31
- Analog input.*UW input_analog_4
  /*32-33
- Analog input.*UW input_analog_5
  /*34-35
- Analog input.*UW input_analog_6
  /*36-37
- Analog input.*UW input_analog_7
  /*38-39
- Analog input.*UW output_bank_0
  /*40-41
- Digital outputs

- ∗UW input_bank_0

  /∗42-43
- Digital inputs
- ∗UL pulse_count_0

  /∗44-47
- Encoder channel.Data only valid for parts with BISS
- Encoder channel.Data only valid for parts with QUAD
- Encoder channel.Data only valid for parts with or SSI.∗SL encoder_1

  /∗60-63
- Encoder channel.Data only valid for parts with or SSI.∗SL encoder_2

  /∗64-67
- Encoder channel.Data only valid for parts with or SSI.∗SL encoder_3

  /∗68-71

9.32.1 Detailed Description

Definition at line 1193 of file gclib.vb.

9.32.2 Member Function Documentation

9.32.2.1 byte_array()

```
Encoder channel.Data only valid for parts with or SSI.* byte_array ( )
```

Implements GDataRecord.byte_array
Implements Gclib.GDataRecord.

9.32.3 Field Documentation

9.32.3.1 BISS

```
Encoder channel Data only valid for parts with BISS
```

Definition at line 1224 of file gclib.vb.

9.32.3.2 encoder_1

```
Encoder channel.Data only valid for parts with or SSI.* SL encoder_1
```

/∗60-63
Definition at line 1225 of file gclib.vb.

9.32.3.3 encoder_2

```
Encoder channel.Data only valid for parts with or SSI.* SL encoder_2
```

/∗64-67
Definition at line 1226 of file gclib.vb.

9.32.3.4 encoder_3

```
Encoder channel.Data only valid for parts with or SSI.* SL encoder_3
```

/∗68-71
Definition at line 1227 of file gclib.vb.

9.32.3.5 error_code

```
Sample number.* UB error_code
```

/∗06
Definition at line 1201 of file gclib.vb.

9.32.3.6 general_status

```
Error code.* UB general_status
```
/*07
Definition at line 1202 of file gclib.vb.

9.32.3.7 header_0

```
UB header_0
```
/*00
Definition at line 1196 of file gclib.vb.

9.32.3.8 header_1

```
byte of Header.* UB header_1
```
/*01
Definition at line 1197 of file gclib.vb.

9.32.3.9 header_2

```
byte of Header.* UB header_2
```
/*02
Definition at line 1198 of file gclib.vb.

9.32.3.10 header_3

```
byte of Header.* UB header_3
```
/*03
Definition at line 1199 of file gclib.vb.

9.32.3.11 input_analog_0

```
Analog output.* UW input_analog_0
```
/*24-25
Definition at line 1211 of file gclib.vb.

9.32.3.12 input_analog_1

```
Analog input.* UW input_analog_1
```
/*26-27
Definition at line 1212 of file gclib.vb.

9.32.3.13 input_analog_2

```
Analog input.* UW input_analog_2
```
/*28-29
Definition at line 1213 of file gclib.vb.

9.32.3.14 input_analog_3

```
Analog input.* UW input_analog_3
```
/*30-31
Definition at line 1214 of file gclib.vb.

9.32.3.15 input_analog_4

```
Analog input.* UW input_analog_4
```
/*32-33
Definition at line 1215 of file gclib.vb.

9.32.3.16 input_analog_5

```
Analog input.* UW input_analog_5
```
/*34-35
Definition at line 1216 of file gclib.vb.

9.32.3.17 input_analog_6

```
Analog input.* UW input_analog_6
```
/*36-37
Definition at line 1217 of file gclib.vb.

9.32.3.18 input_analog_7

```
Analog input.* UW input_analog_7
```
/*38-39
Definition at line 1218 of file gclib.vb.

9.32.3.19 input_bank_0

```
* UW input_bank_0
```
/*42-43
Definition at line 1220 of file gclib.vb.

9.32.3.20 inputs

```
Digital inputs
```
Definition at line 1220 of file gclib.vb.

9.32.3.21 output_analog_0

```
General status.* UW output_analog_0
```
/*08-09
Definition at line 1203 of file gclib.vb.

9.32.3.22 output_analog_1

```
Analog output.* UW output_analog_1
```
/*10-11
Definition at line 1204 of file gclib.vb.

9.32.3.23 output_analog_2

```
Analog output.* UW output_analog_2
```
/*12-13
Definition at line 1205 of file gclib.vb.

9.32.3.24 output_analog_3

```
Analog output.* UW output_analog_3
```
/*14-15
Definition at line 1206 of file gclib.vb.

9.32.3.25 output_analog_4

```
Analog output.* UW output_analog_4
```
/*16-17
Definition at line 1207 of file gclib.vb.

9.32.3.26 output_analog_5

```
Analog output.* UW output_analog_5
```
/*18-19
Definition at line 1208 of file gclib.vb.

9.32.3.27 output_analog_6

```
Analog output.* UW output_analog_6
```
/*20-21
Definition at line 1209 of file gclib.vb.

9.32.3.28 output_analog_7

```
Analog output.* UW output_analog_7
```
/*22-23
Definition at line 1210 of file gclib.vb.

9.32.3.29 output_bank_0

```
Analog input.* UW output_bank_0
```
/*40-41
Definition at line 1219 of file gclib.vb.

9.32.3.30 outputs

```
Digital outputs
```
Definition at line 1219 of file gclib.vb.

9.32.3.31 pulse_count_0

```
* UL pulse_count_0
```
/*44-47
Definition at line 1221 of file gclib.vb.

9.32.3.32 QUAD

```
Encoder channel Data only valid for parts with QUAD
```
Definition at line 1224 of file gclib.vb.

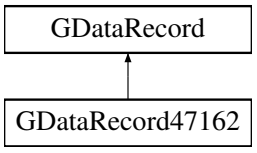
9.32.3.33 sample_number

```
byte of Header.* UW sample_number
```
/*04-05
Definition at line 1200 of file gclib.vb.
The documentation for this struct was generated from the following file:

- gclib.vb

9.33 GDataRecord47162 Struct Reference

Data record struct for RIO-47162.
Inheritance diagram for GDataRecord47162:



Public Member Functions

- byte[] byte_array ()

  Returns the data record as a byte array and allows for access to individual bytes.

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.
- UB header_3

  4th Byte of Header.
- UW sample_number

  Sample number.
- UB error_code

  Error code.
- UB general_status

  General status.
- UW output_analog_0

  Analog output 0.
- UW output_analog_1

  Analog output 1.
- UW output_analog_2

  Analog output 2.
- UW output_analog_3

  Analog output 3.
- UW output_analog_4

  Analog output 4.
- UW output_analog_5

  Analog output 5.
- UW output_analog_6

  Analog output 6.
- UW output_analog_7

  Analog output 7.
- UW input_analog_0

  Analog input 0.
- UW input_analog_1

  Analog input 1.
- UW input_analog_2

  Analog input 2.
- UW input_analog_3

  Analog input 3.
- UW input_analog_4

  Analog input 4.
- UW input_analog_5

  Analog input 5.
- UW input_analog_6

  Analog input 6.
- UW input_analog_7

Analog input 7.

- UB output_byte_0

  Digital outputs 0-7.
- UB output_byte_1

  Digital outputs 8-15.
- UB output_byte_2

  Digital outputs 16-23.
- UB input_byte_0

  Digital inputs 0-7.
- UB input_byte_1

  Digital inputs 8-15.
- UB input_byte_2

  Digital inputs 16-23.
- UB input_byte_3

  Digital inputs 24-31.
- UB input_byte_4

  Digital inputs 32-39.
- UL pulse_count_0

  Pulse counter (see PC).
- SL zc_variable

  ZC User-defined variable (see ZC).
- SL zd_variable

  ZD User-defined variable (see ZD).
- SL encoder_0

  Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.
- SL encoder_1

  Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.
- SL encoder_2

  Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.
- SL encoder_3

  Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.

9.33.1 Detailed Description

Data record struct for RIO-47162.
Definition at line 1917 of file gclib.cs.

9.33.2 Member Function Documentation

9.33.2.1 byte_array()

```
byte[] byte_array ( )  [inline]
```

Returns the data record as a byte array and allows for access to individual bytes.
Implements GDataRecord.
Definition at line 1919 of file gclib.cs.

9.33.3 Field Documentation

9.33.3.1 encoder_0

```
SL encoder_0
```

Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1963 of file gclib.cs.

9.33.3.2 encoder_1

`SL encoder_1`
Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1964 of file gclib.cs.

9.33.3.3 encoder_2

`SL encoder_2`
Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1965 of file gclib.cs.

9.33.3.4 encoder_3

`SL encoder_3`
Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1966 of file gclib.cs.

9.33.3.5 error_code

`UB error_code`
Error code.
Definition at line 1928 of file gclib.cs.

9.33.3.6 general_status

`UB general_status`
General status.
Definition at line 1929 of file gclib.cs.

9.33.3.7 header_0

`UB header_0`
1st Byte of Header.
Definition at line 1922 of file gclib.cs.

9.33.3.8 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 1923 of file gclib.cs.

9.33.3.9 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 1924 of file gclib.cs.

9.33.3.10 header_3

`UB header_3`
4th Byte of Header.
Definition at line 1925 of file gclib.cs.

9.33.3.11 input_analog_0

`UW input_analog_0`
Analog input 0.
Definition at line 1940 of file gclib.cs.

9.33.3.12 input_analog_1

`UW input_analog_1`
Analog input 1.
Definition at line 1941 of file gclib.cs.

9.33.3.13 input_analog_2

`UW input_analog_2`
Analog input 2.
Definition at line 1942 of file gclib.cs.

9.33.3.14 input_analog_3

`UW input_analog_3`
Analog input 3.
Definition at line 1943 of file gclib.cs.

9.33.3.15 input_analog_4

`UW input_analog_4`
Analog input 4.
Definition at line 1944 of file gclib.cs.

9.33.3.16 input_analog_5

`UW input_analog_5`
Analog input 5.
Definition at line 1945 of file gclib.cs.

9.33.3.17 input_analog_6

`UW input_analog_6`
Analog input 6.
Definition at line 1946 of file gclib.cs.

9.33.3.18 input_analog_7

`UW input_analog_7`
Analog input 7.
Definition at line 1947 of file gclib.cs.

9.33.3.19 input_byte_0

`UB input_byte_0`
Digital inputs 0-7.
Definition at line 1953 of file gclib.cs.

9.33.3.20 input_byte_1

`UB input_byte_1`
Digital inputs 8-15.
Definition at line 1954 of file gclib.cs.

9.33.3.21 input_byte_2

`UB input_byte_2`
Digital inputs 16-23.
Definition at line 1955 of file gclib.cs.

9.33.3.22 input_byte_3

`UB input_byte_3`
Digital inputs 24-31.
Definition at line 1956 of file gclib.cs.

9.33.3.23 input_byte_4

`UB input_byte_4`
Digital inputs 32-39.
Definition at line 1957 of file gclib.cs.

9.33.3.24 output_analog_0

`UW output_analog_0`
Analog output 0.
Definition at line 1931 of file gclib.cs.

9.33.3.25 output_analog_1

`UW output_analog_1`
Analog output 1.
Definition at line 1932 of file gclib.cs.

9.33.3.26 output_analog_2

`UW output_analog_2`
Analog output 2.
Definition at line 1933 of file gclib.cs.

9.33.3.27 output_analog_3

`UW output_analog_3`
Analog output 3.
Definition at line 1934 of file gclib.cs.

9.33.3.28 output_analog_4

`UW output_analog_4`
Analog output 4.
Definition at line 1935 of file gclib.cs.

9.33.3.29 output_analog_5

`UW output_analog_5`
Analog output 5.
Definition at line 1936 of file gclib.cs.

9.33.3.30 output_analog_6

`UW output_analog_6`
Analog output 6.
Definition at line 1937 of file gclib.cs.

9.33.3.31 output_analog_7

`UW output_analog_7`
Analog output 7.
Definition at line 1938 of file gclib.cs.

9.33.3.32 output_byte_0

`UB output_byte_0`
Digital outputs 0-7.
Definition at line 1949 of file gclib.cs.

9.33.3.33 output_byte_1

`UB output_byte_1`
Digital outputs 8-15.
Definition at line 1950 of file gclib.cs.

9.33.3.34 output_byte_2

`UB output_byte_2`
Digital outputs 16-23.
Definition at line 1951 of file gclib.cs.

9.33.3.35 pulse_count_0

`UL pulse_count_0`
Pulse counter (see PC).
Definition at line 1959 of file gclib.cs.

9.33.3.36 sample_number

`UW sample_number`
Sample number.
Definition at line 1927 of file gclib.cs.

9.33.3.37 zc_variable

`SL zc_variable`
ZC User-defined variable (see ZC).
Definition at line 1960 of file gclib.cs.

9.33.3.38 zd_variable

`SL zd_variable`
ZD User-defined variable (see ZD).
Definition at line 1961 of file gclib.cs.
The documentation for this struct was generated from the following file:

- gclib.cs

9.34 GDataRecord47162 Struct Reference

Data record struct for RIO-47162.
`#include <gclib_record.h>`

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.

- UB header_3

  4th Byte of Header.
- UW sample_number

  Sample number.
- UB error_code

  Error code.
- UB general_status

  General status.
- UW output_analog_0

  Analog output 0.
- UW output_analog_1

  Analog output 1.
- UW output_analog_2

  Analog output 2.
- UW output_analog_3

  Analog output 3.
- UW output_analog_4

  Analog output 4.
- UW output_analog_5

  Analog output 5.
- UW output_analog_6

  Analog output 6.
- UW output_analog_7

  Analog output 7.
- UW input_analog_0

  Analog input 0.
- UW input_analog_1

  Analog input 1.
- UW input_analog_2

  Analog input 2.
- UW input_analog_3

  Analog input 3.
- UW input_analog_4

  Analog input 4.
- UW input_analog_5

  Analog input 5.
- UW input_analog_6

  Analog input 6.
- UW input_analog_7

  Analog input 7.
- UB output_byte_0

  Digital outputs 0-7.
- UB output_byte_1

  Digital outputs 8-15.
- UB output_byte_2

  Digital outputs 16-23.
- UB input_byte_0

  Digital inputs 0-7.
- UB input_byte_1

  Digital inputs 8-15.
- UB input_byte_2

Digital inputs 16-23.

- UB input_byte_3

  Digital inputs 24-31.

- UB input_byte_4

  Digital inputs 32-39.

- UL pulse_count_0

  Pulse counter (see PC).

- SL zc_variable

  ZC User-defined variable (see ZC).

- SL zd_variable

  ZD User-defined variable (see ZD).

- SL encoder_0

  Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.

- SL encoder_1

  Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.

- SL encoder_2

  Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.

- SL encoder_3

  Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.

9.34.1 Detailed Description

Data record struct for RIO-47162.
Definition at line 1019 of file gclib_record.h.

9.34.2 Field Documentation

9.34.2.1 encoder_0

`SL encoder_0`
Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1064 of file gclib_record.h.

9.34.2.2 encoder_1

`SL encoder_1`
Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1065 of file gclib_record.h.

9.34.2.3 encoder_2

`SL encoder_2`
Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1066 of file gclib_record.h.

9.34.2.4 encoder_3

`SL encoder_3`
Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1067 of file gclib_record.h.

9.34.2.5 error_code

`UB error_code`
Error code.
Definition at line 1029 of file gclib_record.h.

9.34.2.6 general_status

`UB general_status`
General status.
Definition at line 1030 of file gclib_record.h.

9.34.2.7 header_0

`UB header_0`
1st Byte of Header.
Definition at line 1023 of file gclib_record.h.

9.34.2.8 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 1024 of file gclib_record.h.

9.34.2.9 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 1025 of file gclib_record.h.

9.34.2.10 header_3

`UB header_3`
4th Byte of Header.
Definition at line 1026 of file gclib_record.h.

9.34.2.11 input_analog_0

`UW input_analog_0`
Analog input 0.
Definition at line 1041 of file gclib_record.h.

9.34.2.12 input_analog_1

`UW input_analog_1`
Analog input 1.
Definition at line 1042 of file gclib_record.h.

9.34.2.13 input_analog_2

`UW input_analog_2`
Analog input 2.
Definition at line 1043 of file gclib_record.h.

9.34.2.14 input_analog_3

`UW input_analog_3`
Analog input 3.
Definition at line 1044 of file gclib_record.h.

9.34.2.15 input_analog_4

`UW input_analog_4`
Analog input 4.
Definition at line 1045 of file gclib_record.h.

9.34.2.16 input_analog_5

`UW input_analog_5`
Analog input 5.
Definition at line 1046 of file gclib_record.h.

9.34.2.17 input_analog_6

`UW input_analog_6`
Analog input 6.
Definition at line 1047 of file gclib_record.h.

9.34.2.18 input_analog_7

`UW input_analog_7`
Analog input 7.
Definition at line 1048 of file gclib_record.h.

9.34.2.19 input_byte_0

`UB input_byte_0`
Digital inputs 0-7.
Definition at line 1054 of file gclib_record.h.

9.34.2.20 input_byte_1

`UB input_byte_1`
Digital inputs 8-15.
Definition at line 1055 of file gclib_record.h.

9.34.2.21 input_byte_2

`UB input_byte_2`
Digital inputs 16-23.
Definition at line 1056 of file gclib_record.h.

9.34.2.22 input_byte_3

`UB input_byte_3`
Digital inputs 24-31.
Definition at line 1057 of file gclib_record.h.

9.34.2.23 input_byte_4

`UB input_byte_4`
Digital inputs 32-39.
Definition at line 1058 of file gclib_record.h.

9.34.2.24 output_analog_0

`UW output_analog_0`
Analog output 0.
Definition at line 1032 of file gclib_record.h.

9.34.2.25 output_analog_1

`UW output_analog_1`
Analog output 1.
Definition at line 1033 of file gclib_record.h.

9.34.2.26 output_analog_2

`UW output_analog_2`
Analog output 2.
Definition at line 1034 of file gclib_record.h.

9.34.2.27 output_analog_3

`UW output_analog_3`
Analog output 3.
Definition at line 1035 of file gclib_record.h.

9.34.2.28 output_analog_4

`UW output_analog_4`
Analog output 4.
Definition at line 1036 of file gclib_record.h.

9.34.2.29 output_analog_5

`UW output_analog_5`
Analog output 5.
Definition at line 1037 of file gclib_record.h.

9.34.2.30 output_analog_6

`UW output_analog_6`
Analog output 6.
Definition at line 1038 of file gclib_record.h.

9.34.2.31 output_analog_7

`UW output_analog_7`
Analog output 7.
Definition at line 1039 of file gclib_record.h.

9.34.2.32 output_byte_0

`UB output_byte_0`
Digital outputs 0-7.
Definition at line 1050 of file gclib_record.h.

9.34.2.33 output_byte_1

`UB output_byte_1`
Digital outputs 8-15.
Definition at line 1051 of file gclib_record.h.

9.34.2.34 output_byte_2

`UB output_byte_2`
Digital outputs 16-23.
Definition at line 1052 of file gclib_record.h.

9.34.2.35 pulse_count_0

`UL pulse_count_0`
Pulse counter (see PC).
Definition at line 1060 of file gclib_record.h.

9.34.2.36 sample_number

`UW sample_number`
Sample number.
Definition at line 1028 of file gclib_record.h.

9.34.2.37 zc_variable

`SL zc_variable`
ZC User-defined variable (see ZC).
Definition at line 1061 of file gclib_record.h.

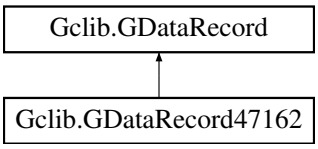
9.34.2.38 zd_variable

`SL zd_variable`
ZD User-defined variable (see ZD).
Definition at line 1062 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

9.35 Gclib.GDataRecord47162 Struct Reference

Inheritance diagram for Gclib.GDataRecord47162:



Public Member Functions

- Pulse **counter** (see PC). ∗/public SL zc_variable
  /∗52-55
- ZC User defined **variable** (see ZC). ∗/public SL zd_variable
  /∗56-59
- ZD User defined **variable** (see ZD). ∗/public SL encoder_0
  /∗60-63
- Encoder channel.Data only valid for parts with or SSI.∗ byte_array ()

Data Fields

- UB header_0
  /∗00
- byte of Header.∗UB header_1
  /∗01
- byte of Header.∗UB header_2
  /∗02
- byte of Header.∗UB header_3
  /∗03
- byte of Header.∗UW sample_number
  /∗04-05
- Sample number.∗UB error_code
  /∗06

- Error code.∗UB general_status
  /∗07
- General status.∗UW output_analog_0
  /∗08-09
- Analog output.∗UW output_analog_1
  /∗10-11
- Analog output.∗UW output_analog_2
  /∗12-13
- Analog output.∗UW output_analog_3
  /∗14-15
- Analog output.∗UW output_analog_4
  /∗16-17
- Analog output.∗UW output_analog_5
  /∗18-19
- Analog output.∗UW output_analog_6
  /∗20-21
- Analog output.∗UW output_analog_7
  /∗22-23
- Analog output.∗UW input_analog_0
  /∗24-25
- Analog input.∗UW input_analog_1
  /∗26-27
- Analog input.∗UW input_analog_2
  /∗28-29
- Analog input.∗UW input_analog_3
  /∗30-31
- Analog input.∗UW input_analog_4
  /∗32-33
- Analog input.∗UW input_analog_5
  /∗34-35
- Analog input.∗UW input_analog_6
  /∗36-37
- Analog input.∗UW input_analog_7
  /∗38-39
- Analog input.∗UB output_byte_0
  /∗40
- Digital outputs.∗UB output_byte_1
  /∗41
- Digital outputs.∗UB output_byte_2
  /∗42
- Digital outputs.∗UB input_byte_0
  /∗43
- Digital inputs.∗UB input_byte_1
  /∗44
- Digital inputs.∗UB input_byte_2
  /∗45
- Digital inputs.∗UB input_byte_3
  /∗46
- Digital inputs.∗UB input_byte_4
  /∗47
- Digital inputs.∗UL pulse_count_0

/∗48-51

- Encoder channel.Data only valid for parts with BISS
- Encoder channel.Data only valid for parts with QUAD
- Encoder channel.Data only valid for parts with or SSI.∗SL encoder_1

  /∗64-67
- Encoder channel.Data only valid for parts with or SSI.∗SL encoder_2

  /∗68-71
- Encoder channel.Data only valid for parts with or SSI.∗SL encoder_3

  /∗72-75

9.35.1 Detailed Description

Definition at line 1308 of file gclib.vb.

9.35.2 Member Function Documentation

9.35.2.1 byte_array()

```
Encoder channel.Data only valid for parts with or SSI.* byte_array ( )
```

Implements GDataRecord.byte_array
Implements Gclib.GDataRecord.

9.35.3 Field Documentation

9.35.3.1 BISS

```
Encoder channel Data only valid for parts with BISS
```

Definition at line 1345 of file gclib.vb.

9.35.3.2 encoder_1

```
Encoder channel.Data only valid for parts with or SSI.* SL encoder_1
```

/∗64-67
Definition at line 1346 of file gclib.vb.

9.35.3.3 encoder_2

```
Encoder channel.Data only valid for parts with or SSI.* SL encoder_2
```

/∗68-71
Definition at line 1347 of file gclib.vb.

9.35.3.4 encoder_3

```
Encoder channel.Data only valid for parts with or SSI.* SL encoder_3
```

/∗72-75
Definition at line 1348 of file gclib.vb.

9.35.3.5 error_code

```
Sample number.* UB error_code
```

/∗06
Definition at line 1316 of file gclib.vb.

9.35.3.6 general_status

```
Error code.* UB general_status
```

/∗07
Definition at line 1317 of file gclib.vb.

9.35.3.7 header_0

```
UB header_0
```
/*00
Definition at line 1311 of file gclib.vb.

9.35.3.8 header_1

```
byte of Header.* UB header_1
```
/*01
Definition at line 1312 of file gclib.vb.

9.35.3.9 header_2

```
byte of Header.* UB header_2
```
/*02
Definition at line 1313 of file gclib.vb.

9.35.3.10 header_3

```
byte of Header.* UB header_3
```
/*03
Definition at line 1314 of file gclib.vb.

9.35.3.11 input_analog_0

```
Analog output.* UW input_analog_0
```
/*24-25
Definition at line 1326 of file gclib.vb.

9.35.3.12 input_analog_1

```
Analog input.* UW input_analog_1
```
/*26-27
Definition at line 1327 of file gclib.vb.

9.35.3.13 input_analog_2

```
Analog input.* UW input_analog_2
```
/*28-29
Definition at line 1328 of file gclib.vb.

9.35.3.14 input_analog_3

```
Analog input.* UW input_analog_3
```
/*30-31
Definition at line 1329 of file gclib.vb.

9.35.3.15 input_analog_4

```
Analog input.* UW input_analog_4
```
/*32-33
Definition at line 1330 of file gclib.vb.

9.35.3.16 input_analog_5

```
Analog input.* UW input_analog_5
```
/*34-35
Definition at line 1331 of file gclib.vb.

9.35.3.17 input_analog_6

```
Analog input.* UW input_analog_6
```
/*36-37
Definition at line 1332 of file gclib.vb.

9.35.3.18 input_analog_7

```
Analog input.* UW input_analog_7
```
/*38-39
Definition at line 1333 of file gclib.vb.

9.35.3.19 input_byte_0

```
Digital outputs.* UB input_byte_0
```
/*43
Definition at line 1337 of file gclib.vb.

9.35.3.20 input_byte_1

```
Digital inputs.* UB input_byte_1
```
/*44
Definition at line 1338 of file gclib.vb.

9.35.3.21 input_byte_2

```
Digital inputs.* UB input_byte_2
```
/*45
Definition at line 1339 of file gclib.vb.

9.35.3.22 input_byte_3

```
Digital inputs.* UB input_byte_3
```
/*46
Definition at line 1340 of file gclib.vb.

9.35.3.23 input_byte_4

```
Digital inputs.* UB input_byte_4
```
/*47
Definition at line 1341 of file gclib.vb.

9.35.3.24 output_analog_0

```
General status.* UW output_analog_0
```
/*08-09
Definition at line 1318 of file gclib.vb.

9.35.3.25 output_analog_1

```
Analog output.* UW output_analog_1
```
/*10-11
Definition at line 1319 of file gclib.vb.

9.35.3.26 output_analog_2

```
Analog output.* UW output_analog_2
```
/*12-13
Definition at line 1320 of file gclib.vb.

9.35.3.27 output_analog_3

```
Analog output.* UW output_analog_3
```
/*14-15
Definition at line 1321 of file gclib.vb.

9.35.3.28 output_analog_4

```
Analog output.* UW output_analog_4
```
/*16-17
Definition at line 1322 of file gclib.vb.

9.35.3.29 output_analog_5

```
Analog output.* UW output_analog_5
```
/*18-19
Definition at line 1323 of file gclib.vb.

9.35.3.30 output_analog_6

```
Analog output.* UW output_analog_6
```
/*20-21
Definition at line 1324 of file gclib.vb.

9.35.3.31 output_analog_7

```
Analog output.* UW output_analog_7
```
/*22-23
Definition at line 1325 of file gclib.vb.

9.35.3.32 output_byte_0

```
Analog input.* UB output_byte_0
```
/*40
Definition at line 1334 of file gclib.vb.

9.35.3.33 output_byte_1

```
Digital outputs.* UB output_byte_1
```
/*41
Definition at line 1335 of file gclib.vb.

9.35.3.34 output_byte_2

```
Digital outputs.* UB output_byte_2
```
/*42
Definition at line 1336 of file gclib.vb.

9.35.3.35 pulse_count_0

```
Digital inputs.* UL pulse_count_0
```
/*48-51
Definition at line 1342 of file gclib.vb.

9.35.3.36 QUAD

```
Encoder channel Data only valid for parts with QUAD
```
Definition at line 1345 of file gclib.vb.

9.35.3.37 sample_number

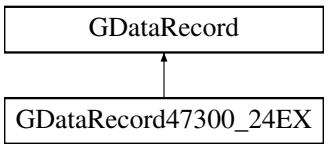
```
byte of Header.* UW sample_number
```

/*04-05
Definition at line 1315 of file gclib.vb.
The documentation for this struct was generated from the following file:

- gclib.vb

9.36 GDataRecord47300_24EX Struct Reference

Data record struct for RIO-47300 with 24EX I/O daughter board.
Inheritance diagram for GDataRecord47300_24EX:



Public Member Functions

- byte[] byte_array ()

  Returns the data record as a byte array and allows for access to individual bytes.

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.
- UB header_3

  4th Byte of Header.
- UW sample_number

  Sample number.
- UB error_code

  Error code.
- UB general_status

  General status.
- UW output_analog_0

  Analog output 0.
- UW output_analog_1

  Analog output 1.
- UW output_analog_2

  Analog output 2.
- UW output_analog_3

  Analog output 3.
- UW output_analog_4

  Analog output 4.
- UW output_analog_5

  Analog output 5.
- UW output_analog_6

Analog output 6.

- UW output_analog_7

  Analog output 7.
- UW input_analog_0

  Analog input 0.
- UW input_analog_1

  Analog input 1.
- UW input_analog_2

  Analog input 2.
- UW input_analog_3

  Analog input 3.
- UW input_analog_4

  Analog input 4.
- UW input_analog_5

  Analog input 5.
- UW input_analog_6

  Analog input 6.
- UW input_analog_7

  Analog input 7.
- UW output_bank_0

  Digital outputs 0-15.
- UW output_bank_1

  Digital outputs 16-23.
- UW input_bank_0

  Digital inputs 0-15.
- UW input_bank_1

  Digital inputs 16-23.
- UL pulse_count_0

  Pulse counter (see PC)8.
- SL zc_variable

  ZC User-defined variable (see ZC).
- SL zd_variable

  ZD User-defined variable (see ZD).
- UW output_bank_2

  Digital outputs 24-39. Data only valid for parts with 24EXOUT.
- UW output_back_3

  Digital outputs 40-47. Data only valid for parts with 24EXOUT.
- UW input_bank_2

  Digital inputs 24-39. Data only valid for parts with 24EXIN.
- UW input_bank_3

  Digital inputs 40-47. Data only valid for parts with 24EXIN.

9.36.1 Detailed Description

Data record struct for RIO-47300 with 24EX I/O daughter board.
Definition at line 1864 of file gclib.cs.

9.36.2 Member Function Documentation

9.36.2.1 byte_array()

```
byte[] byte_array ( )  [inline]
```

Returns the data record as a byte array and allows for access to individual bytes.
Implements GDataRecord.
Definition at line 1866 of file gclib.cs.

9.36.3 Field Documentation

9.36.3.1 error_code

`UB error_code`
Error code.
Definition at line 1876 of file gclib.cs.

9.36.3.2 general_status

`UB general_status`
General status.
Definition at line 1877 of file gclib.cs.

9.36.3.3 header_0

`UB header_0`
1st Byte of Header.
Definition at line 1870 of file gclib.cs.

9.36.3.4 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 1871 of file gclib.cs.

9.36.3.5 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 1872 of file gclib.cs.

9.36.3.6 header_3

`UB header_3`
4th Byte of Header.
Definition at line 1873 of file gclib.cs.

9.36.3.7 input_analog_0

`UW input_analog_0`
Analog input 0.
Definition at line 1888 of file gclib.cs.

9.36.3.8 input_analog_1

`UW input_analog_1`
Analog input 1.
Definition at line 1889 of file gclib.cs.

9.36.3.9 input_analog_2

`UW input_analog_2`
Analog input 2.
Definition at line 1890 of file gclib.cs.

9.36.3.10 input_analog_3

`UW input_analog_3`
Analog input 3.
Definition at line 1891 of file gclib.cs.

9.36.3.11 input_analog_4

`UW input_analog_4`
Analog input 4.
Definition at line 1892 of file gclib.cs.

9.36.3.12 input_analog_5

`UW input_analog_5`
Analog input 5.
Definition at line 1893 of file gclib.cs.

9.36.3.13 input_analog_6

`UW input_analog_6`
Analog input 6.
Definition at line 1894 of file gclib.cs.

9.36.3.14 input_analog_7

`UW input_analog_7`
Analog input 7.
Definition at line 1895 of file gclib.cs.

9.36.3.15 input_bank_0

`UW input_bank_0`
Digital inputs 0-15.
Definition at line 1900 of file gclib.cs.

9.36.3.16 input_bank_1

`UW input_bank_1`
Digital inputs 16-23.
Definition at line 1901 of file gclib.cs.

9.36.3.17 input_bank_2

`UW input_bank_2`
Digital inputs 24-39. Data only valid for parts with 24EXIN.
Definition at line 1910 of file gclib.cs.

9.36.3.18 input_bank_3

`UW input_bank_3`
Digital inputs 40-47. Data only valid for parts with 24EXIN.
Definition at line 1911 of file gclib.cs.

9.36.3.19 output_analog_0

`UW output_analog_0`
Analog output 0.
Definition at line 1879 of file gclib.cs.

9.36.3.20 output_analog_1

`UW output_analog_1`
Analog output 1.
Definition at line 1880 of file gclib.cs.

9.36.3.21 output_analog_2

`UW output_analog_2`
Analog output 2.
Definition at line 1881 of file gclib.cs.

9.36.3.22 output_analog_3

`UW output_analog_3`
Analog output 3.
Definition at line 1882 of file gclib.cs.

9.36.3.23 output_analog_4

`UW output_analog_4`
Analog output 4.
Definition at line 1883 of file gclib.cs.

9.36.3.24 output_analog_5

`UW output_analog_5`
Analog output 5.
Definition at line 1884 of file gclib.cs.

9.36.3.25 output_analog_6

`UW output_analog_6`
Analog output 6.
Definition at line 1885 of file gclib.cs.

9.36.3.26 output_analog_7

`UW output_analog_7`
Analog output 7.
Definition at line 1886 of file gclib.cs.

9.36.3.27 output_back_3

`UW output_back_3`
Digital outputs 40-47. Data only valid for parts with 24EXOUT.
Definition at line 1908 of file gclib.cs.

9.36.3.28 output_bank_0

`UW output_bank_0`
Digital outputs 0-15.
Definition at line 1897 of file gclib.cs.

9.36.3.29 output_bank_1

`UW output_bank_1`
Digital outputs 16-23.
Definition at line 1898 of file gclib.cs.

9.36.3.30 output_bank_2

`UW output_bank_2`
Digital outputs 24-39. Data only valid for parts with 24EXOUT.
Definition at line 1907 of file gclib.cs.

9.36.3.31 pulse_count_0

`UL pulse_count_0`
Pulse counter (see PC)8.
Definition at line 1903 of file gclib.cs.

9.36.3.32 sample_number

`UW sample_number`
Sample number.
Definition at line 1875 of file gclib.cs.

9.36.3.33 zc_variable

`SL zc_variable`
ZC User-defined variable (see ZC).
Definition at line 1904 of file gclib.cs.

9.36.3.34 zd_variable

`SL zd_variable`
ZD User-defined variable (see ZD).
Definition at line 1905 of file gclib.cs.
The documentation for this struct was generated from the following file:

- gclib.cs

9.37 GDataRecord47300_24EX Struct Reference

Data record struct for RIO-47300 with 24EX I/O daughter board.
`#include <gclib_record.h>`

Data Fields

- UB header_0
  1st Byte of Header.
- UB header_1
  2nd Byte of Header.
- UB header_2
  3rd Byte of Header.
- UB header_3
  4th Byte of Header.
- UW sample_number
  Sample number.
- UB error_code
  Error code.
- UB general_status
  General status.
- UW output_analog_0
  Analog output 0.
- UW output_analog_1
  Analog output 1.
- UW output_analog_2
  Analog output 2.
- UW output_analog_3
  Analog output 3.

- UW output_analog_4

  Analog output 4.
- UW output_analog_5

  Analog output 5.
- UW output_analog_6

  Analog output 6.
- UW output_analog_7

  Analog output 7.
- UW input_analog_0

  Analog input 0.
- UW input_analog_1

  Analog input 1.
- UW input_analog_2

  Analog input 2.
- UW input_analog_3

  Analog input 3.
- UW input_analog_4

  Analog input 4.
- UW input_analog_5

  Analog input 5.
- UW input_analog_6

  Analog input 6.
- UW input_analog_7

  Analog input 7.
- UW output_bank_0

  Digital outputs 0-15.
- UW output_bank_1

  Digital outputs 16-23.
- UW input_bank_0

  Digital inputs 0-15.
- UW input_bank_1

  Digital inputs 16-23.
- UL pulse_count_0

  Pulse counter (see PC)8.
- SL zc_variable

  ZC User-defined variable (see ZC).
- SL zd_variable

  ZD User-defined variable (see ZD).
- UW output_bank_2

  Digital outputs 24-39. Data only valid for parts with 24EXOUT.
- UW output_back_3

  Digital outputs 40-47. Data only valid for parts with 24EXOUT.
- UW input_bank_2

  Digital inputs 24-39. Data only valid for parts with 24EXIN.
- UW input_bank_3

  Digital inputs 40-47. Data only valid for parts with 24EXIN.

9.37.1 Detailed Description

Data record struct for RIO-47300 with 24EX I/O daughter board.
Definition at line 968 of file gclib_record.h.

9.37.2 Field Documentation

9.37.2.1 error_code

`UB error_code`
Error code.
Definition at line 979 of file gclib_record.h.

9.37.2.2 general_status

`UB general_status`
General status.
Definition at line 980 of file gclib_record.h.

9.37.2.3 header_0

`UB header_0`
1st Byte of Header.
Definition at line 973 of file gclib_record.h.

9.37.2.4 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 974 of file gclib_record.h.

9.37.2.5 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 975 of file gclib_record.h.

9.37.2.6 header_3

`UB header_3`
4th Byte of Header.
Definition at line 976 of file gclib_record.h.

9.37.2.7 input_analog_0

`UW input_analog_0`
Analog input 0.
Definition at line 991 of file gclib_record.h.

9.37.2.8 input_analog_1

`UW input_analog_1`
Analog input 1.
Definition at line 992 of file gclib_record.h.

9.37.2.9 input_analog_2

`UW input_analog_2`
Analog input 2.
Definition at line 993 of file gclib_record.h.

9.37.2.10 input_analog_3

`UW input_analog_3`
Analog input 3.
Definition at line 994 of file gclib_record.h.

9.37.2.11 input_analog_4

`UW input_analog_4`
Analog input 4.
Definition at line 995 of file gclib_record.h.

9.37.2.12 input_analog_5

`UW input_analog_5`
Analog input 5.
Definition at line 996 of file gclib_record.h.

9.37.2.13 input_analog_6

`UW input_analog_6`
Analog input 6.
Definition at line 997 of file gclib_record.h.

9.37.2.14 input_analog_7

`UW input_analog_7`
Analog input 7.
Definition at line 998 of file gclib_record.h.

9.37.2.15 input_bank_0

`UW input_bank_0`
Digital inputs 0-15.
Definition at line 1003 of file gclib_record.h.

9.37.2.16 input_bank_1

`UW input_bank_1`
Digital inputs 16-23.
Definition at line 1004 of file gclib_record.h.

9.37.2.17 input_bank_2

`UW input_bank_2`
Digital inputs 24-39. Data only valid for parts with 24EXIN.
Definition at line 1013 of file gclib_record.h.

9.37.2.18 input_bank_3

`UW input_bank_3`
Digital inputs 40-47. Data only valid for parts with 24EXIN.
Definition at line 1014 of file gclib_record.h.

9.37.2.19 output_analog_0

`UW output_analog_0`
Analog output 0.
Definition at line 982 of file gclib_record.h.

9.37.2.20 output_analog_1

`UW output_analog_1`
Analog output 1.
Definition at line 983 of file gclib_record.h.

9.37.2.21 output_analog_2

`UW output_analog_2`
Analog output 2.
Definition at line 984 of file gclib_record.h.

9.37.2.22 output_analog_3

`UW output_analog_3`
Analog output 3.
Definition at line 985 of file gclib_record.h.

9.37.2.23 output_analog_4

`UW output_analog_4`
Analog output 4.
Definition at line 986 of file gclib_record.h.

9.37.2.24 output_analog_5

`UW output_analog_5`
Analog output 5.
Definition at line 987 of file gclib_record.h.

9.37.2.25 output_analog_6

`UW output_analog_6`
Analog output 6.
Definition at line 988 of file gclib_record.h.

9.37.2.26 output_analog_7

`UW output_analog_7`
Analog output 7.
Definition at line 989 of file gclib_record.h.

9.37.2.27 output_back_3

`UW output_back_3`
Digital outputs 40-47. Data only valid for parts with 24EXOUT.
Definition at line 1011 of file gclib_record.h.

9.37.2.28 output_bank_0

`UW output_bank_0`
Digital outputs 0-15.
Definition at line 1000 of file gclib_record.h.

9.37.2.29 output_bank_1

`UW output_bank_1`
Digital outputs 16-23.
Definition at line 1001 of file gclib_record.h.

9.37.2.30 output_bank_2

`UW output_bank_2`
Digital outputs 24-39. Data only valid for parts with 24EXOUT.
Definition at line 1010 of file gclib_record.h.

9.37.2.31 pulse_count_0

`UL pulse_count_0`
Pulse counter (see PC)8.
Definition at line 1006 of file gclib_record.h.

9.37.2.32 sample_number

`UW sample_number`
Sample number.
Definition at line 978 of file gclib_record.h.

9.37.2.33 zc_variable

`SL zc_variable`
ZC User-defined variable (see ZC).
Definition at line 1007 of file gclib_record.h.

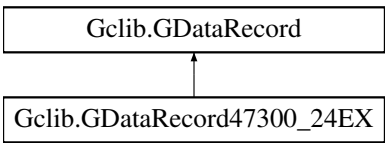
9.37.2.34 zd_variable

`SL zd_variable`
ZD User-defined variable (see ZD).
Definition at line 1008 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

9.38 Gclib.GDataRecord47300_24EX Struct Reference

Inheritance diagram for Gclib.GDataRecord47300_24EX:



Public Member Functions

- Pulse **counter** (see PC) 8. ∗/public SL zc_variable
  /∗52-55
- ZC User defined **variable** (see ZC). ∗/public SL zd_variable
  /∗56-59
- ZD User defined **variable** (see ZD). ∗/public UW output_bank_2
  /∗60-61
- Digital inputs.Data only valid for parts with.∗ byte_array ()

Data Fields

- UB header_0
  /∗00
- byte of Header.∗UB header_1
  /∗01
- byte of Header.∗UB header_2
  /∗02
- byte of Header.∗UB header_3

/*03

- byte of Header.*UW sample_number

/*04-05

- Sample number.*UB error_code

/*06

- Error code.*UB general_status

/*07

- General status.*UW output_analog_0

/*08-09

- Analog output.*UW output_analog_1

/*10-11

- Analog output.*UW output_analog_2

/*12-13

- Analog output.*UW output_analog_3

/*14-15

- Analog output.*UW output_analog_4

/*16-17

- Analog output.*UW output_analog_5

/*18-19

- Analog output.*UW output_analog_6

/*20-21

- Analog output.*UW output_analog_7

/*22-23

- Analog output.*UW input_analog_0

/*24-25

- Analog input.*UW input_analog_1

/*26-27

- Analog input.*UW input_analog_2

/*28-29

- Analog input.*UW input_analog_3

/*30-31

- Analog input.*UW input_analog_4

/*32-33

- Analog input.*UW input_analog_5

/*34-35

- Analog input.*UW input_analog_6

/*36-37

- Analog input.*UW input_analog_7

/*38-39

- Analog input.*UW output_bank_0

/*40-41

- Digital outputs.*UW output_bank_1

/*42-43

- Digital outputs.*UW input_bank_0

/*44-45

- Digital inputs.*UW input_bank_1

/*46-47

- Digital inputs.*UL pulse_count_0

/*48-51

- Digital outputs.Data only valid for parts with.*UW output_back_3

/*62-63

- Digital outputs.Data only valid for parts with.∗UW input_bank_2

  /∗64-65
- Digital inputs.Data only valid for parts with.∗UW input_bank_3

  /∗66-67

9.38.1 Detailed Description

Definition at line 1269 of file gclib.vb.

9.38.2 Member Function Documentation

9.38.2.1 byte_array()

```
Digital inputs.Data only valid for parts with.* byte_array ( )
```

Implements GDataRecord.byte_array
Implements Gclib.GDataRecord.

9.38.3 Field Documentation

9.38.3.1 error_code

```
Sample number.* UB error_code
```

/∗06
Definition at line 1277 of file gclib.vb.

9.38.3.2 general_status

```
Error code.* UB general_status
```

/∗07
Definition at line 1278 of file gclib.vb.

9.38.3.3 header_0

```
UB header_0
```

/∗00
Definition at line 1272 of file gclib.vb.

9.38.3.4 header_1

```
byte of Header.* UB header_1
```

/∗01
Definition at line 1273 of file gclib.vb.

9.38.3.5 header_2

```
byte of Header.* UB header_2
```

/∗02
Definition at line 1274 of file gclib.vb.

9.38.3.6 header_3

```
byte of Header.* UB header_3
```

/∗03
Definition at line 1275 of file gclib.vb.

9.38.3.7 input_analog_0

```
Analog output.* UW input_analog_0
```

/∗24-25
Definition at line 1287 of file gclib.vb.

9.38.3.8 input_analog_1

```
Analog input.* UW input_analog_1
```
/*26-27
Definition at line 1288 of file gclib.vb.

9.38.3.9 input_analog_2

```
Analog input.* UW input_analog_2
```
/*28-29
Definition at line 1289 of file gclib.vb.

9.38.3.10 input_analog_3

```
Analog input.* UW input_analog_3
```
/*30-31
Definition at line 1290 of file gclib.vb.

9.38.3.11 input_analog_4

```
Analog input.* UW input_analog_4
```
/*32-33
Definition at line 1291 of file gclib.vb.

9.38.3.12 input_analog_5

```
Analog input.* UW input_analog_5
```
/*34-35
Definition at line 1292 of file gclib.vb.

9.38.3.13 input_analog_6

```
Analog input.* UW input_analog_6
```
/*36-37
Definition at line 1293 of file gclib.vb.

9.38.3.14 input_analog_7

```
Analog input.* UW input_analog_7
```
/*38-39
Definition at line 1294 of file gclib.vb.

9.38.3.15 input_bank_0

```
Digital outputs.* UW input_bank_0
```
/*44-45
Definition at line 1297 of file gclib.vb.

9.38.3.16 input_bank_1

```
Digital inputs.* UW input_bank_1
```
/*46-47
Definition at line 1298 of file gclib.vb.

9.38.3.17 input_bank_2

```
Digital outputs.Data only valid for parts with.* UW input_bank_2
```
/*64-65
Definition at line 1304 of file gclib.vb.

9.38.3.18 input_bank_3

```
Digital inputs.Data only valid for parts with.* UW input_bank_3
```
/*66-67
Definition at line 1305 of file gclib.vb.

9.38.3.19 output_analog_0

```
General status.* UW output_analog_0
```
/*08-09
Definition at line 1279 of file gclib.vb.

9.38.3.20 output_analog_1

```
Analog output.* UW output_analog_1
```
/*10-11
Definition at line 1280 of file gclib.vb.

9.38.3.21 output_analog_2

```
Analog output.* UW output_analog_2
```
/*12-13
Definition at line 1281 of file gclib.vb.

9.38.3.22 output_analog_3

```
Analog output.* UW output_analog_3
```
/*14-15
Definition at line 1282 of file gclib.vb.

9.38.3.23 output_analog_4

```
Analog output.* UW output_analog_4
```
/*16-17
Definition at line 1283 of file gclib.vb.

9.38.3.24 output_analog_5

```
Analog output.* UW output_analog_5
```
/*18-19
Definition at line 1284 of file gclib.vb.

9.38.3.25 output_analog_6

```
Analog output.* UW output_analog_6
```
/*20-21
Definition at line 1285 of file gclib.vb.

9.38.3.26 output_analog_7

```
Analog output.* UW output_analog_7
```
/*22-23
Definition at line 1286 of file gclib.vb.

9.38.3.27 output_back_3

```
Digital outputs.Data only valid for parts with.* UW output_back_3
```
/*62-63
Definition at line 1303 of file gclib.vb.

9.38.3.28 output_bank_0

```
Analog input.* UW output_bank_0
```
/*40-41
Definition at line 1295 of file gclib.vb.

9.38.3.29 output_bank_1

```
Digital outputs.* UW output_bank_1
```
/*42-43
Definition at line 1296 of file gclib.vb.

9.38.3.30 pulse_count_0

```
Digital inputs.* UL pulse_count_0
```
/*48-51
Definition at line 1299 of file gclib.vb.

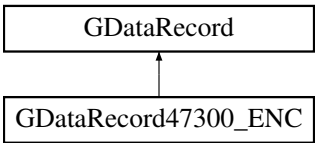
9.38.3.31 sample_number

```
byte of Header.* UW sample_number
```
/*04-05
Definition at line 1276 of file gclib.vb.
The documentation for this struct was generated from the following file:

- gclib.vb

9.39 GDataRecord47300_ENC Struct Reference

Data record struct for RIO-47300. Includes encoder fields.
Inheritance diagram for GDataRecord47300_ENC:



Public Member Functions

- byte[] byte_array ()

  Returns the data record as a byte array and allows for access to individual bytes.

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.
- UB header_3

  4th Byte of Header.
- UW sample_number

  Sample number.
- UB error_code

Error code.

- UB general_status

  General status.
- UW output_analog_0

  Analog output 0.
- UW output_analog_1

  Analog output 1.
- UW output_analog_2

  Analog output 2.
- UW output_analog_3

  Analog output 3.
- UW output_analog_4

  Analog output 4.
- UW output_analog_5

  Analog output 5.
- UW output_analog_6

  Analog output 6.
- UW output_analog_7

  Analog output 7.
- UW input_analog_0

  Analog input 0.
- UW input_analog_1

  Analog input 1.
- UW input_analog_2

  Analog input 2.
- UW input_analog_3

  Analog input 3.
- UW input_analog_4

  Analog input 4.
- UW input_analog_5

  Analog input 5.
- UW input_analog_6

  Analog input 6.
- UW input_analog_7

  Analog input 7.
- UW output_bank_0

  Digital outputs 0-15;.
- UW output_bank_1

  Digital outputs 16-23;.
- UW input_bank_0

  Digital inputs 0-15;.
- UW input_bank_1

  Digital inputs 16-23;.
- UL pulse_count_0

  Pulse counter (see PC).
- SL zc_variable

  ZC User-defined variable (see ZC).
- SL zd_variable

  ZD User-defined variable (see ZD).
- SL encoder_0

  Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.

- SL encoder_1

  Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.
- SL encoder_2

  Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.
- SL encoder_3

  Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.

9.39.1 Detailed Description

Data record struct for RIO-47300. Includes encoder fields.
Definition at line 1812 of file gclib.cs.

9.39.2 Member Function Documentation

9.39.2.1 byte_array()

```
byte[] byte_array ( )  [inline]
```

Returns the data record as a byte array and allows for access to individual bytes.
Implements GDataRecord.
Definition at line 1814 of file gclib.cs.

9.39.3 Field Documentation

9.39.3.1 encoder_0

```
SL encoder_0
```

Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1855 of file gclib.cs.

9.39.3.2 encoder_1

```
SL encoder_1
```

Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1856 of file gclib.cs.

9.39.3.3 encoder_2

```
SL encoder_2
```

Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1857 of file gclib.cs.

9.39.3.4 encoder_3

```
SL encoder_3
```

Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 1858 of file gclib.cs.

9.39.3.5 error_code

```
UB error_code
```

Error code.
Definition at line 1824 of file gclib.cs.

9.39.3.6 general_status

```
UB general_status
```

General status.
Definition at line 1825 of file gclib.cs.

9.39.3.7 header_0

`UB header_0`
1st Byte of Header.
Definition at line 1818 of file gclib.cs.

9.39.3.8 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 1819 of file gclib.cs.

9.39.3.9 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 1820 of file gclib.cs.

9.39.3.10 header_3

`UB header_3`
4th Byte of Header.
Definition at line 1821 of file gclib.cs.

9.39.3.11 input_analog_0

`UW input_analog_0`
Analog input 0.
Definition at line 1836 of file gclib.cs.

9.39.3.12 input_analog_1

`UW input_analog_1`
Analog input 1.
Definition at line 1837 of file gclib.cs.

9.39.3.13 input_analog_2

`UW input_analog_2`
Analog input 2.
Definition at line 1838 of file gclib.cs.

9.39.3.14 input_analog_3

`UW input_analog_3`
Analog input 3.
Definition at line 1839 of file gclib.cs.

9.39.3.15 input_analog_4

`UW input_analog_4`
Analog input 4.
Definition at line 1840 of file gclib.cs.

9.39.3.16 input_analog_5

`UW input_analog_5`
Analog input 5.
Definition at line 1841 of file gclib.cs.

9.39.3.17 input_analog_6

`UW input_analog_6`
Analog input 6.
Definition at line 1842 of file gclib.cs.

9.39.3.18 input_analog_7

`UW input_analog_7`
Analog input 7.
Definition at line 1843 of file gclib.cs.

9.39.3.19 input_bank_0

`UW input_bank_0`
Digital inputs 0-15;.
Definition at line 1848 of file gclib.cs.

9.39.3.20 input_bank_1

`UW input_bank_1`
Digital inputs 16-23;.
Definition at line 1849 of file gclib.cs.

9.39.3.21 output_analog_0

`UW output_analog_0`
Analog output 0.
Definition at line 1827 of file gclib.cs.

9.39.3.22 output_analog_1

`UW output_analog_1`
Analog output 1.
Definition at line 1828 of file gclib.cs.

9.39.3.23 output_analog_2

`UW output_analog_2`
Analog output 2.
Definition at line 1829 of file gclib.cs.

9.39.3.24 output_analog_3

`UW output_analog_3`
Analog output 3.
Definition at line 1830 of file gclib.cs.

9.39.3.25 output_analog_4

`UW output_analog_4`
Analog output 4.
Definition at line 1831 of file gclib.cs.

9.39.3.26 output_analog_5

`UW output_analog_5`
Analog output 5.
Definition at line 1832 of file gclib.cs.

9.39.3.27 output_analog_6

`UW output_analog_6`
Analog output 6.
Definition at line 1833 of file gclib.cs.

9.39.3.28 output_analog_7

`UW output_analog_7`
Analog output 7.
Definition at line 1834 of file gclib.cs.

9.39.3.29 output_bank_0

`UW output_bank_0`
Digital outputs 0-15;.
Definition at line 1845 of file gclib.cs.

9.39.3.30 output_bank_1

`UW output_bank_1`
Digital outputs 16-23;.
Definition at line 1846 of file gclib.cs.

9.39.3.31 pulse_count_0

`UL pulse_count_0`
Pulse counter (see PC).
Definition at line 1851 of file gclib.cs.

9.39.3.32 sample_number

`UW sample_number`
Sample number.
Definition at line 1823 of file gclib.cs.

9.39.3.33 zc_variable

`SL zc_variable`
ZC User-defined variable (see ZC).
Definition at line 1852 of file gclib.cs.

9.39.3.34 zd_variable

`SL zd_variable`
ZD User-defined variable (see ZD).
Definition at line 1853 of file gclib.cs.
The documentation for this struct was generated from the following file:

- gclib.cs

9.40 GDataRecord47300_ENC Struct Reference

Data record struct for RIO-47300. Includes encoder fields.
`#include <gclib_record.h>`

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.
- UB header_3

  4th Byte of Header.
- UW sample_number

  Sample number.
- UB error_code

  Error code.
- UB general_status

  General status.
- UW output_analog_0

  Analog output 0.
- UW output_analog_1

  Analog output 1.
- UW output_analog_2

  Analog output 2.
- UW output_analog_3

  Analog output 3.
- UW output_analog_4

  Analog output 4.
- UW output_analog_5

  Analog output 5.
- UW output_analog_6

  Analog output 6.
- UW output_analog_7

  Analog output 7.
- UW input_analog_0

  Analog input 0.
- UW input_analog_1

  Analog input 1.
- UW input_analog_2

  Analog input 2.
- UW input_analog_3

  Analog input 3.
- UW input_analog_4

  Analog input 4.
- UW input_analog_5

  Analog input 5.
- UW input_analog_6

  Analog input 6.
- UW input_analog_7

  Analog input 7.
- UW output_bank_0

  Digital outputs 0-15;.
- UW output_bank_1

Digital outputs 16-23;.

- UW input_bank_0

  Digital inputs 0-15;.

- UW input_bank_1

  Digital inputs 16-23;.

- UL pulse_count_0

  Pulse counter (see PC).

- SL zc_variable

  ZC User-defined variable (see ZC).

- SL zd_variable

  ZD User-defined variable (see ZD).

- SL encoder_0

  Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.

- SL encoder_1

  Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.

- SL encoder_2

  Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.

- SL encoder_3

  Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.

9.40.1 Detailed Description

Data record struct for RIO-47300. Includes encoder fields.
Definition at line 918 of file gclib_record.h.

9.40.2 Field Documentation

9.40.2.1 encoder_0

`SL encoder_0`
Encoder channel 0. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 960 of file gclib_record.h.

9.40.2.2 encoder_1

`SL encoder_1`
Encoder channel 1. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 961 of file gclib_record.h.

9.40.2.3 encoder_2

`SL encoder_2`
Encoder channel 2. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 962 of file gclib_record.h.

9.40.2.4 encoder_3

`SL encoder_3`
Encoder channel 3. Data only valid for parts with -BISS, -QUAD, or -SSI.
Definition at line 963 of file gclib_record.h.

9.40.2.5 error_code

`UB error_code`
Error code.
Definition at line 929 of file gclib_record.h.

9.40.2.6 general_status

`UB general_status`
General status.
Definition at line 930 of file gclib_record.h.

9.40.2.7 header_0

`UB header_0`
1st Byte of Header.
Definition at line 923 of file gclib_record.h.

9.40.2.8 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 924 of file gclib_record.h.

9.40.2.9 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 925 of file gclib_record.h.

9.40.2.10 header_3

`UB header_3`
4th Byte of Header.
Definition at line 926 of file gclib_record.h.

9.40.2.11 input_analog_0

`UW input_analog_0`
Analog input 0.
Definition at line 941 of file gclib_record.h.

9.40.2.12 input_analog_1

`UW input_analog_1`
Analog input 1.
Definition at line 942 of file gclib_record.h.

9.40.2.13 input_analog_2

`UW input_analog_2`
Analog input 2.
Definition at line 943 of file gclib_record.h.

9.40.2.14 input_analog_3

`UW input_analog_3`
Analog input 3.
Definition at line 944 of file gclib_record.h.

9.40.2.15 input_analog_4

`UW input_analog_4`
Analog input 4.
Definition at line 945 of file gclib_record.h.

9.40.2.16 input_analog_5

`UW input_analog_5`
Analog input 5.
Definition at line 946 of file gclib_record.h.

9.40.2.17 input_analog_6

`UW input_analog_6`
Analog input 6.
Definition at line 947 of file gclib_record.h.

9.40.2.18 input_analog_7

`UW input_analog_7`
Analog input 7.
Definition at line 948 of file gclib_record.h.

9.40.2.19 input_bank_0

`UW input_bank_0`
Digital inputs 0-15;.
Definition at line 953 of file gclib_record.h.

9.40.2.20 input_bank_1

`UW input_bank_1`
Digital inputs 16-23;.
Definition at line 954 of file gclib_record.h.

9.40.2.21 output_analog_0

`UW output_analog_0`
Analog output 0.
Definition at line 932 of file gclib_record.h.

9.40.2.22 output_analog_1

`UW output_analog_1`
Analog output 1.
Definition at line 933 of file gclib_record.h.

9.40.2.23 output_analog_2

`UW output_analog_2`
Analog output 2.
Definition at line 934 of file gclib_record.h.

9.40.2.24 output_analog_3

`UW output_analog_3`
Analog output 3.
Definition at line 935 of file gclib_record.h.

9.40.2.25 output_analog_4

`UW output_analog_4`
Analog output 4.
Definition at line 936 of file gclib_record.h.

9.40.2.26 output_analog_5

`UW output_analog_5`
Analog output 5.
Definition at line 937 of file gclib_record.h.

9.40.2.27 output_analog_6

`UW output_analog_6`
Analog output 6.
Definition at line 938 of file gclib_record.h.

9.40.2.28 output_analog_7

`UW output_analog_7`
Analog output 7.
Definition at line 939 of file gclib_record.h.

9.40.2.29 output_bank_0

`UW output_bank_0`
Digital outputs 0-15;.
Definition at line 950 of file gclib_record.h.

9.40.2.30 output_bank_1

`UW output_bank_1`
Digital outputs 16-23;.
Definition at line 951 of file gclib_record.h.

9.40.2.31 pulse_count_0

`UL pulse_count_0`
Pulse counter (see PC).
Definition at line 956 of file gclib_record.h.

9.40.2.32 sample_number

`UW sample_number`
Sample number.
Definition at line 928 of file gclib_record.h.

9.40.2.33 zc_variable

`SL zc_variable`
ZC User-defined variable (see ZC).
Definition at line 957 of file gclib_record.h.

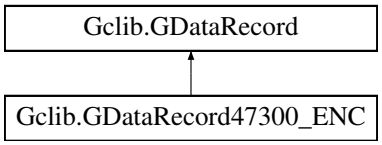
9.40.2.34 zd_variable

`SL zd_variable`
ZD User-defined variable (see ZD).
Definition at line 958 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

9.41 Gclib.GDataRecord47300_ENC Struct Reference

Inheritance diagram for Gclib.GDataRecord47300_ENC:



Public Member Functions

- Pulse **counter** (see PC). ∗/public SL zc_variable
  /∗52-55
- ZC User defined **variable** (see ZC). ∗/public SL zd_variable
  /∗56-59
- ZD User defined **variable** (see ZD). ∗/public SL encoder_0
  /∗60-63
- Encoder channel.Data only valid for parts with or SSI.∗ byte_array ()

Data Fields

- UB header_0
  /∗00
- byte of Header.∗UB header_1
  /∗01
- byte of Header.∗UB header_2
  /∗02
- byte of Header.∗UB header_3
  /∗03
- byte of Header.∗UW sample_number
  /∗04-05
- Sample number.∗UB error_code
  /∗06
- Error code.∗UB general_status
  /∗07
- General status.∗UW output_analog_0
  /∗08-09
- Analog output.∗UW output_analog_1
  /∗10-11
- Analog output.∗UW output_analog_2
  /∗12-13
- Analog output.∗UW output_analog_3
  /∗14-15
- Analog output.∗UW output_analog_4
  /∗16-17
- Analog output.∗UW output_analog_5
  /∗18-19
- Analog output.∗UW output_analog_6
  /∗20-21
- Analog output.∗UW output_analog_7
  /∗22-23

- Analog output.∗UW input_analog_0
  /∗24-25
- Analog input.∗UW input_analog_1
  /∗26-27
- Analog input.∗UW input_analog_2
  /∗28-29
- Analog input.∗UW input_analog_3
  /∗30-31
- Analog input.∗UW input_analog_4
  /∗32-33
- Analog input.∗UW input_analog_5
  /∗34-35
- Analog input.∗UW input_analog_6
  /∗36-37
- Analog input.∗UW input_analog_7
  /∗38-39
- Analog input.∗UW output_bank_0
  /∗40-41
- Digital outputs
- ∗UW output_bank_1
  /∗42-43
- ∗UW input_bank_0
  /∗44-45
- Digital inputs
- ∗UW input_bank_1
  /∗46-47
- ∗UL pulse_count_0
  /∗48-51
- Encoder channel.Data only valid for parts with BISS
- Encoder channel.Data only valid for parts with QUAD
- Encoder channel.Data only valid for parts with or SSI.∗SL encoder_1
  /∗64-67
- Encoder channel.Data only valid for parts with or SSI.∗SL encoder_2
  /∗68-71
- Encoder channel.Data only valid for parts with or SSI.∗SL encoder_3
  /∗72-75

9.41.1 Detailed Description

Definition at line 1230 of file gclib.vb.

9.41.2 Member Function Documentation

9.41.2.1 byte_array()

```
Encoder channel.Data only valid for parts with or SSI.* byte_array ( )
```

Implements GDataRecord.byte_array
Implements Gclib.GDataRecord.

9.41.3 Field Documentation

9.41.3.1 BISS

```
Encoder channel Data only valid for parts with BISS
```

Definition at line 1263 of file gclib.vb.

9.41.3.2 encoder_1

```
Encoder channel.Data only valid for parts with or SSI.* SL encoder_1
```
/*64-67
Definition at line 1264 of file gclib.vb.

9.41.3.3 encoder_2

```
Encoder channel.Data only valid for parts with or SSI.* SL encoder_2
```
/*68-71
Definition at line 1265 of file gclib.vb.

9.41.3.4 encoder_3

```
Encoder channel.Data only valid for parts with or SSI.* SL encoder_3
```
/*72-75
Definition at line 1266 of file gclib.vb.

9.41.3.5 error_code

```
Sample number.* UB error_code
```
/*06
Definition at line 1238 of file gclib.vb.

9.41.3.6 general_status

```
Error code.* UB general_status
```
/*07
Definition at line 1239 of file gclib.vb.

9.41.3.7 header_0

```
UB header_0
```
/*00
Definition at line 1233 of file gclib.vb.

9.41.3.8 header_1

```
byte of Header.* UB header_1
```
/*01
Definition at line 1234 of file gclib.vb.

9.41.3.9 header_2

```
byte of Header.* UB header_2
```
/*02
Definition at line 1235 of file gclib.vb.

9.41.3.10 header_3

```
byte of Header.* UB header_3
```
/*03
Definition at line 1236 of file gclib.vb.

9.41.3.11 input_analog_0

```
Analog output.* UW input_analog_0
```
/*24-25
Definition at line 1248 of file gclib.vb.

9.41.3.12 input_analog_1

```
Analog input.* UW input_analog_1
```
/*26-27
Definition at line 1249 of file gclib.vb.

9.41.3.13 input_analog_2

```
Analog input.* UW input_analog_2
```
/*28-29
Definition at line 1250 of file gclib.vb.

9.41.3.14 input_analog_3

```
Analog input.* UW input_analog_3
```
/*30-31
Definition at line 1251 of file gclib.vb.

9.41.3.15 input_analog_4

```
Analog input.* UW input_analog_4
```
/*32-33
Definition at line 1252 of file gclib.vb.

9.41.3.16 input_analog_5

```
Analog input.* UW input_analog_5
```
/*34-35
Definition at line 1253 of file gclib.vb.

9.41.3.17 input_analog_6

```
Analog input.* UW input_analog_6
```
/*36-37
Definition at line 1254 of file gclib.vb.

9.41.3.18 input_analog_7

```
Analog input.* UW input_analog_7
```
/*38-39
Definition at line 1255 of file gclib.vb.

9.41.3.19 input_bank_0

```
* UW input_bank_0
```
/*44-45
Definition at line 1258 of file gclib.vb.

9.41.3.20 input_bank_1

```
* UW input_bank_1
```
/*46-47
Definition at line 1259 of file gclib.vb.

9.41.3.21 inputs

```
Digital inputs
```
Definition at line 1258 of file gclib.vb.

9.41.3.22 output_analog_0

```
General status.* UW output_analog_0
```
/*08-09
Definition at line 1240 of file gclib.vb.

9.41.3.23 output_analog_1

```
Analog output.* UW output_analog_1
```
/*10-11
Definition at line 1241 of file gclib.vb.

9.41.3.24 output_analog_2

```
Analog output.* UW output_analog_2
```
/*12-13
Definition at line 1242 of file gclib.vb.

9.41.3.25 output_analog_3

```
Analog output.* UW output_analog_3
```
/*14-15
Definition at line 1243 of file gclib.vb.

9.41.3.26 output_analog_4

```
Analog output.* UW output_analog_4
```
/*16-17
Definition at line 1244 of file gclib.vb.

9.41.3.27 output_analog_5

```
Analog output.* UW output_analog_5
```
/*18-19
Definition at line 1245 of file gclib.vb.

9.41.3.28 output_analog_6

```
Analog output.* UW output_analog_6
```
/*20-21
Definition at line 1246 of file gclib.vb.

9.41.3.29 output_analog_7

```
Analog output.* UW output_analog_7
```
/*22-23
Definition at line 1247 of file gclib.vb.

9.41.3.30 output_bank_0

```
Analog input.* UW output_bank_0
```
/*40-41
Definition at line 1256 of file gclib.vb.

9.41.3.31 output_bank_1

```
* UW output_bank_1
```
/*42-43
Definition at line 1257 of file gclib.vb.

9.41.3.32 outputs

```
Digital outputs
```
Definition at line 1256 of file gclib.vb.

9.41.3.33 pulse_count_0

```
* UL pulse_count_0
```
/*48-51
Definition at line 1260 of file gclib.vb.

9.41.3.34 QUAD

```
Encoder channel Data only valid for parts with QUAD
```
Definition at line 1263 of file gclib.vb.

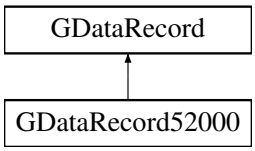
9.41.3.35 sample_number

```
byte of Header.* UW sample_number
```
/*04-05
Definition at line 1237 of file gclib.vb.
The documentation for this struct was generated from the following file:

- gclib.vb

9.42 GDataRecord52000 Struct Reference

Data record struct for DMC-52000 controller. Same as DMC-4000, with bank indicator added at byte 40.
Inheritance diagram for GDataRecord52000:



Public Member Functions

- byte[] byte_array ()

  Returns the data record as a byte array and allows for access to individual bytes.

Data Fields

- UB header_0

  1st Byte of Header.
- UB header_1

  2nd Byte of Header.
- UB header_2

  3rd Byte of Header.
- UB header_3

  4th Byte of Header.
- UW sample_number

  sample number.
- UB input_bank_0

  general input bank 0 (inputs 1-8).
- UB input_bank_1

general input bank 1 (inputs 9-16).

- UB input_bank_2

  general input bank 2 (inputs 17-24).
- UB input_bank_3

  general input bank 3 (inputs 25-32).
- UB input_bank_4

  general input bank 4 (inputs 33-40).
- UB input_bank_5

  general input bank 5 (inputs 41-48).
- UB input_bank_6

  general input bank 6 (inputs 49-56).
- UB input_bank_7

  general input bank 7 (inputs 57-64).
- UB input_bank_8

  general input bank 8 (inputs 65-72).
- UB input_bank_9

  general input bank 9 (inputs 73-80).
- UB output_bank_0

  general output bank 0 (outputs 1-8).
- UB output_bank_1

  general output bank 1 (outputs 9-16).
- UB output_bank_2

  general output bank 2 (outputs 17-24).
- UB output_bank_3

  general output bank 3 (outputs 25-32).
- UB output_bank_4

  general output bank 4 (outputs 33-40).
- UB output_bank_5

  general output bank 5 (outputs 41-48).
- UB output_bank_6

  general output bank 6 (outputs 49-56).
- UB output_bank_7

  general output bank 7 (outputs 57-64).
- UB output_bank_8

  general output bank 8 (outputs 65-72).
- UB output_bank_9

  general output bank 9 (outputs 73-80).
- SW reserved_0

  Reserved.
- SW reserved_2

  Reserved.
- SW reserved_4

  Reserved.
- SW reserved_6

  Reserved.
- SW reserved_8

  Reserved.
- SW reserved_10

  Reserved.
- SW reserved_12

  Reserved.

- UB ethercat_bank

  EtherCAT Bank Indicator.
- UB reserved_14

  Reserved.
- UB ethernet_status_a

  Ethernet Handle A Status.
- UB ethernet_status_b

  Ethernet Handle B Status.
- UB ethernet_status_c

  Ethernet Handle C Status.
- UB ethernet_status_d

  Ethernet Handle D Status.
- UB ethernet_status_e

  Ethernet Handle E Status.
- UB ethernet_status_f

  Ethernet Handle F Status.
- UB ethernet_status_g

  Ethernet Handle G Status.
- UB ethernet_status_h

  Ethernet Handle H Status.
- UB error_code

  error code.
- UB thread_status

  thread status
- UL amplifier_status

  Amplifier Status.
- UL contour_segment_count

  Segment Count for Contour Mode.
- UW contour_buffer_available

  Buffer space remaining, Contour Mode.
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

  coordinated move status for S plane.
- SL s_distance

  distance traveled in coordinated move for S plane.
- UW s_plane_buffer_available

  Buffer space remaining, S Plane.
- UW t_plane_segment_count

  segment count of coordinated move for T plane.
- UW t_plane_move_status

  Coordinated move status for T plane.
- SL t_distance

  distance traveled in coordinated move for T plane.
- UW t_plane_buffer_available

  Buffer space remaining, T Plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

A axis stop code.

- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

  A axis auxiliary position.
- SL axis_a_velocity

  A axis velocity.
- SL axis_a_torque

  A axis torque.
- UW axis_a_analog_in

  A axis analog input.
- UB axis_a_halls

  A Hall Input Status.
- UB axis_a_reserved

  Reserved.
- SL axis_a_variable

  A User-defined variable (ZA).
- UW axis_b_status

  B axis status.
- UB axis_b_switches

  B axis switches.
- UB axis_b_stop_code

  B axis stop code.
- SL axis_b_reference_position

  B axis reference position.
- SL axis_b_motor_position

  B axis motor position.
- SL axis_b_position_error

  B axis position error.
- SL axis_b_aux_position

  B axis auxiliary position.
- SL axis_b_velocity

  B axis velocity.
- SL axis_b_torque

  B axis torque.
- UW axis_b_analog_in

  B axis analog input.
- UB axis_b_halls

  B Hall Input Status.
- UB axis_b_reserved

  Reserved.
- SL axis_b_variable

  B User-defined variable (ZA).
- UW axis_c_status

  C axis status.
- UB axis_c_switches

  C axis switches.

- UB axis_c_stop_code

  C axis stop code.
- SL axis_c_reference_position

  C axis reference position.
- SL axis_c_motor_position

  C axis motor position.
- SL axis_c_position_error

  C axis position error.
- SL axis_c_aux_position

  C axis auxiliary position.
- SL axis_c_velocity

  C axis velocity.
- SL axis_c_torque

  C axis torque.
- UW axis_c_analog_in

  C axis analog input.
- UB axis_c_halls

  C Hall Input Status.
- UB axis_c_reserved

  Reserved.
- SL axis_c_variable

  C User-defined variable (ZA).
- UW axis_d_status

  D axis status.
- UB axis_d_switches

  D axis switches.
- UB axis_d_stop_code

  D axis stop code.
- SL axis_d_reference_position

  D axis reference position.
- SL axis_d_motor_position

  D axis motor position.
- SL axis_d_position_error

  D axis position error.
- SL axis_d_aux_position

  D axis auxiliary position.
- SL axis_d_velocity

  D axis velocity.
- SL axis_d_torque

  D axis torque.
- UW axis_d_analog_in

  D axis analog input.
- UB axis_d_halls

  D Hall Input Status.
- UB axis_d_reserved

  Reserved.
- SL axis_d_variable

  D User-defined variable (ZA).
- UW axis_e_status

  E axis status.
- UB axis_e_switches

E axis switches.

- UB axis_e_stop_code

  E axis stop code.
- SL axis_e_reference_position

  E axis reference position.
- SL axis_e_motor_position

  E axis motor position.
- SL axis_e_position_error

  E axis position error.
- SL axis_e_aux_position

  E axis auxiliary position.
- SL axis_e_velocity

  E axis velocity.
- SL axis_e_torque

  E axis torque.
- UW axis_e_analog_in

  E axis analog input.
- UB axis_e_halls

  E Hall Input Status.
- UB axis_e_reserved

  Reserved.
- SL axis_e_variable

  E User-defined variable (ZA).
- UW axis_f_status

  F axis status.
- UB axis_f_switches

  F axis switches.
- UB axis_f_stop_code

  F axis stop code.
- SL axis_f_reference_position

  F axis reference position.
- SL axis_f_motor_position

  F axis motor position.
- SL axis_f_position_error

  F axis position error.
- SL axis_f_aux_position

  F axis auxiliary position.
- SL axis_f_velocity

  F axis velocity.
- SL axis_f_torque

  F axis torque.
- UW axis_f_analog_in

  F axis analog input.
- UB axis_f_halls

  F Hall Input Status.
- UB axis_f_reserved

  Reserved.
- SL axis_f_variable

  F User-defined variable (ZA).
- UW axis_g_status

  G axis status.

- UB axis_g_switches

  G axis switches.
- UB axis_g_stop_code

  G axis stop code.
- SL axis_g_reference_position

  G axis reference position.
- SL axis_g_motor_position

  G axis motor position.
- SL axis_g_position_error

  G axis position error.
- SL axis_g_aux_position

  G axis auxiliary position.
- SL axis_g_velocity

  G axis velocity.
- SL axis_g_torque

  G axis torque.
- UW axis_g_analog_in

  G axis analog input.
- UB axis_g_halls

  G Hall Input Status.
- UB axis_g_reserved

  Reserved.
- SL axis_g_variable

  G User-defined variable (ZA).
- UW axis_h_status

  H axis status.
- UB axis_h_switches

  H axis switches.
- UB axis_h_stop_code

  H axis stop code.
- SL axis_h_reference_position

  H axis reference position.
- SL axis_h_motor_position

  H axis motor position.
- SL axis_h_position_error

  H axis position error.
- SL axis_h_aux_position

  H axis auxiliary position.
- SL axis_h_velocity

  H axis velocity.
- SL axis_h_torque

  H axis torque.
- UW axis_h_analog_in

  H axis analog input.
- UB axis_h_halls

  H Hall Input Status.
- UB axis_h_reserved

  Reserved.
- SL axis_h_variable

  H User-defined variable (ZA).

9.42.1 Detailed Description

Data record struct for DMC-52000 controller. Same as DMC-4000, with bank indicator added at byte 40.
Definition at line 1102 of file gclib.cs.

9.42.2 Member Function Documentation

9.42.2.1 byte_array()

```
byte[] byte_array ( )  [inline]
```

Returns the data record as a byte array and allows for access to individual bytes.
Implements GDataRecord.
Definition at line 1104 of file gclib.cs.

9.42.3 Field Documentation

9.42.3.1 amplifier_status

```
UL amplifier_status
```

Amplifier Status.
Definition at line 1157 of file gclib.cs.

9.42.3.2 axis_a_analog_in

```
UW axis_a_analog_in
```

A axis analog input.
Definition at line 1181 of file gclib.cs.

9.42.3.3 axis_a_aux_position

```
SL axis_a_aux_position
```

A axis auxiliary position.
Definition at line 1178 of file gclib.cs.

9.42.3.4 axis_a_halls

```
UB axis_a_halls
```

A Hall Input Status.
Definition at line 1182 of file gclib.cs.

9.42.3.5 axis_a_motor_position

```
SL axis_a_motor_position
```

A axis motor position.
Definition at line 1176 of file gclib.cs.

9.42.3.6 axis_a_position_error

```
SL axis_a_position_error
```

A axis position error.
Definition at line 1177 of file gclib.cs.

9.42.3.7 axis_a_reference_position

```
SL axis_a_reference_position
```

A axis reference position.
Definition at line 1175 of file gclib.cs.

9.42.3.8 axis_a_reserved

`UB axis_a_reserved`
Reserved.
Definition at line 1183 of file gclib.cs.

9.42.3.9 axis_a_status

`UW axis_a_status`
A axis status.
Definition at line 1172 of file gclib.cs.

9.42.3.10 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 1174 of file gclib.cs.

9.42.3.11 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 1173 of file gclib.cs.

9.42.3.12 axis_a_torque

`SL axis_a_torque`
A axis torque.
Definition at line 1180 of file gclib.cs.

9.42.3.13 axis_a_variable

`SL axis_a_variable`
A User-defined variable (ZA).
Definition at line 1184 of file gclib.cs.

9.42.3.14 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 1179 of file gclib.cs.

9.42.3.15 axis_b_analog_in

`UW axis_b_analog_in`
B axis analog input.
Definition at line 1195 of file gclib.cs.

9.42.3.16 axis_b_aux_position

`SL axis_b_aux_position`
B axis auxiliary position.
Definition at line 1192 of file gclib.cs.

9.42.3.17 axis_b_halls

`UB axis_b_halls`
B Hall Input Status.
Definition at line 1196 of file gclib.cs.

9.42.3.18 axis_b_motor_position

`SL axis_b_motor_position`
B axis motor position.
Definition at line 1190 of file gclib.cs.

9.42.3.19 axis_b_position_error

`SL axis_b_position_error`
B axis position error.
Definition at line 1191 of file gclib.cs.

9.42.3.20 axis_b_reference_position

`SL axis_b_reference_position`
B axis reference position.
Definition at line 1189 of file gclib.cs.

9.42.3.21 axis_b_reserved

`UB axis_b_reserved`
Reserved.
Definition at line 1197 of file gclib.cs.

9.42.3.22 axis_b_status

`UW axis_b_status`
B axis status.
Definition at line 1186 of file gclib.cs.

9.42.3.23 axis_b_stop_code

`UB axis_b_stop_code`
B axis stop code.
Definition at line 1188 of file gclib.cs.

9.42.3.24 axis_b_switches

`UB axis_b_switches`
B axis switches.
Definition at line 1187 of file gclib.cs.

9.42.3.25 axis_b_torque

`SL axis_b_torque`
B axis torque.
Definition at line 1194 of file gclib.cs.

9.42.3.26 axis_b_variable

`SL axis_b_variable`
B User-defined variable (ZA).
Definition at line 1198 of file gclib.cs.

9.42.3.27 axis_b_velocity

`SL axis_b_velocity`
B axis velocity.
Definition at line 1193 of file gclib.cs.

9.42.3.28 axis_c_analog_in

`UW axis_c_analog_in`
C axis analog input.
Definition at line 1209 of file gclib.cs.

9.42.3.29 axis_c_aux_position

`SL axis_c_aux_position`
C axis auxiliary position.
Definition at line 1206 of file gclib.cs.

9.42.3.30 axis_c_halls

`UB axis_c_halls`
C Hall Input Status.
Definition at line 1210 of file gclib.cs.

9.42.3.31 axis_c_motor_position

`SL axis_c_motor_position`
C axis motor position.
Definition at line 1204 of file gclib.cs.

9.42.3.32 axis_c_position_error

`SL axis_c_position_error`
C axis position error.
Definition at line 1205 of file gclib.cs.

9.42.3.33 axis_c_reference_position

`SL axis_c_reference_position`
C axis reference position.
Definition at line 1203 of file gclib.cs.

9.42.3.34 axis_c_reserved

`UB axis_c_reserved`
Reserved.
Definition at line 1211 of file gclib.cs.

9.42.3.35 axis_c_status

`UW axis_c_status`
C axis status.
Definition at line 1200 of file gclib.cs.

9.42.3.36 axis_c_stop_code

`UB axis_c_stop_code`
C axis stop code.
Definition at line 1202 of file gclib.cs.

9.42.3.37 axis_c_switches

`UB axis_c_switches`
C axis switches.
Definition at line 1201 of file gclib.cs.

9.42.3.38 axis_c_torque

`SL axis_c_torque`
C axis torque.
Definition at line 1208 of file gclib.cs.

9.42.3.39 axis_c_variable

`SL axis_c_variable`
C User-defined variable (ZA).
Definition at line 1212 of file gclib.cs.

9.42.3.40 axis_c_velocity

`SL axis_c_velocity`
C axis velocity.
Definition at line 1207 of file gclib.cs.

9.42.3.41 axis_d_analog_in

`UW axis_d_analog_in`
D axis analog input.
Definition at line 1223 of file gclib.cs.

9.42.3.42 axis_d_aux_position

`SL axis_d_aux_position`
D axis auxiliary position.
Definition at line 1220 of file gclib.cs.

9.42.3.43 axis_d_halls

`UB axis_d_halls`
D Hall Input Status.
Definition at line 1224 of file gclib.cs.

9.42.3.44 axis_d_motor_position

`SL axis_d_motor_position`
D axis motor position.
Definition at line 1218 of file gclib.cs.

9.42.3.45 axis_d_position_error

`SL axis_d_position_error`
D axis position error.
Definition at line 1219 of file gclib.cs.

9.42.3.46 axis_d_reference_position

`SL axis_d_reference_position`
D axis reference position.
Definition at line 1217 of file gclib.cs.

9.42.3.47 axis_d_reserved

`UB axis_d_reserved`
Reserved.
Definition at line 1225 of file gclib.cs.

9.42.3.48 axis_d_status

`UW axis_d_status`
D axis status.
Definition at line 1214 of file gclib.cs.

9.42.3.49 axis_d_stop_code

`UB axis_d_stop_code`
D axis stop code.
Definition at line 1216 of file gclib.cs.

9.42.3.50 axis_d_switches

`UB axis_d_switches`
D axis switches.
Definition at line 1215 of file gclib.cs.

9.42.3.51 axis_d_torque

`SL axis_d_torque`
D axis torque.
Definition at line 1222 of file gclib.cs.

9.42.3.52 axis_d_variable

`SL axis_d_variable`
D User-defined variable (ZA).
Definition at line 1226 of file gclib.cs.

9.42.3.53 axis_d_velocity

`SL axis_d_velocity`
D axis velocity.
Definition at line 1221 of file gclib.cs.

9.42.3.54 axis_e_analog_in

`UW axis_e_analog_in`
E axis analog input.
Definition at line 1237 of file gclib.cs.

9.42.3.55 axis_e_aux_position

`SL axis_e_aux_position`
E axis auxiliary position.
Definition at line 1234 of file gclib.cs.

9.42.3.56 axis_e_halls

`UB axis_e_halls`
E Hall Input Status.
Definition at line 1238 of file gclib.cs.

9.42.3.57 axis_e_motor_position

`SL axis_e_motor_position`
E axis motor position.
Definition at line 1232 of file gclib.cs.

9.42.3.58 axis_e_position_error

`SL axis_e_position_error`
E axis position error.
Definition at line 1233 of file gclib.cs.

9.42.3.59 axis_e_reference_position

`SL axis_e_reference_position`
E axis reference position.
Definition at line 1231 of file gclib.cs.

9.42.3.60 axis_e_reserved

`UB axis_e_reserved`
Reserved.
Definition at line 1239 of file gclib.cs.

9.42.3.61 axis_e_status

`UW axis_e_status`
E axis status.
Definition at line 1228 of file gclib.cs.

9.42.3.62 axis_e_stop_code

`UB axis_e_stop_code`
E axis stop code.
Definition at line 1230 of file gclib.cs.

9.42.3.63 axis_e_switches

`UB axis_e_switches`
E axis switches.
Definition at line 1229 of file gclib.cs.

9.42.3.64 axis_e_torque

`SL axis_e_torque`
E axis torque.
Definition at line 1236 of file gclib.cs.

9.42.3.65 axis_e_variable

`SL axis_e_variable`
E User-defined variable (ZA).
Definition at line 1240 of file gclib.cs.

9.42.3.66 axis_e_velocity

`SL axis_e_velocity`
E axis velocity.
Definition at line 1235 of file gclib.cs.

9.42.3.67 axis_f_analog_in

`UW axis_f_analog_in`
F axis analog input.
Definition at line 1251 of file gclib.cs.

9.42.3.68 axis_f_aux_position

`SL axis_f_aux_position`
F axis auxiliary position.
Definition at line 1248 of file gclib.cs.

9.42.3.69 axis_f_halls

`UB axis_f_halls`
F Hall Input Status.
Definition at line 1252 of file gclib.cs.

9.42.3.70 axis_f_motor_position

`SL axis_f_motor_position`
F axis motor position.
Definition at line 1246 of file gclib.cs.

9.42.3.71 axis_f_position_error

`SL axis_f_position_error`
F axis position error.
Definition at line 1247 of file gclib.cs.

9.42.3.72 axis_f_reference_position

`SL axis_f_reference_position`
F axis reference position.
Definition at line 1245 of file gclib.cs.

9.42.3.73 axis_f_reserved

`UB axis_f_reserved`
Reserved.
Definition at line 1253 of file gclib.cs.

9.42.3.74 axis_f_status

`UW axis_f_status`
F axis status.
Definition at line 1242 of file gclib.cs.

9.42.3.75 axis_f_stop_code

`UB axis_f_stop_code`
F axis stop code.
Definition at line 1244 of file gclib.cs.

9.42.3.76 axis_f_switches

`UB axis_f_switches`
F axis switches.
Definition at line 1243 of file gclib.cs.

9.42.3.77 axis_f_torque

`SL axis_f_torque`
F axis torque.
Definition at line 1250 of file gclib.cs.

9.42.3.78 axis_f_variable

`SL axis_f_variable`
F User-defined variable (ZA).
Definition at line 1254 of file gclib.cs.

9.42.3.79 axis_f_velocity

`SL axis_f_velocity`
F axis velocity.
Definition at line 1249 of file gclib.cs.

9.42.3.80 axis_g_analog_in

`UW axis_g_analog_in`
G axis analog input.
Definition at line 1265 of file gclib.cs.

9.42.3.81 axis_g_aux_position

`SL axis_g_aux_position`
G axis auxiliary position.
Definition at line 1262 of file gclib.cs.

9.42.3.82 axis_g_halls

`UB axis_g_halls`
G Hall Input Status.
Definition at line 1266 of file gclib.cs.

9.42.3.83 axis_g_motor_position

`SL axis_g_motor_position`
G axis motor position.
Definition at line 1260 of file gclib.cs.

9.42.3.84 axis_g_position_error

`SL axis_g_position_error`
G axis position error.
Definition at line 1261 of file gclib.cs.

9.42.3.85 axis_g_reference_position

`SL axis_g_reference_position`
G axis reference position.
Definition at line 1259 of file gclib.cs.

9.42.3.86 axis_g_reserved

`UB axis_g_reserved`
Reserved.
Definition at line 1267 of file gclib.cs.

9.42.3.87 axis_g_status

`UW axis_g_status`
G axis status.
Definition at line 1256 of file gclib.cs.

9.42.3.88 axis_g_stop_code

`UB axis_g_stop_code`
G axis stop code.
Definition at line 1258 of file gclib.cs.

9.42.3.89 axis_g_switches

`UB axis_g_switches`
G axis switches.
Definition at line 1257 of file gclib.cs.

9.42.3.90 axis_g_torque

`SL axis_g_torque`
G axis torque.
Definition at line 1264 of file gclib.cs.

9.42.3.91 axis_g_variable

`SL axis_g_variable`
G User-defined variable (ZA).
Definition at line 1268 of file gclib.cs.

9.42.3.92 axis_g_velocity

`SL axis_g_velocity`
G axis velocity.
Definition at line 1263 of file gclib.cs.

9.42.3.93 axis_h_analog_in

`UW axis_h_analog_in`
H axis analog input.
Definition at line 1279 of file gclib.cs.

9.42.3.94 axis_h_aux_position

`SL axis_h_aux_position`
H axis auxiliary position.
Definition at line 1276 of file gclib.cs.

9.42.3.95 axis_h_halls

`UB axis_h_halls`
H Hall Input Status.
Definition at line 1280 of file gclib.cs.

9.42.3.96 axis_h_motor_position

`SL axis_h_motor_position`
H axis motor position.
Definition at line 1274 of file gclib.cs.

9.42.3.97 axis_h_position_error

`SL axis_h_position_error`
H axis position error.
Definition at line 1275 of file gclib.cs.

9.42.3.98 axis_h_reference_position

`SL axis_h_reference_position`
H axis reference position.
Definition at line 1273 of file gclib.cs.

9.42.3.99 axis_h_reserved

`UB axis_h_reserved`
Reserved.
Definition at line 1281 of file gclib.cs.

9.42.3.100 axis_h_status

`UW axis_h_status`
H axis status.
Definition at line 1270 of file gclib.cs.

9.42.3.101 axis_h_stop_code

`UB axis_h_stop_code`
H axis stop code.
Definition at line 1272 of file gclib.cs.

9.42.3.102 axis_h_switches

`UB axis_h_switches`
H axis switches.
Definition at line 1271 of file gclib.cs.

9.42.3.103 axis_h_torque

`SL axis_h_torque`
H axis torque.
Definition at line 1278 of file gclib.cs.

9.42.3.104 axis_h_variable

`SL axis_h_variable`
H User-defined variable (ZA).
Definition at line 1282 of file gclib.cs.

9.42.3.105 axis_h_velocity

`SL axis_h_velocity`
H axis velocity.
Definition at line 1277 of file gclib.cs.

9.42.3.106 contour_buffer_available

`UW contour_buffer_available`
Buffer space remaining, Contour Mode.
Definition at line 1160 of file gclib.cs.

9.42.3.107 contour_segment_count

`UL contour_segment_count`
Segment Count for Contour Mode.
Definition at line 1159 of file gclib.cs.

9.42.3.108 error_code

`UB error_code`
error code.
Definition at line 1155 of file gclib.cs.

9.42.3.109 ethercat_bank

`UB ethercat_bank`
EtherCAT Bank Indicator.
Definition at line 1143 of file gclib.cs.

9.42.3.110 ethernet_status_a

`UB ethernet_status_a`
Ethernet Handle A Status.
Definition at line 1146 of file gclib.cs.

9.42.3.111 ethernet_status_b

`UB ethernet_status_b`
Ethernet Handle B Status.
Definition at line 1147 of file gclib.cs.

9.42.3.112 ethernet_status_c

`UB ethernet_status_c`
Ethernet Handle C Status.
Definition at line 1148 of file gclib.cs.

9.42.3.113 ethernet_status_d

`UB ethernet_status_d`
Ethernet Handle D Status.
Definition at line 1149 of file gclib.cs.

9.42.3.114 ethernet_status_e

`UB ethernet_status_e`
Ethernet Handle E Status.
Definition at line 1150 of file gclib.cs.

9.42.3.115 ethernet_status_f

`UB ethernet_status_f`
Ethernet Handle F Status.
Definition at line 1151 of file gclib.cs.

9.42.3.116 ethernet_status_g

`UB ethernet_status_g`
Ethernet Handle G Status.
Definition at line 1152 of file gclib.cs.

9.42.3.117 ethernet_status_h

`UB ethernet_status_h`
Ethernet Handle H Status.
Definition at line 1153 of file gclib.cs.

9.42.3.118 header_0

`UB header_0`
1st Byte of Header.
Definition at line 1107 of file gclib.cs.

9.42.3.119 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 1108 of file gclib.cs.

9.42.3.120 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 1109 of file gclib.cs.

9.42.3.121 header_3

`UB header_3`
4th Byte of Header.
Definition at line 1110 of file gclib.cs.

9.42.3.122 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 1114 of file gclib.cs.

9.42.3.123 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 1115 of file gclib.cs.

9.42.3.124 input_bank_2

`UB input_bank_2`
general input bank 2 (inputs 17-24).
Definition at line 1116 of file gclib.cs.

9.42.3.125 input_bank_3

`UB input_bank_3`
general input bank 3 (inputs 25-32).
Definition at line 1117 of file gclib.cs.

9.42.3.126 input_bank_4

`UB input_bank_4`
general input bank 4 (inputs 33-40).
Definition at line 1118 of file gclib.cs.

9.42.3.127 input_bank_5

`UB input_bank_5`
general input bank 5 (inputs 41-48).
Definition at line 1119 of file gclib.cs.

9.42.3.128 input_bank_6

`UB input_bank_6`
general input bank 6 (inputs 49-56).
Definition at line 1120 of file gclib.cs.

9.42.3.129 input_bank_7

`UB input_bank_7`
general input bank 7 (inputs 57-64).
Definition at line 1121 of file gclib.cs.

9.42.3.130 input_bank_8

`UB input_bank_8`
general input bank 8 (inputs 65-72).
Definition at line 1122 of file gclib.cs.

9.42.3.131 input_bank_9

`UB input_bank_9`
general input bank 9 (inputs 73-80).
Definition at line 1123 of file gclib.cs.

9.42.3.132 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 1125 of file gclib.cs.

9.42.3.133 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 1126 of file gclib.cs.

9.42.3.134 output_bank_2

`UB output_bank_2`
general output bank 2 (outputs 17-24).
Definition at line 1127 of file gclib.cs.

9.42.3.135 output_bank_3

`UB output_bank_3`
general output bank 3 (outputs 25-32).
Definition at line 1128 of file gclib.cs.

9.42.3.136 output_bank_4

`UB output_bank_4`
general output bank 4 (outputs 33-40).
Definition at line 1129 of file gclib.cs.

9.42.3.137 output_bank_5

`UB output_bank_5`
general output bank 5 (outputs 41-48).
Definition at line 1130 of file gclib.cs.

9.42.3.138 output_bank_6

`UB output_bank_6`
general output bank 6 (outputs 49-56).
Definition at line 1131 of file gclib.cs.

9.42.3.139 output_bank_7

`UB output_bank_7`
general output bank 7 (outputs 57-64).
Definition at line 1132 of file gclib.cs.

9.42.3.140 output_bank_8

`UB output_bank_8`
general output bank 8 (outputs 65-72).
Definition at line 1133 of file gclib.cs.

9.42.3.141 output_bank_9

`UB output_bank_9`
general output bank 9 (outputs 73-80).
Definition at line 1134 of file gclib.cs.

9.42.3.142 reserved_0

`SW reserved_0`
Reserved.
Definition at line 1136 of file gclib.cs.

9.42.3.143 reserved_10

`SW reserved_10`
Reserved.
Definition at line 1141 of file gclib.cs.

9.42.3.144 reserved_12

`SW reserved_12`
Reserved.
Definition at line 1142 of file gclib.cs.

9.42.3.145 reserved_14

`UB reserved_14`
Reserved.
Definition at line 1144 of file gclib.cs.

9.42.3.146 reserved_2

`SW reserved_2`
Reserved.
Definition at line 1137 of file gclib.cs.

9.42.3.147 reserved_4

`SW reserved_4`
Reserved.
Definition at line 1138 of file gclib.cs.

9.42.3.148 reserved_6

`SW reserved_6`
Reserved.
Definition at line 1139 of file gclib.cs.

9.42.3.149 reserved_8

`SW reserved_8`
Reserved.
Definition at line 1140 of file gclib.cs.

9.42.3.150 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 1164 of file gclib.cs.

9.42.3.151 s_plane_buffer_available

`UW s_plane_buffer_available`
Buffer space remaining, S Plane.
Definition at line 1165 of file gclib.cs.

9.42.3.152 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 1163 of file gclib.cs.

9.42.3.153 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 1162 of file gclib.cs.

9.42.3.154 sample_number

`UW sample_number`
sample number.
Definition at line 1112 of file gclib.cs.

9.42.3.155 t_distance

`SL t_distance`
distance traveled in coordinated move for T plane.
Definition at line 1169 of file gclib.cs.

9.42.3.156 t_plane_buffer_available

`UW t_plane_buffer_available`
Buffer space remaining, T Plane.
Definition at line 1170 of file gclib.cs.

9.42.3.157 t_plane_move_status

`UW t_plane_move_status`
Coordinated move status for T plane.
Definition at line 1168 of file gclib.cs.

9.42.3.158 t_plane_segment_count

`UW t_plane_segment_count`
segment count of coordinated move for T plane.
Definition at line 1167 of file gclib.cs.

9.42.3.159 thread_status

`UB thread_status`
thread status
Definition at line 1156 of file gclib.cs.
The documentation for this struct was generated from the following file:

- gclib.cs

9.43 GDataRecord52000 Struct Reference

Data record struct for DMC-52000 controller. Same as DMC-4000, with bank indicator added at byte 40.
`#include <gclib_record.h>`

Data Fields

- UB header_0
  1st Byte of Header.
- UB header_1
  2nd Byte of Header.
- UB header_2
  3rd Byte of Header.
- UB header_3
  4th Byte of Header.
- UW sample_number
  sample number.
- UB input_bank_0
  general input bank 0 (inputs 1-8).
- UB input_bank_1
  general input bank 1 (inputs 9-16).
- UB input_bank_2
  general input bank 2 (inputs 17-24).
- UB input_bank_3
  general input bank 3 (inputs 25-32).
- UB input_bank_4
  general input bank 4 (inputs 33-40).
- UB input_bank_5
  general input bank 5 (inputs 41-48).
- UB input_bank_6
  general input bank 6 (inputs 49-56).
- UB input_bank_7
  general input bank 7 (inputs 57-64).
- UB input_bank_8
  general input bank 8 (inputs 65-72).
- UB input_bank_9
  general input bank 9 (inputs 73-80).
- UB output_bank_0
  general output bank 0 (outputs 1-8).

- UB output_bank_1

  general output bank 1 (outputs 9-16).
- UB output_bank_2

  general output bank 2 (outputs 17-24).
- UB output_bank_3

  general output bank 3 (outputs 25-32).
- UB output_bank_4

  general output bank 4 (outputs 33-40).
- UB output_bank_5

  general output bank 5 (outputs 41-48).
- UB output_bank_6

  general output bank 6 (outputs 49-56).
- UB output_bank_7

  general output bank 7 (outputs 57-64).
- UB output_bank_8

  general output bank 8 (outputs 65-72).
- UB output_bank_9

  general output bank 9 (outputs 73-80).
- SW reserved_0

  Reserved.
- SW reserved_2

  Reserved.
- SW reserved_4

  Reserved.
- SW reserved_6

  Reserved.
- SW reserved_8

  Reserved.
- SW reserved_10

  Reserved.
- SW reserved_12

  Reserved.
- UB ethercat_bank

  EtherCAT Bank Indicator.
- UB reserved_14

  Reserved.
- UB ethernet_status_a

  Ethernet Handle A Status.
- UB ethernet_status_b

  Ethernet Handle B Status.
- UB ethernet_status_c

  Ethernet Handle C Status.
- UB ethernet_status_d

  Ethernet Handle D Status.
- UB ethernet_status_e

  Ethernet Handle E Status.
- UB ethernet_status_f

  Ethernet Handle F Status.
- UB ethernet_status_g

  Ethernet Handle G Status.
- UB ethernet_status_h

Ethernet Handle H Status.

- UB error_code

  error code.
- UB thread_status

  thread status
- UL amplifier_status

  Amplifier Status.
- UL contour_segment_count

  Segment Count for Contour Mode.
- UW contour_buffer_available

  Buffer space remaining, Contour Mode.
- UW s_plane_segment_count

  segment count of coordinated move for S plane.
- UW s_plane_move_status

  coordinated move status for S plane.
- SL s_distance

  distance traveled in coordinated move for S plane.
- UW s_plane_buffer_available

  Buffer space remaining, S Plane.
- UW t_plane_segment_count

  segment count of coordinated move for T plane.
- UW t_plane_move_status

  Coordinated move status for T plane.
- SL t_distance

  distance traveled in coordinated move for T plane.
- UW t_plane_buffer_available

  Buffer space remaining, T Plane.
- UW axis_a_status

  A axis status.
- UB axis_a_switches

  A axis switches.
- UB axis_a_stop_code

  A axis stop code.
- SL axis_a_reference_position

  A axis reference position.
- SL axis_a_motor_position

  A axis motor position.
- SL axis_a_position_error

  A axis position error.
- SL axis_a_aux_position

  A axis auxiliary position.
- SL axis_a_velocity

  A axis velocity.
- SL axis_a_torque

  A axis torque.
- UW axis_a_analog_in

  A axis analog input.
- UB axis_a_halls

  A Hall Input Status.
- UB axis_a_reserved

  Reserved.

- SL axis_a_variable

  A User-defined variable (ZA).
- UW axis_b_status

  B axis status.
- UB axis_b_switches

  B axis switches.
- UB axis_b_stop_code

  B axis stop code.
- SL axis_b_reference_position

  B axis reference position.
- SL axis_b_motor_position

  B axis motor position.
- SL axis_b_position_error

  B axis position error.
- SL axis_b_aux_position

  B axis auxiliary position.
- SL axis_b_velocity

  B axis velocity.
- SL axis_b_torque

  B axis torque.
- UW axis_b_analog_in

  B axis analog input.
- UB axis_b_halls

  B Hall Input Status.
- UB axis_b_reserved

  Reserved.
- SL axis_b_variable

  B User-defined variable (ZA).
- UW axis_c_status

  C axis status.
- UB axis_c_switches

  C axis switches.
- UB axis_c_stop_code

  C axis stop code.
- SL axis_c_reference_position

  C axis reference position.
- SL axis_c_motor_position

  C axis motor position.
- SL axis_c_position_error

  C axis position error.
- SL axis_c_aux_position

  C axis auxiliary position.
- SL axis_c_velocity

  C axis velocity.
- SL axis_c_torque

  C axis torque.
- UW axis_c_analog_in

  C axis analog input.
- UB axis_c_halls

  C Hall Input Status.
- UB axis_c_reserved

Reserved.

- SL axis_c_variable

  C User-defined variable (ZA).
- UW axis_d_status

  D axis status.
- UB axis_d_switches

  D axis switches.
- UB axis_d_stop_code

  D axis stop code.
- SL axis_d_reference_position

  D axis reference position.
- SL axis_d_motor_position

  D axis motor position.
- SL axis_d_position_error

  D axis position error.
- SL axis_d_aux_position

  D axis auxiliary position.
- SL axis_d_velocity

  D axis velocity.
- SL axis_d_torque

  D axis torque.
- UW axis_d_analog_in

  D axis analog input.
- UB axis_d_halls

  D Hall Input Status.
- UB axis_d_reserved

  Reserved.
- SL axis_d_variable

  D User-defined variable (ZA).
- UW axis_e_status

  E axis status.
- UB axis_e_switches

  E axis switches.
- UB axis_e_stop_code

  E axis stop code.
- SL axis_e_reference_position

  E axis reference position.
- SL axis_e_motor_position

  E axis motor position.
- SL axis_e_position_error

  E axis position error.
- SL axis_e_aux_position

  E axis auxiliary position.
- SL axis_e_velocity

  E axis velocity.
- SL axis_e_torque

  E axis torque.
- UW axis_e_analog_in

  E axis analog input.
- UB axis_e_halls

  E Hall Input Status.

- UB axis_e_reserved

  Reserved.
- SL axis_e_variable

  E User-defined variable (ZA).
- UW axis_f_status

  F axis status.
- UB axis_f_switches

  F axis switches.
- UB axis_f_stop_code

  F axis stop code.
- SL axis_f_reference_position

  F axis reference position.
- SL axis_f_motor_position

  F axis motor position.
- SL axis_f_position_error

  F axis position error.
- SL axis_f_aux_position

  F axis auxiliary position.
- SL axis_f_velocity

  F axis velocity.
- SL axis_f_torque

  F axis torque.
- UW axis_f_analog_in

  F axis analog input.
- UB axis_f_halls

  F Hall Input Status.
- UB axis_f_reserved

  Reserved.
- SL axis_f_variable

  F User-defined variable (ZA).
- UW axis_g_status

  G axis status.
- UB axis_g_switches

  G axis switches.
- UB axis_g_stop_code

  G axis stop code.
- SL axis_g_reference_position

  G axis reference position.
- SL axis_g_motor_position

  G axis motor position.
- SL axis_g_position_error

  G axis position error.
- SL axis_g_aux_position

  G axis auxiliary position.
- SL axis_g_velocity

  G axis velocity.
- SL axis_g_torque

  G axis torque.
- UW axis_g_analog_in

  G axis analog input.
- UB axis_g_halls

G Hall Input Status.

- UB axis_g_reserved

  Reserved.
- SL axis_g_variable

  G User-defined variable (ZA).
- UW axis_h_status

  H axis status.
- UB axis_h_switches

  H axis switches.
- UB axis_h_stop_code

  H axis stop code.
- SL axis_h_reference_position

  H axis reference position.
- SL axis_h_motor_position

  H axis motor position.
- SL axis_h_position_error

  H axis position error.
- SL axis_h_aux_position

  H axis auxiliary position.
- SL axis_h_velocity

  H axis velocity.
- SL axis_h_torque

  H axis torque.
- UW axis_h_analog_in

  H axis analog input.
- UB axis_h_halls

  H Hall Input Status.
- UB axis_h_reserved

  Reserved.
- SL axis_h_variable

  H User-defined variable (ZA).

9.43.1 Detailed Description

Data record struct for DMC-52000 controller. Same as DMC-4000, with bank indicator added at byte 40.
Definition at line 218 of file gclib_record.h.

9.43.2 Field Documentation

9.43.2.1 amplifier_status

```
UL amplifier_status
```

Amplifier Status.
Definition at line 273 of file gclib_record.h.

9.43.2.2 axis_a_analog_in

```
UW axis_a_analog_in
```

A axis analog input.
Definition at line 297 of file gclib_record.h.

9.43.2.3 axis_a_aux_position

```
SL axis_a_aux_position
```

A axis auxiliary position.
Definition at line 294 of file gclib_record.h.

9.43.2.4 axis_a_halls

`UB axis_a_halls`
A Hall Input Status.
Definition at line 298 of file gclib_record.h.

9.43.2.5 axis_a_motor_position

`SL axis_a_motor_position`
A axis motor position.
Definition at line 292 of file gclib_record.h.

9.43.2.6 axis_a_position_error

`SL axis_a_position_error`
A axis position error.
Definition at line 293 of file gclib_record.h.

9.43.2.7 axis_a_reference_position

`SL axis_a_reference_position`
A axis reference position.
Definition at line 291 of file gclib_record.h.

9.43.2.8 axis_a_reserved

`UB axis_a_reserved`
Reserved.
Definition at line 299 of file gclib_record.h.

9.43.2.9 axis_a_status

`UW axis_a_status`
A axis status.
Definition at line 288 of file gclib_record.h.

9.43.2.10 axis_a_stop_code

`UB axis_a_stop_code`
A axis stop code.
Definition at line 290 of file gclib_record.h.

9.43.2.11 axis_a_switches

`UB axis_a_switches`
A axis switches.
Definition at line 289 of file gclib_record.h.

9.43.2.12 axis_a_torque

`SL axis_a_torque`
A axis torque.
Definition at line 296 of file gclib_record.h.

9.43.2.13 axis_a_variable

`SL axis_a_variable`
A User-defined variable (ZA).
Definition at line 300 of file gclib_record.h.

9.43.2.14 axis_a_velocity

`SL axis_a_velocity`
A axis velocity.
Definition at line 295 of file gclib_record.h.

9.43.2.15 axis_b_analog_in

`UW axis_b_analog_in`
B axis analog input.
Definition at line 311 of file gclib_record.h.

9.43.2.16 axis_b_aux_position

`SL axis_b_aux_position`
B axis auxiliary position.
Definition at line 308 of file gclib_record.h.

9.43.2.17 axis_b_halls

`UB axis_b_halls`
B Hall Input Status.
Definition at line 312 of file gclib_record.h.

9.43.2.18 axis_b_motor_position

`SL axis_b_motor_position`
B axis motor position.
Definition at line 306 of file gclib_record.h.

9.43.2.19 axis_b_position_error

`SL axis_b_position_error`
B axis position error.
Definition at line 307 of file gclib_record.h.

9.43.2.20 axis_b_reference_position

`SL axis_b_reference_position`
B axis reference position.
Definition at line 305 of file gclib_record.h.

9.43.2.21 axis_b_reserved

`UB axis_b_reserved`
Reserved.
Definition at line 313 of file gclib_record.h.

9.43.2.22 axis_b_status

`UW axis_b_status`
B axis status.
Definition at line 302 of file gclib_record.h.

9.43.2.23 axis_b_stop_code

`UB axis_b_stop_code`
B axis stop code.
Definition at line 304 of file gclib_record.h.

9.43.2.24 axis_b_switches

`UB axis_b_switches`
B axis switches.
Definition at line 303 of file gclib_record.h.

9.43.2.25 axis_b_torque

`SL axis_b_torque`
B axis torque.
Definition at line 310 of file gclib_record.h.

9.43.2.26 axis_b_variable

`SL axis_b_variable`
B User-defined variable (ZA).
Definition at line 314 of file gclib_record.h.

9.43.2.27 axis_b_velocity

`SL axis_b_velocity`
B axis velocity.
Definition at line 309 of file gclib_record.h.

9.43.2.28 axis_c_analog_in

`UW axis_c_analog_in`
C axis analog input.
Definition at line 325 of file gclib_record.h.

9.43.2.29 axis_c_aux_position

`SL axis_c_aux_position`
C axis auxiliary position.
Definition at line 322 of file gclib_record.h.

9.43.2.30 axis_c_halls

`UB axis_c_halls`
C Hall Input Status.
Definition at line 326 of file gclib_record.h.

9.43.2.31 axis_c_motor_position

`SL axis_c_motor_position`
C axis motor position.
Definition at line 320 of file gclib_record.h.

9.43.2.32 axis_c_position_error

`SL axis_c_position_error`
C axis position error.
Definition at line 321 of file gclib_record.h.

9.43.2.33 axis_c_reference_position

`SL axis_c_reference_position`
C axis reference position.
Definition at line 319 of file gclib_record.h.

9.43.2.34 axis_c_reserved

`UB axis_c_reserved`
Reserved.
Definition at line 327 of file gclib_record.h.

9.43.2.35 axis_c_status

`UW axis_c_status`
C axis status.
Definition at line 316 of file gclib_record.h.

9.43.2.36 axis_c_stop_code

`UB axis_c_stop_code`
C axis stop code.
Definition at line 318 of file gclib_record.h.

9.43.2.37 axis_c_switches

`UB axis_c_switches`
C axis switches.
Definition at line 317 of file gclib_record.h.

9.43.2.38 axis_c_torque

`SL axis_c_torque`
C axis torque.
Definition at line 324 of file gclib_record.h.

9.43.2.39 axis_c_variable

`SL axis_c_variable`
C User-defined variable (ZA).
Definition at line 328 of file gclib_record.h.

9.43.2.40 axis_c_velocity

`SL axis_c_velocity`
C axis velocity.
Definition at line 323 of file gclib_record.h.

9.43.2.41 axis_d_analog_in

`UW axis_d_analog_in`
D axis analog input.
Definition at line 339 of file gclib_record.h.

9.43.2.42 axis_d_aux_position

`SL axis_d_aux_position`
D axis auxiliary position.
Definition at line 336 of file gclib_record.h.

9.43.2.43 axis_d_halls

`UB axis_d_halls`
D Hall Input Status.
Definition at line 340 of file gclib_record.h.

9.43.2.44 axis_d_motor_position

`SL axis_d_motor_position`
D axis motor position.
Definition at line 334 of file gclib_record.h.

9.43.2.45 axis_d_position_error

`SL axis_d_position_error`
D axis position error.
Definition at line 335 of file gclib_record.h.

9.43.2.46 axis_d_reference_position

`SL axis_d_reference_position`
D axis reference position.
Definition at line 333 of file gclib_record.h.

9.43.2.47 axis_d_reserved

`UB axis_d_reserved`
Reserved.
Definition at line 341 of file gclib_record.h.

9.43.2.48 axis_d_status

`UW axis_d_status`
D axis status.
Definition at line 330 of file gclib_record.h.

9.43.2.49 axis_d_stop_code

`UB axis_d_stop_code`
D axis stop code.
Definition at line 332 of file gclib_record.h.

9.43.2.50 axis_d_switches

`UB axis_d_switches`
D axis switches.
Definition at line 331 of file gclib_record.h.

9.43.2.51 axis_d_torque

`SL axis_d_torque`
D axis torque.
Definition at line 338 of file gclib_record.h.

9.43.2.52 axis_d_variable

`SL axis_d_variable`
D User-defined variable (ZA).
Definition at line 342 of file gclib_record.h.

9.43.2.53 axis_d_velocity

`SL axis_d_velocity`
D axis velocity.
Definition at line 337 of file gclib_record.h.

9.43.2.54 axis_e_analog_in

`UW axis_e_analog_in`
E axis analog input.
Definition at line 353 of file gclib_record.h.

9.43.2.55 axis_e_aux_position

`SL axis_e_aux_position`
E axis auxiliary position.
Definition at line 350 of file gclib_record.h.

9.43.2.56 axis_e_halls

`UB axis_e_halls`
E Hall Input Status.
Definition at line 354 of file gclib_record.h.

9.43.2.57 axis_e_motor_position

`SL axis_e_motor_position`
E axis motor position.
Definition at line 348 of file gclib_record.h.

9.43.2.58 axis_e_position_error

`SL axis_e_position_error`
E axis position error.
Definition at line 349 of file gclib_record.h.

9.43.2.59 axis_e_reference_position

`SL axis_e_reference_position`
E axis reference position.
Definition at line 347 of file gclib_record.h.

9.43.2.60 axis_e_reserved

`UB axis_e_reserved`
Reserved.
Definition at line 355 of file gclib_record.h.

9.43.2.61 axis_e_status

`UW axis_e_status`
E axis status.
Definition at line 344 of file gclib_record.h.

9.43.2.62 axis_e_stop_code

`UB axis_e_stop_code`
E axis stop code.
Definition at line 346 of file gclib_record.h.

9.43.2.63 axis_e_switches

`UB axis_e_switches`
E axis switches.
Definition at line 345 of file gclib_record.h.

9.43.2.64 axis_e_torque

`SL axis_e_torque`
E axis torque.
Definition at line 352 of file gclib_record.h.

9.43.2.65 axis_e_variable

`SL axis_e_variable`
E User-defined variable (ZA).
Definition at line 356 of file gclib_record.h.

9.43.2.66 axis_e_velocity

`SL axis_e_velocity`
E axis velocity.
Definition at line 351 of file gclib_record.h.

9.43.2.67 axis_f_analog_in

`UW axis_f_analog_in`
F axis analog input.
Definition at line 367 of file gclib_record.h.

9.43.2.68 axis_f_aux_position

`SL axis_f_aux_position`
F axis auxiliary position.
Definition at line 364 of file gclib_record.h.

9.43.2.69 axis_f_halls

`UB axis_f_halls`
F Hall Input Status.
Definition at line 368 of file gclib_record.h.

9.43.2.70 axis_f_motor_position

`SL axis_f_motor_position`
F axis motor position.
Definition at line 362 of file gclib_record.h.

9.43.2.71 axis_f_position_error

`SL axis_f_position_error`
F axis position error.
Definition at line 363 of file gclib_record.h.

9.43.2.72 axis_f_reference_position

`SL axis_f_reference_position`
F axis reference position.
Definition at line 361 of file gclib_record.h.

9.43.2.73 axis_f_reserved

`UB axis_f_reserved`
Reserved.
Definition at line 369 of file gclib_record.h.

9.43.2.74 axis_f_status

`UW axis_f_status`
F axis status.
Definition at line 358 of file gclib_record.h.

9.43.2.75 axis_f_stop_code

`UB axis_f_stop_code`
F axis stop code.
Definition at line 360 of file gclib_record.h.

9.43.2.76 axis_f_switches

`UB axis_f_switches`
F axis switches.
Definition at line 359 of file gclib_record.h.

9.43.2.77 axis_f_torque

`SL axis_f_torque`
F axis torque.
Definition at line 366 of file gclib_record.h.

9.43.2.78 axis_f_variable

`SL axis_f_variable`
F User-defined variable (ZA).
Definition at line 370 of file gclib_record.h.

9.43.2.79 axis_f_velocity

`SL axis_f_velocity`
F axis velocity.
Definition at line 365 of file gclib_record.h.

9.43.2.80 axis_g_analog_in

`UW axis_g_analog_in`
G axis analog input.
Definition at line 381 of file gclib_record.h.

9.43.2.81 axis_g_aux_position

`SL axis_g_aux_position`
G axis auxiliary position.
Definition at line 378 of file gclib_record.h.

9.43.2.82 axis_g_halls

`UB axis_g_halls`
G Hall Input Status.
Definition at line 382 of file gclib_record.h.

9.43.2.83 axis_g_motor_position

`SL axis_g_motor_position`
G axis motor position.
Definition at line 376 of file gclib_record.h.

9.43.2.84 axis_g_position_error

`SL axis_g_position_error`
G axis position error.
Definition at line 377 of file gclib_record.h.

9.43.2.85 axis_g_reference_position

`SL axis_g_reference_position`
G axis reference position.
Definition at line 375 of file gclib_record.h.

9.43.2.86 axis_g_reserved

`UB axis_g_reserved`
Reserved.
Definition at line 383 of file gclib_record.h.

9.43.2.87 axis_g_status

`UW axis_g_status`
G axis status.
Definition at line 372 of file gclib_record.h.

9.43.2.88 axis_g_stop_code

`UB axis_g_stop_code`
G axis stop code.
Definition at line 374 of file gclib_record.h.

9.43.2.89 axis_g_switches

`UB axis_g_switches`
G axis switches.
Definition at line 373 of file gclib_record.h.

9.43.2.90 axis_g_torque

`SL axis_g_torque`
G axis torque.
Definition at line 380 of file gclib_record.h.

9.43.2.91 axis_g_variable

`SL axis_g_variable`
G User-defined variable (ZA).
Definition at line 384 of file gclib_record.h.

9.43.2.92 axis_g_velocity

`SL axis_g_velocity`
G axis velocity.
Definition at line 379 of file gclib_record.h.

9.43.2.93 axis_h_analog_in

`UW axis_h_analog_in`
H axis analog input.
Definition at line 395 of file gclib_record.h.

9.43.2.94 axis_h_aux_position

`SL axis_h_aux_position`
H axis auxiliary position.
Definition at line 392 of file gclib_record.h.

9.43.2.95 axis_h_halls

`UB axis_h_halls`
H Hall Input Status.
Definition at line 396 of file gclib_record.h.

9.43.2.96 axis_h_motor_position

`SL axis_h_motor_position`
H axis motor position.
Definition at line 390 of file gclib_record.h.

9.43.2.97 axis_h_position_error

`SL axis_h_position_error`
H axis position error.
Definition at line 391 of file gclib_record.h.

9.43.2.98 axis_h_reference_position

`SL axis_h_reference_position`
H axis reference position.
Definition at line 389 of file gclib_record.h.

9.43.2.99 axis_h_reserved

`UB axis_h_reserved`
Reserved.
Definition at line 397 of file gclib_record.h.

9.43.2.100 axis_h_status

`UW axis_h_status`
H axis status.
Definition at line 386 of file gclib_record.h.

9.43.2.101 axis_h_stop_code

`UB axis_h_stop_code`
H axis stop code.
Definition at line 388 of file gclib_record.h.

9.43.2.102 axis_h_switches

`UB axis_h_switches`
H axis switches.
Definition at line 387 of file gclib_record.h.

9.43.2.103 axis_h_torque

`SL axis_h_torque`
H axis torque.
Definition at line 394 of file gclib_record.h.

9.43.2.104 axis_h_variable

`SL axis_h_variable`
H User-defined variable (ZA).
Definition at line 398 of file gclib_record.h.

9.43.2.105 axis_h_velocity

`SL axis_h_velocity`
H axis velocity.
Definition at line 393 of file gclib_record.h.

9.43.2.106 contour_buffer_available

`UW contour_buffer_available`
Buffer space remaining, Contour Mode.
Definition at line 276 of file gclib_record.h.

9.43.2.107 contour_segment_count

`UL contour_segment_count`
Segment Count for Contour Mode.
Definition at line 275 of file gclib_record.h.

9.43.2.108 error_code

`UB error_code`
error code.
Definition at line 271 of file gclib_record.h.

9.43.2.109 ethercat_bank

`UB ethercat_bank`
EtherCAT Bank Indicator.
Definition at line 259 of file gclib_record.h.

9.43.2.110 ethernet_status_a

`UB ethernet_status_a`
Ethernet Handle A Status.
Definition at line 262 of file gclib_record.h.

9.43.2.111 ethernet_status_b

`UB ethernet_status_b`
Ethernet Handle B Status.
Definition at line 263 of file gclib_record.h.

9.43.2.112 ethernet_status_c

`UB ethernet_status_c`
Ethernet Handle C Status.
Definition at line 264 of file gclib_record.h.

9.43.2.113 ethernet_status_d

`UB ethernet_status_d`
Ethernet Handle D Status.
Definition at line 265 of file gclib_record.h.

9.43.2.114 ethernet_status_e

`UB ethernet_status_e`
Ethernet Handle E Status.
Definition at line 266 of file gclib_record.h.

9.43.2.115 ethernet_status_f

`UB ethernet_status_f`
Ethernet Handle F Status.
Definition at line 267 of file gclib_record.h.

9.43.2.116 ethernet_status_g

`UB ethernet_status_g`
Ethernet Handle G Status.
Definition at line 268 of file gclib_record.h.

9.43.2.117 ethernet_status_h

`UB ethernet_status_h`
Ethernet Handle H Status.
Definition at line 269 of file gclib_record.h.

9.43.2.118 header_0

`UB header_0`
1st Byte of Header.
Definition at line 223 of file gclib_record.h.

9.43.2.119 header_1

`UB header_1`
2nd Byte of Header.
Definition at line 224 of file gclib_record.h.

9.43.2.120 header_2

`UB header_2`
3rd Byte of Header.
Definition at line 225 of file gclib_record.h.

9.43.2.121 header_3

`UB header_3`
4th Byte of Header.
Definition at line 226 of file gclib_record.h.

9.43.2.122 input_bank_0

`UB input_bank_0`
general input bank 0 (inputs 1-8).
Definition at line 230 of file gclib_record.h.

9.43.2.123 input_bank_1

`UB input_bank_1`
general input bank 1 (inputs 9-16).
Definition at line 231 of file gclib_record.h.

9.43.2.124 input_bank_2

`UB input_bank_2`
general input bank 2 (inputs 17-24).
Definition at line 232 of file gclib_record.h.

9.43.2.125 input_bank_3

`UB input_bank_3`
general input bank 3 (inputs 25-32).
Definition at line 233 of file gclib_record.h.

9.43.2.126 input_bank_4

`UB input_bank_4`
general input bank 4 (inputs 33-40).
Definition at line 234 of file gclib_record.h.

9.43.2.127 input_bank_5

`UB input_bank_5`
general input bank 5 (inputs 41-48).
Definition at line 235 of file gclib_record.h.

9.43.2.128 input_bank_6

`UB input_bank_6`
general input bank 6 (inputs 49-56).
Definition at line 236 of file gclib_record.h.

9.43.2.129 input_bank_7

`UB input_bank_7`
general input bank 7 (inputs 57-64).
Definition at line 237 of file gclib_record.h.

9.43.2.130 input_bank_8

`UB input_bank_8`
general input bank 8 (inputs 65-72).
Definition at line 238 of file gclib_record.h.

9.43.2.131 input_bank_9

`UB input_bank_9`
general input bank 9 (inputs 73-80).
Definition at line 239 of file gclib_record.h.

9.43.2.132 output_bank_0

`UB output_bank_0`
general output bank 0 (outputs 1-8).
Definition at line 241 of file gclib_record.h.

9.43.2.133 output_bank_1

`UB output_bank_1`
general output bank 1 (outputs 9-16).
Definition at line 242 of file gclib_record.h.

9.43.2.134 output_bank_2

`UB output_bank_2`
general output bank 2 (outputs 17-24).
Definition at line 243 of file gclib_record.h.

9.43.2.135 output_bank_3

`UB output_bank_3`
general output bank 3 (outputs 25-32).
Definition at line 244 of file gclib_record.h.

9.43.2.136 output_bank_4

`UB output_bank_4`
general output bank 4 (outputs 33-40).
Definition at line 245 of file gclib_record.h.

9.43.2.137 output_bank_5

`UB output_bank_5`
general output bank 5 (outputs 41-48).
Definition at line 246 of file gclib_record.h.

9.43.2.138 output_bank_6

`UB output_bank_6`
general output bank 6 (outputs 49-56).
Definition at line 247 of file gclib_record.h.

9.43.2.139 output_bank_7

`UB output_bank_7`
general output bank 7 (outputs 57-64).
Definition at line 248 of file gclib_record.h.

9.43.2.140 output_bank_8

`UB output_bank_8`
general output bank 8 (outputs 65-72).
Definition at line 249 of file gclib_record.h.

9.43.2.141 output_bank_9

`UB output_bank_9`
general output bank 9 (outputs 73-80).
Definition at line 250 of file gclib_record.h.

9.43.2.142 reserved_0

`SW reserved_0`
Reserved.
Definition at line 252 of file gclib_record.h.

9.43.2.143 reserved_10

`SW reserved_10`
Reserved.
Definition at line 257 of file gclib_record.h.

9.43.2.144 reserved_12

`SW reserved_12`
Reserved.
Definition at line 258 of file gclib_record.h.

9.43.2.145 reserved_14

`UB reserved_14`
Reserved.
Definition at line 260 of file gclib_record.h.

9.43.2.146 reserved_2

`SW reserved_2`
Reserved.
Definition at line 253 of file gclib_record.h.

9.43.2.147 reserved_4

`SW reserved_4`
Reserved.
Definition at line 254 of file gclib_record.h.

9.43.2.148 reserved_6

`SW reserved_6`
Reserved.
Definition at line 255 of file gclib_record.h.

9.43.2.149 reserved_8

`SW reserved_8`
Reserved.
Definition at line 256 of file gclib_record.h.

9.43.2.150 s_distance

`SL s_distance`
distance traveled in coordinated move for S plane.
Definition at line 280 of file gclib_record.h.

9.43.2.151 s_plane_buffer_available

`UW s_plane_buffer_available`
Buffer space remaining, S Plane.
Definition at line 281 of file gclib_record.h.

9.43.2.152 s_plane_move_status

`UW s_plane_move_status`
coordinated move status for S plane.
Definition at line 279 of file gclib_record.h.

9.43.2.153 s_plane_segment_count

`UW s_plane_segment_count`
segment count of coordinated move for S plane.
Definition at line 278 of file gclib_record.h.

9.43.2.154 sample_number

`UW sample_number`
sample number.
Definition at line 228 of file gclib_record.h.

9.43.2.155 t_distance

`SL t_distance`
distance traveled in coordinated move for T plane.
Definition at line 285 of file gclib_record.h.

9.43.2.156 t_plane_buffer_available

`UW t_plane_buffer_available`
Buffer space remaining, T Plane.
Definition at line 286 of file gclib_record.h.

9.43.2.157 t_plane_move_status

`UW t_plane_move_status`
Coordinated move status for T plane.
Definition at line 284 of file gclib_record.h.

9.43.2.158 t_plane_segment_count

`UW t_plane_segment_count`
segment count of coordinated move for T plane.
Definition at line 283 of file gclib_record.h.

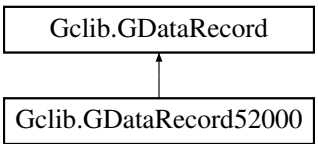
9.43.2.159 thread_status

`UB thread_status`
thread status
Definition at line 272 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

9.44 Gclib.GDataRecord52000 Struct Reference

Inheritance diagram for Gclib.GDataRecord52000:



Public Member Functions

- general input **bank** (inputs 1-8). ∗/public UB input_bank_1
  /∗07
- general input **bank** (inputs 9-16). ∗/public UB input_bank_2
  /∗08
- general input **bank** (inputs 17-24). ∗/public UB input_bank_3
  /∗09

- general input **bank** (inputs 25-32). ∗/public UB input_bank_4
  /∗10
- general input **bank** (inputs 33-40). ∗/public UB input_bank_5
  /∗11
- general input **bank** (inputs 41-48). ∗/public UB input_bank_6
  /∗12
- general input **bank** (inputs 49-56). ∗/public UB input_bank_7
  /∗13
- general input **bank** (inputs 57-64). ∗/public UB input_bank_8
  /∗14
- general input **bank** (inputs 65-72). ∗/public UB input_bank_9
  /∗15
- general input **bank** (inputs 73-80). ∗/public UB output_bank_0
  /∗16
- general output **bank** (outputs 1-8). ∗/public UB output_bank_1
  /∗17
- general output **bank** (outputs 9-16). ∗/public UB output_bank_2
  /∗18
- general output **bank** (outputs 17-24). ∗/public UB output_bank_3
  /∗19
- general output **bank** (outputs 25-32). ∗/public UB output_bank_4
  /∗20
- general output **bank** (outputs 33-40). ∗/public UB output_bank_5
  /∗21
- general output **bank** (outputs 41-48). ∗/public UB output_bank_6
  /∗22
- general output **bank** (outputs 49-56). ∗/public UB output_bank_7
  /∗23
- general output **bank** (outputs 57-64). ∗/public UB output_bank_8
  /∗24
- general output **bank** (outputs 65-72). ∗/public UB output_bank_9
  /∗25
- general output **bank** (outputs 73-80). ∗/public SW reserved_0
  /∗26-27
- H User defined variable.∗[] byte_array ()

Data Fields

- UB header_0
  /∗00
- byte of Header.∗UB header_1
  /∗01
- byte of Header.∗UB header_2
  /∗02
- byte of Header.∗UB header_3
  /∗03
- byte of Header.∗UW sample_number
  /∗04-05
- sample number.∗UB input_bank_0
  /∗06
- Reserved.∗SW reserved_2

/*28-29
- Reserved.*SW reserved_4

/*30-31
- Reserved.*SW reserved_6

/*32-33
- Reserved.*SW reserved_8

/*34-35
- Reserved.*SW reserved_10

/*36-37
- Reserved.*SW reserved_12

/*38-39
- Reserved.*UB ethercat_bank

/*40
- EtherCAT Bank Indicator.*UB reserved_14

/*41
- Reserved.*UB ethernet_status_a

/*42
- Ethernet Handle A Status.*UB ethernet_status_b

/*43
- Ethernet Handle B Status.*UB ethernet_status_c

/*44
- Ethernet Handle C Status.*UB ethernet_status_d

/*45
- Ethernet Handle D Status.*UB ethernet_status_e

/*46
- Ethernet Handle E Status.*UB ethernet_status_f

/*47
- Ethernet Handle F Status.*UB ethernet_status_g

/*48
- Ethernet Handle G Status.*UB ethernet_status_h

/*49
- Ethernet Handle H Status.*UB error_code

/*50
- error code.*UB thread_status

/*51
- thread status *UL amplifier_status

/*52-55
- Amplifier Status.*UL contour_segment_count

/*56-59
- Segment Count for Contour Mode.*UW contour_buffer_available

/*60-61
- Buffer space remaining
- Buffer space Contour Mode.*UW s_plane_segment_count

/*62-63
- segment count of coordinated move for S plane.*UW s_plane_move_status

/*64-65
- coordinated move status for S plane.*SL s_distance

/*66-69
- distance traveled in coordinated move for S plane.*UW s_plane_buffer_available

/*70-71
- Buffer space S Plane.*UW t_plane_segment_count

/*72-73

- segment count of coordinated move for T plane.*UW t_plane_move_status

/*74-75

- Coordinated move status for T plane.*SL t_distance

/*76-79

- distance traveled in coordinated move for T plane.*UW t_plane_buffer_available

/*80-81

- Buffer space T Plane.*UW axis_a_status

/*82-83

- A axis status.*UB axis_a_switches

/*84

- A axis switches.*UB axis_a_stop_code

/*85

- A axis stop code.*SL axis_a_reference_position

/*86-89

- A axis reference position.*SL axis_a_motor_position

/*90-93

- A axis motor position.*SL axis_a_position_error

/*94-97

- A axis position error.*SL axis_a_aux_position

/*98-101

- A axis auxiliary position.*SL axis_a_velocity

/*102-105

- A axis velocity.*SL axis_a_torque

/*106-109

- A axis torque.*UW axis_a_analog_in

/*110-111

- A axis analog input.*UB axis_a_halls

/*112

- A Hall Input Status.*UB axis_a_reserved

/*113

- Reserved.*SL axis_a_variable

/*114-117

- A User defined variable.*[] UW axis_b_status

/*118-119

- B axis status.*UB axis_b_switches

/*120

- B axis switches.*UB axis_b_stop_code

/*121

- B axis stop code.*SL axis_b_reference_position

/*122-125

- B axis reference position.*SL axis_b_motor_position

/*126-129

- B axis motor position.*SL axis_b_position_error

/*130-133

- B axis position error.*SL axis_b_aux_position

/*134-137

- B axis auxiliary position.*SL axis_b_velocity

/*138-141

- B axis velocity.*SL axis_b_torque

/*142-145

- B axis torque.∗UW axis_b_analog_in
  /∗146-147
- B axis analog input.∗UB axis_b_halls
  /∗148
- B Hall Input Status.∗UB axis_b_reserved
  /∗149
- Reserved.∗SL axis_b_variable
  /∗150-153
- B User defined variable.∗[] UW axis_c_status
  /∗154-155
- C axis status.∗UB axis_c_switches
  /∗156
- C axis switches.∗UB axis_c_stop_code
  /∗157
- C axis stop code.∗SL axis_c_reference_position
  /∗158-161
- C axis reference position.∗SL axis_c_motor_position
  /∗162-165
- C axis motor position.∗SL axis_c_position_error
  /∗166-169
- C axis position error.∗SL axis_c_aux_position
  /∗170-173
- C axis auxiliary position.∗SL axis_c_velocity
  /∗174-177
- C axis velocity.∗SL axis_c_torque
  /∗178-181
- C axis torque.∗UW axis_c_analog_in
  /∗182-183
- C axis analog input.∗UB axis_c_halls
  /∗184
- C Hall Input Status.∗UB axis_c_reserved
  /∗185
- Reserved.∗SL axis_c_variable
  /∗186-189
- C User defined variable.∗[] UW axis_d_status
  /∗190-191
- D axis status.∗UB axis_d_switches
  /∗192
- D axis switches.∗UB axis_d_stop_code
  /∗193
- D axis stop code.∗SL axis_d_reference_position
  /∗194-197
- D axis reference position.∗SL axis_d_motor_position
  /∗198-201
- D axis motor position.∗SL axis_d_position_error
  /∗202-205
- D axis position error.∗SL axis_d_aux_position
  /∗206-209
- D axis auxiliary position.∗SL axis_d_velocity
  /∗210-213
- D axis velocity.∗SL axis_d_torque

/*214-217*

- D axis torque.*UW axis_d_analog_in

  /*218-219*
- D axis analog input.*UB axis_d_halls

  /*220*
- D Hall Input Status.*UB axis_d_reserved

  /*221*
- Reserved.*SL axis_d_variable

  /*222-225*
- D User defined variable.*[] UW axis_e_status

  /*226-227*
- E axis status.*UB axis_e_switches

  /*228*
- E axis switches.*UB axis_e_stop_code

  /*229*
- E axis stop code.*SL axis_e_reference_position

  /*230-233*
- E axis reference position.*SL axis_e_motor_position

  /*234-237*
- E axis motor position.*SL axis_e_position_error

  /*238-241*
- E axis position error.*SL axis_e_aux_position

  /*242-245*
- E axis auxiliary position.*SL axis_e_velocity

  /*246-249*
- E axis velocity.*SL axis_e_torque

  /*250-253*
- E axis torque.*UW axis_e_analog_in

  /*254-255*
- E axis analog input.*UB axis_e_halls

  /*256*
- E Hall Input Status.*UB axis_e_reserved

  /*257*
- Reserved.*SL axis_e_variable

  /*258-261*
- E User defined variable.*[] UW axis_f_status

  /*262-263*
- F axis status.*UB axis_f_switches

  /*264*
- F axis switches.*UB axis_f_stop_code

  /*265*
- F axis stop code.*SL axis_f_reference_position

  /*266-269*
- F axis reference position.*SL axis_f_motor_position

  /*270-273*
- F axis motor position.*SL axis_f_position_error

  /*274-277*
- F axis position error.*SL axis_f_aux_position

  /*278-281*
- F axis auxiliary position.*SL axis_f_velocity

  /*282-285*

- F axis velocity.∗SL axis_f_torque
  /∗286-289
- F axis torque.∗UW axis_f_analog_in
  /∗290-291
- F axis analog input.∗UB axis_f_halls
  /∗292
- F Hall Input Status.∗UB axis_f_reserved
  /∗293
- Reserved.∗SL axis_f_variable
  /∗294-297
- F User defined variable.∗[] UW axis_g_status
  /∗298-299
- G axis status.∗UB axis_g_switches
  /∗300
- G axis switches.∗UB axis_g_stop_code
  /∗301
- G axis stop code.∗SL axis_g_reference_position
  /∗302-305
- G axis reference position.∗SL axis_g_motor_position
  /∗306-309
- G axis motor position.∗SL axis_g_position_error
  /∗310-313
- G axis position error.∗SL axis_g_aux_position
  /∗314-317
- G axis auxiliary position.∗SL axis_g_velocity
  /∗318-321
- G axis velocity.∗SL axis_g_torque
  /∗322-325
- G axis torque.∗UW axis_g_analog_in
  /∗326-327
- G axis analog input.∗UB axis_g_halls
  /∗328
- G Hall Input Status.∗UB axis_g_reserved
  /∗329
- Reserved.∗SL axis_g_variable
  /∗330-333
- G User defined variable.∗[] UW axis_h_status
  /∗334-335
- H axis status.∗UB axis_h_switches
  /∗336
- H axis switches.∗UB axis_h_stop_code
  /∗337
- H axis stop code.∗SL axis_h_reference_position
  /∗338-341
- H axis reference position.∗SL axis_h_motor_position
  /∗342-345
- H axis motor position.∗SL axis_h_position_error
  /∗346-349
- H axis position error.∗SL axis_h_aux_position
  /∗350-353
- H axis auxiliary position.∗SL axis_h_velocity

/*354-357

- H axis velocity.*SL axis_h_torque

  /*358-361

- H axis torque.*UW axis_h_analog_in

  /*362-363

- H axis analog input.*UB axis_h_halls

  /*364

- H Hall Input Status.*UB axis_h_reserved

  /*365

- Reserved.*SL axis_h_variable

  /*366-369

9.44.1 Detailed Description

Definition at line 635 of file gclib.vb.

9.44.2 Member Function Documentation

9.44.2.1 byte_array()

```
H User defined variable.*[] byte_array ( )
```

Implements GDataRecord.byte_array
Implements Gclib.GDataRecord.

9.44.3 Field Documentation

9.44.3.1 amplifier_status

```
thread status* UL amplifier_status
```

/*52-55
Definition at line 682 of file gclib.vb.

9.44.3.2 axis_a_analog_in

```
A axis torque.* UW axis_a_analog_in
```

/*110-111
Definition at line 702 of file gclib.vb.

9.44.3.3 axis_a_aux_position

```
A axis position error.* SL axis_a_aux_position
```

/*98-101
Definition at line 699 of file gclib.vb.

9.44.3.4 axis_a_halls

```
A axis analog input.* UB axis_a_halls
```

/*112
Definition at line 703 of file gclib.vb.

9.44.3.5 axis_a_motor_position

```
A axis reference position.* SL axis_a_motor_position
```

/*90-93
Definition at line 697 of file gclib.vb.

9.44.3.6 axis_a_position_error

```
A axis motor position.* SL axis_a_position_error
```
/*94-97
Definition at line 698 of file gclib.vb.

9.44.3.7 axis_a_reference_position

```
A axis stop code.* SL axis_a_reference_position
```
/*86-89
Definition at line 696 of file gclib.vb.

9.44.3.8 axis_a_reserved

```
A Hall Input Status.* UB axis_a_reserved
```
/*113
Definition at line 704 of file gclib.vb.

9.44.3.9 axis_a_status

```
Buffer space T Plane.* UW axis_a_status
```
/*82-83
Definition at line 693 of file gclib.vb.

9.44.3.10 axis_a_stop_code

```
A axis switches.* UB axis_a_stop_code
```
/*85
Definition at line 695 of file gclib.vb.

9.44.3.11 axis_a_switches

```
A axis status.* UB axis_a_switches
```
/*84
Definition at line 694 of file gclib.vb.

9.44.3.12 axis_a_torque

```
A axis velocity.* SL axis_a_torque
```
/*106-109
Definition at line 701 of file gclib.vb.

9.44.3.13 axis_a_variable

```
Reserved.* SL axis_a_variable
```
/*114-117
Definition at line 705 of file gclib.vb.

9.44.3.14 axis_a_velocity

```
A axis auxiliary position.* SL axis_a_velocity
```
/*102-105
Definition at line 700 of file gclib.vb.

9.44.3.15 axis_b_analog_in

```
B axis torque.* UW axis_b_analog_in
```
/*146-147
Definition at line 715 of file gclib.vb.

9.44.3.16 axis_b_aux_position

```
B axis position error.* SL axis_b_aux_position
```
/*134-137
Definition at line 712 of file gclib.vb.

9.44.3.17 axis_b_halls

```
B axis analog input.* UB axis_b_halls
```
/*148
Definition at line 716 of file gclib.vb.

9.44.3.18 axis_b_motor_position

```
B axis reference position.* SL axis_b_motor_position
```
/*126-129
Definition at line 710 of file gclib.vb.

9.44.3.19 axis_b_position_error

```
B axis motor position.* SL axis_b_position_error
```
/*130-133
Definition at line 711 of file gclib.vb.

9.44.3.20 axis_b_reference_position

```
B axis stop code.* SL axis_b_reference_position
```
/*122-125
Definition at line 709 of file gclib.vb.

9.44.3.21 axis_b_reserved

```
B Hall Input Status.* UB axis_b_reserved
```
/*149
Definition at line 717 of file gclib.vb.

9.44.3.22 axis_b_status

```
A User defined variable.* [] UW axis_b_status
```
/*118-119
Definition at line 706 of file gclib.vb.

9.44.3.23 axis_b_stop_code

```
B axis switches.* UB axis_b_stop_code
```
/*121
Definition at line 708 of file gclib.vb.

9.44.3.24 axis_b_switches

```
B axis status.* UB axis_b_switches
```
/*120
Definition at line 707 of file gclib.vb.

9.44.3.25 axis_b_torque

```
B axis velocity.* SL axis_b_torque
```
/*142-145
Definition at line 714 of file gclib.vb.

9.44.3.26 axis_b_variable

```
Reserved.* SL axis_b_variable
```
/*150-153
Definition at line 718 of file gclib.vb.

9.44.3.27 axis_b_velocity

```
B axis auxiliary position.* SL axis_b_velocity
```
/*138-141
Definition at line 713 of file gclib.vb.

9.44.3.28 axis_c_analog_in

```
C axis torque.* UW axis_c_analog_in
```
/*182-183
Definition at line 728 of file gclib.vb.

9.44.3.29 axis_c_aux_position

```
C axis position error.* SL axis_c_aux_position
```
/*170-173
Definition at line 725 of file gclib.vb.

9.44.3.30 axis_c_halls

```
C axis analog input.* UB axis_c_halls
```
/*184
Definition at line 729 of file gclib.vb.

9.44.3.31 axis_c_motor_position

```
C axis reference position.* SL axis_c_motor_position
```
/*162-165
Definition at line 723 of file gclib.vb.

9.44.3.32 axis_c_position_error

```
C axis motor position.* SL axis_c_position_error
```
/*166-169
Definition at line 724 of file gclib.vb.

9.44.3.33 axis_c_reference_position

```
C axis stop code.* SL axis_c_reference_position
```
/*158-161
Definition at line 722 of file gclib.vb.

9.44.3.34 axis_c_reserved

```
C Hall Input Status.* UB axis_c_reserved
```
/*185
Definition at line 730 of file gclib.vb.

9.44.3.35 axis_c_status

```
B User defined variable.* [] UW axis_c_status
```
/*154-155
Definition at line 719 of file gclib.vb.

9.44.3.36 axis_c_stop_code

```
C axis switches.* UB axis_c_stop_code
```
/*157
Definition at line 721 of file gclib.vb.

9.44.3.37 axis_c_switches

```
C axis status.* UB axis_c_switches
```
/*156
Definition at line 720 of file gclib.vb.

9.44.3.38 axis_c_torque

```
C axis velocity.* SL axis_c_torque
```
/*178-181
Definition at line 727 of file gclib.vb.

9.44.3.39 axis_c_variable

```
Reserved.* SL axis_c_variable
```
/*186-189
Definition at line 731 of file gclib.vb.

9.44.3.40 axis_c_velocity

```
C axis auxiliary position.* SL axis_c_velocity
```
/*174-177
Definition at line 726 of file gclib.vb.

9.44.3.41 axis_d_analog_in

```
D axis torque.* UW axis_d_analog_in
```
/*218-219
Definition at line 741 of file gclib.vb.

9.44.3.42 axis_d_aux_position

```
D axis position error.* SL axis_d_aux_position
```
/*206-209
Definition at line 738 of file gclib.vb.

9.44.3.43 axis_d_halls

```
D axis analog input.* UB axis_d_halls
```
/*220
Definition at line 742 of file gclib.vb.

9.44.3.44 axis_d_motor_position

```
D axis reference position.* SL axis_d_motor_position
```
/*198-201
Definition at line 736 of file gclib.vb.

9.44.3.45 axis_d_position_error

```
D axis motor position.* SL axis_d_position_error
```
/*202-205
Definition at line 737 of file gclib.vb.

9.44.3.46 axis_d_reference_position

```
D axis stop code.* SL axis_d_reference_position
```
/*194-197
Definition at line 735 of file gclib.vb.

9.44.3.47 axis_d_reserved

```
D Hall Input Status.* UB axis_d_reserved
```
/*221
Definition at line 743 of file gclib.vb.

9.44.3.48 axis_d_status

```
C User defined variable.* [] UW axis_d_status
```
/*190-191
Definition at line 732 of file gclib.vb.

9.44.3.49 axis_d_stop_code

```
D axis switches.* UB axis_d_stop_code
```
/*193
Definition at line 734 of file gclib.vb.

9.44.3.50 axis_d_switches

```
D axis status.* UB axis_d_switches
```
/*192
Definition at line 733 of file gclib.vb.

9.44.3.51 axis_d_torque

```
D axis velocity.* SL axis_d_torque
```
/*214-217
Definition at line 740 of file gclib.vb.

9.44.3.52 axis_d_variable

```
Reserved.* SL axis_d_variable
```
/*222-225
Definition at line 744 of file gclib.vb.

9.44.3.53 axis_d_velocity

```
D axis auxiliary position.* SL axis_d_velocity
```
/*210-213
Definition at line 739 of file gclib.vb.

9.44.3.54 axis_e_analog_in

```
E axis torque.* UW axis_e_analog_in
```
/*254-255
Definition at line 754 of file gclib.vb.

9.44.3.55 axis_e_aux_position

```
E axis position error.* SL axis_e_aux_position
```
/*242-245
Definition at line 751 of file gclib.vb.

9.44.3.56 axis_e_halls

```
E axis analog input.* UB axis_e_halls
```
/*256
Definition at line 755 of file gclib.vb.

9.44.3.57 axis_e_motor_position

```
E axis reference position.* SL axis_e_motor_position
```
/*234-237
Definition at line 749 of file gclib.vb.

9.44.3.58 axis_e_position_error

```
E axis motor position.* SL axis_e_position_error
```
/*238-241
Definition at line 750 of file gclib.vb.

9.44.3.59 axis_e_reference_position

```
E axis stop code.* SL axis_e_reference_position
```
/*230-233
Definition at line 748 of file gclib.vb.

9.44.3.60 axis_e_reserved

```
E Hall Input Status.* UB axis_e_reserved
```
/*257
Definition at line 756 of file gclib.vb.

9.44.3.61 axis_e_status

```
D User defined variable.* [] UW axis_e_status
```
/*226-227
Definition at line 745 of file gclib.vb.

9.44.3.62 axis_e_stop_code

```
E axis switches.* UB axis_e_stop_code
```
/*229
Definition at line 747 of file gclib.vb.

9.44.3.63 axis_e_switches

```
E axis status.* UB axis_e_switches
```
/*228
Definition at line 746 of file gclib.vb.

9.44.3.64 axis_e_torque

```
E axis velocity.* SL axis_e_torque
```
/*250-253
Definition at line 753 of file gclib.vb.

9.44.3.65 axis_e_variable

```
Reserved.* SL axis_e_variable
```
/*258-261
Definition at line 757 of file gclib.vb.

9.44.3.66 axis_e_velocity

```
E axis auxiliary position.* SL axis_e_velocity
```
/*246-249
Definition at line 752 of file gclib.vb.

9.44.3.67 axis_f_analog_in

```
F axis torque.* UW axis_f_analog_in
```
/*290-291
Definition at line 767 of file gclib.vb.

9.44.3.68 axis_f_aux_position

```
F axis position error.* SL axis_f_aux_position
```
/*278-281
Definition at line 764 of file gclib.vb.

9.44.3.69 axis_f_halls

```
F axis analog input.* UB axis_f_halls
```
/*292
Definition at line 768 of file gclib.vb.

9.44.3.70 axis_f_motor_position

```
F axis reference position.* SL axis_f_motor_position
```
/*270-273
Definition at line 762 of file gclib.vb.

9.44.3.71 axis_f_position_error

```
F axis motor position.* SL axis_f_position_error
```
/*274-277
Definition at line 763 of file gclib.vb.

9.44.3.72 axis_f_reference_position

```
F axis stop code.* SL axis_f_reference_position
```
/*266-269
Definition at line 761 of file gclib.vb.

9.44.3.73 axis_f_reserved

```
F Hall Input Status.* UB axis_f_reserved
```
/*293
Definition at line 769 of file gclib.vb.

9.44.3.74 axis_f_status

```
E User defined variable.* [] UW axis_f_status
```
/*262-263
Definition at line 758 of file gclib.vb.

9.44.3.75 axis_f_stop_code

```
F axis switches.* UB axis_f_stop_code
```
/*265
Definition at line 760 of file gclib.vb.

9.44.3.76 axis_f_switches

```
F axis status.* UB axis_f_switches
```
/*264
Definition at line 759 of file gclib.vb.

9.44.3.77 axis_f_torque

```
F axis velocity.* SL axis_f_torque
```
/*286-289
Definition at line 766 of file gclib.vb.

9.44.3.78 axis_f_variable

```
Reserved.* SL axis_f_variable
```
/*294-297
Definition at line 770 of file gclib.vb.

9.44.3.79 axis_f_velocity

```
F axis auxiliary position.* SL axis_f_velocity
```
/*282-285
Definition at line 765 of file gclib.vb.

9.44.3.80 axis_g_analog_in

```
G axis torque.* UW axis_g_analog_in
```
/*326-327
Definition at line 780 of file gclib.vb.

9.44.3.81 axis_g_aux_position

```
G axis position error.* SL axis_g_aux_position
```
/*314-317
Definition at line 777 of file gclib.vb.

9.44.3.82 axis_g_halls

```
G axis analog input.* UB axis_g_halls
```
/*328
Definition at line 781 of file gclib.vb.

9.44.3.83 axis_g_motor_position

```
G axis reference position.* SL axis_g_motor_position
```
/*306-309
Definition at line 775 of file gclib.vb.

9.44.3.84 axis_g_position_error

```
G axis motor position.* SL axis_g_position_error
```
/*310-313
Definition at line 776 of file gclib.vb.

9.44.3.85 axis_g_reference_position

```
G axis stop code.* SL axis_g_reference_position
```
/*302-305
Definition at line 774 of file gclib.vb.

9.44.3.86 axis_g_reserved

```
G Hall Input Status.* UB axis_g_reserved
```
/*329
Definition at line 782 of file gclib.vb.

9.44.3.87 axis_g_status

```
F User defined variable.* [] UW axis_g_status
```
/*298-299
Definition at line 771 of file gclib.vb.

9.44.3.88 axis_g_stop_code

```
G axis switches.* UB axis_g_stop_code
```
/*301
Definition at line 773 of file gclib.vb.

9.44.3.89 axis_g_switches

```
G axis status.* UB axis_g_switches
```
/*300
Definition at line 772 of file gclib.vb.

9.44.3.90 axis_g_torque

```
G axis velocity.* SL axis_g_torque
```
/*322-325
Definition at line 779 of file gclib.vb.

9.44.3.91 axis_g_variable

```
Reserved.* SL axis_g_variable
```
/*330-333
Definition at line 783 of file gclib.vb.

9.44.3.92 axis_g_velocity

```
G axis auxiliary position.* SL axis_g_velocity
```
/*318-321
Definition at line 778 of file gclib.vb.

9.44.3.93 axis_h_analog_in

```
H axis torque.* UW axis_h_analog_in
```
/*362-363
Definition at line 793 of file gclib.vb.

9.44.3.94 axis_h_aux_position

```
H axis position error.* SL axis_h_aux_position
```
/*350-353
Definition at line 790 of file gclib.vb.

9.44.3.95 axis_h_halls

```
H axis analog input.* UB axis_h_halls
```
/*364
Definition at line 794 of file gclib.vb.

9.44.3.96 axis_h_motor_position

```
H axis reference position.* SL axis_h_motor_position
```
/*342-345
Definition at line 788 of file gclib.vb.

9.44.3.97 axis_h_position_error

```
H axis motor position.* SL axis_h_position_error
```
/*346-349
Definition at line 789 of file gclib.vb.

9.44.3.98 axis_h_reference_position

```
H axis stop code.* SL axis_h_reference_position
```
/*338-341
Definition at line 787 of file gclib.vb.

9.44.3.99 axis_h_reserved

```
H Hall Input Status.* UB axis_h_reserved
```
/*365
Definition at line 795 of file gclib.vb.

9.44.3.100 axis_h_status

```
G User defined variable.* [] UW axis_h_status
```
/*334-335
Definition at line 784 of file gclib.vb.

9.44.3.101 axis_h_stop_code

```
H axis switches.* UB axis_h_stop_code
```
/*337
Definition at line 786 of file gclib.vb.

9.44.3.102 axis_h_switches

```
H axis status.* UB axis_h_switches
```
/*336
Definition at line 785 of file gclib.vb.

9.44.3.103 axis_h_torque

```
H axis velocity.* SL axis_h_torque
```
/*358-361
Definition at line 792 of file gclib.vb.

9.44.3.104 axis_h_variable

```
Reserved.* SL axis_h_variable
```
/*366-369
Definition at line 796 of file gclib.vb.

9.44.3.105 axis_h_velocity

```
H axis auxiliary position.* SL axis_h_velocity
```
/*354-357
Definition at line 791 of file gclib.vb.

9.44.3.106 contour_buffer_available

```
Segment Count for Contour Mode.* UW contour_buffer_available
```

/*60-61
Definition at line 684 of file gclib.vb.

9.44.3.107 contour_segment_count

```
Amplifier Status.* UL contour_segment_count
```

/*56-59
Definition at line 683 of file gclib.vb.

9.44.3.108 error_code

```
Ethernet Handle H Status.* UB error_code
```

/*50
Definition at line 680 of file gclib.vb.

9.44.3.109 ethercat_bank

```
Reserved.* UB ethercat_bank
```

/*40
Definition at line 670 of file gclib.vb.

9.44.3.110 ethernet_status_a

```
Reserved.* UB ethernet_status_a
```

/*42
Definition at line 672 of file gclib.vb.

9.44.3.111 ethernet_status_b

```
Ethernet Handle A Status.* UB ethernet_status_b
```

/*43
Definition at line 673 of file gclib.vb.

9.44.3.112 ethernet_status_c

```
Ethernet Handle B Status.* UB ethernet_status_c
```

/*44
Definition at line 674 of file gclib.vb.

9.44.3.113 ethernet_status_d

```
Ethernet Handle C Status.* UB ethernet_status_d
```

/*45
Definition at line 675 of file gclib.vb.

9.44.3.114 ethernet_status_e

```
Ethernet Handle D Status.* UB ethernet_status_e
```

/*46
Definition at line 676 of file gclib.vb.

9.44.3.115 ethernet_status_f

```
Ethernet Handle E Status.* UB ethernet_status_f
```

/*47
Definition at line 677 of file gclib.vb.

9.44.3.116 ethernet_status_g

```
Ethernet Handle F Status.* UB ethernet_status_g
```

/*48
Definition at line 678 of file gclib.vb.

9.44.3.117 ethernet_status_h

```
Ethernet Handle G Status.* UB ethernet_status_h
```

/*49
Definition at line 679 of file gclib.vb.

9.44.3.118 header_0

```
UB header_0
```

/*00
Definition at line 638 of file gclib.vb.

9.44.3.119 header_1

```
byte of Header.* UB header_1
```

/*01
Definition at line 639 of file gclib.vb.

9.44.3.120 header_2

```
byte of Header.* UB header_2
```

/*02
Definition at line 640 of file gclib.vb.

9.44.3.121 header_3

```
byte of Header.* UB header_3
```

/*03
Definition at line 641 of file gclib.vb.

9.44.3.122 input_bank_0

```
sample number.* UB input_bank_0
```

/*06
Definition at line 643 of file gclib.vb.

9.44.3.123 remaining

```
Buffer space remaining
```

Definition at line 684 of file gclib.vb.

9.44.3.124 reserved_10

```
Reserved.* SW reserved_10
```

/*36-37
Definition at line 668 of file gclib.vb.

9.44.3.125 reserved_12

```
Reserved.* SW reserved_12
```

/*38-39
Definition at line 669 of file gclib.vb.

9.44.3.126 reserved_14

```
EtherCAT Bank Indicator.* UB reserved_14
```
/*41
Definition at line 671 of file gclib.vb.

9.44.3.127 reserved_2

```
Reserved.* SW reserved_2
```
/*28-29
Definition at line 664 of file gclib.vb.

9.44.3.128 reserved_4

```
Reserved.* SW reserved_4
```
/*30-31
Definition at line 665 of file gclib.vb.

9.44.3.129 reserved_6

```
Reserved.* SW reserved_6
```
/*32-33
Definition at line 666 of file gclib.vb.

9.44.3.130 reserved_8

```
Reserved.* SW reserved_8
```
/*34-35
Definition at line 667 of file gclib.vb.

9.44.3.131 s_distance

```
coordinated move status for S plane.* SL s_distance
```
/*66-69
Definition at line 687 of file gclib.vb.

9.44.3.132 s_plane_buffer_available

```
distance traveled in coordinated move for S plane.* UW s_plane_buffer_available
```
/*70-71
Definition at line 688 of file gclib.vb.

9.44.3.133 s_plane_move_status

```
segment count of coordinated move for S plane.* UW s_plane_move_status
```
/*64-65
Definition at line 686 of file gclib.vb.

9.44.3.134 s_plane_segment_count

```
Buffer space Contour Mode.* UW s_plane_segment_count
```
/*62-63
Definition at line 685 of file gclib.vb.

9.44.3.135 sample_number

```
byte of Header.* UW sample_number
```
/*04-05
Definition at line 642 of file gclib.vb.

9.44.3.136 t_distance

```
Coordinated move status for T plane.* SL t_distance
```
/*76-79
Definition at line 691 of file gclib.vb.

9.44.3.137 t_plane_buffer_available

```
distance traveled in coordinated move for T plane.* UW t_plane_buffer_available
```
/*80-81
Definition at line 692 of file gclib.vb.

9.44.3.138 t_plane_move_status

```
segment count of coordinated move for T plane.* UW t_plane_move_status
```
/*74-75
Definition at line 690 of file gclib.vb.

9.44.3.139 t_plane_segment_count

```
Buffer space S Plane.* UW t_plane_segment_count
```
/*72-73
Definition at line 689 of file gclib.vb.

9.44.3.140 thread_status

```
error code.* UB thread_status
```
/*51
Definition at line 681 of file gclib.vb.
The documentation for this struct was generated from the following file:

- gclib.vb

9.45 H_ArrayData Struct Reference

Structure to create a linked list for array data.

Data Fields

- char name [16]
- char * data
- int len
- int elements
- int index
- struct H_ArrayData * next
- struct H_ArrayData * tail
- int count

9.45.1 Detailed Description

Structure to create a linked list for array data.
Definition at line 20 of file arrays.c.

9.45.2 Field Documentation

9.45.2.1 count

```
int count
```
Definition at line 32 of file arrays.c.

9.45.2.2 data

```
char* data
```
Definition at line 23 of file arrays.c.

9.45.2.3 elements

```
int elements
```
Definition at line 25 of file arrays.c.

9.45.2.4 index

```
int index
```
Definition at line 26 of file arrays.c.

9.45.2.5 len

```
int len
```
Definition at line 24 of file arrays.c.

9.45.2.6 name

```
char name[16]
```
Definition at line 22 of file arrays.c.

9.45.2.7 next

```
struct H_ArrayData* next
```
Definition at line 28 of file arrays.c.

9.45.2.8 tail

```
struct H_ArrayData* tail
```
Definition at line 31 of file arrays.c.
The documentation for this struct was generated from the following file:

- arrays.c

9.46 IP_Assigner_Example Class Reference

Assigns controller an IP Adress given a serial number and a 1 byte address.

Static Public Member Functions

- static int Main (string[] args)

  Main function for the IP Assigner example.

9.46.1 Detailed Description

Assigns controller an IP Adress given a serial number and a 1 byte address.
The first argument should be the Serial # of a Galil Controller.
The second argument should be a 1 Byte value that will be used as the final byte in the newly assigned IP Address.
For VB.NET, see definition in file ip_assigner_example.vb
Definition at line 25 of file ip_assigner_example.cs.

9.46.2 Member Function Documentation

9.46.2.1 Main()

```
static int Main (
            string[] args )  [inline], [static]
```

Main function for the IP Assigner example.

Parameters

args	An array of command line arguments.

Returns

The success status or error code of the function.

The first argument should be the Serial # of a Galil Controller.
The second argument should be a 1 Byte value that will be used as the final byte in the newly assigned IP Address.
Definition at line 35 of file ip_assigner_example.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GClose(), Examples.IP_Assigner(), Examples.PrintError(), and vector().
The documentation for this class was generated from the following file:

- ip_assigner_example.cs

9.47 Jog_Example Class Reference

Accepts user-input at the command line to control the speed of the controller in Jog mode.

Static Public Member Functions

- static int Main (string[] args)

  Main function for the jog example.

9.47.1 Detailed Description

Accepts user-input at the command line to control the speed of the controller in Jog mode.
The first argument should be the IP Address of a Galil controller.
For VB.NET, see definition in file jog_example.vb
Definition at line 23 of file jog_example.cs.

9.47.2 Member Function Documentation

9.47.2.1 Main()

```
static int Main (
            string[] args )  [inline], [static]
```

Main function for the jog example.

Parameters

args	An array of command line arguments.

Returns

The success status or error code of the function.

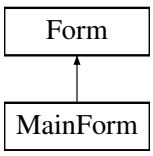
The first argument should be the IP Address of a Galil controller.
Definition at line 31 of file jog_example.cs.

References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GClose(), gclib.GOpen(), Examples.Jog(), Examples.PrintError(), and vector().
The documentation for this class was generated from the following file:

- jog_example.cs

9.48 MainForm Class Reference

Demonstrates using gclib in a Windows Form, including using a second thread to free the GUI.
Inheritance diagram for MainForm:



9.48.1 Detailed Description

Demonstrates using gclib in a Windows Form, including using a second thread to free the GUI.
Demonstrates using gclib in a Windows Form, including using a second thread to free the GUI.

Remarks

Handles the MyBase.Load event.

Definition at line 22 of file Form1.cs.

9.48.2 Constructor & Destructor Documentation

9.48.2.1 MainForm()

```
MainForm ( )  [inline]
```

Definition at line 28 of file Form1.cs.
The documentation for this class was generated from the following files:

- Form1.cs
- Form1.vb

9.49 Message_Example Class Reference

Demonstrates how to handle and interpret messages from the controller.

Static Public Member Functions

- static int Main (string[] args)

  Main function for the message example.

9.49.1 Detailed Description

Demonstrates how to handle and interpret messages from the controller.
The first argument should be the IP Address of a Galil controller.
For VB.NET, see definition in file message_example.vb
Definition at line 23 of file message_example.cs.

9.49.2 Member Function Documentation

9.49.2.1 Main()

```
static int Main (
            string[] args )  [inline], [static]
```

Main function for the message example.

Parameters

args	An array of command line arguments.

Returns

The success status or error code of the function.

The first argument should be the IP Address of a Galil controller.
Definition at line 31 of file message_example.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GClose(), gclib.GOpen(), Examples.Message(), Examples.PrintError(), and vector().
The documentation for this class was generated from the following file:

- message_example.cs

9.50 Motion_Complete_Example Class Reference

Uses controller interrupts to detect when motion is complete.

Static Public Member Functions

- static int Main (string[] args)

  Main function for the Motion Complete example.

9.50.1 Detailed Description

Uses controller interrupts to detect when motion is complete.
The first argument should be the IP Address of a Galil controller.
For VB.NET, see definition in file motion_complete_example.vb
Definition at line 23 of file motion_complete_example.cs.

9.50.2 Member Function Documentation

9.50.2.1 Main()

```
static int Main (
            string[] args )  [inline], [static]
```

Main function for the Motion Complete example.

Parameters

args	An array of command line arguments.

Returns

The success status or error code of the function.

The first argument should be the IP Address of a Galil controller.
Definition at line 31 of file motion_complete_example.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GClose(), gclib.GOpen(), Examples.Motion_Complete(), Examples.PrintError(), and vector().

The documentation for this class was generated from the following file:

- motion_complete_example.cs

9.51 Position_Tracking_Example Class Reference

Places controller into position tracking mode. Accepts user-defined positional values at the command line.

Static Public Member Functions

- static int Main (string[] args)

  Main function for the position tracking example.

9.51.1 Detailed Description

Places controller into position tracking mode. Accepts user-defined positional values at the command line.
The first argument should be the IP Address of a Galil controller.
The second argument is optional and defines the default speed of the controller in Position Tracking mode.
For VB.NET, see definition in file position_tracking_example.vb
Definition at line 26 of file position_tracking_example.cs.

9.51.2 Member Function Documentation

9.51.2.1 Main()

```
static int Main (
            string[] args )  [inline], [static]
```

Main function for the position tracking example.

Parameters

args	An array of command line arguments.

Returns

The success status or error code of the function.

The first argument should be the IP Address of a Galil controller. The second argument is optional and defines the default speed of the controller in Position Tracking mode.
Definition at line 36 of file position_tracking_example.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GClose(), gclib.GOpen(), Examples.Position_Tracking(), Examples.PrintError(), and vector().
The documentation for this class was generated from the following file:

- position_tracking_example.cs

9.52 Program Class Reference

9.52.1 Detailed Description

Definition at line 11 of file Program.cs.
The documentation for this class was generated from the following file:

- Program.cs

9.53 py Class Reference

Public Member Functions

- __init__ (self)

- __del__ (self)
- GOpen (self, address)
- GClose (self)
- GCommand (self, command)
- GSleep (self, val)
- GVersion (self)
- GServerStatus (self)
- GSetServer (self, server_name)
- GListServers (self)
- GPublishServer (self, server_name, publish, save)
- GRemoteConnections (self)
- GInfo (self)
- GIpRequests (self)
- GAssign (self, ip, mac)
- GAddresses (self)
- GProgramDownload (self, program, preprocessor="")
- GProgramUpload (self)
- GProgramDownloadFile (self, file_path, preprocessor="")
- GProgramUploadFile (self, file_path)
- GArrayDownload (self, name, first, last, array_data)
- GArrayUploadFile (self, file_path, names=[])
- GArrayDownloadFile (self, file_path)
- GArrayUpload (self, name, first, last)
- GTimeout (self, timeout)
- timeout (self)
- timeout (self, timeout)
- GFirmwareDownload (self, file_path)
- GMessage (self)
- GMotionComplete (self, axes)
- GInterrupt (self)
- GSetupDownloadFile (self, file_path, options)

Protected Member Functions

- _cc (self)

Protected Attributes

- _gcon
- _buf
- _timeout

9.53.1 Detailed Description

```
Represents a single Python connection to a Galil Controller or PLC.
```

Definition at line 141 of file gclib.py.

9.53.2 Constructor & Destructor Documentation

9.53.2.1 __init__()

```
__init__ (
            self )
```

```
Constructor for the Connection class. Initializes gclib's handle and read buffer.
```

Definition at line 144 of file gclib.py.
References py._buf, py._gcon, py._timeout, and vector().

9.53.2.2 __del__()

```
__del__ (
              self )
```

```
Destructor for the Connection class. Ensures close gets called to release Galil resource (Sockets, Kernel Driv
```

Definition at line 151 of file gclib.py.
References gclib.GClose(), GclibJava.GClose(), Gclib.GClose(), GclibJava.Gclib.GClose(), and py.GClose().

9.53.3 Member Function Documentation

9.53.3.1 _cc()

```
_cc (
              self )  [protected]
```

```
Checks if connection is established, throws error if not.
```

Definition at line 156 of file gclib.py.
References py._gcon.
Referenced by py.GArrayDownload(), py.GArrayDownloadFile(), py.GArrayUpload(), py.GArrayUploadFile(), py.GCommand(), py.GFirmwareDownload(), py.GInterrupt(), py.GMessage(), py.GMotionComplete(), py.GProgramDownload(), py.GProgramDownloadFile(), py.GProgramUpload(), py.GProgramUploadFile(), py.GSetupDownloadFile(), and py.GTimeout().

9.53.3.2 GAddresses()

```
GAddresses (
              self )
```

```
Provides a dictionary of all available connection addresses.

Returns a dictionary mapping 'address' -> 'revision reports', where possible
e.g. {}
```

Definition at line 269 of file gclib.py.
References py._buf, and vector().

9.53.3.3 GArrayDownload()

```
GArrayDownload (
              self,
              name,
              first,
              last,
              array_data )
```

```
Downloads array data to a pre-dimensioned array in the controller's array table.
array_data should be a list of values (e.g. int or float)
```

Definition at line 329 of file gclib.py.
References py._cc(), py._gcon, and vector().

9.53.3.4 GArrayDownloadFile()

```
GArrayDownloadFile (
              self,
              file_path )
```

```
Downloads a csv file containing array data at file_path.
```

Definition at line 361 of file gclib.py.
References py._cc(), py._gcon, and vector().

9.53.3.5 GArrayUpload()

```
GArrayUpload (
              self,
              name,
              first,
              last )

Uploads array data from the controller's array table.
```

Definition at line 371 of file gclib.py.
References py._buf, py._cc(), py._gcon, and vector().

9.53.3.6 GArrayUploadFile()

```
GArrayUploadFile (
              self,
              file_path,
              names = [] )

Uploads the entire controller array table or a subset and saves the data as a csv file specified by file_path.
names is optional and should be a list of array names on the controller.
```

Definition at line 344 of file gclib.py.
References py._cc(), py._gcon, and vector().

9.53.3.7 GAssign()

```
GAssign (
              self,
              ip,
              mac )

Assigns IP address over the Ethernet to a controller at a given MAC address.
Linux/OS X users must be root to use GAssign() and have UDP access to send on port 68.
```

Definition at line 258 of file gclib.py.
References vector().

9.53.3.8 GClose()

```
GClose (
              self )

Closes a connection to a Galil Controller.
```

Definition at line 171 of file gclib.py.
References py._gcon, and vector().
Referenced by py.__del__().

9.53.3.9 GCommand()

```
GCommand (
              self,
              command )

Performs a command-and-response transaction on the connection.
Trims the response.
```

Definition at line 181 of file gclib.py.
References py._buf, py._cc(), py._gcon, and vector().

9.53.3.10 GFirmwareDownload()

```
GFirmwareDownload (
              self,
              file_path )

Upgrade firmware.
```

Definition at line 411 of file gclib.py.
References py._cc(), py._gcon, and vector().

9.53.3.11 GInfo()

```
GInfo (
              self )

Provides a useful connection string. Please include the output of this function on all support cases.
```

Definition at line 230 of file gclib.py.
References py._buf, py._gcon, and vector().

9.53.3.12 GInterrupt()

```
GInterrupt (
              self )

Provides access to PCI and UDP interrupts from the controller.
```

Definition at line 439 of file gclib.py.
References py._cc(), py._gcon, and vector().

9.53.3.13 GIpRequests()

```
GIpRequests (
              self )

Provides a dictionary of all Galil controllers requesting IP addresses via BOOT-P or DHCP.

Returns a dictionary mapping 'model-serial' --> 'mac address'
e.g. {'DMC4000-783': '00:50:4c:20:03:0f', 'DMC4103-9998': '00:50:4c:38:27:0e'}

Linux/OS X users must be root to use GIpRequests() and have UDP access to bind and listen on port 67.
```

Definition at line 238 of file gclib.py.
References py._buf, and vector().

9.53.3.14 GListServers()

```
GListServers (
              self )
```

Definition at line 217 of file gclib.py.

9.53.3.15 GMessage()

```
GMessage (
              self )

Provides access to unsolicited messages from the controller.
```

Definition at line 421 of file gclib.py.
References py._buf, py._cc(), py._gcon, and vector().

9.53.3.16 GMotionComplete()

```
GMotionComplete (
              self,
              axes )
```

```
Blocking call that returns once all axes specified have completed their motion.
```

Definition at line 430 of file gclib.py.
References py._cc(), py._gcon, and vector().

9.53.3.17 GOpen()

```
GOpen (
              self,
              address )
```

```
Opens a connection a galil controller.
See the gclib docs for address string formatting.
```

Definition at line 161 of file gclib.py.
References py._gcon, and vector().

9.53.3.18 GProgramDownload()

```
GProgramDownload (
              self,
              program,
              preprocessor = "" )
```

```
Downloads a program to the controller's program buffer.
See the gclib docs for preprocessor options.
```

Definition at line 288 of file gclib.py.
References py._cc(), py._gcon, and vector().

9.53.3.19 GProgramDownloadFile()

```
GProgramDownloadFile (
              self,
              file_path,
              preprocessor = "" )
```

```
Program download from file.
See the gclib docs for preprocessor options.
```

Definition at line 309 of file gclib.py.
References py._cc(), py._gcon, and vector().

9.53.3.20 GProgramUpload()

```
GProgramUpload (
              self )
```

```
Uploads a program from the controller's program buffer.
```

Definition at line 300 of file gclib.py.
References py._buf, py._cc(), py._gcon, and vector().

9.53.3.21 GProgramUploadFile()

```
GProgramUploadFile (
              self,
              file_path )
```

```
Program upload to file.
```

Definition at line 320 of file gclib.py.
References py._cc(), py._gcon, and vector().

9.53.3.22 GPublishServer()

```
GPublishServer (
              self,
              server_name,
              publish,
              save )
```

Definition at line 221 of file gclib.py.

9.53.3.23 GRemoteConnections()

```
GRemoteConnections (
              self )
```

Definition at line 226 of file gclib.py.

9.53.3.24 GServerStatus()

```
GServerStatus (
              self )
```

Definition at line 208 of file gclib.py.

9.53.3.25 GSetServer()

```
GSetServer (
              self,
              server_name )
```

Definition at line 212 of file gclib.py.

9.53.3.26 GSetupDownloadFile()

```
GSetupDownloadFile (
              self,
              file_path,
              options )
```

```
Downloads specified sectors from a Galil compressed backup (gcb) file to a controller.

Returns a dictionary with the controller information stored in the gcb file.
If options is specified as 0, an additional "options" key will be in the dictionary indicating the info sector
```

Definition at line 448 of file gclib.py.
References py._buf, py._cc(), py._gcon, and vector().

9.53.3.27 GSleep()

```
GSleep (
              self,
              val )
```

```
Provides a blocking sleep call which can be useful for timing-based chores.
```

Definition at line 193 of file gclib.py.
References vector().

9.53.3.28 GTimeout()

```
GTimeout (
              self,
              timeout )
```

```
Set the library timeout. Set to -1 to use the intitial library timeout, as specified in GOpen.
```

Definition at line 385 of file gclib.py.
References py._cc(), py._gcon, py._timeout, and vector().
Referenced by py.timeout().

9.53.3.29 GVersion()

```
GVersion (
            self )
```

Provides the gclib version number. Please include the output of this function on all support cases.

Definition at line 201 of file gclib.py.
References py._buf, and vector().

9.53.3.30 timeout() [1/2]

```
timeout (
            self )
```

Convenience property read access to timeout value. If -1, gclib uses the initial library timeout, as specified

Definition at line 396 of file gclib.py.
References py._timeout.

9.53.3.31 timeout() [2/2]

```
timeout (
            self,
            timeout )
```

Convenience property write access to timeout value. Set to -1 to use the initial library timeout, as specified

Definition at line 403 of file gclib.py.
References gclib.GTimeout(), Gclib.GTimeout(), GclibJava.Gclibo.GTimeout(), py.GTimeout(), and GclibJava.GTimeout().

9.53.4 Field Documentation

9.53.4.1 _buf

```
_buf  [protected]
```

Definition at line 147 of file gclib.py.
Referenced by py.__init__(), py.GAddresses(), py.GArrayUpload(), py.GCommand(), py.GInfo(), py.GIpRequests(), py.GMessage(), py.GProgramUpload(), py.GSetupDownloadFile(), and py.GVersion().

9.53.4.2 _gcon

```
_gcon  [protected]
```

Definition at line 146 of file gclib.py.
Referenced by py.__init__(), py._cc(), py.GArrayDownload(), py.GArrayDownloadFile(), py.GArrayUpload(), py.GArrayUploadFile(), py.GClose(), py.GCommand(), py.GFirmwareDownload(), py.GInfo(), py.GInterrupt(), py.GMessage(), py.GMotionComplete(), py.GOpen(), py.GProgramDownload(), py.GProgramDownloadFile(), py.GProgramUpload(), py.GProgramUploadFile(), py.GSetupDownloadFile(), and py.GTimeout().

9.53.4.3 _timeout

```
_timeout  [protected]
```

Definition at line 148 of file gclib.py.
Referenced by py.__init__(), py.GTimeout(), and py.timeout().
The documentation for this class was generated from the following file:

- gclib.py

9.54 Record_Position_Example Class Reference

Takes two file paths at the command line to hold positional data for Axis A and Axis B. Positional data is saved to the two files until an analog input value changes.

Static Public Member Functions

- static int Main (string[] args)

  Main function for the Record Position example.

9.54.1 Detailed Description

Takes two file paths at the command line to hold positional data for Axis A and Axis B. Positional data is saved to the two files until an analog input value changes.
The first argument should be the IP Address of a Galil controller.
The second argument should be a path to a file to save Axis A positional data.
The third argument should be a path to a file to save Axis B positional data.

For VB.NET, see definition in file record_position_example.vb
Definition at line 27 of file record_position_example.cs.

9.54.2 Member Function Documentation

9.54.2.1 Main()

```
static int Main (
            string[] args )  [inline], [static]
```

Main function for the Record Position example.

Parameters

args	An array of command line arguments.

Returns

The success status or error code of the function.

The first argument should be the IP Address of a Galil controller.
The second argument should be a path to a file to save Axis A positional data.
The third argument should be a path to a file to save Axis B positional data.

Definition at line 38 of file record_position_example.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GClose(), gclib.GOpen(), Examples.PrintError(), Examples.Record_Position(), and vector().
The documentation for this class was generated from the following file:

- record_position_example.cs

9.55 Remote_Client_Example Class Reference

Demonstrates various uses of GListServers() and GSetServer()

Static Public Member Functions

- static int Main ()

  Main function for the Remote Client example.

9.55.1 Detailed Description

Demonstrates various uses of GListServers() and GSetServer()
This example requires no command line arguments.
For VB.NET, see definition in file remote_client_example.vb
Definition at line 24 of file remote_client_example.cs.

9.55.2 Member Function Documentation

9.55.2.1 Main()

```
static int Main ( )  [inline], [static]
```

Main function for the Remote Client example.

Returns

The success status or error code of the function.

The first argument is an optional name to publish your client under.
Definition at line 31 of file remote_client_example.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, Examples.Remote_Client(), and vector().
The documentation for this class was generated from the following file:

- remote_client_example.cs

9.56 Remote_Server_Example Class Reference

Demonstrates various uses of GPublishServer()

Static Public Member Functions

- static int Main (string[] args)

  Main function for the Remote Server example.

9.56.1 Detailed Description

Demonstrates various uses of GPublishServer()
The first argument is an optional name to publish your server under.
For VB.NET, see definition in file remote_server_example.vb
Definition at line 24 of file remote_server_example.cs.

9.56.2 Member Function Documentation

9.56.2.1 Main()

```
static int Main (
            string[] args )  [inline], [static]
```

Main function for the Remote Server example.

Parameters

args	An array of command line arguments.

Returns

The success status or error code of the function.

The first argument is optional and defines the name to publish your server under.
Definition at line 32 of file remote_server_example.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, Examples.Remote_Server(), and vector().
The documentation for this class was generated from the following file:

- remote_server_example.cs

9.57 Source Struct Reference

Public Member Functions

- Source (int byte=0, std::string type="Ux", int bit=-1, std::string units="", std::string description="", double scale=1, double offset=0)

Data Fields

- int byte
- std::string type
- int bit
- std::string units
- std::string description
- double scale
- double offset

9.57.1 Detailed Description

Definition at line 16 of file gcl_galil.h.

9.57.2 Constructor & Destructor Documentation

9.57.2.1 Source()

```
Source (
            int byte = 0,
            std::string type = "Ux",
            int bit = -1,
            std::string units = "",
            std::string description = "",
            double scale = 1,
            double offset = 0 )  [inline]
```

Definition at line 26 of file gcl_galil.h.

9.57.3 Field Documentation

9.57.3.1 bit

```
int bit
```

Definition at line 20 of file gcl_galil.h.

9.57.3.2 byte

```
int byte
```

Definition at line 18 of file gcl_galil.h.

9.57.3.3 description

```
std::string description
```

Definition at line 22 of file gcl_galil.h.

9.57.3.4 offset

```
double offset
```

Definition at line 24 of file gcl_galil.h.

9.57.3.5 scale

```
double scale
```

Definition at line 23 of file gcl_galil.h.

9.57.3.6 type

```
std::string type
```
Definition at line 19 of file gcl_galil.h.

9.57.3.7 units

```
std::string units
```
Definition at line 21 of file gcl_galil.h.
The documentation for this struct was generated from the following file:

- gcl_galil.h

9.58 Vector_Mode_Example Class Reference

Takes a path to a file at the command line holding vector commands for the controller. The controller is placed into vector mode and commands are read from the file and sent to the controller.

Static Public Member Functions

- static int Main (string[] args)

  Main function for the vector mode example.

9.58.1 Detailed Description

Takes a path to a file at the command line holding vector commands for the controller. The controller is placed into vector mode and commands are read from the file and sent to the controller.
The first argument should be the IP Address of a Galil controller. The second argument should be a path to a file containing vector commands.
For VB.NET, see definition in file vector_mode_example.vb
Definition at line 26 of file vector_mode_example.cs.

9.58.2 Member Function Documentation

9.58.2.1 Main()

```
static int Main (
            string[] args )  [inline], [static]
```
Main function for the vector mode example.

Parameters

args	An array of command line arguments.

Returns

The success status or error code of the function.

The first argument should be the IP Address of a Galil controller. The second argument should be a path to a file containing vector commands.
Definition at line 35 of file vector_mode_example.cs.
References Examples.GALIL_EXAMPLE_ERROR, Examples.GALIL_EXAMPLE_OK, gclib.GClose(), gclib.GOpen(), Examples.PrintError(), vector(), and Examples.Vector_Mode().
The documentation for this class was generated from the following file:

- vector_mode_example.cs

Chapter 10

File Documentation

10.1 gclib_record.h File Reference

```
#include <stdint.h>
```

Data Structures

- struct GDataRecord4000

  Data record struct for DMC-4000 controllers, including 4000, 4200, 4103, and 500x0.
- struct GDataRecord52000

  Data record struct for DMC-52000 controller. Same as DMC-4000, with bank indicator added at byte 40.
- struct GDataRecord1806

  Data record struct for DMC-1806 controller.
- struct GDataRecord2103

  Data record struct for DMC-2103 controllers.
- struct GDataRecord1802
- struct GDataRecord30000

  Data record struct for DMC-30010 controllers.
- struct GDataRecord47000_ENC

  Data record struct for RIO-471xx and RIO-472xx PLCs. Includes encoder fields.
- struct GDataRecord47300_ENC

  Data record struct for RIO-47300. Includes encoder fields.
- struct GDataRecord47300_24EX

  Data record struct for RIO-47300 with 24EX I/O daughter board.
- struct GDataRecord47162

  Data record struct for RIO-47162.
- union GDataRecord

  Data record union, containing all structs and a generic byte array accessor.

Macros

- #define GALILDATARECORDMAXLENGTH 512

  Max size for any Galil data record, equal to dual port ram size of PCI.

Typedefs

- typedef uint8_t UB
- typedef uint16_t UW
- typedef int16_t SW
- typedef int32_t SL
- typedef uint32_t UL

10.1.1 Detailed Description

Defines a union for data records. Each supported controller has a struct member in the union with named record types. Offsets into the data record can also be used by referencing the member `byte_array`.
Definition in file gclib_record.h.

10.1.2 Macro Definition Documentation

10.1.2.1 GALILDATARECORDMAXLENGTH

```
#define GALILDATARECORDMAXLENGTH 512
```

Max size for any Galil data record, equal to dual port ram size of PCI.
Definition at line 32 of file gclib_record.h.

10.1.3 Typedef Documentation

10.1.3.1 SL

```
typedef int32_t SL
```

Definition at line 19 of file gclib_record.h.

10.1.3.2 SW

```
typedef int16_t SW
```

Definition at line 18 of file gclib_record.h.

10.1.3.3 UB

```
typedef uint8_t UB
```

Definition at line 16 of file gclib_record.h.

10.1.3.4 UL

```
typedef uint32_t UL
```

Definition at line 20 of file gclib_record.h.

10.1.3.5 UW

```
typedef uint16_t UW
```

Definition at line 17 of file gclib_record.h.

10.2 gclib_record.h

Go to the documentation of this file.

```
00001
00011 #ifndef I_210405D9_D9EF_484F_8258_BB29A1BBA217
00012 #define I_210405D9_D9EF_484F_8258_BB29A1BBA217
00013
00014 //typedefs to keep the layout of the structs clean and matching the Galil user manual docs
00015 #include <stdint.h>
00016 typedef uint8_t UB;  //unsigned byte
00017 typedef uint16_t UW; //unsigned word
00018 typedef int16_t SW;  //signed word
00019 typedef int32_t SL;  //signed long
00020 typedef uint32_t UL; //unsigned long
00021
00022 #if defined(_MSC_VER) || defined(__GNUC__) || defined(__BORLANDC__)
00023 #define PACKOK
00024 #endif
00025
00026 #ifdef PACKOK
00027 #pragma pack(1)
00028 #else
00029 #error "Need to set structure packing for compiler"
00030 #endif
00031
00032 #define GALILDATARECORDMAXLENGTH 512
```

```
00033
00035 struct GDataRecord4000
00036 {
00037
00038   /*Offset   type name        description*/
00039
00040   /*00*/      UB  header_0;
00041   /*01*/      UB  header_1;
00042   /*02*/      UB  header_2;
00043   /*03*/      UB  header_3;
00044
00045   /*04-05*/   UW  sample_number;
00046
00047   /*06*/      UB  input_bank_0;
00048   /*07*/      UB  input_bank_1;
00049   /*08*/      UB  input_bank_2;
00050   /*09*/      UB  input_bank_3;
00051   /*10*/      UB  input_bank_4;
00052   /*11*/      UB  input_bank_5;
00053   /*12*/      UB  input_bank_6;
00054   /*13*/    UB  input_bank_7;
00055   /*14*/    UB  input_bank_8;
00056   /*15*/      UB  input_bank_9;
00057
00058   /*16*/      UB  output_bank_0;
00059   /*17*/      UB  output_bank_1;
00060   /*18*/    UB  output_bank_2;
00061   /*19*/      UB  output_bank_3;
00062   /*20*/    UB  output_bank_4;
00063   /*21*/      UB  output_bank_5;
00064   /*22*/      UB  output_bank_6;
00065   /*23*/      UB  output_bank_7;
00066   /*24*/    UB  output_bank_8;
00067   /*25*/    UB  output_bank_9;
00068
00069   /*26-27*/ SW  reserved_0;
00070   /*28-29*/ SW  reserved_2;
00071   /*30-31*/ SW  reserved_4;
00072   /*32-33*/ SW  reserved_6;
00073   /*34-35*/ SW  reserved_8;
00074   /*36-37*/ SW  reserved_10;
00075   /*38-39*/ SW  reserved_12;
00076   /*40-41*/ SW  reserved_14;
00077
00078   /*42*/      UB  ethernet_status_a;
00079   /*43*/    UB  ethernet_status_b;
00080   /*44*/      UB  ethernet_status_c;
00081   /*45*/    UB  ethernet_status_d;
00082   /*46*/    UB  ethernet_status_e;
00083   /*47*/      UB  ethernet_status_f;
00084   /*48*/    UB  ethernet_status_g;
00085   /*49*/    UB  ethernet_status_h;
00086
00087   /*50*/      UB  error_code;
00088   /*51*/    UB  thread_status;
00089   /*52-55*/ UL  amplifier_status;
00090
00091   /*56-59*/ UL  contour_segment_count;
00092   /*60-61*/ UW  contour_buffer_available;
00093
00094   /*62-63*/ UW  s_plane_segment_count;
00095   /*64-65*/ UW  s_plane_move_status;
00096   /*66-69*/ SL  s_distance;
00097   /*70-71*/ UW  s_plane_buffer_available;
00098
00099   /*72-73*/ UW  t_plane_segment_count;
00100   /*74-75*/ UW  t_plane_move_status;
00101   /*76-79*/ SL  t_distance;
00102   /*80-81*/ UW  t_plane_buffer_available;
00103
00104   /*82-83*/ UW  axis_a_status;
00105   /*84*/      UB  axis_a_switches;
00106   /*85*/      UB  axis_a_stop_code;
00107   /*86-89*/ SL  axis_a_reference_position;
00108   /*90-93*/ SL  axis_a_motor_position;
00109   /*94-97*/ SL  axis_a_position_error;
00110   /*98-101*/  SL  axis_a_aux_position;
00111   /*102-105*/ SL  axis_a_velocity;
00112   /*106-109*/ SL  axis_a_torque;
00113   /*110-111*/ UW  axis_a_analog_in;
00114   /*112*/     UB  axis_a_halls;
00115   /*113*/   UB  axis_a_reserved;
00116   /*114-117*/ SL  axis_a_variable;
00117
00118   /*118-119*/ UW  axis_b_status;
00119   /*120*/     UB  axis_b_switches;
00120   /*121*/     UB  axis_b_stop_code;
```

```
00121   /*122-125*/ SL  axis_b_reference_position;
00122   /*126-129*/ SL  axis_b_motor_position;
00123   /*130-133*/ SL  axis_b_position_error;
00124   /*134-137*/ SL  axis_b_aux_position;
00125   /*138-141*/ SL  axis_b_velocity;
00126   /*142-145*/ SL  axis_b_torque;
00127   /*146-147*/ UW  axis_b_analog_in;
00128   /*148*/     UB  axis_b_halls;
00129   /*149*/     UB  axis_b_reserved;
00130   /*150-153*/ SL  axis_b_variable;
00131
00132   /*154-155*/ UW  axis_c_status;
00133   /*156*/   UB  axis_c_switches;
00134   /*157*/   UB  axis_c_stop_code;
00135   /*158-161*/ SL  axis_c_reference_position;
00136   /*162-165*/ SL  axis_c_motor_position;
00137   /*166-169*/ SL  axis_c_position_error;
00138   /*170-173*/ SL  axis_c_aux_position;
00139   /*174-177*/ SL  axis_c_velocity;
00140   /*178-181*/ SL  axis_c_torque;
00141   /*182-183*/ UW  axis_c_analog_in;
00142   /*184*/     UB  axis_c_halls;
00143   /*185*/     UB  axis_c_reserved;
00144   /*186-189*/ SL  axis_c_variable;
00145
00146   /*190-191*/ UW  axis_d_status;
00147   /*192*/     UB  axis_d_switches;
00148   /*193*/   UB  axis_d_stop_code;
00149   /*194-197*/ SL  axis_d_reference_position;
00150   /*198-201*/ SL  axis_d_motor_position;
00151   /*202-205*/ SL  axis_d_position_error;
00152   /*206-209*/ SL  axis_d_aux_position;
00153   /*210-213*/ SL  axis_d_velocity;
00154   /*214-217*/ SL  axis_d_torque;
00155   /*218-219*/ UW  axis_d_analog_in;
00156   /*220*/   UB  axis_d_halls;
00157   /*221*/   UB  axis_d_reserved;
00158   /*222-225*/ SL  axis_d_variable;
00159
00160   /*226-227*/ UW  axis_e_status;
00161   /*228*/     UB  axis_e_switches;
00162   /*229*/     UB  axis_e_stop_code;
00163   /*230-233*/ SL  axis_e_reference_position;
00164   /*234-237*/ SL  axis_e_motor_position;
00165   /*238-241*/ SL  axis_e_position_error;
00166   /*242-245*/ SL  axis_e_aux_position;
00167   /*246-249*/ SL  axis_e_velocity;
00168   /*250-253*/ SL  axis_e_torque;
00169   /*254-255*/ UW  axis_e_analog_in;
00170   /*256*/   UB  axis_e_halls;
00171   /*257*/     UB  axis_e_reserved;
00172   /*258-261*/ SL  axis_e_variable;
00173
00174   /*262-263*/ UW  axis_f_status;
00175   /*264*/     UB  axis_f_switches;
00176   /*265*/     UB  axis_f_stop_code;
00177   /*266-269*/ SL  axis_f_reference_position;
00178   /*270-273*/ SL  axis_f_motor_position;
00179   /*274-277*/ SL  axis_f_position_error;
00180   /*278-281*/ SL  axis_f_aux_position;
00181   /*282-285*/ SL  axis_f_velocity;
00182   /*286-289*/ SL  axis_f_torque;
00183   /*290-291*/ UW  axis_f_analog_in;
00184   /*292*/     UB  axis_f_halls;
00185   /*293*/     UB  axis_f_reserved;
00186   /*294-297*/ SL  axis_f_variable;
00187
00188   /*298-299*/ UW  axis_g_status;
00189   /*300*/   UB  axis_g_switches;
00190   /*301*/   UB  axis_g_stop_code;
00191   /*302-305*/ SL  axis_g_reference_position;
00192   /*306-309*/ SL  axis_g_motor_position;
00193   /*310-313*/ SL  axis_g_position_error;
00194   /*314-317*/ SL  axis_g_aux_position;
00195   /*318-321*/ SL  axis_g_velocity;
00196   /*322-325*/ SL  axis_g_torque;
00197   /*326-327*/ UW  axis_g_analog_in;
00198   /*328*/     UB  axis_g_halls;
00199   /*329*/     UB  axis_g_reserved;
00200   /*330-333*/ SL  axis_g_variable;
00201
00202   /*334-335*/ UW  axis_h_status;
00203   /*336*/   UB  axis_h_switches;
00204   /*337*/     UB  axis_h_stop_code;
00205   /*338-341*/ SL  axis_h_reference_position;
00206   /*342-345*/ SL  axis_h_motor_position;
00207   /*346-349*/ SL  axis_h_position_error;
```

```
00208   /*350-353*/ SL  axis_h_aux_position;
00209   /*354-357*/ SL  axis_h_velocity;
00210   /*358-361*/ SL  axis_h_torque;
00211   /*362-363*/ UW  axis_h_analog_in;
00212   /*364*/     UB  axis_h_halls;
00213   /*365*/   UB  axis_h_reserved;
00214   /*366-369*/ SL  axis_h_variable;
00215 }; //DataRecord4000
00216
00218 struct GDataRecord52000
00219 {
00220
00221   /*Offset   type name        description*/
00222
00223   /*00*/      UB  header_0;
00224   /*01*/      UB  header_1;
00225   /*02*/      UB  header_2;
00226   /*03*/      UB  header_3;
00227
00228   /*04-05*/   UW  sample_number;
00229
00230   /*06*/      UB  input_bank_0;
00231   /*07*/      UB  input_bank_1;
00232   /*08*/      UB  input_bank_2;
00233   /*09*/      UB  input_bank_3;
00234   /*10*/      UB  input_bank_4;
00235   /*11*/      UB  input_bank_5;
00236   /*12*/      UB  input_bank_6;
00237   /*13*/    UB  input_bank_7;
00238   /*14*/    UB  input_bank_8;
00239   /*15*/      UB  input_bank_9;
00240
00241   /*16*/      UB  output_bank_0;
00242   /*17*/      UB  output_bank_1;
00243   /*18*/    UB  output_bank_2;
00244   /*19*/      UB  output_bank_3;
00245   /*20*/    UB  output_bank_4;
00246   /*21*/      UB  output_bank_5;
00247   /*22*/      UB  output_bank_6;
00248   /*23*/      UB  output_bank_7;
00249   /*24*/    UB  output_bank_8;
00250   /*25*/    UB  output_bank_9;
00251
00252   /*26-27*/ SW  reserved_0;
00253   /*28-29*/ SW  reserved_2;
00254   /*30-31*/ SW  reserved_4;
00255   /*32-33*/ SW  reserved_6;
00256   /*34-35*/ SW  reserved_8;
00257   /*36-37*/ SW  reserved_10;
00258   /*38-39*/ SW  reserved_12;
00259   /*40*/      UB  ethercat_bank;
00260   /*41*/    UB  reserved_14;
00261
00262   /*42*/      UB  ethernet_status_a;
00263   /*43*/    UB  ethernet_status_b;
00264   /*44*/      UB  ethernet_status_c;
00265   /*45*/    UB  ethernet_status_d;
00266   /*46*/    UB  ethernet_status_e;
00267   /*47*/      UB  ethernet_status_f;
00268   /*48*/    UB  ethernet_status_g;
00269   /*49*/    UB  ethernet_status_h;
00270
00271   /*50*/      UB  error_code;
00272   /*51*/    UB  thread_status;
00273   /*52-55*/ UL  amplifier_status;
00274
00275   /*56-59*/ UL  contour_segment_count;
00276   /*60-61*/ UW  contour_buffer_available;
00277
00278   /*62-63*/ UW  s_plane_segment_count;
00279   /*64-65*/ UW  s_plane_move_status;
00280   /*66-69*/ SL  s_distance;
00281   /*70-71*/ UW  s_plane_buffer_available;
00282
00283   /*72-73*/ UW  t_plane_segment_count;
00284   /*74-75*/ UW  t_plane_move_status;
00285   /*76-79*/ SL  t_distance;
00286   /*80-81*/ UW  t_plane_buffer_available;
00287
00288   /*82-83*/ UW  axis_a_status;
00289   /*84*/      UB  axis_a_switches;
00290   /*85*/      UB  axis_a_stop_code;
00291   /*86-89*/ SL  axis_a_reference_position;
00292   /*90-93*/ SL  axis_a_motor_position;
00293   /*94-97*/ SL  axis_a_position_error;
00294   /*98-101*/  SL  axis_a_aux_position;
00295   /*102-105*/ SL  axis_a_velocity;
```

```
00296   /*106-109*/ SL  axis_a_torque;
00297   /*110-111*/ UW  axis_a_analog_in;
00298   /*112*/     UB  axis_a_halls;
00299   /*113*/   UB  axis_a_reserved;
00300   /*114-117*/ SL  axis_a_variable;
00301
00302   /*118-119*/ UW  axis_b_status;
00303   /*120*/     UB  axis_b_switches;
00304   /*121*/     UB  axis_b_stop_code;
00305   /*122-125*/ SL  axis_b_reference_position;
00306   /*126-129*/ SL  axis_b_motor_position;
00307   /*130-133*/ SL  axis_b_position_error;
00308   /*134-137*/ SL  axis_b_aux_position;
00309   /*138-141*/ SL  axis_b_velocity;
00310   /*142-145*/ SL  axis_b_torque;
00311   /*146-147*/ UW  axis_b_analog_in;
00312   /*148*/     UB  axis_b_halls;
00313   /*149*/     UB  axis_b_reserved;
00314   /*150-153*/ SL  axis_b_variable;
00315
00316   /*154-155*/ UW  axis_c_status;
00317   /*156*/   UB  axis_c_switches;
00318   /*157*/   UB  axis_c_stop_code;
00319   /*158-161*/ SL  axis_c_reference_position;
00320   /*162-165*/ SL  axis_c_motor_position;
00321   /*166-169*/ SL  axis_c_position_error;
00322   /*170-173*/ SL  axis_c_aux_position;
00323   /*174-177*/ SL  axis_c_velocity;
00324   /*178-181*/ SL  axis_c_torque;
00325   /*182-183*/ UW  axis_c_analog_in;
00326   /*184*/     UB  axis_c_halls;
00327   /*185*/     UB  axis_c_reserved;
00328   /*186-189*/ SL  axis_c_variable;
00329
00330   /*190-191*/ UW  axis_d_status;
00331   /*192*/     UB  axis_d_switches;
00332   /*193*/   UB  axis_d_stop_code;
00333   /*194-197*/ SL  axis_d_reference_position;
00334   /*198-201*/ SL  axis_d_motor_position;
00335   /*202-205*/ SL  axis_d_position_error;
00336   /*206-209*/ SL  axis_d_aux_position;
00337   /*210-213*/ SL  axis_d_velocity;
00338   /*214-217*/ SL  axis_d_torque;
00339   /*218-219*/ UW  axis_d_analog_in;
00340   /*220*/   UB  axis_d_halls;
00341   /*221*/   UB  axis_d_reserved;
00342   /*222-225*/ SL  axis_d_variable;
00343
00344   /*226-227*/ UW  axis_e_status;
00345   /*228*/     UB  axis_e_switches;
00346   /*229*/     UB  axis_e_stop_code;
00347   /*230-233*/ SL  axis_e_reference_position;
00348   /*234-237*/ SL  axis_e_motor_position;
00349   /*238-241*/ SL  axis_e_position_error;
00350   /*242-245*/ SL  axis_e_aux_position;
00351   /*246-249*/ SL  axis_e_velocity;
00352   /*250-253*/ SL  axis_e_torque;
00353   /*254-255*/ UW  axis_e_analog_in;
00354   /*256*/   UB  axis_e_halls;
00355   /*257*/     UB  axis_e_reserved;
00356   /*258-261*/ SL  axis_e_variable;
00357
00358   /*262-263*/ UW  axis_f_status;
00359   /*264*/     UB  axis_f_switches;
00360   /*265*/     UB  axis_f_stop_code;
00361   /*266-269*/ SL  axis_f_reference_position;
00362   /*270-273*/ SL  axis_f_motor_position;
00363   /*274-277*/ SL  axis_f_position_error;
00364   /*278-281*/ SL  axis_f_aux_position;
00365   /*282-285*/ SL  axis_f_velocity;
00366   /*286-289*/ SL  axis_f_torque;
00367   /*290-291*/ UW  axis_f_analog_in;
00368   /*292*/     UB  axis_f_halls;
00369   /*293*/     UB  axis_f_reserved;
00370   /*294-297*/ SL  axis_f_variable;
00371
00372   /*298-299*/ UW  axis_g_status;
00373   /*300*/   UB  axis_g_switches;
00374   /*301*/   UB  axis_g_stop_code;
00375   /*302-305*/ SL  axis_g_reference_position;
00376   /*306-309*/ SL  axis_g_motor_position;
00377   /*310-313*/ SL  axis_g_position_error;
00378   /*314-317*/ SL  axis_g_aux_position;
00379   /*318-321*/ SL  axis_g_velocity;
00380   /*322-325*/ SL  axis_g_torque;
00381   /*326-327*/ UW  axis_g_analog_in;
00382   /*328*/     UB  axis_g_halls;
```

```
00383   /*329*/     UB  axis_g_reserved;
00384   /*330-333*/ SL  axis_g_variable;
00385
00386   /*334-335*/ UW  axis_h_status;
00387   /*336*/   UB  axis_h_switches;
00388   /*337*/     UB  axis_h_stop_code;
00389   /*338-341*/ SL  axis_h_reference_position;
00390   /*342-345*/ SL  axis_h_motor_position;
00391   /*346-349*/ SL  axis_h_position_error;
00392   /*350-353*/ SL  axis_h_aux_position;
00393   /*354-357*/ SL  axis_h_velocity;
00394   /*358-361*/ SL  axis_h_torque;
00395   /*362-363*/ UW  axis_h_analog_in;
00396   /*364*/     UB  axis_h_halls;
00397   /*365*/   UB  axis_h_reserved;
00398   /*366-369*/ SL  axis_h_variable;
00399 }; //DataRecord52000
00400
00402
00409 struct GDataRecord1806
00410 {
00411   /*Offset   type name        description*/
00412
00413   /*00-01*/   UW  sample_number;
00414
00415   /*02*/      UB  input_bank_0;
00416   /*03*/      UB  input_bank_1;
00417   /*04*/      UB  input_bank_2;
00418   /*05*/      UB  input_bank_3;
00419   /*06*/      UB  input_bank_4;
00420   /*07*/      UB  input_bank_5;
00421   /*08*/      UB  input_bank_6;
00422   /*09*/    UB  input_bank_7;
00423   /*10*/    UB  input_bank_8;
00424   /*11*/      UB  input_bank_9;
00425
00426   /*12*/      UB  output_bank_0;
00427   /*13*/      UB  output_bank_1;
00428   /*14*/    UB  output_bank_2;
00429   /*15*/      UB  output_bank_3;
00430   /*16*/    UB  output_bank_4;
00431   /*17*/      UB  output_bank_5;
00432   /*18*/      UB  output_bank_6;
00433   /*19*/      UB  output_bank_7;
00434   /*20*/    UB  output_bank_8;
00435   /*21*/    UB  output_bank_9;
00436
00437   /*22-23*/ SW  reserved_0;
00438   /*24-25*/ SW  reserved_2;
00439   /*26-27*/ SW  reserved_4;
00440   /*28-29*/ SW  reserved_6;
00441   /*30-31*/ SW  reserved_8;
00442   /*32-33*/ SW  reserved_10;
00443   /*34-35*/ SW  reserved_12;
00444   /*36-37*/ SW  reserved_14;
00445
00446   /*38*/      UB  reserved_16;
00447   /*39*/    UB  reserved_17;
00448   /*40*/      UB  reserved_18;
00449   /*41*/    UB  reserved_19;
00450   /*42*/    UB  reserved_20;
00451   /*43*/      UB  reserved_21;
00452   /*44*/    UB  reserved_22;
00453   /*45*/    UB  reserved_23;
00454
00455   /*46*/      UB  error_code;
00456   /*47*/    UB  thread_status;
00457   /*48-51*/ UL  reserved_24;
00458
00459   /*52-55*/ UL  contour_segment_count;
00460   /*56-57*/ UW  contour_buffer_available;
00461
00462   /*58-59*/ UW  s_plane_segment_count;
00463   /*60-61*/ UW  s_plane_move_status;
00464   /*62-65*/ SL  s_distance;
00465   /*66-67*/ UW  s_plane_buffer_available;
00466
00467   /*68-69*/ UW  t_plane_segment_count;
00468   /*70-71*/ UW  t_plane_move_status;
00469   /*72-75*/ SL  t_distance;
00470   /*76-77*/ UW  t_plane_buffer_available;
00471
00472   /*78-79*/ UW  axis_a_status;
00473   /*80*/      UB  axis_a_switches;
00474   /*81*/      UB  axis_a_stop_code;
00475   /*82-85*/ SL  axis_a_reference_position;
00476   /*86-89*/ SL  axis_a_motor_position;
```

```
00477   /*90-93*/ SL  axis_a_position_error;
00478   /*94-97*/ SL  axis_a_aux_position;
00479   /*98-101*/  SL  axis_a_velocity;
00480   /*102-105*/ SL  axis_a_torque;
00481   /*106-107*/ UW  axis_a_analog_in;
00482   /*108*/     UB  axis_a_reserved_0;
00483   /*109*/   UB  axis_a_reserved_1;
00484   /*110-113*/ SL  axis_a_variable;
00485
00486   /*114-115*/ UW  axis_b_status;
00487   /*116*/     UB  axis_b_switches;
00488   /*117*/     UB  axis_b_stop_code;
00489   /*118-121*/ SL  axis_b_reference_position;
00490   /*122-125*/ SL  axis_b_motor_position;
00491   /*126-129*/ SL  axis_b_position_error;
00492   /*130-133*/ SL  axis_b_aux_position;
00493   /*134-137*/ SL  axis_b_velocity;
00494   /*138-141*/ SL  axis_b_torque;
00495   /*142-143*/ UW  axis_b_analog_in;
00496   /*144*/     UB  axis_b_reserved_0;
00497   /*145*/     UB  axis_b_reserved_1;
00498   /*146-149*/ SL  axis_b_variable;
00499
00500   /*150-151*/ UW  axis_c_status;
00501   /*152*/   UB  axis_c_switches;
00502   /*153*/   UB  axis_c_stop_code;
00503   /*154-157*/ SL  axis_c_reference_position;
00504   /*158-161*/ SL  axis_c_motor_position;
00505   /*162-165*/ SL  axis_c_position_error;
00506   /*166-169*/ SL  axis_c_aux_position;
00507   /*170-173*/ SL  axis_c_velocity;
00508   /*174-177*/ SL  axis_c_torque;
00509   /*178-179*/ UW  axis_c_analog_in;
00510   /*180*/     UB  axis_c_reserved_0;
00511   /*181*/     UB  axis_c_reserved_1;
00512   /*182-185*/ SL  axis_c_variable;
00513
00514   /*186-187*/ UW  axis_d_status;
00515   /*188*/     UB  axis_d_switches;
00516   /*189*/   UB  axis_d_stop_code;
00517   /*190-193*/ SL  axis_d_reference_position;
00518   /*194-197*/ SL  axis_d_motor_position;
00519   /*198-201*/ SL  axis_d_position_error;
00520   /*202-205*/ SL  axis_d_aux_position;
00521   /*206-209*/ SL  axis_d_velocity;
00522   /*210-213*/ SL  axis_d_torque;
00523   /*214-215*/ UW  axis_d_analog_in;
00524   /*216*/   UB  axis_d_reserved_0;
00525   /*217*/   UB  axis_d_reserved_1;
00526   /*218-221*/ SL  axis_d_variable;
00527
00528   /*222-223*/ UW  axis_e_status;
00529   /*224*/     UB  axis_e_switches;
00530   /*225*/     UB  axis_e_stop_code;
00531   /*226-229*/ SL  axis_e_reference_position;
00532   /*230-233*/ SL  axis_e_motor_position;
00533   /*234-237*/ SL  axis_e_position_error;
00534   /*238-241*/ SL  axis_e_aux_position;
00535   /*242-245*/ SL  axis_e_velocity;
00536   /*256-249*/ SL  axis_e_torque;
00537   /*250-251*/ UW  axis_e_analog_in;
00538   /*252*/   UB  axis_e_reserved_0;
00539   /*253*/     UB  axis_e_reserved_1;
00540   /*254-257*/ SL  axis_e_variable;
00541
00542   /*258-259*/ UW  axis_f_status;
00543   /*260*/     UB  axis_f_switches;
00544   /*261*/     UB  axis_f_stop_code;
00545   /*262-265*/ SL  axis_f_reference_position;
00546   /*266-269*/ SL  axis_f_motor_position;
00547   /*270-273*/ SL  axis_f_position_error;
00548   /*274-277*/ SL  axis_f_aux_position;
00549   /*278-281*/ SL  axis_f_velocity;
00550   /*282-285*/ SL  axis_f_torque;
00551   /*286-287*/ UW  axis_f_analog_in;
00552   /*288*/     UB  axis_f_reserved_0;
00553   /*289*/     UB  axis_f_reserved_1;
00554   /*290-293*/ SL  axis_f_variable;
00555
00556   /*294-295*/ UW  axis_g_status;
00557   /*296*/   UB  axis_g_switches;
00558   /*297*/   UB  axis_g_stop_code;
00559   /*298-301*/ SL  axis_g_reference_position;
00560   /*302-305*/ SL  axis_g_motor_position;
00561   /*306-309*/ SL  axis_g_position_error;
00562   /*310-313*/ SL  axis_g_aux_position;
00563   /*314-317*/ SL  axis_g_velocity;
```

```
00564   /*318-321*/ SL  axis_g_torque;
00565   /*322-323*/ UW  axis_g_analog_in;
00566   /*324*/     UB  axis_g_reserved_0;
00567   /*325*/     UB  axis_g_reserved_1;
00568   /*326-329*/ SL  axis_g_variable;
00569
00570   /*330-331*/ UW  axis_h_status;
00571   /*332*/   UB  axis_h_switches;
00572   /*333*/     UB  axis_h_stop_code;
00573   /*334-337*/ SL  axis_h_reference_position;
00574   /*338-341*/ SL  axis_h_motor_position;
00575   /*342-345*/ SL  axis_h_position_error;
00576   /*346-349*/ SL  axis_h_aux_position;
00577   /*350-353*/ SL  axis_h_velocity;
00578   /*354-357*/ SL  axis_h_torque;
00579   /*358-359*/ UW  axis_h_analog_in;
00580   /*360*/     UB  axis_h_reserved_0;
00581   /*361*/   UB  axis_h_reserved_1;
00582   /*362-365*/ SL  axis_h_variable;
00583 }; //DataRecord1806
00584
00586 struct GDataRecord2103
00587 {
00588
00589   /*Offset   type name        description*/
00590
00591   /*00*/      UB  header_0;
00592   /*01*/      UB  header_1;
00593   /*02*/      UB  header_2;
00594   /*03*/      UB  header_3;
00595
00596   /*04-05*/   UW  sample_number;
00597
00598   /*06*/      UB  input_bank_0;
00599   /*07*/      UB  input_bank_1;
00600   /*08*/      UB  input_bank_2;
00601   /*09*/      UB  input_bank_3;
00602   /*10*/      UB  input_bank_4;
00603   /*11*/      UB  input_bank_5;
00604   /*12*/      UB  input_bank_6;
00605   /*13*/    UB  input_bank_7;
00606   /*14*/    UB  input_bank_8;
00607   /*15*/      UB  input_bank_9;
00608
00609   /*16*/      UB  output_bank_0;
00610   /*17*/      UB  output_bank_1;
00611   /*18*/    UB  output_bank_2;
00612   /*19*/      UB  output_bank_3;
00613   /*20*/    UB  output_bank_4;
00614   /*21*/      UB  output_bank_5;
00615   /*22*/      UB  output_bank_6;
00616   /*23*/      UB  output_bank_7;
00617   /*24*/    UB  output_bank_8;
00618   /*25*/    UB  output_bank_9;
00619
00620   /*26*/      UB  error_code;
00621   /*27*/    UB  general_status;
00622
00623   /*28-29*/ UW  s_plane_segment_count;
00624   /*30-31*/ UW  s_plane_move_status;
00625   /*32-35*/ SL  s_distance;
00626
00627   /*36-37*/ UW  t_plane_segment_count;
00628   /*38-39*/ UW  t_plane_move_status;
00629   /*40-43*/ SL  t_distance;
00630
00631   /*44-45*/ UW  axis_a_status;
00632   /*46*/      UB  axis_a_switches;
00633   /*47*/      UB  axis_a_stop_code;
00634   /*48-51*/ SL  axis_a_reference_position;
00635   /*52-55*/ SL  axis_a_motor_position;
00636   /*56-59*/ SL  axis_a_position_error;
00637   /*60-63*/ SL  axis_a_aux_position;
00638   /*64-67*/ SL  axis_a_velocity;
00639   /*68-69*/ SW  axis_a_torque;
00640   /*70-71*/ UW  axis_a_analog_in;
00641
00642   /*72-73*/ UW  axis_b_status;
00643   /*74*/      UB  axis_b_switches;
00644   /*75*/      UB  axis_b_stop_code;
00645   /*76-79*/ SL  axis_b_reference_position;
00646   /*80-83*/ SL  axis_b_motor_position;
00647   /*84-87*/ SL  axis_b_position_error;
00648   /*88-91*/ SL  axis_b_aux_position;
00649   /*92-95*/ SL  axis_b_velocity;
00650   /*96-97*/ SW  axis_b_torque;
00651   /*98-99*/ UW  axis_b_analog_in;
```

```
00652
00653   /*100-101*/ UW  axis_c_status;
00654   /*102*/   UB  axis_c_switches;
00655   /*103*/   UB  axis_c_stop_code;
00656   /*104-107*/ SL  axis_c_reference_position;
00657   /*108-111*/ SL  axis_c_motor_position;
00658   /*112-115*/ SL  axis_c_position_error;
00659   /*116-119*/ SL  axis_c_aux_position;
00660   /*120-123*/ SL  axis_c_velocity;
00661   /*124-125*/ SW  axis_c_torque;
00662   /*126-127*/ UW  axis_c_analog_in;
00663
00664   /*128-129*/ UW  axis_d_status;
00665   /*130*/     UB  axis_d_switches;
00666   /*131*/   UB  axis_d_stop_code;
00667   /*132-135*/ SL  axis_d_reference_position;
00668   /*136-139*/ SL  axis_d_motor_position;
00669   /*140-143*/ SL  axis_d_position_error;
00670   /*144-147*/ SL  axis_d_aux_position;
00671   /*148-151*/ SL  axis_d_velocity;
00672   /*152-153*/ SW  axis_d_torque;
00673   /*154-155*/ UW  axis_d_analog_in;
00674
00675   /*156-157*/ UW  axis_e_status;
00676   /*158*/     UB  axis_e_switches;
00677   /*159*/     UB  axis_e_stop_code;
00678   /*160-163*/ SL  axis_e_reference_position;
00679   /*164-167*/ SL  axis_e_motor_position;
00680   /*168-171*/ SL  axis_e_position_error;
00681   /*172-175*/ SL  axis_e_aux_position;
00682   /*176-179*/ SL  axis_e_velocity;
00683   /*180-181*/ SW  axis_e_torque;
00684   /*182-183*/ UW  axis_e_analog_in;
00685
00686   /*184-185*/ UW  axis_f_status;
00687   /*186*/     UB  axis_f_switches;
00688   /*187*/     UB  axis_f_stop_code;
00689   /*188-191*/ SL  axis_f_reference_position;
00690   /*192-195*/ SL  axis_f_motor_position;
00691   /*196-199*/ SL  axis_f_position_error;
00692   /*200-203*/ SL  axis_f_aux_position;
00693   /*204-207*/ SL  axis_f_velocity;
00694   /*208-209*/ SW  axis_f_torque;
00695   /*210-211*/ UW  axis_f_analog_in;
00696
00697   /*212-213*/ UW  axis_g_status;
00698   /*214*/   UB  axis_g_switches;
00699   /*215*/   UB  axis_g_stop_code;
00700   /*216-219*/ SL  axis_g_reference_position;
00701   /*220-223*/ SL  axis_g_motor_position;
00702   /*224-227*/ SL  axis_g_position_error;
00703   /*228-231*/ SL  axis_g_aux_position;
00704   /*232-235*/ SL  axis_g_velocity;
00705   /*236-237*/ SW  axis_g_torque;
00706   /*238-239*/ UW  axis_g_analog_in;
00707
00708   /*240-241*/ UW  axis_h_status;
00709   /*242*/   UB  axis_h_switches;
00710   /*243*/     UB  axis_h_stop_code;
00711   /*244-247*/ SL  axis_h_reference_position;
00712   /*248-251*/ SL  axis_h_motor_position;
00713   /*252-255*/ SL  axis_h_position_error;
00714   /*256-259*/ SL  axis_h_aux_position;
00715   /*260-263*/ SL  axis_h_velocity;
00716   /*264-265*/ SW  axis_h_torque;
00717   /*266-267*/ UW  axis_h_analog_in;
00718 }; //DataRecord2013
00719
00722
00727 struct GDataRecord1802
00728 {
00729
00730   /*Offset   type name        description*/
00731
00732   /*00-01*/   UW  sample_number;
00733
00734   /*02*/      UB  input_bank_0;
00735   /*03*/      UB  input_bank_1;
00736   /*04*/      UB  input_bank_2;
00737   /*05*/      UB  input_bank_3;
00738   /*06*/      UB  input_bank_4;
00739   /*07*/      UB  input_bank_5;
00740   /*08*/      UB  input_bank_6;
00741   /*09*/    UB  input_bank_7;
00742   /*10*/    UB  input_bank_8;
00743   /*11*/      UB  input_bank_9;
00744
```

```
00745   /*12*/      UB  output_bank_0;
00746   /*13*/      UB  output_bank_1;
00747   /*14*/    UB  output_bank_2;
00748   /*15*/      UB  output_bank_3;
00749   /*16*/    UB  output_bank_4;
00750   /*17*/      UB  output_bank_5;
00751   /*18*/      UB  output_bank_6;
00752   /*19*/      UB  output_bank_7;
00753   /*20*/    UB  output_bank_8;
00754   /*21*/    UB  output_bank_9;
00755
00756   /*22*/      UB  error_code;
00757   /*23*/    UB  general_status;
00758
00759   /*24-25*/ UW  s_plane_segment_count;
00760   /*26-27*/ UW  s_plane_move_status;
00761   /*28-31*/ SL  s_distance;
00762
00763   /*32-33*/ UW  t_plane_segment_count;
00764   /*34-35*/ UW  t_plane_move_status;
00765   /*36-39*/ SL  t_distance;
00766
00767   /*40-41*/ UW  axis_a_status;
00768   /*42*/      UB  axis_a_switches;
00769   /*43*/      UB  axis_a_stop_code;
00770   /*44-47*/ SL  axis_a_reference_position;
00771   /*48-51*/ SL  axis_a_motor_position;
00772   /*52-55*/ SL  axis_a_position_error;
00773   /*56-59*/ SL  axis_a_aux_position;
00774   /*60-63*/ SL  axis_a_velocity;
00775   /*64-65*/ SW  axis_a_torque;
00776   /*66*/      UB  axis_a_reserved_0;
00777   /*67*/      UB  axis_a_reserved_1;
00778
00779   /*68-69*/   UW  axis_b_status;
00780   /*70*/      UB  axis_b_switches;
00781   /*71*/      UB  axis_b_stop_code;
00782   /*72-75*/ SL  axis_b_reference_position;
00783   /*76-79*/ SL  axis_b_motor_position;
00784   /*80-83*/ SL  axis_b_position_error;
00785   /*84-87*/ SL  axis_b_aux_position;
00786   /*88-91*/ SL  axis_b_velocity;
00787   /*92-93*/ SW  axis_b_torque;
00788   /*94*/      UB  axis_b_reserved_0;
00789   /*95*/      UB  axis_b_reserved_1;
00790
00791   /*96-97*/ UW  axis_c_status;
00792   /*98*/    UB  axis_c_switches;
00793   /*99*/    UB  axis_c_stop_code;
00794   /*100-103*/ SL  axis_c_reference_position;
00795   /*104-107*/ SL  axis_c_motor_position;
00796   /*108-111*/ SL  axis_c_position_error;
00797   /*112-115*/ SL  axis_c_aux_position;
00798   /*116-119*/ SL  axis_c_velocity;
00799   /*120-121*/ SW  axis_c_torque;
00800   /*122*/     UB  axis_c_reserved_0;
00801   /*123*/     UB  axis_c_reserved_1;
00802
00803   /*124-125*/ UW  axis_d_status;
00804   /*126*/     UB  axis_d_switches;
00805   /*127*/   UB  axis_d_stop_code;
00806   /*128-131*/ SL  axis_d_reference_position;
00807   /*132-135*/ SL  axis_d_motor_position;
00808   /*136-139*/ SL  axis_d_position_error;
00809   /*140-143*/ SL  axis_d_aux_position;
00810   /*144-147*/ SL  axis_d_velocity;
00811   /*148-149*/ SW  axis_d_torque;
00812   /*150*/     UB  axis_d_reserved_0;
00813   /*151*/     UB  axis_d_reserved_1;
00814
00815 }; //DataRecord1802
00816
00818 struct GDataRecord30000
00819 {
00820
00821   /*Offset   type name        description*/
00822
00823   /*00*/      UB  header_0;
00824   /*01*/      UB  header_1;
00825   /*02*/      UB  header_2;
00826   /*03*/      UB  header_3;
00827
00828   /*04-05*/   UW  sample_number;
00829
00830   /*06*/      UB  input_bank_0;
00831   /*07*/      UB  input_bank_1;
00832
```

```
00833   /*08*/      UB  output_bank_0;
00834   /*09*/      UB  output_bank_1;
00835
00836   /*10*/      UB  error_code;
00837   /*11*/    UB  thread_status;
00838
00839   /*12-13*/ UW  input_analog_2;
00840
00841   /*14-15*/ UW  output_analog_1;
00842   /*16-17*/ UW  output_analog_2;
00843
00844   /*18-21*/ UL  amplifier_status;
00845
00846   /*22-25*/ UL  contour_segment_count;
00847   /*26-27*/ UW  contour_buffer_available;
00848
00849   /*28-29*/ UW  s_plane_segment_count;
00850   /*30-31*/ UW  s_plane_move_status;
00851   /*32-35*/ SL  s_distance;
00852   /*36-37*/ UW  s_plane_buffer_available;
00853
00854   /*38-39*/ UW  axis_a_status;
00855   /*40*/      UB  axis_a_switches;
00856   /*41*/      UB  axis_a_stop_code;
00857   /*42-45*/ SL  axis_a_reference_position;
00858   /*46-49*/ SL  axis_a_motor_position;
00859   /*50-53*/ SL  axis_a_position_error;
00860   /*54-57*/ SL  axis_a_aux_position;
00861   /*58-61*/ SL  axis_a_velocity;
00862   /*62-65*/ SL  axis_a_torque;
00863   /*66-67*/ UW  axis_a_analog_in;
00864   /*68*/      UB  axis_a_halls;
00865   /*69*/    UB  axis_a_reserved;
00866   /*70-73*/ SL  axis_a_variable;
00867 }; //DataRecord30000
00868
00870 struct GDataRecord47000_ENC
00871 {
00872
00873   /*Offset   type name        description*/
00874
00875   /*00*/      UB  header_0;
00876   /*01*/      UB  header_1;
00877   /*02*/      UB  header_2;
00878   /*03*/      UB  header_3;
00879
00880   /*04-05*/   UW  sample_number;
00881   /*06*/      UB  error_code;
00882   /*07*/    UB  general_status;
00883
00884   /*08-09*/ UW  output_analog_0;
00885   /*10-11*/ UW  output_analog_1;
00886   /*12-13*/ UW  output_analog_2;
00887   /*14-15*/ UW  output_analog_3;
00888   /*16-17*/ UW  output_analog_4;
00889   /*18-19*/ UW  output_analog_5;
00890   /*20-21*/ UW  output_analog_6;
00891   /*22-23*/ UW  output_analog_7;
00892
00893   /*24-25*/ UW  input_analog_0;
00894   /*26-27*/ UW  input_analog_1;
00895   /*28-29*/ UW  input_analog_2;
00896   /*30-31*/ UW  input_analog_3;
00897   /*32-33*/ UW  input_analog_4;
00898   /*34-35*/ UW  input_analog_5;
00899   /*36-37*/ UW  input_analog_6;
00900   /*38-39*/ UW  input_analog_7;
00901
00902   /*40-41*/ UW  output_bank_0;
00903
00904   /*42-43*/ UW  input_bank_0;
00905
00906   /*44-47*/ UL  pulse_count_0;
00907   /*48-51*/ SL  zc_variable;
00908   /*52-55*/ SL  zd_variable;
00909
00910   /*56-59*/ SL  encoder_0;
00911   /*60-63*/ SL  encoder_1;
00912   /*64-67*/ SL  encoder_2;
00913   /*68-71*/ SL  encoder_3;
00914
00915 }; //GDataRecord47000_ENC
00916
00918 struct GDataRecord47300_ENC
00919 {
00920
00921   /*Offset   type name        description*/
```

```
00922
00923   /*00*/      UB  header_0;
00924   /*01*/      UB  header_1;
00925   /*02*/      UB  header_2;
00926   /*03*/      UB  header_3;
00927
00928   /*04-05*/   UW  sample_number;
00929   /*06*/      UB  error_code;
00930   /*07*/    UB  general_status;
00931
00932   /*08-09*/ UW  output_analog_0;
00933   /*10-11*/ UW  output_analog_1;
00934   /*12-13*/ UW  output_analog_2;
00935   /*14-15*/ UW  output_analog_3;
00936   /*16-17*/ UW  output_analog_4;
00937   /*18-19*/ UW  output_analog_5;
00938   /*20-21*/ UW  output_analog_6;
00939   /*22-23*/ UW  output_analog_7;
00940
00941   /*24-25*/ UW  input_analog_0;
00942   /*26-27*/ UW  input_analog_1;
00943   /*28-29*/ UW  input_analog_2;
00944   /*30-31*/ UW  input_analog_3;
00945   /*32-33*/ UW  input_analog_4;
00946   /*34-35*/ UW  input_analog_5;
00947   /*36-37*/ UW  input_analog_6;
00948   /*38-39*/ UW  input_analog_7;
00949
00950   /*40-41*/ UW  output_bank_0;
00951   /*42-43*/ UW  output_bank_1;
00952
00953   /*44-45*/ UW  input_bank_0;
00954   /*46-47*/ UW  input_bank_1;
00955
00956   /*48-51*/ UL  pulse_count_0;
00957   /*52-55*/ SL  zc_variable;
00958   /*56-59*/ SL  zd_variable;
00959
00960   /*60-63*/ SL  encoder_0;
00961   /*64-67*/ SL  encoder_1;
00962   /*68-71*/ SL  encoder_2;
00963   /*72-75*/ SL  encoder_3;
00964
00965 }; //GDataRecord47300_ENC
00966
00968 struct GDataRecord47300_24EX
00969 {
00970
00971   /*Offset   type name        description*/
00972
00973   /*00*/      UB  header_0;
00974   /*01*/      UB  header_1;
00975   /*02*/      UB  header_2;
00976   /*03*/      UB  header_3;
00977
00978   /*04-05*/   UW  sample_number;
00979   /*06*/      UB  error_code;
00980   /*07*/    UB  general_status;
00981
00982   /*08-09*/ UW  output_analog_0;
00983   /*10-11*/ UW  output_analog_1;
00984   /*12-13*/ UW  output_analog_2;
00985   /*14-15*/ UW  output_analog_3;
00986   /*16-17*/ UW  output_analog_4;
00987   /*18-19*/ UW  output_analog_5;
00988   /*20-21*/ UW  output_analog_6;
00989   /*22-23*/ UW  output_analog_7;
00990
00991   /*24-25*/ UW  input_analog_0;
00992   /*26-27*/ UW  input_analog_1;
00993   /*28-29*/ UW  input_analog_2;
00994   /*30-31*/ UW  input_analog_3;
00995   /*32-33*/ UW  input_analog_4;
00996   /*34-35*/ UW  input_analog_5;
00997   /*36-37*/ UW  input_analog_6;
00998   /*38-39*/ UW  input_analog_7;
00999
01000   /*40-41*/ UW  output_bank_0;
01001   /*42-43*/ UW  output_bank_1;
01002
01003   /*44-45*/ UW  input_bank_0;
01004   /*46-47*/ UW  input_bank_1;
01005
01006   /*48-51*/ UL  pulse_count_0;
01007   /*52-55*/ SL  zc_variable;
01008   /*56-59*/ SL  zd_variable;
01009
```

```
01010   /*60-61*/ UW output_bank_2;
01011   /*62-63*/ UW output_back_3;
01012
01013   /*64-65*/ UW input_bank_2;
01014   /*66-67*/ UW input_bank_3;
01015
01016 }; //GDataRecord47300_24EX
01017
01019 struct GDataRecord47162
01020 {
01021   /*Offset   type name        description*/
01022
01023   /*00*/      UB  header_0;
01024   /*01*/      UB  header_1;
01025   /*02*/      UB  header_2;
01026   /*03*/      UB  header_3;
01027
01028   /*04-05*/   UW  sample_number;
01029   /*06*/      UB  error_code;
01030   /*07*/    UB  general_status;
01031
01032   /*08-09*/ UW  output_analog_0;
01033   /*10-11*/ UW  output_analog_1;
01034   /*12-13*/ UW  output_analog_2;
01035   /*14-15*/ UW  output_analog_3;
01036   /*16-17*/ UW  output_analog_4;
01037   /*18-19*/ UW  output_analog_5;
01038   /*20-21*/ UW  output_analog_6;
01039   /*22-23*/ UW  output_analog_7;
01040
01041   /*24-25*/ UW  input_analog_0;
01042   /*26-27*/ UW  input_analog_1;
01043   /*28-29*/ UW  input_analog_2;
01044   /*30-31*/ UW  input_analog_3;
01045   /*32-33*/ UW  input_analog_4;
01046   /*34-35*/ UW  input_analog_5;
01047   /*36-37*/ UW  input_analog_6;
01048   /*38-39*/ UW  input_analog_7;
01049
01050   /*40*/      UB  output_byte_0;
01051   /*41*/      UB  output_byte_1;
01052   /*42*/      UB  output_byte_2;
01053
01054   /*43*/      UB  input_byte_0;
01055   /*44*/      UB  input_byte_1;
01056   /*45*/      UB  input_byte_2;
01057   /*46*/      UB  input_byte_3;
01058   /*47*/      UB  input_byte_4;
01059
01060   /*48-51*/ UL  pulse_count_0;
01061   /*52-55*/ SL  zc_variable;
01062   /*56-59*/ SL  zd_variable;
01063
01064   /*60-63*/ SL  encoder_0;
01065   /*64-67*/ SL  encoder_1;
01066   /*68-71*/ SL  encoder_2;
01067   /*72-75*/ SL  encoder_3;
01068
01069 }; //GDataRecord47162
01070
01072
01082 union GDataRecord
01083 {
01084   struct GDataRecord4000 dmc4000;
01085   struct GDataRecord4000 dmc4103;
01086   struct GDataRecord4000 dmc50000;
01087
01088   struct GDataRecord52000 dmc52000;
01089
01090   struct GDataRecord30000 dmc30000;
01091
01092   struct GDataRecord2103 dmc2103;
01093
01094   struct GDataRecord1806 dmc1806;
01095
01096   struct GDataRecord1802 dmc1802;
01097
01098   struct GDataRecord47000_ENC rio47000;
01099   struct GDataRecord47300_ENC rio47300;
01100   struct GDataRecord47300_24EX rio47300_24ex;
01101   struct GDataRecord47162 rio47162;
01102
01103   unsigned char byte_array[GALILDATARECORDMAXLENGTH];
01104 };
01105
01106
01107 #ifdef PACKOK
```

```
01108 #pragma pack() //return pack to default
01109 #else
01110 #error "Need to return structure packing for compiler"
01111 #endif
01112
01113 #endif //I_210405D9_D9EF_484F_8258_BB29A1BBA217
```

10.3 commands.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- GReturn commands (GCon g)

  Demonstrates various uses of GCommand() and GUtility().

10.3.1 Detailed Description

Function calls for the Command Example Project.
Definition in file commands.cpp.

10.3.2 Function Documentation

10.3.2.1 commands()

```
GReturn commands (
            GCon g )
```

Demonstrates various uses of GCommand() and GUtility().

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See commands_example.cpp for an example.
Definition at line 16 of file commands.cpp.
References e(), G_SMALL_BUFFER, G_UTIL_ERROR_CONTEXT, GCmd(), GCmdD(), GCmdI(), GCmdT(), GCommand(), and GUtility().
Referenced by main().

10.4 commands.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013
00014 using namespace std;
00015
00016 GReturn commands(GCon g)
00017 {
00018   char buf[G_SMALL_BUFFER]; //traffic buffer
00019   GSize read_bytes = 0; //bytes read in GCommand
00020   int value;
00021   double d_value;
00022
00023   cout << "*********************************************************************************\n";
00024   cout << "*************************    GCmdT() example    *************************\n";
```

```
00025   cout « "*********************************************************************************\n";
00026   cout « "GCmdT() will return a trimmed response of GCommand()\n";
00027   cout « "The command 'PR ?,?' will return the relative "
00028       "position of the A and B axes\n";
00029   e(GCommand(g, "PR ?,?", buf, G_SMALL_BUFFER, &read_bytes));
00030   cout « "«PR ?,? with GCommand(): " « buf « "»\n";
00031   e(GCmdT(g, "PR ?,?", buf, G_SMALL_BUFFER, NULL));
00032   cout « "«PR ?,? with GCmdT(): " « buf « "»\n";
00033   char* front; //this must not be a pointer on the heap, it will be modified.
00034   e(GCmdT(g, "MG TIME", buf, sizeof(buf), &front)); //Trim back and front.
00035   cout « "«MG TIME with GCmdT() and front trimmed: " « front « "»" « "\n\n";
00036
00037   cout « "*********************************************************************************\n";
00038   cout « "*************************    GCmdI() example   **************************\n";
00039   cout « "*********************************************************************************\n";
00040   cout « "GCmdI() will return the value of GCommand() parsed as an int\n";
00041   cout « "The command 'MG _LMS' will return the available "
00042       "space in the vector buffer of the S plane.\n";
00043   e(GCmdT(g, "MG _LMS", buf, G_SMALL_BUFFER, NULL));
00044   cout « "MG _LMS with GCmdT(): " « buf « "\n";
00045   e(GCmdI(g, "MG _LMS", &value));
00046   cout « "MG _LMS with GCmdI(): " « value « "\n\n";
00047
00048   cout « "*********************************************************************************\n";
00049   cout « "*************************    GCmd() example   ***************************\n";
00050   cout « "*********************************************************************************\n";
00051   cout « "GCmd() will execute the given command but does not return a value.\n";
00052   cout « "GCmd is useful for basic operations such as beginning"
00053       " motion or setting speed\n";
00054   e(GCmd(g, "BG A"));
00055   e(GCmd(g, "SP 5000"));
00056   cout « "GCmd(g, \"BG A\");\n";
00057   cout « "GCmd(g, \"SP 5000\");\n\n";
00058
00059
00060   cout « "*********************************************************************************\n";
00061   cout « "*************************    GCmdD() example   ***************************\n";
00062   cout « "*********************************************************************************\n";
00063   cout « "GCmdD() will return the value of GCommand parsed as a double\n";
00064   cout « "The command 'MG @AN[1]' will return the value of Analog Input 1\n";
00065   e(GCmdD(g, "MG @AN[1]", &d_value));
00066   cout « "MG @AN[1] with GCmdD(): " « d_value « "\n\n";
00067
00068   cout « "*********************************************************************************\n";
00069   cout « "************************    Galil Double Format   ************************\n";
00070   cout « "*********************************************************************************\n";
00071   double d_val = 0.00235;
00072   sprintf(buf, "%.4f", d_val);
00073   cout « "Galil Controllers expect double values to be formatted to 4 "
00074       "decimal places\n";
00075   cout « "Unformatted double value: " « d_val « "\n";
00076   cout « "Formatted double value rounded to 4 decimal places: " « buf « "\n\n";
00077
00078   cout « "*********************************************************************************\n";
00079   cout « "*******************    G_UTIL_ERROR_CONTEXT example   ********************\n";
00080   cout « "*********************************************************************************\n";
00081   //To check any OS errors - call GUtility with G_UTIL_ERROR_CONTEXT
00082   GSize size = sizeof(buf);
00083   GUtility(g, G_UTIL_ERROR_CONTEXT, buf, &size);
00084   cout « "GUtility() with G_UTIL_ERROR_CONTEXT: " « buf « "\n";
00085
00086   return  GALIL_EXAMPLE_OK;
00087 }
```

10.5 commands_example.cpp File Reference

```
#include "examples.h"
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

10.5.1 Detailed Description

See commands() for implementation of logic
Definition in file commands_example.cpp.

10.5.2 Function Documentation

10.5.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.
Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.
contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.
message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

10.6 commands_example.cpp

Go to the documentation of this file.

```
00001
00009 #include "examples.h"
00010
00011 using namespace std;
00012
00014
00018 int main(int argc, char * argv[])
00019 {
00020   char buf[G_SMALL_BUFFER];
00021
00022   //var used to refer to a unique connection. A valid connection is nonzero.
00023   GCon g = 0;
00024
00025   try
00026   {
00027     if (argc != 2) //Invalid number of arguments
00028     {
00029       cerr « "Incorrect number of arguments provided\n";
00030       cerr « "Usage: commands_example.exe <ADDRESS>\n";
00031       pause();
00032       return GALIL_EXAMPLE_ERROR;
00033     }
00034
```

```
00035     char* address = argv[1];  //Retrieve address from command line
00036     e(GOpen(address, &g)); //Opens a connection at the provided address
00037
00038     //Demonstrates various uses of GCommand() and GUtility().
00039     commands(g);
00040   }
00041   catch (GReturn gr)
00042   {
00043       error(g, gr); //see examples.h for error handling
00044       pause();
00045     return GALIL_EXAMPLE_ERROR;
00046   }
00047
00048   pause();
00049   return GALIL_EXAMPLE_OK;
00050 }
```

10.7 contour.cpp File Reference

```
#include "examples.h"
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
```

Functions

- bool load_buf (GCon g, const std::vector< int > &positions_A, const std::vector< int > &positions_B, int capacity, int &cmd)

  Loads contour buffer with commands from the given text file.
- std::vector< int > csv_to_vector (ifstream &is)

  Converts a file of comma separated values to a vector.
- GReturn contour (GCon g, char *fileA, char *fileB)

  Record user's training and plays back training through contour mode.

10.7.1 Detailed Description

Function calls for the Contour Example project
Definition in file contour.cpp.

10.7.2 Function Documentation

10.7.2.1 contour()

```
GReturn contour (
            GCon g,
            char * fileA,
            char * fileB )
```

Record user's training and plays back training through contour mode.

Parameters

g	Connection's handle.
fileA	A Path to a text file where training for Axis A will be recorded.
fileB	A Path to a text file where training for Axis B will be recorded.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See contour_example.cpp for an example.

Definition at line 20 of file contour.cpp.
References csv_to_vector(), e(), G_SMALL_BUFFER, GCmd(), GCmdI(), GMotionComplete(), GSleep(), load_buf(), record_position(), and vector().

10.7.2.2 csv_to_vector()

```
std::vector< int > csv_to_vector (
            ifstream & is )
```

Converts a file of comma separated values to a vector.
Definition at line 102 of file contour.cpp.
References G_SMALL_BUFFER, and vector().
Referenced by contour().

10.7.2.3 load_buf()

```
bool load_buf (
            GCon g,
            const std::vector< int > & positions_A,
            const std::vector< int > & positions_B,
            int capacity,
            int & cmd )
```

Loads contour buffer with commands from the given text file.
Definition at line 74 of file contour.cpp.
References e(), G_SMALL_BUFFER, GCmd(), and vector().
Referenced by contour().

10.8 contour.cpp

Go to the documentation of this file.

```
00001
00008 #include "examples.h"
00009
00010 #include <iostream> //std::cout
00011 #include <fstream>
00012 #include <string> //to_string, string, etc.
00013 #include <vector>
00014
00015 using namespace std;
00016 bool load_buf(GCon g, const std::vector<int>& positions_A,
00017         const std::vector<int>& positions_B, int capacity, int& cmd);
00018 std::vector<int> csv_to_vector(ifstream& is);
00019
00020 GReturn contour(GCon g, char* fileA, char* fileB)
00021 {
00022   char buf[G_SMALL_BUFFER]; //traffic buffer
00023   int rc; //Holds the status of the RC command
00024   int rd;
00025   int positionA = 0, positionB = 0;
00026
00027   record_position(g, fileA, fileB); //Record positional data on Axis A and B
00028
00029   ifstream isA, isB;
00030   isA.open(fileA);
00031   isB.open(fileB);
00032
00033   std::vector<int> positions_A = csv_to_vector(isA);
00034   std::vector<int> positions_B = csv_to_vector(isB);
00035
00036   e(GCmd(g, "SH AB")); //Set servo here
00037   e(GCmd(g, "PA 0, 0")); //Set current position to 0
00038   e(GMotionComplete(g, "AB")); //Wait for motion to complete
00039   e(GCmd(g, "CM AB")); //Put axis A & B in Contour Mode
00040   e(GCmd(g, "DT -1")); //Pauses contour mode to pre-load buffer
00041   e(GCmd(g, "CD 0,0")); //Pre load buffer with zeros to prevent under buffering
00042   e(GCmd(g, "CD 0,0")); //Pre load buffer with zeros to prevent under buffering
00043   e(GCmd(g, "CD 0,0")); //Pre load buffer with zeros to prevent under buffering
00044   e(GCmd(g, "DT 1")); //Sets time interval for contour mode to be 2 samples
00045
00046   int capacity = 0; //Holds the capacity of the contour buffer
00047   int cmd = 0; //Holds the counter for which position to send next
00048
00049   if (positions_A.size() != positions_B.size())
```

```
00050   {
00051     cout « "Error: The two datasets are not the same size\n";
00052     return GALIL_EXAMPLE_ERROR;
00053   }
00054
00055   do //loop while there are still commands in the buffer
00056   {
00057     //sleep while buffer is emptying
00058     GSleep(400);
00059
00060     //Stores the available space of the contour buffer in the capacity variable
00061     e(GCmdI(g, "CM?", &capacity));
00062
00063   } while (load_buf(g, positions_A, positions_B, capacity, cmd));
00064
00065   e(GCmd(g, "CD 0,0=0"));  //End contour mode
00066
00067   isA.close();
00068   isB.close();
00069
00070   return  GALIL_EXAMPLE_OK;
00071 }
00072
00074 bool load_buf(GCon g, const std::vector<int>& positions_A,
00075         const std::vector<int>& positions_B, int capacity, int& cmd)
00076 {
00077   char buf[G_SMALL_BUFFER]; //traffic buffer
00078   for (capacity; capacity > 0; capacity--) // Fully load contour buffer
00079   {
00080     // While there are commands in the position vectors
00081     if (cmd + 1 < positions_A.size())
00082     {
00083       //Subtract previous position from new position to get how far of a move to make
00084       int cdA = positions_A[cmd + 1] - positions_A[cmd];
00085
00086       //Subtract previous position from new position to get how far of a move to make
00087       int cdB = positions_B[cmd + 1] - positions_B[cmd];
00088
00089       sprintf(buf, "CD %d,%d", cdA, cdB);
00090       e(GCmd(g, buf)); //Send contour distance command
00091
00092       cmd++;
00093     }
00094     else
00095       return false;
00096   }
00097
00098   return true;
00099 }
00100
00102 std::vector<int> csv_to_vector(ifstream& is)
00103 {
00104   std::vector<int> positions;
00105   while (is.good()) //While there are still characters in the file
00106   {
00107     char position[G_SMALL_BUFFER];
00108
00109     //Get a char array of all the characters leading up to the next ','
00110     is.getline(position, 16, ',');
00111
00112     //Convert read value to an integer
00113     char* end;
00114     int i_position = strtol(position, &end, 10);
00115     positions.push_back(i_position); //Add value to the vector
00116   }
00117
00118   return positions;
00119 }
```

10.9 contour_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

10.9.1 Detailed Description

See contour() for implementation of logic
Definition in file contour_example.cpp.

10.9.2 Function Documentation

10.9.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.
Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.
contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.
message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

10.10 contour_example.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013 using namespace std;
00014
00016
00021 int main(int argc, char * argv[])
00022 {
00023   GReturn rc = GALIL_EXAMPLE_OK;
00024   char buf[G_SMALL_BUFFER];
00025
00026   //var used to refer to a unique connection. A valid connection is nonzero.
00027   GCon g = 0;
00028
00029   try
```

```
00030   {
00031     if (argc != 4) //Invalid number of arguments
00032     {
00033       cerr « "Incorrect number of arguments provided\n";
00034       cerr « "Usage: contour_example.exe <ADDRESS> <FileA> <FileB>\n";
00035       pause();
00036       return GALIL_EXAMPLE_ERROR;
00037     }
00038
00039     char* fileA = argv[2];
00040     char* fileB = argv[3];
00041     char* address = argv[1];  //Retrieve address from command line
00042     e(GOpen(address, &g)); //Opens a connection at the provided address
00043
00044     // Record user's training and plays back training through contour mode
00045     rc = contour(g, fileA, fileB);
00046   }
00047   catch (GReturn gr)
00048   {
00049     error(g, gr); //see examples.h for error handling
00050     pause();
00051     return GALIL_EXAMPLE_ERROR;
00052   }
00053
00054     pause();
00055   return GALIL_EXAMPLE_OK;
00056 }
```

10.11 examples.h File Reference

```
#include "gclib.h"
#include "gclibo.h"
#include <iostream>
#include <cstdio>
```

Macros

- #define _CRT_SECURE_NO_WARNINGS
- #define GALIL_EXAMPLE_OK 0
- #define GALIL_EXAMPLE_ERROR -100

Functions

- void e (GReturn rc)

  A trivial, C++ style return code check used in Galil's examples and demos.
- void error (GCon g, GReturn rc)

  An example of error handling and debugging information.
- int pause ()

  Pauses console apps for a user key stroke.
- GReturn position_tracking (GCon g, int speed=5000)

  Puts controller into Position Tracking Mode and accepts user-entered positions.
- GReturn jog (GCon g)

  Puts controller into Jog Mode and accepts user input to adjust the speed.
- GReturn vector (GCon g, char *file)

  Puts controller into Vector Mode and accepts a file defining vector points.
- GReturn ip_assigner (char *serial_num, int address)

  Assigns controller an IP Adress given a serial number and a 1 byte address.
- GReturn commands (GCon g)

  Demonstrates various uses of GCommand() and GUtility().
- GReturn motion_complete (GCon g)

  Uses interrupts to track when the motion of controller is completed.
- GReturn message (GCon g)

Demonstrates how to receive messages from the controller and detect differences in Trace and crashed code.

- GReturn record_position (GCon g, char *fileA, char *fileB)

  Record user's training and saves to a text file.

- GReturn contour (GCon g, char *fileA, char *fileB)

  Record user's training and plays back training through contour mode.

- GReturn remote_server (const char *server_name)

  Publishes local gcaps server to the network.

- GReturn remote_client ()

  Lists available remote servers and allows connection to remote server.

10.11.1 Detailed Description

Header file for Galil gclib example projects.
Definition in file examples.h.

10.11.2 Macro Definition Documentation

10.11.2.1 _CRT_SECURE_NO_WARNINGS

```
#define _CRT_SECURE_NO_WARNINGS
```

Definition at line 16 of file examples.h.

10.11.2.2 GALIL_EXAMPLE_ERROR

```
#define GALIL_EXAMPLE_ERROR -100
```

Definition at line 25 of file examples.h.

10.11.2.3 GALIL_EXAMPLE_OK

```
#define GALIL_EXAMPLE_OK 0
```

Definition at line 24 of file examples.h.

10.11.3 Function Documentation

10.11.3.1 commands()

```
GReturn commands (
            GCon g )
```

Demonstrates various uses of GCommand() and GUtility().

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See commands_example.cpp for an example.
Definition at line 16 of file commands.cpp.
References e(), G_SMALL_BUFFER, G_UTIL_ERROR_CONTEXT, GCmd(), GCmdD(), GCmdI(), GCmdT(), GCommand(), and GUtility().
Referenced by main().

10.11.3.2 contour()

```
GReturn contour (
            GCon g,
```

```
            char * fileA,
            char * fileB )
```

Record user's training and plays back training through contour mode.

Parameters

g	Connection's handle.
fileA	A Path to a text file where training for Axis A will be recorded.
fileB	A Path to a text file where training for Axis B will be recorded.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See contour_example.cpp for an example.
Definition at line 20 of file contour.cpp.
References csv_to_vector(), e(), G_SMALL_BUFFER, GCmd(), GCmdI(), GMotionComplete(), GSleep(), load_buf(), record_position(), and vector().

10.11.3.3 e()

```
void e (
            GReturn rc )  [inline]
```

A trivial, C++ style return code check used in Galil's examples and demos.
Throws GReturn if return value is not G_NO_ERROR. See Commands_Example.cpp for example usage and catch() handler.
Definition at line 33 of file examples.h.
References G_NO_ERROR, and vector().
Referenced by check_interrupts(), commands(), contour(), GclibJava::GArrayUpload(), H_ArrayDownloadFromMemory(), ip_assigner(), jog(), load_buf(), load_buffer(), main(), message(), motion_complete(), position_tracking(), record_position(), remote_client(), remote_server(), vector(), and write_array_to_file().

10.11.3.4 error()

```
void error (
            GCon g,
            GReturn rc )  [inline]
```

An example of error handling and debugging information.
Definition at line 40 of file examples.h.
References G_BAD_RESPONSE_QUESTION_MARK, G_NO_ERROR, G_SMALL_BUFFER, G_UTIL_ERROR_CONTEXT, GCommand(), GError(), GUtility(), and vector().
Referenced by GError(), and main().

10.11.3.5 ip_assigner()

```
GReturn ip_assigner (
            char * serial_num,
            int address )
```

Assigns controller an IP Adress given a serial number and a 1 byte address.

Parameters

serial_num	Serial Number of the controller.
address	A 1 byte address that defines the last byte of the IP Address.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See ip_assigner_example.cpp for an example.
This function will listen on the network for controllers requesting an IP Address. If a detected controller matches the serial number provided by the user, a new IP Address will be assigned based on the first 3 bytes of the detected IP Address combined with the user defined 1 byte address.
Definition at line 26 of file ip_assigner.cpp.
References e(), G_SMALL_BUFFER, GAssign(), GCmd(), GInfo(), GIpRequests(), GOpen(), string_split(), and vector().

10.11.3.6 jog()

```
GReturn jog (
            GCon g )
```

Puts controller into Jog Mode and accepts user input to adjust the speed.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See jog_example.cpp for an example.

Key	Usage
q	Quit Jogging
a	-2000 counts / second
s	-500 counts / second
d	+500 counts / second
f	+2000 counts / second
r	Direction Reversal

Definition at line 29 of file jog.cpp.
References e(), G_CONNECTION_NOT_ESTABLISHED, G_SMALL_BUFFER, GCmd(), GMotionComplete(), and vector().

10.11.3.7 message()

```
GReturn message (
            GCon g )
```

Demonstrates how to receive messages from the controller and detect differences in Trace and crashed code.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See message_example.cpp for an example.
Definition at line 14 of file message.cpp.
References e(), G_NO_ERROR, G_SMALL_BUFFER, G_UTIL_GCAPS_KEEPALIVE, GCmd(), GMessage(), GProgramDownload(), GUtility(), message(), and vector().
Referenced by Examples::Message(), and message().

10.11.3.8 motion_complete()

```
GReturn motion_complete (
            GCon g )
```

Uses interrupts to track when the motion of controller is completed.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See motion_complete_example.cpp for an example.
Definition at line 18 of file motion_complete.cpp.
References check_interrupts(), e(), G_NO_ERROR, G_SMALL_BUFFER, G_UNSUPPORTED_FUNCTION, GCmd(), GCmdT(), GCommand(), GInterrupt(), GTimeout(), and vector().

10.11.3.9 pause()

```
int pause ( )  [inline]
```

Pauses console apps for a user key stroke.
Definition at line 62 of file examples.h.
Referenced by main().

10.11.3.10 position_tracking()

```
GReturn position_tracking (
            GCon g,
            int speed = 5000 )
```

Puts controller into Position Tracking Mode and accepts user-entered positions.

Parameters

g	Connection's handle.
speed	Optional speed of the controller in Position Tracking Mode. Default value of 5000.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See position_tracking_example.cpp for an example.
Definition at line 15 of file position_tracking.cpp.
References e(), G_CONNECTION_NOT_ESTABLISHED, G_SMALL_BUFFER, GCmd(), GMotionComplete(), and vector().

10.11.3.11 record_position()

```
GReturn record_position (
            GCon g,
            char * fileA,
            char * fileB )
```

Record user's training and saves to a text file.

Parameters

g	Connection's handle.
fileA	A Path to a text file where training for Axis A will be recorded.
fileB	A Path to a text file where training for Axis B will be recorded.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See record_position_example.cpp for an example.
Definition at line 20 of file record_position.cpp.
References e(), GCmd(), GCmdI(), GProgramDownload(), GSleep(), vector(), and write_array_to_file().
Referenced by contour().

10.11.3.12 remote_client()

```
GReturn remote_client ( )
```

Lists available remote servers and allows connection to remote server.

Returns

The success status or error code of the function.See gclib_errors.h for possible values.

See remote_client_example for an example.

Key	Usage
q	Quit
s	List available servers on then network
h	List available hardware on the current server
0-9	Connect to server instance by number
l	Connect back to local server

Definition at line 89 of file remote_client.cpp.
References e(), G_SMALL_BUFFER, GAddresses(), GListServers(), GSetServer(), and vector().

10.11.3.13 remote_server()

```
GReturn remote_server (
            const char * server_name )
```

Publishes local gcaps server to the network.

Parameters

Name	to publish server under.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See remote_server_example for an example.

Key	Usage
q	Quit
p	Publish this server to the network
r	Remove this server from the network

Definition at line 39 of file remote_server.cpp.
References e(), G_SMALL_BUFFER, GPublishServer(), and vector().

10.11.3.14 vector()

```
GReturn vector (
            GCon g,
            char * file )
```

Puts controller into Vector Mode and accepts a file defining vector points.

Parameters

g	Connection's handle.
file	A Path to a file that defines vector commands.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See vector_example.cpp for an example.
Example text file:

```
VP -2219,-2667
VP -2523,-2832
VP 2844,-1425
VP 728,1971
VP 2127,183
VP -997,688
VP 725,-1893
VP 527,2899
VP -37,2523
VP 1277,1425
VP 857,2388
VP 1096,-1694
CR 1000,0,90
```

Definition at line 36 of file vector.cpp.
References e(), G_BAD_FILE, G_CONNECTION_NOT_ESTABLISHED, GCmd(), GCmdI(), GMotionComplete(), GSleep(), load_buffer(), and vector().
Referenced by py::__init__(), AddressConvert(), check_interrupts(), Examples::Commands(), Examples::Contour(), contour(), csv_to_vector(), e(), error(), GclibJava::finalize(), glib::GAddresses(), GAddresses(), py::GAddresses(), py::GArrayDownload(), glib::GArrayDownload(), GclibJava::GArrayDownload(), GArrayDownloadFile(), py::GArrayDownloadFile(), GclibJava::GArrayDownloadFile(), glib::GArrayDownloadFile(), py::GArrayUpload(), glib::GArrayUpload(), GclibJava::GArrayUpload(), GArrayUploadFile(), py::GArrayUploadFile(), GclibJava::GArrayUploadFile(), GclibJava::GArrayUploadFile(), glib::GArrayUploadFile(), GAssign(), py::GAssign(), glib::GAssign(), GclibJava::GAssign(), glib::glib(), py::GClose(), GCmd(), GCmdD(), glib::GCmdD(), GCmdI(), glib::GCmdI(), GCmdT(), py::GCommand(), GclibJava::GCommand(), glib::GCommand(), GError(), py::GFirmwareDownload(), glib::GFirmwareDownload(), GclibJava::GFirmwareDownload(), glib::GInfo(), GInfo(), py::GInfo(), glib::GInterrupt(), GclibJava::GInterrupt(), py::GInterrupt(), glib::GIpRequests(), GIpRequests(), py::GIpRequests(), glib::GListServers(), GListServers(), glib::GMessage(), py::GMessage(), GMotionComplete(), py::GMotionComplete(), glib::GMotionComplete(), py::GOpen(), glib::GOpen(), GclibJava::GOpen(), py::GProgramDownload(), GclibJava::GProgramDownload(), GclibJava::GProgramDownload(), glib::GProgramDownload(), GProgramDownloadFile(), py::GProgramDownloadFile(), glib::GProgramDownloadFile(), GclibJava::GProgramDownloadFile(), GclibJava::GProgramDownloadFile(), glib::GProgramUpload(), py::GProgramUpload(), GProgramUploadFile(), py::GProgramUploadFile(), glib::GProgramUploadFile(), GclibJava::GProgramUploadFile(), GPublishServer(), glib::GPublishServer(), GclibJava::GPublishServer(), glib::GRead(), glib::GRecord< T >(), glib::GRecordRate(), GRecordRate(), glib::GRemoteConnections(), GRemoteConnections(), glib::GServerStatus(), GServerStatus(), GSetServer(), glib::GSetServer(), GclibJava::GSetServer(), GSetupDownloadFile(), py::GSetupDownloadFile(), glib::GSetupDownloadFile(), py::GSleep(), py::GTimeout(), glib::GVersion(), GVersion(), py::GVersion(), GWaitForBool(), glib::GWrite(), H_AddArray(), H_ArrayAddElement(), H_ArrayDownloadFromMemory(), H_CreateArrayNode(), H_DownloadArraysFromList(), H_DownloadData(), H_FindSector(), H_FreeArrays(), H_InitArrayNode(), H_UploadArrayToList(), H_WriteArrayCsv(), ip_assigner(), Examples::IP_Assigner(), Examples::Jog(), jog(), load_buf(), load_buffer(), Remote_Client_Example::Main(), Commands_Example::Main(), Contour_Example::Main(), IP_Assigner_Example::Main(), Jog_Example::Main(), Message_Example::Main(), Motion_Complete_Example::Main(), Position_Tracking_Example::Main(), Record_Position_Example::Main(), Remote_Server_Example::Main(), Vector_Mode_Example::Main(), Examples::Message(), message(), Examples::Motion_Complete(), motion_complete(), Examples::Position_Tracking(), position_tracking(), Examples::PrintError(), Examples::Record_Position(), record_position(), remote_client(), Examples::Remote_Client(), remote_server(), Examples::Remote_Server(), string_split(), vector(), Examples::Vector_Mode(), write_array_to_file(), x_arrays(), x_dr_motioncomplete(), x_e(), x_ei_motioncomplete(), x_gcommand(), x_ginterrupt(), x_gmessage(), x_gmotioncomplete(), x_gread_gwrite(), x_grecord(), x_nonblocking(), and x_programs().

10.12 examples.h

Go to the documentation of this file.

```
00001
00013 #ifndef examples_h
00014 #define examples_h
00015
00016 #define _CRT_SECURE_NO_WARNINGS //use traditional C calls like sprintf()
00017
00018 #include "gclib.h"
00019 #include "gclibo.h"
00020
00021 #include <iostream> //std::cout
00022 #include <cstdio> //std::getchar
00023
00024 #define GALIL_EXAMPLE_OK 0//return code for correct code execution
00025 #define GALIL_EXAMPLE_ERROR -100 //return code for error in example code
00026
00027
00029
00033 inline void e(GReturn rc)
00034 {
00035   if (rc != G_NO_ERROR)
00036     throw rc;
00037 }
00038
00040 inline void error(GCon g, GReturn rc)
00041 {
00042     char buf[G_SMALL_BUFFER];
00043     GError(rc, buf, G_SMALL_BUFFER); //Get Error Information
00044     std::cout « buf « '\n';
00045   if (g)
00046   {
00047       GSize size = sizeof(buf);
00048       GUtility(g, G_UTIL_ERROR_CONTEXT, buf, &size);
00049
00050       if (buf[0])
00051         std::cout « buf « '\n'; //further context
00052
00053     if ((rc == G_BAD_RESPONSE_QUESTION_MARK)
00054         && (GCommand(g, "TC1", buf, G_SMALL_BUFFER, 0) == G_NO_ERROR))
00055     {
00056         std::cout « buf « '\n'; //Error code from controller
00057     }
00058   }
00059 }
00060
00062 inline int pause()
00063 {
00064
00065     std::cout « "Enter any key to exit..." « std::endl;
00066     return std::getchar();
00067
00068 }
00069
00071
00079 GReturn position_tracking(GCon g, int speed = 5000);
00080
00082
00089 GReturn jog(GCon g);
00090
00092
00100 GReturn vector(GCon g, char* file);
00101
00103
00111 GReturn ip_assigner(char* serial_num, int address);
00112
00114
00121 GReturn commands(GCon g);
00122
00124
00131 GReturn motion_complete(GCon g);
00132
00134
00141 GReturn message(GCon g);
00142
00144
00153 GReturn record_position(GCon g, char* fileA, char* fileB);
00154
00156
00165 GReturn contour(GCon g, char* fileA, char* fileB);
00166
00168
00175 GReturn remote_server(const char* server_name);
00176
00178
```

```
00183 GReturn remote_client();
00184
00185 #endif //examples_h
00186
00187
```

10.13 ip_assigner.cpp File Reference

```
#include "examples.h"
#include <iostream>
#include <vector>
#include <string.h>
```

Typedefs

- typedef std::vector< string > tokens

Functions

- tokens string_split (const string &str, const string &token)

  Splits a string into a vector based on a token.
- GReturn ip_assigner (char *serial_num, int address)

  Assigns controller an IP Adress given a serial number and a 1 byte address.

10.13.1 Detailed Description

Function calls for the IP Assigner Example Project.
Definition in file ip_assigner.cpp.

10.13.2 Typedef Documentation

10.13.2.1 tokens

```
typedef std::vector<string> tokens
```

Definition at line 16 of file ip_assigner.cpp.

10.13.3 Function Documentation

10.13.3.1 ip_assigner()

```
GReturn ip_assigner (
            char * serial_num,
            int address )
```

Assigns controller an IP Adress given a serial number and a 1 byte address.

Parameters

serial_num	Serial Number of the controller.
address	A 1 byte address that defines the last byte of the IP Address.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See ip_assigner_example.cpp for an example.
This function will listen on the network for controllers requesting an IP Address. If a detected controller matches the serial number provided by the user, a new IP Address will be assigned based on the first 3 bytes of the detected IP Address combined with the user defined 1 byte address.

Definition at line 26 of file ip_assigner.cpp.
References e(), G_SMALL_BUFFER, GAssign(), GCmd(), GInfo(), GIpRequests(), GOpen(), string_split(), and vector().

10.13.3.2 string_split()

```
tokens string_split (
            const string & str,
            const string & token )
```

Splits a string into a vector based on a token.
Definition at line 96 of file ip_assigner.cpp.
References vector().
Referenced by ip_assigner().

10.14 ip_assigner.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013 #include <vector>
00014 #include <string.h>
00015 using namespace std;
00016 typedef std::vector<string> tokens;
00017
00018 tokens string_split(const string& str, const string& token);
00019
00026 GReturn ip_assigner(char* serial_num, int address)
00027 {
00028   char buf[G_SMALL_BUFFER]; //traffic buffer
00029   bool controller_found = false;
00030
00031   do //Loop while no requests are found.
00032   {
00033     cout « "Searching...\n";
00034
00035     //listen for ~5 seconds for controllers requesting IP addresses.
00036     e(GIpRequests(buf, G_SMALL_BUFFER));
00037     cout « buf « "\n";
00038   } while (strlen(buf) <= 1);
00039
00040   //Get a list of all found controllers
00041   tokens controllers = string_split(buf, "\n");
00042   //Iterate over all found controllers
00043   for (tokens::iterator it = controllers.begin(); it != controllers.end(); it++)
00044   {
00045     const string& controller = *it;
00046     //Get a list of the parameters of the controller
00047     tokens controller_params = string_split(controller, ", ");
00048     //Parameters are ordered as:
00049     //[Model #], [Serial #], [MAC Address], [Connection Name], [IP Address]
00050     if (controller_params.size() < 5)
00051     {
00052       cerr « "Unexpected controller format";
00053       return GALIL_EXAMPLE_ERROR;
00054     }
00055     const char* mac = controller_params[2].c_str();
00056     const char* ip = controller_params[4].c_str();
00057     //If controller contains the user entered serial number
00058     if (string(serial_num) == controller_params[1])
00059     {
00060       controller_found = true;
00061       int ip1 = 0, ip2 = 0, ip3 = 0, ip4 = 0;
00062       //Parses the IP Address and breaks it into chunks in order to change the last byte
00063       sscanf(ip, "%d.%d.%d.%d", &ip1, &ip2, &ip3, &ip4);
00064
00065       char ipaddress[G_SMALL_BUFFER];
00066       //Store the first 3 bytes of the original IP Address with the user supplied address byte
00067       sprintf(ipaddress, "%d.%d.%d.%d", ip1, ip2, ip3, address);
00068
00069       //Assign the controller using the newly created IP Address and MAC Address
00070       e(GAssign(ipaddress, mac));
00071
00072       char info[G_SMALL_BUFFER];
00073       GCon g = 0;
00074       e(GOpen(ipaddress, &g)); //Opens a connection to the controller
00075       if (g != 0)
```

```
00076       {
00077         //Burns newly assigned IP Address to non-volatile EEPROM memory
00078         e(GCmd(g, "BN"));
00079
00080         e(GInfo(g, info, G_SMALL_BUFFER));
00081         cout « info;
00082       }
00083       break;
00084     }
00085   }
00086
00087   if (!controller_found)
00088   {
00089     cout « "No controller matched the entered serial number";
00090   }
00091
00092   return  GALIL_EXAMPLE_OK;
00093 }
00094
00096 tokens string_split(const string& str, const string& token)
00097 {
00098   tokens split;  // A list that will contain strings separated by the given token
00099   int prev_position = 0 - token.length(); // Treat first position as if it came after a token
00100   int position = str.find(token); // Find position of first token
00101
00102   while (position != string::npos) // Find new position of token in string
00103   {
00104     // Add characters between previous token and next token to the list
00105     split.push_back(str.substr(prev_position + token.length(),
00106                     position - prev_position - token.length()));
00107     prev_position = position;
00108     position = str.find(token, prev_position + token.length());
00109   }
00110
00111   // If there are any remaining characters
00112   if (str.length() > prev_position + token.length())
00113   {
00114     // Add remaining characters to the list
00115     split.push_back(str.substr(prev_position + token.length(),
00116                     str.length() - prev_position - token.length()));
00117   }
00118
00119   return split;
00120 }
```

10.15 ip_assigner_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

10.15.1 Detailed Description

See ip_assigner() for implementation of logic
Definition in file ip_assigner_example.cpp.

10.15.2 Function Documentation

10.15.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.

Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.
contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.
message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

10.16 ip_assigner_example.cpp

Go to the documentation of this file.

```
00001
00009 #include "examples.h"
00010
00011 #include <iostream> //std::cout
00012 using namespace std;
00013
00015
00019 int main(int argc, char * argv[])
00020 {
00021   GReturn rc = GALIL_EXAMPLE_OK;
00022   char buf[G_SMALL_BUFFER];
00023
00024   try
00025   {
00026     if (argc != 3) //Invalid number of arguments
00027     {
00028       cerr « "Incorrect number of arguments provided\n";
00029       cerr « "Usage: ipassigner_example.exe <SERIAL #> <1 Byte Address>\n";
00030       pause();
00031       return GALIL_EXAMPLE_ERROR;
00032     }
00033     else
00034     {
00035       char* serial_num = argv[1];
00036
00037       //Retrieve address from the command line and convert to int
00038       char* end;
00039       int address = strtol(argv[2], &end, 10);
00040       if (*end != '\0' || address < 0 || address > 255) //If invalid address
00041       {
00042         cerr « "Please enter a number between 0 and 255 for the address."
00043           " This will be used as the last number in the IP Address\n"
00044         "Usage: ipassigner_example.exe <SERIAL #> <1 Byte Address>\n";
00045         return GALIL_EXAMPLE_ERROR;
00046       }
00047       //Assigns controller an IP Adress given a serial number and a 1 byte address
00048       rc = ip_assigner(serial_num, address);
00049     }
00050   }
```

```
00051   catch (GReturn gr)
00052   {
00053     error(nullptr, gr); //see examples.h for error handling
00054     pause();
00055     return GALIL_EXAMPLE_ERROR;
00056   }
00057
00058     pause();
00059   return GALIL_EXAMPLE_OK;
00060 }
```

10.17 jog.cpp File Reference

```
#include "examples.h"
#include <conio.h>
#include <iostream>
```

Functions

- GReturn jog (GCon g)

  Puts controller into Jog Mode and accepts user input to adjust the speed.

10.17.1 Detailed Description

Function calls for the Jog Example Project.
Definition in file jog.cpp.

10.17.2 Function Documentation

10.17.2.1 jog()

```
GReturn jog (
            GCon g )
```

Puts controller into Jog Mode and accepts user input to adjust the speed.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See jog_example.cpp for an example.

Key	Usage
q	Quit Jogging
a	-2000 counts / second
s	-500 counts / second
d	+500 counts / second
f	+2000 counts / second
r	Direction Reversal

Definition at line 29 of file jog.cpp.
References e(), G_CONNECTION_NOT_ESTABLISHED, G_SMALL_BUFFER, GCmd(), GMotionComplete(), and vector().

10.18 jog.cpp

Go to the documentation of this file.

```
00001
00008 #include "examples.h"
00009
00010 #ifdef _WIN32
00011   #include <conio.h>
00012 #elif __linux__
00013   #include <ncurses.h>
00014 #endif
00015
00016 #include <iostream> //std::cout
00017 using namespace std;
00018
00029 GReturn jog(GCon g)
00030 {
00031   char buf[G_SMALL_BUFFER]; //traffic buffer
00032   if (g == 0) //Bad connection
00033   {
00034     cerr « "There was an error with the connection." « endl;
00035     return G_CONNECTION_NOT_ESTABLISHED;
00036   }
00037
00038 #ifdef __linux__
00039   //These functions set up the ncurses library to capture keyboard input
00040   initscr(); // Initialization of ncurses library
00041   cbreak();  // Capture one character at a time
00042   noecho();  // Do not write back entered characters to the console
00043 #endif
00044
00045   e(GCmd(g, "ST"));     // stop all motors
00046   e(GMotionComplete(g, "A")); //Wait for motion to complete
00047   e(GCmd(g, "SHA"));     // Set servo here
00048   e(GCmd(g, "DPA=0"));  // Start position at absolute zero
00049   e(GCmd(g, "JGA=0"));  // Start jogging with 0 speed
00050   e(GCmd(g, "BGA"));   // Begin motion on A Axis
00051
00052   int isJogging = 1;
00053   int speed = 0;
00054
00055   char instructions[] = "Enter a character on the keyboard to change the"
00056               " motor's speed:\n<q> Quit\n<a> -2000 counts/s\n"
00057               "<s> -500  counts/s\n<d> +500  counts/s\n<f> "
00058               "+2000 counts/s\n<r> Direction Reversal\n";
00059
00060 #ifdef _WIN32
00061   cout « instructions;
00062 #elif __linux__
00063   printw(instructions); //Print instructions to console
00064 #endif
00065
00066   while (isJogging)
00067   {
00068     sprintf(buf, "JGA=%d", speed);
00069     e(GCmd(g, buf)); // Set speed
00070 #ifdef _WIN32
00071     cout « "Jog Speed: " « speed « "\n";
00072     switch (_getch()) //Capture keypress
00073 #elif __linux__
00074     sprintf(buf, "Jog Speed: %d\n", speed);
00075     printw(buf); //Print speed to console
00076     switch (getch()) //Capture keypress
00077 #endif
00078     {
00079       case 'q': //Quit Jogging
00080         isJogging = 0;
00081         break;
00082       case 'a': //Large speed change in negative direction
00083         speed -= 2000;
00084         break;
00085       case 's': //Small speed change in negative direction
00086         speed -= 500;
00087         break;
00088       case 'd': //Small speed change in positive direction
00089         speed += 500;
00090         break;
00091       case 'f': //Large speed change in positive direction
00092         speed += 2000;
00093         break;
00094       case 'r': //Reverse direction of speed
00095         speed *= -1;
00096         break;
00097     }
00098   }
00099
```

```
00100 #if __linux__
00101   endwin(); //Restores terminal to previous state
00102 #endif
00103   e(GCmd(g, "ST")); // Stop all motors
00104   e(GMotionComplete(g, "A")); //Wait for motion to complete
00105
00106   return  GALIL_EXAMPLE_OK;
00107 }
```

10.19 jog_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

10.19.1 Detailed Description

See jog() for implementation of logic
Definition in file jog_example.cpp.

10.19.2 Function Documentation

10.19.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.
Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.
contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.
message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with

the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

10.20 jog_example.cpp

Go to the documentation of this file.

```
00001
00011 #include "examples.h"
00012
00013 #include <iostream> //std::cout
00014 using namespace std;
00015
00017
00022 int main(int argc, char * argv[])
00023 {
00024   GReturn rc = GALIL_EXAMPLE_OK;
00025   char buf[G_SMALL_BUFFER];
00026
00027   //var used to refer to a unique connection. A valid connection is nonzero.
00028   GCon g = 0;
00029
00030   try
00031   {
00032     if (argc != 2) //Invalid number of arguments
00033     {
00034       cerr << "Incorrect number of arguments provided\n";
00035       cerr << "Usage: jog_example.exe <ADDRESS>\n";
00036       pause();
00037       return GALIL_EXAMPLE_ERROR;
00038     }
00039
00040     char* address = argv[1];  //Retrieve address from command line
00041     e(GOpen(address, &g)); //Opens a connection at the provided address
00042
00043     //Puts controller into Jog Mode and accepts user input to adjust the speed
00044     rc = jog(g);
00045   }
00046   catch (GReturn gr)
00047   {
00048     error(g, gr);
00049     pause();
00050     return GALIL_EXAMPLE_ERROR;
00051   }
00052
00053     pause();
00054   return GALIL_EXAMPLE_OK;
00055 }
```

10.21 message.cpp File Reference

```
#include "examples.h"
#include <iostream>
#include <string.h>
```

Functions

- GReturn message (GCon g)

  Demonstrates how to receive messages from the controller and detect differences in Trace and crashed code.

10.21.1 Detailed Description

Function calls for the Message Example project
Definition in file message.cpp.

10.21.2 Function Documentation

10.21.2.1 message()

```
GReturn message (
            GCon g )
```

Demonstrates how to receive messages from the controller and detect differences in Trace and crashed code.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See message_example.cpp for an example.
Definition at line 14 of file message.cpp.
References e(), G_NO_ERROR, G_SMALL_BUFFER, G_UTIL_GCAPS_KEEPALIVE, GCmd(), GMessage(), GProgramDownload(), GUtility(), message(), and vector().
Referenced by Examples::Message(), and message().

10.22 message.cpp

Go to the documentation of this file.

```
00001
00008 #include "examples.h"
00009
00010 #include <iostream> //std::cout
00011 #include <string.h>
00012 using namespace std;
00013
00014 GReturn message(GCon g)
00015 {
00016   char buf[G_SMALL_BUFFER]; //traffic buffer
00017   cout << "****************************************************************\n";
00018   cout << "Example GMessage() usage\n";
00019   cout << "****************************************************************\n";
00020
00021   e(GCmd(g, "TR0")); // Turn off trace
00022
00023   //This program will force one message to appear as two separate packets.
00024   e(GProgramDownload(g,
00025     "MG \"HELLO \" {N}\r"
00026     "MG \"WORLD \"\r"
00027     "EN", 0));
00028   e(GCmd(g, "XQ")); //Begins execution of program on controller
00029
00030   int rc = 0;
00031   char message[G_SMALL_BUFFER];
00032   int b = 0; //iterator for buf
00033   int m = 0; //iterator for message
00034
00035   // It is important to note that a message can be too large to read in one
00036   // GMessage() call. Keep calling GMessage() while there are no errors to
00037   // get the full message.
00038
00039   //While still receiving messages
00040   while ((rc = GMessage(g, buf, G_SMALL_BUFFER)) == G_NO_ERROR)
00041   {
00042     b = 0; //reset buffer index
00043
00044     while (buf[b] != '\0') //While message characters are in the buffer
00045     {
00046       message[m] = buf[b]; //Copy chars from buffer to message
00047
00048       //If the message ends in "\r\n" its ready to be terminated
00049       if (m > 0 && message[m] == '\n' && message[m - 1] == '\r')
00050       {
00051         message[m + 1] = '\0'; //Null terminate the message
00052         cout << '<' << message << ">\n";
00053         m = 0;  //Reset message index
00054       }
00055       else
00056       {
```

```
00057         m++; //Increment message index
00058       }
00059
00060       b++; //Increment buf index
00061     }
00062   }
00063
00064   //Downloads program to the controller
00065   e(GCmd(g, "TR1")); // Turn on trace
00066   e(GProgramDownload(g, "i=0\r"
00067     "#A\r"
00068     "MGi\r"
00069     "i=i+1\r"
00070     "WT100\r"
00071     "JP#A,i<1\r"
00072     "i=i/0\r"
00073     "EN", 0));
00074   e(GCmd(g, "XQ")); //Begins execution of program on controller
00075
00076   m = 0; //Reset message buffer
00077
00078   // Lines returned by GMessage() can be one of three types:
00079   // 1) Standard Lines begin with a space (" ")
00080   // 2) Crashed code begins with a question mark ("?")
00081   // 3) Trace Lines begin with a line number ("1,6,15...")
00082
00083   //While still receiving messages
00084   while ((rc = GMessage(g, buf, G_SMALL_BUFFER)) == G_NO_ERROR)
00085   {
00086     b = 0; //reset buf index
00087
00088     while (buf[b] != '\0') //While message characters are in the buffer
00089     {
00090       message[m] = buf[b]; //Copy chars from buffer to message
00091
00092       //If the message ends in "\r\n" its ready to be terminated
00093       if (m > 0 && message[m] == '\n' && message[m - 1] == '\r')
00094       {
00095         message[m + 1] = '\0'; //Null terminate the message
00096
00097         if (message[0] == ' ') //Standard Lines begin with a space (" ")
00098           cout « "Standard Line: ";
00099         else if (message[0] == '?') //Crashed code begins with a question mark ("?")
00100           cout « "Crashed Code: ";
00101         else //Trace Lines begin with a line number ("1,6,15...")
00102           cout « "Trace Line: ";
00103
00104         cout « message;
00105
00106         m = 0;  //Reset message index
00107       }
00108       else
00109       {
00110         m++; //Increment message index
00111       }
00112
00113       b++; //Increment buf index
00114     }
00115   }
00116
00117   // If no communication has been made to gcaps for 10 minutes the connection
00118   // will expire. This can be prevented by periodically sending the GUtility()
00119   // Keep Alive command
00120   e(GUtility(g, G_UTIL_GCAPS_KEEPALIVE, NULL, NULL));
00121
00122   return  GALIL_EXAMPLE_OK;
00123 }
```

10.23 message_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

10.23.1 Detailed Description

See message() for implementation of logic
Definition in file message_example.cpp.

10.23.2 Function Documentation

10.23.2.1 main()

```
int main (
              int argc,
              char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.
Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.
contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.
message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

10.24 message_example.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013 using namespace std;
00014
00016
00020 int main(int argc, char * argv[])
00021 {
00022   GReturn rc = GALIL_EXAMPLE_OK;
00023   char buf[G_SMALL_BUFFER];
00024
00025   //var used to refer to a unique connection. A valid connection is nonzero.
00026   GCon g = 0;
00027
00028   try
```

```
00029   {
00030     if (argc != 2) //Invalid number of arguments
00031     {
00032       cerr « "Incorrect number of arguments provided\n";
00033       cerr « "Usage: message_example.exe <ADDRESS>\n";
00034       pause();
00035       return GALIL_EXAMPLE_ERROR;
00036     }
00037
00038     char* address = argv[1];  //Retrieve address from command line
00039     sprintf(buf, "%s --subscribe MG", address);
00040     e(GOpen(buf, &g)); //Opens a connection at the provided address
00041
00042     //Demonstrates how to receive messages from the controller
00043     //and detect differences in Trace and crashed code.
00044     rc = message(g);
00045   }
00046   catch (GReturn gr)
00047   {
00048     error(g, gr); //see examples.h for error handling
00049     pause();
00050     return GALIL_EXAMPLE_ERROR;
00051   }
00052
00053   pause();
00054   return GALIL_EXAMPLE_OK;
00055 }
```

10.25 motion_complete.cpp File Reference

```
#include "examples.h"
#include <iostream>
#include <string.h>
```

Functions

- int check_interrupts (GCon g, GCStringIn axes)

  Monitors interrupt status on the given axes and returns when interrupts are fired.
- GReturn motion_complete (GCon g)

  Uses interrupts to track when the motion of controller is completed.

10.25.1 Detailed Description

Function calls for the Motion Complete Example Project.
Definition in file motion_complete.cpp.

10.25.2 Function Documentation

10.25.2.1 check_interrupts()

```
int check_interrupts (
            GCon g,
            GCStringIn axes )
```

Monitors interrupt status on the given axes and returns when interrupts are fired.
Definition at line 77 of file motion_complete.cpp.
References e(), G_NO_ERROR, G_SMALL_BUFFER, GCmd(), GInterrupt(), and vector().
Referenced by motion_complete().

10.25.2.2 motion_complete()

```
GReturn motion_complete (
            GCon g )
```

Uses interrupts to track when the motion of controller is completed.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See motion_complete_example.cpp for an example.
Definition at line 18 of file motion_complete.cpp.
References check_interrupts(), e(), G_NO_ERROR, G_SMALL_BUFFER, G_UNSUPPORTED_FUNCTION, GCmd(), GCmdT(), GCommand(), GInterrupt(), GTimeout(), and vector().

10.26 motion_complete.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013 #include <string.h>
00014 using namespace std;
00015
00016 int check_interrupts(GCon g, GCStringIn axes);
00017
00018 GReturn motion_complete(GCon g)
00019 {
00020   char* trimmed;
00021
00022   cout << "\n*************************************************************\n";
00023   cout << "Example GInterrupt() usage\n";
00024   cout << "*************************************************************\n";
00025
00026   char buf[G_SMALL_BUFFER]; //traffic buffer
00027   GSize bytes_read;
00028
00029   //-----------------------------------------------------------------------
00030   //simple check for appropriate communication bus
00031   //EI will fail below if interrupts are not supported at all
00032   bool ei_support = true;
00033   if (GCommand(g, "WH", buf, G_SMALL_BUFFER, &bytes_read) == G_NO_ERROR)
00034   {
00035     //for this example, assume Ethernet connections supports interrupts.
00036     ei_support = (strstr(buf, "IH") != 0);
00037   }//else assume PCI supports
00038
00039   if (!ei_support)
00040   {
00041     cout << "No support on this bus\n";
00042     return G_UNSUPPORTED_FUNCTION;
00043   }
00044
00045   //-----------------------------------------------------------------------
00046   //Flush interrupts
00047   e(GCmd(g, "EI0,0")); //turn off interrupts
00048   GStatus status;
00049   e(GTimeout(g, 0)); //zero timeout
00050
00051   //flush interrupts, status will be zero when queue is empty
00052   while ((GInterrupt(g, &status) == G_NO_ERROR) && status);
00053   e(GTimeout(g, -1)); //restore timeout
00054
00055
00056   //-----------------------------------------------------------------------
00057   // Independent motion
00058   e(GCmd(g, "DP 0,0")); //define position zero on A and B
00059   e(GCmdT(g, "RP", buf, G_SMALL_BUFFER, &trimmed));
00060   cout << "\nPosition: " << trimmed << '\n';
00061   e(GCmd(g, "SP 4000,4000")); //set up speed
00062   e(GCmd(g, "AC 1280000,1280000")); //acceleration
00063   e(GCmd(g, "DC 1280000,1280000")); //deceleration
00064   e(GCmd(g, "PR 8000,10000")); //Postion Relative. B will take longer to make its move.
00065   e(GCmd(g, "SH AB")); //Servo Here
00066   cout << "Beginning independent motion...\n";
00067   e(GCmd(g, "BG AB")); //Begin motion
00068   check_interrupts(g, "AB"); //Block until motion is complete on axes A and B
00069   cout << "Motion Complete on A and B\n";
00070   e(GCmdT(g, "RP", buf, G_SMALL_BUFFER, &trimmed));
00071   cout << "Position: " << trimmed << '\n';
```

```
00072
00073   return  GALIL_EXAMPLE_OK;
00074 }
00075
00077 int check_interrupts(GCon g, GCStringIn axes)
00078 {
00079   char buf[G_SMALL_BUFFER]; //traffic buffer
00080   GReturn rc;
00081   GStatus status;
00082   //bit mask of running axes, axes arg is trusted to provide running axes.
00083   //Low bit indicates running.
00084   unsigned char axis_mask = 0xFF;
00085
00086   int len = strlen(axes);
00087   //iterate through all chars in axes to make the axis mask
00088   for (int i = 0; i < len; i++)
00089   {
00090     //support just A-H
00091     switch (axes[i])
00092     {
00093     case 'A':
00094       axis_mask &= 0xFE;
00095       break;
00096     case 'B':
00097       axis_mask &= 0xFD;
00098       break;
00099     case 'C':
00100       axis_mask &= 0xFB;
00101       break;
00102     case 'D':
00103       axis_mask &= 0xF7;
00104       break;
00105     case 'E':
00106       axis_mask &= 0xEF;
00107       break;
00108     case 'F':
00109       axis_mask &= 0xDF;
00110       break;
00111     case 'G':
00112       axis_mask &= 0xBF;
00113       break;
00114     case 'H':
00115       axis_mask &= 0x7F;
00116       break;
00117     }
00118   }
00119   sprintf(buf, "EI %u", (unsigned char)~axis_mask);
00120   e(GCmd(g, buf)); //send EI axis mask to set up interrupt events.
00121
00122   while (axis_mask != 0xFF) //wait for all interrupts to come in
00123   {
00124     if ((rc = GInterrupt(g, &status)) == G_NO_ERROR)
00125     {
00126       switch (status)
00127       {
00128       case 0xD0: //Axis A complete
00129         axis_mask |= 0x01;
00130         break;
00131       case 0xD1: //Axis B complete
00132         axis_mask |= 0x02;
00133         break;
00134       case 0xD2: //Axis C complete
00135         axis_mask |= 0x04;
00136         break;
00137       case 0xD3: //Axis D complete
00138         axis_mask |= 0x08;
00139         break;
00140       case 0xD4: //Axis E complete
00141         axis_mask |= 0x10;
00142         break;
00143       case 0xD5: //Axis F complete
00144         axis_mask |= 0x20;
00145         break;
00146       case 0xD6: //Axis G complete
00147         axis_mask |= 0x40;
00148         break;
00149       case 0xD7: //Axis H complete
00150         axis_mask |= 0x80;
00151         break;
00152       }
00153     }
00154   }
00155
00156   return rc;
00157 }
```

10.27 motion_complete_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

10.27.1 Detailed Description

See motion_complete() for implementation of logic
Definition in file motion_complete_example.cpp.

10.27.2 Function Documentation

10.27.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.
Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.
contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.
message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

10.28 motion_complete_example.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013 using namespace std;
00014
00016
00020 int main(int argc, char * argv[])
00021 {
00022   GReturn rc = GALIL_EXAMPLE_OK;
00023   char buf[G_SMALL_BUFFER];
00024
00025   //var used to refer to a unique connection. A valid connection is nonzero.
00026   GCon g = 0;
00027   try
00028   {
00029     if (argc != 2) //Invalid number of arguments
00030     {
00031       cerr << "Incorrect number of arguments provided\n";
00032       cerr << "Usage: motion_complete_example.exe <ADDRESS>\n";
00033       pause();
00034       return GALIL_EXAMPLE_ERROR;
00035     }
00036
00037     char* address = argv[1];  //Retrieve address from command line
00038     sprintf(buf, "%s --subscribe EI", address);  //Subscribe to event interrupts
00039     e(GOpen(buf, &g)); //Opens a connection at the provided address
00040
00041     //Uses interrupts to track when the motion of controller is completed
00042     rc = motion_complete(g);
00043   }
00044   catch (GReturn gr)
00045   {
00046     error(g, gr);
00047     pause();
00048     return GALIL_EXAMPLE_ERROR;
00049   }
00050
00051     pause();
00052   return GALIL_EXAMPLE_OK;
00053 }
```

10.29 position_tracking.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- GReturn position_tracking (GCon g, int speed=5000)

  Puts controller into Position Tracking Mode and accepts user-entered positions.

10.29.1 Detailed Description

Function calls for the Position Tracking Example Project.
Definition in file position_tracking.cpp.

10.29.2 Function Documentation

10.29.2.1 position_tracking()

```
GReturn position_tracking (
            GCon g,
            int speed = 5000 )
```

Puts controller into Position Tracking Mode and accepts user-entered positions.

Parameters

g	Connection's handle.
speed	Optional speed of the controller in Position Tracking Mode. Default value of 5000.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See position_tracking_example.cpp for an example.
Definition at line 15 of file position_tracking.cpp.
References e(), G_CONNECTION_NOT_ESTABLISHED, G_SMALL_BUFFER, GCmd(), GMotionComplete(), and vector().

10.30 position_tracking.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013 using namespace std;
00014
00015 GReturn position_tracking(GCon g, int speed)
00016 {
00017   char buf[G_SMALL_BUFFER]; //traffic buffer
00018   int acc = 100 * speed; //set acceleration/deceleration to 100 times speed
00019
00020   if (g == 0) //Bad connection
00021   {
00022     cerr « "There was an error with the connection." « endl;
00023     return G_CONNECTION_NOT_ESTABLISHED;
00024   }
00025
00026   e(GCmd(g, "STA"));    // stop motor
00027   e(GMotionComplete(g, "A")); //Wait for motion to complete
00028   e(GCmd(g, "SHA"));   // Set servo here
00029   e(GCmd(g, "DPA=0")); // Start position at absolute zero
00030   e(GCmd(g, "PTA=1")); // Start position tracking mode on A axis
00031
00032   sprintf(buf, "SPA=%d", speed);
00033   e(GCmd(g, buf)); // Set speed
00034
00035   sprintf(buf, "ACA=%d", acc);
00036   e(GCmd(g, buf)); // Set acceleration
00037
00038   sprintf(buf, "DCA=%d", acc);
00039   e(GCmd(g, buf)); // Set deceleration
00040
00041   cout « "Begin Position Tracking with speed " « speed;
00042   cout « ". Enter a non-number to exit.\n";
00043   int position;
00044
00045   //Loop asking user for new position.  End loop when user enters a non-number
00046   while (1)
00047   {
00048     cout « "Enter a new position:\n";
00049     cin » position;
00050
00051     if (cin.fail()) //A non-number was entered
00052     {
00053       cout « "Position Tracking has exited\n";
00054       break;
00055     }
00056     else
00057     {
00058       sprintf(buf, "PAA=%d", position);
00059       e(GCmd(g, buf)); // Go to new position
00060     }
00061   }
00062
00063   e(GCmd(g, "STA")); //stop motor
00064   e(GMotionComplete(g, "A")); //Wait for motion to complete
00065
00066   return  GALIL_EXAMPLE_OK;
00067 }
```

10.31 position_tracking_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

10.31.1 Detailed Description

See position_tracking() for implementation of logic
Definition in file position_tracking_example.cpp.

10.31.2 Function Documentation

10.31.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.
Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.
contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.
message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

10.32 position_tracking_example.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013 using namespace std;
00014
00016
00021 int main(int argc, char * argv[])
00022 {
00023   GReturn rc = GALIL_EXAMPLE_OK;
00024   char buf[G_SMALL_BUFFER];
00025
00026   //var used to refer to a unique connection. A valid connection is nonzero.
00027   GCon g = 0;
00028
00029   try
00030   {
00031     if (argc < 2 || argc > 3) //Invalid number of arguments
00032     {
00033       cerr << "Incorrect number of arguments provided\n";
00034       cerr << "Usage: position_tracking_example.exe <ADDRESS> <SPEED=5000>\n";
00035            pause();
00036       return GALIL_EXAMPLE_ERROR;
00037     }
00038
00039     char* address = argv[1];  //Retrieve address from command line
00040     e(GOpen(address, &g)); //Opens a connection at the provided address
00041
00042     if (argc == 3) //Position tracking with custom speed
00043     {
00044       //Retrieve speed from command line and convert to int
00045       char* end;
00046       int speed = strtol(argv[2], &end, 10);
00047       //If this character is not a null character,
00048       //the user did not enter a valid integer for speed
00049       if (*end != '\0')
00050       {
00051         cerr << "An invalid speed was entered.  "
00052         "Please enter a valid integer for speed.\n"
00053         "Usage: position_tracking_example.exe <ADDRESS> <SPEED=5000>\n";
00054         pause();
00055         return GALIL_EXAMPLE_ERROR;
00056       }
00057
00058       //Puts controller into Position Tracking Mode and accepts user-entered positions
00059       rc = position_tracking(g, speed);
00060     }
00061     else if (argc == 2) //Position tracking with default speed
00062     {
00063       //Puts controller into Position Tracking Mode and accepts user - entered positions
00064       rc = position_tracking(g);
00065     }
00066   }
00067   catch (GReturn gr)
00068   {
00069     error(g, gr);
00070     pause();
00071     return GALIL_EXAMPLE_ERROR;
00072   }
00073
00074     pause();
00075   return GALIL_EXAMPLE_OK;
00076 }
```

10.33 record_position.cpp File Reference

```
#include "examples.h"
#include <iostream>
#include <fstream>
```

Macros

- #define G_LASTINDEX 999

Functions

- void write_array_to_file (GCon g, ofstream &os, const char *array_name, int previous_rd, int rd)

  Grabs data from array on controller and writes it to the given text file.
- GReturn record_position (GCon g, char *fileA, char *fileB)

  Record user's training and saves to a text file.

10.33.1 Detailed Description

Function calls for the Record Position Example project
Definition in file record_position.cpp.

10.33.2 Macro Definition Documentation

10.33.2.1 G_LASTINDEX

```
#define G_LASTINDEX 999
```

Definition at line 13 of file record_position.cpp.

10.33.3 Function Documentation

10.33.3.1 record_position()

```
GReturn record_position (
            GCon g,
            char * fileA,
            char * fileB )
```

Record user's training and saves to a text file.

Parameters

g	Connection's handle.
fileA	A Path to a text file where training for Axis A will be recorded.
fileB	A Path to a text file where training for Axis B will be recorded.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See record_position_example.cpp for an example.
Definition at line 20 of file record_position.cpp.
References e(), GCmd(), GCmdI(), GProgramDownload(), GSleep(), vector(), and write_array_to_file().
Referenced by contour().

10.33.3.2 write_array_to_file()

```
void write_array_to_file (
            GCon g,
            ofstream & os,
            const char * array_name,
            int previous_rd,
            int rd )
```

Grabs data from array on controller and writes it to the given text file.
Definition at line 67 of file record_position.cpp.
References e(), G_COMMA, G_HUGE_BUFFER, GArrayUpload(), and vector().
Referenced by record_position().

10.34 record_position.cpp

Go to the documentation of this file.

```
00001
00008 #include "examples.h"
00009
00010 #include <iostream> //std::cout
00011 #include <fstream>
00012
00013 #define G_LASTINDEX 999
00014
00015 using namespace std;
00016
00017 void write_array_to_file(GCon g, ofstream& os, const char* array_name,
00018              int previous_rd, int rd);
00019
00020 GReturn record_position(GCon g, char* fileA, char* fileB)
00021 {
00022   ofstream osA, osB;
00023   osA.open(fileA, ios::out | ios::trunc);
00024   osB.open(fileB, ios::out | ios::trunc);
00025
00026   int recording = 1;
00027
00028   e(GProgramDownload(g,
00029   "RC 0;' Disable Recording\n"
00030   "DP 0, 0;' Set current position to 0\r"
00031   "DM posA[1000], posB[1000];' Define a new array that will hold positional data\r"
00032   "RA posA[], posB[];' Sets position array to be where recorded data will be stored\r"
00033   "RD _TPA, _TPB;' Defines Position to be the type of data that will be recorded\r"
00034   "RC 1,-1000;' Begins recording at 512Hz in continuous mode\r"
00035   "MO AB;' Turns motors off\r"
00036   "AI -1;' Waits for active low on Input 1\r"
00037   "RC 0;' Disable Recording after Input 1 goes low\r"
00038   "EN;' End program", 0));
00039   e(GCmd(g, "XQ")); //Begins execution of program
00040
00041   int rd = 0;
00042   int previous_rd = 0;
00043
00044   do
00045   {
00046     GSleep(1000); //Sleep while we wait for roughly half the array to be written
00047     e(GCmdI(g, "MG _RD", &rd)); //Gets address of next value in the position array
00048
00049     //Get values from posA[] array and write to file
00050     write_array_to_file(g, osA, "posA[]", previous_rd, rd);
00051
00052     //Get values from posB[] array and write to file
00053     write_array_to_file(g, osB, "posB[]", previous_rd, rd);
00054
00055     e(GCmdI(g, "MG _RC", &recording)); // Check status of RC
00056
00057     previous_rd = rd;
00058   } while (recording); //While recording is active
00059
00060   osA.close();
00061   osB.close();
00062
00063   return GALIL_EXAMPLE_OK;
00064 }
00065
00067 void write_array_to_file(GCon g, ofstream& os, const char* array_name,
00068              int previous_rd, int rd)
00069 {
00070   char buf[G_HUGE_BUFFER]; //traffic buffer
00071
00072   if (previous_rd < rd) // No array wrap around
00073   {
00074     e(GArrayUpload(g, array_name, previous_rd, rd - 1, G_COMMA, buf, G_HUGE_BUFFER));
00075     os « ',' « buf;
00076   }
00077   else // Array wrapped around - grab two separate chunks from the array
00078   {
00079     e(GArrayUpload(g, array_name, previous_rd, G_LASTINDEX, G_COMMA, buf, G_HUGE_BUFFER));
00080     os « ',' « buf;
00081
00082     if (rd != 0)
00083     {
00084       e(GArrayUpload(g, array_name, 0, rd - 1, G_COMMA, buf, G_HUGE_BUFFER));
00085       os « ',' « buf;
00086     }
00087   }
00088 }
```

10.35 record_position_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

10.35.1 Detailed Description

See record_position() for implementation of logic
Definition in file record_position_example.cpp.

10.35.2 Function Documentation

10.35.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.
Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.
contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.
message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

10.36 record_position_example.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013 using namespace std;
00014
00016
00021 int main(int argc, char * argv[])
00022 {
00023   GReturn rc = GALIL_EXAMPLE_OK;
00024   char buf[G_SMALL_BUFFER];
00025
00026   //var used to refer to a unique connection. A valid connection is nonzero.
00027   GCon g = 0;
00028
00029   try
00030   {
00031     if (argc != 4) //Invalid number of arguments
00032     {
00033       cerr << "Incorrect number of arguments provided\n";
00034       cerr << "Usage: record_position_example.exe <ADDRESS> <File A> <File B>\n";
00035       pause();
00036       return GALIL_EXAMPLE_ERROR;
00037     }
00038
00039     char* fileA = argv[2]; //Retrieve filepath from command line
00040     char* fileB = argv[3]; //Retrieve filepath from command line
00041     char* address = argv[1];  //Retrieve address from command line
00042     e(GOpen(address, &g)); //Opens a connection at the provided address
00043
00044     //Record user's training and saves to a text file
00045     rc = record_position(g, fileA, fileB);
00046   }
00047   catch (GReturn gr)
00048   {
00049     error(g, gr);
00050     pause();
00051     return GALIL_EXAMPLE_ERROR;
00052   }
00053
00054   pause();
00055   return GALIL_EXAMPLE_OK;
00056 }
```

10.37 remote_client.cpp File Reference

```
#include "examples.h"
#include <iostream>
#include <vector>
#include <string>
#include <conio.h>
```

Functions

- void print_client_message (const char *message)
- void print_servers_list (const std::vector< std::string > &server_list)
- void servers_to_list (std::vector< std::string > &server_list, std::string servers)
- GReturn remote_client ()

  Lists available remote servers and allows connection to remote server.

10.37.1 Detailed Description

Function calls for the Remote Client Example Project.
Definition in file remote_client.cpp.

10.37.2 Function Documentation

10.37.2.1 print_client_message()

```
void print_client_message (
            const char * message )
```

Definition at line 24 of file remote_client.cpp.

10.37.2.2 print_servers_list()

```
void print_servers_list (
            const std::vector< std::string > & server_list )
```

Definition at line 34 of file remote_client.cpp.

10.37.2.3 remote_client()

```
GReturn remote_client ( )
```

Lists available remote servers and allows connection to remote server.

Returns

The success status or error code of the function.See gclib_errors.h for possible values.

See remote_client_example for an example.

Key	Usage
q	Quit
s	List available servers on then network
h	List available hardware on the current server
0-9	Connect to server instance by number
l	Connect back to local server

Definition at line 89 of file remote_client.cpp.
References e(), G_SMALL_BUFFER, GAddresses(), GListServers(), GSetServer(), and vector().

10.37.2.4 servers_to_list()

```
void servers_to_list (
            std::vector< std::string > & server_list,
            std::string servers )
```

Definition at line 54 of file remote_client.cpp.

10.38 remote_client.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013 #include <vector>
00014 #include <string>
00015
00016 #ifdef _WIN32
00017 #include <conio.h>
00018 #elif __linux__
00019 #include <ncurses.h>
00020 #endif
00021
00022 using namespace std;
00023
00024 void print_client_message(const char* message)
00025 {
00026 #ifdef _WIN32
00027   std::cout << message << std::endl;
00028 #elif __linux__
00029   printw(message);
```

```
00030   printw("\n");
00031 #endif
00032 }
00033
00034 void print_servers_list(const std::vector<std::string>& server_list)
00035 {
00036   char buf[G_SMALL_BUFFER];
00037
00038
00039   if (server_list.size() == 0)
00040   {
00041     print_client_message("none");
00042   }
00043   else
00044   {
00045     for (int i = 0; i < server_list.size(); i++)
00046     {
00047       std::string test = server_list[i];
00048       sprintf(buf, "<%d> %s", i, test.c_str());
00049       print_client_message(buf);
00050     }
00051   }
00052 }
00053
00054 void servers_to_list(std::vector<std::string>& server_list, std::string servers)
00055 {
00056   server_list.clear();
00057
00058   if (servers.length() == 0)
00059     return;
00060
00061   int index = 0;
00062   std::string server;
00063   while (index < servers.length())
00064   {
00065     if (servers[index] == '\n')
00066     {
00067       server_list.push_back(server);
00068       server.clear();
00069     }
00070     else
00071     {
00072       server += servers[index];
00073     }
00074     index++;
00075   }
00076
00077   server_list.push_back(server);
00078 }
00079
00089 GReturn remote_client()
00090 {
00091   bool loop = true;
00092   char servers[G_SMALL_BUFFER];
00093   char buf[G_SMALL_BUFFER];
00094   std::vector<std::string> server_list;
00095
00096   char instructions[] = "<s> List available servers on the network\n"
00097     "<h> List available hardware on currently connected server\n"
00098     "<0-9> Enter numbers 0-9 to connect to a server by index\n"
00099     "<l> Set active server back to local server\n"
00100     "<q> Quit\n";
00101
00102 #ifdef _WIN32
00103   cout « instructions « std::endl;
00104 #elif __linux__
00105   //These functions set up the ncurses library to capture keyboard input
00106   initscr(); // Initialization of ncurses library
00107   cbreak();  // Capture one character at a time
00108   noecho();  // Do not write back entered characters to the console
00109   printw(instructions); //Print instructions to console
00110 #endif
00111
00112   while (loop)
00113   {
00114 #ifdef _WIN32
00115     char input = _getch(); //Capture keypress
00116 #elif __linux__
00117     char input = getch(); //Capture keypress
00118 #endif
00119     if(input == 'q')
00120       loop = false;
00121     else if (input == 's')
00122     {
00123       print_client_message("Available Servers:");
00124       e(GListServers(servers, G_SMALL_BUFFER));
00125
```

```
00126       servers_to_list(server_list, servers);
00127
00128       print_servers_list(server_list);
00129     }
00130     else if(input >= '0' && input <= '9')
00131     {
00132       int index = input - '0';
00133       if (server_list.size() > 0 && index < server_list.size())
00134       {
00135         e(GSetServer(server_list[index].c_str()));
00136         sprintf(buf, "Server set to: %s", server_list[index].c_str());
00137         print_client_message(buf);
00138       }
00139     }
00140     else if (input == 'l')
00141     {
00142       e(GSetServer("Local"));
00143       print_client_message("Server set to: Local");
00144     }
00145     else if (input == 'h')
00146     {
00147       e(GAddresses(buf, G_SMALL_BUFFER));
00148       print_client_message(buf);
00149     }
00150   }
00151
00152 #if __linux__
00153   endwin(); //Restores terminal to previous state
00154 #endif
00155
00156   return GALIL_EXAMPLE_OK;
00157 }
```

10.39 remote_client_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

10.39.1 Detailed Description

See remote_client() for implementation of logic
Definition in file remote_client_example.cpp.

10.39.2 Function Documentation

10.39.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.
Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.

contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.
message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

10.40 remote_client_example.cpp

Go to the documentation of this file.

```
00001
00009 #include "examples.h"
00010
00011 #include <iostream> //std::cout
00012
00013 using namespace std;
00014
00016
00019 int main(int argc, char * argv[])
00020 {
00021   char buf[G_SMALL_BUFFER];
00022   int rc = GALIL_EXAMPLE_OK;
00023
00024   try
00025   {
00026     //Demonstrates various uses of GListServers() and GSetServer().
00027     remote_client();
00028   }
00029   catch (GReturn gr)
00030   {
00031     error(nullptr, gr); //see examples.h for error handling
00032     pause();
00033     return GALIL_EXAMPLE_ERROR;
00034   }
00035
00036     pause();
00037   return GALIL_EXAMPLE_OK;
00038 }
```

10.41 remote_server.cpp File Reference

```
#include "examples.h"
#include <iostream>
#include <conio.h>
```

Functions

- void print_server_message (const char *message)
- GReturn remote_server (const char *server_name)

  Publishes local gcaps server to the network.

10.41.1 Detailed Description

Function calls for the Remote Server Example Project.
Definition in file remote_server.cpp.

10.41.2 Function Documentation

10.41.2.1 print_server_message()

```
void print_server_message (
            const char * message )
```

Definition at line 22 of file remote_server.cpp.

10.41.2.2 remote_server()

```
GReturn remote_server (
            const char * server_name )
```

Publishes local gcaps server to the network.

Parameters

Name	to publish server under.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See remote_server_example for an example.

Key	Usage
q	Quit
p	Publish this server to the network
r	Remove this server from the network

Definition at line 39 of file remote_server.cpp.
References e(), G_SMALL_BUFFER, GPublishServer(), and vector().

10.42 remote_server.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013
00014 #ifdef _WIN32
00015   #include <conio.h>
00016 #elif __linux__
00017   #include <ncurses.h>
00018 #endif
00019
00020 using namespace std;
00021
00022 void print_server_message(const char* message)
00023 {
00024 #ifdef _WIN32
00025   std::cout « message « std::endl;
00026 #elif __linux__
00027   printw(message);
00028   printw("\n");
00029 #endif
00030 }
00031
00039 GReturn remote_server(const char* server_name)
00040 {
00041   bool loop = true;
00042   char connections[G_SMALL_BUFFER];
```

```
00043   char buf[G_SMALL_BUFFER];
00044
00045   char instructions[] = "<p> Publish this server to the network\n"
00046               "<r> Remove this server from the network\n"
00047               "<q> Quit\n";
00048
00049 #ifdef _WIN32
00050   cout « instructions « std::endl;
00051 #elif __linux__
00052   //These functions set up the ncurses library to capture keyboard input
00053   initscr(); // Initialization of ncurses library
00054   cbreak();  // Capture one character at a time
00055   noecho();  // Do not write back entered characters to the console
00056   printw(instructions); //Print instructions to console
00057 #endif
00058
00059   while (loop)
00060   {
00061 #ifdef _WIN32
00062     switch (_getch()) //Capture keypress
00063 #elif __linux__
00064     switch (getch()) //Capture keypress
00065 #endif
00066     {
00067     case 'q': //Quit
00068       loop = false;
00069       break;
00070     case 'p': //Publish this server to the network
00071       e(GPublishServer(server_name, true, false));
00072       print_server_message("Published Server");
00073       break;
00074     case 'r': //Remove this server from the network
00075       e(GPublishServer(server_name, false, false));
00076       print_server_message("Removed Server");
00077       break;
00078     }
00079   }
00080 #if __linux__
00081   endwin(); //Restores terminal to previous state
00082 #endif
00083
00084   return GALIL_EXAMPLE_OK;
00085 }
```

10.43 remote_server_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
#include <string>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

10.43.1 Detailed Description

See remote_server() for implementation of logic
Definition in file remote_server_example.cpp.

10.43.2 Function Documentation

10.43.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.

Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.
contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.
message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

10.44 remote_server_example.cpp

Go to the documentation of this file.

```
00001
00009 #include "examples.h"
00010
00011 #include <iostream> //std::cout
00012 #include <string> //std::getline
00013
00014 using namespace std;
00015
00017
00021 int main(int argc, char * argv[])
00022 {
00023   char buf[G_SMALL_BUFFER];
00024   int rc = GALIL_EXAMPLE_OK;
00025
00026   try
00027   {
00028     std::string server_name;
00029
00030     if (argc != 2) //Invalid number of arguments
00031     {
00032       std::cout << "Enter name of your server: ";
00033       std::getline(std::cin, server_name);
00034     }
00035     else
00036     {
00037       server_name = argv[1];  //Retrieve address from command line
00038     }
00039
00040     //Demonstrates various uses of GPublishServer().
00041     remote_server(server_name.c_str());
00042   }
00043   catch (GReturn gr)
00044   {
00045     error(nullptr, gr);
00046     pause();
00047     return GALIL_EXAMPLE_ERROR;
00048   }
00049
00050     pause();
```

```
00051   return GALIL_EXAMPLE_OK;
00052 }
```

10.45 vector.cpp File Reference

```
#include "examples.h"
#include <iostream>
#include <string>
#include <fstream>
```

Functions

- bool load_buffer (GCon g, ifstream &fs, int capacity)
- GReturn vector (GCon g, char *file)

  Puts controller into Vector Mode and accepts a file defining vector points.

10.45.1 Detailed Description

Function calls the Vector Mode Example Project.
Definition in file vector.cpp.

10.45.2 Function Documentation

10.45.2.1 load_buffer()

```
bool load_buffer (
            GCon g,
            ifstream & fs,
            int capacity )
```

Loads vector buffer with commands from the given text file.
Returns false when there are no more lines in the text file
Definition at line 88 of file vector.cpp.
References e(), GCmd(), and vector().
Referenced by vector().

10.45.2.2 vector()

```
GReturn vector (
            GCon g,
            char * file )
```

Puts controller into Vector Mode and accepts a file defining vector points.

Parameters

g	Connection's handle.
file	A Path to a file that defines vector commands.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See vector_example.cpp for an example.
Example text file:

```
VP -2219,-2667
VP -2523,-2832
VP 2844,-1425
VP 728,1971
VP 2127,183
VP -997,688
VP 725,-1893
```

```
VP 527,2899
VP -37,2523
VP 1277,1425
VP 857,2388
VP 1096,-1694
CR 1000,0,90
```

Definition at line 36 of file vector.cpp.
References e(), G_BAD_FILE, G_CONNECTION_NOT_ESTABLISHED, GCmd(), GCmdI(), GMotionComplete(), GSleep(), load_buffer(), and vector().
Referenced by py::__init__(), AddressConvert(), check_interrupts(), Examples::Commands(), Examples::Contour(), contour(), csv_to_vector(), e(), error(), GclibJava::finalize(), gclib::GAddresses(), GAddresses(), py::GAddresses(), py::GArrayDownload(), gclib::GArrayDownload(), GclibJava::GArrayDownload(), GArrayDownloadFile(), py::GArrayDownloadFile(), GclibJava::GArrayDownloadFile(), gclib::GArrayDownloadFile(), py::GArrayUpload(), gclib::GArrayUpload(), GclibJava::GArrayUpload(), GArrayUploadFile(), py::GArrayUploadFile(), GclibJava::GArrayUploadFile(), GclibJava::GArrayUploadFile(), gclib::GArrayUploadFile(), GAssign(), py::GAssign(), gclib::GAssign(), GclibJava::GAssign(), gclib::gclib(), py::GClose(), GCmd(), GCmdD(), gclib::GCmdD(), GCmdI(), gclib::GCmdI(), GCmdT(), py::GCommand(), GclibJava::GCommand(), gclib::GCommand(), GError(), py::GFirmwareDownload(), gclib::GFirmwareDownload(), GclibJava::GFirmwareDownload(), gclib::GInfo(), GInfo(), py::GInfo(), gclib::GInterrupt(), GclibJava::GInterrupt(), py::GInterrupt(), gclib::GIpRequests(), GIpRequests(), py::GIpRequests(), gclib::GListServers(), GListServers(), gclib::GMessage(), py::GMessage(), GMotionComplete(), py::GMotionComplete(), gclib::GMotionComplete(), py::GOpen(), gclib::GOpen(), GclibJava::GOpen(), py::GProgramDownload(), GclibJava::GProgramDownload(), GclibJava::GProgramDownload(), gclib::GProgramDownload(), GProgramDownloadFile(), py::GProgramDownloadFile(), gclib::GProgramDownloadFile(), GclibJava::GProgramDownloadFile(), GclibJava::GProgramDownloadFile(), gclib::GProgramUpload(), py::GProgramUpload(), GProgramUploadFile(), py::GProgramUploadFile(), gclib::GProgramUploadFile(), GclibJava::GProgramUploadFile(), GPublishServer(), gclib::GPublishServer(), GclibJava::GPublishServer(), gclib::GRead(), gclib::GRecord< T >(), gclib::GRecordRate(), GRecordRate(), gclib::GRemoteConnections(), GRemoteConnections(), gclib::GServerStatus(), GServerStatus(), GSetServer(), gclib::GSetServer(), GclibJava::GSetServer(), GSetupDownloadFile(), py::GSetupDownloadFile(), gclib::GSetupDownloadFile(), py::GSleep(), py::GTimeout(), gclib::GVersion(), GVersion(), py::GVersion(), GWaitForBool(), gclib::GWrite(), H_AddArray(), H_ArrayAddElement(), H_ArrayDownloadFromMemory(), H_CreateArrayNode(), H_DownloadArraysFromList(), H_DownloadData(), H_FindSector(), H_FreeArrays(), H_InitArrayNode(), H_UploadArrayToList(), H_WriteArrayCsv(), ip_assigner(), Examples::IP_Assigner(), Examples::Jog(), jog(), load_buf(), load_buffer(), Remote_Client_Example::Main(), Commands_Example::Main(), Contour_Example::Main(), IP_Assigner_Example::Main(), Jog_Example::Main(), Message_Example::Main(), Motion_Complete_Example::Main(), Position_Tracking_Example::Main(), Record_Position_Example::Main(), Remote_Server_Example::Main(), Vector_Mode_Example::Main(), Examples::Message(), message(), Examples::Motion_Complete(), motion_complete(), Examples::Position_Tracking(), position_tracking(), Examples::PrintError(), Examples::Record_Position(), record_position(), remote_client(), Examples::Remote_Client(), remote_server(), Examples::Remote_Server(), string_split(), vector(), Examples::Vector_Mode(), write_array_to_file(), x_arrays(), x_dr_motioncomplete(), x_e(), x_ei_motioncomplete(), x_gcommand(), x_ginterrupt(), x_gmessage(), x_gmotioncomplete(), x_gread_gwrite(), x_grecord(), x_nonblocking(), and x_programs().

10.46 vector.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011 #include <iostream> //std::cout
00012 #include <string> //to_string, string, etc.
00013 #include <fstream>
00014 using namespace std;
00015
00016 bool load_buffer(GCon g, ifstream& fs, int capacity);
00017
00036 GReturn vector(GCon g, char* file)
00037 {
00038   if (g == 0) //Bad connection
00039   {
00040     cerr « "There was an error with the connection." « endl;
00041     return G_CONNECTION_NOT_ESTABLISHED;
00042   }
00043
00044   e(GCmd(g, "ST"));     // stop all motors
00045   e(GCmd(g, "SH AB"));  // Set servo here
00046   e(GCmd(g, "DP 0,0")); // Start position at absolute zero
00047
00048   e(GCmd(g, "CAS"));     //Defines S as active coordinate system
00049   e(GCmd(g, "VS 20000"));  //Defines vector speed
```

```
00050   e(GCmd(g, "VA 200000")); //Defines vector acceleration
00051   e(GCmd(g, "VD 200000")); //Defines vector deceleration
00052   e(GCmd(g, "VM AB")); //Begin Vector Segment
00053
00054   ifstream fs;
00055   fs.open(file);  //Open user defined text file
00056   if (!fs) //If file could not be opened
00057   {
00058     cout « "Unable to open file\n";
00059     return G_BAD_FILE;
00060   }
00061
00062   int capacity;
00063   //Stores the available space of the vector buffer in the capacity variable
00064   e(GCmdI(g, "MG _LMS", &capacity));
00065   load_buffer(g, fs, capacity);
00066
00067   e(GCmd(g, "BG S")); //Begin Motion
00068
00069   do //Load buffer with more commands
00070   {
00071     GSleep(100); //Sleep a bit while buffer is emptying
00072
00073     //Stores the available space of the vector buffer in the capacity variable
00074     e(GCmdI(g, "MG _LMS", &capacity));
00075   } while (load_buffer(g, fs, capacity));
00076
00077   fs.close(); //Close the file
00078
00079   e(GCmd(g, "VE")); //Segment End
00080   e(GMotionComplete(g, "S"));
00081
00082   return  GALIL_EXAMPLE_OK;
00083 }
00084
00088 bool load_buffer(GCon g, ifstream& fs, int capacity)
00089 {
00090   string s_cmd;
00091   // Fully load the vector buffer leaving room for one VE command
00092   for (capacity; capacity > 1; capacity--)
00093   {
00094     if (getline(fs, s_cmd)) //if there is another line of the text file
00095     {
00096       //Run the command on each line of the text file
00097       e(GCmd(g, s_cmd.c_str()));
00098     }
00099     else
00100       return false;
00101   }
00102
00103   return true;
00104 }
```

10.47 vector_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

10.47.1 Detailed Description

See vector() for implementation of logic
Definition in file vector_example.cpp.

10.47.2 Function Documentation

10.47.2.1 main()

```
int main (
            int argc,
```

```
            char * argv[] )
```

Main function for Commands Example.
Main function for Vector Mode Example.
Main function for Remote Server Example.
Main function for Record Position Example.
Main function for Position Tracking Example.
Main Function for Motion Complete Example.
Main function for Message Example.
Main function for Jog Example.
Main function for IP Assigner Example.
Main function for Contour Example.
commands_example.cpp takes one arguments at the command line: an IP Address to a Galil controllers.
contour_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
ip_assigner_example.cpp takes two arguments at the command line: a Serial Number of a Galil controller and 1 byte address.
jog_example.cpp takes one argument at the command line: an IP Address to a Galil controller. When the program is run the controller will be at rest. Press a key at the console to adjust the speed of the controller.
message_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
motion_complete_example.cpp takes one argument at the command line: an IP Address to a Galil controller.
position_tracking_example.cpp takes up to two arguments at the command line: an IP Address to a Galil controller and an optional speed value. If only one argument is provided the program will default to a speed value of 5000.
record_position_example.cpp takes three arguments at the command line: an IP Address to a Galil controller and a two text files to hold the positional data for two axes.
remote_client_example.cpp takes no arguments at the command line.
remote_server_example.cpp takes one argument at the command line: the name you wish to publish your server under.
vector_example.cpp takes two arguments at the command line: an IP Address to a Galil controller and a path to a text file defining vector points. When the program is run the controller will be put into vector mode and loaded with the points defined in the text file. The controller will run until it reaches all points defined in the text file.
Definition at line 18 of file commands_example.cpp.
References commands(), e(), error(), G_SMALL_BUFFER, GOpen(), main(), and pause().
Referenced by main().

10.48 vector_example.cpp

Go to the documentation of this file.

```
00001
00010 #include "examples.h"
00011
00012 #include <iostream> //std::cout
00013 using namespace std;
00014
00016
00022 int main(int argc, char * argv[])
00023 {
00024   GReturn rc = GALIL_EXAMPLE_OK;
00025   char buf[G_SMALL_BUFFER];
00026
00027   //var used to refer to a unique connection. A valid connection is nonzero.
00028   GCon g = 0;
00029
00030   try
00031   {
00032     if (argc != 3)
00033     {
00034       cerr << "Incorrect number of arguments provided\n";
00035       cerr << "Usage: Vector_Example.exe <ADDRESS> <FILE>\n";
00036       pause();
00037       return GALIL_EXAMPLE_ERROR;
00038     }
00039
00040     char* file = argv[2]; //Retrieve file from command line
00041     char* address = argv[1];  //Retrieve address from command line
00042     e(GOpen(address, &g)); //Opens a connection at the provided address
00043
00044     // Puts controller into Vector Mode and accepts a file defining vector points
00045     rc = vector(g, file);
```

```
00046   }
00047   catch (GReturn gr)
00048   {
00049     error(g, gr); //see examples.h for error handling
00050     pause();
00051     return GALIL_EXAMPLE_ERROR;
00052   }
00053
00054     pause();
00055   return GALIL_EXAMPLE_OK;
00056 }
```

10.49 Form1.cs File Reference

Data Structures

- class MainForm

  Demonstrates using gclib in a Windows Form, including using a second thread to free the GUI.

10.49.1 Detailed Description

Function calls for the C# examples.

For VB.NET, see definition in file Form1.vb
Definition in file Form1.cs.

10.50 Form1.cs

Go to the documentation of this file.

```
00001
00007 using System;
00008 using System.Collections.Generic;
00009 using System.ComponentModel;
00010 using System.Data;
00011 using System.Drawing;
00012 using System.Linq;
00013 using System.Text;
00014 using System.Threading.Tasks;
00015 using System.Windows.Forms;
00016
00017 namespace gclib_example
00018 {
00022     public partial class MainForm : Form
00023     {
00024
00025         #region "UI"
00026
00027         //form's ctor
00028         public MainForm()
00029         {
00030             InitializeComponent();
00031         }
00032
00033         //Runs when form loads
00034         private void MainForm_Load(object sender, EventArgs e)
00035         {
00036             PrintOutput("Enter a FULL GOpen() address above and click Go", PrintStyle.Instruction);
00037             PrintOutput("NOTE: This demo will attempt to move Axis A", PrintStyle.Instruction);
00038         }
00039
00040         // Opens Galil's help to show GOpen() options
00041         private void HelpLabel_Click(object sender, EventArgs e)
00042         {
00043             //link to GOpen() documentation.
00044
      System.Diagnostics.Process.Start("http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h_aef4aec8a85630eed029b7a46aea7c
00045         }
00046
00047         //Runs when user clicks Go button
00048         private void GoButton_Click(object sender, EventArgs e)
00049         {
00050             if (AddressTextBox.Text.Length == 0)
00051             {
00052                 PrintOutput("Enter a FULL GOpen() address above and click Go",
      PrintStyle.Instruction);
00053                 return;
```

```
00054             }
00055             RunDemo(AddressTextBox.Text);
00056         }
00057 
00058         //Various print styles.
00059         private enum PrintStyle
00060         {
00061             Instruction,
00062             Normal,
00063             GalilData,
00064             GclibData,
00065             Err,
00066         }
00067 
00074         private void PrintOutput(string Message, PrintStyle Style = PrintStyle.Normal, bool
      SuppressCrLf = false)
00075         {
00076             if (Output.InvokeRequired)
00077             {
00078                 Output.Invoke(new Printer(PrintOutput), new object[] { Message, Style, SuppressCrLf
      });
00079             }
00080             else
00081             {
00082                 Color color;
00083 
00084                 switch (Style)
00085                 {
00086                     case PrintStyle.Instruction:
00087                         color = Color.Black;
00088                         break;
00089                     case PrintStyle.GalilData:
00090                         color = Color.Green;
00091                         break;
00092                     case PrintStyle.Normal:
00093                         color = Color.Blue;
00094                         break;
00095                     case PrintStyle.Err:
00096                         color = Color.Red;
00097                         break;
00098                     case PrintStyle.GclibData:
00099                         color = Color.Magenta;
00100                         break;
00101                     default:
00102                         color = Color.Blue;
00103                         break;
00104                 }//switch
00105 
00106                 Output.SelectionStart = Output.Text.Length;
00107                 Output.SelectionColor = color;
00108                 Output.AppendText(Message);
00109 
00110                 if (!SuppressCrLf)
00111                     Output.AppendText("\r\n");
00112 
00113             }//invoke check
00114         }
00115 
00116         #endregion
00117 
00118         #region "Threading"
00119 
00126         private delegate void Printer(string Message, PrintStyle Style, bool SuppressCrLf);
00127 
00133         private void RunDemo(string address)
00134         {
00135             MainToolStrip.Enabled = false;
00136             Output.Clear();
00137             GclibBackgroundWorker.RunWorkerAsync(address);
00138         }
00139 
00140 
00144         private void GclibBackgroundWorker_DoWork(object sender, DoWorkEventArgs e)
00145         {
00146             PrintOutput("Running Demo with address " + e.Argument, PrintStyle.Normal);
00147             TheDemo((string)e.Argument); //call the actual demo code
00148         }
00149 
00153         private void GclibBackgroundWorker_RunWorkerCompleted(object sender,
      RunWorkerCompletedEventArgs e)
00154         {
00155             PrintOutput("Demo thread done.", PrintStyle.Normal);
00156             MainToolStrip.Enabled = true;
00157         }
00158 
00159         #endregion
00160 
```

```
00161         #region "Demo Code"
00162
00163         private void TheDemo(string address)
00164         {
00165             gclib gclib = null;
00166             try
00167             {
00168                 gclib = new gclib(); //constructor can throw, so keep it in a Try block
00169
00170                 PrintOutput("gclib version: ", PrintStyle.Normal, true);
00171                 PrintOutput(gclib.GVersion(), PrintStyle.GclibData);
00172
00173                 //*** Uncomment below for network utilities ***
00174                 //PrintOutput("Controllers requesting IP addresses...");
00175                 //string[] macs = gclib.GIpRequests();
00176                 //if (macs.Length == 0)
00177                 //    PrintOutput("None");
00178                 //else
00179                 //    foreach (string m in macs)
00180                 //        PrintOutput(m);
00181
00182                 //gclib.GAssign("192.168.0.42", "00:50:4c:20:01:23"); //Assign an IP to an unassigned
    controller
00183
00184                 PrintOutput("Available connections:");
00185                 string[] addrs = gclib.GAddresses();
00186                 if (addrs.Length == 0)
00187                 {
00188                     PrintOutput("None");
00189                 }
00190                 else
00191                 {
00192                     foreach (string a in addrs)
00193                     {
00194                         PrintOutput(a, PrintStyle.GclibData);
00195                     }
00196                 }
00197
00198                 PrintOutput("Opening connection to \"" + address + "\"... ", PrintStyle.Normal, true);
00199                 gclib.GOpen(address);
00200                 PrintOutput("Connected.", PrintStyle.Normal);
00201                 PrintOutput(gclib.GInfo(), PrintStyle.GalilData);
00202
00203                // gclib.GCommand("BN"); //send BN if IP address was assigned above
00204
00205                 PrintOutput("Sending \"MG TIME\"", PrintStyle.Normal);
00206                 PrintOutput(gclib.GCommand("MG TIME", false), PrintStyle.GalilData);
00207
00208                 PrintOutput("Downloading Program... ", PrintStyle.Normal, true);
00209                 gclib.GProgramDownload("i=0\r#A;MG i{N};i=i+1;WT10;JP#A,i<10;EN", "");
00210
00211                 PrintOutput("Uploading Program");
00212                 PrintOutput(gclib.GProgramUpload(), PrintStyle.GalilData);
00213
00214                 PrintOutput("Blocking GMessage call");
00215                 gclib.GCommand("XQ");
00216                 System.Threading.Thread.Sleep(200);
00217                 //wait a bit to queue up some messages
00218                 PrintOutput(gclib.GMessage(), PrintStyle.GalilData);
00219                 //get them all in one blocking read
00220
00221                 PrintOutput("Downloading Program... ", PrintStyle.Normal, true);
00222                 gclib.GProgramDownload("WT 1000; MG TIME; EN", "");
00223                 //prints a messsage after 1 second
00224
00225                 PrintOutput("Uploading Program");
00226                 PrintOutput(gclib.GProgramUpload(), PrintStyle.GalilData);
00227
00228                 PrintOutput("Non-blocking GMessage call", PrintStyle.Normal, true);
00229                 gclib.GCommand("XQ");
00230                 gclib.GTimeout(0);
00231                 //set a zero timeout for a non-blocking read
00232                 string msg = "";
00233                 while ((string.IsNullOrEmpty(msg)))
00234                 {
00235                     msg = gclib.GMessage();
00236                     PrintOutput(".", PrintStyle.Normal, true);
00237                     System.Threading.Thread.Sleep(20);
00238                     //do something useful here...
00239                 }
00240                 PrintOutput("Message: ", PrintStyle.Normal, true);
00241                 PrintOutput(msg.Trim(), PrintStyle.GalilData);
00242                 gclib.GTimeout(-1);
00243                 //put the timeout back
00244                 //NOTE: Both GRecord and GInterrupt also have non-blocking mode with 0 timeout.
00245
00246                 PrintOutput("Downloading Program... ", PrintStyle.Normal, true);
```

```
00247                 gclib.GProgramDownload("WT 1000; UI 8; EN", "");
00248                 //fires an interrupt after 1 second
00249
00250                 PrintOutput("Uploading Program");
00251                 PrintOutput(gclib.GProgramUpload(), PrintStyle.GalilData);
00252
00253                 PrintOutput("Non-blocking GInterrupt call", PrintStyle.Normal, true);
00254                 gclib.GCommand("XQ");
00255                 gclib.GTimeout(0);
00256                 //set a zero timeout for a non-blocking read
00257                 byte b = 0;
00258                 while ((b == 0))
00259                 {
00260                     b = gclib.GInterrupt();
00261                     PrintOutput(".", PrintStyle.Normal, true);
00262                     System.Threading.Thread.Sleep(20);
00263                     //do something useful here...
00264                 }
00265                 PrintOutput("Byte: ", PrintStyle.Normal, true);
00266                 PrintOutput(b.ToString("X02"), PrintStyle.GalilData);
00267                 gclib.GTimeout(-1);
00268                 //put the timeout back
00269
00270                 PrintOutput("Getting some synchronous data records");
00271                 gclib.GDataRecord4000 DataRecord;
00272                 for (int i = 0; i <= 10; i++)
00273                 {
00274                     DataRecord = gclib.GRecord<gclib.GDataRecord4000>(false);
00275                     PrintOutput(DataRecord.sample_number + " ", PrintStyle.GalilData, true);
00276                     //need help accessing the data record? Contact softwaresupport@galil.com
00277                     System.Threading.Thread.Sleep(10);
00278                 }
00279                 PrintOutput("");
00280
00281                 PrintOutput("Getting some asynchronous data records");
00282                 gclib.GRecordRate(10);
00283                 //set up data records every 10 ms
00284                 for (int i = 0; i <= 10; i++)
00285                 {
00286                     DataRecord = gclib.GRecord<gclib.GDataRecord4000>(true);
00287                     PrintOutput(DataRecord.sample_number + " ", PrintStyle.GalilData, true);
00288                     //no need to delay, asynchronous mode is dispatched by the Galil's RTOS.
00289                 }
00290                 gclib.GRecordRate(0);
00291                 //turn off data records
00292                 PrintOutput("");
00293
00294                 PrintOutput("Downloading an array... ", PrintStyle.Normal, true);
00295                 List<double> array = new List<double>();
00296                 for (double i = 0; i <= 9; i++)
00297                 {
00298                     array.Add(i * 2);
00299                 }
00300                 gclib.GCommand("DA *[];DM array[10]");
00301                 //arrays must be dimensioned prior to download
00302                 gclib.GArrayDownload("array", ref array);
00303
00304                 PrintOutput("Ok. Uploading array");
00305                 array = gclib.GArrayUpload("array");
00306                 foreach (double d in array)
00307                 {
00308                     PrintOutput(d.ToString("F4") + " ", PrintStyle.GalilData, true);
00309                 }
00310                 PrintOutput("");
00311
00312                 PrintOutput("Performing a write... ", PrintStyle.Normal, true);
00313                 gclib.GWrite("QR\r");
00314                 //QR returns the binary data record
00315                 PrintOutput("Ok. Reading binary data... ", PrintStyle.Normal, true);
00316                 byte[] data = gclib.GRead();
00317                 PrintOutput("Ok. Read " + data.Length + " bytes.");
00318
00319                 PrintOutput("Preparing A axis. This could cause errors if the axis is not
      initialized...", PrintStyle.Normal, true);
00320                 gclib.GCommand("AB;MO;SHA");
00321                 //compound commands are possible though typically not recommended
00322                 PrintOutput("Ok");
00323                 gclib.GCommand("PRA=5000");
00324                 gclib.GCommand("SPA=5000");
00325                 PrintOutput("Profiling a move on axis A... ", PrintStyle.Normal, true);
00326                 gclib.GCommand("BGA");
00327                 PrintOutput("Waiting for motion to complete... ", PrintStyle.Normal, true);
00328                 gclib.GMotionComplete("A");
00329                 PrintOutput("done");
00330                 PrintOutput("Going back... ", PrintStyle.Normal, true);
00331                 gclib.GCommand("PRA=-5000");
00332                 gclib.GCommand("BGA");
```

```
00333                 gclib.GMotionComplete("A");
00334                 PrintOutput("done");
00335             }
00336             catch (Exception ex)
00337             {
00338                 PrintOutput("Error: " + ex.Message, PrintStyle.Err);
00339             }
00340             finally
00341             {
00342                 if (gclib != null)
00343                     gclib.GClose(); //don't forget to close the connection
00344             }
00345         }
00346         #endregion
00347     }
00348 }
```

10.51 Program.cs File Reference

Data Structures

- class Program

10.52 Program.cs

Go to the documentation of this file.

```
00001
00003 using System;
00004 using System.Collections.Generic;
00005 using System.Linq;
00006 using System.Threading.Tasks;
00007 using System.Windows.Forms;
00008
00009 namespace gclib_example
00010 {
00011     static class Program
00012     {
00016         [STAThread]
00017         static void Main()
00018         {
00019             Application.EnableVisualStyles();
00020             Application.SetCompatibleTextRenderingDefault(false);
00021             Application.Run(new MainForm());
00022         }
00023     }
00024 }
```

10.53 commands.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.53.1 Detailed Description

Function calls for the Command Example Project.

For VB.NET, see definition in file commands.vb
Definition in file commands.cs.

10.54 commands.cs

Go to the documentation of this file.

```
00001
00011 using System;
00012
00013 namespace examples
00014 {
```

```
00015     public static partial class Examples
00016     {
00027
00028         public static int Commands(gclib gclib)
00029         {
00030
      Console.WriteLine("*********************************************************************************");
00031             Console.WriteLine("**********************    GCommand Trimmed example
      ***********************");
00032
      Console.WriteLine("*********************************************************************************");
00033
00034             Console.WriteLine("GCommand(\"PR ?,? \", true) will return a trimmed response of
      GCommand()");
00035             Console.WriteLine("The command 'PR ?,?' will return the relative " +
00036                               "position of the A and B axes");
00037             Console.WriteLine("«PR ?,? with no trim: " + gclib.GCommand("PR ?,?", false) + "»");
00038             Console.WriteLine("«PR ?,? with trim: " + gclib.GCommand("PR ?,?", true) + "»");
00039
00040
      Console.WriteLine("*********************************************************************************");
00041             Console.WriteLine("************************    GCommand Int example
      *************************");
00042
      Console.WriteLine("*********************************************************************************");
00043
00044             Console.WriteLine("Use GCmdI() to retrieve the value of GCommand as an int.");
00045             Console.WriteLine("The command 'MG _LMS' will return the available " +
00046                               "space in the vector buffer of the S plane.");
00047
00048             Console.WriteLine("MG _LMS with GCmdI(): " + gclib.GCmdI("MG _LMS"));
00049
00050
      Console.WriteLine("*********************************************************************************");
00051             Console.WriteLine("**********************    GCommand Double example
      ************************");
00052
      Console.WriteLine("*********************************************************************************");
00053
00054             Console.WriteLine("Use GCmdD() to retrieve the value of GCommand as a double.");
00055             Console.WriteLine("The command 'MG @AN[1]' will return the value of Analog Input 1");
00056
00057             Console.WriteLine("MG @AN[1] with GCmdD(): " + gclib.GCmdD("MG @AN[1]"));
00058
00059             return GALIL_EXAMPLE_OK;
00060         }
00062     }
00063 }
```

10.55 commands_example.cs File Reference

Data Structures

- class Commands_Example

  Demonstrates various uses of GCommand() and basic controller queries.

10.55.1 Detailed Description

See Commands() for implementation of logic

For VB.NET, see definition in file commands_example.vb
Definition in file commands_example.cs.

10.56 commands_example.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011 using System.Linq;
00012
00013 namespace examples
00014 {
00022
00023     class Commands_Example
00024     {
00031         public static int Main(string[] args)
```

```
00032         {
00033             int rc = Examples.GALIL_EXAMPLE_OK;
00034             gclib gclib = new gclib();
00035 
00036             try
00037             {
00038                 if(args.Count() != 1)
00039                 {
00040                     Console.WriteLine("Incorrect number of arguments provided");
00041                     Console.WriteLine("Usage: commands_example.exe <ADDRESS>");
00042 
00043                     Console.Write("\nPress any key to close the example");
00044                     Console.ReadKey();
00045                     return Examples.GALIL_EXAMPLE_ERROR;
00046                 }
00047 
00048                 string address = args[0];
00049 
00050                 gclib.GOpen(address);
00051 
00052                 rc = Examples.Commands(gclib);
00053             }
00054             catch(Exception ex)
00055             {
00056                 Examples.PrintError(gclib, ex);
00057                 rc = Examples.GALIL_EXAMPLE_ERROR;
00058             }
00059             finally
00060             {
00061                 gclib.GClose();
00062             }
00063 
00064             Console.Write("\nPress any key to close the example");
00065             Console.ReadKey();
00066 
00067             return rc;
00068         }
00069     }
00071 }
00072 
```

10.57 contour.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.57.1 Detailed Description

Function calls for the Contour Example Project.

For VB.NET, see definition in file contour.vb
Definition in file contour.cs.

10.58 contour.cs

Go to the documentation of this file.

```
00001 
00010 using System;
00011 using System.Collections.Generic;
00012 using System.Linq;
00013 using System.IO;
00014 using System.Threading;
00015 
00016 namespace examples
00017 {
00018     public static partial class Examples
00019     {
00031 
00032         public static int Contour(gclib gclib, string fileA, string fileB)
00033         {
00034             Record_Position(gclib, fileA, fileB); //Record positional data on Axis A and B
00035 
00036             List<string> positions_A = File.ReadAllText(fileA).Split(',').ToList();
00037             List<string> positions_B = File.ReadAllText(fileB).Split(',').ToList();
```

```
00038
00039             gclib.GCommand("SH AB"); //Set servo here
00040             gclib.GCommand("PA 0, 0"); //Set current position to 0
00041             gclib.GMotionComplete("AB"); //Wait for motion to complete
00042             gclib.GCommand("CM AB"); //Put axis A & B in contour mode
00043             gclib.GCommand("DT -1"); //Pauses contour mode to pre-load buffer
00044             gclib.GCommand("CD 0,0"); //Pre load buffer with zeros to prevent under buffering
00045             gclib.GCommand("CD 0,0"); //Pre load buffer with zeros to prevent under buffering
00046             gclib.GCommand("CD 0,0"); //Pre load buffer with zeros to prevent under buffering
00047             gclib.GCommand("DT 1"); //Sets the time interval for contour mode to be 2 samples
00048
00049             int capacity = 0; //Holds the capacity of the contour buffer
00050             int cmd = 0; //Holds the counter for which position to send next
00051
00052             if (positions_A.Count() != positions_B.Count())
00053             {
00054                 Console.WriteLine("Error: The two datasets are not the same size");
00055                 return Examples.GALIL_EXAMPLE_ERROR;
00056             }
00057
00058             do
00059             {
00060                 //Sleep while buffer is emptying
00061                 Thread.Sleep(400);
00062
00063                 //Stores the available space of the contour buffer in the capacity variable
00064                 capacity = gclib.GCmdI("CM?");
00065             } while (Load_Buffer(gclib, positions_A, positions_B, capacity, ref cmd));
00066
00067             gclib.GCommand("CD 0,0=0"); //End contour mode
00068
00069             return Examples.GALIL_EXAMPLE_OK;
00070         }
00071
00072         private static bool Load_Buffer(gclib gclib, List<string> positions_A, List<string>
      positions_B,
00073                                         int capacity, ref int cmd)
00074         {
00075             for (; capacity > 0; capacity--) //Fully load contour buffer
00076             {
00077                 if (cmd + 1 < positions_A.Count())
00078                 {
00079                     //Subtract previous position from new position to get how far of a move to make
00080                     double cdA = double.Parse(positions_A[cmd + 1]) - double.Parse(positions_A[cmd]);
00081
00082                     //Subtract previous position from new position to get how far of a move to make
00083                     double cdB = double.Parse(positions_B[cmd + 1]) - double.Parse(positions_B[cmd]);
00084
00085                     gclib.GCommand($"CD {cdA},{cdB}");
00086
00087                     cmd++;
00088                 }
00089                 else
00090                     return false;
00091             }
00092             return true;
00093         }
00095     }
00096 }
```

10.59 contour_example.cs File Reference

Data Structures

- class Contour_Example

  Record user's training and plays back training through contour mode.

10.59.1 Detailed Description

See Contour() for implementation of logic

For VB.NET, see definition in file contour_example.vb
Definition in file contour_example.cs.

10.60 contour_example.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011 using System.Linq;
00012
00013 namespace examples
00014 {
00026
00027     class Contour_Example
00028     {
00039         public static int Main(string[] args)
00040         {
00041             gclib gclib = new gclib();
00042             int rc = Examples.GALIL_EXAMPLE_OK;
00043
00044             if (args.Count() != 3)
00045             {
00046                 Console.WriteLine("Incorrect number of arguments provided");
00047                 Console.WriteLine("Usage: record_position_example.exe <ADDRESS> <FILE A> <FILE B>");
00048
00049                 Console.Write("\nPress any key to close the example");
00050                 Console.ReadKey();
00051                 return Examples.GALIL_EXAMPLE_ERROR;
00052             }
00053
00054             try
00055             {
00056                 string address = args[0]; //Retrieve address from command line
00057                 string fileA = args[1]; //Retrieve filepath from command line
00058                 string fileB = args[2]; //Retrieve filepath from command line
00059
00060                 gclib.GOpen(address); //Opens connection at the provided address
00061
00062                 //Record user's training and play back training through contour mode
00063                 rc = Examples.Contour(gclib, fileA, fileB);
00064             }
00065             catch(Exception ex)
00066             {
00067                 Examples.PrintError(gclib, ex);
00068                 rc = Examples.GALIL_EXAMPLE_ERROR;
00069             }
00070             finally
00071             {
00072                 gclib.GClose();
00073             }
00074
00075             Console.Write("\nPress any key to close the example");
00076             Console.ReadKey();
00077
00078             return rc;
00079         }
00080
00081     }
00083 }
00084
```

10.61 examples.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.61.1 Detailed Description

Shared methods and constants for gclib example projects.

For VB.NET, see definition in file examples.vb
Definition in file examples.cs.

10.62 examples.cs

Go to the documentation of this file.

```
00001
00014 using System;
00015
00016 namespace examples
00017 {
00026
00027     public static partial class Examples
00028     {
00029         public const int GALIL_EXAMPLE_OK = 0;
00030         public const int GALIL_EXAMPLE_ERROR = -100;
00031
00039         public static void PrintError(gclib gclib, Exception ex)
00040         {
00041             Console.WriteLine(ex.Message);
00042
00043             //If exception was not a GOpen() exception, safe to query
00044             //the controller for a human readable error string
00045             if(!ex.Message.Contains("-1101"))
00046                 Console.WriteLine(gclib.GCommand("TC 1"));
00047         }
00048     }
00050 }
```

10.63 ip_assigner.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.63.1 Detailed Description

Function calls for the IP Assigner Example Project.

For VB.NET, see definition in file ip_assigner.vb
Definition in file ip_assigner.cs.

10.64 ip_assigner.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011 using System.Linq;
00012
00013 namespace examples
00014 {
00015     public static partial class Examples
00016     {
00035
00036         public static int IP_Assigner(gclib gclib, string serial_num, byte address)
00037         {
00038             bool controller_found = false;
00039             string[] requests;
00040             do //Loop while no requests are found.
00041             {
00042                 Console.WriteLine("Searching...");
00043
00044                 //Listen for ~5 secods for controllers requesting IP addresses.
00045                 requests = gclib.GIpRequests();
00046
00047                 foreach (string request in requests)
00048                 {
00049                     Console.WriteLine(request);
00050                 }
00051
00052             } while (requests.Count() < 1);
00053
00054             foreach (string request in requests)
00055             {
00056                 string[] controller_params = request.Split(new string[] { ", " },
      StringSplitOptions.None);
00057
00058                 //Parameters are ordered as:
00059                 //[Model #], [Serial #], [MAC Address], [Connection Name], [IP Address]
00060
00061                 if (controller_params.Count() < 5)
```

```
00062                 {
00063                     Console.WriteLine("Unexpected controller format");
00064                     return GALIL_EXAMPLE_ERROR;
00065                 }
00066 
00067                 string mac = controller_params[2];
00068                 string ip = controller_params[4];
00069 
00070                 //If controller contains the user entered serial number
00071                 if (serial_num == controller_params[1])
00072                 {
00073                     Console.WriteLine("Controller Match Found");
00074                     controller_found = true;
00075 
00076                     //Splits the found ip address into individual bytes
00077                     string[] ip_bytes = ip.Split('.');
00078 
00079                     //Rebuild the ip address using the user provided address as the last byte
00080                     string new_ip = $"{ip_bytes[0]}.{ip_bytes[1]}.{ip_bytes[2]}.{address}";
00081 
00082                     //Assign the new ip address to the controller
00083                     gclib.GAssign(new_ip, mac);
00084 
00085                     //Open a connection at the new ip address
00086                     gclib.GOpen(new_ip);
00087 
00088                     //Burns the newly assigned ip address to non-volatile EEPROM memory
00089                     gclib.GCommand("BN");
00090 
00091                     Console.WriteLine("IP Address assigned");
00092 
00093                     //Write the connection string to the console
00094                     Console.WriteLine(gclib.GInfo());
00095 
00096                     break;
00097                 }
00098             }
00099 
00100             if (!controller_found)
00101                 Console.Write("No controller matched the entered serial number");
00102 
00103             return GALIL_EXAMPLE_OK;
00104         }
00106     }
00107 }
```

10.65 ip_assigner_example.cs File Reference

Data Structures

- class IP_Assigner_Example

  Assigns controller an IP Adress given a serial number and a 1 byte address.

10.65.1 Detailed Description

See IP_Assigner() for implementation of logic

For VB.NET, see definition in file ip_assigner_example.vb
Definition in file ip_assigner_example.cs.

10.66 ip_assigner_example.cs

Go to the documentation of this file.

```
00001 
00010 using System;
00011 using System.Linq;
00012 
00013 namespace examples
00014 {
00024 
00025     class IP_Assigner_Example
00026     {
00035         public static int Main(string[] args)
00036         {
00037             gclib gclib = new gclib();
00038             int rc = Examples.GALIL_EXAMPLE_OK;
```

```
00039
00040             try
00041             {
00042                 if(args.Count() != 2)
00043                 {
00044                     Console.WriteLine("Incorrect number of arguments provided");
00045                     Console.WriteLine("Usage: ip_assigner_example.exe <SERIAL #> <1 Byte Address>");
00046
00047                     Console.Write("\nPress any key to close the example");
00048                     Console.ReadKey();
00049                     return Examples.GALIL_EXAMPLE_ERROR;
00050                 }
00051
00052                 string serial_num = args[0];
00053                 bool ok = Byte.TryParse(args[1], out byte address);
00054
00055                 if(!ok || address < 0 || address > 255)
00056                 {
00057                     Console.WriteLine("Please enter a number between 0 and 255 for the address.\n" +
00058                     " This will be used as the last number in the IP Address\n" +
00059                     "Usage: ip_assigner_example.exe <SERIAL #> <1 Byte Address>");
00060
00061                     Console.Write("\nPress any key to close the example");
00062                     Console.ReadKey();
00063
00064                     return Examples.GALIL_EXAMPLE_ERROR;
00065                 }
00066
00067                 rc = Examples.IP_Assigner(gclib, serial_num, address);
00068             }
00069             catch(Exception ex)
00070             {
00071                 Examples.PrintError(gclib, ex);
00072                 rc = Examples.GALIL_EXAMPLE_ERROR;
00073             }
00074             finally
00075             {
00076                 gclib.GClose();
00077             }
00078
00079             Console.Write("\nPress any key to close the example");
00080             Console.ReadKey();
00081
00082             return rc;
00083         }
00084     }
00086 }
```

10.67 jog.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.67.1 Detailed Description

Function calls for the Jog Example Project.

For VB.NET, see definition in file jog.vb
Definition in file jog.cs.

10.68 jog.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011
00012 namespace examples
00013 {
00014     public static partial class Examples
00015     {
00034
00035         public static int Jog(gclib gclib)
00036         {
00037             gclib.GCommand("ST");       // Stop all motors
```

```
00038             gclib.GMotionComplete("A"); // Wait for motion to complete
00039             gclib.GCommand("SHA");      // Set servo here
00040             gclib.GCommand("DPA=0");    // Start position at absolute zero
00041             gclib.GCommand("JGA=0");    // Start jogging with 0 speed
00042             gclib.GCommand("BGA");      // Begin motion on A Axis
00043
00044             bool isJogging = true;
00045             int speed = 0;
00046
00047             Console.WriteLine("Enter a character on the keyboard to change the" +
00048                               " motor's speed:\n<q> Quit\n<a> -2000 counts/s\n" +
00049                               "<s> -500  counts/s\n<d> +500  counts/s\n<f> " +
00050                               "+2000 counts/s\n<r> Direction Reversal\n");
00051
00052             while (isJogging)
00053             {
00054                 gclib.GCommand("JGA=" + speed);
00055
00056                 Console.WriteLine("Jog Speed: " + speed);
00057
00058                 switch (Console.ReadKey(true).Key)
00059                 {
00060                     case ConsoleKey.Q:
00061                         isJogging = false;
00062                         break;
00063                     case ConsoleKey.A:
00064                         speed -= 2000;
00065                         break;
00066                     case ConsoleKey.S:
00067                         speed -= 500;
00068                         break;
00069                     case ConsoleKey.D:
00070                         speed += 500;
00071                         break;
00072                     case ConsoleKey.F:
00073                         speed += 2000;
00074                         break;
00075                     case ConsoleKey.R:
00076                         speed *= -1;
00077                         break;
00078                 }
00079             }
00080
00081             gclib.GCommand("ST");
00082             gclib.GMotionComplete("A");
00083
00084             return GALIL_EXAMPLE_OK;
00085         }
00087     }
00088 }
```

10.69 jog_example.cs File Reference

Data Structures

- class Jog_Example

  Accepts user-input at the command line to control the speed of the controller in Jog mode.

10.69.1 Detailed Description

See Jog() for implementation of logic

For VB.NET, see definition in file jog_example.vb
Definition in file jog_example.cs.

10.70 jog_example.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011 using System.Linq;
00012
00013 namespace examples
00014 {
00022
00023     class Jog_Example
```

```
00024     {
00031         public static int Main(string[] args)
00032         {
00033             int rc = Examples.GALIL_EXAMPLE_OK;
00034             gclib gclib = new gclib();
00035 
00036             try
00037             {
00038                 if (args.Count() != 1)
00039                 {
00040                     Console.WriteLine("Incorrect number of arguments provided");
00041                     Console.WriteLine("Usage: jog_example.exe <ADDRESS>");
00042 
00043                     Console.Write("\nPress any key to close the example");
00044                     Console.ReadKey();
00045                     return Examples.GALIL_EXAMPLE_ERROR;
00046                 }
00047 
00048                 string address = args[0]; //Get IP address from the command line
00049 
00050                 gclib.GOpen(address); //Open a connection at the provided address
00051 
00052                 rc = Examples.Jog(gclib);
00053             }
00054             catch(Exception ex)
00055             {
00056                 Examples.PrintError(gclib, ex);
00057                 rc = Examples.GALIL_EXAMPLE_ERROR;
00058             }
00059             finally
00060             {
00061                 gclib.GClose();
00062             }
00063 
00064 
00065             Console.Write("\nPress any key to close the example");
00066             Console.ReadKey();
00067             return rc;
00068         }
00069     }
00071 }
```

10.71 message.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.71.1 Detailed Description

Function calls for the Message Example Project.

For VB.NET, see definition in file message.vb
Definition in file message.cs.

10.72 message.cs

Go to the documentation of this file.

```
00001 
00010 using System;
00011 
00012 namespace examples
00013 {
00014     public static partial class Examples
00015     {
00026 
00027         public static int Message(gclib gclib)
00028         {
00029             Console.WriteLine("*****************************************************************");
00030             Console.WriteLine("Example GMessage() usage");
00031             Console.WriteLine("*****************************************************************");
00032 
00033             gclib.GCommand("TR0"); //Turn off trace
00034 
00035             //This program will force one message to appear as two separate packets.
```

```
00036             gclib.GProgramDownload("MG \"HELLO \" {N}\r" +
00037                                    "MG \"WORLD \"\r" +
00038                                    "EN");
00039
00040             gclib.GCommand("XQ"); //Begins execution of program on controller
00041
00042             string buf = "";
00043             string message = "";
00044
00045             // It is important to note that a message can be too large to read in one
00046             // GMessage() call. Keep calling GMessage() while there are no errors to
00047             // get the full message.
00048
00049             //While still receiving messages
00050             while ((buf = gclib.GMessage()) != "")
00051             {
00052                 for (int b = 0; b < buf.Length; b++) //While message characters are in the buffer
00053                 {
00054                     message += buf[b]; //Copy chars from  buffer to message
00055
00056                     //If the message ends in "\r\n" it is ready to be terminated
00057                     if (message.Length > 2 && message[message.Length - 1] == '\n' &&
      message[message.Length - 2] == '\r')
00058                     {
00059                         Console.WriteLine(message);
00060                         message = ""; //Reset message index
00061                     }
00062                 }
00063             }
00064
00065             //Downloads program to the controller
00066             gclib.GCommand("TR1"); //Turn on trace
00067             gclib.GProgramDownload(
00068                 "i=0\r" +
00069                 "#A\r" +
00070                 "MGi\r" +
00071                 "i=i+1\r" +
00072                 "WT100\r" +
00073                 "JP#A,i<1\r" +
00074                 "i=i/0\r" +
00075                 "EN");
00076             gclib.GCommand("XQ"); //Begins execution of program on controller
00077
00078             // Lines returned by GMessage() can be one of three types:
00079             // 1) Standard Lines begin with a space (" ")
00080             // 2) Crashed code begins with a question mark ("?")
00081             // 3) Trace Lines begin with a line number ("1,6,15...")
00082
00083             //While still receiving messages
00084             while ((buf = gclib.GMessage()) != "")
00085             {
00086                 for (int b = 0; b < buf.Length; b++) //While message characters are in the buffer
00087                 {
00088                     message += buf[b]; //Copy chars from buffer to message
00089
00090                     //If the message ends in "\r\n" its ready to be terminated
00091                     if (message.Length > 2 && message[message.Length - 1] == '\n' &&
      message[message.Length - 2] == '\r')
00092                     {
00093                         if (message[0] == ' ') //Standard Lines begin with a space (" ")
00094                             Console.Write("Standard Line: ");
00095                         else if (message[0] == '?') //Crashed code begins with a question mark ("?")
00096                             Console.Write("Crashed Code: ");
00097                         else //Trace Lines begin with a line number ("1,6,15...")
00098                             Console.Write("Trace Line: ");
00099
00100                         Console.WriteLine(message);
00101                         message = "";
00102                     }
00103                 }
00104             }
00105
00106             return Examples.GALIL_EXAMPLE_OK;
00107         }
00109     }
00110 }
```

10.73 message_example.cs File Reference

Data Structures

- class Message_Example

  Demonstrates how to handle and interpret messages from the controller.

10.73.1 Detailed Description

See Message() for implementation of logic

For VB.NET, see definition in file message_example.vb
Definition in file message_example.cs.

10.74 message_example.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011 using System.Linq;
00012
00013 namespace examples
00014 {
00022
00023     class Message_Example
00024     {
00031         public static int Main(string[] args)
00032         {
00033             int rc = Examples.GALIL_EXAMPLE_OK;
00034             gclib gclib = new gclib();
00035
00036             try
00037             {
00038                 if (args.Count() != 1)
00039                 {
00040                     Console.WriteLine("Incorrect number of arguments provided");
00041                     Console.WriteLine("Usage: message_example.exe <ADDRESS>");
00042
00043                     Console.Write("\nPress any key to close the example");
00044                     Console.ReadKey();
00045                     return Examples.GALIL_EXAMPLE_ERROR;
00046                 }
00047
00048                 string address = args[0]; //Retrieve address from command line
00049
00050                 // Opens a connection at the provided address
00051                 // The --subscribe MG addition is needed to subscribe to
00052                 // controller messages
00053                 gclib.GOpen(address + " --subscribe MG");
00054
00055                 rc = Examples.Message(gclib);
00056             }
00057             catch(Exception ex)
00058             {
00059                 Examples.PrintError(gclib, ex);
00060                 rc = Examples.GALIL_EXAMPLE_ERROR;
00061             }
00062             finally
00063             {
00064                 gclib.GClose();
00065             }
00066
00067
00068             Console.Write("\nPress any key to close the example");
00069             Console.ReadKey();
00070             return rc;
00071         }
00072     }
00074 }
```

10.75 motion_complete.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.75.1 Detailed Description

Function calls for the Motion Complete Example Project.

For VB.NET, see definition in file motion_complete.vb

Definition in file motion_complete.cs.

10.76 motion_complete.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011
00012 namespace examples
00013 {
00014     public static partial class Examples
00015     {
00025
00026         public static int Motion_Complete(gclib gclib)
00027         {
00028             Console.WriteLine("************************************************************");
00029             Console.WriteLine("Example GInterrupt() usage");
00030             Console.WriteLine("************************************************************");
00031
00032             //Simple check for appropriate communication bus
00033             //EI will fail below if interrupts are not supported at all
00034             bool ei_support = false;
00035
00036             if (gclib.GCommand("WH").Contains("IH"))
00037                 ei_support = true;
00038
00039             if (!ei_support)
00040             {
00041                 Console.WriteLine("No support on this bus");
00042                 return GALIL_EXAMPLE_ERROR;
00043             }
00044
00045             //Flush interrupts
00046             gclib.GCommand("EI0,0"); //Turn off interrupts
00047             gclib.GTimeout(0); //Zero timeout
00048
00049             //Flush interrupts, status will be zero when queue is empty
00050             while (gclib.GInterrupt() > 0) ;
00051
00052             gclib.GTimeout(-1); //Restore timeout
00053
00054             //Independent Motion
00055             gclib.GCommand("DP 0,0"); //define position zero on A and B
00056             Console.WriteLine("Position: " + gclib.GCommand("RP")); //Print reference position
00057
00058             gclib.GCommand("SP 4000,4000"); //Set up speed
00059             gclib.GCommand("AC 1280000, 1280000"); //acceleration
00060             gclib.GCommand("DC 1280000, 1280000"); //deceleration
00061             gclib.GCommand("PR 8000, 10000"); //Position Relative.  B will take longer to make its
    move.
00062             gclib.GCommand("SH AB"); //Servo Here
00063
00064             Console.WriteLine("Beginning independent motion...");
00065             gclib.GCommand("BG AB"); //Begin motion
00066             Check_Interrupts(gclib, "AB"); //Block until motion is complete on axes A and B
00067             Console.WriteLine("Motion Complete on A and B");
00068             Console.WriteLine("Position: " + gclib.GCommand("RP")); //Print reference position
00069
00070             return GALIL_EXAMPLE_OK;
00071         }
00072
00073         private static void Check_Interrupts(gclib gclib, string axes)
00074         {
00075             //bit mask of running axes, axes arg is trusted to provide running axes.
00076             //Low bit indicates running.
00077             byte axis_mask = 0xFF;
00078
00079             //iterate through all chars in axes to make the axis mask
00080             for (int i = 0; i < axes.Length; i++)
00081             {
00082                 //support just A-H
00083                 switch (axes[i])
00084                 {
00085                     case 'A':
00086                         axis_mask &= 0xFE;
00087                         break;
00088                     case 'B':
00089                         axis_mask &= 0xFD;
00090                         break;
00091                     case 'C':
00092                         axis_mask &= 0xFB;
00093                         break;
00094                     case 'D':
00095                         axis_mask &= 0xF7;
```

```
00096                       break;
00097                   case 'E':
00098                       axis_mask &= 0xEF;
00099                       break;
00100                   case 'F':
00101                       axis_mask &= 0xDF;
00102                       break;
00103                   case 'G':
00104                       axis_mask &= 0xBF;
00105                       break;
00106                   case 'H':
00107                       axis_mask &= 0x7F;
00108                       break;
00109               }
00110           }
00111
00112           //send EI axis mask to set up interrupt events.
00113           gclib.GCommand("EI " + ~axis_mask);
00114
00115           byte status;
00116
00117           while (axis_mask != 0xFF) //wait for all interrupts to come in
00118           {
00119               switch (status = gclib.GInterrupt())
00120               {
00121                   case 0xD0: //Axis A complete
00122                       axis_mask |= 0x01;
00123                       break;
00124                   case 0xD1: //Axis B complete
00125                       axis_mask |= 0x02;
00126                       break;
00127                   case 0xD2: //Axis C complete
00128                       axis_mask |= 0x04;
00129                       break;
00130                   case 0xD3: //Axis D complete
00131                       axis_mask |= 0x08;
00132                       break;
00133                   case 0xD4: //Axis E complete
00134                       axis_mask |= 0x10;
00135                       break;
00136                   case 0xD5: //Axis F complete
00137                       axis_mask |= 0x20;
00138                       break;
00139                   case 0xD6: //Axis G complete
00140                       axis_mask |= 0x40;
00141                       break;
00142                   case 0xD7: //Axis H complete
00143                       axis_mask |= 0x80;
00144                       break;
00145               }
00146           }
00147       }
00149   }
00150 }
```

10.77 motion_complete_example.cs File Reference

Data Structures

- class Motion_Complete_Example

  Uses controller interrupts to detect when motion is complete.

10.77.1 Detailed Description

See Motion_Complete() for implementation of logic

For VB.NET, see definition in file motion_complete_example.vb
Definition in file motion_complete_example.cs.

10.78 motion_complete_example.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011 using System.Linq;
00012
```

```
00013 namespace examples
00014 {
00022
00023     class Motion_Complete_Example
00024     {
00031         public static int Main(string[] args)
00032         {
00033             int rc = Examples.GALIL_EXAMPLE_OK;
00034             gclib gclib = new gclib();
00035
00036             try
00037             {
00038                 if(args.Count() != 1)
00039                 {
00040                     Console.WriteLine("Incorrect number of arguments provided");
00041                     Console.WriteLine("Usage: motion_complete_exmample.exe <ADDRESS>");
00042
00043                     Console.Write("\nPress any key to close the example");
00044                     Console.ReadKey();
00045                     return Examples.GALIL_EXAMPLE_ERROR;
00046                 }
00047
00048                 //Retrieves address from the command line
00049                 string address = args[0];
00050
00051                 //Open a connection at the provided address and subcribe to event interrupts
00052                 gclib.GOpen(address + " --subscribe EI");
00053
00054                 //Uses interrupts to track when the motion of the controller is completed

00055                 rc = Examples.Motion_Complete(gclib);
00056             }
00057             catch(Exception ex)
00058             {
00059                 Examples.PrintError(gclib, ex);
00060                 rc = Examples.GALIL_EXAMPLE_ERROR;
00061             }
00062             finally
00063             {
00064                 gclib.GClose();
00065             }
00066
00067             Console.Write("\nPress any key to close the example");
00068             Console.ReadKey();
00069
00070             return rc;
00071         }
00072     }
00074 }
```

10.79 position_tracking.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.79.1 Detailed Description

Function calls for the Position Tracking Example Project.

For VB.NET, see definition in file position_tracking.vb
Definition in file position_tracking.cs.

10.80 position_tracking.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011
00012 namespace examples
00013 {
00014     public static partial class Examples
00015     {
00027
00028         public static int Position_Tracking(gclib gclib, int speed)
```

```
00029         {
00030             int acc = 100 * speed; // Set acceleration/deceleration to 100 times speed
00031 
00032             gclib.GCommand("STA");      // Stop motor
00033             gclib.GMotionComplete("A"); // Wait for motion to complete
00034             gclib.GCommand("SHA");      // Set servo here
00035             gclib.GCommand("DPA=0");    // Start position at absolute zero
00036             gclib.GCommand("PTA=1");    // Start position tracking mode on A axis
00037 
00038             gclib.GCommand("SPA=" + speed); // Set speed
00039             gclib.GCommand("ACA=" + acc); // Set acceleration
00040             gclib.GCommand("DCA=" + acc); // Set deceleration
00041 
00042             Console.WriteLine("Begin Position Tracking with speed " + speed +
00043                               ". Enter a non-number to exit.\n");
00044             int position;
00045 
00046             //Loop asking user for new position.  End loop when user enters a non-number
00047             while (true)
00048             {
00049                 Console.WriteLine("Enter a new position:");
00050                 bool ok = int.TryParse(Console.ReadLine(), out position);
00051 
00052                 if (ok) //A valid position was provided
00053                 {
00054                     gclib.GCommand("PAA=" + position); // Go to new position
00055                 }
00056                 else //A non-number was entered
00057                 {
00058                     Console.WriteLine("Position Tracking has exited");
00059                     break;
00060                 }
00061             }
00062 
00063             gclib.GCommand("STA"); //stop motor
00064             gclib.GMotionComplete("A"); //Wait for motion to complete
00065 
00066             return GALIL_EXAMPLE_OK;
00067         }
00069     }
00070 }
```

10.81 position_tracking_example.cs File Reference

Data Structures

- class Position_Tracking_Example

  Places controller into position tracking mode. Accepts user-defined positional values at the command line.

10.81.1 Detailed Description

See Position_Tracking() for implementation of logic

For VB.NET, see definition in file position_tracking_example.vb
Definition in file position_tracking_example.cs.

10.82 position_tracking_example.cs

Go to the documentation of this file.

```
00001 
00010 using System;
00011 using System.Linq;
00012 
00013 namespace examples
00014 {
00025 
00026     class Position_Tracking_Example
00027     {
00036         public static int Main(string[] args)
00037         {
00038             int rc = Examples.GALIL_EXAMPLE_OK;
00039             gclib gclib = new gclib();
00040 
00041             try
00042             {
00043                 int speed = 0;
```

```
00044
00045                 if(args.Count() == 1)
00046                 {
00047                     speed = 5000;
00048                 }
00049                 else if(args.Count() == 2)
00050                 {
00051                     bool ok = int.TryParse(args[1], out speed);
00052
00053                     if(!ok)
00054                     {
00055                         Console.WriteLine("An invalid speed was entered.  " +
00056                                           "Please enter a valid integer for speed.");
00057                         Console.WriteLine("Usage: position_tracking_example.exe <ADDRESS>
      <SPEED=5000>");
00058
00059                         Console.Write("\nPress any key to close the example");
00060                         Console.ReadKey();
00061                         return Examples.GALIL_EXAMPLE_ERROR;
00062                     }
00063                 }
00064                 else
00065                 {
00066                     Console.WriteLine("Incorrect number of arguments provided");
00067                     Console.WriteLine("Usage: position_tracking_example.exe <ADDRESS> <SPEED=5000>");
00068
00069                     Console.Write("\nPress any key to close the example");
00070                     Console.ReadKey();
00071                     return Examples.GALIL_EXAMPLE_ERROR;
00072                 }
00073
00074                 string address = args[0]; //Retrieve address from command line
00075                 gclib.GOpen(address); // Open a connection at the provided address
00076
00077                 rc = Examples.Position_Tracking(gclib, speed); // Begin position tracking mode
00078             }
00079             catch(Exception ex)
00080             {
00081                 Examples.PrintError(gclib, ex);
00082                 rc = Examples.GALIL_EXAMPLE_ERROR;
00083             }
00084             finally
00085             {
00086                 gclib.GClose();
00087             }
00088
00089             Console.Write("\nPress any key to close the example");
00090             Console.ReadKey();
00091
00092             return rc;
00093         }
00094     }
00096 }
```

10.83 record_position.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.83.1 Detailed Description

Function calls for the Record Position Example Project.

For VB.NET, see definition in file record_position.vb
Definition in file record_position.cs.

10.84 record_position.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011 using System.Collections.Generic;
00012 using System.Linq;
00013 using System.IO;
```

```
00014 using System.Threading;
00015
00016 namespace examples
00017 {
00018     public static partial class Examples
00019     {
00031
00032         public static int Record_Position(gclib gclib, string fileA, string fileB)
00033         {
00034             StreamWriter writerA = new StreamWriter(fileA, false);
00035             StreamWriter writerB = new StreamWriter(fileB, false);
00036
00037             int recording = 1;
00038
00039             gclib.GProgramDownload(
00040                 "RC 0;' Disable Recording\n" +
00041                 "DP 0, 0;' Set current position to 0\r" +
00042                 "DM posA[1000], posB[1000];' Define a new array that will hold positional data\r" +
00043                 "RA posA[], posB[];' Sets position array to be where recorded data will be stored\r" +
00044                 "RD _TPA, _TPB;' Defines Position to be the type of data that will be recorded\r" +
00045                 "RC 1,-1000;' Begins recording at 512Hz in continuous mode\r" +
00046                 "MO AB;' Turns motors off\r" +
00047                 "AI -1;' Waits for active low on Input 1\r" +
00048                 "RC 0;' Disable Recording after Input 1 goes low\r" +
00049                 "EN;' End program"
00050             );
00051
00052             gclib.GCommand("XQ");
00053
00054             int rd = 0;
00055             int previous_rd = 0;
00056             bool leading_comma = false;
00057
00058             do
00059             {
00060                 Thread.Sleep(1000); //Sleep while we wait for roughly half the array to be written
00061                 rd = gclib.GCmdI("MG _RD"); //Gets address of next value in the position array
00062
00063                 //Get values from posA[] array and write to file
00064                 Write_Array_To_File(gclib, writerA, "posA", previous_rd, rd, leading_comma);
00065
00066                 //Get values from posB[] array and write to file
00067                 Write_Array_To_File(gclib, writerB, "posB", previous_rd, rd, leading_comma);
00068
00069                 leading_comma = true;
00070
00071                 recording = gclib.GCmdI("MG _RC"); //Check status of RC
00072
00073                 previous_rd = rd;
00074
00075             } while (recording > 0); //While recording is active
00076
00077             writerA.Close();
00078             writerB.Close();
00079
00080             return Examples.GALIL_EXAMPLE_OK;
00081         }
00082
00083         private static void Write_Array_To_File(gclib gclib, StreamWriter writer, string array_name,
      int previous_rd, int rd, bool leading_comma)
00084         {
00085             List<double> values = new List<double>();
00086             if (previous_rd < rd) // No array wrap around
00087             {
00088                 //Grab list of doubles from controller and add it to values
00089                 values.AddRange(gclib.GArrayUpload(array_name, (short)previous_rd, (short)(rd - 1)));
00090             }
00091             else
00092             {
00093                 //Grab list of doubles from controller and add it to values
00094                 values.AddRange(gclib.GArrayUpload(array_name, (short)previous_rd, 999));
00095
00096                 if (rd != 0)
00097                 {
00098                     //Grab list of doubles from controller and add it to values
00099                     values.AddRange(gclib.GArrayUpload(array_name, 0, (short)(rd - 1)));
00100                 }
00101             }
00102
00103
00104             for (int i = 0; i < values.Count(); i++)
00105             {
00106                 if (leading_comma)
00107                     writer.Write(", ");
00108
00109                 leading_comma = true;
00110
```

```
00111                 writer.Write(String.Format(("{0:0.000}"), values[i]));
00112             }
00113         }
00115     }
00116 }
```

10.85 record_position_example.cs File Reference

Data Structures

- class Record_Position_Example

  Takes two file paths at the command line to hold positional data for Axis A and Axis B. Positional data is saved to the two files until an analog input value changes.

10.85.1 Detailed Description

See Record_Position() for implementation of logic

For VB.NET, see definition in file record_position_example.vb
Definition in file record_position_example.cs.

10.86 record_position_example.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011 using System.Linq;
00012
00013 namespace examples
00014 {
00026
00027     class Record_Position_Example
00028     {
00038         public static int Main(string[] args)
00039         {
00040             int rc = Examples.GALIL_EXAMPLE_OK;
00041             gclib gclib = new gclib();
00042
00043             try
00044             {
00045                 if(args.Count() != 3)
00046                 {
00047                     Console.WriteLine("Incorrect number of arguments provided");
00048                     Console.WriteLine("Usage: record_position_example.exe <ADDRESS> <FILE A> <FILE
      B>");
00049
00050                     Console.Write("\nPress any key to close the example");
00051                     Console.ReadKey();
00052                     return Examples.GALIL_EXAMPLE_ERROR;
00053                 }
00054
00055                 string address = args[0]; //Retrieve address from command line
00056                 string fileA = args[1]; //Retrieve filepath from command line
00057                 string fileB = args[2]; //Retrieve filepath from command line
00058
00059                 gclib.GOpen(address); //Opens connection at the provided address
00060
00061                 //Record user's training and saves to a text file
00062                 rc = Examples.Record_Position(gclib, fileA, fileB);
00063             }
00064             catch(Exception ex)
00065             {
00066                 Examples.PrintError(gclib, ex);
00067                 return Examples.GALIL_EXAMPLE_ERROR;
00068             }
00069             finally
00070             {
00071                 gclib.GClose();
00072             }
00073
00074             Console.Write("\nPress any key to close the example");
00075             Console.ReadKey();
00076
00077             return rc;
00078         }
00079     }
```

```
00081 }
```

10.87 Remote_Client.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.87.1 Detailed Description

Function calls for the Remote Client Example Project.

For VB.NET, see definition in file remote_client.vb
Definition in file Remote_Client.cs.

10.88 Remote_Client.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011
00012 namespace examples
00013 {
00014     public static partial class Examples
00015     {
00032
00033         public static int Remote_Client()
00034         {
00035             gclib gclib = new gclib();
00036
00037             bool loop = true;
00038             string[] servers_list = Array.Empty<string>();
00039
00040             Console.WriteLine("<s> List available servers on the network\n" +
00041                               "<h> List available hardware on currently connected server\n" +
00042                               "<0-9> Enter numbers 0-9 to connect to a server by index\n" +
00043                               "<l> Set active server back to local server\n" +
00044                               "<q> Quit");
00045
00046             while (loop)
00047             {
00048                 char input = Console.ReadKey(true).KeyChar;
00049
00050                 if(input == 'q')
00051                 {
00052                     loop = false;
00053                 }
00054                 else if(input == 's')
00055                 {
00056                     Console.WriteLine("Available Servers:");
00057                     servers_list = gclib.GListServers();
00058                     Print_Servers_List(servers_list);
00059                 }
00060                 else if(input >= '0' && input <= '9')
00061                 {
00062                     int index = input - '0';
00063                     if(servers_list.Length > 0 && index < servers_list.Length)
00064                     {
00065                         gclib.GSetServer(servers_list[index]);
00066                         Console.WriteLine("Server set to: " + servers_list[index]);
00067                     }
00068                 }
00069                 else if(input == 'l')
00070                 {
00071                     gclib.GSetServer("Local");
00072                     Console.WriteLine("Server set to: Local");
00073                 }
00074                 else if(input == 'h')
00075                 {
00076                     string[] addresses = gclib.GAddresses();
00077
00078                     foreach(string address in addresses)
00079                     {
00080                         Console.WriteLine(address);
00081                     }
```

```
00082                 }
00083             }
00084
00085             return GALIL_EXAMPLE_OK;
00086         }
00087
00088         private static void Print_Servers_List(string[] servers_list)
00089         {
00090             if(servers_list.Length == 0)
00091             {
00092                 Console.WriteLine("none");
00093             }
00094             else
00095             {
00096                 for(int i = 0; i < servers_list.Length; i++)
00097                 {
00098                     Console.WriteLine("<" + i + "> " + servers_list[i]);
00099                 }
00100             }
00101         }
00103     }
00104 }
```

10.89 remote_client_example.cs File Reference

Data Structures

- class Remote_Client_Example

  Demonstrates various uses of GListServers() and GSetServer()

10.89.1 Detailed Description

See Remote_Client() for implementation of logic

For VB.NET, see definition in file remote_client_example.vb
Definition in file remote_client_example.cs.

10.90 remote_client_example.cs

Go to the documentation of this file.

```
00001
00011 using System;
00012 using System.Linq;
00013
00014 namespace examples
00015 {
00023
00024     class Remote_Client_Example
00025     {
00031         public static int Main()
00032         {
00033             int rc = Examples.GALIL_EXAMPLE_OK;
00034
00035             try
00036             {
00037                 rc = Examples.Remote_Client();
00038             }
00039             catch (Exception ex)
00040             {
00041                 Console.WriteLine(ex.Message);
00042                 rc = Examples.GALIL_EXAMPLE_ERROR;
00043             }
00044
00045             Console.Write("\nPress any key to close the example");
00046             Console.ReadKey();
00047
00048             return rc;
00049         }
00050     }
00052 }
```

10.91 Remote_Server.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.91.1 Detailed Description

Function calls for the Remote Server Example Project.

For VB.NET, see definition in file remote_server.vb
Definition in file Remote_Server.cs.

10.92 Remote_Server.cs

Go to the documentation of this file.

```
00001
00010 using System;
00011
00012 namespace examples
00013 {
00014     public static partial class Examples
00015     {
00031
00032         public static int Remote_Server(string server_name)
00033         {
00034             gclib gclib = new gclib();
00035
00036             bool loop = true;
00037
00038             Console.WriteLine("<p> Publish this server to the network\n" +
00039                  "<r> Remove this server from the network\n" +
00040                  "<q> Quit");
00041
00042             while (loop)
00043             {
00044                 switch (Console.ReadKey(true).Key)
00045                 {
00046                     case ConsoleKey.Q:
00047                         loop = false;
00048                         break;
00049                     case ConsoleKey.P:
00050                         gclib.GPublishServer(server_name, true, false);
00051                         Console.WriteLine("Published Server");
00052                         break;
00053                     case ConsoleKey.R:
00054                         gclib.GPublishServer(server_name, false, false);
00055                         Console.WriteLine("Removed Server");
00056                         break;
00057                 }
00058             }
00059
00060             return GALIL_EXAMPLE_OK;
00061         }
00063     }
00064 }
```

10.93 remote_server_example.cs File Reference

Data Structures

- class Remote_Server_Example

  Demonstrates various uses of GPublishServer()

10.93.1 Detailed Description

See Remote_Server() for implementation of logic

For VB.NET, see definition in file remote_server_example.vb
Definition in file remote_server_example.cs.

10.94 remote_server_example.cs

Go to the documentation of this file.

```
00001
00011 using System;
00012 using System.Linq;
00013
00014 namespace examples
00015 {
00023
00024     class Remote_Server_Example
00025     {
00032         public static int Main(string[] args)
00033         {
00034             int rc = Examples.GALIL_EXAMPLE_OK;
00035
00036             try
00037             {
00038                 string server_name;
00039
00040                 if (args.Count() < 1)
00041                 {
00042                     Console.Write("Enter server name: ");
00043                     server_name = Console.ReadLine();
00044                 }
00045                 else
00046                 {
00047                     server_name = args[0];
00048                 }
00049
00050                 rc = Examples.Remote_Server(server_name);
00051             }
00052             catch (Exception ex)
00053             {
00054                 Console.WriteLine(ex.Message);
00055                 rc = Examples.GALIL_EXAMPLE_ERROR;
00056             }
00057
00058             Console.Write("\nPress any key to close the example");
00059             Console.ReadKey();
00060
00061             return rc;
00062         }
00063     }
00065 }
```

10.95 vector_mode.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

10.95.1 Detailed Description

Function calls for the Vector Mode Example Project.

For VB.NET, see definition in file vector_mode.vb
Definition in file vector_mode.cs.

10.96 vector_mode.cs

Go to the documentation of this file.

```
00001
00010 using System.IO;
00011 using System.Threading;
00012
00013 namespace examples
00014 {
00015     public static partial class Examples
00016     {
00044
00045         public static int Vector_Mode(gclib gclib, string file)
00046         {
00047             gclib.GCommand("ST"); // Stop all motors
```

```
00048             gclib.GCommand("SH AB"); // Set servo here
00049             gclib.GCommand("DP 0,0"); // Start position at absolute zero
00050 
00051             gclib.GCommand("CAS"); // Defines S as active coordinate system
00052             gclib.GCommand("VS 20000"); // Defines vector speed
00053             gclib.GCommand("VA 200000"); // Defines vector acceleration
00054             gclib.GCommand("VD 200000"); // Defines vector decelerlation
00055             gclib.GCommand("VM AB"); // Begin vector segment
00056 
00057             using (StreamReader reader = new StreamReader(file))
00058             {
00059                 //Stores the available space of the vector buffer in the capacity variable
00060                 int capacity = gclib.GCmdI("MG _LMS");
00061                 Load_Buffer(gclib, reader, capacity);
00062 
00063                 gclib.GCommand("BG S");
00064 
00065                 do // Load buffer with more commands
00066                 {
00067                     Thread.Sleep(100);
00068 
00069                     //Stores the available space of the vector buffer in the capacity variable
00070                     capacity = gclib.GCmdI("MG _LMS");
00071                 } while (Load_Buffer(gclib, reader, capacity));
00072             }
00073 
00074             gclib.GCommand("VE"); // Segment End
00075             gclib.GMotionComplete("S");
00076 
00077             return GALIL_EXAMPLE_OK;
00078         }
00079 
00080         private static bool Load_Buffer(gclib gclib, StreamReader reader, int capacity)
00081         {
00082             string s_cmd;
00083             // Fully load the vector buffer leaving room for one VE command
00084             for (; capacity > 1; capacity--)
00085             {
00086                 // If there is another line of the text file
00087                 if ((s_cmd = reader.ReadLine()) != null)
00088                 {
00089                     // Run the command on each line of the text file
00090                     gclib.GCommand(s_cmd);
00091                 }
00092                 else
00093                     return false;
00094             }
00095 
00096             return true;
00097         }
00099     }
00100 }
```

10.97 vector_mode_example.cs File Reference

Data Structures

- class Vector_Mode_Example

  Takes a path to a file at the command line holding vector commands for the controller. The controller is placed into vector mode and commands are read from the file and sent to the controller.

10.97.1 Detailed Description

See Vector_Mode() for implementation of logic

For VB.NET, see definition in file vector_mode_example.vb
Definition in file vector_mode_example.cs.

10.98 vector_mode_example.cs

Go to the documentation of this file.

```
00001 
00010 using System;
00011 using System.Linq;
00012 
00013 namespace examples
```

```
00014 {
00025
00026     class Vector_Mode_Example
00027     {
00035         public static int Main(string[] args)
00036         {
00037             gclib gclib = new gclib();
00038             int rc = Examples.GALIL_EXAMPLE_OK;
00039
00040             try
00041             {
00042                 if (args.Count() != 2)
00043                 {
00044                     Console.WriteLine("Incorrect number of arguments provided");
00045                     Console.WriteLine("Usage: Vector_Example.exe <ADDRESS> <FILE>");
00046
00047                     Console.Write("\nPress any key to close the example");
00048                     Console.ReadKey();
00049                     return Examples.GALIL_EXAMPLE_ERROR;
00050                 }
00051
00052                 string file = args[1]; //Retrieve file from command line
00053                 string address = args[0]; //Retrieve address from command line
00054                 gclib.GOpen(address); //Open a connection at the provided address
00055
00056                 rc = Examples.Vector_Mode(gclib, file);
00057             }
00058             catch(Exception ex)
00059             {
00060                 Examples.PrintError(gclib, ex);
00061                 rc = Examples.GALIL_EXAMPLE_ERROR;
00062             }
00063             finally
00064             {
00065                 gclib.GClose();
00066             }
00067
00068             Console.Write("\nPress any key to close the example");
00069             Console.ReadKey();
00070             return rc;
00071         }
00072     }
00074 }
```

10.99 GclibTest.java File Reference

```
import gclibjava.GclibJava;
```

Data Structures

- class GclibTest

10.100 GclibTest.java

Go to the documentation of this file.

```
00001
00003 package gclibtest;
00004
00005 import gclibjava.GclibJava; //the java gclib wrapper
00006 import gclibjava.GclibJavaException; //the Galil exception class
00007 import java.util.ArrayList;
00008 import java.util.List;
00009
00013 public class GclibTest {
00014
00018     public static void main(String[] args) {
00019
00020         System.out.println("Java gclib Test");
00021         test();
00022         System.out.println("Done");
00023
00024     }
00025
00026     static void test()
00027     {
00028          GclibJava gclib = new GclibJava(); //our instance of the GclibJava class
00029
00030          try
```

```
00031         {
00032             System.out.println("gclib version " + gclib.GVersion());
00033 
00034             /*
00035             System.out.println("Hardware Requesting IP Addresses:");
00036             System.out.print(gclib.GIpRequests());
00037             System.out.println();
00038             */
00039 
00040             /*
00041             System.out.println("Available addresses:");
00042             System.out.print(gclib.GAddresses());
00043             System.out.println();
00044             */
00045 
00046             /*
00047             System.out.println("Assigning IP Address");
00048             gclib.GAssign("192.168.0.42", "00-50-4C-20-01-23");
00049             */
00050 
00051             System.out.println("Connecting");
00052             gclib.GOpen("192.168.0.42 -s ALL"); //connect to hardware, through gcaps
00053             //see http://galil.com/sw/pub/all/doc/gclib/html/gcaps.html
00054 
00055             System.out.println(gclib.GInfo());
00056 
00057             System.out.println("MG TIME: " + gclib.GCommand("MG TIME"));
00058 
00059             System.out.println("Downloading program");
00060             gclib.GProgramDownload("#A\rWT100\rMG\"Hello from program!\",TIME\rEN");
00061             //gclib.GProgramDownloadFile("c:\\temp\\test.dmc");
00062 
00063             System.out.println("Program now on hardware\n********************");
00064             System.out.print(gclib.GProgramUpload());
00065             //gclib.GProgramUploadFile("c:\\temp\\uploaded.dmc");
00066             System.out.println("\n********************");
00067 
00068             /*
00069             gclib.GTimeout((short) 10000); //adjust timeout for long command
00070             gclib.GCommand("BP"); //burn program
00071             gclib.GTimeout((short) -1); //use initial GOpen() timeout (--timeout)
00072             */
00073 
00074             System.out.println("Executing program");
00075             gclib.GCommand("XQ"); //execute program that was just downloaded
00076 
00077             System.out.println("Reading for a message");
00078             System.out.println(gclib.GMessage()); //print the message the program generated
00079 
00080             System.out.println("Downloading array data");
00081             List<Double> doubleList = new ArrayList();
00082             for (double i = 0; i < 10.0; ++i)
00083             {
00084                 doubleList.add(i);
00085             }
00086             gclib.GCommand("DA*[]"); //deallocate all array data
00087             gclib.GCommand("DM array[10]"); //dimension the array
00088             gclib.GArrayDownload("array", doubleList);
00089             //gclib.GArrayDownloadFile("c:\\temp\\arrays.csv");
00090 
00091             System.out.println("Uploading array data");
00092             List<Double> arrayUpload = gclib.GArrayUpload("array");
00093             //gclib.GArrayUploadFile("c:\\temp\\upload_arrays.csv", "array");
00094             for (Double d : arrayUpload)
00095             {
00096                 System.out.println(d.toString());
00097             }
00098 
00099             /*
00100             gclib.GCommand("UI 1"); //should evoke $F1 from products that
00101             //support interrupts. Look for the existance of UI in the hardware
00102             //command reference.
00103 
00104             System.out.println("Interrupt status byte: " +
00105                     String.format("%02X", gclib.GInterrupt()));
00106             */
00107 
00108             /*
00109             //loading firmware
00110             System.out.println("Firmware loader test");
00111             System.out.println("Currently loaded firmware: " + gclib.GCommand("\u0012\u0016"));
    //^R^V
00112             gclib.GFirmwareDownload("c:\\temp\\dmc-4000-r12h-cer.hex");
00113             System.out.println("Currently loaded firmware: " + gclib.GCommand("\u0012\u0016"));
    //^R^V
00114             gclib.GFirmwareDownload("c:\\temp\\dmc-4000-r12h.hex");
00115             System.out.println("Currently loaded firmware: " + gclib.GCommand("\u0012\u0016"));
```

```
      //^R^V
00116                 */
00117
00118           }
00119           catch (GclibJavaException e)
00120           {
00121                 System.out.println(Integer.toString(e.getErrorCode()) + " " + e.getMessage()); //print
      the message
00122           }
00123           finally
00124           {
00125                 gclib.GClose(); //must call close
00126           }
00127
00128     }//test()
00129
00130 }
00131
```

10.101 example.py File Reference

Functions

- main ()

10.101.1 Function Documentation

10.101.1.1 main()

```
main ( )
```

Definition at line 9 of file example.py.

10.102 example.py

Go to the documentation of this file.

```
00001 "'! \file example.py
00002 "'
00003
00004 import sys
00005 import string
00006 import gclib
00007
00008
00009 def main():
00010   g = gclib.py() #make an instance of the gclib python class
00011
00012   try:
00013     print('gclib version:', g.GVersion())
00014
00015
00018     "'
00019     #Get Ethernet controllers requesting IP addresses
00020     print('Listening for controllers requesting IP addresses...')
00021     ip_requests = g.GIpRequests()
00022     for id in ip_requests.keys():
00023       print(id, 'at mac', ip_requests[id])
00024
00025     #define a mapping of my hardware to ip addresses
00026     ips = {}
00027     ips['DMC4000-783'] = '192.168.0.42'
00028     ips['DMC4103-9998'] = '192.168.0.43'
00029
00030     for id in ips.keys():
00031       if id in ip_requests: #if our controller needs an IP
00032         print("\nAssigning", ips[id], "to", ip_requests[id])
00033         g.GAssign(ips[id], ip_requests[id]) #send the mapping
00034         g.GOpen(ips[id]) #connect to it
00035         print(g.GInfo())
00036         g.GCommand('BN') #burn the IP
00037         g.GClose() #disconnect
00038
00039     print('\nAvailable addresses:') #print ready connections
00040     available = g.GAddresses()
00041     for a in sorted(available.keys()):
00042       print(a, available[a])
00043
00044     print('\n')
00045     "'
```

```
    g.GOpen('192.168.0.42 -s ALL')
    #g.GOpen('COM1')
    print(g.GInfo())


    '''
    print('Programs')
    program =
'#A;i=0;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i
    #Program line above is too large to fit on any Galil controller, however it can easily fit if
broken up with level 4 compression.
    #the value of i at the end of code execution is 20.
    try:
      g.GProgramDownload(program, '--max 3') #program won't fit at level 3
    except gclib.GclibError as e:
      print(' GProgramDownload() correctly errored. Can\'t fit with level 3 compression')

    g.GProgramDownload(program, '')
    g.GProgramUploadFile('temp.dmc')
    g.GProgramDownload('','') #erase program
    g.GProgramDownloadFile('temp.dmc', '')
    print(' Uploaded program:\n%s' % g.GProgramUpload())
    g.GCommand('XQ') #execute the code
    g.GSleep(10) #wait a brief interval for the code to complete.
    if (float(g.GCommand('i=?')) == 20): #python can convert '20.0000' to float
      print(' Downloaded program verified')
    else:
      print(' Unexpected program result')

    #g.timeout = 5000 #set longish timeout
    #g.GCommand('BP') #burn program
    '''

    '''
    print('Arrays')
    g.GCommand('DA *,*[]')
    g.GCommand('DM A[5], B[10]')
    array_a = [1, 2, 3.14, 4, 5]
    array_b = [30, 42, 50, 60, 70]
    g.GArrayDownload('A', 0, -1, array_a)
    g.GArrayDownload('B', 2, -1, array_b)
    g.GArrayUploadFile('arrays.csv', ['A','B'])
    g.GArrayDownloadFile('arrays.csv')
    array_b_up = g.GArrayUpload('B', 3, 5)
    if array_b_up[0] == 42:
      print(' Array element verified')
    else:
      print(' Unexpected aray element', array_b_up[0])
    '''

    '''
    print('Messages')
    g.GProgramDownload('WT100\rMGTIME\rEN')
    g.GCommand('XQ')
    print(g.GMessage())
    '''

    '''
    firmware = r'c:\temp\dmc-4000-r12c.hex'
    #firmware = r'~/temp/dmc-4000-r12c.hex'
    print('Loading firmware', firmware)
    g.GFirmwareDownload(firmware)
    print(g.GInfo())
    '''

    '''
    #Motion Complete
    print('Motion Complete')
    c = g.GCommand #alias the command callable
    c('AB') #abort motion and program
    c('MO') #turn off all motors
    c('SHA') #servo A
    c('SPA=1000') #speead, 1000 cts/sec
    c('PRA=3000') #relative move, 3000 cts
    print(' Starting move...')
    c('BGA') #begin motion
    g.GMotionComplete('A')
    print(' done.')
    del c #delete the alias
    '''

  except gclib.GclibError as e:
    print('Unexpected GclibError:', e)

  finally:
    g.GClose() #don't forget to close connections!
```

```
00145
00146   return
00147
00148
00149 #runs main() if example.py called from the console
00150 if __name__ == '__main__':
00151   main()
```

10.103 Form1.vb File Reference

Data Structures

- class MainForm

  Demonstrates using gclib in a Windows Form, including using a second thread to free the GUI.

10.104 Form1.vb

Go to the documentation of this file.

```
00001 ''' <summary>
00002 ''' Demonstrates using gclib in a Windows Form, including using a second thread to free the GUI.
00003 ''' </summary>
00004 Public Class MainForm
00005
00006 #Region "UI"
00007
00008     'Runs when form loads
00009     Private Sub MainForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
00010         Print("Enter a FULL GOpen() address above and click Go", PrintStyle.Instruction)
00011         Print("NOTE: This demo will attempt to move Axis A", PrintStyle.Instruction)
00012     End Sub
00013
00014     'Opens Galil's help to show GOpen() options
00015     Private Sub HelpLabel_Click(sender As Object, e As EventArgs) Handles HelpLabel.Click
00016         'link to GOpen() documentation.
00017
      System.Diagnostics.Process.Start("http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h_aef4aec8a85630eed029b7a46aea7c
00018     End Sub
00019
00020     'Runs when user clicks Go button
00021     Private Sub GoButton_Click(sender As Object, e As EventArgs) Handles GoButton.Click
00022         If AddressTextBox.Text.Length = 0 Then
00023             Print("Enter a FULL GOpen() address above and click Go", PrintStyle.Instruction)
00024             Return
00025         End If
00026         RunDemo(AddressTextBox.Text)
00027     End Sub
00028
00029     'Various print styles.
00030     Private Enum PrintStyle
00031         Instruction
00032         Normal
00033         GalilData
00034         GclibData
00035         Err
00036     End Enum
00037
00038     ''' <summary>
00039     ''' Thread safe printing call.
00040     ''' </summary>
00041     ''' <param name="Message">The message to print</param>
00042     ''' <param name="Style">The style enum to indicate how to print</param>
00043     ''' <param name="SuppressCrLf">If true, the string will be printed without a trailing cr+lf</param>
00044     Private Sub Print(Message As String, Optional Style As PrintStyle = PrintStyle.Normal, Optional
      SuppressCrLf As Boolean = False)
00045         If Output.InvokeRequired Then 'tests if call is coming from another thread
00046             Output.Invoke(New Printer(AddressOf Print), Message, Style, SuppressCrLf) 'invoke the call
      on the correct thread
00047         Else 'we're on the right thread, just print
00048             Dim color As Color
00049             Select Case Style
00050                 Case PrintStyle.Instruction
00051                     color = Drawing.Color.Black
00052                 Case PrintStyle.GalilData
00053                     color = Drawing.Color.Green
00054                 Case PrintStyle.Normal
00055                     color = Drawing.Color.Blue
00056                 Case PrintStyle.Err
00057                     color = Drawing.Color.Red
00058                 Case PrintStyle.GclibData
00059                     color = Drawing.Color.Magenta
```

```
00060                 Case Else
00061                     color = Drawing.Color.Blue
00062             End Select
00063             Output.SelectionStart = Output.Text.Length
00064             Output.SelectionColor = color
00065             Output.AppendText(Message)
00066             If Not SuppressCrLf Then
00067                 Output.AppendText(vbCrLf)
00068             End If
00069         End If 'invoke check
00070     End Sub
00071 #End Region
00072
00073 #Region "Threading"
00074     ''' <summary>
00075     ''' Delegate used to print status when the status is generated in a thread other than the UI
      thread.
00076     ''' </summary>
00077     ''' <param name="Message">Message to print</param>
00078     ''' <param name="Style">Print Style</param>
00079     ''' <param name="SuppressCrLf">If true, the string will be printed without a trailing cr+lf</param>
00080     Private Delegate Sub Printer(Message As String, Style As PrintStyle, SuppressCrLf As Boolean)
00081
00082     ''' <summary>
00083     ''' Fires up the demo via the background worker thread
00084     ''' </summary>
00085     ''' <param name="address">The full GOpen() addresss</param>
00086     ''' <remarks>Runs in UI thread</remarks>
00087     Private Sub RunDemo(address As String)
00088         MainToolStrip.Enabled = False
00089         Output.Clear()
00090         GclibBackgroundWorker.RunWorkerAsync(address)
00091     End Sub
00092
00093     ''' <summary>
00094     ''' Runs in second thread to call the demo.
00095     ''' </summary>
00096     Private Sub GclibBackgroundWorker_DoWork(sender As Object, e As
      System.ComponentModel.DoWorkEventArgs) Handles GclibBackgroundWorker.DoWork
00097         Print("Running Demo with address " & e.Argument, PrintStyle.Normal)
00098         TheDemo(e.Argument) 'call the actual demo code
00099     End Sub
00100
00101     ''' <summary>
00102     ''' Runs in the main thread after the second thread returns.
00103     ''' </summary>
00104     Private Sub GclibBackgroundWorker_RunWorkerCompleted(sender As Object, e As
      System.ComponentModel.RunWorkerCompletedEventArgs) Handles GclibBackgroundWorker.RunWorkerCompleted
00105         Print("Demo thread done.", PrintStyle.Normal)
00106         MainToolStrip.Enabled = True
00107     End Sub
00108 #End Region
00109
00110 #Region "Demo Code"
00111
00112     ''' <summary>
00113     ''' Runs in a different thread than the UI, allowing the UI to stay active
00114     ''' </summary>
00115     ''' <param name="address">The full GOpen() addresss</param>
00116     Private Sub TheDemo(address As String)
00117
00118         Dim gclib As Gclib = Nothing 'keep gclib calls all in one thread.
00119         Try
00120             gclib = New Gclib 'constructor can throw, so keep it in a Try block
00121             Print("gclib version: ", PrintStyle.Normal, True)
00122             Print(gclib.GVersion, PrintStyle.GclibData)
00123
00124             '*** Uncomment below for network utilities ***
00125             'Print("Controllers requesting IP addresses...")
00126             'Dim macs As String() = gclib.GIpRequests()
00127             'If macs.Length = 0 Then
00128             '    Print("None")
00129             'Else
00130             '    For Each m As String In macs
00131             '        Print(m, PrintStyle.GclibData)
00132             '    Next
00133             'End If
00134
00135             'gclib.GAssign("192.168.0.42", "00:50:4c:20:01:23") 'Assign an IP to an unassigned
      controller
00136
00137             Print("Available connections:")
00138             Dim addrs As String() = gclib.GAddresses()
00139             If addrs.Length = 0 Then
00140                 Print("None")
00141             Else
00142                 For Each a As String In addrs
```

```
00143                   Print(a, PrintStyle.GclibData)
00144               Next
00145           End If
00146
00147           Print("Opening connection to """ & address & """... ", PrintStyle.Normal, True)
00148           gclib.GOpen(address)
00149           Print("Connected.", PrintStyle.Normal)
00150           Print(gclib.GInfo(), PrintStyle.GalilData)
00151
00152           'gclib.GCommand("BN") 'send BN if IP address was assigned above
00153
00154           Print("Sending ""MG TIME""", PrintStyle.Normal)
00155           Print(gclib.GCommand("MG TIME", False), PrintStyle.GalilData)
00156
00157           Print("Downloading Program... ", , True)
00158           gclib.GProgramDownload("i=0" & vbCr & "#A;MG i{N};i=i+1;WT10;JP#A,i<10;EN", "")
00159
00160           Print("Uploading Program")
00161           Print(gclib.GProgramUpload(), PrintStyle.GalilData)
00162
00163           Print("Blocking GMessage call")
00164           gclib.GCommand("XQ")
00165           System.Threading.Thread.Sleep(200) 'wait a bit to queue up some messages
00166           Print(gclib.GMessage(), PrintStyle.GalilData) 'get them all in one blocking read
00167
00168           Print("Downloading Program... ", , True)
00169           gclib.GProgramDownload("WT 1000; MG TIME; EN", "") 'prints a messsage after 1 second
00170
00171           Print("Uploading Program")
00172           Print(gclib.GProgramUpload(), PrintStyle.GalilData)
00173
00174           Print("Non-blocking GMessage call", , True)
00175           gclib.GCommand("XQ")
00176           gclib.GTimeout(0) 'set a zero timeout for a non-blocking read
00177           Dim msg As String = ""
00178           While (msg = "")
00179               msg = gclib.GMessage()
00180               Print(".", PrintStyle.Normal, True)
00181               System.Threading.Thread.Sleep(20) 'do something useful here...
00182           End While
00183           Print("Message: ", PrintStyle.Normal, True)
00184           Print(msg.Trim(), PrintStyle.GalilData)
00185           gclib.GTimeout(-1) 'put the timeout back
00186           'NOTE: Both GRecord and GInterrupt also have non-blocking mode with 0 timeout.
00187
00188           Print("Downloading Program... ", , True)
00189           gclib.GProgramDownload("WT 1000; UI 8; EN", "") 'fires an interrupt after 1 second
00190
00191           Print("Uploading Program")
00192           Print(gclib.GProgramUpload(), PrintStyle.GalilData)
00193
00194           Print("Non-blocking GInterrupt call", , True)
00195           gclib.GCommand("XQ")
00196           gclib.GTimeout(0) 'set a zero timeout for a non-blocking read
00197           Dim b As Byte = 0
00198           While (b = 0)
00199               b = gclib.GInterrupt()
00200               Print(".", PrintStyle.Normal, True)
00201               System.Threading.Thread.Sleep(20) 'do something useful here...
00202           End While
00203           Print("Byte: ", PrintStyle.Normal, True)
00204           Print(b.ToString("X02"), PrintStyle.GalilData)
00205           gclib.GTimeout(-1) 'put the timeout back
00206
00207           Print("Getting some synchronous data records")
00208           Dim DataRecord As Gclib.GDataRecord4000
00209           For i = 0 To 10
00210               DataRecord = gclib.GRecord(Of Gclib.GDataRecord4000)(False)
00211               Print(DataRecord.sample_number.ToString() & " ", PrintStyle.GalilData, True) 'byte 4
    and 5 are typically TIME counter
00212               'need help accessing the data record? Contact softwaresupport@galil.com
00213               System.Threading.Thread.Sleep(10)
00214           Next
00215           Print("")
00216
00217           Print("Getting some asynchronous data records")
00218           gclib.GRecordRate(10) 'set up data records every 10 ms
00219           For i = 0 To 10
00220               DataRecord = gclib.GRecord(Of Gclib.GDataRecord4000)(True)
00221               Print(DataRecord.sample_number.ToString() & " ", PrintStyle.GalilData, True) 'byte 4
    and 5 are typically TIME counter
00222               'no need to delay, asynchronous mode is dispatched by the Galil's RTOS.
00223           Next
00224           gclib.GRecordRate(0) 'turn off data records
00225           Print("")
00226
00227           Print("Downloading an array... ", , True)
```

```
00228             Dim array As New List(Of Double)
00229             For i As Double = 0 To 9
00230                 array.Add(i * 2)
00231             Next
00232             gclib.GCommand("DA *[];DM array[10]") 'arrays must be dimensioned prior to download
00233             gclib.GArrayDownload("array", array)
00234
00235             Print("Ok. Uploading array")
00236             array = gclib.GArrayUpload("array")
00237             For Each d As Double In array
00238                 Print(d.ToString("F4") & " ", PrintStyle.GalilData, True)
00239             Next
00240             Print("")
00241
00242             Print("Performing a write... ", , True)
00243             gclib.GWrite("QR" & vbCr) 'QR returns the binary data record
00244             Print("Ok. Reading binary data... ", , True)
00245             Dim data As Byte() = gclib.GRead()
00246             Print("Ok. Read " & data.Length() & " bytes.")
00247
00248             Print("Preparing A axis. This could cause errors if the axis is not initialized...", ,
    True)
00249             gclib.GCommand("AB;MO;SHA") 'compound commands are possible though typically not
    recommended
00250             Print("Ok")
00251             gclib.GCommand("PRA=5000")
00252             gclib.GCommand("SPA=5000")
00253             Print("Profiling a move on axis A... ", , True)
00254             gclib.GCommand("BGA")
00255             Print("Waiting for motion to complete... ", , True)
00256             gclib.GMotionComplete("A")
00257             Print("done")
00258             Print("Going back... ", , True)
00259             gclib.GCommand("PRA=-5000")
00260             gclib.GCommand("BGA")
00261             gclib.GMotionComplete("A")
00262             Print("done")
00263         Catch ex As Exception
00264             Print("Error: " & ex.Message, PrintStyle.Err)
00265         Finally
00266             If Not gclib Is Nothing Then
00267                 gclib.GClose() 'don't forget to close connections!
00268             End If
00269         End Try
00270     End Sub
00271 #End Region
00272
00273 End Class
```

10.105 Commands.vb File Reference

Functions

- int Commands (Gclib gclib)

  Demonstrates various uses of GCommand() And basic controller queries.

Variables

- partial Module Examples

10.106 Commands.vb

Go to the documentation of this file.

```
00001 Partial Public Module Examples
00002     ''' <summary>
00003     ''' Demonstrates various uses of GCommand() And basic controller queries.
00004     ''' </summary>
00005     ''' <param name="gclib">A gclib object with a valid connection.</param>
00006     ''' <returns>The success status Or error code of the function.</returns>
00007     ''' <remarks>See commands_example.cs for an example.</remarks>
00008     Function Commands(gclib As Gclib) As Integer
00009
      Console.WriteLine("********************************************************************************")
00010         Console.WriteLine("**********************    GCommand Trimmed example
    **********************")
00011
      Console.WriteLine("********************************************************************************")
00012
```

```
00013         Console.WriteLine("GCommand(""PR ?,? "", true) will return a trimmed response of GCommand()")
00014         Console.WriteLine("The command 'PR ?,?' will return the relative " +
00015                           "position of the A and B axes")
00016         Console.WriteLine("«PR ?,? with no trim: " + gclib.GCommand("PR ?,?", False) + "»")
00017         Console.WriteLine("«PR ?,? with trim: " + gclib.GCommand("PR ?,?", True) + "»")
00018
00019
      Console.WriteLine("********************************************************************************")
00020         Console.WriteLine("************************    GCommand Int example
      ************************")
00021
      Console.WriteLine("********************************************************************************")
00022
00023         Console.WriteLine("Use GCmdI() to retrieve the value of GCommand as an int.")
00024         Console.WriteLine("The command 'MG _LMS' will return the available " +
00025                           "space in the vector buffer of the S plane.")
00026
00027         Console.WriteLine("MG _LMS with GCmdI(): " + gclib.GCmdI("MG _LMS").ToString())
00028
00029
      Console.WriteLine("********************************************************************************")
00030         Console.WriteLine("***********************    GCommand Double example
      ***********************")
00031
      Console.WriteLine("********************************************************************************")
00032
00033         Console.WriteLine("Use GCmdD() to retrieve the value of GCommand as a double.")
00034         Console.WriteLine("The command 'MG @AN[1]' will return the value of Analog Input 1")
00035
00036         Console.WriteLine("MG @AN[1] with GCmdD(): " + gclib.GCmdD("MG @AN[1]").ToString())
00037
00038         Return Examples.GALIL_EXAMPLE_OK
00039     End Function
00040 End Module
```

10.107 Commands_Example.vb File Reference

Functions

- int Main ()

  Main function for the commands example.

10.108 Commands_Example.vb

Go to the documentation of this file.

```
00001 ''' <summary>
00002 ''' Demonstrates various uses of GCommand() And basic controller queries.
00003 ''' </summary>
00004 ''' <remarks>The first argument should be the IP Address of a Galil controller.</remarks>
00005 Module Commands_Example
00006
00007     ''' <summary>
00008     ''' Main function for the commands example.
00009     ''' </summary>
00010     ''' <returns>The success status Or error code of the function.</returns>
00011     ''' <remarks>The first argument should be the IP Address of a Galil controller.</remarks>
00012     Function Main() As Integer
00013         Dim rc As Integer = Examples.GALIL_EXAMPLE_OK
00014         Dim gclib As Gclib = New Gclib()
00015         Dim args() As String = Environment.GetCommandLineArgs()
00016         Try
00017             If args.Count() <> 2 Then
00018                 Console.WriteLine("Incorrect number of arguments provided")
00019                 Console.WriteLine("Usage: commands_example.exe <ADDRESS>")
00020
00021                 Console.Write(vbCrLf + "Press any key to close the example.")
00022                 Console.ReadKey()
00023                 Return Examples.GALIL_EXAMPLE_ERROR
00024             End If
00025
00026             Dim address As String = args(1)
00027
00028             gclib.GOpen(address)
00029
00030             rc = Examples.Commands(gclib)
00031         Catch ex As Exception
00032             Examples.PrintError(gclib, ex)
00033             rc = Examples.GALIL_EXAMPLE_ERROR
00034         Finally
00035             gclib.GClose()
```

```
00036         End Try
00037
00038         Console.Write(vbCrLf + "Press any key to close the example.")
00039         Console.ReadKey()
00040
00041         Return rc
00042     End Function
00043
00044 End Module
```

10.109 Contour.vb File Reference

Functions

- int **Contour** (Gclib gclib, string fileA, string fileB)

Variables

- positions_A As List< string >

10.110 Contour.vb

Go to the documentation of this file.

```
00001 Partial Public Module Examples
00002     Public Function Contour(gclib As Gclib, fileA As String, fileB As String) As Integer
00003         Record_Position(gclib, fileA, fileB) 'Record positional data on Axis A and B
00004
00005         Dim positions_A = System.IO.File.ReadAllText(fileA).Split(",").ToList()
00006         Dim positions_B = System.IO.File.ReadAllText(fileB).Split(",").ToList()
00007
00008         gclib.GCommand("SH AB") 'Set servo here
00009         gclib.GCommand("PA 0, 0") 'Set current position To 0
00010         gclib.GMotionComplete("AB") 'Wait For motion To complete
00011         gclib.GCommand("CM AB") 'Put axis A & B In contour mode
00012         gclib.GCommand("DT -1") 'Pauses contour mode To pre-load buffer
00013         gclib.GCommand("CD 0,0") 'Pre load buffer With zeros To prevent under buffering
00014         gclib.GCommand("CD 0,0") 'Pre load buffer With zeros To prevent under buffering
00015         gclib.GCommand("CD 0,0") 'Pre load buffer With zeros To prevent under buffering
00016         gclib.GCommand("DT 1") 'Sets the time interval For contour mode To be 2 samples
00017
00018         Dim capacity = 0 'Holds the capacity of the contour buffer
00019         Dim cmd = 0 'Holds the counter for which position to send next
00020
00021         If positions_A.Count <> positions_B.Count() Then
00022             Console.WriteLine("Error: The two datasets are not the same size")
00023             Return Examples.GALIL_EXAMPLE_ERROR
00024         End If
00025
00026         Do
00027             'Sleep while buffer is emptying
00028             System.Threading.Thread.Sleep(400)
00029
00030             'Stores the available space of the contour buffer in the capacity variable
00031             capacity = gclib.GCmdI("CM?")
00032         Loop While Load_Buffer(gclib, positions_A, positions_B, capacity, cmd)
00033
00034         gclib.GCommand("CD 0,0=0") 'End contour mode
00035
00036         Return Examples.GALIL_EXAMPLE_OK
00037
00038     End Function
00039
00040     Private Function Load_Buffer(gclib As Gclib, positions_A As List(Of String), positions_B As
      List(Of String),
00041                                  capacity As Integer, ByRef cmd As Integer)
00042         For i = capacity To 1 Step -1
00043             If cmd + 1 < positions_A.Count() Then
00044                 'Subtract previous position from new position to get how for of a move to make
00045                 Dim cdA = Double.Parse(positions_A(cmd + 1)) - Double.Parse(positions_A(cmd))
00046
00047                 'Subtract previous position from new position to get how for of a move to make
00048                 Dim cdB = Double.Parse(positions_B(cmd + 1)) - Double.Parse(positions_B(cmd))
00049
00050                 gclib.GCommand($"CD {cdA},{cdB}")
00051
00052                 cmd = cmd + 1
00053             Else
00054                 Return False
00055             End If
```

```
00056         Next
00057         Return True
00058     End Function
00059 End Module
```

10.111 Contour_Example.vb File Reference

Functions

- int Main ()

  Main function for the commands example.

10.112 Contour_Example.vb

Go to the documentation of this file.

```
00001 Module Contour_Example
00002     Function Main() As Integer
00003         Dim rc As Integer = Examples.GALIL_EXAMPLE_OK
00004         Dim gclib As Gclib = New Gclib()
00005         Dim args() As String = Environment.GetCommandLineArgs()
00006         Try
00007             If args.Count <> 4 Then
00008                 Console.WriteLine("Incorrect number of arguments provided")
00009                 Console.WriteLine("Usage: contour_example.exe <ADDRESS> <FILE A> <FILE B>")
00010 
00011                 Console.Write(vbCrLf + "Press any key to close the example.")
00012                 Console.ReadKey()
00013                 Return Examples.GALIL_EXAMPLE_ERROR
00014             End If
00015 
00016             Dim address As String = args(1) 'Retrieve address from command line
00017             Dim fileA As String = args(2) 'Retrieve filepath from command line
00018             Dim fileB As String = args(3) 'Retrieve filepath from command line
00019 
00020             gclib.GOpen(address) 'Open a connection at the provided address
00021 
00022             'Record user's training and play back training through contour mode
00023             rc = Examples.Contour(gclib, fileA, fileB)
00024         Catch ex As Exception
00025             Examples.PrintError(gclib, ex)
00026             rc = Examples.GALIL_EXAMPLE_ERROR
00027         Finally
00028             gclib.GClose()
00029         End Try
00030 
00031         Console.Write(vbCrLf + "Press any key to close the example.")
00032         Console.ReadKey()
00033 
00034         Return rc
00035     End Function
00036 End Module
```

10.113 Examples.vb File Reference

Functions

- void PrintError (Gclib gclib, Exception ex)

  Prints the exception to the console And queries the controller for the most recent error message.

Variables

- const int GALIL_EXAMPLE_OK = 0
- const GALIL_EXAMPLE_ERROR = -100

10.114 Examples.vb

Go to the documentation of this file.

```
00001 ''' <summary>
```

```
00002 ''' Provides a class of shared constants And methods for gclib's example projects.
00003 ''' </summary>
00004 Partial Public Module Examples
00005     Public Const GALIL_EXAMPLE_OK As Integer = 0
00006     Public Const GALIL_EXAMPLE_ERROR = -100
00007
00008     ''' <summary>
00009     ''' Prints the exception to the console And queries the controller for the most recent
00010     ''' error message.
00011     ''' </summary>
00012     ''' <param name="gclib">The gclib object from where the exception originated.</param>
00013     ''' <param name="ex">The exception object caught by the example.</param>
00014     ''' <remarks>See commands_example.cs for an example.</remarks>
00015     Public Sub PrintError(gclib As Gclib, ex As Exception)
00016         Console.WriteLine(ex.Message)
00017
00018         'If exception was Not a GOpen() exception, safe to query
00019         'the controller for a human readable error string
00020         If Not ex.Message.Contains("-1101") Then
00021             Console.WriteLine(gclib.GCommand("TC 1"))
00022         End If
00023     End Sub
00024 End Module
```

10.115 IP_Assigner.vb File Reference

Functions

- int **IP_Assigner** (Gclib gclib, string serial_num, byte address)

10.116 IP_Assigner.vb

Go to the documentation of this file.

```
00001 Partial Public Module Examples
00002     Public Function IP_Assigner(gclib As Gclib, serial_num As String, address As Byte) As Integer
00003         Dim controller_found As Boolean = False
00004         Dim requests() As String
00005
00006         Do 'Loop while no requests are found.
00007             Console.WriteLine("Searching...")
00008
00009             'Listen for ~5 secods for controllers requesting IP addresses.
00010             requests = gclib.GIpRequests()
00011
00012             For Each request In requests
00013                 Console.WriteLine(request)
00014             Next request
00015         Loop While (requests.Count() < 1)
00016
00017         For Each request In requests
00018             Dim controller_params() As String = request.Split({", "}, StringSplitOptions.None)
00019
00020             'Parameters are ordered as
00021             '[Model #], [Serial #], [MAC Address], [Connection Name], [IP Address]
00022
00023             If (controller_params.Count() < 5) Then
00024                 Console.WriteLine("Unexpected controller format")
00025                 Return GALIL_EXAMPLE_ERROR
00026             End If
00027
00028             Dim mac As String = controller_params(2)
00029             Dim ip As String = controller_params(4)
00030
00031             'If controller contains the user entered serial number
00032             If (serial_num = controller_params(1)) Then
00033                 Console.WriteLine("Controller Match Found")
00034                 controller_found = True
00035
00036                 'Splits the found ip address into individual bytes
00037                 Dim ip_bytes() As String = ip.Split(".")
00038
00039                 'Rebuild the ip address using the user provided address as the last byte
00040                 Dim new_ip = $"{ip_bytes(0)}.{ip_bytes(1)}.{ip_bytes(2)}.{address}"
00041
00042                 'Assign the New ip address to the controller
00043                 gclib.GAssign(new_ip, mac)
00044
00045                 'Open a connection at the New ip address
00046                 gclib.GOpen(new_ip)
00047
00048                 'Burns the newly assigned ip address to non-volatile EEPROM memory
```

```
00049                 gclib.GCommand("BN")
00050
00051                 Console.WriteLine("IP Address assigned")
00052
00053                 'Write the connection string to the console
00054                 Console.WriteLine(gclib.GInfo())
00055
00056                 Exit For
00057             End If
00058         Next request
00059
00060         If Not controller_found Then
00061             Console.Write("No controller matched the entered serial number")
00062         End If
00063
00064         Return GALIL_EXAMPLE_OK
00065
00066     End Function
00067 End Module
```

10.117 IP_Assigner_Example.vb File Reference

Functions

- int Main ()

  Main function for the commands example.

10.118 IP_Assigner_Example.vb

Go to the documentation of this file.

```
00001 Module IP_Assigner_Example
00002
00003     Function Main() As Integer
00004         Dim rc As Integer = Examples.GALIL_EXAMPLE_OK
00005         Dim gclib As Gclib = New Gclib()
00006         Dim args() As String = Environment.GetCommandLineArgs()
00007         Try
00008             If args.Count() <> 3 Then
00009                 Console.WriteLine("Incorrect number of arguments provided")
00010                 Console.WriteLine("Usage: ip_assigner_example.exe <SERIAL #> <1 Byte Address>")
00011
00012                 Console.Write(vbCrLf + "Press any key to close the example.")
00013                 Console.ReadKey()
00014                 Return Examples.GALIL_EXAMPLE_ERROR
00015             End If
00016
00017             Dim serial_num As String = args(1)
00018             Dim address As Byte
00019             Dim ok As Boolean = Byte.TryParse(args(2), address)
00020
00021             If Not ok Then
00022                 Console.WriteLine("Please enter a number between 0 and 255 for the address.\n" +
00023                     " This will be used as the last number in the IP Address\n" +
00024                     "Usage: ip_assigner_example.exe <SERIAL #> <1 Byte Address>")
00025
00026                 Console.Write(vbCrLf + "Press any key to close the example.")
00027                 Console.ReadKey()
00028                 Return Examples.GALIL_EXAMPLE_ERROR
00029             End If
00030
00031             rc = Examples.IP_Assigner(gclib, serial_num, address)
00032         Catch ex As Exception
00033             Examples.PrintError(gclib, ex)
00034             rc = Examples.GALIL_EXAMPLE_ERROR
00035         Finally
00036             gclib.GClose()
00037         End Try
00038
00039         Console.Write(vbCrLf + "Press any key to close the example.")
00040         Console.ReadKey()
00041
00042         Return rc
00043     End Function
00044
00045 End Module
```

10.119 Jog.vb File Reference

Functions

- int **Jog** (Gclib gclib)

10.120 Jog.vb

Go to the documentation of this file.

```
00001 Partial Public Module Examples
00002     Public Function Jog(gclib As Gclib) As Integer
00003         gclib.GCommand("ST")       ' Stop all motors
00004         gclib.GMotionComplete("A") ' Wait For motion To complete
00005         gclib.GCommand("SHA")      ' Set servo here
00006         gclib.GCommand("DPA=0")    ' Start position at absolute zero
00007         gclib.GCommand("JGA=0")    ' Start jogging With 0 speed
00008         gclib.GCommand("BGA")      ' Begin motion On A Axis
00009
00010         Dim isJogging As Boolean = True
00011         Dim speed As Integer = 0
00012
00013         Console.WriteLine("Enter a character on the keyboard to change the" +
00014                           " motor's speed:" + Environment.NewLine +
00015                           "<q> Quit" + Environment.NewLine +
00016                           "<a> -2000 counts/s" + Environment.NewLine +
00017                           "<s> -500  counts/s" + Environment.NewLine +
00018                           "<d> +500  counts/s" + Environment.NewLine +
00019                           "<f> +2000 counts/s" + Environment.NewLine +
00020                           "<r> Direction Reversal" + Environment.NewLine)
00021
00022         While isJogging
00023             gclib.GCommand("JGA=" + speed.ToString())
00024             Console.WriteLine("Jog Speed: " + speed.ToString())
00025
00026             Select Case Console.ReadKey(True).Key
00027                 Case ConsoleKey.Q
00028                     isJogging = False
00029                 Case ConsoleKey.A
00030                     speed -= 2000
00031                 Case ConsoleKey.S
00032                     speed -= 500
00033                 Case ConsoleKey.D
00034                     speed += 500
00035                 Case ConsoleKey.F
00036                     speed += 2000
00037                 Case ConsoleKey.R
00038                     speed *= -1
00039             End Select
00040         End While
00041
00042         gclib.GCommand("ST")
00043         gclib.GMotionComplete("A")
00044
00045         Return GALIL_EXAMPLE_OK
00046     End Function
00047 End Module
```

10.121 Jog_Example.vb File Reference

Functions

- int Main ()

  Main function for the commands example.

10.122 Jog_Example.vb

Go to the documentation of this file.

```
00001 '
00007 Module Jog_Example
00008
00009     Function Main() As Integer
00010         Dim rc As Integer = Examples.GALIL_EXAMPLE_OK
00011         Dim gclib As Gclib = New Gclib()
00012         Dim args() As String = Environment.GetCommandLineArgs()
00013         Try
```

```
00014             If args.Count() <> 2 Then
00015                 Console.WriteLine("Incorrect number of arguments provided")
00016                 Console.WriteLine("Usage: jog_example.exe <ADDRESS>")
00017 
00018                 Console.Write(vbCrLf + "Press any key to close the example.")
00019                 Console.ReadKey()
00020                 Return Examples.GALIL_EXAMPLE_ERROR
00021             End If
00022 
00023             Dim address As String = args(1)
00024 
00025             gclib.GOpen(address)
00026 
00027             rc = Examples.Jog(gclib)
00028         Catch ex As Exception
00029             Examples.PrintError(gclib, ex)
00030             rc = Examples.GALIL_EXAMPLE_ERROR
00031         Finally
00032             gclib.GClose()
00033         End Try
00034 
00035         Console.Write(vbCrLf + "Press any key to close the example.")
00036         Console.ReadKey()
00037 
00038         Return rc
00039     End Function
00040 
00041 End Module
```

10.123 Message.vb File Reference

Functions

- **Message** (Gclib gclib)

10.124 Message.vb

Go to the documentation of this file.

```
00001 Partial Public Module Examples
00002     Public Function Message(gclib As Gclib)
00003         Console.WriteLine("*****************************************************************")
00004         Console.WriteLine("Example GMessage() usage")
00005         Console.WriteLine("*****************************************************************")
00006 
00007         gclib.GCommand("TR0") 'Turn off trace
00008 
00009         'This program will force one message to appear as two separate packets.
00010         gclib.GProgramDownload("MG ""HELLO "" {N}" + vbCr +
00011                                "MG ""WORLD """ + vbCr +
00012                                "EN")
00013 
00014         gclib.GCommand("XQ") 'Begins execution Of program On controller
00015 
00016         Dim buf = ""
00017         Dim msg = ""
00018 
00019         'It Is important to note that a message can be too large to read in one
00020         'GMessage() call. Keep calling GMessage() while there are no errors to
00021         'get the full message.
00022 
00023         'While still receiving messages
00024         buf = gclib.GMessage()
00025         While buf <> ""
00026             For b = 0 To buf.Length - 1 'While message characters are in the buffer
00027 
00028                 msg += buf(b) 'Copy chars from  buffer To message
00029 
00030                 'If the message ends in "\r\n" it Is ready to be terminated
00031                 If (msg.Length > 2) AndAlso (msg(msg.Length - 1) = vbLf) AndAlso (msg(msg.Length - 2)
      = vbCr) Then
00032                     Console.WriteLine(msg)
00033                     msg = "" 'Reset message index
00034                     Exit While
00035                 End If
00036             Next
00037             buf = gclib.GMessage()
00038         End While
00039 
00040 
00041         'Downloads program to the controller
00042         gclib.GCommand("TR1") 'Turn On trace
```

```
00043         gclib.GProgramDownload(
00044                         "i=0" + vbCr +
00045                         "#A" + vbCr +
00046                         "MGi" + vbCr +
00047                         "i=i+1" + vbCr +
00048                         "WT100" + vbCr +
00049                         "JP#A,i<1" + vbCr +
00050                         "i=i/0" + vbCr +
00051                         "EN")
00052
00053         gclib.GCommand("XQ") 'Begins execution Of program On controller
00054
00055         'Lines returned by GMessage() can be one of three types
00056         '1) Standard Lines begin with a space (" ")
00057         '2) Crashed code begins with a question mark ("?")
00058         '3) Trace Lines begin with a line number ("1,6,15...")
00059
00060         'While still receiving messages
00061         buf = gclib.GMessage()
00062         While buf <> ""
00063             For b = 0 To buf.Length - 1 'While message characters are in the buffer
00064
00065                 msg += buf(b) 'Copy chars from buffer To message
00066
00067                 'If the message ends in "\r\n" its ready to be terminated
00068                 If (msg.Length > 2) AndAlso (msg(msg.Length - 1) = vbLf) AndAlso (msg(msg.Length - 2)
      = vbCr) Then
00069
00070                     If (msg(0) = " ") Then 'Standard Lines begin with a space (" ")
00071                         Console.Write("Standard Line: ")
00072                     ElseIf (msg(0) = "?") Then 'Crashed code begins with a question mark ("?")
00073                         Console.Write("Crashed Code: ")
00074                     Else 'Trace Lines begin with a line number ("1,6,15...")
00075                         Console.Write("Trace Line: ")
00076                     End If
00077                     Console.WriteLine(msg)
00078                     msg = ""
00079                 End If
00080             Next
00081             buf = gclib.GMessage()
00082         End While
00083
00084         Return Examples.GALIL_EXAMPLE_OK
00085     End Function
00086 End Module
```

10.125 Message_Example.vb File Reference

Functions

- int Main ()

  Main function for the commands example.

10.126 Message_Example.vb

Go to the documentation of this file.

```
00001 Module Message_Example
00002     Function Main() As Integer
00003         Dim rc As Integer = Examples.GALIL_EXAMPLE_OK
00004         Dim gclib As Gclib = New Gclib()
00005         Dim args() As String = Environment.GetCommandLineArgs()
00006         Try
00007             If args.Count <> 2 Then
00008                 Console.WriteLine("Incorrect number of arguments provided")
00009                 Console.WriteLine("Usage: message_example.exe <ADDRESS>")
00010
00011                 Console.Write(vbCrLf + "Press any key to close the example.")
00012                 Console.ReadKey()
00013                 Return Examples.GALIL_EXAMPLE_ERROR
00014             End If
00015
00016             Dim address As String = args(1) 'Retrieve address from command line
00017
00018             ' Opens a connection at the provided address
00019             ' The --subscribe MG addition Is needed to subscribe to
00020             ' controller messages
00021             gclib.GOpen(address + " --subscribe MG")
00022
00023             rc = Examples.Message(gclib)
00024         Catch ex As Exception
```

```
00025             Examples.PrintError(gclib, ex)
00026             rc = Examples.GALIL_EXAMPLE_ERROR
00027         Finally
00028             gclib.GClose()
00029         End Try
00030
00031         Console.Write(vbCrLf + "Press any key to close the example.")
00032         Console.ReadKey()
00033
00034         Return rc
00035     End Function
00036 End Module
```

10.127 Motion_Complete.vb File Reference

Functions

- **Motion_Complete** (Gclib gclib)

10.128 Motion_Complete.vb

Go to the documentation of this file.

```
00001 Partial Public Module Examples
00002     Public Function Motion_Complete(gclib As Gclib)
00003         Console.WriteLine("************************************************************")
00004         Console.WriteLine("Example GInterrupt() usage")
00005         Console.WriteLine("************************************************************")
00006
00007         'Simple check for appropriate communication bus
00008         'EI will fail below if interrupts are Not supported at all
00009         Dim ei_support = False
00010
00011         If gclib.GCommand("WH").Contains("IH") Then
00012             ei_support = True
00013         End If
00014
00015         If Not ei_support Then
00016             Console.WriteLine("No support on this bus")
00017             Return GALIL_EXAMPLE_ERROR
00018         End If
00019
00020         'Flush interrupts
00021         gclib.GCommand("EI0,0") 'Turn off interrupts
00022         gclib.GTimeout(0) 'Zero timeout
00023
00024         'Flush interrupts, status will be zero when queue Is empty
00025         While (gclib.GInterrupt() > 0)
00026         End While
00027
00028         gclib.GTimeout(-1) 'Restore timeout
00029
00030         'Independent Motion
00031         gclib.GCommand("DP 0,0") 'define position zero On A And B
00032         Console.WriteLine("Position: " + gclib.GCommand("RP")) 'Print reference position
00033
00034         gclib.GCommand("SP 4000,4000") 'Set up speed
00035         gclib.GCommand("AC 1280000, 1280000") 'acceleration
00036         gclib.GCommand("DC 1280000, 1280000") 'deceleration
00037         gclib.GCommand("PR 8000, 10000") 'Position Relative.  B will take longer To make its move.
00038         gclib.GCommand("SH AB") 'Servo Here
00039
00040         Console.WriteLine("Beginning independent motion...")
00041         gclib.GCommand("BG AB") 'Begin motion
00042         Check_Interrupts(gclib, "AB") 'Block until motion Is complete On axes A And B
00043         Console.WriteLine("Motion Complete on A and B")
00044         Console.WriteLine("Position: " + gclib.GCommand("RP")) 'Print reference position
00045
00046         Return GALIL_EXAMPLE_OK
00047     End Function
00048
00049     Private Sub Check_Interrupts(gclib As Gclib, axes As String)
00050         'bit mask of running axes, axes arg Is trusted to provide running axes.
00051         'Low bit indicates running.
00052         Dim axis_mask As Byte = &HFF
00053
00054         'iterate through all chars in axes to make the axis mask
00055         For i = 0 To axes.Length - 1
00056
00057             'support just A-H
00058             Select Case (axes(i))
00059                 Case "A"
```

```
00060                     axis_mask = axis_mask And &HFE
00061                     Exit Select
00062                 Case "B"
00063                     axis_mask = axis_mask And &HFD
00064                     Exit Select
00065                 Case "C"
00066                     axis_mask = axis_mask And &HFB
00067                     Exit Select
00068                 Case "D"
00069                     axis_mask = axis_mask And &HF7
00070                     Exit Select
00071                 Case "E"
00072                     axis_mask = axis_mask And &HEF
00073                     Exit Select
00074                 Case "F"
00075                     axis_mask = axis_mask And &HDF
00076                     Exit Select
00077                 Case "G"
00078                     axis_mask = axis_mask And &HBF
00079                     Exit Select
00080                 Case "H"
00081                     axis_mask = axis_mask And &H7F
00082                     Exit Select
00083             End Select
00084         Next
00085
00086         'send EI axis mask to set up interrupt events.
00087         gclib.GCommand("EI " + (Not axis_mask).ToString())
00088
00089         Dim status As Byte
00090
00091         While (axis_mask <> &HFF) 'wait for all interrupts to come in
00092             status = gclib.GInterrupt()
00093             Select Case (status)
00094                 Case &HD0 'Axis A complete
00095                     axis_mask = axis_mask Or &H1
00096                     Exit Select
00097                 Case &HD1 'Axis B complete
00098                     axis_mask = axis_mask Or &H2
00099                     Exit Select
00100                 Case &HD2 'Axis C complete
00101                     axis_mask = axis_mask Or &H4
00102                     Exit Select
00103                 Case &HD3 'Axis D complete
00104                     axis_mask = axis_mask Or &H8
00105                     Exit Select
00106                 Case &HD4 'Axis E complete
00107                     axis_mask = axis_mask Or &H10
00108                     Exit Select
00109                 Case &HD5 'Axis F complete
00110                     axis_mask = axis_mask Or &H20
00111                     Exit Select
00112                 Case &HD6 'Axis G complete
00113                     axis_mask = axis_mask Or &H40
00114                     Exit Select
00115                 Case &HD7 'Axis H complete
00116                     axis_mask = axis_mask Or &H80
00117                     Exit Select
00118             End Select
00119         End While
00120     End Sub
00121 End Module
```

10.129 Motion_Complete_Example.vb File Reference

Functions

- int Main ()

  Main function for the commands example.

10.130 Motion_Complete_Example.vb

Go to the documentation of this file.

```
00001 Module Motion_Complete_Example
00002     Function Main() As Integer
00003         Dim rc As Integer = Examples.GALIL_EXAMPLE_OK
00004         Dim gclib As Gclib = New Gclib()
00005         Dim args() As String = Environment.GetCommandLineArgs()
00006         Try
00007             If args.Count <> 2 Then
```

```
00008                 Console.WriteLine("Incorrect number of arguments provided")
00009                 Console.WriteLine("Usage: motion_complete_example.exe <ADDRESS>")
00010
00011                 Console.Write(vbCrLf + "Press any key to close the example.")
00012                 Console.ReadKey()
00013                 Return Examples.GALIL_EXAMPLE_ERROR
00014             End If
00015
00016             Dim address As String = args(1) 'Retrieve address from command line
00017
00018             ' Open a connection at the provided address and subcribe to event interrupts
00019             gclib.GOpen(address + " --subscribe EI")
00020
00021             rc = Examples.Motion_Complete(gclib)
00022         Catch ex As Exception
00023             Examples.PrintError(gclib, ex)
00024             rc = Examples.GALIL_EXAMPLE_ERROR
00025         Finally
00026             gclib.GClose()
00027         End Try
00028
00029         Console.Write(vbCrLf + "Press any key to close the example.")
00030         Console.ReadKey()
00031
00032         Return rc
00033     End Function
00034 End Module
```

10.131 Position_Tracking.vb File Reference

Functions

- **Position_Tracking** (Gclib gclib, int speed)

10.132 Position_Tracking.vb

Go to the documentation of this file.

```
00001 Partial Public Module Examples
00002     Public Function Position_Tracking(gclib As Gclib, speed As Integer)
00003         Dim acc = 100 * speed ' Set acceleration/deceleration to 100 times speed
00004
00005         gclib.GCommand("STA")      'Stop motor
00006         gclib.GMotionComplete("A") 'Wait For motion To complete
00007         gclib.GCommand("SHA")      'Set servo here
00008         gclib.GCommand("DPA=0")    'Start position at absolute zero
00009         gclib.GCommand("PTA=1")    'Start position tracking mode On A axis
00010
00011         gclib.GCommand("SPA=" + speed.ToString()) 'Set speed
00012         gclib.GCommand("ACA=" + acc.ToString())   'Set acceleration
00013         gclib.GCommand("DCA=" + acc.ToString())   'Set deceleration
00014
00015         Console.WriteLine("Begin Position Tracking with speed " + speed.ToString() +
00016                           ". Enter a non-number to exit.\n")
00017
00018         Dim position As Integer
00019
00020         'Loop asking user for New position.  End loop when user enters a non-number
00021         While True
00022             Console.WriteLine("Enter a new position:")
00023             Dim ok = Integer.TryParse(Console.ReadLine(), position)
00024
00025             If ok Then ' A Then valid position was provided
00026                 gclib.GCommand("PAA=" + position.ToString()) ' Go To New position
00027             Else 'A non-number was entered
00028                 Console.WriteLine("Position Tracking has exited")
00029                 Exit While
00030             End If
00031         End While
00032         gclib.GCommand("STA") 'Stop motor
00033         gclib.GMotionComplete("A") 'Wait For motion To complete
00034
00035         Return GALIL_EXAMPLE_OK
00036     End Function
00037 End Module
```

10.133 Position_Tracking_Example.vb File Reference

Functions

- int Main ()

  Main function for the commands example.

10.134 Position_Tracking_Example.vb

Go to the documentation of this file.

```
00001 Module Position_Tracking_Example
00002     Function Main() As Integer
00003         Dim rc As Integer = Examples.GALIL_EXAMPLE_OK
00004         Dim gclib As Gclib = New Gclib()
00005         Dim args() As String = Environment.GetCommandLineArgs()
00006         Dim speed As Integer = 0
00007         Try
00008             If args.Count = 2 Then
00009                 speed = 5000
00010             ElseIf args.Count = 3 Then
00011
00012                 Dim ok As Boolean = Byte.TryParse(args(2), speed)
00013
00014                 If Not ok Then
00015                     Console.WriteLine("An invalid speed was entered.  " +
00016                                       "Please enter a valid integer for speed.")
00017                     Console.WriteLine("Usage: position_tracking_example.exe <ADDRESS> <SPEED=5000>")
00018
00019                     Console.Write(vbCrLf + "Press any key to close the example.")
00020                     Console.ReadKey()
00021                     Return Examples.GALIL_EXAMPLE_ERROR
00022                 End If
00023
00024             Else
00025                 Console.WriteLine("Incorrect number of arguments provided")
00026                 Console.WriteLine("Usage: position_tracking_example.exe <ADDRESS> <SPEED=5000>")
00027
00028                 Console.Write(vbCrLf + "Press any key to close the example.")
00029                 Console.ReadKey()
00030                 Return Examples.GALIL_EXAMPLE_ERROR
00031             End If
00032
00033
00034             Dim address As String = args(1) 'Retrieve address from command line
00035             gclib.GOpen(address) 'Open a connection at the provided address
00036
00037             rc = Examples.Position_Tracking(gclib, speed) 'Begin position tracking mode
00038         Catch ex As Exception
00039             Examples.PrintError(gclib, ex)
00040             rc = Examples.GALIL_EXAMPLE_ERROR
00041         Finally
00042             gclib.GClose()
00043         End Try
00044
00045         Console.Write(vbCrLf + "Press any key to close the example.")
00046         Console.ReadKey()
00047
00048         Return rc
00049     End Function
00050 End Module
```

10.135 Record_Position.vb File Reference

Functions

- **Record_Position** (Gclib gclib, string fileA, string fileB)

10.136 Record_Position.vb

Go to the documentation of this file.

```
00001 Partial Public Module Examples
00002     Public Function Record_Position(gclib As Gclib, fileA As String, fileB As String)
00003         Dim writerA = New System.IO.StreamWriter(fileA, False)
00004         Dim writerB = New System.IO.StreamWriter(fileB, False)
00005
```

```
00006         Dim recording = 1
00007
00008         gclib.GProgramDownload(
00009             "RC 0;' Disable Recording" + vbCr +
00010             "DP 0, 0;' Set current position to 0" + vbCr +
00011             "DM posA[1000], posB[1000];' Define a new array that will hold positional data" + vbCr +
00012             "RA posA[], posB[];' Sets position array to be where recorded data will be stored" + vbCr
      +
00013             "RD _TPA, _TPB;' Defines Position to be the type of data that will be recorded" + vbCr +
00014             "RC 1,-1000;' Begins recording at 512Hz in continuous mode" + vbCr +
00015             "MO AB;' Turns motors off" + vbCr +
00016             "AI -1;' Waits for active low on Input 1" + vbCr +
00017             "RC 0;' Disable Recording after Input 1 goes low" + vbCr +
00018             "EN;' End program"
00019         )
00020
00021         gclib.GCommand("XQ")
00022
00023         Dim rd = 0
00024         Dim previous_rd = 0
00025         Dim leading_comma = False
00026
00027         Console.WriteLine("Begin recording")
00028
00029         Do
00030             System.Threading.Thread.Sleep(1000) 'Sleep while we wait for roughly half the array to be
      written
00031             rd = gclib.GCmdI("MG _RD") 'Gets address of next value in the position array
00032
00033             'Get values from posA[] array and write to file
00034             Write_Array_To_File(gclib, writerA, "posA", previous_rd, rd, leading_comma)
00035
00036             'Get values from posB[] array and write to file
00037             Write_Array_To_File(gclib, writerB, "posB", previous_rd, rd, leading_comma)
00038
00039             leading_comma = True
00040
00041             recording = gclib.GCmdI("MG _RC") 'Check status of RC
00042
00043             previous_rd = rd
00044
00045         Loop While recording > 0 'While recording is active
00046
00047         Console.WriteLine("End recording")
00048
00049         writerA.Close()
00050         writerB.Close()
00051
00052         Return Examples.GALIL_EXAMPLE_OK
00053     End Function
00054
00055     Private Sub Write_Array_To_File(gclib As Gclib, writer As System.IO.StreamWriter, array_name As
      String, previous_rd As Integer, rd As Integer, leading_comma As Boolean)
00056         Dim values = New List(Of Double)
00057
00058         If previous_rd < rd Then 'No array wrap around
00059             'Grab list of doubles from controller and add it to values
00060             values.AddRange(gclib.GArrayUpload(array_name, previous_rd, rd - 1))
00061         Else
00062             'Grab list of doubles from controller and add it to values
00063             values.AddRange(gclib.GArrayUpload(array_name, previous_rd, 999))
00064
00065             If rd <> 0 Then
00066                 'Grab list of doubles from controller and add it to values
00067                 values.AddRange(gclib.GArrayUpload(array_name, 0, rd - 1))
00068             End If
00069         End If
00070
00071         For i = 0 To values.Count() - 1
00072             If leading_comma Then
00073                 writer.Write(", ")
00074             End If
00075
00076             leading_comma = True
00077
00078             writer.Write(String.Format("{0:0.000}", values(i)))
00079         Next
00080     End Sub
00081 End Module
```

10.137 Record_Position_Example.vb File Reference

Functions

- int Main ()

  Main function for the commands example.

10.138 Record_Position_Example.vb

Go to the documentation of this file.

```
00001 Module Record_Position_Example
00002     Function Main() As Integer
00003         Dim rc As Integer = Examples.GALIL_EXAMPLE_OK
00004         Dim gclib As Gclib = New Gclib()
00005         Dim args() As String = Environment.GetCommandLineArgs()
00006         Try
00007             If args.Count <> 4 Then
00008                 Console.WriteLine("Incorrect number of arguments provided")
00009                 Console.WriteLine("Usage: record_position_example.exe <ADDRESS> <FILE A> <FILE B>")
00010
00011                 Console.Write(vbCrLf + "Press any key to close the example.")
00012                 Console.ReadKey()
00013                 Return Examples.GALIL_EXAMPLE_ERROR
00014             End If
00015
00016             Dim address As String = args(1) 'Retrieve address from command line
00017             Dim fileA As String = args(2) 'Retrieve filepath from command line
00018             Dim fileB As String = args(3) 'Retrieve filepath from command line
00019
00020             gclib.GOpen(address) 'Open a connection at the provided address
00021
00022             rc = Examples.Record_Position(gclib, fileA, fileB) 'Begin Recording Position
00023         Catch ex As Exception
00024             Examples.PrintError(gclib, ex)
00025             rc = Examples.GALIL_EXAMPLE_ERROR
00026         Finally
00027             gclib.GClose()
00028         End Try
00029
00030         Console.Write(vbCrLf + "Press any key to close the example.")
00031         Console.ReadKey()
00032
00033         Return rc
00034     End Function
00035 End Module
```

10.139 Remote_Client.vb File Reference

Functions

- int Remote_Client ()

  Demonstrates various uses of GListServers() and GSetServer()

10.140 Remote_Client.vb

Go to the documentation of this file.

```
00001 Partial Public Module Examples
00002     ''' <summary>
00003     ''' Demonstrates various uses of GListServers() and GSetServer()
00004     ''' </summary>
00005     ''' <returns>The success status Or error code of the function.</returns>
00006     ''' <remarks>See remote_client_example.cs for an example.</remarks>
00007     Function Remote_Client() As Integer
00008         Dim gclib = New Gclib()
00009
00010         Console.WriteLine("<s> List available servers on the network" + vbNewLine +
00011                           "<h> List available hardware on currently connected server" + vbNewLine +
00012                           "<0-9> Enter numbers 0-9 to connect to a server by index" + vbNewLine +
00013                           "<l> Set active server back to local server" + vbNewLine +
00014                           "<q> Quit")
00015
00016         Dim loop_bool = True
00017         Dim servers_list As String() = Array.Empty(Of String)
00018
```

```
00019         While loop_bool
00020             Dim input As Char = Console.ReadKey(True).KeyChar
00021 
00022             If input = "q" Then
00023                 loop_bool = False
00024             ElseIf input = "s" Then
00025                 Console.WriteLine("Available Servers:")
00026                 servers_list = gclib.GListServers()
00027                 Print_Servers_List(servers_list)
00028             ElseIf input >= "0" And input <= "9" Then
00029                 Dim index As Integer = Convert.ToInt16(input) - Convert.ToInt16("0"c)
00030                 If servers_list.Length > 0 And index < servers_list.Length Then
00031                     gclib.GSetServer(servers_list(index))
00032                     Console.WriteLine("Server set to: " + servers_list(index))
00033                 End If
00034             ElseIf input = "l" Then
00035                 gclib.GSetServer("Local")
00036                 Console.WriteLine("Server set to: Local")
00037             ElseIf input = "h" Then
00038                 Dim addresses As String() = gclib.GAddresses()
00039 
00040                 For Each address As String In addresses
00041                     Console.WriteLine(address)
00042                 Next
00043             End If
00044         End While
00045 
00046         Return Examples.GALIL_EXAMPLE_OK
00047     End Function
00048 
00049     Private Sub Print_Servers_List(servers_list As String())
00050         If servers_list.Length = 0 Then
00051             Console.WriteLine("none")
00052         Else
00053             For i As Integer = 0 To servers_list.Length - 1
00054                 Console.WriteLine("<" + i.ToString() + "> " + servers_list(i))
00055             Next
00056         End If
00057 
00058     End Sub
00059 End Module
```

10.141 Remote_Client_Example.vb File Reference

Functions

- int Main ()

  Main function for the commands example.

10.142 Remote_Client_Example.vb

Go to the documentation of this file.

```
00001 ''' <summary>
00002 ''' Demonstrates various uses of GListServers() and GSetServer().
00003 ''' </summary>
00004 ''' <remarks>This example requires no command line arguments.</remarks>
00005 Module Remote_Client_Example
00006 
00007     ''' <summary>
00008     ''' Main function for the Remote Client example.
00009     ''' </summary>
00010     ''' <returns>The success status Or error code of the function.</returns>
00011     ''' <remarks>This example requires no command line arguments.</remarks>
00012     Function Main() As Integer
00013         Dim rc As Integer = Examples.GALIL_EXAMPLE_OK
00014         Try
00015             rc = Examples.Remote_Client()
00016         Catch ex As Exception
00017             Console.WriteLine(ex.Message)
00018             rc = Examples.GALIL_EXAMPLE_ERROR
00019         End Try
00020 
00021         Console.Write(vbCrLf + "Press any key to close the example.")
00022         Console.ReadKey()
00023 
00024         Return rc
00025     End Function
00026 
00027 End Module
```

10.143 Remote_Server.vb File Reference

Functions

- int Remote_Server (string server_name)

  Demonstrates various uses of GPublishServer()

10.144 Remote_Server.vb

Go to the documentation of this file.

```
00001 Partial Public Module Examples
00002     ''' <summary>
00003     ''' Demonstrates various uses of GPublishServer()
00004     ''' </summary>
00005     ''' <param name="server_name">The name to publish the server under.</param>
00006     ''' <returns>The success status Or error code of the function.</returns>
00007     ''' <remarks>See remote_server_example.cs for an example.</remarks>
00008     Function Remote_Server(server_name As String) As Integer
00009         Dim gclib = New Gclib()
00010 
00011         Console.WriteLine("<p> Publish this server to the network" + vbNewLine +
00012                           "<r> Remove this server from the network" + vbNewLine +
00013                           "<q> Quit")
00014 
00015         Dim loop_bool = True
00016 
00017         While loop_bool
00018             Select Case Console.ReadKey(True).Key
00019                 Case ConsoleKey.Q
00020                     loop_bool = False
00021                 Case ConsoleKey.P
00022                     gclib.GPublishServer(server_name, True, False)
00023                     Console.WriteLine("Published Server")
00024                 Case ConsoleKey.R
00025                     gclib.GPublishServer(server_name, False, False)
00026                     Console.WriteLine("Removed Server")
00027             End Select
00028 
00029         End While
00030 
00031         Return Examples.GALIL_EXAMPLE_OK
00032     End Function
00033 End Module
```

10.145 Remote_Server_Example.vb File Reference

Functions

- int Main ()

  Main function for the commands example.

10.146 Remote_Server_Example.vb

Go to the documentation of this file.

```
00001 ''' <summary>
00002 ''' Demonstrates various uses of GPublishServer().
00003 ''' </summary>
00004 ''' <remarks>The first argument is optional and defines the name to publish the server under.</remarks>
00005 Module Remote_Server_Example
00006 
00007     ''' <summary>
00008     ''' Main function for the Remote Server example.
00009     ''' </summary>
00010     ''' <returns>The success status Or error code of the function.</returns>
00011     ''' <remarks>The first argument is optional and defines the name to publish the server
      under.</remarks>
00012     Function Main() As Integer
00013         Dim rc As Integer = Examples.GALIL_EXAMPLE_OK
00014         Dim args() As String = Environment.GetCommandLineArgs()
00015         Try
00016             Dim server_name As String
00017 
00018             If args.Count() <> 2 Then
00019                 Console.Write("Enter Server Name: ")
```

```
00020                server_name = Console.ReadLine()
00021            Else
00022                server_name = args(1)
00023            End If
00024
00025            rc = Examples.Remote_Server(server_name)
00026        Catch ex As Exception
00027            Console.WriteLine(ex.Message)
00028            rc = Examples.GALIL_EXAMPLE_ERROR
00029        End Try
00030
00031        Console.Write(vbCrLf + "Press any key to close the example.")
00032        Console.ReadKey()
00033
00034        Return rc
00035    End Function
00036
00037 End Module
```

10.147 Vector_Mode.vb File Reference

Functions

- int **Vector_Mode** (Gclib gclib, string file)

10.148 Vector_Mode.vb

Go to the documentation of this file.

```
00001 Partial Public Module Examples
00002     Public Function Vector_Mode(gclib As Gclib, file As String) As Integer
00003         gclib.GCommand("ST") 'Stop all motors
00004         gclib.GCommand("SH AB") 'Set servo here
00005         gclib.GCommand("DP 0,0") 'Start position at absolute zero
00006
00007         gclib.GCommand("CAS") 'Defines S As active coordinate system
00008         gclib.GCommand("VS 20000") 'Defines vector speed
00009         gclib.GCommand("VA 200000") 'Defines vector acceleration
00010         gclib.GCommand("VD 200000") 'Defines vector decelerlation
00011         gclib.GCommand("VM AB") 'Begin vector segment
00012
00013         Using reader As System.IO.StreamReader = New System.IO.StreamReader(file)
00014             'Stores the available space of the vector buffer in the capacity variable
00015             Dim capacity = gclib.GCmdI("MG _LMS")
00016             Load_Buffer(gclib, reader, capacity)
00017
00018             Console.WriteLine("Begin Motion")
00019             gclib.GCommand("BG S")
00020
00021             Do 'Load buffer with more commands
00022                 System.Threading.Thread.Sleep(100)
00023
00024                 'Stores the available space of the vector buffer in the capacity variable
00025                 capacity = gclib.GCmdI("MG _LMS")
00026             Loop While Load_Buffer(gclib, reader, capacity)
00027
00028             gclib.GCommand("VE") 'Segment End
00029             gclib.GMotionComplete("S")
00030             Console.WriteLine("Motion Complete")
00031
00032             Return GALIL_EXAMPLE_OK
00033         End Using
00034     End Function
00035
00036     Private Function Load_Buffer(gclib As Gclib, reader As System.IO.StreamReader, capacity As
      Integer) As Boolean
00037         Dim s_cmd As String = ""
00038         'Fully load the vector buffer leaving room for one VE command
00039         For i = capacity To 1 Step -1
00040             'If there Is another line of the text file
00041             If reader.Peek() >= 0 Then
00042                 s_cmd = reader.ReadLine()
00043                 'Run the command on each line of the text file
00044                 gclib.GCommand(s_cmd)
00045             Else
00046                 Return False
00047             End If
00048         Next
00049
00050         Return True
00051     End Function
00052 End Module
```

10.149 Vector_Mode_Example.vb File Reference

Functions

- int Main ()

  Main function for the commands example.

10.150 Vector_Mode_Example.vb

Go to the documentation of this file.

```
00001 Module Vector_Mode_Example
00002
00003     Function Main() As Integer
00004         Dim rc As Integer = Examples.GALIL_EXAMPLE_OK
00005         Dim gclib As Gclib = New Gclib()
00006         Dim args() As String = Environment.GetCommandLineArgs()
00007         Try
00008             If args.Count() <> 3 Then
00009                 Console.WriteLine("Incorrect number of arguments provided")
00010                 Console.WriteLine("Usage: vector_mode_example.exe <ADDRESS> <FILE>")
00011
00012                 Console.Write(vbCrLf + "Press any key to close the example.")
00013                 Console.ReadKey()
00014                 Return Examples.GALIL_EXAMPLE_ERROR
00015             End If
00016
00017             Dim address As String = args(1) 'Retrieve address from command line
00018             Dim file = args(2) 'Retrieve file from command line
00019
00020             gclib.GOpen(address)
00021
00022             rc = Examples.Vector_Mode(gclib, file)
00023         Catch ex As Exception
00024             Examples.PrintError(gclib, ex)
00025             rc = Examples.GALIL_EXAMPLE_ERROR
00026         Finally
00027             gclib.GClose()
00028         End Try
00029
00030         Console.Write(vbCrLf + "Press any key to close the example.")
00031         Console.ReadKey()
00032
00033         Return rc
00034     End Function
00035
00036 End Module
```

10.151 x_arrays.cpp File Reference

```
#include "x_examples.h"
```

Functions

- int x_arrays (GCon g)

  Example GArrayDownload() and GArrayUpload() usage.

10.151.1 Detailed Description

Example GArrayDownload() and GArrayUpload() usage.
Definition in file x_arrays.cpp.

10.151.2 Function Documentation

10.151.2.1 x_arrays()

```
int x_arrays (
            GCon g )
```

Example GArrayDownload() and GArrayUpload() usage.

Demonstrates array download and upload.
Definition at line 9 of file x_arrays.cpp.
References G_BOUNDS, G_COMMA, GArrayDownload(), GArrayDownloadFile(), GArrayUpload(), GArrayUploadFile(), GCmd(), vector(), and x_e().

10.152 x_arrays.cpp

Go to the documentation of this file.

```
00001
00007 #include "x_examples.h"
00008
00009 int x_arrays(GCon g)
00010 {
00011   cout « "\n*************************************************************************\n";
00012   cout « "Example GArrayDownload(), GArrayUploadFile()\n";
00013   cout « "GArrayDownloadFile(), and GArrayUpload usage\n";
00014   cout « "*************************************************************************\n";
00015
00016   char buf[1024]; //traffic buffer
00017
00018   //-------------------------------------------------------------------------
00019   //Preparing the array table
00020   x_e(GCmd(g, "DA *[]")); //deallocate all arrays
00021   x_e(GCmd(g, "DM A[10]")); //allocate A[] array
00022
00023   //-------------------------------------------------------------------------
00024   //download full array
00025   x_e(GArrayDownload(g, "A", G_BOUNDS, G_BOUNDS, "2,4,6,8,10,12,14,16,18,20"));
00026
00027   //-------------------------------------------------------------------------
00028   //upload full array
00029   x_e(GArrayUpload(g, "A", G_BOUNDS, G_BOUNDS, G_COMMA, buf, sizeof(buf)));
00030   cout « buf « "\n\n";
00031
00032   //-------------------------------------------------------------------------
00033   //download subset
00034   x_e(GArrayDownload(g, "A", 1, 3, "1,3,5"));
00035
00036   //-------------------------------------------------------------------------
00037   //upload full array
00038   x_e(GArrayUpload(g, "A", G_BOUNDS, G_BOUNDS, G_COMMA, buf, sizeof(buf)));
00039   cout « buf « "\n\n";
00040
00041   //-------------------------------------------------------------------------
00042   //upload subset
00043   x_e(GArrayUpload(g, "A", 2, 4, G_COMMA, buf, sizeof(buf)));
00044   cout « buf « '\n';
00045
00046   //-------------------------------------------------------------------------
00047   //upload to file
00048   x_e(GArrayUploadFile(g, "array.csv", 0));
00049
00050   //-------------------------------------------------------------------------
00051   //Download from file
00052   x_e(GArrayDownloadFile(g, "array.csv"));
00053
00054   //-------------------------------------------------------------------------
00055   //upload full array
00056   x_e(GArrayUpload(g, "A", G_BOUNDS, G_BOUNDS, G_COMMA, buf, sizeof(buf)));
00057   cout « buf « "\n\n";
00058
00059   return GALIL_EXAMPLE_OK;
00060 }
```

10.153 x_examples.cpp File Reference

```
#include "x_examples.h"
#include <iomanip>
```

Functions

- int main (int argc, char *argv[])

10.153.1 Detailed Description

Examples main(). Calls example code.
Definition in file x_examples.cpp.

10.153.2 Function Documentation

10.153.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Definition at line 10 of file x_examples.cpp.

10.154 x_examples.cpp

Go to the documentation of this file.

```
00001
00006 #include "x_examples.h"
00007
00008 #include <iomanip>
00009
00010 int main(int argc, char * argv[])
00011 {
00012   int rc = GALIL_EXAMPLE_OK; //return code
00013   int buf_size = G_SMALL_BUFFER;
00014   char buf[G_SMALL_BUFFER]; //traffic buffer
00015
00016   GCon g = 0; //var used to refer to a unique connection. A valid connection is nonzero.
00017   try
00018   {
00019
00020     x_e(GVersion(buf, sizeof(buf))); //library version
00021     cout « "Library versions: " « buf « "\n";
00022
00023 #if 0
00024     /*
00025     *  Change above line to "#if 1" to run examples below.
00026     *  Listening to requests for IP addresses.
00027     */
00028     x_e(GIpRequests(buf, sizeof(buf))); //listen for ~5 seconds for controllers requesting IP
      addresses
00029     cout « "Controllers without IP Address:\n";
00030     if (strlen(buf) != 0)
00031       cout « buf « "\n\n";
00032     else
00033       cout « "none\n\n";
00034
00035 #endif
00036
00037 #if 0
00038     /*
00039     *  Change above line to "#if 1" to run examples below.
00040     *  Assign IP addresses.
00041     */
00042     x_e(GAssign("192.168.42.100", "00:50:4C:20:52:90")); //assign an ip address to a known MAC
00043     //NOTE: GAssign does not burn the IP address with BN. This can be done after assignment through
      GOpen() and GCmd().
00044 #endif
00045
00046 #if 0
00047     /*
00048     *  Change above line to "#if 1" to run examples below.
00049     *  Listing available hardware addresses.
00050     */
00051     x_e(GAddresses(buf, sizeof(buf))); //list available addresses
00052     cout « "Available addresses:\n";
00053     if (strlen(buf) != 0)
00054       cout « buf « "\n\n";
00055     else
00056       cout « "none\n\n";
00057 #endif
00058
00059     cout « "Connecting to hardware\n";
00060
00061     //Basic connections
00062     x_e(GOpen("192.168.42.100 --subscribe ALL", &g)); //connect and assign a value to g.
00063     //x_e(GOpen("/dev/galilpci0 --subscribe ALL", &g));
00064     //x_e(GOpen("COM1 --baud 115200 --subscribe ALL", &g));
```

```
00065
00066     x_e(GInfo(g, buf, sizeof(buf))); //grab connection string
00067     cout « buf « '\n';
00068
00069     //x_e(GCmd(g, "BN")); //example to burn the IP address if it was set above
00070
00071 #if 0
00072     /*
00073     *  Change above line to "#if 1" to run examples below.
00074     *  Comment out the function calls below to be avoided.
00075     *  Note some calls attempt to move motors and not all
00076     *  functions are compatible with all Galil products.
00077     */
00078
00079
00080     x_e(x_gread_gwrite(g)); //call examples for GRead() and GWrite().
00081     x_e(x_gcommand(g)); //call examples for GCommand().
00082     x_e(x_programs(g)); //call examples for GProgramDownload() and GProgramUpload().
00083     x_e(x_arrays(g)); //call examples for GArrayDownload() and GArrayUpload().
00084     x_e(x_grecord(g)); //call examples for GRecord(). WARNING, this call will attempt to move motors.
00085     x_e(x_gmessage(g)); //call examples for GMessage().
00086     x_e(x_ginterrupt(g)); //call examples for GInterrupt().  WARNING, this call will attempt to move
      motors.
00087     x_e(x_gmotioncomplete(g)); //call examples for GMotionComplete. WARNING, this call will attempt to
      move motors.
00088     x_e(x_nonblocking(g)); //call examples for using GRecord(), GMessage(), and GInterrupt() in a
      non-blocking mode.
00089
00090
00091 #endif
00092
00093 #if 0
00094     /*
00095     *  Change above line to "#if 1" to run examples below.
00096     *  Loading Firmware
00097     */
00098     x_e(GFirmwareDownload(g, "c:/temp/d212r10r2.hex"));
00099     x_e(GInfo(g, buf, sizeof(buf)));
00100     cout « buf « '\n';
00101 #endif
00102
00103 #if 0
00104     /*
00105     *  Change above line to "#if 1" to run examples below.
00106     *  Calling GSetupDownloadFile
00107     */
00108     GOption opt = 0;
00109     const char* file_path = "C:/dev/test/gcb/test.gcb";
00110     opt = GSetupDownloadFile(g, file_path, opt, buf, buf_size);
00111     cout « "Setup file " « file_path « endl;
00112     cout « "Parameters " « (opt & 0x02 ? "present" : "absent") « endl;
00113     cout « "Variables " « (opt & 0x08 ? "present" : "absent") « endl;
00114     cout « "Arrays " « (opt & 0x10 ? "present" : "absent") « endl;
00115     cout « "Program " « (opt & 0x20 ? "present" : "absent") « endl;
00116     //cout « buf; //print the setup info
00117
00118     x_e(GSetupDownloadFile(g, "C:/dev/test/gcb/test.gcb", 0xff, 0, 0));
00119 #endif
00120
00121     if (g) x_e(GClose(g)); g = 0; //close g
00122
00123
00124   }//try
00125   catch (GReturn gr) //for x_e() function
00126   {
00127     if (gr == GALIL_EXAMPLE_ERROR)
00128       cout « "ERROR: Example code failed\n";
00129     else
00130     {
00131       cout « "Function returned " « gr « '\n';
00132       GError(gr, buf, sizeof(buf));
00133       cout « buf « '\n';
00134       GSize size = sizeof(buf);
00135
00136       if (g)
00137       {
00138         GUtility(g, G_UTIL_ERROR_CONTEXT, buf, &size);
00139         cout « buf « '\n'; //further context
00140       }
00141
00142     }
00143     rc = GALIL_EXAMPLE_ERROR;
00144     if (g) GClose(g); g = 0; //close g
00145   }
00146   catch (std::exception& e)
00147   {
00148     std::cerr « "Unexpected std::exception... Kaboom. " « e.what() « std::endl;
```

```
00149     rc = GALIL_EXAMPLE_ERROR;
00150     if (g) GClose(g); g = 0; //close g
00151   }
00152   catch (...)
00153   {
00154     cout « "Unexpected error... Kaboom." « endl;
00155     rc = GALIL_EXAMPLE_ERROR;
00156     if (g) GClose(g); g = 0; //close g
00157   }
00158
00159   if (argc == 1) //if no args on command line, report and pause
00160   {
00161     cout « endl « endl;
00162     if (rc == GALIL_EXAMPLE_OK)
00163       cout « "examples.cpp executed OK\n";
00164     else
00165       cout « "examples.cpp returning error " « rc « '\n';
00166
00167     cout « "main() is finished. Press Enter to exit:";
00168     getchar(); //keep window open
00169   }
00170   return rc;
00171 }
```

10.155 x_examples.h File Reference

```
#include <iostream>
#include <string>
#include <cstdio>
#include <cstring>
#include "gclib.h"
#include "gclibo.h"
```

Macros

- #define _CRT_SECURE_NO_WARNINGS
- #define GALIL_EXAMPLE_OK G_NO_ERROR
- #define GALIL_EXAMPLE_ERROR -100

Functions

- void x_e (GReturn rc)

  A trivial, C++ style return code check used in Galil's examples and demos.
- int x_gcommand (GCon g)

  Example GCommand() usage.
- int x_gmotioncomplete (GCon g)

  Example GMotionComplete() usage.
- int x_gread_gwrite (GCon g)

  Example GRead() and GWrite() usage.
- int x_programs (GCon g)

  Example GProgramDownload() and GProgramUpload() usage.
- int x_arrays (GCon g)

  Example GArrayDownload() and GArrayUpload() usage.
- int x_grecord (GCon g)

  Example GRecord() usage.
- int x_dr_motioncomplete (GCon g, GCStringIn axes)

  Example of MotionComplete with data records.
- int x_gmessage (GCon g)

  Example GMessage() usage.
- int x_ginterrupt (GCon g)

  Example GInterrupt() usage.

- int x_ei_motioncomplete (GCon g, GCStringIn axes)

  Example of MotionComplete with interrupts.
- int x_nonblocking (GCon g)

  Examples of using non-blocking operation of GRecord(), GInterrupt(), and GMessage().

10.155.1 Detailed Description

Header file for Galil gclib examples. All example functions start with the letter *e*. Example function calls demonstrating the use of library functions start with *x_*.
Definition in file x_examples.h.

10.155.2 Macro Definition Documentation

10.155.2.1 _CRT_SECURE_NO_WARNINGS

```
#define _CRT_SECURE_NO_WARNINGS
```
Definition at line 10 of file x_examples.h.

10.155.2.2 GALIL_EXAMPLE_ERROR

```
#define GALIL_EXAMPLE_ERROR -100
```
Definition at line 21 of file x_examples.h.

10.155.2.3 GALIL_EXAMPLE_OK

```
#define GALIL_EXAMPLE_OK G_NO_ERROR
```
Definition at line 20 of file x_examples.h.

10.155.3 Function Documentation

10.155.3.1 x_arrays()

```
int x_arrays (
            GCon g )
```
Example GArrayDownload() and GArrayUpload() usage.
Demonstrates array download and upload.
Definition at line 9 of file x_arrays.cpp.
References G_BOUNDS, G_COMMA, GArrayDownload(), GArrayDownloadFile(), GArrayUpload(), GArrayUploadFile(), GCmd(), vector(), and x_e().

10.155.3.2 x_dr_motioncomplete()

```
int x_dr_motioncomplete (
            GCon g,
            GCStringIn axes )
```
Example of MotionComplete with data records.
Uses axis status in data record to determine when motion has completed.

Parameters

g	Connection's handle.
axes	Mult-axis mask for determining motion complete. ABCDEFGH valid.

Definition at line 92 of file x_grecord.cpp.
References G_DR, GCmd(), GCmdI(), GRecord(), GRecordRate(), vector(), and x_e().
Referenced by x_grecord().

10.155.3.3 x_e()

```
void x_e (
            GReturn rc )  [inline]
```

A trivial, C++ style return code check used in Galil's examples and demos.
Throws GReturn if return value is not G_NO_ERROR. See x_examples.cpp for example usage and catch() handler.
Definition at line 30 of file x_examples.h.
References G_NO_ERROR, and vector().
Referenced by x_arrays(), x_dr_motioncomplete(), x_ei_motioncomplete(), x_gcommand(), x_ginterrupt(), x_gmessage(), x_gmotioncomplete(), x_gread_gwrite(), x_grecord(), x_nonblocking(), and x_programs().

10.155.3.4 x_ei_motioncomplete()

```
int x_ei_motioncomplete (
            GCon g,
            GCStringIn axes )
```

Example of MotionComplete with interrupts.
Uses motion complete status bytes to determine when motion has completed.

Parameters

g	Connection's handle.
axes	Mult-axis mask for determining motion complete. ABCDEFGH valid. Axes must be in motion when function is called.

Definition at line 85 of file x_ginterrupt.cpp.
References G_NO_ERROR, GCmd(), GInterrupt(), vector(), and x_e().
Referenced by x_ginterrupt().

10.155.3.5 x_gcommand()

```
int x_gcommand (
            GCon g )
```

Example GCommand() usage.
Examples of GCommand() and open-source wrappers like GCmd().
Definition at line 9 of file x_gcommand.cpp.
References G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND, G_NO_ERROR, G_USE_INITIAL_TIMEOUT, GCmd(), GCmdD(), GCmdI(), GCmdT(), GCommand(), GTimeout(), vector(), and x_e().

10.155.3.6 x_ginterrupt()

```
int x_ginterrupt (
            GCon g )
```

Example GInterrupt() usage.
Demonstrates retrieving status bytes via UI, and a MotionComplete function with interrupts.

Warning

This function will attempt to move motors.

Definition at line 9 of file x_ginterrupt.cpp.
References G_NO_ERROR, GCmd(), GCmdT(), GCommand(), GInterrupt(), GProgramDownload(), GTimeout(), vector(), x_e(), and x_ei_motioncomplete().

10.155.3.7 x_gmessage()

```
int x_gmessage (
            GCon g )
```

Example GMessage() usage.
Demonstrates retrieving messages.

Definition at line 9 of file x_gmessage.cpp.
References G_NO_ERROR, G_USE_INITIAL_TIMEOUT, GCmd(), GMessage(), GProgramDownload(), GTimeout(), vector(), and x_e().

10.155.3.8 x_gmotioncomplete()

```
int x_gmotioncomplete (
            GCon g )
```

Example GMotionComplete() usage.

Warning

This function will attempt to move motors.

Definition at line 9 of file x_gmotioncomplete.cpp.
References GCmd(), GCmdT(), GMotionComplete(), vector(), and x_e().

10.155.3.9 x_gread_gwrite()

```
int x_gread_gwrite (
            GCon g )
```

Example GRead() and GWrite() usage.
Demonstrates usage of Read/Write operations.
Definition at line 9 of file x_gread_gwrite.cpp.
References G_NO_ERROR, G_USE_INITIAL_TIMEOUT, GCmd(), GCmdI(), GRead(), GSleep(), GTimeout(), GWrite(), vector(), and x_e().

10.155.3.10 x_grecord()

```
int x_grecord (
            GCon g )
```

Example GRecord() usage.
Demonstrates QR and DR data records with struct names and pointer arithmetic. Also demonstrates a Motion↩ Complete function with data records.

Warning

This function will attempt to move motors.

Definition at line 10 of file x_grecord.cpp.
References G_DR, G_NO_ERROR, G_QR, GCmd(), GCmdI(), GCmdT(), GCommand(), GRecord(), GRecordRate(), vector(), x_dr_motioncomplete(), and x_e().

10.155.3.11 x_nonblocking()

```
int x_nonblocking (
            GCon g )
```

Examples of using non-blocking operation of GRecord(), GInterrupt(), and GMessage().
Typical usage of the asynchronous data streams is to call the function with a given timeout. The function will then block until the desired data is received of the timeout occurs. However, to check for available messages, ascynchronous records, or interrupts, the user can set a timeout of zero and the functions will return waiting data. See the source of this example for more detail.
Definition at line 22 of file x_nonblocking.cpp.
References G_DR, G_GCLIB_NON_BLOCKING_READ_EMPTY, GCmd(), GInterrupt(), GMessage(), GProgramDownload(), GRecord(), GRecordRate(), GTimeout(), vector(), and x_e().

10.155.3.12 x_programs()

```
int x_programs (
            GCon g )
```

Example GProgramDownload() and GProgramUpload() usage.

Demonstrates program download and upload including compression.
Definition at line 9 of file x_programs.cpp.
References G_NO_ERROR, G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT, GCmd(), GCmdI(), GProgramDownload(), GProgramDownloadFile(), GProgramUpload(), GProgramUploadFile(), GSleep(), vector(), and x_e().

10.156 x_examples.h

Go to the documentation of this file.

```
00001
00007 #ifndef I_7B02A40E_869B_4650_A5A8_859F0A6E3325
00008 #define I_7B02A40E_869B_4650_A5A8_859F0A6E3325
00009
00010 #define _CRT_SECURE_NO_WARNINGS //use traditional C calls like sprintf()
00011
00012 #include <iostream> //std::cout
00013 #include <string> //to_string, string, etc.
00014 #include <cstdio> //sprintf, etc.
00015 #include <cstring> //strlen, etc.
00016
00017 #include "gclib.h"
00018 #include "gclibo.h"
00019
00020 #define GALIL_EXAMPLE_OK G_NO_ERROR //return code for correct code execution
00021 #define GALIL_EXAMPLE_ERROR -100 //return code for error in example code
00022
00023 using namespace std;
00024
00026
00030 inline void x_e(GReturn rc)
00031 {
00032   if (rc != G_NO_ERROR)
00033     throw rc;
00034 }
00035
00037
00040 int x_gcommand(GCon g);
00041
00043
00046 int x_gmotioncomplete(GCon g);
00047
00049
00052 int x_gread_gwrite(GCon g);
00053
00055
00058 int x_programs(GCon g);
00059
00061
00064 int x_arrays(GCon g);
00065
00067
00073 int x_grecord(GCon g);
00074
00076
00082 int x_dr_motioncomplete(GCon g, GCStringIn axes);
00083
00085
00088 int x_gmessage(GCon g);
00089
00091
00096 int x_ginterrupt(GCon g);
00097
00099
00105 int x_ei_motioncomplete(GCon g, GCStringIn axes);
00106
00108
00115 int x_nonblocking(GCon g);
00116
00117
00118 #endif //I_7B02A40E_869B_4650_A5A8_859F0A6E3325
00119
00120
```

10.157 x_gcommand.cpp File Reference

```
#include "x_examples.h"
```

Functions

- int x_gcommand (GCon g)

  Example GCommand() usage.

10.157.1 Detailed Description

Example GCommand() usage.
Definition in file x_gcommand.cpp.

10.157.2 Function Documentation

10.157.2.1 x_gcommand()

```
int x_gcommand (
            GCon g )
```

Example GCommand() usage.
Examples of GCommand() and open-source wrappers like GCmd().
Definition at line 9 of file x_gcommand.cpp.
References G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND, G_NO_ERROR, G_USE_INITIAL_TIMEOUT, GCmd(), GCmdD(), GCmdI(), GCmdT(), GCommand(), GTimeout(), vector(), and x_e().

10.158 x_gcommand.cpp

Go to the documentation of this file.

```
00001
00007 #include "x_examples.h"
00008
00009 int x_gcommand(GCon g)
00010 {
00011   cout « "\n*******************************************************************************\n";
00012   cout « "Example GCommand() usage\n";
00013   cout « "*******************************************************************************\n";
00014
00015   char buf[1024]; //traffic buffer
00016   GSize read_bytes = 0; //bytes read in GCommand
00017
00018   //------------------------------------------------------------------------
00019   cout « "Revision report, ^R^V\n";
00020   x_e(GCommand(g, "\x12\x16", buf, sizeof(buf), &read_bytes));
00021   cout « buf « "\n\n";
00022
00023   //------------------------------------------------------------------------
00024   //Simple Command, no response accessible.
00025   x_e(GCmd(g, "ST"));
00026
00027   //------------------------------------------------------------------------
00028   cout « "Command Values\n";
00029   x_e(GCmd(g, "val=10")); //set initial value to an integer
00030   int vali = 0; //integer to be used
00031   x_e(GCmdI(g, "val=?", &vali)); //stuff value into integer
00032   cout « "val is " « vali « '\n';
00033   vali++;
00034   sprintf(buf, "val=%d", vali);
00035   x_e(GCmd(g, buf));
00036   x_e(GCmdI(g, "val=?", &vali));
00037   cout « "val is " « vali « '\n';
00038
00039   x_e(GCmd(g, "val=3.1415")); //set initial value to a decimal
00040   double vald = 0; //double to be used
00041   x_e(GCmdD(g, "val=?", &vald));
00042   cout « "val is " « vald « '\n';
00043   vald *= vald; //square
00044   sprintf(buf, "val=%f", vald);
00045   x_e(GCmd(g, buf));
00046   x_e(GCmdD(g, "val=?", &vald));
00047   cout « "val is " « vald « '\n';
00048
00049   //------------------------------------------------------------------------
00050   cout « "\nCommand Trimming\n";
00051   x_e(GCommand(g, "MGTIME", buf, sizeof(buf), &read_bytes)); //standard command call.
00052   cout « ">" « buf « "<" « '\n';
00053
00054   x_e(GCmdT(g, "MG TIME", buf, sizeof(buf), 0)); //Trim back.
00055   cout « ">" « buf « "<" « '\n';
```

```
00056
00057   char* front; //this must not be a pointer on the heap, it will be modified.
00058   x_e(GCmdT(g, "MG TIME", buf, sizeof(buf), &front)); //Trim back and front.
00059   cout « ">" « front « "<" « '\n';
00060
00061   //-------------------------------------------------------------------------
00062   cout « "\nReceiving Binary Data\n";
00063   x_e(GCommand(g, "QR", buf, sizeof(buf), &read_bytes)); //QR Response is binary
00064   cout « "QR read " « read_bytes « " bytes\n"; //Normally use GRecord() for QR.
00065
00066   //-------------------------------------------------------------------------
00067   cout « "\nError handling\n";
00068   GReturn rc = G_NO_ERROR;
00069   if ((rc = GCommand(g, "QDA[]", buf, sizeof(buf), &read_bytes)) ==
      G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND)
00070     cout « "QD correctly trapped, not allowed, try GArrayDownload()\n";
00071   else
00072   {
00073     cout « "Unexpected QD behaviour\n";
00074     return GALIL_EXAMPLE_ERROR;
00075   }
00076
00077   if ((rc = GCommand(g, "DL\rvar=3.14\rEN\r\\", buf, sizeof(buf), &read_bytes)) ==
      G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND)
00078     cout « "DL correctly trapped, not allowed, try GProgramDownload()\n";
00079   else
00080   {
00081     cout « "Unexpected DL behaviour\n";
00082     return GALIL_EXAMPLE_ERROR;
00083   }
00084
00085   //-------------------------------------------------------------------------
00086   cout « "\nModifying timeout\n";
00087   x_e(GTimeout(g, 10000)); //increase timeout for BP. 10 seconds is excessive, but sufficient for all
      products.
00088   cout « "Burning program...";
00089   x_e(GCommand(g, "BP", buf, sizeof(buf), &read_bytes));
00090   x_e(GTimeout(g, G_USE_INITIAL_TIMEOUT)); //restore timeout
00091   cout « "OK\n";
00092
00093   return GALIL_EXAMPLE_OK;
00094 }
```

10.159 x_ginterrupt.cpp File Reference

```
#include "x_examples.h"
```

Functions

- int x_ginterrupt (GCon g)

  Example GInterrupt() usage.
- int x_ei_motioncomplete (GCon g, GCStringIn axes)

  Example of MotionComplete with interrupts.

10.159.1 Detailed Description

Example GInterrupt() usage.
Definition in file x_ginterrupt.cpp.

10.159.2 Function Documentation

10.159.2.1 x_ei_motioncomplete()

```
int x_ei_motioncomplete (
            GCon g,
            GCStringIn axes )
```

Example of MotionComplete with interrupts.
Uses motion complete status bytes to determine when motion has completed.

Parameters

g	Connection's handle.
axes	Mult-axis mask for determining motion complete. ABCDEFGH valid. Axes must be in motion when function is called.

Definition at line 85 of file x_ginterrupt.cpp.
References G_NO_ERROR, GCmd(), GInterrupt(), vector(), and x_e().
Referenced by x_ginterrupt().

10.159.2.2 x_ginterrupt()

```
int x_ginterrupt (
            GCon g )
```

Example GInterrupt() usage.
Demonstrates retrieving status bytes via UI, and a MotionComplete function with interrupts.

Warning

This function will attempt to move motors.

Definition at line 9 of file x_ginterrupt.cpp.
References G_NO_ERROR, GCmd(), GCmdT(), GCommand(), GInterrupt(), GProgramDownload(), GTimeout(), vector(), x_e(), and x_ei_motioncomplete().

10.160 x_ginterrupt.cpp

Go to the documentation of this file.

```
00001
00007 #include "x_examples.h"
00008
00009 int x_ginterrupt(GCon g)
00010 {
00011   cout « "\n*********************************************************************\n";
00012   cout « "Example GInterrupt() usage\n";
00013   cout « "*********************************************************************\n";
00014
00015   char buf[1024]; //traffic buffer
00016   GSize bytes_read;
00017
00018   //---------------------------------------------------------------------
00019   //simple check for a interrupt support
00020   bool ei_support = true;
00021   if (GCommand(g, "WH", buf, 1024, &bytes_read) == G_NO_ERROR)
00022   {
00023     ei_support = (strstr(buf, "IH") != 0); //for this example, assume Ethernet connections supports
      interrupts.
00024   }//else assume PCI supports
00025
00026   if (!ei_support)
00027   {
00028     cout « "No support on this bus\n";
00029     return G_NO_ERROR;
00030   }
00031
00032
00033
00034   //---------------------------------------------------------------------
00035   //Flush interrupts
00036   x_e(GCmd(g, "EI0,0")); //turn off interrupts
00037   GStatus status;
00038   x_e(GTimeout(g, 0)); //zero timeout
00039   while ((GInterrupt(g, &status) == G_NO_ERROR) && status); //flush interrupts, status will be zero
      when queue is empty
00040   x_e(GTimeout(g, -1)); //restore timeout
00041
00042   //---------------------------------------------------------------------
00043   // User Interrupts, UI
00044   GTimeout(g, 1000); //set timeout to 1 sec
00045   x_e(GProgramDownload(g, "WT500\rUI8\rEN", 0));
00046
00047   x_e(GCmd(g, "XQ")); //execute program
00048
00049   x_e(GInterrupt(g, &status));
```

```
00050   GTimeout(g, -1); //restore timeout
00051
00052   if ((status & 0xF0) == 0xF0) //UI are F0 - FF
00053   {
00054     cout « "\"UI " « (int)(status & 0x0F) « "\" executed.\n";
00055   }
00056   else
00057   {
00058     cout « "Unexpected interrupt, " « hex « (int)status « dec « '\n';
00059     return GALIL_EXAMPLE_ERROR;
00060   }
00061
00062
00063 #if 0
00064   //-------------------------------------------------------------------------
00065   // Independent motion
00066   x_e(GCmd(g, "DP 0,0")); //define position zero on A and B
00067   x_e(GCmdT(g, "RP", buf, sizeof(buf), &trimmed));
00068   cout « "\nPosition: " « trimmed « '\n';
00069   x_e(GCmd(g, "SP 4000,4000")); //set up speed
00070   x_e(GCmd(g, "AC 1280000,1280000")); //acceleration
00071   x_e(GCmd(g, "DC 1280000,1280000")); //deceleration
00072   x_e(GCmd(g, "PR 8000,10000")); //Postion Relative. B will take longer to make its move.
00073   x_e(GCmd(g, "SH AB")); //Servo Here
00074   cout « "Beginning independent motion... ";
00075   x_e(GCmd(g, "BG AB")); //Begin motion
00076   x_e(x_ei_motioncomplete(g, "AB")); //Block until motion is complete on axes A and B
00077   cout « "Motion Complete on A and B\n";
00078   x_e(GCmdT(g, "RP", buf, sizeof(buf), &trimmed));
00079   cout « "Position: " « trimmed « '\n';;
00080 #endif
00081
00082   return GALIL_EXAMPLE_OK;
00083 }
00084
00085 int x_ei_motioncomplete(GCon g, GCStringIn axes) //Motion Complete with interrupts.
00086 {
00087   char buf[1024]; //traffic buffer
00088   GReturn rc;
00089   GStatus status;
00090   unsigned char axis_mask = 0xFF; //bit mask of running axes, axes arg is trusted to provide running
      axes. Low bit indicates running.
00091
00092   int len = strlen(axes);
00093   for (int i = 0; i < len; i++) //iterate through all chars in axes to make the axis mask
00094   {
00095     //support just A-H
00096     switch (axes[i])
00097     {
00098     case 'A':
00099       axis_mask &= 0xFE;
00100       break;
00101     case 'B':
00102       axis_mask &= 0xFD;
00103       break;
00104     case 'C':
00105       axis_mask &= 0xFB;
00106       break;
00107     case 'D':
00108       axis_mask &= 0xF7;
00109       break;
00110     case 'E':
00111       axis_mask &= 0xEF;
00112       break;
00113     case 'F':
00114       axis_mask &= 0xDF;
00115       break;
00116     case 'G':
00117       axis_mask &= 0xBF;
00118       break;
00119     case 'H':
00120       axis_mask &= 0x7F;
00121       break;
00122     }
00123   }
00124   sprintf(buf, "EI %u", (unsigned char) ~axis_mask);
00125   x_e(GCmd(g, buf)); //send EI axis mask to set up interrupt events.
00126
00127   while (axis_mask != 0xFF) //wait for all interrupts to come in
00128   {
00129     if ((rc = GInterrupt(g, &status)) == G_NO_ERROR)
00130     {
00131       switch (status)
00132       {
00133       case 0xD0: //Axis A complete
00134         axis_mask |= 0x01;
00135         break;
```

```
00136       case 0xD1: //Axis B complete
00137         axis_mask |= 0x02;
00138         break;
00139       case 0xD2: //Axis C complete
00140         axis_mask |= 0x04;
00141         break;
00142       case 0xD3: //Axis D complete
00143         axis_mask |= 0x08;
00144         break;
00145       case 0xD4: //Axis E complete
00146         axis_mask |= 0x10;
00147         break;
00148       case 0xD5: //Axis F complete
00149         axis_mask |= 0x20;
00150         break;
00151       case 0xD6: //Axis G complete
00152         axis_mask |= 0x40;
00153         break;
00154       case 0xD7: //Axis H complete
00155         axis_mask |= 0x80;
00156         break;
00157       }
00158     }
00159   }
00160
00161   return rc;
00162 }
```

10.161 x_gmessage.cpp File Reference

```
#include "x_examples.h"
```

Functions

- int x_gmessage (GCon g)

  Example GMessage() usage.

10.161.1 Detailed Description

Example GMessage() usage.
Definition in file x_gmessage.cpp.

10.161.2 Function Documentation

10.161.2.1 x_gmessage()

```
int x_gmessage (
            GCon g )
```

Example GMessage() usage.
Demonstrates retrieving messages.
Definition at line 9 of file x_gmessage.cpp.
References G_NO_ERROR, G_USE_INITIAL_TIMEOUT, GCmd(), GMessage(), GProgramDownload(), GTimeout(), vector(), and x_e().

10.162 x_gmessage.cpp

Go to the documentation of this file.

```
00001
00007 #include "x_examples.h"
00008
00009 int x_gmessage(GCon g)
00010 {
00011   cout << "\n*************************************************************************\n";
00012   cout << "Example GMessage() usage\n";
00013   cout << "*************************************************************************\n";
00014
00015   //-------------------------------------------------------------------------
00016   char buf[1024]; //traffic buffer
00017   x_e(GProgramDownload(g, "i=0\r#A\rMGi\ri=i+1\rWT100\rJP#A,i<10\rEN", 0));
```

```
00018   x_e(GCmd(g, "XQ"));
00019
00020   int rc;
00021   GTimeout(g, 200);//adjust timeout
00022   int i = 0;
00023   while ((rc = GMessage(g, buf, sizeof(buf))) == G_NO_ERROR)
00024   {
00025     cout « buf;
00026     if (strstr(buf, ".") != 0) //each MG has a decimal point
00027       i++; //count it
00028   }
00029   GTimeout(g, G_USE_INITIAL_TIMEOUT); //restore timeout
00030   if (i == 10)
00031     return GALIL_EXAMPLE_OK;
00032   else
00033   {
00034     cout « "Expected 10 messages\n";
00035     return GALIL_EXAMPLE_ERROR;
00036   }
00037 }
```

10.163 x_gmotioncomplete.cpp File Reference

```
#include "x_examples.h"
```

Functions

- int x_gmotioncomplete (GCon g)

  Example GMotionComplete() usage.

10.163.1 Detailed Description

Example GMotionComplete() usage.
Definition in file x_gmotioncomplete.cpp.

10.163.2 Function Documentation

10.163.2.1 x_gmotioncomplete()

```
int x_gmotioncomplete (
            GCon g )
```

Example GMotionComplete() usage.

Warning

This function will attempt to move motors.

Definition at line 9 of file x_gmotioncomplete.cpp.
References GCmd(), GCmdT(), GMotionComplete(), vector(), and x_e().

10.164 x_gmotioncomplete.cpp

Go to the documentation of this file.

```
00001
00007 #include "x_examples.h"
00008
00009 int x_gmotioncomplete(GCon g)
00010 {
00011   cout « "\n*************************************************************************\n";
00012   cout « "Example GMotionComplete() usage\n";
00013   cout « "*************************************************************************\n";
00014
00015   char buf[1024]; //traffic buffer
00016   char* trimmed; //trimmed string pointer
00017
00018   x_e(GCmd(g, "ST")); //stop all motion and programs
00019
00020   //-------------------------------------------------------------------
00021   // Independent motion
```

```
00022   x_e(GCmd(g, "DP 0")); //define A position zero
00023   x_e(GCmdT(g, "RP", buf, sizeof(buf), &trimmed));
00024   cout « "\nPosition: " « trimmed « '\n';
00025   x_e(GCmd(g, "SP 4000")); //set up speed
00026   x_e(GCmd(g, "AC 1280000")); //acceleration
00027   x_e(GCmd(g, "DC 1280000")); //deceleration
00028   x_e(GCmd(g, "PR 8000")); //Postion Relative.
00029   x_e(GCmd(g, "SH A")); //Servo Here
00030   cout « "Beginning independent motion... ";
00031   x_e(GCmd(g, "BG A")); //Begin motion
00032   x_e(GMotionComplete(g, "A")); //Block until motion is complete on axis A
00033   cout « "Motion Complete on A\n";
00034   x_e(GCmdT(g, "RP", buf, sizeof(buf), &trimmed));
00035   cout « "Position: " « trimmed « '\n';;
00036
00037   //---------------------------------------------------------------------
00038   // Vector motion
00039   x_e(GCmd(g, "DP 0")); //define position zero on A
00040   x_e(GCmdT(g, "RP", buf, sizeof(buf), &trimmed));
00041   cout « "\nPosition: " « trimmed « '\n';;
00042   x_e(GCmd(g, "VS 2000"));  //set up vector speed, S plane
00043   x_e(GCmd(g, "VA 100000")); //vector Acceleration
00044   x_e(GCmd(g, "VD 100000")); //vector deceleration
00045   x_e(GCmd(g, "VM AN")); //invoke vector mode, use virtual axis for 1-axis controllers
00046   x_e(GCmd(g, "VP 3000, 3000")); //buffer Vector Position
00047   x_e(GCmd(g, "VP 6000,0")); //buffer Vector Position
00048   x_e(GCmd(g, "VE")); //indicate Vector End
00049   cout « "Beginning vector motion... ";
00050   x_e(GCmd(g, "BG S")); //begin S plane
00051   x_e(GMotionComplete(g, "S")); //Block until motion is complete on vector plane S
00052   cout « "Motion Complete on vector plane S\n";
00053   x_e(GCmdT(g, "RP", buf, sizeof(buf), &trimmed));
00054   cout « "Position: " « trimmed « '\n';;
00055
00056   return GALIL_EXAMPLE_OK;
00057 }
```

10.165 x_gread_gwrite.cpp File Reference

```
#include "x_examples.h"
```

Functions

- int x_gread_gwrite (GCon g)

  Example GRead() and GWrite() usage.

10.165.1 Detailed Description

Example GRead() and GWrite() usage.
Definition in file x_gread_gwrite.cpp.

10.165.2 Function Documentation

10.165.2.1 x_gread_gwrite()

```
int x_gread_gwrite (
            GCon g )
```

Example GRead() and GWrite() usage.
Demonstrates usage of Read/Write operations.
Definition at line 9 of file x_gread_gwrite.cpp.
References G_NO_ERROR, G_USE_INITIAL_TIMEOUT, GCmd(), GCmdI(), GRead(), GSleep(), GTimeout(), GWrite(), vector(), and x_e().

10.166 x_gread_gwrite.cpp

Go to the documentation of this file.

```
00001
00007 #include "x_examples.h"
00008
```

```
00009 int x_gread_gwrite(GCon g)
00010 {
00011   cout « "\n*********************************************************************\n";
00012   cout « "Example GRead() and GWrite() usage\n";
00013   cout « "*********************************************************************\n";
00014
00015   char buf[1024]; //traffic buffer
00016   GSize read_bytes = 0; //bytes read
00017   GSize total_bytes = 0; //total bytes read
00018
00019   //-------------------------------------------------------------------
00020   /*
00021   *  Ad hoc program downlader
00022   *
00023   *  GProgramDownload() should be used for all DMC program downloads. DL is used here to demonstrate
00024   *  GRead()/GWrite() usage.
00025   *
00026   *  GRead()/GWrite() are useful, for example, for a firmware NRE that provides a binary response, or
      for a command
00027   *  that does not follow the colon termination of typical Galil commands. DL is an example of such a
      command.
00028   *
00029   */
00030
00031   string program = "i=0\r#loop\ri=i+1\rWT10\rJP#loop,i<10,\rEN\r"; //simple program for download
00032
00033   x_e(GCmd(g, "AB")); //Abort any running programs
00034
00035   x_e(GWrite(g, "DL\r", 3)); //Write the DL command to the controller to access the program buffer.
      The controller does not respond to this command.
00036   //don't forget the carriage return when using GWrite() for commands.
00037
00038   //GSleep(100); //some products need a delay to erase flash. 100ms is overkill, but sufficient.
00039
00040   x_e(GWrite(g, program.c_str(), program.size())); //send the program to the program buffer
00041   x_e(GWrite(g, "\\", 1)); //send a backslash to exit the program buffer.
00042
00043
00044   /*
00045   *  Assuming the return format of DL is unknown, this demo reads until a read times out.
00046   *  A faster approach would be to read for a known terminating sequence that doesn't appear in the
      data.
00047   *  For example, standard Galil commands terminate data with a colon.
00048   */
00049   x_e(GTimeout(g, 100)); //adjust timeout
00050   GReturn rc = G_NO_ERROR; //return code
00051   while (rc == G_NO_ERROR) //read until timeout
00052   {
00053     total_bytes += read_bytes;
00054     rc = GRead(g, buf, sizeof(buf), &read_bytes);
00055   }
00056
00057   x_e(GTimeout(g, G_USE_INITIAL_TIMEOUT)); //restore timeout
00058   cout « "\nRead " « total_bytes « " byte(s)\n";
00059   cout.write(buf, total_bytes); //print the received data
00060   cout « "\n";
00061
00062   //now test the downloaded program
00063   x_e(GCmd(g, "i=0")); //force i to zero
00064   x_e(GCmd(g, "XQ")); //execute the program
00065   GSleep(200); //loop should take about 100ms
00066
00067   int i;
00068   x_e(GCmdI(g, "i=?", &i));// pull the controller's value of i into an int
00069   if (i == 10)
00070   {
00071     cout « "Program test OK.\n";
00072     return GALIL_EXAMPLE_OK;
00073   }
00074   else
00075   {
00076     cout « "Program test failed. " « i « "\n";
00077     return GALIL_EXAMPLE_ERROR;
00078   }
00079
00080
00081
00082
00083 }
```

10.167 x_grecord.cpp File Reference

```
#include "x_examples.h"
```

Functions

- int x_grecord (GCon g)

  Example GRecord() usage.
- int x_dr_motioncomplete (GCon g, GCStringIn axes)

  Example of MotionComplete with data records.

10.167.1 Detailed Description

Example GRecord() usage.
Definition in file x_grecord.cpp.

10.167.2 Function Documentation

10.167.2.1 x_dr_motioncomplete()

```
int x_dr_motioncomplete (
            GCon g,
            GCStringIn axes )
```

Example of MotionComplete with data records.
Uses axis status in data record to determine when motion has completed.

Parameters

g	Connection's handle.
axes	Mult-axis mask for determining motion complete. ABCDEFGH valid.

Definition at line 92 of file x_grecord.cpp.
References G_DR, GCmd(), GCmdI(), GRecord(), GRecordRate(), vector(), and x_e().
Referenced by x_grecord().

10.167.2.2 x_grecord()

```
int x_grecord (
            GCon g )
```

Example GRecord() usage.
Demonstrates QR and DR data records with struct names and pointer arithmetic. Also demonstrates a Motion↩ Complete function with data records.

Warning

This function will attempt to move motors.

Definition at line 10 of file x_grecord.cpp.
References G_DR, G_NO_ERROR, G_QR, GCmd(), GCmdI(), GCmdT(), GCommand(), GRecord(), GRecordRate(), vector(), x_dr_motioncomplete(), and x_e().

10.168 x_grecord.cpp

Go to the documentation of this file.

```
00001
00008 #include "x_examples.h"
00009
00010 int x_grecord(GCon g)
00011 {
00012   cout « "\n*************************************************************************\n";
00013   cout « "Example GRecord() usage\n";
00014   cout « "*************************************************************************\n";
00015
00016   char buf[1024]; //traffic buffer
00017   GSize bytes_read;
00018
00019   //------------------------------------------------------------------------
```

```
00020   //Grab a data record via QR and pull out some values.
00021   cout « "\nQR-based data record\n";
00022   GDataRecord r; //user's data record union.
00023   x_e(GRecord(g, &r, G_QR)); //poll for data record.
00024   cout « r.dmc4000.sample_number « '\n'; //sample number
00025   cout « r.dmc4000.axis_a_reference_position « "\n"; //Axis A's reference, "RPA"
00026
00027   //-------------------------------------------------------------------------
00028   //simple check for a serial connection.
00029   bool dr_support = true;
00030   if (GCommand(g, "WH", buf, 1024, &bytes_read) == G_NO_ERROR)
00031   {
00032     dr_support = (strstr(buf, "IH") != 0); //for this example, assume Ethernet connections supports
      DR.
00033   }//else assume PCI supports, note 18x2 doesn't
00034
00035
00036   if (dr_support)
00037   {
00038     //-------------------------------------------------------------------------
00039     //Read data records asynchronously for a given interval.
00040     //note -s DR must have been specified in GOpen()
00041     cout « "\nDR-based data record\n";
00042     unsigned short time = 0; //controller's sample time, 1 tick a ms with TM 1000.
00043     unsigned short deadline = 1000; //listen time, in ms, at TM 1000
00044     int original_dr;
00045     x_e(GCmdI(g, "MG_DR", &original_dr)); //grab the current DR value
00046     x_e(GRecordRate(g, 100)); //set up DR to 10Hz
00047     for (size_t i = 0; deadline > time; i++)
00048     {
00049       x_e(GRecord(g, &r, G_DR)); //get record
00050       time = r.dmc4000.sample_number; //pull out the desired value
00051       //time = r.dmc1806.sample_number; //note, these are product-specific
00052       cout « time « '\n'; //print time
00053       if (!i) deadline = time + deadline; //sample on first iteration to set deadline
00054     }
00055
00056     sprintf(buf, "DR %d", original_dr);
00057     x_e(GCmd(g, buf)); //restore DR
00058   }
00059
00060   //-------------------------------------------------------------------------
00061   //Use pointer arithmetic to pull out arbitrary types and offsets.
00062   cout « "\nQR-based data record with offsets\n";
00063   x_e(GRecord(g, &r, G_QR)); //poll for data record.
00064   cout « r.dmc4000.sample_number « '\n';
00065   cout « *((unsigned short*)(r.byte_array + 4)) « '\n'; //equivalent to sample_number for DMC-4000
00066
00067
00068 #if 0
00069     char* trimmed; //trimmed string pointer
00070     //-------------------------------------------------------------------------
00071     // Independent motion
00072     x_e(GCmd(g, "DP 0")); //define position zero on A
00073     x_e(GCmdT(g, "RP", buf, sizeof(buf), &trimmed));
00074     cout « "\nPosition: " « trimmed « '\n';
00075     x_e(GCmd(g, "SP 4000")); //set up speed
00076     x_e(GCmd(g, "AC 1280000")); //acceleration
00077     x_e(GCmd(g, "DC 1280000")); //deceleration
00078     x_e(GCmd(g, "PR 8000")); //Postion Relative. B will take longer to make its move.
00079     x_e(GCmd(g, "SH A")); //Servo Here
00080     cout « "Beginning independent motion... ";
00081     x_e(GCmd(g, "BG A")); //Begin motion
00082     x_e(x_dr_motioncomplete(g, "A")); //Block until motion is complete on axes A and B
00083     cout « "Motion Complete on A\n";
00084     x_e(GCmdT(g, "RP", buf, sizeof(buf), &trimmed));
00085     cout « "Position: " « trimmed « '\n';;
00086 # endif
00087
00088   return GALIL_EXAMPLE_OK;
00089 }
00090
00091
00092 int x_dr_motioncomplete(GCon g, GCStringIn axes)
00093 {
00094
00095   GReturn rc;
00096   GDataRecord r;
00097
00098
00099   int original_dr;
00100   x_e(GCmdI(g, "MG_DR", &original_dr)); //grab the current DR value
00101   x_e(GRecordRate(g, 100)); //set up DR to 10Hz
00102
00103   int len = strlen(axes);
00104   UW* axis_status;
00105   for (int i = 0; i < len; /*blank*/) //iterate through all chars in axes to make the axis mask
```

```
00106   {
00107     rc = GRecord(g, &r, G_DR);
00108     //support just A-H
00109     switch (axes[i])
00110     {
00111     case 'A':
00112       axis_status = &r.dmc4000.axis_a_status;
00113       break;
00114     case 'B':
00115       axis_status = &r.dmc4000.axis_b_status;
00116       break;
00117     case 'C':
00118       axis_status = &r.dmc4000.axis_c_status;
00119       break;
00120     case 'D':
00121       axis_status = &r.dmc4000.axis_d_status;
00122       break;
00123     case 'E':
00124       axis_status = &r.dmc4000.axis_e_status;
00125       break;
00126     case 'F':
00127       axis_status = &r.dmc4000.axis_f_status;
00128       break;
00129     case 'G':
00130       axis_status = &r.dmc4000.axis_g_status;
00131       break;
00132     case 'H':
00133       axis_status = &r.dmc4000.axis_h_status;
00134       break;
00135     default:
00136       axis_status = 0;
00137     }
00138
00139     if (axis_status)
00140       if (!(*axis_status & 0x8000)) //bit 15 is "Move in progress"
00141         i++;
00142   }
00143
00144   char buf[16];
00145   sprintf(buf, "DR %d", original_dr);
00146   x_e(GCmd(g, buf)); //restore DR
00147
00148   return GALIL_EXAMPLE_OK;
00149
00150 }
```

10.169 x_nonblocking.cpp File Reference

```
#include "x_examples.h"
#include <iomanip>
```

Functions

- void progress ()
- int x_nonblocking (GCon g)

  Examples of using non-blocking operation of GRecord(), GInterrupt(), and GMessage().

Variables

- int cur = 0
- char chars [] = { '|', '\\', '-','/'}

10.169.1 Detailed Description

Example usage of GMessage, GRecord, and GInterrupt for non-blocking operation.
Definition in file x_nonblocking.cpp.

10.169.2 Function Documentation

10.169.2.1 progress()

```
void progress ( )
```

Definition at line 13 of file x_nonblocking.cpp.

10.169.2.2 x_nonblocking()

```
int x_nonblocking (
            GCon g )
```

Examples of using non-blocking operation of GRecord(), GInterrupt(), and GMessage().
Typical usage of the asynchronous data streams is to call the function with a given timeout. The function will then block until the desired data is received of the timeout occurs. However, to check for available messages, ascynchronous records, or interrupts, the user can set a timeout of zero and the functions will return waiting data. See the source of this example for more detail.
Definition at line 22 of file x_nonblocking.cpp.
References G_DR, G_GCLIB_NON_BLOCKING_READ_EMPTY, GCmd(), GInterrupt(), GMessage(), GProgramDownload(), GRecord(), GRecordRate(), GTimeout(), vector(), and x_e().

10.169.3 Variable Documentation

10.169.3.1 chars

```
char chars[] = { '|', '\\', '-','/'}
```

Definition at line 11 of file x_nonblocking.cpp.

10.169.3.2 cur

```
int cur = 0
```

Definition at line 10 of file x_nonblocking.cpp.

10.170 x_nonblocking.cpp

Go to the documentation of this file.

```
00001
00007 #include "x_examples.h"
00008 #include <iomanip> //hex printing
00009
00010 int cur = 0;
00011 char chars[] = { '|', '\\', '-','/'};
00012 //simple progress indicator.
00013 void progress()
00014 {
00015   cout << chars[cur] << '\r';
00016   cout.flush();
00017   ++cur %= 4;
00018 }
00019
00020
00021 //demonstrates non-blocking calls of GMessage(), GInterrupt(), and GRecord().
00022 int x_nonblocking(GCon g)
00023 {
00024
00025   // *** we assume -s ALL was passed to Open. ***
00026   GReturn rc;
00027   char buf[1024]; //read buffer
00028
00029   cout << "\n***************************************************************************\n";
00030   cout << "Example GMessage non-blocking usage\n";
00031   cout << "***************************************************************************\n";
00032
00033   x_e(GProgramDownload(g, "WT2000;MG TIME;EN", 0)); //wait 2 seconds, then print message
00034   x_e(GCmd(g, "XQ"));
00035
00036   x_e(GTimeout(g, 0)); //set timeout to zero for non-blocking read
00037
00038   for (int i = 0;
00039     (rc = GMessage(g, buf, sizeof(buf))) == G_GCLIB_NON_BLOCKING_READ_EMPTY;
00040     i++)
00041   {
00042     x_e(GCmd(g, "MGTIME")); //do something useful here...
00043     progress(); //and here
00044   }
00045
00046   x_e(rc); //verify the return code
00047   x_e(GTimeout(g, -1)); //put the timeout back to the GOpen() setting
00048   cout << buf << '\n'; //print the message
```

```
00049
00050   cout « "\n*************************************************************************\n";
00051   cout « "Example GInterrupt non-blocking usage\n";
00052   cout « "*************************************************************************\n";
00053   x_e(GProgramDownload(g, "WT2000;UI1;EN", 0)); //wait 2 seconds, then fire interrupt
00054   x_e(GCmd(g, "XQ")); //start code
00055
00056   GStatus byte = 0; //variable for the interrupt byte
00057   x_e(GTimeout(g, 0)); //set timeout to zero for non-blocking read
00058
00059   while ((rc = GInterrupt(g, &byte)) == G_GCLIB_NON_BLOCKING_READ_EMPTY)
00060   {
00061     x_e(GCmd(g, "MGTIME")); //do something useful here...
00062     progress(); //and here
00063   }
00064   x_e(rc); //verify the return code
00065   x_e(GTimeout(g, -1)); //put the timeout back to the GOpen() setting
00066   cout « " " « hex « uppercase « (int) byte « dec « '\n'; //print the byte in hex
00067
00068
00069   cout « "\n*************************************************************************\n";
00070   cout « "Example GRecord non-blocking usage\n";
00071   cout « "*************************************************************************\n";
00072
00073   GDataRecord dr;
00074   x_e(GRecordRate(g, 0)); //turn off data record
00075   x_e(GTimeout(g, 0)); //set timeout to zero for non-blocking read
00076   GRecord(g, &dr, G_DR); //throw away a waiting record
00077   x_e(GTimeout(g, -1)); //put the timeout back to the GOpen() setting
00078   x_e(GRecordRate(g, 2000)); //turn on data record
00079   x_e(GTimeout(g, 0)); //set timeout to zero for non-blocking read
00080   while ((rc = GRecord(g, &dr, G_DR)) == G_GCLIB_NON_BLOCKING_READ_EMPTY)
00081   {
00082     x_e(GCmd(g, "MGTIME")); //do something useful here...
00083     progress(); //and here
00084   }
00085   x_e(rc); //verify the return code
00086   x_e(GTimeout(g, -1)); //put the timeout back to the GOpen() setting
00087   x_e(GRecordRate(g, 0)); //turn off data record
00088   cout « " " « dr.dmc4000.sample_number « '\n';
00089
00090   return GALIL_EXAMPLE_OK;
00091 }
```

10.171 x_programs.cpp File Reference

```
#include "x_examples.h"
```

Functions

- int x_programs (GCon g)

  Example GProgramDownload() and GProgramUpload() usage.

10.171.1 Detailed Description

Example GProgramDownload() and GProgramUpload() usage.
Definition in file x_programs.cpp.

10.171.2 Function Documentation

10.171.2.1 x_programs()

```
int x_programs (
            GCon g )
```

Example GProgramDownload() and GProgramUpload() usage.
Demonstrates program download and upload including compression.
Definition at line 9 of file x_programs.cpp.
References G_NO_ERROR, G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT, GCmd(), GCmdI(), GProgramDownload(), GProgramDownloadFile(), GProgramUpload(), GProgramUploadFile(), GSleep(), vector(), and x_e().

10.172 x_programs.cpp

Go to the documentation of this file.

```
#include "x_examples.h"

int x_programs(GCon g)
{
  cout « "\n************************************************************************\n";
  cout « "Example GProgramDownload() and GProgramUpload() usage\n";
  cout « "************************************************************************\n";
  GReturn rc = G_NO_ERROR; //return code
  char buf[1024]; //traffic buffer

  //----------------------------------------------------------------------
  //capping compression
  string program =
"#A;i=0;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i
  //Program line above is too large to fit on any Galil controller, however it can easily fit if
broken up with level 4 compression.
  //the value of i at the end of code execution is 20.

  if ((rc = GProgramDownload(g, program.c_str(), "--max 3"))
    == G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT)
    cout « "GProgramDownload() correctly errored. Can't fit with level 3 compression\n";
  else
  {
    cout « "Unexpected GProgramDownload() behaviour\n";
    return GALIL_EXAMPLE_ERROR;
  }

  //----------------------------------------------------------------------
  //full compression
  if ((rc = GProgramDownload(g, program.c_str(), "--max 4"))
    == G_NO_ERROR)
    cout « "Program Downloaded with compression level 4\n";
  else
  {
    cout « "Unexpected GProgramDownload() behaviour\n";
    return GALIL_EXAMPLE_ERROR;
  }

  x_e(GProgramDownload(g, program.c_str(), 0)); //null for preprocessor equivalent to
GProgramDownload(g, program.c_str(), "--compress 4")

  //----------------------------------------------------------------------
  //program upload
  cout « "Uploading program:\n";
  x_e(GProgramUpload(g, buf, sizeof(buf)));
  cout « buf;

  //----------------------------------------------------------------------
  //program upload file
  x_e(GProgramUploadFile(g, "temp.dmc"));

  //----------------------------------------------------------------------
  //program download file
  x_e(GProgramDownload(g, "", 0)); //erase program
  x_e(GProgramDownloadFile(g, "temp.dmc", 0));

  //----------------------------------------------------------------------
  //executing/verifying code
  GCmd(g, "XQ"); //execute the code
  GSleep(10); //wait a brief interval for the code to complete, TB could be checked with a mask too.
  int val;
  GCmdI(g, "i=?", &val);
  if (val == 20)
    cout « "\n\nProgram executed as expected";
  else
  {
    cout « "\n\nUnexpected i value " « val « '\n';
    return GALIL_EXAMPLE_ERROR;
  }

  return GALIL_EXAMPLE_OK;
}
```

10.173 x_simple.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "gclibo.h"
```

Functions

- void check (GReturn rc)
- int main ()

Variables

- GCon g = 0

10.173.1 Detailed Description

A very simple example for using gclib. See x_examples.cpp for more in-depth examples.
Definition in file x_simple.c.

10.173.2 Function Documentation

10.173.2.1 check()

```
void check (
            GReturn rc )
```

Definition at line 14 of file x_simple.c.

10.173.2.2 main()

```
int main ( )
```

Definition at line 25 of file x_simple.c.

10.173.3 Variable Documentation

10.173.3.1 g

```
GCon g = 0
```

Definition at line 11 of file x_simple.c.

10.174 x_simple.c

Go to the documentation of this file.

```
00001
00006 #include <stdio.h> //printf()
00007 #include <stdlib.h> //exit()
00008
00009 #include "gclibo.h" //by including the open-source header, all other headers are pulled in.
00010
00011 GCon g = 0; //var used to refer to a unique connection
00012
00013 //check return code from most gclib calls
00014 void check(GReturn rc)
00015 {
00016   if (rc != G_NO_ERROR)
00017   {
00018     printf("ERROR: %d", rc);
00019     if (g)
00020       GClose(g);
00021     exit (rc);
00022   }
00023 }
00024
00025 int main()
00026 {
00027   char buf[1024]; //traffic buffer
00028
00029   check(GVersion(buf, sizeof(buf)));
00030   printf("version: %s\n", buf); //Print the library version
00031
00032   check(GOpen("192.168.0.43", &g)); //Open a connection to Galil, store the identifier in g.
00033
00034   check(GInfo(g, buf, sizeof(buf)));
```

```
00035   printf("info: %s\n", buf); //Print the connection info
00036
00037   check(GCommand(g, "MG TIME", buf, sizeof(buf), 0)); //Send MG TIME. Because response is ASCII, don't
      care about bytes read.
00038   printf("response: %s\n", buf); //Print the response
00039
00040   if (g) //don't call close on a nullptr
00041     GClose(g); //Don't forget to close!
00042
00043   return G_NO_ERROR;
00044 }
```

10.175 gclib.h File Reference

```
#include "gclib_record.h"
#include "gclib_errors.h"
```

Macros

- #define GCLIB_DLL_EXPORTED
- #define GCALL __stdcall

  Specify calling convention for Windows.
- #define G_DR 1

  Value for GRecord() `method` variable for acquiring a data record via DR mode.
- #define G_QR 0

  Value for GRecord() `method` variable for acquiring a data record via QR mode.
- #define G_BOUNDS -1

  For functions that take range options, e.g. GArrayUpload(), use this value for full range.
- #define G_CR 0

  For GArrayUpload(), use this value in the delim field to delimit with carriage returns.
- #define G_COMMA 1

  For GArrayUpload(), use this value in the delim field to delimit with commas.
- #define G_PUBLISH_SERVER 1

  For GPublishServer(), use this value to publish server to local network.
- #define G_REMOVE_SERVER 0

  For GPublishServer(), use this value to remove server from local network.
- #define G_UTIL_TIMEOUT 1

  GUtility(), Access to timeout.
- #define G_UTIL_TIMEOUT_OVERRIDE 2

  GUtility(), read/write access to timeout override.
- #define G_USE_INITIAL_TIMEOUT -1

  GUtility(), for timeout override. Set `G_UTIL_TIMEOUT_OVERRIDE` to this value to use initial GOpen() timeout (`--timeout`).
- #define G_UTIL_VERSION 128

  GUtility(), get a library version string.
- #define G_UTIL_INFO 129

  GUtility(), get a connection info string.
- #define G_UTIL_SLEEP 130

  GUtility(), specify an interval to sleep.
- #define G_UTIL_ADDRESSES 131

  GUtility(), get a list of available connections.
- #define G_UTIL_IPREQUEST 132

  GUtility(), get a list of hardware requesting IPs.
- #define G_UTIL_ASSIGN 133

  GUtility(), assign IP addresses over the network.

- #define G_UTIL_DEVICE_INITIALIZE 134

  GUtility(), sends CF, CW, EO etc. to initialize the connection. Useful after RS or other reset.
- #define G_UTIL_PING 135

  GUtility(), uses ICMP ping to determine if an IP address is reachable and assigned.
- #define G_UTIL_ERROR_CONTEXT 136

  GUtility(), provides additional error context, where available.
- #define G_UTIL_GCAPS_HOST 256
- #define G_UTIL_GCAPS_VERSION 257

  GUtility(), get the version of the gcaps server.
- #define G_UTIL_GCAPS_KEEPALIVE 258

  GUtility(), Deprecated 20210119. No longer functional.
- #define G_UTIL_GCAPS_ADDRESSES 259

  GUtility(), get a list of available connections from the gcaps server.
- #define G_UTIL_GCAPS_IPREQUEST 260

  GUtility(), get a list of hardware requesting IPs from the gcaps server.
- #define G_UTIL_GCAPS_ASSIGN 261

  GUtility(), assign IP addresses over the network from the gcaps server.
- #define G_UTIL_GCAPS_PING 262

  GUtility(), uses ICMP ping to determine if an IP address is reachable and assigned. Ping sent from the gcaps server.
- #define G_UTIL_GCAPS_LIST_SERVERS 263

  GUtility(), get a list of all available gcaps servers on the local network.
- #define G_UTIL_GCAPS_PUBLISH_SERVER 264

  GUtility(), make local gcaps server discoverable by other gcaps servers on the local network.
- #define G_UTIL_GCAPS_SET_SERVER 265

  GUtility(), set the new active gcaps server.
- #define G_UTIL_GCAPS_SERVER_STATUS 266

  GUtility(), get information on the local server's name and if it is published to the local network.
- #define G_UTIL_GCAPS_REMOTE_CONNECTIONS 267

  GUtility(), get a list of remote addresses connected to local server.
- #define G_UTIL_GCAPS_SERVER_INFO 268
- #define G_UTIL_GCAPS_ADDRESSES_GET_REMEMBERED 269

  GUtility(), returns true if gcaps is remembering ip assignments.
- #define G_UTIL_GCAPS_ADDRESSES_SET_REMEMBERED 270

  GUtility(), sets if gcaps should remember ip assignments.
- #define G_SMALL_BUFFER 1024

  Most reads from Galil are small. This value will easily hold most, e.g. TH, TZ, etc.
- #define G_HUGE_BUFFER 524288

  Most reads from Galil hardware are small. This value will hold the largest array or program upload/download possible.
- #define G_LINE_BUFFER 80

  For writes, via command interpreter, to the Galil.

Typedefs

- typedef int GReturn

  Every function returns a value of type GReturn. See gclib_errors.h for possible values.
- typedef void * GCon

  Connection handle. Unique for each connection in process. Assigned a non-zero value in GOpen().
- typedef unsigned int GSize

  Size of buffers, etc.
- typedef int GOption

  Option integer for various formatting, etc.

- typedef char * GCStringOut

  C-string output from the library. Implies null-termination.
- typedef const char * GCStringIn

  C-string input to the library. Implies null-termination.
- typedef char * GBufOut

  Data output from the library. No null-termination implied. Returned values may be null-terminated, see function documentation for details.
- typedef const char * GBufIn

  Data input to the library. No null-termination, function will have a GSize to indicate bytes to write .
- typedef unsigned char GStatus

  Interrupt status byte.
- typedef void * GMemory

  Pointer to untyped memory for use in GUtility().

Functions

- GCLIB_DLL_EXPORTED GReturn GCALL GOpen (GCStringIn address, GCon *g)

  Open a connection to a Galil Controller.
- GCLIB_DLL_EXPORTED GReturn GCALL GClose (GCon g)

  Closes a connection to a Galil Controller.
- GCLIB_DLL_EXPORTED GReturn GCALL GRead (GCon g, GBufOut buffer, GSize buffer_len, GSize *bytes_read)

  Performs a read on the connection.
- GCLIB_DLL_EXPORTED GReturn GCALL GWrite (GCon g, GBufIn buffer, GSize buffer_len)

  Performs a write on the connection.
- GCLIB_DLL_EXPORTED GReturn GCALL GCommand (GCon g, GCStringIn command, GBufOut buffer, GSize buffer_len, GSize *bytes_returned)

  Performs a command-and-response transaction on the connection.
- GCLIB_DLL_EXPORTED GReturn GCALL GProgramDownload (GCon g, GCStringIn program, GCStringIn preprocessor)

  Downloads a program to the controller's program buffer.
- GCLIB_DLL_EXPORTED GReturn GCALL GProgramUpload (GCon g, GBufOut buffer, GSize buffer_len)

  Uploads a program from the controller's program buffer.
- GCLIB_DLL_EXPORTED GReturn GCALL GArrayDownload (GCon g, const GCStringIn array_name, GOption first, GOption last, GCStringIn buffer)

  Downloads array data to a pre-dimensioned array in the controller's array table.
- GCLIB_DLL_EXPORTED GReturn GCALL GArrayUpload (GCon g, const GCStringIn array_name, GOption first, GOption last, GOption delim, GBufOut buffer, GSize buffer_len)

  Uploads array data from the controller's array table.
- GCLIB_DLL_EXPORTED GReturn GCALL GRecord (GCon g, union GDataRecord *record, GOption method)

  Provides a fresh copy of the controller's data record. Data is cast into a union, GDataRecord.
- GCLIB_DLL_EXPORTED GReturn GCALL GMessage (GCon g, GCStringOut buffer, GSize buffer_len)

  Provides access to unsolicited messages from the controller.
- GCLIB_DLL_EXPORTED GReturn GCALL GInterrupt (GCon g, GStatus *status_byte)

  Provides access to PCI and UDP interrupts from the controller.
- GCLIB_DLL_EXPORTED GReturn GCALL GFirmwareDownload (GCon g, GCStringIn filepath)

  Upgrade firmware.
- GCLIB_DLL_EXPORTED GReturn GCALL GUtility (GCon g, GOption request, GMemory memory1, GMemory memory2)

  Provides read/write access to driver settings and convenience features based on the request variable.

10.175.1 Detailed Description

Defines the interface for the Galil C Library (GCLIB).
Definition in file gclib.h.

10.175.2 Macro Definition Documentation

10.175.2.1 G_BOUNDS

```
#define G_BOUNDS -1
```

For functions that take range options, e.g. GArrayUpload(), use this value for full range.
Definition at line 52 of file gclib.h.

10.175.2.2 G_COMMA

```
#define G_COMMA 1
```

For GArrayUpload(), use this value in the delim field to delimit with commas.
Definition at line 54 of file gclib.h.

10.175.2.3 G_CR

```
#define G_CR 0
```

For GArrayUpload(), use this value in the delim field to delimit with carriage returns.
Definition at line 53 of file gclib.h.

10.175.2.4 G_DR

```
#define G_DR 1
```

Value for GRecord() `method` variable for acquiring a data record via DR mode.
Definition at line 50 of file gclib.h.

10.175.2.5 G_HUGE_BUFFER

```
#define G_HUGE_BUFFER 524288
```

Most reads from Galil hardware are small. This value will hold the largest array or program upload/download possible.
Definition at line 90 of file gclib.h.

10.175.2.6 G_LINE_BUFFER

```
#define G_LINE_BUFFER 80
```

For writes, via command interpreter, to the Galil.
Definition at line 91 of file gclib.h.

10.175.2.7 G_PUBLISH_SERVER

```
#define G_PUBLISH_SERVER 1
```

For GPublishServer(), use this value to publish server to local network.
Definition at line 55 of file gclib.h.

10.175.2.8 G_QR

```
#define G_QR 0
```

Value for GRecord() `method` variable for acquiring a data record via QR mode.
Definition at line 51 of file gclib.h.

10.175.2.9 G_REMOVE_SERVER

```
#define G_REMOVE_SERVER 0
```

For GPublishServer(), use this value to remove server from local network.
Definition at line 56 of file gclib.h.

10.175.2.10 G_SMALL_BUFFER

```
#define G_SMALL_BUFFER 1024
```

Most reads from Galil are small. This value will easily hold most, e.g. TH, TZ, etc.
Definition at line 89 of file gclib.h.

10.175.2.11 G_USE_INITIAL_TIMEOUT

```
#define G_USE_INITIAL_TIMEOUT -1
```

GUtility(), for timeout override. Set `G_UTIL_TIMEOUT_OVERRIDE` to this value to use initial GOpen() timeout (`--timeout`).
Definition at line 61 of file gclib.h.

10.175.2.12 G_UTIL_ADDRESSES

```
#define G_UTIL_ADDRESSES 131
```

GUtility(), get a list of available connections.
Definition at line 65 of file gclib.h.

10.175.2.13 G_UTIL_ASSIGN

```
#define G_UTIL_ASSIGN 133
```

GUtility(), assign IP addresses over the network.
Definition at line 67 of file gclib.h.

10.175.2.14 G_UTIL_DEVICE_INITIALIZE

```
#define G_UTIL_DEVICE_INITIALIZE 134
```

GUtility(), sends CF, CW, EO etc. to initialize the connection. Useful after RS or other reset.
Definition at line 68 of file gclib.h.

10.175.2.15 G_UTIL_ERROR_CONTEXT

```
#define G_UTIL_ERROR_CONTEXT 136
```

GUtility(), provides additional error context, where available.
Definition at line 70 of file gclib.h.

10.175.2.16 G_UTIL_GCAPS_ADDRESSES

```
#define G_UTIL_GCAPS_ADDRESSES 259
```

GUtility(), get a list of available connections from the gcaps server.
Definition at line 75 of file gclib.h.

10.175.2.17 G_UTIL_GCAPS_ADDRESSES_GET_REMEMBERED

```
#define G_UTIL_GCAPS_ADDRESSES_GET_REMEMBERED 269
```

GUtility(), returns true if gcaps is remembering ip assignments.
Definition at line 85 of file gclib.h.

10.175.2.18 G_UTIL_GCAPS_ADDRESSES_SET_REMEMBERED

```
#define G_UTIL_GCAPS_ADDRESSES_SET_REMEMBERED 270
```

GUtility(), sets if gcaps should remember ip assignments.
Definition at line 86 of file gclib.h.

10.175.2.19 G_UTIL_GCAPS_ASSIGN

```
#define G_UTIL_GCAPS_ASSIGN 261
```

GUtility(), assign IP addresses over the network from the gcaps server.
Definition at line 77 of file gclib.h.

10.175.2.20 G_UTIL_GCAPS_HOST

```
#define G_UTIL_GCAPS_HOST 256
```
Definition at line 72 of file gclib.h.

10.175.2.21 G_UTIL_GCAPS_IPREQUEST

```
#define G_UTIL_GCAPS_IPREQUEST 260
```
GUtility(), get a list of hardware requesting IPs from the gcaps server.
Definition at line 76 of file gclib.h.

10.175.2.22 G_UTIL_GCAPS_KEEPALIVE

```
#define G_UTIL_GCAPS_KEEPALIVE 258
```
GUtility(), Deprecated 20210119. No longer functional.
Definition at line 74 of file gclib.h.

10.175.2.23 G_UTIL_GCAPS_LIST_SERVERS

```
#define G_UTIL_GCAPS_LIST_SERVERS 263
```
GUtility(), get a list of all available gcaps servers on the local network.
Definition at line 79 of file gclib.h.

10.175.2.24 G_UTIL_GCAPS_PING

```
#define G_UTIL_GCAPS_PING 262
```
GUtility(), uses ICMP ping to determine if an IP address is reachable and assigned. Ping sent from the gcaps server.
Definition at line 78 of file gclib.h.

10.175.2.25 G_UTIL_GCAPS_PUBLISH_SERVER

```
#define G_UTIL_GCAPS_PUBLISH_SERVER 264
```
GUtility(), make local gcaps server discoverable by other gcaps servers on the local network.
Definition at line 80 of file gclib.h.

10.175.2.26 G_UTIL_GCAPS_REMOTE_CONNECTIONS

```
#define G_UTIL_GCAPS_REMOTE_CONNECTIONS 267
```
GUtility(), get a list of remote addresses connected to local server.
Definition at line 83 of file gclib.h.

10.175.2.27 G_UTIL_GCAPS_SERVER_INFO

```
#define G_UTIL_GCAPS_SERVER_INFO 268
```
Definition at line 84 of file gclib.h.

10.175.2.28 G_UTIL_GCAPS_SERVER_STATUS

```
#define G_UTIL_GCAPS_SERVER_STATUS 266
```
GUtility(), get information on the local server's name and if it is published to the local network.
Definition at line 82 of file gclib.h.

10.175.2.29 G_UTIL_GCAPS_SET_SERVER

```
#define G_UTIL_GCAPS_SET_SERVER 265
```
GUtility(), set the new active gcaps server.
Definition at line 81 of file gclib.h.

10.175.2.30 G_UTIL_GCAPS_VERSION

```
#define G_UTIL_GCAPS_VERSION 257
```

GUtility(), get the version of the gcaps server.
Definition at line 73 of file gclib.h.

10.175.2.31 G_UTIL_INFO

```
#define G_UTIL_INFO 129
```

GUtility(), get a connection info string.
Definition at line 63 of file gclib.h.

10.175.2.32 G_UTIL_IPREQUEST

```
#define G_UTIL_IPREQUEST 132
```

GUtility(), get a list of hardware requesting IPs.
Definition at line 66 of file gclib.h.

10.175.2.33 G_UTIL_PING

```
#define G_UTIL_PING 135
```

GUtility(), uses ICMP ping to determine if an IP address is reachable and assigned.
Definition at line 69 of file gclib.h.

10.175.2.34 G_UTIL_SLEEP

```
#define G_UTIL_SLEEP 130
```

GUtility(), specify an interval to sleep.
Definition at line 64 of file gclib.h.

10.175.2.35 G_UTIL_TIMEOUT

```
#define G_UTIL_TIMEOUT 1
```

GUtility(), Access to timeout.
Definition at line 59 of file gclib.h.

10.175.2.36 G_UTIL_TIMEOUT_OVERRIDE

```
#define G_UTIL_TIMEOUT_OVERRIDE 2
```

GUtility(), read/write access to timeout override.
Definition at line 60 of file gclib.h.

10.175.2.37 G_UTIL_VERSION

```
#define G_UTIL_VERSION 128
```

GUtility(), get a library version string.
Definition at line 62 of file gclib.h.

10.175.2.38 GCALL

```
#define GCALL __stdcall
```

Specify calling convention for Windows.
Definition at line 38 of file gclib.h.

10.175.2.39 GCLIB_DLL_EXPORTED

```
#define GCLIB_DLL_EXPORTED
```

Definition at line 31 of file gclib.h.

10.175.3 Typedef Documentation

10.175.3.1 GBufIn

```
typedef const char* GBufIn
```

Data input to the library. No null-termination, function will have a GSize to indicate bytes to write .
Definition at line 100 of file gclib.h.

10.175.3.2 GBufOut

```
typedef char* GBufOut
```

Data output from the library. No null-termination implied. Returned values may be null-terminated, see function documentation for details.
Definition at line 99 of file gclib.h.

10.175.3.3 GCon

```
typedef void* GCon
```

Connection handle. Unique for each connection in process. Assigned a non-zero value in GOpen().
Definition at line 94 of file gclib.h.

10.175.3.4 GCStringIn

```
typedef const char* GCStringIn
```

C-string input to the library. Implies null-termination.
Definition at line 98 of file gclib.h.

10.175.3.5 GCStringOut

```
typedef char* GCStringOut
```

C-string output from the library. Implies null-termination.
Definition at line 97 of file gclib.h.

10.175.3.6 GMemory

```
typedef void* GMemory
```

Pointer to untyped memory for use in GUtility().
Definition at line 102 of file gclib.h.

10.175.3.7 GOption

```
typedef int GOption
```

Option integer for various formatting, etc.
Definition at line 96 of file gclib.h.

10.175.3.8 GReturn

```
typedef int GReturn
```

Every function returns a value of type GReturn. See gclib_errors.h for possible values.
Definition at line 93 of file gclib.h.

10.175.3.9 GSize

```
typedef unsigned int GSize
```

Size of buffers, etc.
Definition at line 95 of file gclib.h.

10.175.3.10 GStatus

```
typedef unsigned char GStatus
```

Interrupt status byte.
Definition at line 101 of file gclib.h.

10.175.4 Function Documentation

10.175.4.1 GArrayDownload()

```
GCLIB_DLL_EXPORTED GReturn GCALL GArrayDownload (
            GCon g,
            const GCStringIn array_name,
            GOption first,
            GOption last,
            GCStringIn buffer )
```

Downloads array data to a pre-dimensioned array in the controller's array table.

Warning

The array must already exist on the controller and be sufficient dimension to hold the desired array data, e.g. via DM.

Parameters

g	Connection's handle.
array_name	Null-terminated string containing the name of the array to download. Must match the array name used in DM.
first	The first element of the array for sub-array downloads. G_BOUNDS to omit.
last	The last element of the array for sub-array downloads. G_BOUNDS to omit.
buffer	Buffer containing the null-terminated data to be sent to the controller. The array data may be separated with *carriage return*, *carriage return + line feed*, or a *comma*. No spaces.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_arrays.cpp for an example.
Referenced by H_DownloadArraysFromList(), and x_arrays().

10.175.4.2 GArrayUpload()

```
GCLIB_DLL_EXPORTED GReturn GCALL GArrayUpload (
            GCon g,
            const GCStringIn array_name,
            GOption first,
            GOption last,
            GOption delim,
            GBufOut buffer,
            GSize buffer_len )
```

Uploads array data from the controller's array table.

Parameters

g	Connection's handle.
array_name	Null-terminated string containing the name of the array to upload.
first	The first element of the array for sub-array uploads. G_BOUNDS to omit.
last	The last element of the array for sub-array uploads. G_BOUNDS to omit.

Parameters

delim	Sets the delimeter between array elements in the returned data, G_CR specifies carriage return, G_COMMA specifies comma.
buffer	Buffer to receive the uploaded data. The data will be null terminated unless function returns G_BAD_LOST_DATA due to the buffer being too small to hold the data.
buffer_len	The length of the receive buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_arrays.cpp for an example.
Referenced by H_UploadArrayToList(), write_array_to_file(), and x_arrays().

10.175.4.3 GClose()

```
GCLIB_DLL_EXPORTED GReturn GCALL GClose (
            GCon g )
```

Closes a connection to a Galil Controller.

Attention

gclib requires that GClose() be called whenever a program is finished with a controller. This includes when a program closes. A rule of thumb is that for every GOpen() call on a given connection, a GClose() call should be found on every code path. Failing to call GClose() may cause controller resources to not be released or can hang the process if there are outstanding asynchronous operations. The latter can occur, for example, if a call to GRead() times out and the process exits without calling GClose(). In this case, GRead() still has an outstanding asynchronous read pending. GClose() will terminate this operation allowing the process to exit correctly.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_examples.cpp for an example.

10.175.4.4 GCommand()

```
GCLIB_DLL_EXPORTED GReturn GCALL GCommand (
            GCon g,
            GCStringIn command,
            GBufOut buffer,
            GSize buffer_len,
            GSize * bytes_returned )
```

Performs a *command-and-response* transaction on the connection.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller. The library will append a carriage return to the command string.
buffer	Buffer for the response. Will be filled with the response from the controller. The data will be null terminated unless the function returns G_BAD_LOST_DATA due to the buffer being too small to hold the data.

Parameters

buffer_len	The size of the response buffer.
bytes_returned	The size of the data returned from the controller. This does not include null termination. This argument may be null if the value is not desired.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Referenced by commands(), error(), GCmd(), GCmdD(), gclib::GCmdD(), GCmdI(), gclib::GCmdI(), GCmdT(), GWaitForBool(), motion_complete(), x_gcommand(), x_ginterrupt(), and x_grecord().

10.175.4.5 GFirmwareDownload()

```
GCLIB_DLL_EXPORTED GReturn GCALL GFirmwareDownload (
            GCon g,
            GCStringIn filepath )
```

Upgrade firmware.

Parameters

g	Connection's handle.
filepath	The full file path to the Galil-supplied firmware hex file. See http://www.galil.com/downloads/firmware

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

```
ec(GInfo(g, buf, sizeof(buf))); //get conntroller info
cout « buf « '\n'; //print the info
ec(GFirmwareDownload(g, "F:/1806.dmc/dmc-1806-r11a.hex"));
ec(GInfo(g, buf, sizeof(buf))); //get the info again
cout « buf « '\n';
// example output:
// GALILPCI1, DMC1846 Rev 1.1a-CM, 4232
// GALILPCI1, DMC1846 Rev 1.1a, 4232
```

10.175.4.6 GInterrupt()

```
GCLIB_DLL_EXPORTED GReturn GCALL GInterrupt (
            GCon g,
            GStatus * status_byte )
```

Provides access to PCI and UDP interrupts from the controller.
Interrupts can be generated automatically by the firmware on important events via `EI` (Enable Interrupt) or by the user in embedded DMC code via `UI` (User Interrupt). To use this function, `-s EI` must be used in the GOpen() address string to subscribe to interrupts.

Parameters

g	Connection's handle.
status_byte	A pointer to a GStatus to receive the status byte.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

GInterrupt() will block until an interrupt is received, or the function times out.

Note

If this function is called with a timeout of zero, a non-blocking read is performed. If interrupt data is waiting in the interrupt queue, the oldest byte will be popped off the queue. If there is no interrupt data queued, but there is data waiting in the socket or PCI FIFO, one read will be performed to process the waiting data. If new data is still not found after these two attempts, G_GCLIB_NON_BLOCKING_READ_EMPTY will be returned.

See x_ginterrupt.cpp for an example. See x_nonblocking.cpp for an example of non-blocking usage.
Referenced by check_interrupts(), motion_complete(), x_ei_motioncomplete(), x_ginterrupt(), and x_nonblocking().

10.175.4.7 GMessage()

```
GCLIB_DLL_EXPORTED GReturn GCALL GMessage (
            GCon g,
            GCStringOut buffer,
            GSize buffer_len )
```

Provides access to unsolicited messages from the controller.
To use this function, `-s MG` must be used in the GOpen() `address` string to subscribe to messages. Unsolicited bytes must be flagged by the high-bit setting, `CW 1`. The driver will automatically set this when subscribing to messages. The user should not overwrite this setting.
Unsolicited messages are data generated by the controller that are not in response to a command, a data record, or an interrupt. Examples follow.

1. Data generated by the `MG` command from embedded code. `MG` sent from the host is solicited.
2. Any command in an embedded program that returns data, e.g. `TP`, `RP`, `var=?`
3. A run time error in an embedded program, e.g. `?55 i=var`

Note

Messages are unframed byte streams. There is no guarantee that the user will get complete messages or single messages in a call to GMessage(). If multiple messages have been sent from the controller since the last call to `GMessage()`, they will all be placed in the buffer, separated by newline characters.

Parameters

g	Connection's handle.
buffer	The buffer to write the message data. The buffer will be null terminated.
buffer_len	The length of the user's buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

GMessage() will block until a message is received, or the function times out.

Note

If this function is called with a timeout of zero, a non-blocking read is performed. If message data has been processed since the last time the function was called, this data will be returned. If there is no processed message data, but there is data waiting in the socket or PCI FIFO, one read will be performed to process the waiting data. If new data is still not found after these two attempts, G_GCLIB_NON_BLOCKING_READ_↩ EMPTY will be returned.

Warning

When sending message streams through gcaps, the following non-printable bytes are illegal, `$00-$07` and `$10-$17`. These bytes may be routed to a third party device such as am HMI or display panel. See MG and CF.

See x_gmessage.cpp for an example. See x_nonblocking.cpp for an example of non-blocking usage.
Referenced by message(), x_gmessage(), and x_nonblocking().

10.175.4.8 GOpen()

```
GCLIB_DLL_EXPORTED GReturn GCALL GOpen (
            GCStringIn address,
            GCon * g )
```

Open a connection to a Galil Controller.

Parameters

address	Null-terminated address string. See table below.
g	Pointer to user's GCon variable. On success, the library will fill the user's variable with the handle to use for the rest of the connection. A valid g value is nonzero.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

address switch	Meaning	Arguments (default), other options	Examples
--address	**Simple address to hardware**	*IP address*, *PCI*, *COM port*	--address COM1
-a	shorthand for --address	See *Address Ranges* below	-a GALILPCI1
{no switch}	--address is implicit for any lone token		192.168.0.42
--baud	**Baud rate**	(115200), *valid baud...*	COM2 --baud 19200
-b	shorthand for --baud		COM3 -b 38400
--command	**Command-and-response socket protocol**	(TCP), UDP	192.168.0.42 --command TCP
-c	shorthand for --command		192.168.0.42 -c UDP
--direct	**Connect directly to hardware instead of via gcaps**		-a GALILPCI2 --direct
-d	shorthand for --direct		GALILPCI2 -d
--subscribe	**Subscribe to messages, data records, and/or interrupts**	(NONE), MG, DR, EI, ALL	192.168.0.42 --subscribe MG
-s	shorthand for --subscribe		192.168.0.42 -s DR -s EI
--timeout	**timeout in ms**	(5000), *0-65535*	192.168.0.42 --timeout 5000
-t	shorthand for --timeout		GALILPCI2 -t 500
--unsolicited	**Unsolicited socket protocol**	(UDP), NONE	192.168.0.42 --unsolicited NONE
-u	shorthand for --unsolicited		192.168.1.42 -u UDP
The following address switches are deprecated and will be unavailable starting July 1st, 2020.			
--p1	**Primary port for command-and-response traffic**	(23), *valid port number*	192.168.0.42 --p1 5000
--p2	**Secondary port for unsolicited traffic**	(60007), *valid port number*	192.168.0.42 --p2 5000

Operating System	Address Range	Notes
Windows	COM1 - COM256	RS232 and USB-to-serial
Linux	/dev/ttyS0 - /dev/ttyS255	RS232
Linux	/dev/ttyUSB0 - /dev/ttyUSB255	USB-to-serial, e.g. DMC-4103
Windows	GALILPCI1 - GALILPCI8	PCI
Linux	/dev/galilpci0 - /dev/galilpci7	PCI

See x_examples.cpp for an example.
When connecting to a network device, if the command-and-response socket is opened successfully but the unsolicited socket fails, GOpen() will still complete successfully. This allows connection to a Galil controller when only one Ethernet handle is available. Unsolicited traffic will not be accessible in this case.
Referenced by ip_assigner(), and main().

10.175.4.9 GProgramDownload()

```
GCLIB_DLL_EXPORTED GReturn GCALL GProgramDownload (
            GCon g,
            GCStringIn program,
            GCStringIn preprocessor )
```

Downloads a program to the controller's program buffer.

Parameters

g	Connection's handle.
program	Null-terminated program for download.
preprocessor	Options string for preprocessing the program before sending it to the controller. Null allows the library to use defaults for the download. See the Program Preprocessor documentation for options.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_programs.cpp for an example.
Referenced by GProgramDownloadFile(), GSetupDownloadFile(), message(), record_position(), x_ginterrupt(), x_gmessage(), x_nonblocking(), and x_programs().

10.175.4.10 GProgramUpload()

```
GCLIB_DLL_EXPORTED GReturn GCALL GProgramUpload (
            GCon g,
            GBufOut buffer,
            GSize buffer_len )
```

Uploads a program from the controller's program buffer.

Parameters

g	Connection's handle.
buffer	Buffer to receive the controller's program. The data will be null terminated unless function returns G_BAD_LOST_DATA due to the buffer being too small to hold the data.
buffer_len	The length of the receive buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_programs.cpp for an example.

Referenced by GProgramUploadFile(), and x_programs().

10.175.4.11 GRead()

```
GCLIB_DLL_EXPORTED GReturn GCALL GRead (
            GCon g,
            GBufOut buffer,
            GSize buffer_len,
            GSize * bytes_read )
```

Performs a read on the connection.

Parameters

g	Connection's handle.
buffer	The user's read buffer.
buffer_len	The length of the user's read buffer.
bytes_read	Pointer to a GSize which will be filled with the number of bytes read upon return.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Warning

This function is deprecated and will be removed in a future gclib version. Please contact Galil for needs not covered by the other gclib functions.

Unsolicited messages may be returned in the read data. The high bit of each message byte will be set unless the user changes the CW setting. Interrupts and Data Records are always filtered from a read.
See x_gread_gwrite.cpp for an example.
Referenced by x_gread_gwrite().

10.175.4.12 GRecord()

```
GCLIB_DLL_EXPORTED GReturn GCALL GRecord (
            GCon g,
            union GDataRecord * record,
            GOption method )
```

Provides a fresh copy of the controller's data record. Data is cast into a union, GDataRecord.

Parameters

g	Connection's handle.
record	A pointer to the user's DataRecord union to hold the copy.
method	Determines the method for acquiring the data. • `G_QR`: QR is used via command-and-response. • `G_DR`: DR is used for asynchronous acquisition.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

When using `G_DR`, the asynchronous data record must already be set up.

- `-s DR` must be used in the GOpen() `address` string to subscribe to records. The driver will automatically set the second argument of `DR`, where applicable.

- GRecordRate() should be issued to set DR to an appropriate interval, n. The interval must be no faster than the rate at which GRecord() is called.
- If GRecord() is called more slowly than the data record rate, stale data will be returned until GRecord() has been called once for each record sent by the controller.

GRecord() will block until the data record is received, or the transaction times out.

Note

If this function is called with a timeout of zero and the G_DR method, a non-blocking read is performed. If a data record has been processed since the last time the function was called, this data will be returned. If there is not a processed data reecord, but there is data waiting in the socket or PCI FIFO, one read will be performed to process the waiting data. If new data is still not found after these two attempts, G_GCLIB_↩ NON_BLOCKING_READ_EMPTY will be returned.

See x_grecord.cpp for an example. See x_nonblocking.cpp for an example of non-blocking usage.
Referenced by x_dr_motioncomplete(), x_grecord(), and x_nonblocking().

10.175.4.13 GUtility()

```
GCLIB_DLL_EXPORTED GReturn GCALL GUtility (
            GCon g,
            GOption request,
            GMemory memory1,
            GMemory memory2 )
```

Provides read/write access to driver settings and convenience features based on the request variable.

Note

The open source library, gclibo.h, has wrappers for most of these utilities.

Parameters

g	Connection's handle.
request	Defines the request. Input/Output and type of memory are implicit in the value of request. The following lists the supported request values.

- G_UTIL_TIMEOUT Read initial timeout value, as specified in GOpen() via `--timeout` switch.
  - `memory1` is output and must be an `unsigned short*`.
  - `memory2` is ignored, use null.
- G_UTIL_TIMEOUT_OVERRIDE See GTimeout(). Write/Read override timeout value.
  - `memory1` is input. If nonnull, value must be a `short*` holding the override, in milliseconds, for the timeout. Write `G_USE_INITIAL_TIMEOUT` to use initial timeout. If null, no write occurs.
  - `memory2` is output. If nonnul, value must be a `short*` which will be filled with the current override. `G_USE_INITIAL_TIMEOUT` indicates initial timeout used. If null, no read occurs. `memory2` is processed before 'memory1`.
- G_UTIL_VERSION See GVersion(). Returns the library version. A valid connection (g) is not necessary, i.e. g may be null.
  - `memory1` is output, and must be a `char*`. Data will be null terminated, even if the data must be truncated to do so.
  - `memory2` is input and must be an `unsigned int*` holding the length of the buffer in `memory1`.
- G_UTIL_INFO See GInfo(). Returns information about the connection.

  - memory1 is output and must be a char*. Data will be null terminated, even if the data must be truncated to do so.
  - memory2 is input and must be an unsigned int* holding the length of the buffer in memory1.

- G_UTIL_SLEEP See GSleep(). Platform-independent, non-busy, sleep. A valid connection (g) is not necessary, i.e. g may be null.
  - memory1 is input and must be an unsigned int*, units are milliseconds.
  - memory2 is ignored, use null.

- G_UTIL_ADDRESSES see GAddresses(). Provides a \n delimited listing of all available IP addresses, PCI addresses, and COM ports. A valid connection (g) is not necessary, i.e. g may be null. The suffix -d will be appended to each address to indicate these addresses are available via direct connection. See G_UTIL_↩ GCAPS_ADDRESSES for addresses through gcaps.
  - memory1 is output and must be a char*. Data will be null terminated, even if the data must be truncated to do so.
  - memory2 is input and must be an unsigned int* holding the length of the buffer in memory1.

- G_UTIL_IPREQUEST see GIpRequests(). Listens and returns a \n delimited listing of Galil MAC addresses sending BOOT-P or DHCP requests. The function will listen, and block, for roughly 5 seconds. A valid connection (g) is not necessary, i.e. g may be null.
  - memory1 is output and must be a char*. Data will be null terminated, even if the data must be truncated to do so.
  - memory2 is input and must be an unsigned int* holding the length of the buffer in memory1.

- G_UTIL_ASSIGN see GAssign(). Provides a method to assign an IP address given a Galil MAC address. A valid connection (g) is not necessary, i.e. g may be null.
  - memory1 is input and must be a char* containing the null terminated address that is to be assigned. e.g. "192.168.0.43".
  - memory2 is input and must be a char* containing the null terminated controller MAC address. e.g. "00:50:4C:20:01:23".

- G_UTIL_DEVICE_INITIALIZE Provides a method to reinitialize a connection after a reset, e.g. an RS command. Depending on the device type, the appropriate commands will be sent to configure the communication bus for optimal performance.
  - memory1 is ignored, use null.
  - memory2 is ignored, use null.

- G_UTIL_PING Uses ICMP ping to determine if an IP address is reachable and assigned. A valid connection (g) is not necessary, i.e. g may be null.
  - memory1 is input and must be a char* containing the null terminated address that is to be pinged. e.g. "192.168.0.43".
  - memory2 is output and must be an int*. The value will be set to zero if the ping times out, and nonzero if a ping reply is returned.

- G_UTIL_ERROR_CONTEXT More error detail for the last error on GCon, where available. The internal error message is cleared upon read.
  - memory1 is output and must be a char*. Data will be null terminated, even if the data must be truncated to do so.
  - memory2 is input and must be an unsigned int* holding the length of the buffer in memory1.

The following request values are for use with a @ref gcaps server.

- G_UTIL_GCAPS_VERSION see GVersion(). Returns the gcaps server version. A valid connection (g) is not necessary, i.e. g may be null. This operation will connect to the server to determine the version.
  - `memory1` is output and must be a `char*`. Data will be null terminated, even if the data must be truncated to do so.
  - `memory2` is input and must be an `unsigned int*` holding the length of the buffer in `memory1`.
- G_UTIL_GCAPS_ADDRESSES see GAddresses(). Provides a \n delimited listing of all available IP addresses, PCI addresses, and COM ports as available from the gcaps server. A valid connection (g) is not necessary, i.e. g may be null.
  - `memory1` is output and must be a `char*`. Data will be null terminated, even if the data must be truncated to do so.
  - `memory2` is input and must be an `unsigned int*` holding the length of the buffer in `memory1`.
- G_UTIL_GCAPS_IPREQUEST see GIpRequests(). Connects to gcaps and returns a \n delimited listing of Galil MAC addresses sending BOOT-P or DHCP requests. A valid connection (g) is not necessary, i.e. g may be null.
  - `memory1` is output and must be a `char*`. Data will be null terminated, even if the data must be truncated to do so.
  - `memory2` is input and must be an `unsigned int*` holding the length of the buffer in `memory1`.
- G_UTIL_GCAPS_ASSIGN see GAssign(). Provides a method to assign an IP address through gcaps given a Galil MAC address. A valid connection (g) is not necessary, i.e. g may be null.
  - `memory1` is input and must be a `char*` containing the null terminated address that is to be assigned. e.g. `"192.168.0.43"`.
  - `memory2` is input and must be a `char*` containing the null terminated controller MAC address. e.g. `"00:50:4C:20:01:23"`.
- G_UTIL_GCAPS_PING Uses ICMP ping to determine if an IP address is reachable and assigned. Ping sent from the gcaps server. A valid connection (g) is not necessary, i.e. g may be null.
  - `memory1` is input and must be a `char*` containing the null terminated address that is to be pinged. e.g. `"192.168.0.43"`.
  - `memory2` is output and must be an `int*`. The value will be set to zero if the ping times out, and nonzero if a ping reply is returned.

Parameters

memory1	An untyped pointer to data required for request. The data type is defined by the request variable.
memory2	An untyped pointer to data required for request. The data type is defined by the request variable.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See the following functions from gclibo, the open source portion, for implementation of several GUtility() requests.:

- GAddresses()
- GAssign()
- GInfo()
- GIpRequests()
- GSleep()

- GTimeout()
- GVersion()

Referenced by commands(), error(), GAddresses(), GAssign(), GInfo(), GIpRequests(), GListServers(), GPublishServer(), GRemoteConnections(), GServerStatus(), GSetServer(), GSleep(), GTimeout(), GVersion(), and message().

10.175.4.14 GWrite()

```
GCLIB_DLL_EXPORTED GReturn GCALL GWrite (
            GCon g,
            GBufIn buffer,
            GSize buffer_len )
```

Performs a write on the connection.

Parameters

g	Connection's handle.
buffer	The user's write buffer. To send a Galil command, a terminating carriage return is usually required.
buffer_len	The length of the data in the buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values. If G_NO_ERROR is returned, all bytes were written.

Warning

This function is deprecated and will be removed in a future gclib version. Please contact Galil for needs not covered by the other gclib functions.

See x_gread_gwrite.cpp for an example.
Referenced by x_gread_gwrite().

10.176 gclib.h

Go to the documentation of this file.

```
00001
00020 #ifndef I_D48432D9_1FA3_4C7D_B44C_05F8B9000ADF
00021 #define I_D48432D9_1FA3_4C7D_B44C_05F8B9000ADF
00022
00023 //set library visibility for gcc and msvc
00024 #if BUILDING_GCLIB && HAVE_VISIBILITY
00025 #define GCLIB_DLL_EXPORTED __attribute__((__visibility__("default")))
00026 #elif BUILDING_GCLIB && defined _MSC_VER
00027 #define GCLIB_DLL_EXPORTED __declspec(dllexport)
00028 #elif defined _MSC_VER
00029 #define GCLIB_DLL_EXPORTED __declspec(dllimport)
00030 #else
00031 #define GCLIB_DLL_EXPORTED
00032 #endif
00033
00034 #include "gclib_record.h" // Galil data record structs and unions.
00035 #include "gclib_errors.h" // GReturn error code values and strings.
00036
00037 #ifdef _WIN32
00038 #define GCALL __stdcall
00039 #else
00040 #define GCALL
00041 #endif
00042
00043 //#define G_USE_GCOMPOUND //!< GCompound() is not part of the standard gclib release. Contact Galil
      Applications for a special build, http://galil.com/contact.
00044
00045 #ifdef __cplusplus
00046 extern "C" {
00047 #endif
```

```
00048
00049   //Constants for function arguments
00050 #define G_DR 1
00051 #define G_QR 0
00052 #define G_BOUNDS -1
00053 #define G_CR 0
00054 #define G_COMMA 1
00055 #define G_PUBLISH_SERVER 1
00056 #define G_REMOVE_SERVER 0
00057
00058   //Constants for GUtility()
00059 #define G_UTIL_TIMEOUT 1
00060 #define G_UTIL_TIMEOUT_OVERRIDE 2
00061 #define G_USE_INITIAL_TIMEOUT -1
00062 #define G_UTIL_VERSION 128
00063 #define G_UTIL_INFO 129
00064 #define G_UTIL_SLEEP 130
00065 #define G_UTIL_ADDRESSES 131
00066 #define G_UTIL_IPREQUEST 132
00067 #define G_UTIL_ASSIGN 133
00068 #define G_UTIL_DEVICE_INITIALIZE 134
00069 #define G_UTIL_PING 135
00070 #define G_UTIL_ERROR_CONTEXT 136
00071
00072 #define G_UTIL_GCAPS_HOST 256
00073 #define G_UTIL_GCAPS_VERSION 257
00074 #define G_UTIL_GCAPS_KEEPALIVE 258
00075 #define G_UTIL_GCAPS_ADDRESSES 259
00076 #define G_UTIL_GCAPS_IPREQUEST 260
00077 #define G_UTIL_GCAPS_ASSIGN 261
00078 #define G_UTIL_GCAPS_PING 262
00079 #define G_UTIL_GCAPS_LIST_SERVERS 263
00080 #define G_UTIL_GCAPS_PUBLISH_SERVER 264
00081 #define G_UTIL_GCAPS_SET_SERVER 265
00082 #define G_UTIL_GCAPS_SERVER_STATUS 266
00083 #define G_UTIL_GCAPS_REMOTE_CONNECTIONS 267
00084 #define G_UTIL_GCAPS_SERVER_INFO 268
00085 #define G_UTIL_GCAPS_ADDRESSES_GET_REMEMBERED 269
00086 #define G_UTIL_GCAPS_ADDRESSES_SET_REMEMBERED 270
00087
00088   //Convenience constants
00089 #define G_SMALL_BUFFER 1024
00090 #define G_HUGE_BUFFER 524288
00091 #define G_LINE_BUFFER 80
00092
00093   typedef int GReturn;
00094   typedef void* GCon;
00095   typedef unsigned int GSize;
00096   typedef int GOption;
00097   typedef char* GCStringOut;
00098   typedef const char* GCStringIn;
00099   typedef char* GBufOut;
00100   typedef const char* GBufIn;
00101   typedef unsigned char GStatus;
00102   typedef void* GMemory;
00103
00105   GCLIB_DLL_EXPORTED GReturn GCALL GOpen(GCStringIn address, GCon* g);
00152   GCLIB_DLL_EXPORTED GReturn GCALL GClose(GCon g);
00172   GCLIB_DLL_EXPORTED GReturn GCALL GRead(GCon g, GBufOut buffer, GSize buffer_len, GSize* bytes_read);
00193   GCLIB_DLL_EXPORTED GReturn GCALL GWrite(GCon g, GBufIn buffer, GSize buffer_len);
00210   GCLIB_DLL_EXPORTED GReturn GCALL GCommand(GCon g, GCStringIn command, GBufOut buffer, GSize
      buffer_len, GSize* bytes_returned);
00227   GCLIB_DLL_EXPORTED GReturn GCALL GProgramDownload(GCon g, GCStringIn program, GCStringIn
      preprocessor);
00243   GCLIB_DLL_EXPORTED GReturn GCALL GProgramUpload(GCon g, GBufOut buffer, GSize buffer_len);
00256   GCLIB_DLL_EXPORTED GReturn GCALL GArrayDownload(GCon g, const GCStringIn array_name, GOption first,
      GOption last, GCStringIn buffer);
00273   GCLIB_DLL_EXPORTED GReturn GCALL GArrayUpload(GCon g, const GCStringIn array_name, GOption first,
      GOption last, GOption delim, GBufOut buffer, GSize buffer_len);
00290   GCLIB_DLL_EXPORTED GReturn GCALL GRecord(GCon g, union GDataRecord* record, GOption method);
00318   GCLIB_DLL_EXPORTED GReturn GCALL GMessage(GCon g, GCStringOut buffer, GSize buffer_len);
00352   GCLIB_DLL_EXPORTED GReturn GCALL GInterrupt(GCon g, GStatus* status_byte);
00374   GCLIB_DLL_EXPORTED GReturn GCALL GFirmwareDownload(GCon g, GCStringIn filepath);
00395   GCLIB_DLL_EXPORTED GReturn GCALL GUtility(GCon g, GOption request, GMemory memory1, GMemory
      memory2);
00501 #ifdef G_USE_GCOMPOUND
00503   GCLIB_DLL_EXPORTED GReturn GCALL GCompound(GCon g, GCStringIn buffer);
00548 #endif
00549
00550 #ifdef __cplusplus
00551 } //extern "C"
00552 #endif
00553
00554 #endif //I_D48432D9_1FA3_4C7D_B44C_05F8B9000ADF
```

10.177 arrays.c File Reference

```
#include "gclibo.h"
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <math.h>
#include <zlib.h>
```

Data Structures

- struct H_ArrayData

  Structure to create a linked list for array data.

Typedefs

- typedef struct H_ArrayData ArrayNode

Functions

- void H_InitArrayNode (ArrayNode *node)

  Function to initialize the memory of a new node.
- GReturn H_AddArray (ArrayNode *head, char *name, char *data)

  Add an ArrayData node to the linked list.
- void H_FreeArrays (ArrayNode *node)

  Frees all memory downstream of node. After passing list head to this function, all memory is freed and the head node is invalid.
- GReturn H_UploadArrayToList (GCon g, ArrayNode *head, char *name)

  Uploads a particular array and adds it to the linked list.
- GReturn H_CreateArrayNode (ArrayNode *head, char *name)

  Creates a buffer on the heap to write data, and adds it to the linked list.
- GReturn H_ArrayAddElement (ArrayNode *node, GCStringIn element)

  Adds an array element to an array node.
- GReturn H_DownloadArraysFromList (GCon g, ArrayNode *head, int fail)

  Walks through the array linked list, downloading each.
- GReturn H_WriteArrayCsv (ArrayNode *head, GCStringIn file_path)

  After filling the array list, this function is called to write out the CSV.
- GReturn H_ArrayDownloadFromMemory (GCon g, const char *array_data, int fail)

  Helper function to download a block of arrays to the controller.
- GReturn H_DownloadData (GCon g, const char *data, int fail)

  Helper function to send a string of commands to the controller, one at at time.
- char * H_FindSector (char *arr, int arr_size, int index)

  Function that returns a pointer to the start of the specified sector in the GCB data.
- GReturn GCALL GArrayDownloadFile (GCon g, GCStringIn file_path)

  Array download from file.
- GReturn GCALL GArrayUploadFile (GCon g, GCStringIn file_path, GCStringIn names)

  Array upload to file.
- GReturn GCALL GSetupDownloadFile (GCon g, GCStringIn file_path, GOption options, GCStringOut info, GSize info_len)

  Download a saved controller configuration from a file.

10.177.1 Detailed Description

Function calls for uploading and downloading arrays with CSV files. Also contains functions for support of GSetupDownloadFile().

Warning

All helper functions (names beginning with H_) are subject to change without notice

Definition in file arrays.c.

10.177.2 Typedef Documentation

10.177.2.1 ArrayNode

```
typedef struct H_ArrayData ArrayNode
```

Definition at line 35 of file arrays.c.

10.177.3 Function Documentation

10.177.3.1 GArrayDownloadFile()

```
GReturn GCALL GArrayDownloadFile (
            GCon g,
            GCStringIn file_path )
```

Array download from file.
Downloads a csv file containing array data at `file_path`. If the arrays don't exist, they will be dimensioned.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the array file.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_arrays.cpp for an example.
Definition at line 380 of file arrays.c.
References G_BAD_FILE, G_NO_ERROR, H_ArrayDownloadFromMemory(), and vector().
Referenced by x_arrays().

10.177.3.2 GArrayUploadFile()

```
GReturn GCALL GArrayUploadFile (
            GCon g,
            GCStringIn file_path,
            GCStringIn names )
```

Array upload to file.
Uploads the entire controller array table or a subset and saves the data as a csv file specified by `file_path`.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the array file, file will be overwritten if it exists.
names	Null-terminated string containing the arrays to upload, delimited with space. "" or null uploads all arrays listed in LA.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_arrays.cpp for an example.
Definition at line 408 of file arrays.c.
References G_NO_ERROR, GCmdT(), H_FreeArrays(), H_InitArrayNode(), H_UploadArrayToList(), H_WriteArrayCsv(), and vector().
Referenced by x_arrays().

10.177.3.3 GSetupDownloadFile()

```
GReturn GCALL GSetupDownloadFile (
            GCon g,
            GCStringIn file_path,
            GOption options,
            GCStringOut info,
            GSize info_len )
```

Download a saved controller configuration from a file.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the gcb file.
options	Bit mask to determine what configuration data to download. See below for all options.
info	Optional pointer to a buffer to store the controller info. If no info is needed, specify as NULL.
info_len	Length of optional info buffer. If no info is needed, specify as NULL.

Returns

The success status or error code of the function. If the options parameter is set to 0, the return value will be a bit mask indicating which sectors in the specified GCB are not empty. Otherwise, see gclib_errors.h for possible error values.

Note

By default, GSetupDownloadFile() will stop immediately if an error is encountered downloading data. This can be overridden in the options parameter. For example, you may want to override the error if you have a backup from an 8-axis controller and want to restore the parameters for the first 4 axes to a 4-axis controller.

If both info and info_len are not NULL, the controller information will be provided regardless of the options parameter. The options parameter is a bit mask. If options is set to 0, GSetupDownloadFile() will return a bit mask indicating which sectors in the specified GCB are not empty. The following contains a list of all currently available options:

Bit	Value	Function	Description
1	0x0002	Restore parameters	**KPA**, **KIA**, **KDA**, etc...
3	0x0008	Restore variables	Variables are listed by the **LV** command
4	0x0010	Restore arrays	Arrays are listed by the **LA** command
5	0x0020	Restore program	The program is listed by the **LS** command
31	0x8000	Ignore errors	Ignore invalid parameter errors and continue restoring data. GSetupDownloadFile() will still stop immediately if a connection issue or other fatal error is encountered

Usage example:

```
GCon g;
GOption opt = 0;

GCStringOut info;
```

```
GSize info_len = 4096;

GReturn rc = GOpen("192.168.0.50", &g);
if (rc) return rc;

// Call GSetupDownloadFile() with options set to 0 so we can get the non-empty sector bit mask
opt = GSetupDownloadFile(g, "C:\\path\\to\\gcb\\file.gcb", 0, NULL, NULL);

info = (GCStringOut)malloc(sizeof(GCStringOut) * info_len);

// Call GSetupDownloadFile() with the bit mask returned in the previous function call
rc = GSetupDownloadFile(g, "C:\\path\\to\\gcb\\file.gcb", opt, info, info_len);

printf("Info:\n\n%s", info);

GClose(g);

free(info);
return rc;
```

Definition at line 476 of file arrays.c.
References G_BAD_FILE, G_NO_ERROR, GProgramDownload(), H_ArrayDownloadFromMemory(), H_DownloadData(), H_FindSector(), and vector().

10.177.3.4 H_AddArray()

```
GReturn H_AddArray (
            ArrayNode * head,
            char * name,
            char * data )
```

Add an ArrayData node to the linked list.
Definition at line 52 of file arrays.c.
References G_BAD_FULL_MEMORY, G_NO_ERROR, H_InitArrayNode(), and vector().
Referenced by H_CreateArrayNode(), and H_UploadArrayToList().

10.177.3.5 H_ArrayAddElement()

```
GReturn H_ArrayAddElement (
            ArrayNode * node,
            GCStringIn element )
```

Adds an array element to an array node.
Definition at line 113 of file arrays.c.
References G_BAD_FULL_MEMORY, G_NO_ERROR, MAXARRAY, and vector().
Referenced by H_ArrayDownloadFromMemory().

10.177.3.6 H_ArrayDownloadFromMemory()

```
GReturn H_ArrayDownloadFromMemory (
            GCon g,
            const char * array_data,
            int fail )
```

Helper function to download a block of arrays to the controller.
Definition at line 261 of file arrays.c.
References e(), G_NO_ERROR, H_ArrayAddElement(), H_CreateArrayNode(), H_DownloadArraysFromList(), H_FreeArrays(), H_InitArrayNode(), and vector().
Referenced by GArrayDownloadFile(), and GSetupDownloadFile().

10.177.3.7 H_CreateArrayNode()

```
GReturn H_CreateArrayNode (
            ArrayNode * head,
            char * name )
```

Creates a buffer on the heap to write data, and adds it to the linked list.
Definition at line 103 of file arrays.c.
References G_BAD_FULL_MEMORY, H_AddArray(), MAXARRAY, and vector().
Referenced by H_ArrayDownloadFromMemory().

10.177.3.8 H_DownloadArraysFromList()

```
GReturn H_DownloadArraysFromList (
            GCon g,
            ArrayNode * head,
            int fail )
```

Walks through the array linked list, downloading each.

Warning

This function will call DA and DM which modifies the controllers' array table. This should NOT be done while running record array (see RA/RC/RD) or while using the MODBUS array sharing feature (see ME). To prevent any possibility of array table issues, dimension all the arrays used in the applications with the appropriate lengths before use and comment out the *array table modification* section below.

Definition at line 136 of file arrays.c.
References G_BAD_RESPONSE_QUESTION_MARK, G_BOUNDS, G_NO_ERROR, GArrayDownload(), GCmd(), and vector().
Referenced by H_ArrayDownloadFromMemory().

10.177.3.9 H_DownloadData()

```
GReturn H_DownloadData (
            GCon g,
            const char * data,
            int fail )
```

Helper function to send a string of commands to the controller, one at at time.
Definition at line 336 of file arrays.c.
References G_BAD_RESPONSE_QUESTION_MARK, G_NO_ERROR, GCmd(), and vector().
Referenced by GSetupDownloadFile().

10.177.3.10 H_FindSector()

```
char * H_FindSector (
            char * arr,
            int arr_size,
            int index )
```

Function that returns a pointer to the start of the specified sector in the GCB data.
Definition at line 369 of file arrays.c.
References vector().
Referenced by GSetupDownloadFile().

10.177.3.11 H_FreeArrays()

```
void H_FreeArrays (
            ArrayNode * node )
```

Frees all memory downstream of node. After passing list head to this function, all memory is freed and the head node is invalid.
Definition at line 79 of file arrays.c.
References H_FreeArrays(), and vector().
Referenced by GArrayUploadFile(), H_ArrayDownloadFromMemory(), and H_FreeArrays().

10.177.3.12 H_InitArrayNode()

```
void H_InitArrayNode (
            ArrayNode * node )
```

Function to initialize the memory of a new node.
Definition at line 38 of file arrays.c.
References vector().
Referenced by GArrayUploadFile(), H_AddArray(), and H_ArrayDownloadFromMemory().

10.177.3.13 H_UploadArrayToList()

```
GReturn H_UploadArrayToList (
            GCon g,
            ArrayNode * head,
            char * name )
```

Uploads a particular array and adds it to the linked list.
Definition at line 89 of file arrays.c.
References G_BAD_FULL_MEMORY, G_BOUNDS, G_CR, G_NO_ERROR, GArrayUpload(), H_AddArray(), MAXARRAY, and vector().
Referenced by GArrayUploadFile().

10.177.3.14 H_WriteArrayCsv()

```
GReturn H_WriteArrayCsv (
            ArrayNode * head,
            GCStringIn file_path )
```

After filling the array list, this function is called to write out the CSV.
Definition at line 163 of file arrays.c.
References G_BAD_FILE, G_NO_ERROR, and vector().
Referenced by GArrayUploadFile().

10.178 arrays.c

Go to the documentation of this file.

```
00001
00009 #include "gclibo.h"
00010 #include <stdlib.h> //atoi, atof
00011 #include <string.h> //strcpy
00012 #include <stdio.h> //fopen
00013 #include <math.h> //log()
00014
00015 #ifndef G_OMIT_GSETUPDDOWNLOADFILE
00016 #include <zlib.h> //GSetupDownloadFile(), https://www.zlib.net/
00017 #endif
00018
00020 struct H_ArrayData
00021 {
00022   char name[16]; //copy of array name
00023   char* data; //pointer to the ASCII array data
00024   int len; //length of data
00025   int elements; //found data elements for properly dimensioning on download
00026   int index; //access index into data for write/read operations
00027
00028   struct H_ArrayData* next; //pointer to next ArrayNode in the list
00029
00030   //The following fields are only valid on the head node
00031   struct H_ArrayData * tail; //if this node is the head node, last ptr will be maintained for faster
      tail insertion
00032   int count; //if this node is the head node, count will be maintained for total number of arrays
00033   //note, head node also holds array data
00034 };
00035 typedef struct H_ArrayData ArrayNode;
00036
00038 void H_InitArrayNode(ArrayNode* node)
00039 {
00040   node->count = 0;
00041   node->data = 0;
00042   node->index = 0;
00043   node->len = 0;
00044   node->name[0] = 0;
00045   node->next = 0; //null indicates end of list
00046   node->tail = 0;
00047   node->elements = 0;
00048   //could memset to zero...
00049 }
00050
00052 GReturn H_AddArray(ArrayNode* head, char* name, char* data)
00053 {
00054   ArrayNode* node; //the node to fill with data
00055   if (head->count == 0) //no need to malloc, just fill the head
00056     node = head;
00057   else
00058   {
00059     node = malloc(sizeof(ArrayNode));
```

```
00060     if (node) //malloc ok
00061       H_InitArrayNode(node);
00062     else
00063       return G_BAD_FULL_MEMORY; //malloc failed
00064   }
00065
00066   node->data = data; //copy pointer
00067   strcpy(node->name, name); //copy name array
00068   node->len = strlen(node->data); //output of GArrayUpload is null terminated.
00069
00070   head->count++; //count the node we just made
00071   head->tail->next = node; //link the new node. If node == head this breaks the last-node-next-null
      guarantee
00072   node->next = 0; //enforce the last-node-next-null guarantee
00073   head->tail = node; //update head's tail pointer
00074
00075   return G_NO_ERROR;
00076 }
00077
00079 void H_FreeArrays(ArrayNode* node)
00080 {
00081   if (node == 0) return; //recursive exit condition
00082   free(node->data); //free this node's data
00083   H_FreeArrays(node->next); //let downstream nodes recursively free their data
00084   free(node->next); //free the struct this node points to
00085   //no need to free node, the head is declared on the stack
00086 }
00087
00089 GReturn H_UploadArrayToList(GCon g, ArrayNode* head, char* name)
00090 {
00091   GReturn rc = G_NO_ERROR; //return code
00092   char* array_buf; //buffer to hold array as it's uploaded
00093   if (!(array_buf = malloc(MAXARRAY))) //allocate memory for single array upload
00094     return G_BAD_FULL_MEMORY;
00095
00096   if ((rc = GArrayUpload(g, name, G_BOUNDS, G_BOUNDS, G_CR, array_buf, MAXARRAY)) != G_NO_ERROR) //get
      this array's data
00097     return rc;
00098
00099   return H_AddArray(head, name, array_buf); //push the data into the array linked list
00100 }
00101
00103 GReturn H_CreateArrayNode(ArrayNode* head, char* name)
00104 {
00105   char* array_buf; //buffer to hold array data
00106   if (!(array_buf = malloc(MAXARRAY))) //allocate memory for single array upload
00107     return G_BAD_FULL_MEMORY;
00108   array_buf[0] = 0; //null terminate so len is correct when added
00109   return H_AddArray(head, name, array_buf); //push the data into the array linked list
00110 }
00111
00113 GReturn H_ArrayAddElement(ArrayNode* node, GCStringIn element)
00114 {
00115   int len = strlen(element);
00116   if ((len + node->index + 1) >= MAXARRAY) //+1 for \r
00117     return G_BAD_FULL_MEMORY;
00118
00119   strcpy(node->data + node->index, element); //copy the data to the array
00120   node->index += len;
00121   node->data[node->index++] = '\r'; //delim
00122   node->data[node->index] = 0; //null terminate
00123   node->len = node->index; //maintain len field
00124   node->elements++; //count the element just added
00125   return G_NO_ERROR;
00126 }
00127
00129
00136 GReturn H_DownloadArraysFromList(GCon g, ArrayNode* head, int fail)
00137 {
00138   ArrayNode* node = head;
00139   GReturn rc = G_NO_ERROR;
00140   char command[32]; //buffer for holding command calls
00141   while ((node != 0) && (rc == G_NO_ERROR))
00142   {
00143     //*** Start array table modification
00144     //    Deallocate the array in case it's the wrong size.
00145     sprintf(command, "DA %s[]", node->name);
00146     if ((rc = GCmd(g, command)) != G_NO_ERROR)
00147       return rc;
00148
00149     //    Dimension the array with the correct length.
00150     sprintf(command, "DM %s[%i]", node->name, node->elements);
00151     if ((rc = GCmd(g, command)) != G_NO_ERROR)
00152       return rc;
00153     //*** End array table modification
00154
00155     rc = GArrayDownload(g, node->name, G_BOUNDS, G_BOUNDS, node->data); //download the array
```

```
00156     if (!fail && rc == G_BAD_RESPONSE_QUESTION_MARK) rc = G_NO_ERROR; //reset return code if we get a
      ? and fail is 0
00157     node = node->next;
00158   }
00159   return rc;
00160 }
00161
00163 GReturn H_WriteArrayCsv(ArrayNode* head, GCStringIn file_path)
00164 {
00165   if (head->count == 0) //nothing to do
00166     return G_NO_ERROR;
00167
00168   FILE *file; //file pointer
00169   size_t bytes; //length of data to write
00170   size_t bytes_written; //bytes actually written to file
00171   int colcount = 0; //column counter, used to prevent a trailing colon
00172   int data_left = head->count; //counter for number of arrays that still have data left to be written
00173   ArrayNode* node = head; //pointer to an array node in the list
00174
00175   if (!(file = fopen(file_path, "wb"))) //open file for writing, binary mode
00176     return G_BAD_FILE;
00177
00178   //write the header
00179   do
00180   {
00181     bytes = strlen(node->name);
00182     bytes_written = fwrite(node->name, 1, bytes, file);
00183     colcount++;
00184
00185     if (colcount != head->count) //write a comma if it's not the last column
00186     {
00187       bytes_written += fwrite(",", 1, 1, file);
00188       bytes++;
00189     }
00190     else //write a carriage return
00191     {
00192       bytes_written += fwrite("\r", 1, 1, file);
00193       bytes++;
00194     }
00195
00196     if (bytes_written != bytes) //ensure we wrote what we wanted
00197     {
00198       fclose(file);
00199       return G_BAD_FILE;
00200     }
00201     node = node->next;
00202   } while (node != 0);
00203
00204
00205   //now write the data
00206   while (data_left) //continue writing rows as long as arrays have data to write
00207   {
00208     node = head;
00209     colcount = 0;
00210     do //write one row
00211     {
00212       bytes_written = 0;
00213       bytes = 0;
00214       if (node->index != node->len) //data available
00215       {
00216         while ((node->data[node->index] != '\r') //search for the carriage return delim
00217           && (node->index < node->len)) //unless we reach the end
00218         {
00219           if (node->data[node->index] != ' ') //don't keep spaces
00220           {
00221             bytes_written += fwrite(node->data + node->index, 1, 1, file);
00222             bytes++;
00223           }
00224           node->index++;
00225         }
00226
00227         if (node->index == node->len) //reached the end of this data
00228           data_left--; //decrement counter to indicate one less array with data to write
00229
00230         node->index++; //jump over \r delim
00231       }
00232
00233       colcount++; //count the cell we just filled
00234       if (colcount != head->count) //write a comma if it's not the last column
00235       {
00236         bytes_written += fwrite(",", 1, 1, file);
00237         bytes++;
00238       }
00239       else //write a carriage return, even on the last line
00240       {
00241         bytes_written += fwrite("\r", 1, 1, file);
00242         bytes++;
```

```
00243       }
00244
00245       //check for write failure
00246       if (bytes_written != bytes)
00247       {
00248         fclose(file);
00249         return G_BAD_FILE;
00250       }
00251
00252       node = node->next;
00253     } while (node != 0);
00254   } //while (data_left)
00255
00256   fclose(file);
00257   return G_NO_ERROR;
00258 }
00259
00261 GReturn H_ArrayDownloadFromMemory(GCon g, const char* array_data, int fail) {
00262   GReturn rc = G_NO_ERROR;
00263   int pos = 0; //character position in memory
00264
00265   int n = 0; //index into name[]
00266   char name[32]; //buffer to parse array name
00267
00268   int e = 0; //index into element[]
00269   char element[32]; //buffer to parse csv element cell data
00270
00271   char c = array_data[pos]; //first char of array data
00272
00273   //Linked list to hold array data as it's organized for download
00274   ArrayNode head; //first element (list head) lives on this stack
00275   ArrayNode* node; //current node
00276   H_InitArrayNode(&head);
00277   head.tail = &head; //circular reference
00278
00279   while (c != 0)
00280   {
00281     if (c == ',' || c == '\r')
00282     {
00283       if (n)
00284       {
00285         name[n] = 0; //null terminate
00286         n = 0; // next time start filling name from start
00287         H_CreateArrayNode(&head, name);
00288       }
00289
00290       if (c == '\r') { //end of line
00291         pos++;
00292         break; //done reading headers
00293       }
00294     }
00295     else
00296     {
00297       name[n++] = c;
00298     }
00299
00300     pos++;
00301     c = array_data[pos];
00302   }
00303
00304   c = array_data[pos];
00305   node = &head;
00306
00307   while (c != 0)
00308   {
00309     if ((c == ',') || (c == '\r'))
00310     {
00311       if (e) //if anything read into element
00312       {
00313         element[e] = 0; //null terminate
00314         H_ArrayAddElement(node, element);
00315         e = 0; //start writing at start of element on next pass
00316       }
00317       node = node->next; //go to the next array
00318       if (node == 0) node = &head; //wrap around to front
00319     }
00320     else
00321     {
00322       element[e++] = c;
00323     }
00324
00325     pos++;
00326     c = array_data[pos];
00327   }
00328
00329   rc = H_DownloadArraysFromList(g, &head, fail);
00330   H_FreeArrays(&head); // don't forget to free memory
```

```
00331
00332   return rc;
00333 }
00334
00336 GReturn H_DownloadData(GCon g, const char* data, int fail)
00337 {
00338   GReturn rc = G_NO_ERROR;
00339
00340   int pos = 0, n = 0; //position in data, index in cmd[]
00341   char c = data[pos];
00342   char cmd[32]; //buffer for the command to send
00343
00344   while (c != 0)
00345   {
00346     if (c > 31 && c < 127) //"printable" ASCII content
00347     {
00348       cmd[n] = c;
00349       n++;
00350     }
00351     else if (c == '\n')
00352     {
00353       cmd[n] = '\0'; //null terminate
00354       n = 0; //back to beginning for next time
00355       rc = GCmd(g, cmd);
00356       if (rc && (fail || rc != G_BAD_RESPONSE_QUESTION_MARK))
00357         return rc; //if fail is 0 and response is a ?, don't error out
00358       else
00359         rc = G_NO_ERROR; //reset return code
00360     }
00361
00362     pos++;
00363     c = data[pos];
00364   }
00365   return rc;
00366 }
00367
00369 char* H_FindSector(char* arr, int arr_size, int index) {
00370   int i;
00371   for (i = 0; i < arr_size; i++)
00372   {
00373     if ((int)arr[i] == index)
00374       return &arr[i + 1];
00375   }
00376
00377   return NULL;
00378 }
00379
00380 GReturn GCALL GArrayDownloadFile(GCon g, GCStringIn file_path)
00381 {
00382   FILE *file;
00383   GReturn rc = G_NO_ERROR;
00384
00385   if (!(file = fopen(file_path, "rb"))) //open file for reading, binary mode
00386     return G_BAD_FILE;
00387
00388   // Find the length of the file
00389   fseek(file, 0, SEEK_END);
00390   int in_len = ftell(file);
00391   rewind(file);
00392
00393   // Read in all the data to memory
00394   char* in_buf = (char*)malloc(sizeof(char) * in_len);
00395   fread(in_buf, sizeof(char), in_len, file);
00396
00397   fclose(file);
00398
00399   // Send array data to the controller
00400   rc = H_ArrayDownloadFromMemory(g, in_buf, 1);
00401
00402   // Free allocated memory
00403   free(in_buf);
00404
00405   return rc;
00406 }
00407
00408 GReturn GCALL GArrayUploadFile(GCon g, GCStringIn file_path, GCStringIn names)
00409 {
00410
00411   GReturn rc = G_NO_ERROR; //return code
00412   long bytes; //strlen placeholder
00413   char array_names[1024]; //buffer to hold copy of names, or response to list arrays LA
00414   char name[32]; //buffer to hold a single array name
00415   int i, n; //indices
00416   char c; //holder for the char currently being read
00417   int bracket = 0; //increments when [ seen on a line, [ marks the end of the array name
00418
00419   //Linked list to hold array data
```

```
00420   ArrayNode head; //first element (list head) lives on this stack
00421   H_InitArrayNode(&head);
00422   head.tail = &head; //circular reference
00423
00424   if (names == 0) //check for null pointer in arg
00425     bytes = 0;
00426   else
00427     bytes = strlen(strcpy(array_names, names));
00428
00429   if (bytes == 0) //null or "", need to get the arrays from the controller
00430   {
00431     if ((rc = GCmdT(g, "LA", array_names, sizeof(array_names), 0)) != G_NO_ERROR) //Trimming command,
      get names from List Arrays (LA)
00432       return rc; //no mallocs yet, so we can exit without free
00433
00434     bytes = strlen(array_names); //count the response
00435   }
00436
00437   n = 0; //n is name[] index
00438   for (i = 0; i < bytes; i++)
00439   {
00440     c = array_names[i];
00441
00442     if (c == '[')
00443       bracket++; //[ marks the end of the array name
00444
00445     if ((c != ' ') && (c != '\r') && (c != '\n') && !bracket)
00446     {
00447       name[n++] = array_names[i]; //keep the char
00448     }
00449
00450     if ((c == ' ') || (c == '\r') || (i == bytes - 1))
00451     {
00452       if (n) //if we have anything in name
00453       {
00454         name[n] = 0; // null terminate name
00455         n = 0; // next time start filling name from start
00456         bracket = 0; //forget any brackets we've seen
00457
00458         if ((rc = H_UploadArrayToList(g, &head, name)) != G_NO_ERROR) //Add data to list
00459         {
00460           H_FreeArrays(&head); // don't forget to free memory
00461           return rc;
00462         }
00463       }
00464       continue;
00465     }
00466
00467   }
00468
00469   //By here, all the array data is in the linked-list starting at head.
00470   rc = H_WriteArrayCsv(&head, file_path);
00471   H_FreeArrays(&head); // don't forget to free memory
00472   return rc;
00473 }
00474
00475 #ifndef G_OMIT_GSETUPDDOWNLOADFILE
00476 GReturn GCALL GSetupDownloadFile(GCon g, GCStringIn file_path, GOption options, GCStringOut info,
      GSize info_len) {
00477   GReturn rc = G_NO_ERROR;
00478   int fail = 1; //stop on error from controller
00479   FILE *file;
00480   int i;
00481   long in_len; //length of file
00482
00483   file = fopen(file_path, "rb");
00484   if (file == NULL) return G_BAD_FILE;
00485
00486   // Find the length of the file
00487   fseek(file, 0, SEEK_END);
00488   in_len = ftell(file);
00489   rewind(file);
00490
00491   // Read in all the compressed data
00492   unsigned char* in_buf = (unsigned char*)malloc(sizeof(unsigned char) * in_len);
00493   fread(in_buf, sizeof(unsigned char), in_len, file);
00494
00495   fclose(file);
00496
00497   // Determine the uncompressed data length and allocate an array to store it
00498   uLong out_len = (unsigned int)((in_buf[0] << 24) | (in_buf[1] << 16) | (in_buf[2] << 8) | (in_buf[3]));
00499   unsigned char* out_buf = (unsigned char*)malloc(sizeof(unsigned char) * out_len);
00500
00501   // Uncompress the data and store it in the allocated array
00502   rc = uncompress(out_buf, &out_len, in_buf + 4, in_len - 4);
00503   free(in_buf); //we're done with the compressed version
00504   if (rc)
```

```
00505   {
00506     free(out_buf);
00507     return G_BAD_FILE;
00508   }
00509
00510   // Return the controller info if an info array pointer is provided
00511   if (info != NULL && info_len > 0) {
00512     char* arr = H_FindSector((char*)out_buf, out_len, 1);
00513     if (arr == NULL)
00514     {
00515       free(out_buf);
00516       return G_BAD_FILE;
00517     }
00518
00519     for (i = 0; i < info_len; i++)
00520     {
00521       info[i] = arr[i];
00522
00523       if (arr[i] == '\0') //end of info sector
00524         break;
00525       else if (i == (info_len - 1))
00526       {
00527         info[i] = '\0'; //force null termination
00528       }
00529     }
00530   }
00531
00532   // Determine which GCB sectors contain info and return the bit mask
00533   if (options == 0)
00534   {
00535     for (i = 1; i < 7; i++)
00536     {
00537       if (i == 1 || i == 3)
00538         continue;  // Skip reserved sectors
00539
00540       char* arr = H_FindSector((char*)out_buf, out_len, i);
00541       if (arr == NULL)
00542         continue;  // Continue if sector not found
00543
00544       if (arr[0] != '\0') //non-empty sector
00545         rc += (1 << (i - 1));
00546     }
00547     return rc;
00548   }
00549
00550   // Determine if ignore non-fatal errors bit is set
00551   if (options & 0x8000)
00552     fail = 0;
00553
00554   char* sector;
00555   // Options processing
00556
00557   int len = out_len - 1; //stop one before the end for null termination
00558   for (i = 0; i < len; i++)
00559   {
00560     sector = (char*)out_buf + i + 1; //start of possible sector
00561     if (*sector == 0) //all data is in ascii, so null indicates an empty sector
00562     {
00563       ++i; //jump over null
00564       continue;
00565     }
00566
00567     switch (out_buf[i])
00568     {
00569     case 0x02: //parameters sector start
00570       if (options & 0x0002) //bit 1
00571         rc = H_DownloadData(g, sector, fail);
00572
00573       break;
00574
00575     case 0x04: //variables sector start
00576       if (options & 0x0008) //bit 3
00577         rc = H_DownloadData(g, sector, fail);
00578
00579       break;
00580
00581     case 0x05: //arrays sector start
00582       if (options & 0x0010) //bit 4
00583         rc = H_ArrayDownloadFromMemory(g, sector, fail);
00584
00585       break;
00586
00587     case 0x06: //program sector start
00588       if (options & 0x0020) //bit 5
00589         rc = GProgramDownload(g, sector, NULL);
00590
00591       break;
```

```
00592     }//switch
00593
00594     if (rc)
00595       break; //exit for
00596   }//for
00597
00598   // Free allocated memory
00599   free(out_buf);
00600   return rc;
00601 }
00602 #endif
```

10.179 gclibo.c File Reference

```
#include "gclibo.h"
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <math.h>
#include <time.h>
```

Functions

- GCLIB_DLL_EXPORTED void GCALL GSleep (unsigned int timeout_ms)

  Uses GUtility() and G_UTIL_SLEEP to provide a blocking sleep call which can be useful for timing-based chores.
- GCLIB_DLL_EXPORTED GReturn GCALL GVersion (GCStringOut ver, GSize ver_len)

  Uses GUtility(), G_UTIL_VERSION and G_UTIL_GCAPS_VERSION to provide the library and gcaps version numbers.
- GCLIB_DLL_EXPORTED GReturn GCALL GInfo (GCon g, GCStringOut info, GSize info_len)

  Uses GUtility() and G_UTIL_INFO to provide a useful connection string.
- GCLIB_DLL_EXPORTED GReturn GCALL GAddresses (GCStringOut addresses, GSize addresses_len)

  Uses GUtility(), G_UTIL_GCAPS_ADDRESSES or G_UTIL_ADDRESSES to provide a listing of all available connection addresses.
- GCLIB_DLL_EXPORTED GReturn GCALL GTimeout (GCon g, short timeout_ms)

  Uses GUtility() and G_UTIL_TIMEOUT_OVERRIDE to set the library timeout.
- GCLIB_DLL_EXPORTED GReturn GCALL GAssign (GCStringIn ip, GCStringIn mac)

  Uses GUtility(), G_UTIL_GCAPS_ASSIGN or G_UTIL_ASSIGN to assign an IP address over the Ethernet to a controller at a given MAC address.
- GCLIB_DLL_EXPORTED GReturn GCALL GIpRequests (GCStringOut requests, GSize requests_len)

  Uses GUtility(), G_UTIL_GCAPS_IPREQUEST or G_UTIL_IPREQUEST to provide a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.
- GCLIB_DLL_EXPORTED GReturn GCALL GSetServer (GCStringIn server_name)

  Uses GUtility(), G_UTIL_GCAPS_SET_SERVER to set the new active server.
- GCLIB_DLL_EXPORTED GReturn GCALL GServerStatus (GCStringOut status, GSize status_len)

  Uses GUtility(), G_UTIL_GCAPS_SERVER_STATUS to get information on the local server name and if it is published to the local network.
- GCLIB_DLL_EXPORTED GReturn GCALL GListServers (GCStringOut servers, GSize servers_len)

  Uses GUtility(), G_UTIL_GCAPS_LIST_SERVERS to provide a list of all available gcaps services on the local network.
- GCLIB_DLL_EXPORTED GReturn GCALL GPublishServer (GCStringIn name, GOption publish, GOption save)

  Uses GUtility(), G_UTIL_GCAPS_PUBLISH_SERVER to publish local gcaps server to the local network.
- GCLIB_DLL_EXPORTED GReturn GCALL GRemoteConnections (GCStringOut connections, GSize connections_length)

  Uses GUtility(), G_UTIL_GCAPS_REMOTE_CONNECTIONS to get a list of remote addresses connected to the local server.
- GCLIB_DLL_EXPORTED GReturn GCALL GCmd (GCon g, GCStringIn command)

Wrapper around GCommand for use when the return value is not desired.

- GCLIB_DLL_EXPORTED GReturn GCALL GCmdT (GCon g, GCStringIn command, GCStringOut trimmed_response, GSize response_len, GCStringOut *front)

  Wrapper around GCommand that trims the response.
- GCLIB_DLL_EXPORTED GReturn GCALL GCmdI (GCon g, GCStringIn command, int *value)

  Wrapper around GCommand that provides the return value of a command parsed into an int.
- GCLIB_DLL_EXPORTED GReturn GCALL GCmdD (GCon g, GCStringIn command, double *value)

  Wrapper around GCommand that provides the return value of a command parsed into a double.
- GCLIB_DLL_EXPORTED GReturn GCALL GMotionComplete (GCon g, GCStringIn axes)

  Blocking call that returns once all axes specified have completed their motion.
- GCLIB_DLL_EXPORTED GReturn GCALL GWaitForBool (GCon g, GCStringIn predicate, int trials)

  Blocking call that returns when the controller evaluates the predicate as true.
- GCLIB_DLL_EXPORTED GReturn GCALL GRecordRate (GCon g, double period_ms)

  Sets the asynchronous data record to a user-specified period via `DR`.
- GCLIB_DLL_EXPORTED GReturn GCALL GProgramDownloadFile (GCon g, GCStringIn file_path, GCStringIn preprocessor)

  Program download from file.
- GCLIB_DLL_EXPORTED GReturn GCALL GProgramUploadFile (GCon g, GCStringIn file_path)

  Program upload to file.
- GCLIB_DLL_EXPORTED void GCALL GError (GReturn rc, GCStringOut error, GSize error_len)

  Provides a human-readable description string for return codes.

10.179.1 Detailed Description

Partial implementation of gclibo.h
Definition in file gclibo.c.

10.179.2 Macro Definition Documentation

10.179.2.1 _CRT_SECURE_NO_WARNINGS

```
#define _CRT_SECURE_NO_WARNINGS
```

Definition at line 14 of file gclibo.c.

10.179.3 Function Documentation

10.179.3.1 GAddresses()

```
GReturn GCALL GAddresses (
            GCStringOut addresses,
            GSize addresses_len )
```

Uses GUtility(), G_UTIL_GCAPS_ADDRESSES or G_UTIL_ADDRESSES to provide a listing of all available connection addresses.

Note

Serial ports are listed, e.g. COM1. Upon open, it may be necessary to specify a baud rate for the controller, e.g. `--baud 19200`. Default baud is 115200. See GOpen().

Parameters

addresses	Buffer to hold the output string. Buffer will be null terminated, even if the data must be truncated to do so. See below for more information.
addresses_len	Length of buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

If gcaps is available, the listing will come from the server via G_UTIL_GCAPS_ADDRESSES. In the absence of the server, gclib will use G_UTIL_ADDRESSES to generate the list.

- Ethernet controllers will be listed as *ip_address, revision_report, network_adapter_name, network_adapter_ip_address*. If an IP address is unreachable via ping, the address will be in parentheses.
- PCI controllers will be listed by their identifier, e.g. GALILPCI1.
- Serial ports will be listed by their identifier, e.g. COM1.

```
10.1.3.91, DMC4020 Rev 1.2e, LAN, 10.1.3.10
192.168.0.63, DMC4040 Rev 1.2f, Static, 192.168.0.41
(192.0.0.42), RIO47102 Rev 1.1j, Static, 192.168.0.41
GALILPCI1
COM1
COM2
```

Note

GAddresses() will take up to 1 second to look for gcaps.

See x_examples.cpp for an example.
Definition at line 54 of file gclibo.c.
References G_NO_ERROR, G_UTIL_ADDRESSES, G_UTIL_GCAPS_ADDRESSES, GUtility(), and vector().
Referenced by remote_client().

10.179.3.2 GAssign()

```
GReturn GCALL GAssign (
            GCStringIn ip,
            GCStringIn mac )
```

Uses GUtility(), G_UTIL_GCAPS_ASSIGN or G_UTIL_ASSIGN to assign an IP address over the Ethernet to a controller at a given MAC address.

Parameters

ip	The null-terminated ip address to assign. The hardware should not yet have an IP address.
mac	The null-terminated MAC address of the hardware.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

On Linux and Mac, the desired IP address will be pinged prior to the assignment. If the ping is returned, GAssign() will return G_GCLIB_UTILITY_IP_TAKEN.
If gcaps is available, the assign will be performed from the server via G_UTIL_GCAPS_ASSIGN. gcaps will remember the assignment and will automatically assign the desired IP address should the controller ever request one again, e.g. after a controller master reset. To clear the remembered IP address from gcaps, call GAssign() with a blank string in place of the ip address. To remove all remembered ip addresses, specfify a blank string for the mac address.
In the absence of the server, gclib will use G_UTIL_ASSIGN to assign. GAssign() will take up to 1 second to look for gcaps. When not using gcaps, Linux/OS X users must be root to use GAssign() and have UDP access to send on port 68.
See x_examples.cpp for an example.
Definition at line 70 of file gclibo.c.
References G_GCLIB_UTILITY_IP_TAKEN, G_NO_ERROR, G_UTIL_ASSIGN, G_UTIL_GCAPS_ASSIGN, G_UTIL_GCAPS_PING, G_UTIL_PING, GUtility(), and vector().
Referenced by ip_assigner().

10.179.3.3 GCmd()

```
GReturn GCALL GCmd (
            GCon g,
            GCStringIn command )
```

Wrapper around GCommand for use when the return value is not desired.
The returned data is still checked for error, e.g. ? or timeout, but is not brought out through the prototype.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 237 of file gclibo.c.
References G_SMALL_BUFFER, GCommand(), and vector().
Referenced by check_interrupts(), commands(), contour(), GRecordRate(), H_DownloadArraysFromList(), H_DownloadData(), ip_assigner(), jog(), load_buf(), load_buffer(), message(), motion_complete(), position_tracking(), record_position(), vector(), x_arrays(), x_dr_motioncomplete(), x_ei_motioncomplete(), x_gcommand(), x_ginterrupt(), x_gmessage(), x_gmotioncomplete(), x_gread_gwrite(), x_grecord(), x_nonblocking(), and x_programs().

10.179.3.4 GCmdD()

```
GReturn GCALL GCmdD (
            GCon g,
            GCStringIn command,
            double * value )
```

Wrapper around GCommand that provides the return value of a command parsed into a double.
Use this function to retrieve the full Galil 4.2 range, e.g. for a variable value with fractional data, or the value of an Analog input or Output.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.
value	Pointer to a double that will be filled with the return value.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 289 of file gclibo.c.
References G_NO_ERROR, G_SMALL_BUFFER, GCommand(), and vector().
Referenced by commands(), GRecordRate(), and x_gcommand().

10.179.3.5 GCmdI()

```
GReturn GCALL GCmdI (
            GCon g,
            GCStringIn command,
            int * value )
```

Wrapper around GCommand that provides the return value of a command parsed into an int.
Use this function to get most values including TP, RP, TE, Digital I/O states, etc.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.
value	Pointer to an int that will be filled with the return value.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 278 of file gclibo.c.
References G_NO_ERROR, G_SMALL_BUFFER, GCommand(), and vector().
Referenced by commands(), contour(), record_position(), vector(), x_dr_motioncomplete(), x_gcommand(), x_gread_gwrite(), x_grecord(), and x_programs().

10.179.3.6 GCmdT()

```
GReturn GCALL GCmdT (
            GCon g,
            GCStringIn command,
            GCStringOut trimmed_response,
            GSize response_len,
            GCStringOut * front )
```

Wrapper around GCommand that trims the response.
For use when the return value is desired, is ASCII (not binary), and the response should be trimmed of trailing colon, whitespace, and optionally leading space.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.
trimmed_response	The trimmed response from the controller. Trailing space is trimmed by null terminating any trailing spaces, carriage returns, or line feeds.
response_len	The length of the trimmed_response buffer.
front	If non-null, upon return ∗front will point to the first non-space character in trimmed_response. This allows trimming the front of the string without modifying the user's buffer pointer, which may be allocated on the heap.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 243 of file gclibo.c.
References G_NO_ERROR, GCommand(), and vector().
Referenced by commands(), GArrayUploadFile(), GRecordRate(), motion_complete(), x_gcommand(), x_ginterrupt(), x_gmotioncomplete(), and x_grecord().

10.179.3.7 GError()

```
void GCALL GError (
            GReturn rc,
            GCStringOut error,
            GSize error_len )
```

Provides a human-readable description string for return codes.

Parameters

rc	The return code to lookup.
error	The buffer to fill with the error text. Buffer will be null terminated, even if the data must be truncated to do so.
error_len	The length of the error buffer.

See x_examples.cpp for an example.
Definition at line 459 of file gclibo.c.
References error(), G_ALREADY_OPEN, G_ARRAY_NOT_DIMENSIONED, G_BAD_ADDRESS, G_BAD_FILE, G_BAD_FIRMWARE_LOAD, G_BAD_FULL_MEMORY, G_BAD_LOST_DATA, G_BAD_RESPONSE_QUESTION_MARK, G_BAD_VALUE_RANGE, G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND, G_CONNECTION_NOT_ESTABLISHED, G_DATA_RECORD_ERROR, G_FIRMWARE_LOAD_NOT_SUPPORTED, G_GCAPS_OPEN_ERROR, G_GCAPS_SUBSCRIPTION G_GCLIB_ERROR, G_GCLIB_NON_BLOCKING_READ_EMPTY, G_GCLIB_UTILITY_ERROR, G_GCLIB_UTILITY_IP_TAKEN, G_ILLEGAL_DATA_IN_PROGRAM, G_INVALID_PREPROCESSOR_OPTIONS, G_NO_ERROR, G_OPEN_ERROR, G_READ_ERROR, G_TIMEOUT, G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT, G_UNSUPPORTED_FUNCTION, G_WRITE_ERROR, and vector().
Referenced by error(), gclib::GArrayDownload(), gclib::GArrayDownloadFile(), gclib::GArrayUpload(), gclib::GArrayUploadFile(), gclib::GAssign(), gclib::GCommand(), gclib::GFirmwareDownload(), gclib::GListServers(), gclib::GMotionComplete(), gclib::GOpen(), gclib::GProgramDownload(), gclib::GProgramDownloadFile(), gclib::GProgramUpload(), gclib::GProgramUploadFile() gclib::GPublishServer(), gclib::GRecord< T >(), gclib::GRecordRate(), gclib::GRemoteConnections(), gclib::GServerStatus(), gclib::GSetServer(), gclib::GSetupDownloadFile(), and gclib::GWrite().

10.179.3.8 GInfo()

```
GReturn GCALL GInfo (
            GCon g,
            GCStringOut info,
            GSize info_len )
```

Uses GUtility() and G_UTIL_INFO to provide a useful connection string.

Parameters

g	Connection's handle.
info	Buffer to hold the output string. Buffer will be null terminated, even if the data must be truncated to do so.
info_len	Length of buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

The response is *address, revision_report, serial_number*. For example:

```
COM2, RIO47102 Rev 1.1j, 37290
```

See x_examples.cpp for an example.
Definition at line 49 of file gclibo.c.
References G_UTIL_INFO, GUtility(), and vector().
Referenced by ip_assigner().

10.179.3.9 GIpRequests()

```
GReturn GCALL GIpRequests (
            GCStringOut requests,
            GSize requests_len )
```

Uses GUtility(), G_UTIL_GCAPS_IPREQUEST or G_UTIL_IPREQUEST to provide a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.

Parameters

requests	The buffer to hold the list of requesting controllers. Data will be null terminated, even if the data must be truncated to do so. See below for more information.
requests_len	The length of the requests buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

GIpRequests() will block up to 5 seconds while listening for requests.
If gcaps is available, the listing will come from the server via G_UTIL_GCAPS_IPREQUEST. In the absence of the server, gclib will use G_UTIL_IPREQUEST to generate the list. GIpRequests() will take up to 1 second to look for gcaps. When not using gcaps, Linux/OS X users must be root to use GIpRequests() and have UDP access to bind and listen on port 67.
Each line of the returned data will be of the form *model, serial_number, MAC_address, network_adapter_name, network_adapter_ip_address, remembered_ip_assignment*. See GAssign() for more infomation about remembered IP assignments. The following is an example output.

```
DMC2000, 34023, 00:50:4C:00:84:E7, enp5s0, 192.168.42.92, 192.168.42.200
DMC2105, 7, 00:50:4C:58:00:07, enp5s0, 192.168.42.92, 0.0.0.0
DMC2105, 13, 00:50:4C:58:00:0D, enp5s0, 192.168.42.92, 0.0.0.0
```

See x_examples.cpp for an example.
Definition at line 106 of file gclibo.c.
References G_NO_ERROR, G_UTIL_GCAPS_IPREQUEST, G_UTIL_IPREQUEST, GSleep(), GUtility(), and vector().
Referenced by ip_assigner().

10.179.3.10 GListServers()

```
GReturn GCALL GListServers (
            GCStringOut servers,
            GSize servers_len )
```

Uses GUtility(), G_UTIL_GCAPS_LIST_SERVERS to provide a list of all available gcaps services on the local network.

Note

This function is only available on Windows 10 and Linux.

Parameters

servers	The buffer to hold the list of available gcaps servers
servers_len	The length of the servers buffer

This function is used to find a list of available gcaps servers that have made themselves "Discoverable".
The list of available servers are separated by a newline '\n' character.

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 169 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_LIST_SERVERS, GUtility(), and vector().
Referenced by remote_client().

10.179.3.11 GMotionComplete()

```
GReturn GCALL GMotionComplete (
            GCon g,
            GCStringIn axes )
```

Blocking call that returns once all axes specified have completed their motion.

Note

This function uses a profiled motion indicator, not the position of the encoder. E.G. see the difference between AM (profiled) and MC (encoder-based).

Although using the _BGm operand is the most generally compatible method, there are higher-performance ways to check for motion complete by using the data record, or interrupts. See examples x_dr_motioncomplete() and x_ei_motioncomplete().

Parameters

g	Connection's handle.
axes	A null-terminated string containing a multiple-axes mask. Every character in the string should be a valid argument to MG_BGm, i.e. XYZWABCDEFGHST.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gmotioncomplete.cpp for an example.
Definition at line 300 of file gclibo.c.
References G_NO_ERROR, GWaitForBool(), and vector().
Referenced by contour(), jog(), position_tracking(), vector(), and x_gmotioncomplete().

10.179.3.12 GProgramDownloadFile()

```
GReturn GCALL GProgramDownloadFile (
            GCon g,
            GCStringIn file_path,
            GCStringIn preprocessor )
```

Program download from file.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the program file.
preprocessor	Options string for preprocessing the program before sending it to the controller. See GProgramDownload().

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_programs.cpp for an example.
Definition at line 387 of file gclibo.c.
References G_BAD_FILE, G_BAD_FULL_MEMORY, G_NO_ERROR, GProgramDownload(), and vector().
Referenced by x_programs().

10.179.3.13 GProgramUploadFile()

```
GReturn GCALL GProgramUploadFile (
            GCon g,
            GCStringIn file_path )
```

Program upload to file.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the program file, file will be overwritten if it exists.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_programs.cpp for an example.
Definition at line 430 of file gclibo.c.
References G_BAD_FILE, G_BAD_FULL_MEMORY, G_NO_ERROR, GProgramUpload(), MAXPROG, and vector().
Referenced by x_programs().

10.179.3.14 GPublishServer()

```
GReturn GCALL GPublishServer (
            GCStringIn name,
            GOption publish,
            GOption save )
```

Uses GUtility(), G_UTIL_GCAPS_PUBLISH_SERVER to publish local gcaps server to the local network.

Note

This function is only available on Windows 10 and Linux.

Parameters

name	The name of the server to publish or remove
publish	Option to publish or remove server from network
save	Option to save this configuration for future reboots

This function is used to make your local gcaps server "Discoverable" or "Invisible"
publish Option:
Set to 1 to publish server to the network and make "Discoverable"
Set to 0 to remove server from the network and make "Invisible"

save Option:
Set to 1 to save the configuration for future reboots of the server
Set to 0 to use this configuration once, and not overwrite previous server settings

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 189 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_PUBLISH_SERVER, GUtility(), and vector().
Referenced by remote_server().

10.179.3.15 GRecordRate()

```
GReturn GCALL GRecordRate (
            GCon g,
            double period_ms )
```

Sets the asynchronous data record to a user-specified period via DR.
Takes TM and product type into account and sets the DR period to the period requested by the user, if possible.

Parameters

g	Connection's handle.
period_ms	Period, in milliseconds, to set up for the asynchronous data record.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_grecord.cpp for an example.
Definition at line 342 of file gclibo.c.
References G_NO_ERROR, G_SMALL_BUFFER, GCmd(), GCmdD(), GCmdT(), and vector().
Referenced by x_dr_motioncomplete(), x_grecord(), and x_nonblocking().

10.179.3.16 GRemoteConnections()

```
GReturn GCALL GRemoteConnections (
            GCStringOut connections,
            GSize connections_length )
```

Uses GUtility(), G_UTIL_GCAPS_REMOTE_CONNECTIONS to get a list of remote addresses connected to the local server.

Note

This function is only available on Windows 10 and Linux.

Parameters

connections	The buffer to hold the list of remote IP addresses currently connected to your hardware
connections_len	The length of the connections buffer

This function is used to find a list of IP Addresses of machines that currently have open connections to your local hardware. If another user sets your local server as their active server, and then opens a connection to your hardware, their IP Address will appear in this list.
The list of IP addresses are separated by a newline '\n' character.

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 217 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_REMOTE_CONNECTIONS, GUtility(), and vector().

10.179.3.17 GServerStatus()

```
GReturn GCALL GServerStatus (
```

```
            GCStringOut status,
            GSize status_len )
```

Uses GUtility(), G_UTIL_GCAPS_SERVER_STATUS to get information on the local server name and if it is published to the local network.

Note

This function is only available on Windows 10 and Linux.

Parameters

status	The buffer to hold the status of the local gcaps server
status_len	The length of the status buffer

This function is used to find the status of your local gcaps server. Use this function to determine the name your server is currently using, and whether or not your gcaps server is currently set to "Discoverable" or "Invisible"
The status buffer will be filled in the form of "[Server Name], [Discoverable]"
For example, for a server with the name "Example Server" that is set to "Discoverable", the status buffer would contain "Example Server, true".

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 149 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_SERVER_STATUS, GUtility(), and vector().

10.179.3.18 GSetServer()

```
GReturn GCALL GSetServer (
            GCStringIn server_name )
```

Uses GUtility(), G_UTIL_GCAPS_SET_SERVER to set the new active server.

Note

This function is only available on Windows 10 and Linux.

Parameters

server_name	The name of the server to set as your new active server.

Use this function in conjunction with GListServers(). Choose a name received from GListServers() to set as your new active server.
After setting a new active server, all gclib calls will route through that new active server, unless explicitly noted otherwise.
To set your active server back to your local server, simply pass "Local" to GSetServer():

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 128 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_SET_SERVER, GUtility(), and vector().

Referenced by remote_client().

10.179.3.19 GSleep()

```
void GCALL GSleep (
            unsigned int timeout_ms )
```

Uses GUtility() and G_UTIL_SLEEP to provide a blocking sleep call which can be useful for timing-based chores.

Parameters

timeout_ms	The timeout, in milliseconds, to block before returning.

See GWaitForBool() for an example.
Definition at line 24 of file gclibo.c.
References G_UTIL_SLEEP, and GUtility().
Referenced by contour(), GIpRequests(), GWaitForBool(), record_position(), vector(), x_gread_gwrite(), and x_programs().

10.179.3.20 GTimeout()

```
GReturn GCALL GTimeout (
            GCon g,
            short timeout_ms )
```

Uses GUtility() and G_UTIL_TIMEOUT_OVERRIDE to set the library timeout.

Parameters

g	Connection's handle.
timeout_ms	The value to be used for the timeout. Use `G_USE_INITIAL_TIMEOUT` to set the timeout back to the initial GOpen() value, `--timeout`.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp and x_gread_gwrite.cpp for examples.
Definition at line 65 of file gclibo.c.
References G_UTIL_TIMEOUT_OVERRIDE, and GUtility().
Referenced by motion_complete(), x_gcommand(), x_ginterrupt(), x_gmessage(), x_gread_gwrite(), and x_nonblocking().

10.179.3.21 GVersion()

```
GReturn GCALL GVersion (
            GCStringOut ver,
            GSize ver_len )
```

Uses GUtility(), G_UTIL_VERSION and G_UTIL_GCAPS_VERSION to provide the library and gcaps version numbers.

Parameters

ver	Buffer to hold the output string. Buffer will be null terminated, even if the data must be truncated to do so.
ver_len	Length of buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

The version number of gclib is provided first. If the gcaps server can be found, its version will be provided after a space.
Example with gcaps version.

```
154.190.329 1.0.0.82
```

Example with gclib version only.

```
154.190.329
```

Note

GVersion() will take up to 1 second to look for gcaps.

See x_examples.cpp for an example.
Definition at line 29 of file gclibo.c.
References G_NO_ERROR, G_UTIL_GCAPS_VERSION, G_UTIL_VERSION, GUtility(), and vector().

10.179.3.22 GWaitForBool()

```
GReturn GCALL GWaitForBool (
            GCon g,
            GCStringIn predicate,
            int trials )
```

Blocking call that returns when the controller evaluates the predicate as true.
Polls the message command (MG) to check the value of predicate. Polling will continue until the controller responds with a nonzero value or the number of polling trials is reached.
The amount of time until the function fails with G_GCLIB_POLLING_FAILED is roughly (trials ∗ POLLINGINTERVAL) milliseconds.

Parameters

g	Connection's handle.
predicate	A null-terminated string containing the predicate to be polled. The predicate will be enclosed in parentheses and used in the command `MG (predicate)` to return the value.
trials	The number of polling cycles to perform looking for a nonzero value. Use -1 to poll indefinitely.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See GMotionComplete() for an example.
Definition at line 318 of file gclibo.c.
References G_GCLIB_POLLING_FAILED, G_LINE_BUFFER, G_NO_ERROR, G_SMALL_BUFFER, GCommand(), GSleep(), POLLINGINTERVAL, and vector().
Referenced by GMotionComplete().

10.180 gclibo.c

Go to the documentation of this file.

```
00001
00011 #include "gclibo.h"
00012
00013 #ifndef _CRT_SECURE_NO_WARNINGS
00014 #define _CRT_SECURE_NO_WARNINGS //use traditional C calls like strncpy()
00015 #endif
00016
00017 #include <stdlib.h> //atoi, atof
00018 #include <string.h> //strcpy
```

```
00019 #include <stdio.h> //fopen
00020 #include <math.h> //log()
00021 #include <time.h>
00022
00023
00024 void GCALL GSleep(unsigned int timeout_ms)
00025 {
00026   GUtility(0, G_UTIL_SLEEP, &timeout_ms, 0);
00027 }
00028
00029 GReturn GCALL GVersion(GCStringOut ver, GSize ver_len)
00030 {
00031   int str_len;
00032   GReturn rc;
00033   if ((rc = GUtility(0, G_UTIL_VERSION, ver, &ver_len)) != G_NO_ERROR)
00034     return rc;
00035
00036 #ifdef G_USE_GCAPS
00037   str_len = strlen(ver) + 1;
00038   ver_len -= str_len;
00039   if (ver_len > 0
00040     && GUtility(0, G_UTIL_GCAPS_VERSION, ver + str_len, &ver_len) == G_NO_ERROR)
00041   {
00042     ver[str_len - 1] = ' '; //add a delimiter
00043   }
00044 #endif
00045
00046   return rc;
00047 }
00048
00049 GReturn GCALL GInfo(GCon g, GCStringOut info, GSize info_len)
00050 {
00051   return GUtility(g, G_UTIL_INFO, info, &info_len);
00052 }
00053
00054 GReturn GCALL GAddresses(GCStringOut addresses, GSize addresses_len)
00055 {
00056 #ifdef G_USE_GCAPS
00057   GReturn rc;
00058   if (G_NO_ERROR == (rc = GUtility(0, G_UTIL_GCAPS_ADDRESSES, addresses, &addresses_len)))
00059     return rc;
00060 #endif
00061
00062   return GUtility(0, G_UTIL_ADDRESSES, addresses, &addresses_len);
00063 }
00064
00065 GReturn GCALL GTimeout(GCon g, short timeout_ms)
00066 {
00067   return GUtility(g, G_UTIL_TIMEOUT_OVERRIDE, &timeout_ms, 0);
00068 }
00069
00070 GReturn GCALL GAssign(GCStringIn ip, GCStringIn mac)
00071 {
00072   /*
00073   *  On Linux and Apple, the IP address is pinged prior to assigning.
00074   *  On Windows, pinging first can make the arp table stale, and the
00075   *  IP address unreachable for several seconds. We skip ping so that
00076   *  we can immediately connect.
00077   */
00078
00079   GReturn rc;
00080   int reply = 0; //ping reply is nonzero
00081
00082 #ifdef G_USE_GCAPS
00083
00084 #if defined(__linux__) || defined(__APPLE__)
00085   if (strlen(ip)) //can use null string to remove IP assignment from gcaps DHCP feature
00086   {
00087     GUtility(0, G_UTIL_GCAPS_PING, (void*)ip, &reply); //ping to see if IP address is already taken
00088     if (reply)
00089       return G_GCLIB_UTILITY_IP_TAKEN;
00090   }
00091 #endif
00092
00093   if (G_NO_ERROR == (rc = GUtility(0, G_UTIL_GCAPS_ASSIGN, (void*)ip, (void*)mac)))
00094     return rc;
00095 #endif
00096
00097 #if defined(__linux__) || defined(__APPLE__)
00098   GUtility(0, G_UTIL_PING, (void*)ip, &reply);  //ping to see if IP address is already taken
00099   if (reply)
00100     return G_GCLIB_UTILITY_IP_TAKEN;
00101 #endif
00102
00103   return GUtility(0, G_UTIL_ASSIGN, (void*)ip, (void*)mac);
00104 }
00105
```

```
00106 GReturn GCALL GIpRequests(GCStringOut requests, GSize requests_len)
00107 {
00108   GReturn rc;
00109 #ifdef G_USE_GCAPS
00110   rc = GUtility(0, G_UTIL_GCAPS_IPREQUEST, requests, &requests_len);
00111   if (rc == G_NO_ERROR)
00112   {
00113     if (requests_len == 0 || strlen(requests))
00114       return rc;
00115     else
00116     {
00117       //try once more after a 5 second wait
00118       GSleep(5000);
00119       return GUtility(0, G_UTIL_GCAPS_IPREQUEST, requests, &requests_len);
00120     }
00121   }
00122 #endif
00123
00124   //non-gcaps version opens port listens for 5 seconds, and closes
00125   return GUtility(0, G_UTIL_IPREQUEST, requests, &requests_len);
00126 }
00127
00128 GReturn GCALL GSetServer(GCStringIn server_name)
00129 {
00130   GReturn rc;
00131 #ifdef G_USE_GCAPS
00132   rc = GUtility(0, G_UTIL_GCAPS_SET_SERVER, (void*)server_name, 0);
00133 #else
00134   rc = G_GCAPS_OPEN_ERROR;
00135 #endif
00136
00137 #ifdef GCLIB_LOGGING
00138   if (rc != G_NO_ERROR)
00139   {
00140     char buf[256];
00141     sprintf(buf, "GSetServer: %d", rc);
00142     LogMsg(buf);
00143   }
00144 #endif
00145
00146   return rc;
00147 }
00148
00149 GReturn GCALL GServerStatus(GCStringOut status, GSize status_len)
00150 {
00151   GReturn rc = G_NO_ERROR;
00152 #ifdef G_USE_GCAPS
00153   rc = GUtility(0, G_UTIL_GCAPS_SERVER_STATUS, status, &status_len);
00154 #else
00155   rc = G_GCAPS_OPEN_ERROR;
00156 #endif
00157
00158 #ifdef GCLIB_LOGGING
00159   if (rc != G_NO_ERROR)
00160   {
00161     char buf[256];
00162     sprintf(buf, "GServerStatus: %d", rc);
00163     LogMsg(buf);
00164   }
00165 #endif
00166   return rc;
00167 }
00168
00169 GReturn GCALL GListServers(GCStringOut servers, GSize servers_len)
00170 {
00171   GReturn rc;
00172 #ifdef G_USE_GCAPS
00173   rc = GUtility(0, G_UTIL_GCAPS_LIST_SERVERS, servers, &servers_len);
00174 #else
00175   rc = G_GCAPS_OPEN_ERROR;
00176 #endif
00177
00178 #ifdef GCLIB_LOGGING
00179   if (rc != G_NO_ERROR)
00180   {
00181     char buf[256];
00182     sprintf(buf, "GListServers: %d", rc);
00183     LogMsg(buf);
00184   }
00185 #endif
00186   return rc;
00187 }
00188
00189 GReturn GCALL GPublishServer(GCStringIn name, GOption publish, GOption save)
00190 {
00191   GReturn rc;
00192 #ifdef  G_USE_GCAPS
```

```
00193   //bit 0 -> publish = 1, remove = 0
00194   //bit 1 -> save = 1, do not save = 0
00195     unsigned short options = publish | (save << 1);
00196
00197   if (G_NO_ERROR == (rc = GUtility(0, G_UTIL_GCAPS_PUBLISH_SERVER, (void*)name, &options)))
00198   {
00199
00200   }
00201 #else
00202   rc = G_GCAPS_OPEN_ERROR;
00203 #endif
00204
00205 #ifdef GCLIB_LOGGING
00206   if (rc != G_NO_ERROR)
00207   {
00208     char buf[256];
00209     sprintf(buf, "GPublishServer: %d", rc);
00210     LogMsg(buf);
00211   }
00212 #endif
00213
00214   return rc;
00215 }
00216
00217 GReturn GCALL GRemoteConnections(GCStringOut connections, GSize connections_length)
00218 {
00219   GReturn rc;
00220 #ifdef G_USE_GCAPS
00221   rc = GUtility(0, G_UTIL_GCAPS_REMOTE_CONNECTIONS, connections, &connections_length);
00222 #else
00223   rc = G_GCAPS_OPEN_ERROR;
00224 #endif
00225
00226 #ifdef GCLIB_LOGGING
00227   if (rc != G_NO_ERROR)
00228   {
00229     char buf[256];
00230     sprintf(buf, "GRemoteConnections: %d", rc);
00231     LogMsg(buf);
00232   }
00233 #endif
00234   return rc;
00235 }
00236
00237 GReturn GCALL GCmd(GCon g, GCStringIn command)
00238 {
00239   char buf[G_SMALL_BUFFER]; //response usually brief, e.g. :
00240   return GCommand(g, command, buf, G_SMALL_BUFFER, 0);
00241 }
00242
00243 GReturn GCALL GCmdT(GCon g, GCStringIn command, GCStringOut trimmed_response, GSize response_len,
      GCStringOut* front)
00244 {
00245   GSize read;
00246   GReturn rc;
00247   int i;
00248   char c;
00249   if ((rc = GCommand(g, command, trimmed_response, response_len, &read)) != G_NO_ERROR)
00250     return rc;
00251   //if here, the data is already null-terminated, just trim.
00252   for (i = read - 1; i >= 0; i--) //read does NOT include null terminator.
00253   {
00254     c = trimmed_response[i];
00255     if ((c == ':') || (c == '\n') || (c == '\r'))
00256       trimmed_response[i] = 0; //trim it
00257     else
00258       break; //we hit non-trimmable data, bail out.
00259   }
00260
00261   if (front) //null to skip "trim" on front.
00262   {
00263     *front = trimmed_response;
00264     i = 0;
00265     do
00266     {
00267       c = trimmed_response[i++];
00268       if (c == ' ')
00269         (*front)++;
00270       else
00271         break;
00272     } while (1); //exit will be any non-space, including null terminator
00273   }
00274
00275   return G_NO_ERROR;
00276 }
00277
00278 GReturn GCALL GCmdI(GCon g, GCStringIn command, int* value)
```

```
00279 {
00280   char buf[G_SMALL_BUFFER]; //response should be ~19 chars
00281   GSize read;
00282   GReturn rc;
00283   if ((rc = GCommand(g, command, buf, G_SMALL_BUFFER, &read)) != G_NO_ERROR)
00284     return rc;
00285   *value = atoi(buf);
00286   return G_NO_ERROR;
00287 }
00288
00289 GReturn GCALL GCmdD(GCon g, GCStringIn command, double* value)
00290 {
00291   char buf[G_SMALL_BUFFER]; //response should be ~19 chars
00292   GSize read;
00293   GReturn rc;
00294   if ((rc = GCommand(g, command, buf, G_SMALL_BUFFER, &read)) != G_NO_ERROR)
00295     return rc;
00296   *value = atof(buf);
00297   return G_NO_ERROR;
00298 }
00299
00300 GReturn GCALL GMotionComplete(GCon g, GCStringIn axes)
00301 {
00302   char pred[] = "_BGm=0"; //predicate for polling the axis' motion status, m is a place holder
      replaced below.
00303   GReturn rc;
00304   GSize i = 0; //C, not C++
00305   GSize len = strlen(axes);
00306
00307   for (i = 0; i < len; i++) //iterate through all chars in axes
00308   {
00309     pred[3] = axes[i]; //set the axis
00310     rc = GWaitForBool(g, pred, -1); //poll forever. Change this if a premature exit is desired.
00311     if (rc != G_NO_ERROR)
00312       return rc;
00313   }//axes
00314
00315   return G_NO_ERROR;
00316 }
00317
00318 GReturn GCALL GWaitForBool(GCon g, GCStringIn predicate, int trials)
00319 {
00320   char cmd[G_LINE_BUFFER];
00321   char buf[G_LINE_BUFFER];
00322   int rc;
00323   strcpy(cmd, "MG (");
00324   strcat(cmd, predicate);
00325   strcat(cmd, ")"); //enclose in parenthesis
00326
00327   for (; trials != 0; --trials) //negative value will poll "forever"
00328   {
00329     rc = GCommand(g, cmd, buf, G_SMALL_BUFFER, 0); //check the predicate
00330     if (rc != G_NO_ERROR)
00331       return rc;
00332
00333     if (atoi(buf)) //nonzero is true, bad atoi returns 0
00334       return G_NO_ERROR;
00335     else
00336       GSleep(POLLINGINTERVAL);
00337   }
00338   //if we're here, the trials ran out
00339   return G_GCLIB_POLLING_FAILED;
00340 }
00341
00342 GReturn GCALL GRecordRate(GCon g, double period_ms)
00343 {
00344   char buf[G_SMALL_BUFFER];
00345   double dt;
00346   double period_arg;
00347
00348   if (period_ms == 0) //turn off
00349     return GCmd(g, "DR 0");
00350
00351   if (GCmdD(g, "TM?", &dt) == G_NO_ERROR)
00352   {
00353     dt /= 1024.0; //ms per controller sample
00354     if (!dt) dt = 1; //don't want to divide by zero below
00355   }
00356   else
00357   {
00358     dt = 0.9765625; //RIO doesn't have TM
00359   }
00360
00361   period_arg = period_ms / dt; //data record specified in samples between records
00362
00363   if (GCmdT(g, "\x12\x16", buf, sizeof(buf), 0) == G_NO_ERROR) //Revision string, ^R^V
00364   {
```

```
00365     if (strstr(buf, "DMC18")) //PCI controller
00366       period_arg = log(period_arg) / log(2.0); //PCI DR arg is 2^n.
00367     else if ((strstr(buf, "DMC40") != NULL) //4000
00368       || (strstr(buf, "DMC500") != NULL) //50000
00369       || (strstr(buf, "RIO") != NULL)) // RIO
00370     {
00371       if (period_arg < 2) period_arg = 2; //lowest non-zero DR
00372     }
00373     else if ((strstr(buf, "DMC41") != NULL) || (strstr(buf, "DMC21") != NULL)) //4103, 2103
00374     {
00375       if (period_arg < 8) period_arg = 8; //lowest non-zero DR
00376     }
00377     else if ((strstr(buf, "DMC3") != NULL)) //30010, 31010
00378     {
00379       if (period_arg < 4) period_arg = 4; //lowest non-zero DR
00380     }
00381   }
00382
00383   sprintf(buf, "DR %d", (int)period_arg);
00384   return GCmd(g, buf);
00385 }
00386
00387 GReturn GCALL GProgramDownloadFile(GCon g, GCStringIn file_path, GCStringIn preprocessor)
00388 {
00389   FILE *file;
00390   long file_size;
00391   char* program_buffer;
00392   GReturn rc = G_NO_ERROR;
00393
00394   if (!(file = fopen(file_path, "rb"))) //open file for reading, binary mode
00395     return G_BAD_FILE;
00396
00397   fseek(file, 0, SEEK_END); //find end of file
00398   file_size = ftell(file); //add one to null terminate below
00399   rewind(file);
00400
00401   if (file_size) //don't malloc 0.
00402   {
00403
00404     if (!(program_buffer = malloc(file_size + 1))) //allocate memory for the data, +1 for null
      termination below
00405     {
00406       fclose(file);
00407       return G_BAD_FULL_MEMORY;
00408     }
00409
00410     if (file_size != fread(program_buffer, 1, file_size, file))
00411     {
00412       fclose(file);
00413       free(program_buffer); //free memory
00414       return G_BAD_FILE;
00415     }
00416     program_buffer[file_size] = 0; //null terminate, malloc was one byte larger for this
00417   }
00418   else
00419   {
00420     program_buffer = ""; //nullstring
00421   }
00422
00423   fclose(file); //done with file, close it
00424
00425   rc = GProgramDownload(g, program_buffer, preprocessor); //call the gclib downloader
00426   if (file_size) free(program_buffer); //free memory
00427   return rc;
00428 }
00429
00430 GReturn GCALL GProgramUploadFile(GCon g, GCStringIn file_path)
00431 {
00432   FILE *file;
00433   GReturn rc = G_NO_ERROR;
00434   char* program_buffer;
00435   long file_size;
00436
00437   if (!(file = fopen(file_path, "wb"))) //open file for writing, binary mode
00438     return G_BAD_FILE;
00439
00440   if (!(program_buffer = malloc(MAXPROG))) //allocate memory for the data
00441   {
00442     fclose(file);
00443     return G_BAD_FULL_MEMORY;
00444   }
00445
00446   if ((rc = GProgramUpload(g, program_buffer, MAXPROG)) == G_NO_ERROR)
00447   {
00448     file_size = strlen(program_buffer);
00449     if (file_size != fwrite(program_buffer, 1, file_size, file))
00450       rc = G_BAD_FILE;
```

```
00451   }
00452
00453   fclose(file);
00454   free(program_buffer);
00455   return rc;
00456 }
00457
00458
00459 void GCALL GError(GReturn rc, GCStringOut error, GSize error_len)
00460 {
00461   char* error_message;
00462
00463   switch (rc)
00464   {
00465   case G_NO_ERROR:
00466     error_message = G_NO_ERROR_S;
00467     break;
00468
00469   case G_GCLIB_ERROR:
00470     error_message = G_GCLIB_ERROR_S;
00471     break;
00472
00473   case G_GCLIB_UTILITY_ERROR:
00474     error_message = G_GCLIB_UTILITY_ERROR_S;
00475     break;
00476
00477   case G_GCLIB_UTILITY_IP_TAKEN:
00478     error_message = G_GCLIB_UTILITY_IP_TAKEN_S;
00479     break;
00480
00481   case G_GCLIB_NON_BLOCKING_READ_EMPTY:
00482     error_message = G_GCLIB_NON_BLOCKING_READ_EMPTY_S;
00483     break;
00484
00485   case G_TIMEOUT:
00486     error_message = G_TIMEOUT_S;
00487     break;
00488
00489   case G_OPEN_ERROR:
00490     error_message = G_OPEN_ERROR_S;
00491     break;
00492
00493   case G_ALREADY_OPEN:
00494     error_message = G_ALREADY_OPEN_S;
00495     break;
00496
00497   case G_READ_ERROR:
00498     error_message = G_READ_ERROR_S;
00499     break;
00500
00501   case G_WRITE_ERROR:
00502     error_message = G_WRITE_ERROR_S;
00503     break;
00504
00505   case G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND:
00506     error_message = G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND_S;
00507     break;
00508
00509   case G_DATA_RECORD_ERROR:
00510     error_message = G_DATA_RECORD_ERROR_S;
00511     break;
00512
00513   case G_UNSUPPORTED_FUNCTION:
00514     error_message = G_UNSUPPORTED_FUNCTION_S;
00515     break;
00516
00517   case G_BAD_ADDRESS:
00518     error_message = G_BAD_ADDRESS_S;
00519     break;
00520
00521   case G_BAD_FIRMWARE_LOAD:
00522     error_message = G_BAD_FIRMWARE_LOAD_S;
00523     break;
00524
00525   case G_FIRMWARE_LOAD_NOT_SUPPORTED:
00526     error_message = G_FIRMWARE_LOAD_NOT_SUPPORTED_S;
00527     break;
00528
00529   case G_ARRAY_NOT_DIMENSIONED:
00530     error_message = G_ARRAY_NOT_DIMENSIONED_S;
00531     break;
00532
00533   case G_CONNECTION_NOT_ESTABLISHED:
00534     error_message = G_CONNECTION_NOT_ESTABLISHED_S;
00535     break;
00536
00537   case G_ILLEGAL_DATA_IN_PROGRAM:
```

```
00538     error_message = G_ILLEGAL_DATA_IN_PROGRAM_S;
00539     break;
00540
00541   case G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT:
00542     error_message = G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT_S;
00543     break;
00544
00545   case G_INVALID_PREPROCESSOR_OPTIONS:
00546     error_message = G_INVALID_PREPROCESSOR_OPTIONS_S;
00547     break;
00548
00549   case G_BAD_RESPONSE_QUESTION_MARK:
00550     error_message = G_BAD_RESPONSE_QUESTION_MARK_S;
00551     break;
00552
00553   case G_BAD_VALUE_RANGE:
00554     error_message = G_BAD_VALUE_RANGE_S;
00555     break;
00556
00557   case G_BAD_FULL_MEMORY:
00558     error_message = G_BAD_FULL_MEMORY_S;
00559     break;
00560
00561   case G_BAD_LOST_DATA:
00562     error_message = G_BAD_LOST_DATA_S;
00563     break;
00564
00565   case G_BAD_FILE:
00566     error_message = G_BAD_FILE_S;
00567     break;
00568
00569   case G_GCAPS_OPEN_ERROR:
00570     error_message = G_GCAPS_OPEN_ERROR_S;
00571     break;
00572
00573   case G_GCAPS_SUBSCRIPTION_ERROR:
00574     error_message = G_GCAPS_SUBSCRIPTION_ERROR_S;
00575     break;
00576
00577   default:
00578     error_message = "internal error";
00579     break;
00580   }
00581
00582   strncpy(error, error_message, error_len);
00583   error[error_len - 1] = 0; //ensure null termination
00584 }
00585
00586 #ifdef GCLIB_LOGGING
00587 void LogMsg(const char* msg)
00588 {
00589   time_t rawtime;
00590   struct tm * timeinfo;
00591
00592   time(&rawtime);
00593   timeinfo = localtime(&rawtime);
00594   char* time_string = asctime(timeinfo);
00595
00596 #ifdef _WIN32
00597   const char* filename = "C:/ProgramData/Galil/gcaps/gclib_mdns_log.txt";
00598 #else
00599   const char* filename = "/var/tmp/gclib_mdns_log.txt";
00600 #endif // _WIN32
00601
00602   FILE *logfile;
00603   logfile = fopen(filename, "a");
00604   fprintf(logfile, "%s | %s", msg, time_string);
00605   fclose(logfile);
00606 }
00607 #endif
00608
```

10.181 gclibo.h File Reference

```
#include "gclib.h"
```

Macros

- #define GCLIB_DLL_EXPORTED
- #define GCALL __stdcall

- #define MALLOCBUF G_HUGE_BUFFER

  Malloc used for large program and array uploads.
- #define MAXPROG MALLOCBUF

  Maximum size for a program.
- #define MAXARRAY MALLOCBUF

  Maximum size for an array table upload.
- #define POLLINGINTERVAL 100

  Interval, in milliseconds, for polling commands, e.g. GWaitForBool().
- #define G_USE_GCAPS

  Use the GCAPS server in GAddresses(), GAssign(), GIpRequests(), and GVersion(). To avoid GCAPS, comment out this line and recompile, `http://galil.com/sw/pub/all/doc/gclib/html/gclibo.html`.

Functions

- GCLIB_DLL_EXPORTED void GCALL GSleep (unsigned int timeout_ms)

  Uses GUtility() and G_UTIL_SLEEP to provide a blocking sleep call which can be useful for timing-based chores.
- GCLIB_DLL_EXPORTED GReturn GCALL GVersion (GCStringOut ver, GSize ver_len)

  Uses GUtility(), G_UTIL_VERSION and G_UTIL_GCAPS_VERSION to provide the library and gcaps version numbers.
- GCLIB_DLL_EXPORTED GReturn GCALL GAddresses (GCStringOut addresses, GSize addresses_len)

  Uses GUtility(), G_UTIL_GCAPS_ADDRESSES or G_UTIL_ADDRESSES to provide a listing of all available connection addresses.
- GCLIB_DLL_EXPORTED GReturn GCALL GInfo (GCon g, GCStringOut info, GSize info_len)

  Uses GUtility() and G_UTIL_INFO to provide a useful connection string.
- GCLIB_DLL_EXPORTED GReturn GCALL GTimeout (GCon g, short timeout_ms)

  Uses GUtility() and G_UTIL_TIMEOUT_OVERRIDE to set the library timeout.
- GCLIB_DLL_EXPORTED GReturn GCALL GCmd (GCon g, GCStringIn command)

  Wrapper around GCommand for use when the return value is not desired.
- GCLIB_DLL_EXPORTED GReturn GCALL GCmdT (GCon g, GCStringIn command, GCStringOut trimmed_response, GSize response_len, GCStringOut *front)

  Wrapper around GCommand that trims the response.
- GCLIB_DLL_EXPORTED GReturn GCALL GCmdI (GCon g, GCStringIn command, int *value)

  Wrapper around GCommand that provides the return value of a command parsed into an int.
- GCLIB_DLL_EXPORTED GReturn GCALL GCmdD (GCon g, GCStringIn command, double *value)

  Wrapper around GCommand that provides the return value of a command parsed into a double.
- GCLIB_DLL_EXPORTED GReturn GCALL GWaitForBool (GCon g, GCStringIn predicate, int trials)

  Blocking call that returns when the controller evaluates the predicate as true.
- GCLIB_DLL_EXPORTED GReturn GCALL GMotionComplete (GCon g, GCStringIn axes)

  Blocking call that returns once all axes specified have completed their motion.
- GCLIB_DLL_EXPORTED GReturn GCALL GRecordRate (GCon g, double period_ms)

  Sets the asynchronous data record to a user-specified period via `DR`.
- GCLIB_DLL_EXPORTED GReturn GCALL GProgramDownloadFile (GCon g, GCStringIn file_path, GCStringIn preprocessor)

  Program download from file.
- GCLIB_DLL_EXPORTED GReturn GCALL GProgramUploadFile (GCon g, GCStringIn file_path)

  Program upload to file.
- GCLIB_DLL_EXPORTED GReturn GCALL GArrayDownloadFile (GCon g, GCStringIn file_path)

  Array download from file.
- GCLIB_DLL_EXPORTED GReturn GCALL GArrayUploadFile (GCon g, GCStringIn file_path, GCStringIn names)

  Array upload to file.
- GCLIB_DLL_EXPORTED GReturn GCALL GIpRequests (GCStringOut requests, GSize requests_len)

Uses GUtility(), G_UTIL_GCAPS_IPREQUEST or G_UTIL_IPREQUEST to provide a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.

- GCLIB_DLL_EXPORTED GReturn GCALL GSetServer (GCStringIn server_name)

  Uses GUtility(), G_UTIL_GCAPS_SET_SERVER to set the new active server.
- GCLIB_DLL_EXPORTED GReturn GCALL GListServers (GCStringOut servers, GSize servers_len)

  Uses GUtility(), G_UTIL_GCAPS_LIST_SERVERS to provide a list of all available gcaps services on the local network.
- GCLIB_DLL_EXPORTED GReturn GCALL GPublishServer (GCStringIn name, GOption publish, GOption save)

  Uses GUtility(), G_UTIL_GCAPS_PUBLISH_SERVER to publish local gcaps server to the local network.
- GCLIB_DLL_EXPORTED GReturn GCALL GServerStatus (GCStringOut status, GSize status_len)

  Uses GUtility(), G_UTIL_GCAPS_SERVER_STATUS to get information on the local server name and if it is published to the local network.
- GCLIB_DLL_EXPORTED GReturn GCALL GRemoteConnections (GCStringOut connections, GSize connections_length)

  Uses GUtility(), G_UTIL_GCAPS_REMOTE_CONNECTIONS to get a list of remote addresses connected to the local server.
- GCLIB_DLL_EXPORTED GReturn GCALL GAssign (GCStringIn ip, GCStringIn mac)

  Uses GUtility(), G_UTIL_GCAPS_ASSIGN or G_UTIL_ASSIGN to assign an IP address over the Ethernet to a controller at a given MAC address.
- GCLIB_DLL_EXPORTED void GCALL GError (GReturn rc, GCStringOut error, GSize error_len)

  Provides a human-readable description string for return codes.
- GCLIB_DLL_EXPORTED GReturn GCALL GSetupDownloadFile (GCon g, GCStringIn file_path, GOption options, GCStringOut info, GSize info_len)

  Download a saved controller configuration from a file.

10.181.1 Detailed Description

Open-source convenience functions for Galil C Lib. Please email softwarefeedback@galil.com with suggestions for useful/missing functions.
Definition in file gclibo.h.

10.181.2 Macro Definition Documentation

10.181.2.1 G_USE_GCAPS

```
#define G_USE_GCAPS
```

Use the GCAPS server in GAddresses(), GAssign(), GIpRequests(), and GVersion(). To avoid GCAPS, comment out this line and recompile, http://galil.com/sw/pub/all/doc/gclib/html/gclibo.html.
Definition at line 41 of file gclibo.h.

10.181.2.2 GCALL

```
#define GCALL __stdcall
```

Definition at line 28 of file gclibo.h.

10.181.2.3 GCLIB_DLL_EXPORTED

```
#define GCLIB_DLL_EXPORTED
```

Definition at line 22 of file gclibo.h.

10.181.2.4 MALLOCBUF

```
#define MALLOCBUF G_HUGE_BUFFER
```

Malloc used for large program and array uploads.
Definition at line 37 of file gclibo.h.

10.181.2.5 MAXARRAY

```
#define MAXARRAY MALLOCBUF
```

Maximum size for an array table upload.
Definition at line 39 of file gclibo.h.

10.181.2.6 MAXPROG

```
#define MAXPROG MALLOCBUF
```

Maximum size for a program.
Definition at line 38 of file gclibo.h.

10.181.2.7 POLLINGINTERVAL

```
#define POLLINGINTERVAL 100
```

Interval, in milliseconds, for polling commands, e.g. GWaitForBool().
Definition at line 40 of file gclibo.h.

10.181.3 Function Documentation

10.181.3.1 GAddresses()

```
GReturn GCALL GAddresses (
            GCStringOut addresses,
            GSize addresses_len )
```

Uses GUtility(), G_UTIL_GCAPS_ADDRESSES or G_UTIL_ADDRESSES to provide a listing of all available connection addresses.

Note

Serial ports are listed, e.g. COM1. Upon open, it may be necessary to specify a baud rate for the controller, e.g. `--baud 19200`. Default baud is 115200. See GOpen().

Parameters

addresses	Buffer to hold the output string. Buffer will be null terminated, even if the data must be truncated to do so. See below for more information.
addresses_len	Length of buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

If gcaps is available, the listing will come from the server via G_UTIL_GCAPS_ADDRESSES. In the absence of the server, gclib will use G_UTIL_ADDRESSES to generate the list.

- Ethernet controllers will be listed as *ip_address, revision_report, network_adapter_name, network_adapter_ip_address*. If an IP address is unreachable via ping, the address will be in parentheses.
- PCI controllers will be listed by their identifier, e.g. GALILPCI1.
- Serial ports will be listed by their identifier, e.g. COM1.

```
10.1.3.91, DMC4020 Rev 1.2e, LAN, 10.1.3.10
192.168.0.63, DMC4040 Rev 1.2f, Static, 192.168.0.41
(192.0.0.42), RIO47102 Rev 1.1j, Static, 192.168.0.41
GALILPCI1
COM1
COM2
```

Note

GAddresses() will take up to 1 second to look for gcaps.

See x_examples.cpp for an example.
Definition at line 54 of file gclibo.c.
References G_NO_ERROR, G_UTIL_ADDRESSES, G_UTIL_GCAPS_ADDRESSES, GUtility(), and vector().
Referenced by remote_client().

10.181.3.2 GArrayDownloadFile()

```
GCLIB_DLL_EXPORTED GReturn GCALL GArrayDownloadFile (
            GCon g,
            GCStringIn file_path )
```

Array download from file.
Downloads a csv file containing array data at `file_path`. If the arrays don't exist, they will be dimensioned.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the array file.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_arrays.cpp for an example.
Definition at line 380 of file arrays.c.
References G_BAD_FILE, G_NO_ERROR, H_ArrayDownloadFromMemory(), and vector().
Referenced by x_arrays().

10.181.3.3 GArrayUploadFile()

```
GCLIB_DLL_EXPORTED GReturn GCALL GArrayUploadFile (
            GCon g,
            GCStringIn file_path,
            GCStringIn names )
```

Array upload to file.
Uploads the entire controller array table or a subset and saves the data as a csv file specified by `file_path`.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the array file, file will be overwritten if it exists.
names	Null-terminated string containing the arrays to upload, delimited with space. "" or null uploads all arrays listed in LA.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_arrays.cpp for an example.
Definition at line 408 of file arrays.c.
References G_NO_ERROR, GCmdT(), H_FreeArrays(), H_InitArrayNode(), H_UploadArrayToList(), H_WriteArrayCsv(), and vector().
Referenced by x_arrays().

10.181.3.4 GAssign()

```
GReturn GCALL GAssign (
```

```
            GCStringIn ip,
            GCStringIn mac )
```

Uses GUtility(), G_UTIL_GCAPS_ASSIGN or G_UTIL_ASSIGN to assign an IP address over the Ethernet to a controller at a given MAC address.

Parameters

ip	The null-terminated ip address to assign. The hardware should not yet have an IP address.
mac	The null-terminated MAC address of the hardware.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

On Linux and Mac, the desired IP address will be pinged prior to the assignment. If the ping is returned, GAssign() will return G_GCLIB_UTILITY_IP_TAKEN.
If gcaps is available, the assign will be performed from the server via G_UTIL_GCAPS_ASSIGN. gcaps will remember the assignment and will automatically assign the desired IP address should the controller ever request one again, e.g. after a controller master reset. To clear the remembered IP address from gcaps, call GAssign() with a blank string in place of the ip address. To remove all remembered ip addresses, specfify a blank string for the mac address.
In the absence of the server, gclib will use G_UTIL_ASSIGN to assign. GAssign() will take up to 1 second to look for gcaps. When not using gcaps, Linux/OS X users must be root to use GAssign() and have UDP access to send on port 68.
See x_examples.cpp for an example.
Definition at line 70 of file gclibo.c.
References G_GCLIB_UTILITY_IP_TAKEN, G_NO_ERROR, G_UTIL_ASSIGN, G_UTIL_GCAPS_ASSIGN, G_UTIL_GCAPS_PING, G_UTIL_PING, GUtility(), and vector().
Referenced by ip_assigner().

10.181.3.5 GCmd()

```
GReturn GCALL GCmd (
            GCon g,
            GCStringIn command )
```

Wrapper around GCommand for use when the return value is not desired.
The returned data is still checked for error, e.g. ? or timeout, but is not brought out through the prototype.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 237 of file gclibo.c.
References G_SMALL_BUFFER, GCommand(), and vector().
Referenced by check_interrupts(), commands(), contour(), GRecordRate(), H_DownloadArraysFromList(), H_DownloadData(), ip_assigner(), jog(), load_buf(), load_buffer(), message(), motion_complete(), position_tracking(), record_position(), vector(), x_arrays(), x_dr_motioncomplete(), x_ei_motioncomplete(), x_gcommand(), x_ginterrupt(), x_gmessage(), x_gmotioncomplete(), x_gread_gwrite(), x_grecord(), x_nonblocking(), and x_programs().

10.181.3.6 GCmdD()

```
GReturn GCALL GCmdD (
```

```
            GCon g,
            GCStringIn command,
            double * value )
```

Wrapper around GCommand that provides the return value of a command parsed into a double.
Use this function to retrieve the full Galil 4.2 range, e.g. for a variable value with fractional data, or the value of an Analog input or Output.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.
value	Pointer to a double that will be filled with the return value.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 289 of file gclibo.c.
References G_NO_ERROR, G_SMALL_BUFFER, GCommand(), and vector().
Referenced by commands(), GRecordRate(), and x_gcommand().

10.181.3.7 GCmdI()

```
GReturn GCALL GCmdI (
            GCon g,
            GCStringIn command,
            int * value )
```

Wrapper around GCommand that provides the return value of a command parsed into an int.
Use this function to get most values including TP, RP, TE, Digital I/O states, etc.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.
value	Pointer to an int that will be filled with the return value.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 278 of file gclibo.c.
References G_NO_ERROR, G_SMALL_BUFFER, GCommand(), and vector().
Referenced by commands(), contour(), record_position(), vector(), x_dr_motioncomplete(), x_gcommand(), x_gread_gwrite(), x_grecord(), and x_programs().

10.181.3.8 GCmdT()

```
GReturn GCALL GCmdT (
            GCon g,
            GCStringIn command,
            GCStringOut trimmed_response,
            GSize response_len,
            GCStringOut * front )
```

Wrapper around GCommand that trims the response.
For use when the return value is desired, is ASCII (not binary), and the response should be trimmed of trailing colon, whitespace, and optionally leading space.

Parameters

g	Connection's handle.
command	Null-terminated command string to send to the controller.
trimmed_response	The trimmed response from the controller. Trailing space is trimmed by null terminating any trailing spaces, carriage returns, or line feeds.
response_len	The length of the trimmed_response buffer.
front	If non-null, upon return ∗front will point to the first non-space character in trimmed_response. This allows trimming the front of the string without modifying the user's buffer pointer, which may be allocated on the heap.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp for an example.
Definition at line 243 of file gclibo.c.
References G_NO_ERROR, GCommand(), and vector().
Referenced by commands(), GArrayUploadFile(), GRecordRate(), motion_complete(), x_gcommand(), x_ginterrupt(), x_gmotioncomplete(), and x_grecord().

10.181.3.9 GError()

```
void GCALL GError (
            GReturn rc,
            GCStringOut error,
            GSize error_len )
```

Provides a human-readable description string for return codes.

Parameters

rc	The return code to lookup.
error	The buffer to fill with the error text. Buffer will be null terminated, even if the data must be truncated to do so.
error_len	The length of the error buffer.

See x_examples.cpp for an example.
Definition at line 459 of file gclibo.c.
References error(), G_ALREADY_OPEN, G_ARRAY_NOT_DIMENSIONED, G_BAD_ADDRESS, G_BAD_FILE, G_BAD_FIRMWARE_LOAD, G_BAD_FULL_MEMORY, G_BAD_LOST_DATA, G_BAD_RESPONSE_QUESTION_MARK, G_BAD_VALUE_RANGE, G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND, G_CONNECTION_NOT_ESTABLISHED, G_DATA_RECORD_ERROR, G_FIRMWARE_LOAD_NOT_SUPPORTED, G_GCAPS_OPEN_ERROR, G_GCAPS_SUBSCRIPTION, G_GCLIB_ERROR, G_GCLIB_NON_BLOCKING_READ_EMPTY, G_GCLIB_UTILITY_ERROR, G_GCLIB_UTILITY_IP_TAKEN, G_ILLEGAL_DATA_IN_PROGRAM, G_INVALID_PREPROCESSOR_OPTIONS, G_NO_ERROR, G_OPEN_ERROR, G_READ_ERROR, G_TIMEOUT, G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT, G_UNSUPPORTED_FUNCTION, G_WRITE_ERROR, and vector().
Referenced by error(), gclib::GArrayDownload(), gclib::GArrayDownloadFile(), gclib::GArrayUpload(), gclib::GArrayUploadFile(), gclib::GAssign(), gclib::GCommand(), gclib::GFirmwareDownload(), gclib::GListServers(), gclib::GMotionComplete(), gclib::GOpen(), gclib::GProgramDownload(), gclib::GProgramDownloadFile(), gclib::GProgramUpload(), gclib::GProgramUploadFile(), gclib::GPublishServer(), gclib::GRecord< T >(), gclib::GRecordRate(), gclib::GRemoteConnections(), gclib::GServerStatus(), gclib::GSetServer(), gclib::GSetupDownloadFile(), and gclib::GWrite().

10.181.3.10 GInfo()

```
GReturn GCALL GInfo (
            GCon g,
            GCStringOut info,
```

```
            GSize info_len )
```

Uses GUtility() and G_UTIL_INFO to provide a useful connection string.

Parameters

g	Connection's handle.
info	Buffer to hold the output string. Buffer will be null terminated, even if the data must be truncated to do so.
info_len	Length of buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

The response is *address, revision_report, serial_number*. For example:

```
COM2, RIO47102 Rev 1.1j, 37290
```

See x_examples.cpp for an example.
Definition at line 49 of file gclibo.c.
References G_UTIL_INFO, GUtility(), and vector().
Referenced by ip_assigner().

10.181.3.11 GIpRequests()

```
GReturn GCALL GIpRequests (
            GCStringOut requests,
            GSize requests_len )
```

Uses GUtility(), G_UTIL_GCAPS_IPREQUEST or G_UTIL_IPREQUEST to provide a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.

Parameters

requests	The buffer to hold the list of requesting controllers. Data will be null terminated, even if the data must be truncated to do so. See below for more information.
requests_len	The length of the requests buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

GIpRequests() will block up to 5 seconds while listening for requests.
If gcaps is available, the listing will come from the server via G_UTIL_GCAPS_IPREQUEST. In the absence of the server, gclib will use G_UTIL_IPREQUEST to generate the list. GIpRequests() will take up to 1 second to look for gcaps. When not using gcaps, Linux/OS X users must be root to use GIpRequests() and have UDP access to bind and listen on port 67.
Each line of the returned data will be of the form *model, serial_number, MAC_address, network_adapter_name, network_adapter_ip_address, remembered_ip_assignment*. See GAssign() for more infomation about remembered IP assignments. The following is an example output.

```
DMC2000, 34023, 00:50:4C:00:84:E7, enp5s0, 192.168.42.92, 192.168.42.200
DMC2105, 7, 00:50:4C:58:00:07, enp5s0, 192.168.42.92, 0.0.0.0
DMC2105, 13, 00:50:4C:58:00:0D, enp5s0, 192.168.42.92, 0.0.0.0
```

See x_examples.cpp for an example.
Definition at line 106 of file gclibo.c.
References G_NO_ERROR, G_UTIL_GCAPS_IPREQUEST, G_UTIL_IPREQUEST, GSleep(), GUtility(), and vector().
Referenced by ip_assigner().

10.181.3.12 GListServers()

```
GReturn GCALL GListServers (
            GCStringOut servers,
            GSize servers_len )
```

Uses GUtility(), G_UTIL_GCAPS_LIST_SERVERS to provide a list of all available gcaps services on the local network.

Note

This function is only available on Windows 10 and Linux.

Parameters

servers	The buffer to hold the list of available gcaps servers
servers_len	The length of the servers buffer

This function is used to find a list of available gcaps servers that have made themselves "Discoverable".
The list of available servers are separated by a newline '\n' character.

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 169 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_LIST_SERVERS, GUtility(), and vector().
Referenced by remote_client().

10.181.3.13 GMotionComplete()

```
GReturn GCALL GMotionComplete (
            GCon g,
            GCStringIn axes )
```

Blocking call that returns once all axes specified have completed their motion.

Note

This function uses a profiled motion indicator, not the position of the encoder. E.G. see the difference between AM (profiled) and MC (encoder-based).

Although using the _BGm operand is the most generally compatible method, there are higher-performance ways to check for motion complete by using the data record, or interrupts. See examples x_dr_motioncomplete() and x_ei_motioncomplete().

Parameters

g	Connection's handle.
axes	A null-terminated string containing a multiple-axes mask. Every character in the string should be a valid argument to MG_BGm, i.e. XYZWABCDEFGHST.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gmotioncomplete.cpp for an example.

Definition at line 300 of file gclibo.c.
References G_NO_ERROR, GWaitForBool(), and vector().
Referenced by contour(), jog(), position_tracking(), vector(), and x_gmotioncomplete().

10.181.3.14 GProgramDownloadFile()

```
GReturn GCALL GProgramDownloadFile (
            GCon g,
            GCStringIn file_path,
            GCStringIn preprocessor )
```

Program download from file.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the program file.
preprocessor	Options string for preprocessing the program before sending it to the controller. See GProgramDownload().

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_programs.cpp for an example.
Definition at line 387 of file gclibo.c.
References G_BAD_FILE, G_BAD_FULL_MEMORY, G_NO_ERROR, GProgramDownload(), and vector().
Referenced by x_programs().

10.181.3.15 GProgramUploadFile()

```
GReturn GCALL GProgramUploadFile (
            GCon g,
            GCStringIn file_path )
```

Program upload to file.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the program file, file will be overwritten if it exists.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_programs.cpp for an example.
Definition at line 430 of file gclibo.c.
References G_BAD_FILE, G_BAD_FULL_MEMORY, G_NO_ERROR, GProgramUpload(), MAXPROG, and vector().
Referenced by x_programs().

10.181.3.16 GPublishServer()

```
GReturn GCALL GPublishServer (
            GCStringIn name,
            GOption publish,
            GOption save )
```

Uses GUtility(), G_UTIL_GCAPS_PUBLISH_SERVER to publish local gcaps server to the local network.

Note

This function is only available on Windows 10 and Linux.

Parameters

name	The name of the server to publish or remove
publish	Option to publish or remove server from network
save	Option to save this configuration for future reboots

This function is used to make your local gcaps server "Discoverable" or "Invisible"
publish Option:
Set to 1 to publish server to the network and make "Discoverable"
Set to 0 to remove server from the network and make "Invisible"

save Option:
Set to 1 to save the configuration for future reboots of the server
Set to 0 to use this configuration once, and not overwrite previous server settings

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 189 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_PUBLISH_SERVER, GUtility(), and vector().
Referenced by remote_server().

10.181.3.17 GRecordRate()

```
GReturn GCALL GRecordRate (
            GCon g,
            double period_ms )
```

Sets the asynchronous data record to a user-specified period via DR.
Takes TM and product type into account and sets the DR period to the period requested by the user, if possible.

Parameters

g	Connection's handle.
period_ms	Period, in milliseconds, to set up for the asynchronous data record.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_grecord.cpp for an example.
Definition at line 342 of file gclibo.c.
References G_NO_ERROR, G_SMALL_BUFFER, GCmd(), GCmdD(), GCmdT(), and vector().
Referenced by x_dr_motioncomplete(), x_grecord(), and x_nonblocking().

10.181.3.18 GRemoteConnections()

```
GReturn GCALL GRemoteConnections (
            GCStringOut connections,
            GSize connections_length )
```

Uses GUtility(), G_UTIL_GCAPS_REMOTE_CONNECTIONS to get a list of remote addresses connected to the local server.

Note

This function is only available on Windows 10 and Linux.

Parameters

connections	The buffer to hold the list of remote IP addresses currently connected to your hardware
connections_len	The length of the connections buffer

This function is used to find a list of IP Addresses of machines that currently have open connections to your local hardware. If another user sets your local server as their active server, and then opens a connection to your hardware, their IP Address will appear in this list.
The list of IP addresses are separated by a newline '\n' character.

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 217 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_REMOTE_CONNECTIONS, GUtility(), and vector().

10.181.3.19 GServerStatus()

```
GReturn GCALL GServerStatus (
            GCStringOut status,
            GSize status_len )
```

Uses GUtility(), G_UTIL_GCAPS_SERVER_STATUS to get information on the local server name and if it is published to the local network.

Note

This function is only available on Windows 10 and Linux.

Parameters

status	The buffer to hold the status of the local gcaps server
status_len	The length of the status buffer

This function is used to find the status of your local gcaps server. Use this function to determine the name your server is currently using, and whether or not your gcaps server is currently set to "Discoverable" or "Invisible"
The status buffer will be filled in the form of "[Server Name], [Discoverable]"
For example, for a server with the name "Example Server" that is set to "Discoverable", the status buffer would contain "Example Server, true".

Attention

This function will always use your local gcaps server, regardless of which server you have set as your active server.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 149 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_SERVER_STATUS, GUtility(), and vector().

10.181.3.20 GSetServer()

```
GReturn GCALL GSetServer (
            GCStringIn server_name )
```

Uses GUtility(), G_UTIL_GCAPS_SET_SERVER to set the new active server.

Note

This function is only available on Windows 10 and Linux.

Parameters

server_name	The name of the server to set as your new active server.

Use this function in conjunction with GListServers(). Choose a name received from GListServers() to set as your new active server.
After setting a new active server, all gclib calls will route through that new active server, unless explicitly noted otherwise.
To set your active server back to your local server, simply pass "Local" to GSetServer():

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

Definition at line 128 of file gclibo.c.
References G_GCAPS_OPEN_ERROR, G_NO_ERROR, G_UTIL_GCAPS_SET_SERVER, GUtility(), and vector().
Referenced by remote_client().

10.181.3.21 GSetupDownloadFile()

```
GCLIB_DLL_EXPORTED GReturn GCALL GSetupDownloadFile (
            GCon g,
            GCStringIn file_path,
            GOption options,
            GCStringOut info,
            GSize info_len )
```

Download a saved controller configuration from a file.

Parameters

g	Connection's handle.
file_path	Null-terminated string containing the path to the gcb file.
options	Bit mask to determine what configuration data to download. See below for all options.
info	Optional pointer to a buffer to store the controller info. If no info is needed, specify as NULL.
info_len	Length of optional info buffer. If no info is needed, specify as NULL.

Returns

The success status or error code of the function. If the options parameter is set to 0, the return value will be a bit mask indicating which sectors in the specified GCB are not empty. Otherwise, see gclib_errors.h for possible error values.

Note

By default, GSetupDownloadFile() will stop immediately if an error is encountered downloading data. This can be overridden in the options parameter. For example, you may want to override the error if you have a backup from an 8-axis controller and want to restore the parameters for the first 4 axes to a 4-axis controller.

If both info and info_len are not NULL, the controller information will be provided regardless of the options parameter. The options parameter is a bit mask. If options is set to 0, GSetupDownloadFile() will return a bit mask indicating which sectors in the specified GCB are not empty. The following contains a list of all currently available options:

Bit	Value	Function	Description
1	0x0002	Restore parameters	**KPA**, **KIA**, **KDA**, etc...
3	0x0008	Restore variables	Variables are listed by the **LV** command
4	0x0010	Restore arrays	Arrays are listed by the **LA** command
5	0x0020	Restore program	The program is listed by the **LS** command
31	0x8000	Ignore errors	Ignore invalid parameter errors and continue restoring data. GSetupDownloadFile() will still stop immediately if a connection issue or other fatal error is encountered

Usage example:

```
GCon g;
GOption opt = 0;

GCStringOut info;
GSize info_len = 4096;

GReturn rc = GOpen("192.168.0.50", &g);
if (rc) return rc;

// Call GSetupDownloadFile() with options set to 0 so we can get the non-empty sector bit mask
opt = GSetupDownloadFile(g, "C:\\path\\to\\gcb\\file.gcb", 0, NULL, NULL);

info = (GCStringOut)malloc(sizeof(GCStringOut) * info_len);

// Call GSetupDownloadFile() with the bit mask returned in the previous function call
rc = GSetupDownloadFile(g, "C:\\path\\to\\gcb\\file.gcb", opt, info, info_len);

printf("Info:\n\n%s", info);

GClose(g);

free(info);
return rc;
```

Definition at line 476 of file arrays.c.
References G_BAD_FILE, G_NO_ERROR, GProgramDownload(), H_ArrayDownloadFromMemory(), H_DownloadData(), H_FindSector(), and vector().

10.181.3.22 GSleep()

```
void GCALL GSleep (
            unsigned int timeout_ms )
```

Uses GUtility() and G_UTIL_SLEEP to provide a blocking sleep call which can be useful for timing-based chores.

Parameters

timeout_ms	The timeout, in milliseconds, to block before returning.

See GWaitForBool() for an example.
Definition at line 24 of file gclibo.c.
References G_UTIL_SLEEP, and GUtility().

Referenced by contour(), GIpRequests(), GWaitForBool(), record_position(), vector(), x_gread_gwrite(), and x_programs().

10.181.3.23 GTimeout()

```
GReturn GCALL GTimeout (
            GCon g,
            short timeout_ms )
```

Uses GUtility() and G_UTIL_TIMEOUT_OVERRIDE to set the library timeout.

Parameters

g	Connection's handle.
timeout_ms	The value to be used for the timeout. Use `G_USE_INITIAL_TIMEOUT` to set the timeout back to the initial GOpen() value, `--timeout`.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See x_gcommand.cpp and x_gread_gwrite.cpp for examples.
Definition at line 65 of file gclibo.c.
References G_UTIL_TIMEOUT_OVERRIDE, and GUtility().
Referenced by motion_complete(), x_gcommand(), x_ginterrupt(), x_gmessage(), x_gread_gwrite(), and x_nonblocking().

10.181.3.24 GVersion()

```
GReturn GCALL GVersion (
            GCStringOut ver,
            GSize ver_len )
```

Uses GUtility(), G_UTIL_VERSION and G_UTIL_GCAPS_VERSION to provide the library and gcaps version numbers.

Parameters

ver	Buffer to hold the output string. Buffer will be null terminated, even if the data must be truncated to do so.
ver_len	Length of buffer.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

The version number of gclib is provided first. If the gcaps server can be found, its version will be provided after a space.
Example with gcaps version.

```
154.190.329 1.0.0.82
```

Example with gclib version only.

```
154.190.329
```

Note

GVersion() will take up to 1 second to look for gcaps.

See x_examples.cpp for an example.
Definition at line 29 of file gclibo.c.
References G_NO_ERROR, G_UTIL_GCAPS_VERSION, G_UTIL_VERSION, GUtility(), and vector().

10.181.3.25 GWaitForBool()

```
GReturn GCALL GWaitForBool (
            GCon g,
            GCStringIn predicate,
            int trials )
```

Blocking call that returns when the controller evaluates the predicate as true.
Polls the message command (MG) to check the value of predicate. Polling will continue until the controller responds with a nonzero value or the number of polling trials is reached.
The amount of time until the function fails with G_GCLIB_POLLING_FAILED is roughly (trials ∗ POLLINGINTERVAL) milliseconds.

Parameters

g	Connection's handle.
predicate	A null-terminated string containing the predicate to be polled. The predicate will be enclosed in parentheses and used in the command `MG (predicate)` to return the value.
trials	The number of polling cycles to perform looking for a nonzero value. Use -1 to poll indefinitely.

Returns

The success status or error code of the function. See gclib_errors.h for possible values.

See GMotionComplete() for an example.
Definition at line 318 of file gclibo.c.
References G_GCLIB_POLLING_FAILED, G_LINE_BUFFER, G_NO_ERROR, G_SMALL_BUFFER, GCommand(), GSleep(), POLLINGINTERVAL, and vector().
Referenced by GMotionComplete().

10.182 gclibo.h

Go to the documentation of this file.

```
00001
00011 #ifndef I_007AD0AF_C956_4B96_ADE2_AD04FAFFEE99
00012 #define I_007AD0AF_C956_4B96_ADE2_AD04FAFFEE99
00013
00014 //set library visibility for gcc and msvc
00015 #if BUILDING_GCLIB && HAVE_VISIBILITY
00016 #define GCLIB_DLL_EXPORTED __attribute__((__visibility__("default")))
00017 #elif BUILDING_GCLIB && defined _MSC_VER
00018 #define GCLIB_DLL_EXPORTED __declspec(dllexport)
00019 #elif defined _MSC_VER
00020 #define GCLIB_DLL_EXPORTED __declspec(dllimport)
00021 #else
00022 #define GCLIB_DLL_EXPORTED
00023 #endif
00024
00025 #include "gclib.h" //Galil's C Library
00026
00027 #ifdef _WIN32
00028 #define GCALL __stdcall
00029 #else
00030 #define GCALL  /* nothing */
00031 #endif
00032
00033 #ifdef __cplusplus
00034 extern "C" {
00035 #endif
00036
00037 #define MALLOCBUF G_HUGE_BUFFER
00038 #define MAXPROG MALLOCBUF
00039 #define MAXARRAY MALLOCBUF
00040 #define POLLINGINTERVAL 100
00041 #define G_USE_GCAPS
00042
00044   GCLIB_DLL_EXPORTED void GCALL GSleep(unsigned int timeout_ms);
00053   GCLIB_DLL_EXPORTED GReturn GCALL GVersion(GCStringOut ver, GSize ver_len);
00077   GCLIB_DLL_EXPORTED GReturn GCALL GAddresses(GCStringOut addresses, GSize addresses_len);
00109   GCLIB_DLL_EXPORTED GReturn GCALL GInfo(GCon g, GCStringOut info, GSize info_len);
00126   GCLIB_DLL_EXPORTED GReturn GCALL GTimeout(GCon g, short timeout_ms);
00138   GCLIB_DLL_EXPORTED GReturn GCALL GCmd(GCon g, GCStringIn command);
```

```
00152   GCLIB_DLL_EXPORTED GReturn GCALL GCmdT(GCon g, GCStringIn command, GCStringOut trimmed_response,
      GSize response_len, GCStringOut* front);
00170   GCLIB_DLL_EXPORTED GReturn GCALL GCmdI(GCon g, GCStringIn command, int* value);
00185   GCLIB_DLL_EXPORTED GReturn GCALL GCmdD(GCon g, GCStringIn command, double* value);
00200   GCLIB_DLL_EXPORTED GReturn GCALL GWaitForBool(GCon g, GCStringIn predicate, int trials);
00222   GCLIB_DLL_EXPORTED GReturn GCALL GMotionComplete(GCon g, GCStringIn axes);
00239   GCLIB_DLL_EXPORTED GReturn GCALL GRecordRate(GCon g, double period_ms);
00253   GCLIB_DLL_EXPORTED GReturn GCALL GProgramDownloadFile(GCon g, GCStringIn file_path, GCStringIn
      preprocessor);
00266   GCLIB_DLL_EXPORTED GReturn GCALL GProgramUploadFile(GCon g, GCStringIn file_path);
00278   GCLIB_DLL_EXPORTED GReturn GCALL GArrayDownloadFile(GCon g, GCStringIn file_path);
00292   GCLIB_DLL_EXPORTED GReturn GCALL GArrayUploadFile(GCon g, GCStringIn file_path, GCStringIn names);
00307   GCLIB_DLL_EXPORTED GReturn GCALL GIpRequests(GCStringOut requests, GSize requests_len);
00335   GCLIB_DLL_EXPORTED GReturn GCALL GSetServer(GCStringIn server_name);
00353   GCLIB_DLL_EXPORTED GReturn GCALL GListServers(GCStringOut servers, GSize servers_len);
00372   GCLIB_DLL_EXPORTED GReturn GCALL GPublishServer(GCStringIn name, GOption publish, GOption save);
00398   GCLIB_DLL_EXPORTED GReturn GCALL GServerStatus(GCStringOut status, GSize status_len);
00421   GCLIB_DLL_EXPORTED GReturn GCALL GRemoteConnections(GCStringOut connections, GSize
      connections_length);
00443   GCLIB_DLL_EXPORTED GReturn GCALL GAssign(GCStringIn ip, GCStringIn mac);
00468   GCLIB_DLL_EXPORTED void GCALL GError(GReturn rc, GCStringOut error, GSize error_len);
00477 #ifndef G_OMIT_GSETUPDDOWNLOADFILE
00479   GCLIB_DLL_EXPORTED GReturn GCALL GSetupDownloadFile(GCon g, GCStringIn file_path, GOption options,
      GCStringOut info, GSize info_len);
00537 #endif //G_OMIT_GSETUPDDOWNLOADFILE
00538
00539 #ifdef GCLIB_LOGGING
00540   void LogMsg(const char* msg);
00541 #endif
00542
00543 #ifdef __cplusplus
00544 } //extern "C"
00545 #endif
00546
00547 #endif //I_007AD0AF_C956_4B96_ADE2_AD04FAFFEE99
```

10.183 gclib.cs File Reference

Data Structures

- class gclib

  Provides a class that binds to gclib's unmanaged dll. Wraps each call and provides a more user-friendly interface for use in C#.
- interface GDataRecord
- struct GDataRecord4000

  Data record struct for DMC-4000 controllers, including 4000, 4200, 4103, and 500x0.
- struct GDataRecord52000

  Data record struct for DMC-52000 controller. Same as DMC-4000, with bank indicator added at byte 40.
- struct GDataRecord1806

  Data record struct for DMC-1806 controller.
- struct GDataRecord2103

  Data record struct for DMC-2103 controllers.
- struct GDataRecord1802

  Data record struct for DMC-1802 controllers.
- struct GDataRecord30000

  Data record struct for DMC-30010 controllers.
- struct GDataRecord47000_ENC

  Data record struct for RIO-471xx and RIO-472xx PLCs. Includes encoder fields.
- struct GDataRecord47300_ENC

  Data record struct for RIO-47300. Includes encoder fields.
- struct GDataRecord47300_24EX

  Data record struct for RIO-47300 with 24EX I/O daughter board.
- struct GDataRecord47162

  Data record struct for RIO-47162.

Typedefs

- using UB = System.Byte
- using UW = System.UInt16
- using SW = System.Int16
- using SL = System.Int32
- using UL = System.UInt32

10.183.1 Typedef Documentation

10.183.1.1 SL

```
using SL = System.Int32
```

Definition at line 16 of file gclib.cs.

10.183.1.2 SW

```
using SW = System.Int16
```

Definition at line 15 of file gclib.cs.

10.183.1.3 UB

```
using UB = System.Byte
```

Definition at line 13 of file gclib.cs.

10.183.1.4 UL

```
using UL = System.UInt32
```

Definition at line 17 of file gclib.cs.

10.183.1.5 UW

```
using UW = System.UInt16
```

Definition at line 14 of file gclib.cs.

10.184 gclib.cs

Go to the documentation of this file.

```
00001
00005 using System;
00006 using System.Collections.Generic;
00007 using System.Linq;
00008 using System.Text;
00009 using System.Threading.Tasks;
00010 using System.Runtime.InteropServices; //dll import
00011 using System.IO; //file.exists
00012
00013 using UB = System.Byte;  //unsigned byte
00014 using UW = System.UInt16; //unsigned word
00015 using SW = System.Int16;  //signed word
00016 using SL = System.Int32;  //signed long
00017 using UL = System.UInt32; //unsigned long
00018
00019 #if x86 //defined in "Conditional compilation symbols" of Project Properties
00020 using GReturn = System.Int32;
00021 using GCon = System.IntPtr;
00022 using GSize = System.UInt32;
00023 using GOption = System.Int32;
00024 using GCStringOut = System.Text.StringBuilder;
00025 using GCStringIn = System.String;
00026 using GBufOut = System.Text.StringBuilder;
00027 using GBufIn = System.String;
00028 using GStatus = System.Byte;
00029 // IMPORTANT! Be sure that the paths below are correct
00030 public static class LibraryPath
00031 {
00032     public const string GclibDllPath_ = "C:\\Program Files (x86)\\Galil\\gclib\\dll\\x86\\gclib.dll";
00033     public const string GcliboDllPath_ = "C:\\Program Files (x86)\\Galil\\gclib\\dll\\x86\\gclibO.dll";
00034 }
```

```
00035
00036 #elif x64
00037 using GReturn = System.Int32;
00038 using GCon = System.IntPtr;
00039 using GSize = System.UInt32;
00040 using GOption = System.Int32;
00041 using GCStringOut = System.Text.StringBuilder;
00042 using GCStringIn = System.String;
00043 using GBufOut = System.Text.StringBuilder;
00044 using GBufIn = System.String;
00045 using GStatus = System.Byte;
00046 // IMPORTANT! Be sure that the paths below are correct
00047 public static class LibraryPath
00048 {
00049     public const string GclibDllPath_ = "C:\\Program Files (x86)\\Galil\\gclib\\dll\\x64\\gclib.dll";
00050     public const string GcliboDllPath_ = "C:\\Program Files (x86)\\Galil\\gclib\\dll\\x64\\gclibo.dll";
00051 }
00052
00053 #endif
00054
00067 public class gclib
00068 {
00069     #region "C# wrappers of gclib C calls"
00070
00071     #region "Private properties"
00072     private const int BufferSize_ = 500000; //size of "char *" buffer. Big enough to fit entire 4000
      program via UL/LS, or 24000 elements of array data.
00073     private GCStringOut Buffer_ = new System.Text.StringBuilder(BufferSize_); //used to pass a "char
      *" to gclib.
00074     private byte[] ByteArray_ = new byte[512]; //byte array to hold data record and response to GRead
00075     private GCon ConnectionHandle_; //keep track of the gclib connection handle.
00076     private bool ConnectionStatus_ = false; //keep track of the status of gclib's connection
00077     #endregion
00078
00079
00085     public gclib()
00086     {
00087         if (!File.Exists(LibraryPath.GclibDllPath_))
00088             throw new System.Exception("Could not find gclib dll at " + LibraryPath.GclibDllPath_);
00089
00090         if (!File.Exists(LibraryPath.GcliboDllPath_))
00091             throw new System.Exception("Could not find gclibo dll at " + LibraryPath.GcliboDllPath_);
00092
00093     }
00094
00102     public string[] GAddresses()
00103     {
00104         GReturn rc = DllGAddresses(Buffer_, BufferSize_);
00105         if (rc == G_NO_ERROR)
00106         {
00107             char[] delimiters = new char[] { '\r', '\n' };
00108             return Buffer_.ToString().Split(delimiters, System.StringSplitOptions.RemoveEmptyEntries);
00109         }
00110         else
00111             return new string[0];
00112
00113     }
00114
00126     public void GArrayDownload(string array_name, ref List<double> data, Int16 first = -1, Int16 last
      = -1)
00127     {
00128         System.Text.StringBuilder ArrayData = new System.Text.StringBuilder(BufferSize_); //for
      converting to ASCII
00129         int len = data.Count();
00130         for (int i = 0; i <= len - 1; i++)
00131         {
00132             ArrayData.Append(data[i].ToString("F4")); //format to fixed point
00133             if (i < len - 1)
00134             {
00135                 ArrayData.Append(","); //delimiter
00136             }
00137         }
00138         GReturn rc = DllGArrayDownload(ConnectionHandle_, array_name, first, last,
      ArrayData.ToString());
00139         if (!(rc == G_NO_ERROR))
00140         {
00141             throw new System.Exception(GError(rc));
00142         }
00143     }
00144
00153     public void GArrayDownloadFile(string Path)
00154     {
00155         GReturn rc = DllGArrayDownloadFile(ConnectionHandle_, Path);
00156         if (rc != G_NO_ERROR)
00157         {
00158             throw new System.Exception(GError(rc));
00159         }
```

```
00160     }
00161
00173     public List<double> GArrayUpload(string array_name, Int16 first = -1, Int16 last = -1)
00174     {
00175         List<double> array = new List<double>();
00176         GReturn rc = DllGArrayUpload(ConnectionHandle_, array_name, first, last, 1, Buffer_,
      BufferSize_);
00177         //1 = comma delim
00178         if (!(rc == G_NO_ERROR))
00179         {
00180             throw new System.Exception(GError(rc));
00181         }
00182         char[] delimiters = new char[] { ',' };
00183
00184         string[] tokens = Buffer_.ToString().Split(delimiters,
      System.StringSplitOptions.RemoveEmptyEntries);
00185         double value = 0;
00186         foreach (string s in tokens)
00187         {
00188             if (!double.TryParse(s, out value))
00189             {
00190                 throw new System.Exception("Could not parse " + s + " into double");
00191             }
00192             array.Add(value);
00193         }
00194         return array;
00195     }
00196
00206     public void GArrayUploadFile(string Path, string Names)
00207     {
00208         GReturn rc = DllGArrayUploadFile(ConnectionHandle_, Path, Names);
00209         if (rc != G_NO_ERROR)
00210         {
00211             throw new System.Exception(GError(rc));
00212         }
00213     }
00214
00224     public void GAssign(string ip, string mac)
00225     {
00226         GReturn rc = DllGAssign(ip, mac);
00227         if (!(rc == G_NO_ERROR))
00228         {
00229             throw new System.Exception(GError(rc));
00230         }
00231     }
00232
00239     public void GClose()
00240     {
00241         if(ConnectionStatus_)
00242             DllGClose(ConnectionHandle_);
00243
00244         ConnectionStatus_ = false;
00245     }
00246
00257     public string GCommand(string Command, bool Trim = true)
00258     {
00259         GSize bytes_read = 0;
00260         GReturn rc = DllGCommand(ConnectionHandle_, Command, Buffer_, BufferSize_, ref bytes_read);
00261         if (rc != G_NO_ERROR)
00262         {
00263             throw new System.Exception(GError(rc));
00264         }
00265         if (Trim)
00266         {
00267             string r = Buffer_.ToString();
00268             if (r[r.Count() - 1] == ':')
00269             {
00270                 r = r.Substring(0, r.Count() - 1);
00271             }
00272             return r.Trim();
00273         }
00274         else
00275         {
00276             return Buffer_.ToString();
00277         }
00278     }
00279
00288     public Int16 GCmdI(string Command)
00289     {
00290         return Convert.ToInt16(Convert.ToDouble(GCommand(Command)));
00291     }
00292
00301     public double GCmdD(string Command)
00302     {
00303         return Convert.ToDouble(GCommand(Command));
00304     }
00305
```

```
00316     private string GError(GReturn ErrorCode)
00317     {
00318         DllGError(ErrorCode, Buffer_, BufferSize_);
00319         return ErrorCode.ToString() + " " + Buffer_.ToString() + "\n";
00320     }
00321
00330     public void GFirmwareDownload(string filepath)
00331     {
00332         GReturn rc = DllGFirmwareDownload(ConnectionHandle_, filepath);
00333         if (rc != G_NO_ERROR)
00334         {
00335             throw new System.Exception(GError(rc));
00336         }
00337     }
00338
00344     public string GInfo()
00345     {
00346         GReturn rc = DllGInfo(ConnectionHandle_, Buffer_, BufferSize_);
00347         if (rc == G_NO_ERROR)
00348         {
00349             return Buffer_.ToString();
00350         }
00351         else
00352         {
00353             return "";
00354         }
00355     }
00356
00364     public byte GInterrupt()
00365     {
00366         byte StatusByte = 0;
00367         GReturn rc = DllGInterrupt(ConnectionHandle_, ref StatusByte);
00368         if (rc == G_NO_ERROR)
00369         {
00370             return StatusByte;
00371         }
00372         else
00373         {
00374             return 0;
00375         }
00376     }
00377
00386     public string[] GIpRequests()
00387     {
00388         GReturn rc = DllGIpRequests(Buffer_, BufferSize_);
00389         if (rc == G_NO_ERROR)
00390         {
00391             char[] delimiters = new char[] { '\r', '\n' };
00392             return Buffer_.ToString().Split(delimiters, System.StringSplitOptions.RemoveEmptyEntries);
00393         }
00394         else
00395             return new string[0];
00396
00397     }
00398
00407     public string GMessage()
00408     {
00409         GReturn rc = DllGMessage(ConnectionHandle_, Buffer_, BufferSize_);
00410         if (rc == G_NO_ERROR)
00411         {
00412             return Buffer_.ToString();
00413         }
00414         else
00415         {
00416             return "";
00417         }
00418     }
00419
00428     public void GMotionComplete(string axes)
00429     {
00430         GReturn rc = DllGMotionComplete(ConnectionHandle_, axes);
00431         if (!(rc == G_NO_ERROR))
00432         {
00433             throw new System.Exception(GError(rc));
00434         }
00435     }
00436
00445     public void GOpen(string address)
00446     {
00447         GReturn rc = DllGOpen(address, ref ConnectionHandle_);
00448         if (rc != G_NO_ERROR)
00449         {
00450             throw new System.Exception(GError(rc));
00451         }
00452         else
00453             ConnectionStatus_ = true;
00454     }
```

```
00455
00465     public void GProgramDownload(string program, string preprocessor = "")
00466     {
00467         GReturn rc = DllGProgramDownload(ConnectionHandle_, program, preprocessor);
00468         if (rc != G_NO_ERROR)
00469         {
00470             throw new System.Exception(GError(rc));
00471         }
00472     }
00473
00483     public void GProgramDownloadFile(string file_path, string preprocessor = "")
00484     {
00485         GReturn rc = DllGProgramDownloadFile(ConnectionHandle_, file_path, preprocessor);
00486         if (rc != G_NO_ERROR)
00487         {
00488             throw new System.Exception(GError(rc));
00489         }
00490     }
00491
00499     public string GProgramUpload()
00500     {
00501         GReturn rc = DllGProgramUpload(ConnectionHandle_, Buffer_, BufferSize_);
00502         if (rc != G_NO_ERROR)
00503         {
00504             throw new System.Exception(GError(rc));
00505         }
00506         else
00507         {
00508             return Buffer_.ToString();
00509         }
00510     }
00511
00520     public void GProgramUploadFile(string file_path)
00521     {
00522         GReturn rc = DllGProgramUploadFile(ConnectionHandle_, file_path);
00523         if (rc != G_NO_ERROR)
00524         {
00525             throw new System.Exception(GError(rc));
00526         }
00527     }
00528
00536     public byte[] GRead()
00537     {
00538         GSize read = 0;
00539         GReturn rc = DllGRead(ConnectionHandle_, ByteArray_, (uint)ByteArray_.Length, ref read);
00540         if (rc == G_NO_ERROR)
00541         {
00542             byte[] ReturnData = new byte[read];
00543             //create an array of the correct size
00544             for (GSize i = 0; i <= read - 1; i++)
00545             {
00546                 ReturnData[i] = ByteArray_[i];
00547                 //copy over the data
00548             }
00549             return ReturnData;
00550         }
00551         else
00552             return new byte[0];
00553     }
00554
00566     public T GRecord<T>(bool async)
00567             where T : struct, GDataRecord
00568     {
00569         ushort method = 0;
00570         if (async)
00571             method = 1;
00572
00573         GReturn rc = DllGRecord(ConnectionHandle_, ByteArray_, method);
00574         if (rc != G_NO_ERROR)
00575             throw new System.Exception(GError(rc));
00576
00577         return ByteArrayToDataRecord<T>(ByteArray_);
00578     }
00579
00588     public void GRecordRate(double period_ms)
00589     {
00590         GReturn rc = DllGRecordRate(ConnectionHandle_, period_ms);
00591         if (!(rc == G_NO_ERROR))
00592         {
00593             throw new System.Exception(GError(rc));
00594         }
00595     }
00596
00604     public void GTimeout(Int16 timeout_ms)
00605     {
00606         DllGTimeout(ConnectionHandle_, timeout_ms);
00607     }
```

```
00608
00614     public string GVersion()
00615     {
00616         GReturn rc = DllGVersion(Buffer_, BufferSize_);
00617         if (rc == G_NO_ERROR)
00618         {
00619             return Buffer_.ToString();
00620         }
00621         else
00622         {
00623             return "";
00624         }
00625     }
00626
00635     public void GWrite(string buffer)
00636     {
00637         GReturn rc = DllGWrite(ConnectionHandle_, buffer, (uint) buffer.Length);
00638         if (!(rc == G_NO_ERROR))
00639         {
00640             throw new System.Exception(GError(rc));
00641         }
00642     }
00643
00655     public string[] GSetupDownloadFile(string path, Int32 options)
00656   {
00657     GReturn rc = DllGSetupDownloadFile(ConnectionHandle_, path, options, Buffer_, BufferSize_);
00658
00659     string ret_buf = Buffer_.ToString();
00660         ret_buf = ret_buf.Replace("\r\n", ", ");
00661
00662     if (options != 0)
00663     {
00664       if (rc != G_NO_ERROR)
00665       {
00666         throw new System.Exception(GError(rc));
00667       }
00668     }
00669     else
00670     {
00671       ret_buf += "\"options\"," + rc + "\n";
00672     }
00673
00674         char[] delimiters = new char[] { '\n' };
00675         return ret_buf.ToString().Split(delimiters, System.StringSplitOptions.RemoveEmptyEntries);
00676   }
00677
00686     public void GSetServer(string server_name)
00687     {
00688         GReturn rc = DllGSetServer(server_name);
00689
00690         if(rc != G_NO_ERROR)
00691         {
00692             throw new System.Exception(GError(rc));
00693         }
00694     }
00695
00701     public string GServerStatus()
00702     {
00703         GReturn rc = DllGServerStatus(Buffer_, BufferSize_);
00704
00705         if(rc == G_NO_ERROR)
00706             return Buffer_.ToString();
00707         else
00708             throw new System.Exception(GError(rc));
00709     }
00710
00716     public string[] GListServers()
00717     {
00718         GReturn rc = DllGListServers(Buffer_, BufferSize_);
00719
00720         if(rc == G_NO_ERROR)
00721         {
00722             char[] delimiters = new char[] { '\n' };
00723             return Buffer_.ToString().Split(delimiters, System.StringSplitOptions.RemoveEmptyEntries);
00724         }
00725         else
00726         {
00727             throw new System.Exception(GError(rc));
00728         }
00729     }
00730
00738     public void GPublishServer(string server_name, bool publish, bool save)
00739     {
00740         GReturn rc = DllGPublishServer(server_name, Convert.ToInt16(publish), Convert.ToInt16(save));
00741
00742         if (rc != G_NO_ERROR)
00743             throw new System.Exception(GError(rc));
```

```
00744     }
00745
00751     public string[] GRemoteConnections()
00752     {
00753         GReturn rc = DllGRemoteConnections(Buffer_, BufferSize_);
00754
00755         if(rc == G_NO_ERROR)
00756         {
00757             char[] delimiters = new char[] { '\n' };
00758             return Buffer_.ToString().Split(delimiters, System.StringSplitOptions.RemoveEmptyEntries);
00759         }
00760         else
00761         {
00762             throw new System.Exception(GError(rc));
00763         }
00764     }
00765
00766     #endregion
00767
00768     #region "DLL Imports"
00769     //Import declarations for gclib functions. Functions are private to this class and are prefixed
      with "Dll" to distinguish from C# functions.
00770
00771     #region "Error Codes"
00776     private const Int32 G_NO_ERROR = 0;
00777     #endregion
00778
00779     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GAddresses", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00780     private static extern GReturn DllGAddresses(GCStringOut addresses, GSize addresses_len);
00781
00782     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GArrayDownload", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00783     private static extern GReturn DllGArrayDownload(GCon g, GCStringIn array_name, GOption first,
      GOption last, GCStringIn buffer);
00784
00785     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GArrayDownloadFile", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00786     private static extern GReturn DllGArrayDownloadFile(GCon g, GCStringIn path);
00787
00788     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GArrayUpload", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00789     private static extern GReturn DllGArrayUpload(GCon g, GCStringIn array_name, GOption first,
      GOption last, GOption delim, GCStringOut buffer, GSize bufferLength);
00790
00791     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GArrayUploadFile", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00792     private static extern GReturn DllGArrayUploadFile(GCon g, GCStringIn path, GCStringIn names);
00793
00794     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GAssign", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00795     private static extern GReturn DllGAssign(GCStringIn ip, GCStringIn mac);
00796
00797     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GClose", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00798     private static extern GReturn DllGClose(GCon g);
00799
00800     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GCommand", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00801     private static extern GReturn DllGCommand(GCon g, GCStringIn command, GCStringOut buffer, GSize
      bufferLength, ref GSize bytesReturned);
00802
00803     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GError", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00804     private static extern void DllGError(GReturn error_code, GCStringOut errorbuf, GSize error_len);
00805
00806     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GFirmwareDownload", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00807     private static extern GReturn DllGFirmwareDownload(GCon g, GCStringIn path);
00808
00809     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GInfo", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00810     private static extern GReturn DllGInfo(GCon g, GCStringOut info, GSize infoLength);
00811
00812     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GInterrupt", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00813     private static extern GReturn DllGInterrupt(GCon g, ref GStatus status_byte);
00814
00815     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GIpRequests", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00816     private static extern GReturn DllGIpRequests(GCStringOut requests, GSize requests_len);
00817
00818     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GMessage", CharSet = CharSet.Ansi,
      CallingConvention = CallingConvention.StdCall)]
00819     private static extern GReturn DllGMessage(GCon g, GCStringOut buffer, GSize bufferLength);
00820
00821     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GMotionComplete", CharSet = CharSet.Ansi,
```

```
    CallingConvention = CallingConvention.StdCall)]
00822     private static extern GReturn DllGMotionComplete(GCon g, GCStringIn axes);
00823
00824     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GOpen", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00825     private static extern GReturn DllGOpen(GCStringIn address, ref GCon g);
00826
00827     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GProgramDownload", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00828     private static extern GReturn DllGProgramDownload(GCon g, GCStringIn program, GCStringIn
    preprocessor);
00829
00830     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GProgramDownloadFile", CharSet =
    CharSet.Ansi, CallingConvention = CallingConvention.StdCall)]
00831     private static extern GReturn DllGProgramDownloadFile(GCon g, GCStringIn path, GCStringIn
    preprocessor);
00832
00833     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GProgramUpload", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00834     private static extern GReturn DllGProgramUpload(GCon g, GCStringOut buffer, GSize bufferLength);
00835
00836     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GProgramUploadFile", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00837     private static extern GReturn DllGProgramUploadFile(GCon g, GCStringIn path);
00838
00839     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GRead", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00840     private static extern GReturn DllGRead(GCon g, byte[] record, GSize buffer_len, ref GSize
    bytes_read);
00841
00842     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GRecord", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00843     private static extern GReturn DllGRecord(GCon g, byte[] record, GOption method);
00844
00845     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GRecordRate", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00846     private static extern GReturn DllGRecordRate(GCon g, double period_ms);
00847
00848     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GTimeout", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00849     private static extern void DllGTimeout(GCon g, GOption timeoutMs);
00850
00851     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GVersion", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00852     private static extern GReturn DllGVersion(GCStringOut ver, GSize ver_len);
00853
00854     [DllImport(LibraryPath.GclibDllPath_, EntryPoint = "GWrite", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00855     private static extern GReturn DllGWrite(GCon g, GCStringIn buffer, GSize buffer_len);
00856
00857   [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GSetupDownloadFile", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00858     private static extern GReturn DllGSetupDownloadFile(GCon g, GCStringIn file_path, GOption options,
    GCStringOut info, GSize info_len);
00859
00860     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GSetServer", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00861     private static extern GReturn DllGSetServer(GCStringIn server_name);
00862
00863     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GServerStatus", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00864     private static extern GReturn DllGServerStatus(GCStringOut status, GSize status_len);
00865
00866     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GListServers", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00867     private static extern GReturn DllGListServers(GCStringOut servers, GSize servers_len);
00868
00869     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GPublishServer", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00870     private static extern GReturn DllGPublishServer(GCStringIn name, GOption publish, GOption save);
00871
00872     [DllImport(LibraryPath.GcliboDllPath_, EntryPoint = "GRemoteConnections", CharSet = CharSet.Ansi,
    CallingConvention = CallingConvention.StdCall)]
00873     private static extern GReturn DllGRemoteConnections(GCStringOut connections, GSize
    connections_len);
00874
00875     #endregion
00876
00877     #region "Data Record"
00878
00879     private T ByteArrayToDataRecord<T>(byte[] array)
00880         where T : struct, GDataRecord
00881     {
00882         GCHandle handle = GCHandle.Alloc(array, GCHandleType.Pinned);
00883         try
00884         {
00885             return Marshal.PtrToStructure<T>(handle.AddrOfPinnedObject());
```

```
00886         }
00887         catch
00888         {
00889             return default(T);
00890         }
00891         finally
00892         {
00893             handle.Free();
00894         }
00895     }
00896 
00897     public interface GDataRecord
00898     {
00900         byte[] byte_array();
00901     }
00902 
00903     private static byte[] StructToByteArray(GDataRecord record) //Returns this DataRecord as a byte[]
00904     {
00905         int size = Marshal.SizeOf(record);
00906         byte[] arr = new byte[size];
00907 
00908         IntPtr ptr = Marshal.AllocHGlobal(size);
00909         Marshal.StructureToPtr(record, ptr, true);
00910         Marshal.Copy(ptr, arr, 0, size);
00911         Marshal.FreeHGlobal(ptr);
00912         return arr;
00913     }
00914 
00915 
00917     [StructLayout(LayoutKind.Sequential, Pack=1)]
00918     public struct GDataRecord4000 : GDataRecord
00919     {
00920         public byte[] byte_array() { return StructToByteArray(this); }
00921         /*Offset   type name        description*/
00922 
00923         /*00*/      public UB header_0;
00924       /*01*/      public UB header_1;
00925       /*02*/      public UB header_2;
00926       /*03*/      public UB header_3;
00927 
00928       /*04-05*/   public UW sample_number;
00929 
00930       /*06*/      public UB input_bank_0;
00931       /*07*/      public UB input_bank_1;
00932       /*08*/      public UB input_bank_2;
00933       /*09*/      public UB input_bank_3;
00934       /*10*/      public UB input_bank_4;
00935       /*11*/      public UB input_bank_5;
00936       /*12*/      public UB input_bank_6;
00937       /*13*/    public UB input_bank_7;
00938       /*14*/    public UB input_bank_8;
00939       /*15*/      public UB input_bank_9;
00940 
00941       /*16*/      public UB output_bank_0;
00942       /*17*/      public UB output_bank_1;
00943       /*18*/    public UB output_bank_2;
00944       /*19*/      public UB output_bank_3;
00945       /*20*/    public UB output_bank_4;
00946       /*21*/      public UB output_bank_5;
00947       /*22*/      public UB output_bank_6;
00948       /*23*/      public UB output_bank_7;
00949       /*24*/    public UB output_bank_8;
00950       /*25*/    public UB output_bank_9;
00951 
00952       /*26-27*/ public SW  reserved_0;
00953       /*28-29*/ public SW   reserved_2;
00954       /*30-31*/ public SW   reserved_4;
00955       /*32-33*/ public SW reserved_6;
00956       /*34-35*/ public SW reserved_8;
00957       /*36-37*/ public SW reserved_10;
00958       /*38-39*/ public SW reserved_12;
00959       /*40-41*/ public SW reserved_14;
00960 
00961       /*42*/      public UB ethernet_status_a;
00962       /*43*/    public UB ethernet_status_b;
00963       /*44*/      public UB ethernet_status_c;
00964       /*45*/    public UB ethernet_status_d;
00965       /*46*/    public UB ethernet_status_e;
00966       /*47*/      public UB ethernet_status_f;
00967       /*48*/    public UB ethernet_status_g;
00968       /*49*/    public UB ethernet_status_h;
00969 
00970       /*50*/      public UB error_code;
00971       /*51*/    public UB thread_status;
00972       /*52-55*/ public UL amplifier_status;
00973 
00974       /*56-59*/ public UL contour_segment_count;
```

```
00975       /*60-61*/ public UW contour_buffer_available;
00976
00977       /*62-63*/ public UW s_plane_segment_count;
00978       /*64-65*/ public UW s_plane_move_status;
00979       /*66-69*/ public SL s_distance;
00980       /*70-71*/ public UW s_plane_buffer_available;
00981
00982       /*72-73*/ public UW t_plane_segment_count;
00983       /*74-75*/ public UW t_plane_move_status;
00984       /*76-79*/ public SL t_distance;
00985       /*80-81*/ public UW t_plane_buffer_available;
00986
00987       /*82-83*/ public UW axis_a_status;
00988       /*84*/      public UB axis_a_switches;
00989       /*85*/      public UB axis_a_stop_code;
00990       /*86-89*/ public SL axis_a_reference_position;
00991       /*90-93*/ public SL axis_a_motor_position;
00992       /*94-97*/ public SL axis_a_position_error;
00993       /*98-101*/  public SL axis_a_aux_position;
00994       /*102-105*/ public SL axis_a_velocity;
00995       /*106-109*/ public SL   axis_a_torque;
00996       /*110-111*/ public UW  axis_a_analog_in;
00997       /*112*/     public UB axis_a_halls;
00998       /*113*/   public UB axis_a_reserved;
00999       /*114-117*/ public SL axis_a_variable;
01000
01001       /*118-119*/ public UW axis_b_status;
01002       /*120*/     public UB axis_b_switches;
01003       /*121*/     public UB axis_b_stop_code;
01004       /*122-125*/ public SL axis_b_reference_position;
01005       /*126-129*/ public SL axis_b_motor_position;
01006       /*130-133*/ public SL axis_b_position_error;
01007       /*134-137*/ public SL axis_b_aux_position;
01008       /*138-141*/ public SL axis_b_velocity;
01009       /*142-145*/ public SL axis_b_torque;
01010       /*146-147*/ public UW  axis_b_analog_in;
01011       /*148*/     public UB axis_b_halls;
01012       /*149*/     public UB axis_b_reserved;
01013       /*150-153*/ public SL axis_b_variable;
01014
01015       /*154-155*/ public UW axis_c_status;
01016       /*156*/   public UB axis_c_switches;
01017       /*157*/   public UB axis_c_stop_code;
01018       /*158-161*/ public SL axis_c_reference_position;
01019       /*162-165*/ public SL axis_c_motor_position;
01020       /*166-169*/ public SL axis_c_position_error;
01021       /*170-173*/ public SL axis_c_aux_position;
01022       /*174-177*/ public SL axis_c_velocity;
01023       /*178-181*/ public SL axis_c_torque;
01024       /*182-183*/ public UW   axis_c_analog_in;
01025       /*184*/     public UB axis_c_halls;
01026       /*185*/     public UB axis_c_reserved;
01027       /*186-189*/ public SL axis_c_variable;
01028
01029       /*190-191*/ public UW axis_d_status;
01030       /*192*/     public UB axis_d_switches;
01031       /*193*/   public UB axis_d_stop_code;
01032       /*194-197*/ public SL axis_d_reference_position;
01033       /*198-201*/ public SL axis_d_motor_position;
01034       /*202-205*/ public SL axis_d_position_error;
01035       /*206-209*/ public SL axis_d_aux_position;
01036       /*210-213*/ public SL axis_d_velocity;
01037       /*214-217*/ public SL axis_d_torque;
01038       /*218-219*/ public UW  axis_d_analog_in;
01039       /*220*/   public UB axis_d_halls;
01040       /*221*/   public UB axis_d_reserved;
01041       /*222-225*/ public SL axis_d_variable;
01042
01043       /*226-227*/ public UW axis_e_status;
01044       /*228*/     public UB axis_e_switches;
01045       /*229*/     public UB axis_e_stop_code;
01046       /*230-233*/ public SL axis_e_reference_position;
01047       /*234-237*/ public SL axis_e_motor_position;
01048       /*238-241*/ public SL axis_e_position_error;
01049       /*242-245*/ public SL axis_e_aux_position;
01050       /*246-249*/ public SL axis_e_velocity;
01051       /*250-253*/ public SL axis_e_torque;
01052       /*254-255*/ public UW  axis_e_analog_in;
01053       /*256*/   public UB axis_e_halls;
01054       /*257*/     public UB axis_e_reserved;
01055       /*258-261*/ public SL axis_e_variable;
01056
01057       /*262-263*/ public UW axis_f_status;
01058       /*264*/     public UB axis_f_switches;
01059       /*265*/     public UB axis_f_stop_code;
01060       /*266-269*/ public SL axis_f_reference_position;
01061       /*270-273*/ public SL axis_f_motor_position;
```

```
01062       /*274-277*/ public SL axis_f_position_error;
01063       /*278-281*/ public SL axis_f_aux_position;
01064       /*282-285*/ public SL axis_f_velocity;
01065       /*286-289*/ public SL axis_f_torque;
01066       /*290-291*/ public UW axis_f_analog_in;
01067       /*292*/     public UB axis_f_halls;
01068       /*293*/     public UB axis_f_reserved;
01069       /*294-297*/ public SL axis_f_variable;
01070
01071       /*298-299*/ public UW axis_g_status;
01072       /*300*/   public UB axis_g_switches;
01073       /*301*/   public UB axis_g_stop_code;
01074       /*302-305*/ public SL axis_g_reference_position;
01075       /*306-309*/ public SL axis_g_motor_position;
01076       /*310-313*/ public SL axis_g_position_error;
01077       /*314-317*/ public SL axis_g_aux_position;
01078       /*318-321*/ public SL axis_g_velocity;
01079       /*322-325*/ public SL axis_g_torque;
01080       /*326-327*/ public UW  axis_g_analog_in;
01081       /*328*/     public UB axis_g_halls;
01082       /*329*/     public UB axis_g_reserved;
01083       /*330-333*/ public SL axis_g_variable;
01084
01085       /*334-335*/ public UW axis_h_status;
01086       /*336*/   public UB axis_h_switches;
01087       /*337*/     public UB axis_h_stop_code;
01088       /*338-341*/ public SL axis_h_reference_position;
01089       /*342-345*/ public SL axis_h_motor_position;
01090       /*346-349*/ public SL axis_h_position_error;
01091       /*350-353*/ public SL axis_h_aux_position;
01092       /*354-357*/ public SL axis_h_velocity;
01093       /*358-361*/ public SL axis_h_torque;
01094       /*362-363*/ public UW  axis_h_analog_in;
01095       /*364*/     public UB axis_h_halls;
01096       /*365*/   public UB axis_h_reserved;
01097       /*366-369*/ public SL axis_h_variable;
01098     }; //DataRecord4000
01099
01101     [StructLayout(LayoutKind.Sequential, Pack=1)]
01102     public struct GDataRecord52000 : GDataRecord
01103     {
01104         public byte[] byte_array() { return StructToByteArray(this); }
01105         /*Offset   type name        description*/
01106
01107         /*00*/      public UB header_0;
01108       /*01*/      public UB header_1;
01109       /*02*/      public UB header_2;
01110       /*03*/      public UB header_3;
01111
01112       /*04-05*/   public UW sample_number;
01113
01114       /*06*/      public UB input_bank_0;
01115       /*07*/      public UB input_bank_1;
01116       /*08*/      public UB input_bank_2;
01117       /*09*/      public UB input_bank_3;
01118       /*10*/      public UB input_bank_4;
01119       /*11*/      public UB input_bank_5;
01120       /*12*/      public UB input_bank_6;
01121       /*13*/    public UB input_bank_7;
01122       /*14*/    public UB input_bank_8;
01123       /*15*/      public UB input_bank_9;
01124
01125       /*16*/      public UB output_bank_0;
01126       /*17*/      public UB output_bank_1;
01127       /*18*/    public UB output_bank_2;
01128       /*19*/      public UB output_bank_3;
01129       /*20*/    public UB output_bank_4;
01130       /*21*/      public UB output_bank_5;
01131       /*22*/      public UB output_bank_6;
01132       /*23*/      public UB output_bank_7;
01133       /*24*/    public UB output_bank_8;
01134       /*25*/    public UB output_bank_9;
01135
01136       /*26-27*/ public SW  reserved_0;
01137       /*28-29*/ public SW   reserved_2;
01138       /*30-31*/ public SW   reserved_4;
01139       /*32-33*/ public SW reserved_6;
01140       /*34-35*/ public SW reserved_8;
01141       /*36-37*/ public SW reserved_10;
01142       /*38-39*/ public SW reserved_12;
01143       /*40*/      public UB  ethercat_bank;
01144       /*41*/    public UB reserved_14;
01145
01146       /*42*/      public UB ethernet_status_a;
01147       /*43*/    public UB ethernet_status_b;
01148       /*44*/      public UB ethernet_status_c;
01149       /*45*/    public UB ethernet_status_d;
```

```
01150       /*46*/    public UB ethernet_status_e;
01151       /*47*/      public UB ethernet_status_f;
01152       /*48*/    public UB ethernet_status_g;
01153       /*49*/    public UB ethernet_status_h;
01154
01155       /*50*/      public UB error_code;
01156       /*51*/    public UB thread_status;
01157       /*52-55*/ public UL amplifier_status;
01158
01159       /*56-59*/ public UL contour_segment_count;
01160       /*60-61*/ public UW contour_buffer_available;
01161
01162       /*62-63*/ public UW s_plane_segment_count;
01163       /*64-65*/ public UW s_plane_move_status;
01164       /*66-69*/ public SL s_distance;
01165       /*70-71*/ public UW s_plane_buffer_available;
01166
01167       /*72-73*/ public UW t_plane_segment_count;
01168       /*74-75*/ public UW t_plane_move_status;
01169       /*76-79*/ public SL t_distance;
01170       /*80-81*/ public UW t_plane_buffer_available;
01171
01172       /*82-83*/ public UW axis_a_status;
01173       /*84*/      public UB axis_a_switches;
01174       /*85*/      public UB axis_a_stop_code;
01175       /*86-89*/ public SL axis_a_reference_position;
01176       /*90-93*/ public SL axis_a_motor_position;
01177       /*94-97*/ public SL axis_a_position_error;
01178       /*98-101*/  public SL axis_a_aux_position;
01179       /*102-105*/ public SL axis_a_velocity;
01180       /*106-109*/ public SL   axis_a_torque;
01181       /*110-111*/ public UW  axis_a_analog_in;
01182       /*112*/     public UB axis_a_halls;
01183       /*113*/   public UB axis_a_reserved;
01184       /*114-117*/ public SL axis_a_variable;
01185
01186       /*118-119*/ public UW axis_b_status;
01187       /*120*/     public UB axis_b_switches;
01188       /*121*/     public UB axis_b_stop_code;
01189       /*122-125*/ public SL axis_b_reference_position;
01190       /*126-129*/ public SL axis_b_motor_position;
01191       /*130-133*/ public SL axis_b_position_error;
01192       /*134-137*/ public SL axis_b_aux_position;
01193       /*138-141*/ public SL axis_b_velocity;
01194       /*142-145*/ public SL axis_b_torque;
01195       /*146-147*/ public UW  axis_b_analog_in;
01196       /*148*/     public UB axis_b_halls;
01197       /*149*/     public UB axis_b_reserved;
01198       /*150-153*/ public SL axis_b_variable;
01199
01200       /*154-155*/ public UW axis_c_status;
01201       /*156*/   public UB axis_c_switches;
01202       /*157*/   public UB axis_c_stop_code;
01203       /*158-161*/ public SL axis_c_reference_position;
01204       /*162-165*/ public SL axis_c_motor_position;
01205       /*166-169*/ public SL axis_c_position_error;
01206       /*170-173*/ public SL axis_c_aux_position;
01207       /*174-177*/ public SL axis_c_velocity;
01208       /*178-181*/ public SL axis_c_torque;
01209       /*182-183*/ public UW   axis_c_analog_in;
01210       /*184*/     public UB axis_c_halls;
01211       /*185*/     public UB axis_c_reserved;
01212       /*186-189*/ public SL axis_c_variable;
01213
01214       /*190-191*/ public UW axis_d_status;
01215       /*192*/     public UB axis_d_switches;
01216       /*193*/   public UB axis_d_stop_code;
01217       /*194-197*/ public SL axis_d_reference_position;
01218       /*198-201*/ public SL axis_d_motor_position;
01219       /*202-205*/ public SL axis_d_position_error;
01220       /*206-209*/ public SL axis_d_aux_position;
01221       /*210-213*/ public SL axis_d_velocity;
01222       /*214-217*/ public SL axis_d_torque;
01223       /*218-219*/ public UW  axis_d_analog_in;
01224       /*220*/   public UB axis_d_halls;
01225       /*221*/   public UB axis_d_reserved;
01226       /*222-225*/ public SL axis_d_variable;
01227
01228       /*226-227*/ public UW axis_e_status;
01229       /*228*/     public UB axis_e_switches;
01230       /*229*/     public UB axis_e_stop_code;
01231       /*230-233*/ public SL axis_e_reference_position;
01232       /*234-237*/ public SL axis_e_motor_position;
01233       /*238-241*/ public SL axis_e_position_error;
01234       /*242-245*/ public SL axis_e_aux_position;
01235       /*246-249*/ public SL axis_e_velocity;
01236       /*250-253*/ public SL axis_e_torque;
```

```
01237       /*254-255*/ public UW  axis_e_analog_in;
01238       /*256*/   public UB axis_e_halls;
01239       /*257*/     public UB axis_e_reserved;
01240       /*258-261*/ public SL axis_e_variable;
01241
01242       /*262-263*/ public UW axis_f_status;
01243       /*264*/     public UB axis_f_switches;
01244       /*265*/     public UB axis_f_stop_code;
01245       /*266-269*/ public SL axis_f_reference_position;
01246       /*270-273*/ public SL axis_f_motor_position;
01247       /*274-277*/ public SL axis_f_position_error;
01248       /*278-281*/ public SL axis_f_aux_position;
01249       /*282-285*/ public SL axis_f_velocity;
01250       /*286-289*/ public SL axis_f_torque;
01251       /*290-291*/ public UW axis_f_analog_in;
01252       /*292*/     public UB axis_f_halls;
01253       /*293*/     public UB axis_f_reserved;
01254       /*294-297*/ public SL axis_f_variable;
01255
01256       /*298-299*/ public UW axis_g_status;
01257       /*300*/   public UB axis_g_switches;
01258       /*301*/   public UB axis_g_stop_code;
01259       /*302-305*/ public SL axis_g_reference_position;
01260       /*306-309*/ public SL axis_g_motor_position;
01261       /*310-313*/ public SL axis_g_position_error;
01262       /*314-317*/ public SL axis_g_aux_position;
01263       /*318-321*/ public SL axis_g_velocity;
01264       /*322-325*/ public SL axis_g_torque;
01265       /*326-327*/ public UW  axis_g_analog_in;
01266       /*328*/     public UB axis_g_halls;
01267       /*329*/     public UB axis_g_reserved;
01268       /*330-333*/ public SL axis_g_variable;
01269
01270       /*334-335*/ public UW axis_h_status;
01271       /*336*/   public UB axis_h_switches;
01272       /*337*/     public UB axis_h_stop_code;
01273       /*338-341*/ public SL axis_h_reference_position;
01274       /*342-345*/ public SL axis_h_motor_position;
01275       /*346-349*/ public SL axis_h_position_error;
01276       /*350-353*/ public SL axis_h_aux_position;
01277       /*354-357*/ public SL axis_h_velocity;
01278       /*358-361*/ public SL axis_h_torque;
01279       /*362-363*/ public UW  axis_h_analog_in;
01280       /*364*/     public UB axis_h_halls;
01281       /*365*/   public UB axis_h_reserved;
01282       /*366-369*/ public SL axis_h_variable;
01283     }; //DataRecord52000
01284
01286
01293     [StructLayout(LayoutKind.Sequential, Pack=1)]
01294     public struct GDataRecord1806 : GDataRecord
01295     {
01296         public byte[] byte_array() { return StructToByteArray(this); }
01297         /*Offset   type name        description*/
01298
01299         /*00-01*/   public UW sample_number;
01300
01301       /*02*/      public UB input_bank_0;
01302       /*03*/      public UB input_bank_1;
01303       /*04*/      public UB input_bank_2;
01304       /*05*/      public UB input_bank_3;
01305       /*06*/      public UB input_bank_4;
01306       /*07*/      public UB input_bank_5;
01307       /*08*/      public UB input_bank_6;
01308       /*09*/    public UB input_bank_7;
01309       /*10*/    public UB input_bank_8;
01310       /*11*/      public UB input_bank_9;
01311
01312       /*12*/      public UB output_bank_0;
01313       /*13*/      public UB output_bank_1;
01314       /*14*/    public UB output_bank_2;
01315       /*15*/      public UB output_bank_3;
01316       /*16*/    public UB output_bank_4;
01317       /*17*/      public UB output_bank_5;
01318       /*18*/      public UB output_bank_6;
01319       /*19*/      public UB output_bank_7;
01320       /*20*/    public UB output_bank_8;
01321       /*21*/    public UB output_bank_9;
01322
01323       /*22-23*/ public SW  reserved_0;
01324       /*24-25*/ public SW   reserved_2;
01325       /*26-27*/ public SW   reserved_4;
01326       /*28-29*/ public SW reserved_6;
01327       /*30-31*/ public SW reserved_8;
01328       /*32-33*/ public SW reserved_10;
01329       /*34-35*/ public SW reserved_12;
01330       /*36-37*/ public SW reserved_14;
```

```
01331
01332       /*38*/      public UB reserved_16;
01333       /*39*/    public UB reserved_17;
01334       /*40*/      public UB reserved_18;
01335       /*41*/    public UB reserved_19;
01336       /*42*/    public UB reserved_20;
01337       /*43*/      public UB reserved_21;
01338       /*44*/    public UB reserved_22;
01339       /*45*/    public UB reserved_23;
01340
01341       /*46*/      public UB error_code;
01342       /*47*/    public UB thread_status;
01343       /*48-51*/ public UL reserved_24;
01344
01345       /*52-55*/ public UL contour_segment_count;
01346       /*56-57*/ public UW contour_buffer_available;
01347
01348       /*58-59*/ public UW s_plane_segment_count;
01349       /*60-61*/ public UW s_plane_move_status;
01350       /*62-65*/ public SL s_distance;
01351       /*66-67*/ public UW s_plane_buffer_available;
01352
01353       /*68-69*/ public UW t_plane_segment_count;
01354       /*70-71*/ public UW t_plane_move_status;
01355       /*72-75*/ public SL t_distance;
01356       /*76-77*/ public UW t_plane_buffer_available;
01357
01358       /*78-79*/ public UW axis_a_status;
01359       /*80*/      public UB axis_a_switches;
01360       /*81*/      public UB axis_a_stop_code;
01361       /*82-85*/ public SL axis_a_reference_position;
01362       /*86-89*/ public SL axis_a_motor_position;
01363       /*90-93*/ public SL axis_a_position_error;
01364       /*94-97*/ public SL axis_a_aux_position;
01365       /*98-101*/  public SL axis_a_velocity;
01366       /*102-105*/ public SL   axis_a_torque;
01367       /*106-107*/ public UW  axis_a_analog_in;
01368       /*108*/     public UB axis_a_reserved_0;
01369       /*109*/   public UB axis_a_reserved_1;
01370       /*110-113*/ public SL axis_a_variable;
01371
01372       /*114-115*/ public UW axis_b_status;
01373       /*116*/     public UB axis_b_switches;
01374       /*117*/     public UB axis_b_stop_code;
01375       /*118-121*/ public SL axis_b_reference_position;
01376       /*122-125*/ public SL axis_b_motor_position;
01377       /*126-129*/ public SL axis_b_position_error;
01378       /*130-133*/ public SL axis_b_aux_position;
01379       /*134-137*/ public SL axis_b_velocity;
01380       /*138-141*/ public SL axis_b_torque;
01381       /*142-143*/ public UW  axis_b_analog_in;
01382       /*144*/     public UB axis_b_reserved_0;
01383       /*145*/     public UB axis_b_reserved_1;
01384       /*146-149*/ public SL axis_b_variable;
01385
01386       /*150-151*/ public UW axis_c_status;
01387       /*152*/   public UB axis_c_switches;
01388       /*153*/   public UB axis_c_stop_code;
01389       /*154-157*/ public SL axis_c_reference_position;
01390       /*158-161*/ public SL axis_c_motor_position;
01391       /*162-165*/ public SL axis_c_position_error;
01392       /*166-169*/ public SL axis_c_aux_position;
01393       /*170-173*/ public SL axis_c_velocity;
01394       /*174-177*/ public SL axis_c_torque;
01395       /*178-179*/ public UW   axis_c_analog_in;
01396       /*180*/     public UB axis_c_reserved_0;
01397       /*181*/     public UB axis_c_reserved_1;
01398       /*182-185*/ public SL axis_c_variable;
01399
01400       /*186-187*/ public UW axis_d_status;
01401       /*188*/     public UB axis_d_switches;
01402       /*189*/   public UB axis_d_stop_code;
01403       /*190-193*/ public SL axis_d_reference_position;
01404       /*194-197*/ public SL axis_d_motor_position;
01405       /*198-201*/ public SL axis_d_position_error;
01406       /*202-205*/ public SL axis_d_aux_position;
01407       /*206-209*/ public SL axis_d_velocity;
01408       /*210-213*/ public SL axis_d_torque;
01409       /*214-215*/ public UW  axis_d_analog_in;
01410       /*216*/   public UB axis_d_reserved_0;
01411       /*217*/   public UB axis_d_reserved_1;
01412       /*218-221*/ public SL axis_d_variable;
01413
01414       /*222-223*/ public UW axis_e_status;
01415       /*224*/     public UB axis_e_switches;
01416       /*225*/     public UB axis_e_stop_code;
01417       /*226-229*/ public SL axis_e_reference_position;
```

```
01418       /*230-233*/ public SL axis_e_motor_position;
01419       /*234-237*/ public SL axis_e_position_error;
01420       /*238-241*/ public SL axis_e_aux_position;
01421       /*242-245*/ public SL axis_e_velocity;
01422       /*256-249*/ public SL axis_e_torque;
01423       /*250-251*/ public UW  axis_e_analog_in;
01424       /*252*/   public UB axis_e_reserved_0;
01425       /*253*/     public UB axis_e_reserved_1;
01426       /*254-257*/ public SL axis_e_variable;
01427
01428       /*258-259*/ public UW axis_f_status;
01429       /*260*/     public UB axis_f_switches;
01430       /*261*/     public UB axis_f_stop_code;
01431       /*262-265*/ public SL axis_f_reference_position;
01432       /*266-269*/ public SL axis_f_motor_position;
01433       /*270-273*/ public SL axis_f_position_error;
01434       /*274-277*/ public SL axis_f_aux_position;
01435       /*278-281*/ public SL axis_f_velocity;
01436       /*282-285*/ public SL axis_f_torque;
01437       /*286-287*/ public UW axis_f_analog_in;
01438       /*288*/     public UB axis_f_reserved_0;
01439       /*289*/     public UB axis_f_reserved_1;
01440       /*290-293*/ public SL axis_f_variable;
01441
01442       /*294-295*/ public UW axis_g_status;
01443       /*296*/   public UB axis_g_switches;
01444       /*297*/   public UB axis_g_stop_code;
01445       /*298-301*/ public SL axis_g_reference_position;
01446       /*302-305*/ public SL axis_g_motor_position;
01447       /*306-309*/ public SL axis_g_position_error;
01448       /*310-313*/ public SL axis_g_aux_position;
01449       /*314-317*/ public SL axis_g_velocity;
01450       /*318-321*/ public SL axis_g_torque;
01451       /*322-323*/ public UW  axis_g_analog_in;
01452       /*324*/     public UB axis_g_reserved_0;
01453       /*325*/     public UB axis_g_reserved_1;
01454       /*326-329*/ public SL axis_g_variable;
01455
01456       /*330-331*/ public UW axis_h_status;
01457       /*332*/   public UB axis_h_switches;
01458       /*333*/     public UB axis_h_stop_code;
01459       /*334-337*/ public SL axis_h_reference_position;
01460       /*338-341*/ public SL axis_h_motor_position;
01461       /*342-345*/ public SL axis_h_position_error;
01462       /*346-349*/ public SL axis_h_aux_position;
01463       /*350-353*/ public SL axis_h_velocity;
01464       /*354-357*/ public SL axis_h_torque;
01465       /*358-359*/ public UW  axis_h_analog_in;
01466       /*360*/     public UB axis_h_reserved_0;
01467       /*361*/   public UB axis_h_reserved_1;
01468       /*362-365*/ public SL axis_h_variable;
01469     }; //DataRecord1806
01470
01472     [StructLayout(LayoutKind.Sequential, Pack=1)]
01473     public struct GDataRecord2103 : GDataRecord
01474     {
01475         public byte[] byte_array() { return StructToByteArray(this); }
01476
01477         /*Offset   type name        description*/
01478
01479         /*00*/      public UB header_0;
01480       /*01*/      public UB header_1;
01481       /*02*/      public UB header_2;
01482       /*03*/      public UB header_3;
01483
01484       /*04-05*/   public UW sample_number;
01485
01486       /*06*/      public UB input_bank_0;
01487       /*07*/      public UB input_bank_1;
01488       /*08*/      public UB input_bank_2;
01489       /*09*/      public UB input_bank_3;
01490       /*10*/      public UB input_bank_4;
01491       /*11*/      public UB input_bank_5;
01492       /*12*/      public UB input_bank_6;
01493       /*13*/    public UB input_bank_7;
01494       /*14*/    public UB input_bank_8;
01495       /*15*/      public UB input_bank_9;
01496
01497       /*16*/      public UB output_bank_0;
01498       /*17*/      public UB output_bank_1;
01499       /*18*/    public UB output_bank_2;
01500       /*19*/      public UB output_bank_3;
01501       /*20*/    public UB output_bank_4;
01502       /*21*/      public UB output_bank_5;
01503       /*22*/      public UB output_bank_6;
01504       /*23*/      public UB output_bank_7;
01505       /*24*/    public UB output_bank_8;
```

```
01506       /*25*/    public UB output_bank_9;
01507
01508       /*26*/      public UB error_code;
01509       /*27*/    public UB general_status;
01510
01511       /*28-29*/ public UW s_plane_segment_count;
01512       /*30-31*/ public UW s_plane_move_status;
01513       /*32-35*/ public SL s_distance;
01514
01515       /*36-37*/ public UW t_plane_segment_count;
01516       /*38-39*/ public UW t_plane_move_status;
01517       /*40-43*/ public SL t_distance;
01518
01519       /*44-45*/ public UW axis_a_status;
01520       /*46*/      public UB axis_a_switches;
01521       /*47*/      public UB axis_a_stop_code;
01522       /*48-51*/ public SL axis_a_reference_position;
01523       /*52-55*/ public SL axis_a_motor_position;
01524       /*56-59*/ public SL axis_a_position_error;
01525       /*60-63*/ public SL axis_a_aux_position;
01526       /*64-67*/ public SL axis_a_velocity;
01527       /*68-69*/ public SW   axis_a_torque;
01528       /*70-71*/ public UW  axis_a_analog_in;
01529
01530       /*72-73*/ public UW axis_b_status;
01531       /*74*/      public UB axis_b_switches;
01532       /*75*/      public UB axis_b_stop_code;
01533       /*76-79*/ public SL axis_b_reference_position;
01534       /*80-83*/ public SL axis_b_motor_position;
01535       /*84-87*/ public SL axis_b_position_error;
01536       /*88-91*/ public SL axis_b_aux_position;
01537       /*92-95*/ public SL axis_b_velocity;
01538       /*96-97*/ public SW axis_b_torque;
01539       /*98-99*/ public UW  axis_b_analog_in;
01540
01541       /*100-101*/ public UW axis_c_status;
01542       /*102*/   public UB axis_c_switches;
01543       /*103*/   public UB axis_c_stop_code;
01544       /*104-107*/ public SL axis_c_reference_position;
01545       /*108-111*/ public SL axis_c_motor_position;
01546       /*112-115*/ public SL axis_c_position_error;
01547       /*116-119*/ public SL axis_c_aux_position;
01548       /*120-123*/ public SL axis_c_velocity;
01549       /*124-125*/ public SW axis_c_torque;
01550       /*126-127*/ public UW   axis_c_analog_in;
01551
01552       /*128-129*/ public UW axis_d_status;
01553       /*130*/     public UB axis_d_switches;
01554       /*131*/   public UB axis_d_stop_code;
01555       /*132-135*/ public SL axis_d_reference_position;
01556       /*136-139*/ public SL axis_d_motor_position;
01557       /*140-143*/ public SL axis_d_position_error;
01558       /*144-147*/ public SL axis_d_aux_position;
01559       /*148-151*/ public SL axis_d_velocity;
01560       /*152-153*/ public SW axis_d_torque;
01561       /*154-155*/ public UW  axis_d_analog_in;
01562
01563       /*156-157*/ public UW axis_e_status;
01564       /*158*/     public UB axis_e_switches;
01565       /*159*/     public UB axis_e_stop_code;
01566       /*160-163*/ public SL axis_e_reference_position;
01567       /*164-167*/ public SL axis_e_motor_position;
01568       /*168-171*/ public SL axis_e_position_error;
01569       /*172-175*/ public SL axis_e_aux_position;
01570       /*176-179*/ public SL axis_e_velocity;
01571       /*180-181*/ public SW axis_e_torque;
01572       /*182-183*/ public UW  axis_e_analog_in;
01573
01574       /*184-185*/ public UW axis_f_status;
01575       /*186*/     public UB axis_f_switches;
01576       /*187*/     public UB axis_f_stop_code;
01577       /*188-191*/ public SL axis_f_reference_position;
01578       /*192-195*/ public SL axis_f_motor_position;
01579       /*196-199*/ public SL axis_f_position_error;
01580       /*200-203*/ public SL axis_f_aux_position;
01581       /*204-207*/ public SL axis_f_velocity;
01582       /*208-209*/ public SW axis_f_torque;
01583       /*210-211*/ public UW axis_f_analog_in;
01584
01585       /*212-213*/ public UW axis_g_status;
01586       /*214*/   public UB axis_g_switches;
01587       /*215*/   public UB axis_g_stop_code;
01588       /*216-219*/ public SL axis_g_reference_position;
01589       /*220-223*/ public SL axis_g_motor_position;
01590       /*224-227*/ public SL axis_g_position_error;
01591       /*228-231*/ public SL axis_g_aux_position;
01592       /*232-235*/ public SL axis_g_velocity;
```

```
01593       /*236-237*/ public SW axis_g_torque;
01594       /*238-239*/ public UW  axis_g_analog_in;
01595
01596       /*240-241*/ public UW axis_h_status;
01597       /*242*/   public UB axis_h_switches;
01598       /*243*/     public UB axis_h_stop_code;
01599       /*244-247*/ public SL axis_h_reference_position;
01600       /*248-251*/ public SL axis_h_motor_position;
01601       /*252-255*/ public SL axis_h_position_error;
01602       /*256-259*/ public SL axis_h_aux_position;
01603       /*260-263*/ public SL axis_h_velocity;
01604       /*264-265*/ public SW axis_h_torque;
01605       /*266-267*/ public UW  axis_h_analog_in;
01606     }; //DataRecord2013
01607
01614     [StructLayout(LayoutKind.Sequential, Pack=1)]
01615     public struct GDataRecord1802 : GDataRecord
01616     {
01617         public byte[] byte_array() { return StructToByteArray(this); }
01618
01619         /*Offset   type name        description*/
01620
01621         /*00-01*/   public UW sample_number;
01622
01623       /*02*/      public UB input_bank_0;
01624       /*03*/      public UB input_bank_1;
01625       /*04*/      public UB input_bank_2;
01626       /*05*/      public UB input_bank_3;
01627       /*06*/      public UB input_bank_4;
01628       /*07*/      public UB input_bank_5;
01629       /*08*/      public UB input_bank_6;
01630       /*09*/    public UB input_bank_7;
01631       /*10*/    public UB input_bank_8;
01632       /*11*/      public UB input_bank_9;
01633
01634       /*12*/      public UB output_bank_0;
01635       /*13*/      public UB output_bank_1;
01636       /*14*/    public UB output_bank_2;
01637       /*15*/      public UB output_bank_3;
01638       /*16*/    public UB output_bank_4;
01639       /*17*/      public UB output_bank_5;
01640       /*18*/      public UB output_bank_6;
01641       /*19*/      public UB output_bank_7;
01642       /*20*/    public UB output_bank_8;
01643       /*21*/    public UB output_bank_9;
01644
01645       /*22*/      public UB error_code;
01646       /*23*/    public UB general_status;
01647
01648       /*24-25*/ public UW s_plane_segment_count;
01649       /*26-27*/ public UW s_plane_move_status;
01650       /*28-31*/ public SL s_distance;
01651
01652       /*32-33*/ public UW t_plane_segment_count;
01653       /*34-35*/ public UW t_plane_move_status;
01654       /*36-39*/ public SL t_distance;
01655
01656       /*40-41*/ public UW axis_a_status;
01657       /*42*/      public UB axis_a_switches;
01658       /*43*/      public UB axis_a_stop_code;
01659       /*44-47*/ public SL axis_a_reference_position;
01660       /*48-51*/ public SL axis_a_motor_position;
01661       /*52-55*/ public SL axis_a_position_error;
01662       /*56-59*/ public SL axis_a_aux_position;
01663       /*60-63*/ public SL axis_a_velocity;
01664       /*64-65*/ public SW   axis_a_torque;
01665       /*66*/      public UB axis_a_reserved_0;
01666       /*67*/      public UB axis_a_reserved_1;
01667
01668       /*68-69*/   public UW axis_b_status;
01669       /*70*/      public UB axis_b_switches;
01670       /*71*/      public UB axis_b_stop_code;
01671       /*72-75*/ public SL axis_b_reference_position;
01672       /*76-79*/ public SL axis_b_motor_position;
01673       /*80-83*/ public SL axis_b_position_error;
01674       /*84-87*/ public SL axis_b_aux_position;
01675       /*88-91*/ public SL axis_b_velocity;
01676       /*92-93*/ public SW axis_b_torque;
01677       /*94*/      public UB axis_b_reserved_0;
01678       /*95*/      public UB axis_b_reserved_1;
01679
01680       /*96-97*/ public UW axis_c_status;
01681       /*98*/    public UB axis_c_switches;
01682       /*99*/    public UB axis_c_stop_code;
01683       /*100-103*/ public SL axis_c_reference_position;
01684       /*104-107*/ public SL axis_c_motor_position;
01685       /*108-111*/ public SL axis_c_position_error;
```

```
01686       /*112-115*/ public SL axis_c_aux_position;
01687       /*116-119*/ public SL axis_c_velocity;
01688       /*120-121*/ public SW axis_c_torque;
01689       /*122*/     public UB axis_c_reserved_0;
01690       /*123*/     public UB axis_c_reserved_1;
01691
01692       /*124-125*/ public UW axis_d_status;
01693       /*126*/     public UB axis_d_switches;
01694       /*127*/   public UB axis_d_stop_code;
01695       /*128-131*/ public SL axis_d_reference_position;
01696       /*132-135*/ public SL axis_d_motor_position;
01697       /*136-139*/ public SL axis_d_position_error;
01698       /*140-143*/ public SL axis_d_aux_position;
01699       /*144-147*/ public SL axis_d_velocity;
01700       /*148-149*/ public SW axis_d_torque;
01701       /*150*/     public UB axis_d_reserved_0;
01702       /*151*/     public UB axis_d_reserved_1;
01703
01704     }; //DataRecord1802
01705
01707     [StructLayout(LayoutKind.Sequential, Pack=1)]
01708     public struct GDataRecord30000 : GDataRecord
01709     {
01710         public byte[] byte_array() { return StructToByteArray(this); }
01711
01712         /*Offset   type name        description*/
01713
01714         /*00*/      public UB header_0;
01715       /*01*/      public UB header_1;
01716       /*02*/      public UB header_2;
01717       /*03*/      public UB header_3;
01718
01719       /*04-05*/   public UW sample_number;
01720
01721       /*06*/      public UB input_bank_0;
01722       /*07*/      public UB input_bank_1;
01723
01724       /*08*/      public UB output_bank_0;
01725       /*09*/      public UB output_bank_1;
01726
01727       /*10*/      public UB error_code;
01728       /*11*/    public UB thread_status;
01729
01730       /*12-13*/ public UW input_analog_2;
01731
01732       /*14-15*/ public UW output_analog_1;
01733       /*16-17*/ public UW output_analog_2;
01734
01735       /*18-21*/ public UL amplifier_status;
01736
01737       /*22-25*/ public UL contour_segment_count;
01738       /*26-27*/ public UW contour_buffer_available;
01739
01740       /*28-29*/ public UW s_plane_segment_count;
01741       /*30-31*/ public UW s_plane_move_status;
01742       /*32-35*/ public SL s_distance;
01743       /*36-37*/ public UW s_plane_buffer_available;
01744
01745       /*38-39*/ public UW axis_a_status;
01746       /*40*/      public UB axis_a_switches;
01747       /*41*/      public UB axis_a_stop_code;
01748       /*42-45*/ public SL axis_a_reference_position;
01749       /*46-49*/ public SL axis_a_motor_position;
01750       /*50-53*/ public SL axis_a_position_error;
01751       /*54-57*/ public SL axis_a_aux_position;
01752       /*58-61*/ public SL axis_a_velocity;
01753       /*62-65*/ public SL   axis_a_torque;
01754       /*66-67*/ public UW  axis_a_analog_in;
01755       /*68*/      public UB axis_a_halls;
01756       /*69*/    public UB axis_a_reserved;
01757       /*70-73*/ public SL axis_a_variable;
01758     }; //DataRecord30000
01759
01761     [StructLayout(LayoutKind.Sequential, Pack=1)]
01762     public struct GDataRecord47000_ENC : GDataRecord
01763     {
01764         public byte[] byte_array() { return StructToByteArray(this); }
01765
01766         /*Offset   type name        description*/
01767
01768         /*00*/      public UB header_0;
01769       /*01*/      public UB header_1;
01770       /*02*/      public UB header_2;
01771       /*03*/      public UB header_3;
01772
01773       /*04-05*/   public UW sample_number;
01774       /*06*/      public UB error_code;
```

```
01775       /*07*/    public UB general_status;
01776
01777       /*08-09*/ public UW output_analog_0;
01778       /*10-11*/ public UW output_analog_1;
01779       /*12-13*/ public UW output_analog_2;
01780       /*14-15*/ public UW output_analog_3;
01781       /*16-17*/ public UW output_analog_4;
01782       /*18-19*/ public UW output_analog_5;
01783       /*20-21*/ public UW output_analog_6;
01784       /*22-23*/ public UW output_analog_7;
01785
01786       /*24-25*/ public UW input_analog_0;
01787       /*26-27*/ public UW input_analog_1;
01788       /*28-29*/ public UW input_analog_2;
01789       /*30-31*/ public UW input_analog_3;
01790       /*32-33*/ public UW input_analog_4;
01791       /*34-35*/ public UW input_analog_5;
01792       /*36-37*/ public UW input_analog_6;
01793       /*38-39*/ public UW input_analog_7;
01794
01795       /*40-41*/ public UW output_bank_0;
01796
01797       /*42-43*/ public UW input_bank_0;
01798
01799       /*44-47*/ public UL pulse_count_0;
01800       /*48-51*/ public SL zc_variable;
01801       /*52-55*/ public SL zd_variable;
01802
01803       /*56-59*/ public SL encoder_0;
01804       /*60-63*/ public SL encoder_1;
01805       /*64-67*/ public SL encoder_2;
01806       /*68-71*/ public SL encoder_3;
01807
01808   }; //GDataRecord47000_ENC
01809
01811   [StructLayout(LayoutKind.Sequential, Pack=1)]
01812   public struct GDataRecord47300_ENC : GDataRecord
01813   {
01814       public byte[] byte_array() { return StructToByteArray(this); }
01815
01816       /*Offset   type name        description*/
01817
01818       /*00*/      public UB header_0;
01819       /*01*/      public UB header_1;
01820       /*02*/      public UB header_2;
01821       /*03*/      public UB header_3;
01822
01823       /*04-05*/   public UW sample_number;
01824       /*06*/      public UB error_code;
01825       /*07*/    public UB general_status;
01826
01827       /*08-09*/ public UW output_analog_0;
01828       /*10-11*/ public UW output_analog_1;
01829       /*12-13*/ public UW output_analog_2;
01830       /*14-15*/ public UW output_analog_3;
01831       /*16-17*/ public UW output_analog_4;
01832       /*18-19*/ public UW output_analog_5;
01833       /*20-21*/ public UW output_analog_6;
01834       /*22-23*/ public UW output_analog_7;
01835
01836       /*24-25*/ public UW input_analog_0;
01837       /*26-27*/ public UW input_analog_1;
01838       /*28-29*/ public UW input_analog_2;
01839       /*30-31*/ public UW input_analog_3;
01840       /*32-33*/ public UW input_analog_4;
01841       /*34-35*/ public UW input_analog_5;
01842       /*36-37*/ public UW input_analog_6;
01843       /*38-39*/ public UW input_analog_7;
01844
01845       /*40-41*/ public UW output_bank_0;
01846       /*42-43*/ public UW output_bank_1;
01847
01848       /*44-45*/ public UW input_bank_0;
01849       /*46-47*/ public UW input_bank_1;
01850
01851       /*48-51*/ public UL pulse_count_0;
01852       /*52-55*/ public SL zc_variable;
01853       /*56-59*/ public SL zd_variable;
01854
01855       /*60-63*/ public SL encoder_0;
01856       /*64-67*/ public SL encoder_1;
01857       /*68-71*/ public SL encoder_2;
01858       /*72-75*/ public SL encoder_3;
01859
01860   }; //GDataRecord47300_ENC
01861
01863   [StructLayout(LayoutKind.Sequential, Pack=1)]
```

```
01864     public struct GDataRecord47300_24EX : GDataRecord
01865     {
01866         public byte[] byte_array() { return StructToByteArray(this); }
01867
01868         /*Offset   type name        description*/
01869
01870         /*00*/      public UB header_0;
01871       /*01*/      public UB header_1;
01872       /*02*/      public UB header_2;
01873       /*03*/      public UB header_3;
01874
01875       /*04-05*/   public UW sample_number;
01876       /*06*/      public UB error_code;
01877       /*07*/    public UB general_status;
01878
01879       /*08-09*/ public UW output_analog_0;
01880       /*10-11*/ public UW output_analog_1;
01881       /*12-13*/ public UW output_analog_2;
01882       /*14-15*/ public UW output_analog_3;
01883       /*16-17*/ public UW output_analog_4;
01884       /*18-19*/ public UW output_analog_5;
01885       /*20-21*/ public UW output_analog_6;
01886       /*22-23*/ public UW output_analog_7;
01887
01888       /*24-25*/ public UW input_analog_0;
01889       /*26-27*/ public UW input_analog_1;
01890       /*28-29*/ public UW input_analog_2;
01891       /*30-31*/ public UW input_analog_3;
01892       /*32-33*/ public UW input_analog_4;
01893       /*34-35*/ public UW input_analog_5;
01894       /*36-37*/ public UW input_analog_6;
01895       /*38-39*/ public UW input_analog_7;
01896
01897       /*40-41*/ public UW output_bank_0;
01898       /*42-43*/ public UW output_bank_1;
01899
01900       /*44-45*/ public UW input_bank_0;
01901       /*46-47*/ public UW input_bank_1;
01902
01903       /*48-51*/ public UL pulse_count_0;
01904       /*52-55*/ public SL zc_variable;
01905       /*56-59*/ public SL zd_variable;
01906
01907       /*60-61*/ public UW output_bank_2;
01908       /*62-63*/ public UW output_back_3;
01909
01910       /*64-65*/ public UW input_bank_2;
01911       /*66-67*/ public UW input_bank_3;
01912
01913     }; //GDataRecord47300_24EX
01914
01916     [StructLayout(LayoutKind.Sequential, Pack=1)]
01917     public struct GDataRecord47162 : GDataRecord
01918     {
01919         public byte[] byte_array() { return StructToByteArray(this); }
01920         /*Offset   type name        description*/
01921
01922         /*00*/      public UB header_0;
01923       /*01*/      public UB header_1;
01924       /*02*/      public UB header_2;
01925       /*03*/      public UB header_3;
01926
01927       /*04-05*/   public UW sample_number;
01928       /*06*/      public UB error_code;
01929       /*07*/    public UB general_status;
01930
01931       /*08-09*/ public UW output_analog_0;
01932       /*10-11*/ public UW output_analog_1;
01933       /*12-13*/ public UW output_analog_2;
01934       /*14-15*/ public UW output_analog_3;
01935       /*16-17*/ public UW output_analog_4;
01936       /*18-19*/ public UW output_analog_5;
01937       /*20-21*/ public UW output_analog_6;
01938       /*22-23*/ public UW output_analog_7;
01939
01940       /*24-25*/ public UW input_analog_0;
01941       /*26-27*/ public UW input_analog_1;
01942       /*28-29*/ public UW input_analog_2;
01943       /*30-31*/ public UW input_analog_3;
01944       /*32-33*/ public UW input_analog_4;
01945       /*34-35*/ public UW input_analog_5;
01946       /*36-37*/ public UW input_analog_6;
01947       /*38-39*/ public UW input_analog_7;
01948
01949       /*40*/      public UB output_byte_0;
01950       /*41*/      public UB output_byte_1;
01951       /*42*/      public UB output_byte_2;
```

```
01952
01953       /*43*/      public UB input_byte_0;
01954       /*44*/      public UB input_byte_1;
01955       /*45*/      public UB input_byte_2;
01956       /*46*/      public UB input_byte_3;
01957       /*47*/      public UB input_byte_4;
01958
01959       /*48-51*/ public UL pulse_count_0;
01960       /*52-55*/ public SL zc_variable;
01961       /*56-59*/ public SL zd_variable;
01962
01963       /*60-63*/ public SL encoder_0;
01964       /*64-67*/ public SL encoder_1;
01965       /*68-71*/ public SL encoder_2;
01966       /*72-75*/ public SL encoder_3;
01967
01968     }; //GDataRecord47162
01969
01970     #endregion
01971 } //gclib class
01972
```

10.185 Galil.h File Reference

```
#include <string>
#include <vector>
```

Data Structures

- class Galil

10.185.1 Macro Definition Documentation

10.185.1.1 DLL_IMPORT_EXPORT

```
#define DLL_IMPORT_EXPORT
```

Definition at line 20 of file Galil.h.

10.186 Galil.h

Go to the documentation of this file.

```
00001
00004 #ifndef GALIL_H
00005 #define GALIL_H
00006
00007 #include <string>
00008 #include <vector>
00009
00010 class GalilPrivate;
00011
00012 #ifdef _MSC_VER // MSVC Compiler
00013    #ifdef  MAKEDLL
00014       #define DLL_IMPORT_EXPORT  __declspec(dllexport)
00015    #else
00016 //      #define DLL_IMPORT_EXPORT  __declspec(dllimport)
00017   #define DLL_IMPORT_EXPORT
00018    #endif
00019 #else //not Windows (e.g. Linux).  Expand to empty space
00020    #define DLL_IMPORT_EXPORT
00021 #endif
00022
00023 //DLL_IMPORT_EXPORT void SetDynamicLink (void);
00024
00025 class DLL_IMPORT_EXPORT Galil //An instance of the Galil class (Galil object) represents a CONNECTION
      to a controller (not necessarily the controller itself).  Multiple Galil objects can connect to a
      single Ethernet controller.
00026 {
00027   GalilPrivate * d;
00028 public:
00029    static std::string             libraryVersion();                      //returns version string
      of Galil class library for display (e.g. "0.0.4.3 Jan 2 2008 16:04:50 libGalil.so").  Can be called
      WITHOUT an instance of Galil.  Note that this has nothing to do with the version of the controller
      (which can be found with connection() below).
```

```
00030    static std::vector<std::string> addresses();                              //returns list of
available addresses to connect to (e.g. "1.2.3.4").  Each item in the list may be fed to the
constructor Galil()
00031
00032                                    Galil(std::string address = "");  //constructor opens connection
with controller e.g. 192.168.1.2, COM1, /dev/ttyS0, GALILPCI1 /dev/galilpci0 (default constructor with
no arguments will bring up a dialog)
00033                                   ~Galil();                                  //destructor closes
connection with controller
00034    std::string                    connection();                              //returns a string like
"DMC4080 Rev 1.0, 123, 10.0.0.70, IHA"
00035
00036    int                            timeout_ms; //default = 500 milliseconds.  This is the timeout for
everything but BP, BV, RS, ^R^S, and program/array/firmware download
00037    std::string                    command(     const std::string& command = "MG TIME", const
std::string& terminator = "\r", const std::string& ack = ":", bool trim = true); //send a command
(e.g. "MG _RPX") to the controller and get the response
00038    double                         commandValue(const std::string& command = "MG TIME"); //convenience
method that converts response from string to numerical value
00039    std::string                    message(  int timeout_ms = 500); //ms.  get MGs from controller
program
00040    int                            interrupt(int timeout_ms = 500); //ms.  EI, UI (DMC-18xx only).
Returns status byte (e.g. 0xf0 for UI0)
00041
00042    std::string                    programUpload();
//UL upload   a controller program to   an in-memory buffer
00043    void                           programDownload(    const std::string& program = "MG TIME\rEN");
//DL download a controller program from an in-memory buffer
00044    void                           programUploadFile(  const std::string& file    = "program.dmc");
//UL upload   a controller program to   a disk file
00045    void                           programDownloadFile(const std::string& file    = "program.dmc");
//DL download a controller program from a disk file
00046
00047    std::vector<double>            arrayUpload(                                                  const
std::string& name  = "array"); //QU upload   an array to   an in-memory buffer
00048    void                           arrayDownload(    const std::vector<double>& array,       const
std::string& name  = "array"); //QD download an array from an in-memory buffer
00049    void                           arrayUploadFile(  const std::string& file = "arrays.csv", const
std::string& names = "");      //QU upload   array(s) to   a disk file.  "" means upload all arrays,
else separate the array names with a space
00050    void                           arrayDownloadFile(const std::string& file = "arrays.csv");
//QD download array(s) from a disk file
00051
00052    void                           firmwareDownloadFile(const std::string& file = "firmware.hex");
//download hex file (RS-232 only for DMC-21x3)
00053
00054    int                            write(const std::string& bytes = "\r");  //returns actual number of
bytes written
00055    std::string                    read();                                    //returns actual bytes
read
00056
00057    std::vector<std::string>       sources();                          //returns list of sources
(_RPA...) supported by this controller, which are fed to sourceValue(), source(), and setSource()
00058    void                           recordsStart(double period_ms = -1); //milliseconds.  Sends DR.  -1
means leave be if not 0, else run as fast as possible.  > 0 sets the DR sample period in true
milliseconds (rounded to nearest power of 2 for DMC-18xx).  0 sets DR0 (turns it off)
00059    std::vector<char>              record(const std::string& method = "QR"); //reads DR packet OR
sends QR and reads response
00060    double                         sourceValue(const std::vector<char>& record,          const
std::string& source = "TIME"); //get the value for one particular source (e.g. _RPA is 1000)
00061    std::string                    source(     const std::string& field = "Description", const
std::string& source = "TIME"); //get e.g. the description string for one particular source (e.g. _RPA
is "Axis A reference position").  Fields are "Description", "Units", & "Scale"
00062    void                           setSource(  const std::string& field = "Description", const
std::string& source = "TIME", const std::string& to = "Sample counter"); //set e.g. the description
string for one particular source (e.g. _RPA to "Feed axis reference position").  Fields are
"Description", "Units", & "Scale"
00063
00064 };
00065
00066 #endif
```

10.187 gcl_datarecord.cpp File Reference

```
#include "gcl_galil.h"
```

10.188 gcl_datarecord.cpp

Go to the documentation of this file.

```
00001
00004 #include "gcl_galil.h"
00005 using namespace std;
00006
00007 vector<std::string> Galil::sources()
00008 {
00009   vector<std::string> s;
00010   for (auto it = d->map.begin(); it != d->map.end(); ++it)
00011     s.push_back(it->first);
00012
00013   std::sort(s.begin(), s.end());
00014
00015   return s;
00016 }
00017
00018 void Galil::recordsStart(double period_ms)
00019 {
00020   ec(GRecordRate(d->g, period_ms));
00021 }
00022
00023 vector<char> Galil::record(const std::string& method)
00024 {
00025   GDataRecord record;
00026   ec(GRecord(d->g, &record, method == "QR" ? G_QR : G_DR));
00027   vector <char> record_vector;
00028   for (int i = 0; i < sizeof(GDataRecord); i++)
00029     record_vector.push_back(record.byte_array[i]);
00030
00031   return record_vector;
00032 }
00033
00034 double Galil::sourceValue(const std::vector<char>& record, const std::string& source)
00035 {
00036
00037   try
00038   {
00039     const Source& s = d->map.at(source); //use at() function so silent insert does not occur if bad
      source string is used.
00040     int return_value = 0;
00041     if (s.type[0] == 'U')  //unsigned
00042       switch (s.type[1])
00043     {
00044       case 'B':  return_value = *(unsigned char*)(&record[0] + s.byte);  break;
00045       case 'W':  return_value = *(unsigned short*)(&record[0] + s.byte);  break;
00046       case 'L':  return_value = *(unsigned int*)(&record[0] + s.byte);  break;
00047     }
00048     else //s.type[0] == 'S'  //signed
00049       switch (s.type[1])
00050     {
00051       case 'B':  return_value = *(char*)(&record[0] + s.byte);  break;
00052       case 'W':  return_value = *(short*)(&record[0] + s.byte);  break;
00053       case 'L':  return_value = *(int*)(&record[0] + s.byte);  break;
00054     }
00055
00056     if (s.bit >= 0) //this is a bit field
00057     {
00058       bool bTRUE = s.scale > 0; //invert logic if scale is <= 0
00059       return return_value & (1 << s.bit) ? bTRUE : !bTRUE; //check the bit
00060     }
00061     else
00062       return (return_value / s.scale) + s.offset;
00063
00064   }
00065   catch (const std::out_of_range& e) //bad source
00066   {
00067     (void)e; //unused
00068     return 0.0;
00069   }
00070
00071 }
00072
00073 string Galil::source(const std::string& field, const std::string& source)
00074 {
00075   try
00076   {
00077     const Source& s = d->map.at(source); //use at() function so silent insert does not occur if bad
      source string is used.
00078     if (field == "Description")
00079       return s.description;
00080
00081     if (field == "Units")
00082       return s.units;
00083
00084     if (field == "Scale")
00085       return to_string(s.scale);
00086
00087     if (field == "Offset")
```

```
00088       return to_string(s.offset);
00089
00090     return ""; //no matches
00091   }
00092   catch (const std::out_of_range& e) //bad source
00093   {
00094     (void)e;//unused
00095     return ""; //no matches
00096   }
00097
00098 }
00099
00100 void Galil::setSource(const std::string& field, const std::string& source, const std::string& to)
00101 {
00102   try
00103   {
00104     Source& s = d->map.at(source); //use at() function so silent insert does not occur if bad source
      string is used.
00105     if (field == "Description")
00106     {
00107       s.description = to;
00108       return;
00109     }
00110
00111     if (field == "Units")
00112     {
00113       s.units = to;
00114       return;
00115     }
00116
00117     if (field == "Scale")
00118     {
00119       s.scale = stod(to);
00120       return;
00121     }
00122
00123     if (field == "Offset")
00124     {
00125       s.offset = stod(to);
00126       return;
00127     }
00128   }
00129   catch (const std::out_of_range& e) //bad source
00130   {
00131     (void)e;//unused
00132     return;
00133   }
00134 }
00135
00136
00137
00138 void GalilPrivate::InitializeDataRecord()
00139 {
00140   map.clear(); //clear the map if thre is anything in it
00141
00142   //infer data record structure from QZ
00143   string qz = q->command("QZ");
00144   vector<int> qz_split;
00145   size_t start = 0;
00146   for (size_t i = 0; i < qz.length(); i++) //ad hoc split
00147   {
00148     if (qz[i] == ',')
00149     {
00150       qz_split.push_back(stoi(qz.substr(start, i - start)));
00151       start = i + 1; //jump over delim
00152       continue;
00153     }
00154
00155     if (i == qz.length() - 1) //last token
00156     {
00157       qz_split.push_back(stoi(qz.substr(start)));
00158       break;
00159     }
00160   }
00161
00162   if (4 != qz_split.size()) return; //parsing error or bad response
00163
00164   int axes = qz_split[0];
00165
00166
00167   //weed out PCI cards
00168   string rv = q->command("\x12\x16"); //^R^V
00169   if (rv.find("DMC18") != string::npos) //PCI-based card
00170   {
00171     if (rv.length() >= 7)
00172     {
00173       if (rv[6] == '6') return Init1806(axes);
```

```
00174       if (rv[6] == '0') return Init1800(axes, false);
00175       if (rv[6] == '2') return Init1800(axes, true);
00176     }
00177   }
00178
00179   int gen_status = qz_split[1];
00180   if (gen_status == 18) return Init30010(rv.find("DMC31") != string::npos); // DMC300x0 returns 1 18
      16 36, search for "DMC31" in ^R^V to determine 16bit ADC
00181
00182   int  axis_block = qz_split[3]; //size of the axis block data in data record
00183   if (axis_block == 36) return Init4000(axes);  //DMC40x0/DMC41x3/DMC50000         8 52 26 36
00184   if (axis_block == 28) return Init2103(axes);     //DMC14x5/2xxx/                 8 24 16 28 //also
      Optima
00185
00186   //if here, should be an RIO
00187   if (axis_block != 0) return; //RIO has a 0 in the axis block data
00188
00189   //Determine the RIO type
00190   int io_block = qz_split[2];
00191
00192   //RIO-47300 has 4 extra bytes in the I/O block
00193   bool rio3 = ((io_block == 52) || (io_block == 60) || (io_block == 68)); // RIO-47300 Standard, with
      Exteneded I/O, with Quad/Biss/SSi
00194
00195   //SER tacks 4 longs on the end of the data record (4 encoders)
00196   bool rioser = ((io_block == 64) || (io_block == 68)); // 471x2,472x2 OR 47300 with SER
00197
00198   //Extended I/O tacks 8 bytes on the end of the data rrecord, three bytes of each of I/O, one padding
      for each
00199   bool rio3ex = (io_block == 60); // RIO-47300 with extended I/O. Mutually exclusive with SER
00200
00201
00202   InitRio(rio3);
00203
00204   if (rio3ex) InitRio3_24Ex();
00205   if (rioser) InitRioSer(rio3);
00206
00207 }
00208
00209 string GalilPrivate::ax(string prefix, int axis, string suffix)
00210 {
00211   return prefix + (char)('A' + axis) + suffix;
00212 }
00213
00214 void GalilPrivate::input_bits(int byte, int num)
00215 {
00216   stringstream ss;
00217
00218   for (int i = 0; i < 8; i++)
00219   {
00220     ss « "@IN[";
00221     ss « setw(2) « setfill('0') « right « num;
00222     ss « "]";
00223     map[ss.str()] = Source(byte, "UB", i, "Boolean", "Digital input " + to_string(num));
00224     ss.str("");
00225     num++;
00226   }
00227 }
00228
00229 void GalilPrivate::output_bits(int byte, int num)
00230 {
00231   stringstream ss;
00232
00233   for (int i = 0; i < 8; i++)
00234   {
00235     ss « "@OUT[";
00236     ss « setw(2) « setfill('0') « right « num;
00237     ss « "]";
00238     map[ss.str()] = Source(byte, "UB", i, "Boolean", "Digital output " + to_string(num));
00239     ss.str("");
00240     num++;
00241   }
00242 }
00243
00244 void GalilPrivate::aq_analog(int byte, int input_num)
00245 {
00246   //When analog voltage decoding depends upon AQ setting.
00247   string type; //for interpreting analog as signed/unsigned
00248   double divisor; //for dividing ADC counts to calc volts
00249   int val;
00250   string command = "MG{Z10.0}_AQ" + to_string(input_num);
00251   if (GCmdI(g, command.c_str(), &val) == G_NO_ERROR) //don't add analog if error on AQ
00252   {
00253     switch (val)
00254     {
00255     case 1: case -1:  divisor = 32768.0 / 5;   type = "SW";  break;   //  -5 to 5  V   -32768 to 32767
00256     case 3: case -3:  divisor = 65536.0 / 5;   type = "UW";  break;   //   0 to 5  V        0 to 65535
```

```
00257     case 4: case -4:  divisor = 65536.0 / 10;  type = "UW";  break;   //   0 to 10 V        0 to 65535
00258     case 2: case -2:  default: //AQ 2 is the default value
00259       divisor = 32768.0 / 10;  type = "SW";  break;   // -10 to 10 V   -32768 to 32767
00260     }
00261     map["@AN[" + to_string(input_num + 1) + "]"] = Source(byte, type, -1, "V", "Analog input " +
      to_string(input_num), divisor);
00262   }
00263 }
00264
00265 void GalilPrivate::dq_analog(int byte, int input_num)
00266 {
00267   //When analog voltage decoding depends upon DQ setting.
00268   string type; //for interpreting analog as signed/unsigned
00269   double divisor; //for dividing ADC counts to calc volts
00270   int val;
00271   string command = "MG{Z10.0}_DQ" + to_string(input_num);
00272   if (GCmdI(g, command.c_str(), &val) == G_NO_ERROR) //don't add analog if error on AQ
00273   {
00274     switch (val)
00275     {
00276     case 3: divisor = 32768.0 / 5;   type = "SW";  break;   //  -5 to 5  V   -32768 to 32767
00277     case 1: divisor = 65536.0 / 5;   type = "UW";  break;   //   0 to 5  V        0 to 65535
00278     case 2: divisor = 65536.0 / 10;  type = "UW";  break;   //   0 to 10 V        0 to 65535
00279     case 4: default: //DQ 4 is the default value
00280       divisor = 32768.0 / 10;  type = "SW";  break;   // -10 to 10 V   -32768 to 32767
00281     }
00282     map["@AO[" + to_string(input_num + 1) + "]"] = Source(byte, type, -1, "V", "Analog output " +
      to_string(input_num), divisor);
00283   }
00284 }
00285
00286 void GalilPrivate::Init4000(int axes)
00287 {
00288   //0-3 Header is ignored in GCL
00289
00290   map["TIME"] = Source(4, "UW", -1, "samples", "Sample counter");
00291
00292   //Digital Inputs
00293   map["_TI0"] = Source(6, "UB", -1, "", "Digital inputs 1 to 8");
00294   input_bits(6, 1);
00295
00296   map["_TI1"] = Source(7, "UB", -1, "", "Digital inputs 9 to 16"); //TI always included
00297   if (axes > 4) //9-16 depend on axes 5-8
00298     input_bits(7, 9);
00299
00300   //Digital outputs
00301   map["_OP0"] = Source(16, "UW", -1, "", "Digital outputs 1 to 16");
00302   output_bits(16, 1);
00303
00304   if (axes > 4) //9-16 depend on axes 5-8
00305     output_bits(17, 9);
00306
00307   //Extended I/O
00308   int co = -1;
00309   if (GCmdI(g, "MG_CO", &co) == G_NO_ERROR) //41x3 will ? here
00310   {
00311     map["_TI2"] = Source(8, "UB", -1, "", "Digital inputs 17 to 24"); //TI always included in gcl
00312     map["_TI3"] = Source(9, "UB", -1, "", "Digital inputs 25 to 32");
00313     map["_TI4"] = Source(10, "UB", -1, "", "Digital inputs 33 to 40");
00314     map["_TI5"] = Source(11, "UB", -1, "", "Digital inputs 41 to 48");
00315
00316     map["_OP1"] = Source(18, "UW", -1, "", "Digital outputs 17 to 32"); //OP always included in gcl
00317     map["_OP2"] = Source(20, "UW", -1, "", "Digital outputs 33 to 48");
00318
00319     if (co & 0x00000001) //bank 2 is output
00320       output_bits(18, 17);
00321     else //bank 2 in input
00322       input_bits(8, 17);
00323
00324     if (co & 0x00000002) //bank 3 is output
00325       output_bits(19, 25);
00326     else //bank 3 is input
00327       input_bits(9, 25);
00328
00329     if (co & 0x00000004) //bank 4 is output
00330       output_bits(20, 33);
00331     else //bank 4 is input
00332       input_bits(10, 33);
00333
00334     if (co & 0x00000008) //bank 5 is output
00335       output_bits(21, 41);
00336     else
00337       input_bits(11, 41);
00338   }
00339
00340   //Ethernet Handle Status
00341   map["_IHA2"] = Source(42, "UB", -1, "", "Handle A Ethernet status");
```

```
00342   map["_IHB2"] = Source(43, "UB", -1, "", "Handle B Ethernet status");
00343   map["_IHC2"] = Source(44, "UB", -1, "", "Handle C Ethernet status");
00344   map["_IHD2"] = Source(45, "UB", -1, "", "Handle D Ethernet status");
00345   map["_IHE2"] = Source(46, "UB", -1, "", "Handle E Ethernet status");
00346   map["_IHF2"] = Source(47, "UB", -1, "", "Handle F Ethernet status");
00347   map["_IHG2"] = Source(48, "UB", -1, "", "Handle G Ethernet status");
00348   map["_IHH2"] = Source(49, "UB", -1, "", "Handle H Ethernet status");
00349
00350   map["_TC"] = Source(50, "UB", -1, "", "Error code");
00351
00352   //Thread status
00353   map["NO0"] = Source(51, "UB", 0, "Boolean", "Thread 0 running");
00354   map["NO1"] = Source(51, "UB", 1, "Boolean", "Thread 1 running");
00355   map["NO2"] = Source(51, "UB", 2, "Boolean", "Thread 2 running");
00356   map["NO3"] = Source(51, "UB", 3, "Boolean", "Thread 3 running");
00357   map["NO4"] = Source(51, "UB", 4, "Boolean", "Thread 4 running");
00358   map["NO5"] = Source(51, "UB", 5, "Boolean", "Thread 5 running");
00359   map["NO6"] = Source(51, "UB", 6, "Boolean", "Thread 6 running");
00360   map["NO7"] = Source(51, "UB", 7, "Boolean", "Thread 7 running");
00361
00362   //Amplifier error status
00363   map["TA00"] = Source(52, "UB", 0, "Boolean", "Axis A-D over current");
00364   map["TA01"] = Source(52, "UB", 1, "Boolean", "Axis A-D over voltage");
00365   map["TA02"] = Source(52, "UB", 2, "Boolean", "Axis A-D over temperature");
00366   map["TA03"] = Source(52, "UB", 3, "Boolean", "Axis A-D under voltage");
00367   map["TA04"] = Source(52, "UB", 4, "Boolean", "Axis E-H over current");
00368   map["TA05"] = Source(52, "UB", 5, "Boolean", "Axis E-H over voltage");
00369   map["TA06"] = Source(52, "UB", 6, "Boolean", "Axis E-H over temperature");
00370   map["TA07"] = Source(52, "UB", 7, "Boolean", "Axis E-H under voltage");
00371
00372   map["TA1A"] = Source(53, "UB", 0, "Boolean", "Axis A hall error");
00373   map["TA1B"] = Source(53, "UB", 1, "Boolean", "Axis B hall error");
00374   map["TA1C"] = Source(53, "UB", 2, "Boolean", "Axis C hall error");
00375   map["TA1D"] = Source(53, "UB", 3, "Boolean", "Axis D hall error");
00376   map["TA1E"] = Source(53, "UB", 4, "Boolean", "Axis E hall error");
00377   map["TA1F"] = Source(53, "UB", 5, "Boolean", "Axis F hall error");
00378   map["TA1G"] = Source(53, "UB", 6, "Boolean", "Axis G hall error");
00379   map["TA1H"] = Source(53, "UB", 7, "Boolean", "Axis H hall error");
00380
00381   map["TA2A"] = Source(54, "UB", 0, "Boolean", "Axis A at _TKA peak current");
00382   map["TA2B"] = Source(54, "UB", 1, "Boolean", "Axis B at _TKB peak current");
00383   map["TA2C"] = Source(54, "UB", 2, "Boolean", "Axis C at _TVC peak current");
00384   map["TA2D"] = Source(54, "UB", 3, "Boolean", "Axis D at _TKD peak current");
00385   map["TA2E"] = Source(54, "UB", 4, "Boolean", "Axis E at _TKE peak current");
00386   map["TA2F"] = Source(54, "UB", 5, "Boolean", "Axis F at _TKF peak current");
00387   map["TA2G"] = Source(54, "UB", 6, "Boolean", "Axis G at _TKG peak current");
00388   map["TA2H"] = Source(54, "UB", 7, "Boolean", "Axis H at _TKH peak current");
00389
00390   map["TA3AD"] = Source(55, "UB", 0, "Boolean", "Axis A-D ELO active");
00391   map["TA3EH"] = Source(55, "UB", 1, "Boolean", "Axis E-H ELO active");
00392
00393   //contour mode
00394   map["CD"] = Source(56, "UL", -1, "segments", "Contour segment count");
00395   map["_CM"] = Source(60, "UW", -1, "elements", "Contour buffer space");
00396
00397   //S plane
00398   map["_CSS"] = Source(62, "UW", -1, "segments", "Axis S segment count");
00399   map["VDS"] = Source(64, "UB", 3, "Boolean", "Axis S final deceleration");
00400   map["STS"] = Source(64, "UB", 4, "Boolean", "Axis S stopping");
00401   map["VSS"] = Source(64, "UB", 5, "Boolean", "Axis S slewing");
00402   map["_BGS"] = Source(65, "UB", 7, "Boolean", "Axis S moving");
00403   map["_AVS"] = Source(66, "SL", -1, "counts", "Axis S length");
00404   map["_LMS"] = Source(70, "UW", -1, "elements", "Axis S buffer speace");
00405
00406   //T plane
00407   map["_CST"] = Source(72, "UW", -1, "segments", "Axis T segment count");
00408   map["VDT"] = Source(74, "UB", 3, "Boolean", "Axis T final deceleration");
00409   map["STT"] = Source(74, "UB", 4, "Boolean", "Axis T stopping");
00410   map["VST"] = Source(74, "UB", 5, "Boolean", "Axis T slewing");
00411   map["_BGT"] = Source(75, "UB", 7, "Boolean", "Axis T moving");
00412   map["_AVT"] = Source(76, "SL", -1, "counts", "Axis T length");
00413   map["_LMT"] = Source(80, "UW", -1, "elements", "Axis T buffer speace");
00414
00415   //per-axis data
00416   int base = 82; //start of A axis data
00417   for (int i = 0; i < axes; i++)
00418   {
00419     map[ax("_MO", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " motor off"));
00420     map[ax("HM", i, "3")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " found index"));
00421     map[ax("_AL", i, "")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " latch armed"));
00422     map[ax("DC", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " began deceleration"));
00423     map[ax("ST", i, "")] = Source(base, "UW", 4, "Boolean", ax("Axis ", i, " began stop"));
00424     map[ax("SP", i, "")] = Source(base, "UW", 5, "Boolean", ax("Axis ", i, " began slew"));
00425     map[ax("CM", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " in contour mode"));
00426     map[ax("JG", i, "-")] = Source(base, "UW", 7, "Boolean", ax("Axis ", i, " negative move"));
00427     ++base; //83
00428     map[ax("VM", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " in VM or LI mode"));
```

```
00429     map[ax("HM", i, "2")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " finding index"));
00430     map[ax("HM", i, "1")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " coming off home
      switch"));
00431     map[ax("HM", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " home command issued"));
00432     map[ax("FE", i, "")] = Source(base, "UW", 4, "Boolean", ax("Axis ", i, " find edge issued"));
00433     map[ax("PA", i, "")] = Source(base, "UW", 5, "Boolean", ax("Axis ", i, " in PA mode"));
00434     map[ax("PR", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " in PA or PR mode"));
00435     map[ax("_BG", i, "")] = Source(base, "UW", 7, "Boolean", ax("Axis ", i, " move in progress"));
00436     ++base; //84
00437     map[ax("MT", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " in stepper mode"));
00438     map[ax("_HM", i, "")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " home switch"));
00439     map[ax("_LR", i, "")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " reverse limit switch"));
00440     map[ax("_LF", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " forward limit switch"));
00441     //4 and 5 reserved
00442     map[ax("AL", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " latch digital input"));
00443     map[ax("_AL", i, "=0")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " latch occurred"));
00444     ++base; //85
00445     map[ax("_SC", i, "")] = Source(base, "UB", -1, "", ax("Axis ", i, " stop code"));
00446     ++base; //86
00447     map[ax("_RP", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " reference position"));
00448     base += 4; //90
00449     map[ax("_TP", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " encoder position"));
00450     base += 4; //94
00451     map[ax("_TE", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " position error"));
00452     base += 4; //98
00453     map[ax("_TD", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " dual encoder
      position"));
00454     base += 4; //102
00455     map[ax("_TV", i, "")] = Source(base, "SL", -1, "counts/s", ax("Axis ", i, " filtered velocity"),
      64);
00456     base += 4; //106
00457     map[ax("_TT", i, "")] = Source(base, "SL", -1, "V", ax("Axis ", i, " torque (DAC)"), 3255);
00458     base += 4; //110
00459
00460     //Analog voltage decoding depends upon AQ setting.
00461     aq_analog(base, i + 1);
00462     base += 2; //112
00463
00464     map[ax("_QH", i, "")] = Source(base, "UB", -1, "", ax("Axis ", i, " hall sensors"));
00465     base += 1; //113 reserved
00466     base += 1; //114
00467     map[ax("_ZA", i, "")] = Source(base, "SL", -1, "", ax("Axis ", i, " user variable"));
00468     base += 4; //118
00469
00470   }// for, axis data
00471 }
00472
00473 void GalilPrivate::Init1806(int axes)
00474 {
00475   map["TIME"] = Source(0, "UW", -1, "samples", "Sample counter");
00476
00477   //Digital inputs
00478   map["_TI0"] = Source(2, "UB", -1, "", "Digital inputs 1 to 8");
00479   input_bits(2, 1);
00480
00481   map["_TI1"] = Source(3, "UB", -1, "", "Digital inputs 9 to 16"); //TI always included
00482   if (axes > 4) //9-16 depend on axes 5-8
00483     input_bits(3, 9);
00484
00485
00486   //Digital outputs
00487   map["_OP0"] = Source(12, "UW", -1, "", "Digital outputs 1 to 16");
00488   output_bits(12, 1);
00489
00490   if (axes > 4) //9-16 depend on axes 5-8
00491     output_bits(13, 9);
00492
00493
00494   //Extended I/O
00495   int co = -1;
00496   if (GCmdI(g, "MG_CO", &co) == G_NO_ERROR)
00497   {
00498     map["_TI2"] = Source(4, "UB", -1, "", "Digital inputs 17 to 24"); //TI always included in gcl
00499     map["_TI3"] = Source(5, "UB", -1, "", "Digital inputs 25 to 32");
00500     map["_TI4"] = Source(6, "UB", -1, "", "Digital inputs 33 to 40");
00501     map["_TI5"] = Source(7, "UB", -1, "", "Digital inputs 41 to 48");
00502     map["_TI6"] = Source(8, "UB", -1, "", "Digital inputs 49 to 56");
00503     map["_TI7"] = Source(9, "UB", -1, "", "Digital inputs 57 to 64");
00504     map["_TI8"] = Source(10, "UB", -1, "", "Digital inputs 65 to 72");
00505     map["_TI9"] = Source(11, "UB", -1, "", "Digital inputs 73 to 80");
00506
00507     map["_OP1"] = Source(14, "UW", -1, "", "Digital outputs 17 to 32"); //OP always included in gcl
00508     map["_OP2"] = Source(16, "UW", -1, "", "Digital outputs 33 to 48");
00509     map["_OP3"] = Source(18, "UW", -1, "", "Digital outputs 49 to 64");
00510     map["_OP4"] = Source(20, "UW", -1, "", "Digital outputs 65 to 80");
00511
00512     if (co & 0x00000001) //bank 2 is output
```

```
00513       output_bits(14, 17);
00514     else //input
00515       input_bits(4, 17);
00516
00517     if (co & 0x00000002) //bank 3 is output
00518       output_bits(15, 25);
00519     else //input
00520       input_bits(5, 25);
00521
00522     if (co & 0x00000004) //bank 4 is output
00523       output_bits(16, 33);
00524     else //input
00525       input_bits(6, 33);
00526
00527     if (co & 0x00000008) //bank 5 is output
00528       output_bits(17, 41);
00529     else //input
00530       input_bits(7, 41);
00531
00532     if (co & 0x00000010) //bank 6 is output
00533       output_bits(18, 49);
00534     else //input
00535       input_bits(8, 49);
00536
00537     if (co & 0x00000020) //bank 7 is output
00538       output_bits(19, 57);
00539     else //input
00540       input_bits(9, 57);
00541
00542     if (co & 0x00000040) //bank 8 is output
00543       output_bits(20, 65);
00544     else //input
00545       input_bits(10, 65);
00546
00547     if (co & 0x00000080) //bank 9 is output
00548       output_bits(21, 73);
00549     else //input
00550       input_bits(11, 73);
00551   } //extended io
00552
00553   map["_TC"] = Source(46, "UB", -1, "", "Error code");
00554
00555   //Thread status
00556   map["NO0"] = Source(47, "UB", 0, "Boolean", "Thread 0 running");
00557   map["NO1"] = Source(47, "UB", 1, "Boolean", "Thread 1 running");
00558   map["NO2"] = Source(47, "UB", 2, "Boolean", "Thread 2 running");
00559   map["NO3"] = Source(47, "UB", 3, "Boolean", "Thread 3 running");
00560   map["NO4"] = Source(47, "UB", 4, "Boolean", "Thread 4 running");
00561   map["NO5"] = Source(47, "UB", 5, "Boolean", "Thread 5 running");
00562   map["NO6"] = Source(47, "UB", 6, "Boolean", "Thread 6 running");
00563   map["NO7"] = Source(47, "UB", 7, "Boolean", "Thread 7 running");
00564
00565   //contour mode
00566   map["CD"] = Source(52, "UL", -1, "segments", "Contour segment count");
00567   map["_CM"] = Source(56, "UW", -1, "elements", "Contour buffer space");
00568
00569   //S plane
00570   map["_CSS"] = Source(58, "UW", -1, "segments", "Axis S segment count");
00571   map["VDS"] = Source(60, "UB", 3, "Boolean", "Axis S final deceleration");
00572   map["STS"] = Source(60, "UB", 4, "Boolean", "Axis S stopping");
00573   map["VSS"] = Source(60, "UB", 5, "Boolean", "Axis S slewing");
00574   map["_BGS"] = Source(61, "UB", 7, "Boolean", "Axis S moving");
00575   map["_AVS"] = Source(62, "SL", -1, "counts", "Axis S length");
00576   map["_LMS"] = Source(66, "UW", -1, "elements", "Axis S buffer speace");
00577
00578   //T plane
00579   map["_CST"] = Source(68, "UW", -1, "segments", "Axis T segment count");
00580   map["VDT"] = Source(70, "UB", 3, "Boolean", "Axis T final deceleration");
00581   map["STT"] = Source(70, "UB", 4, "Boolean", "Axis T stopping");
00582   map["VST"] = Source(70, "UB", 5, "Boolean", "Axis T slewing");
00583   map["_BGT"] = Source(71, "UB", 7, "Boolean", "Axis T moving");
00584   map["_AVT"] = Source(72, "SL", -1, "counts", "Axis T length");
00585   map["_LMT"] = Source(76, "UW", -1, "elements", "Axis T buffer speace");
00586
00587   //per-axis data
00588   int base = 78; //start of A axis data
00589   for (int i = 0; i < axes; i++)
00590   {
00591     map[ax("_MO", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " motor off"));
00592     map[ax("HM", i, "3")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " found index"));
00593     map[ax("_AL", i, "")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " latch armed"));
00594     map[ax("DC", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " began deceleration"));
00595     map[ax("ST", i, "")] = Source(base, "UW", 4, "Boolean", ax("Axis ", i, " began stop"));
00596     map[ax("SP", i, "")] = Source(base, "UW", 5, "Boolean", ax("Axis ", i, " began slew"));
00597     map[ax("CM", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " in contour mode"));
00598     map[ax("JG", i, "-")] = Source(base, "UW", 7, "Boolean", ax("Axis ", i, " negative move"));
00599     ++base; //79
```

```
00600     map[ax("VM", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " in VM or LI mode"));
00601     map[ax("HM", i, "2")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " finding index"));
00602     map[ax("HM", i, "1")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " coming off home
      switch"));
00603     map[ax("HM", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " home command issued"));
00604     map[ax("FE", i, "")] = Source(base, "UW", 4, "Boolean", ax("Axis ", i, " find edge issued"));
00605     map[ax("PA", i, "")] = Source(base, "UW", 5, "Boolean", ax("Axis ", i, " in PA mode"));
00606     map[ax("PR", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " in PA or PR mode"));
00607     map[ax("_BG", i, "")] = Source(base, "UW", 7, "Boolean", ax("Axis ", i, " move in progress"));
00608     ++base; //80
00609     map[ax("MT", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " in stepper mode"));
00610     map[ax("_HM", i, "")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " home switch"));
00611     map[ax("_LR", i, "")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " reverse limit switch"));
00612     map[ax("_LF", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " forward limit switch"));
00613     //4 and 5 reserved
00614     map[ax("AL", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " latch digital input"));
00615     map[ax("_AL", i, "=0")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " latch occurred"));
00616     ++base; //81
00617     map[ax("_SC", i, "")] = Source(base, "UB", -1, "", ax("Axis ", i, " stop code"));
00618     ++base; //82
00619     map[ax("_RP", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " reference position"));
00620     base += 4; //86
00621     map[ax("_TP", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " encoder position"));
00622     base += 4; //90
00623     map[ax("_TE", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " position error"));
00624     base += 4; //94
00625     map[ax("_TD", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " dual encoder
      position"));
00626     base += 4; //98
00627     map[ax("_TV", i, "")] = Source(base, "SL", -1, "counts/s", ax("Axis ", i, " filtered velocity"),
      64);
00628     base += 4; //102
00629     map[ax("_TT", i, "")] = Source(base, "SL", -1, "V", ax("Axis ", i, " torque (DAC)"), 3255);
00630     base += 4; //106
00631
00632     map["@AN[" + to_string(i + 1) + "]"] = Source(base, "SW", -1, "V", "Analog input " + to_string(i +
      1), 3276.8);
00633
00634     base += 2; //108 reserved
00635     base += 1; //109 reserved
00636     base += 1; //110
00637     map[ax("_ZA", i, "")] = Source(base, "SL", -1, "", ax("Axis ", i, " user variable"));
00638     base += 4; //114
00639   }// for, axis data
00640 }
00641
00642 void GalilPrivate::Init1800(int axes, bool dmc1802)
00643 {
00644
00645   map["TIME"] = Source(0, "UW", -1, "samples", "Sample counter");
00646
00647   //Digital Inputs
00648   map["_TI0"] = Source(2, "UB", -1, "", "Digital inputs 1 to 8");
00649   input_bits(2, 1);
00650
00651   map["_TI1"] = Source(3, "UB", -1, "", "Digital inputs 9 to 16"); //TI always included
00652   if (axes > 4) //9-16 depend on axes 5-8
00653     input_bits(3, 9);
00654
00655   //Digital outputs
00656   map["_OP0"] = Source(12, "UW", -1, "", "Digital outputs 1 to 16");
00657   output_bits(12, 1);
00658   if (axes > 4) //9-16 depend on axes 5-8
00659     output_bits(13, 9);
00660
00661   //Extended I/O
00662   int co = -1;
00663   if (GCmdI(g, "MG_CO", &co) == G_NO_ERROR)
00664   {
00665     map["_TI2"] = Source(4, "UB", -1, "", "Digital inputs 17 to 24"); //TI always included in gcl
00666     map["_TI3"] = Source(5, "UB", -1, "", "Digital inputs 25 to 32");
00667     map["_TI4"] = Source(6, "UB", -1, "", "Digital inputs 33 to 40");
00668     map["_TI5"] = Source(7, "UB", -1, "", "Digital inputs 41 to 48");
00669     map["_TI6"] = Source(8, "UB", -1, "", "Digital inputs 49 to 56");
00670     map["_TI7"] = Source(9, "UB", -1, "", "Digital inputs 57 to 64");
00671     map["_TI8"] = Source(10, "UB", -1, "", "Digital inputs 65 to 72");
00672     map["_TI9"] = Source(11, "UB", -1, "", "Digital inputs 73 to 80");
00673
00674     map["_OP1"] = Source(14, "UW", -1, "", "Digital outputs 17 to 32"); //OP always included in gcl
00675     map["_OP2"] = Source(16, "UW", -1, "", "Digital outputs 33 to 48");
00676     map["_OP3"] = Source(18, "UW", -1, "", "Digital outputs 49 to 64");
00677     map["_OP4"] = Source(20, "UW", -1, "", "Digital outputs 65 to 80");
00678
00679     if (co & 0x00000001) //bank 2 is output
00680       output_bits(14, 17);
00681     else //input
00682       input_bits(4, 17);
```

```
00683
00684     if (co & 0x00000002) //bank 3 is output
00685       output_bits(15, 25);
00686     else //input
00687       input_bits(5, 25);
00688
00689     if (co & 0x00000004) //bank 4 is output
00690       output_bits(16, 33);
00691     else //input
00692       input_bits(6, 33);
00693
00694     if (co & 0x00000008) //bank 5 is output
00695       output_bits(17, 41);
00696     else //input
00697       input_bits(7, 41);
00698
00699     if (co & 0x00000010) //bank 6 is output
00700       output_bits(18, 49);
00701     else //input
00702       input_bits(8, 49);
00703
00704     if (co & 0x00000020) //bank 7 is output
00705       output_bits(19, 57);
00706     else //input
00707       input_bits(9, 57);
00708
00709     if (co & 0x00000040) //bank 8 is output
00710       output_bits(20, 65);
00711     else //input
00712       input_bits(10, 65);
00713
00714     if (co & 0x00000080) //bank 9 is output
00715       output_bits(21, 73);
00716     else //input
00717       input_bits(11, 73);
00718   } //extended io
00719
00720   map["_TC"] = Source(22, "UB", -1, "", "Error code");
00721
00722   //general status
00723   map["_EO"] = Source(23, "UB", 0, "Boolean", "Echo on");
00724   map["TR"] = Source(23, "UB", 1, "Boolean", "Trace on");
00725   map["IN"] = Source(23, "UB", 2, "Boolean", "IN waiting for user input");
00726   map["XQ"] = Source(23, "UB", 7, "Boolean", "Program running");
00727
00728   //S plane
00729   map["_CSS"] = Source(24, "UW", -1, "segments", "Axis S segment count");
00730   map["VDS"] = Source(26, "UB", 3, "Boolean", "Axis S final deceleration");
00731   map["STS"] = Source(26, "UB", 4, "Boolean", "Axis S stopping");
00732   map["VSS"] = Source(26, "UB", 5, "Boolean", "Axis S slewing");
00733   map["_BGS"] = Source(27, "UB", 7, "Boolean", "Axis S moving");
00734   map["_AVS"] = Source(28, "SL", -1, "counts", "Axis S length");
00735
00736   //T plane
00737   map["_CST"] = Source(32, "UW", -1, "segments", "Axis T segment count");
00738   map["VDT"] = Source(34, "UB", 3, "Boolean", "Axis T final deceleration");
00739   map["STT"] = Source(34, "UB", 4, "Boolean", "Axis T stopping");
00740   map["VST"] = Source(34, "UB", 5, "Boolean", "Axis T slewing");
00741   map["_BGT"] = Source(35, "UB", 7, "Boolean", "Axis T moving");
00742   map["_AVT"] = Source(36, "SL", -1, "counts", "Axis T length");
00743
00744   //per-axis data
00745   int base = 40; //start of A axis data
00746   for (int i = 0; i < axes; i++)
00747   {
00748     map[ax("_MO", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " motor off"));
00749     map[ax("_OE", i, "")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " off-on-error set"));
00750     map[ax("_AL", i, "")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " latch armed"));
00751     map[ax("DC", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " began deceleration"));
00752     map[ax("ST", i, "")] = Source(base, "UW", 4, "Boolean", ax("Axis ", i, " began stop"));
00753     map[ax("SP", i, "")] = Source(base, "UW", 5, "Boolean", ax("Axis ", i, " began slew"));
00754     map[ax("CM", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " in contour mode"));
00755     map[ax("JG", i, "-")] = Source(base, "UW", 7, "Boolean", ax("Axis ", i, " negative move"));
00756     ++base; //41
00757     map[ax("VM", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " in VM or LI mode"));
00758     map[ax("HM", i, "2")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " finding index"));
00759     map[ax("HM", i, "1")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " coming off home
    switch"));
00760     map[ax("HM", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " home command issued"));
00761     map[ax("FE", i, "")] = Source(base, "UW", 4, "Boolean", ax("Axis ", i, " find edge issued"));
00762     map[ax("PA", i, "")] = Source(base, "UW", 5, "Boolean", ax("Axis ", i, " in PA mode"));
00763     map[ax("PR", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " in PA or PR mode"));
00764     map[ax("_BG", i, "")] = Source(base, "UW", 7, "Boolean", ax("Axis ", i, " move in progress"));
00765     ++base; //42
00766     map[ax("SM", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " stepper jumper
    installed"));
00767     map[ax("_HM", i, "")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " home switch"));
```

```
00768     map[ax("_LR", i, "")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " reverse limit switch"));
00769     map[ax("_LF", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " forward limit switch"));
00770     //4 and 5 reserved
00771     map[ax("AL", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " latch digital input"));
00772     map[ax("_AL", i, "=0")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " latch occurred"));
00773     ++base; //43
00774     map[ax("_SC", i, "")] = Source(base, "UB", -1, "", ax("Axis ", i, " stop code"));
00775     ++base; //44
00776     map[ax("_RP", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " reference position"));
00777     base += 4; //48
00778     map[ax("_TP", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " encoder position"));
00779     base += 4; //52
00780     map[ax("_TE", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " position error"));
00781     base += 4; //56
00782     map[ax("_TD", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " dual encoder
      position"));
00783     base += 4; //60
00784     map[ax("_TV", i, "")] = Source(base, "SL", -1, "counts/s", ax("Axis ", i, " filtered velocity"),
      64);
00785     base += 4; //64
00786     map[ax("_TT", i, "")] = Source(base, "SW", -1, "V", ax("Axis ", i, " torque (DAC)"), 3255);
00787     base += 2; //66
00788
00789     if (!dmc1802) //1800 has onboard Analog inputs
00790       map["@AN[" + to_string(i + 1) + "]"] = Source(base, "SW", -1, "V", "Analog input " + to_string(i
      + 1), 3276.8);
00791
00792     base += 2; //68, pointing to next axis
00793   } //for
00794 }
00795
00796
00797 void GalilPrivate::Init30010(bool dmc31010)
00798 {
00799   //0-3 Header is ignored in GCL
00800
00801   map["TIME"] = Source(4, "UW", -1, "samples", "Sample counter");
00802
00803   map["@IN[1]"] = Source(6, "UB", 0, "Boolean", "Digital input 1");
00804   map["@IN[2]"] = Source(6, "UB", 1, "Boolean", "Digital input 2");
00805   map["@IN[3]"] = Source(6, "UB", 2, "Boolean", "Digital input 3");
00806   map["@IN[4]"] = Source(6, "UB", 3, "Boolean", "Digital input 4");
00807   map["@IN[5]"] = Source(6, "UB", 4, "Boolean", "Digital input 5");
00808   map["@IN[6]"] = Source(6, "UB", 5, "Boolean", "Digital input 6");
00809   map["@IN[7]"] = Source(6, "UB", 6, "Boolean", "Digital input 7");
00810   map["@IN[8]"] = Source(6, "UB", 7, "Boolean", "Digital input 8");
00811
00812   map["@OUT[1]"] = Source(8, "UB", 0, "Boolean", "Digital output 1");
00813   map["@OUT[2]"] = Source(8, "UB", 1, "Boolean", "Digital output 2");
00814   map["@OUT[3]"] = Source(8, "UB", 2, "Boolean", "Digital output 3");
00815   map["@OUT[4]"] = Source(8, "UB", 3, "Boolean", "Digital output 4");
00816
00817   map["_TC"] = Source(10, "UB", -1, "", "Error code");
00818
00819   //Thread status
00820   map["NO0"] = Source(11, "UB", 0, "Boolean", "Thread 0 running");
00821   map["NO1"] = Source(11, "UB", 1, "Boolean", "Thread 1 running");
00822   map["NO2"] = Source(11, "UB", 2, "Boolean", "Thread 2 running");
00823   map["NO3"] = Source(11, "UB", 3, "Boolean", "Thread 3 running");
00824   map["NO4"] = Source(11, "UB", 4, "Boolean", "Thread 4 running"); //Firmware prior to 1.2a has only 4
      threads
00825   map["NO5"] = Source(11, "UB", 5, "Boolean", "Thread 5 running");
00826
00827   //Analog IO
00828   //version 1.1b provides 16 bit AQ-compatible data in data record
00829   if (dmc31010)
00830     aq_analog(12, 2);
00831   else
00832     map["@AN[2]"] = Source(12, "UW", -1, "V", "Analog input 2", 13107.2); //0-5 16 bit upsampling
00833
00834   map["@AO[1]"] = Source(14, "SW", -1, "V", "Analog output 1", 3276.8); //+/- 10v
00835   map["@AO[2]"] = Source(16, "SW", -1, "V", "Analog output 2", 3276.8);
00836
00837   //Amp status
00838   map["TA00"] = Source(18, "UB", 0, "Boolean", "Axis A over current");
00839   map["TA01"] = Source(18, "UB", 1, "Boolean", "Axis A over voltage");
00840   map["TA02"] = Source(18, "UB", 2, "Boolean", "Axis A over temperature");
00841   map["TA03"] = Source(18, "UB", 3, "Boolean", "Axis A under voltage");
00842   map["TA1A"] = Source(19, "UB", 0, "Boolean", "Axis A hall error");
00843   map["TA2A"] = Source(20, "UB", 0, "Boolean", "Axis A at _TKA peak current");
00844   map["TA3AD"] = Source(21, "UB", 0, "Boolean", "Axis A ELO active");
00845
00846   //contour mode
00847   map["CD"] = Source(22, "UL", -1, "segments", "Contour segment count");
00848   map["_CM"] = Source(26, "UW", -1, "elements", "Contour buffer space");
00849
00850   //S plane
```

```
00851   map["_CSS"] = Source(28, "UW", -1, "segments", "Axis S segment count");
00852   map["VDS"] = Source(30, "UB", 3, "Boolean", "Axis S final deceleration");
00853   map["STS"] = Source(30, "UB", 4, "Boolean", "Axis S stopping");
00854   map["VSS"] = Source(30, "UB", 5, "Boolean", "Axis S slewing");
00855   map["_BGS"] = Source(31, "UB", 7, "Boolean", "Axis S moving");
00856   map["_AVS"] = Source(32, "SL", -1, "counts", "Axis S length");
00857   map["_LMS"] = Source(36, "UW", -1, "elements", "Axis S buffer speace");
00858
00859   //per-axis data
00860   int base = 38; //starting offset
00861   int i = 0; //only one axis on 30010, no need to iterate axes
00862
00863   map[ax("_MO", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " motor off"));
00864   map[ax("HM", i, "3")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " found index"));
00865   map[ax("_AL", i, "")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " latch armed"));
00866   map[ax("DC", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " began deceleration"));
00867   map[ax("ST", i, "")] = Source(base, "UW", 4, "Boolean", ax("Axis ", i, " began stop"));
00868   map[ax("SP", i, "")] = Source(base, "UW", 5, "Boolean", ax("Axis ", i, " began slew"));
00869   map[ax("CM", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " in contour mode"));
00870   map[ax("JG", i, "-")] = Source(base, "UW", 7, "Boolean", ax("Axis ", i, " negative move"));
00871   ++base; //39
00872   map[ax("VM", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " in VM or LI mode"));
00873   map[ax("HM", i, "2")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " finding index"));
00874   map[ax("HM", i, "1")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " coming off home switch"));
00875   map[ax("HM", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " home command issued"));
00876   map[ax("FE", i, "")] = Source(base, "UW", 4, "Boolean", ax("Axis ", i, " find edge issued"));
00877   map[ax("PA", i, "")] = Source(base, "UW", 5, "Boolean", ax("Axis ", i, " in PA mode"));
00878   map[ax("PR", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " in PA or PR mode"));
00879   map[ax("_BG", i, "")] = Source(base, "UW", 7, "Boolean", ax("Axis ", i, " move in progress"));
00880   ++base; //40
00881   map[ax("MT", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " in stepper mode"));
00882   map[ax("_HM", i, "")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " home switch"));
00883   map[ax("_LR", i, "")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " reverse limit switch"));
00884   map[ax("_LF", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " forward limit switch"));
00885   //4 and 5 reserved
00886   map[ax("AL", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " latch digital input"));
00887   map[ax("_AL", i, "=0")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " latch occurred"));
00888   ++base; //41
00889   map[ax("_SC", i, "")] = Source(base, "UB", -1, "", ax("Axis ", i, " stop code"));
00890   ++base; //42
00891   map[ax("_RP", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " reference position"));
00892   base += 4; //46
00893   map[ax("_TP", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " encoder position"));
00894   base += 4; //50
00895   map[ax("_TE", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " position error"));
00896   base += 4; //54
00897   map[ax("_TD", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " dual encoder position"));
00898   base += 4; //58
00899   map[ax("_TV", i, "")] = Source(base, "SL", -1, "counts/s", ax("Axis ", i, " filtered velocity"),
      64);
00900   base += 4; //62
00901   map[ax("_TT", i, "")] = Source(base, "SL", -1, "V", ax("Axis ", i, " torque (DAC)"), 3255);
00902   base += 4; //66
00903
00904   //version 1.1b provides 16 bit AQ-compatible data in data record
00905   if (dmc31010)
00906     aq_analog(base, i + 1);
00907   else
00908     map["@AN[" + to_string(i + 1) + "]"] = Source(base, "UW", -1, "V", "Analog input " + to_string(i +
      1), 13107.2);
00909
00910   base += 2; //68
00911
00912   map[ax("_QH", i, "")] = Source(base, "UB", -1, "", ax("Axis ", i, " hall sensors"));
00913   base++; //69 reserved
00914   base++; //70
00915   map[ax("_ZA", i, "")] = Source(base, "SL", -1, "", ax("Axis ", i, " user variable"));
00916   base += 4; //74
00917
00918 }
00919
00920 void GalilPrivate::Init2103(int axes)
00921 {
00922
00923   bool db28040 = (GCmd(g, "MG @AN[1]") == G_NO_ERROR); //probe @AN for existance of DB-28040
00924
00925   //0-3 Header is ignored in GCL
00926
00927   map["TIME"] = Source(4, "UW", -1, "samples", "Sample counter");
00928
00929   //Digital Inputs
00930   map["_TI0"] = Source(6, "UB", -1, "", "Digital inputs 1 to 8");
00931   input_bits(6, 1);
00932
00933   map["_TI1"] = Source(7, "UB", -1, "", "Digital inputs 9 to 16"); //TI always included
00934   if (axes > 4) //9-16 depend on axes 5-8
00935     input_bits(7, 9);
```

```
00936
00937   //Digital outputs
00938   map["_OP0"] = Source(16, "UW", -1, "", "Digital outputs 1 to 16");
00939   output_bits(16, 1);
00940
00941   if (axes > 4) //9-16 depend on axes 5-8
00942     output_bits(17, 9);
00943
00944   //Extended I/O
00945   int co = -1;
00946   if (db28040 && (GCmdI(g, "MG_CO", &co) == G_NO_ERROR))
00947   {
00948     map["_TI2"] = Source(8, "UB", -1, "", "Digital inputs 17 to 24"); //TI always included in gcl
00949     map["_TI3"] = Source(9, "UB", -1, "", "Digital inputs 25 to 32");
00950     map["_TI4"] = Source(10, "UB", -1, "", "Digital inputs 33 to 40");
00951     map["_TI5"] = Source(11, "UB", -1, "", "Digital inputs 41 to 48");
00952     map["_TI6"] = Source(12, "UB", -1, "", "Digital inputs 49 to 56");
00953
00954     map["_OP1"] = Source(18, "UW", -1, "", "Digital outputs 17 to 32"); //OP always included in gcl
00955     map["_OP2"] = Source(20, "UW", -1, "", "Digital outputs 33 to 48");
00956     map["_OP3"] = Source(22, "UW", -1, "", "Digital outputs 49 to 64");
00957
00958     if (co & 0x00000001) //bank 2 is output
00959       output_bits(18, 17);
00960     else //bank 2 in input
00961       input_bits(8, 17);
00962
00963     if (co & 0x00000002) //bank 3 is output
00964       output_bits(19, 25);
00965     else //bank 3 is input
00966       input_bits(9, 25);
00967
00968     if (co & 0x00000004) //bank 4 is output
00969       output_bits(20, 33);
00970     else //bank 4 is input
00971       input_bits(10, 33);
00972
00973     if (co & 0x00000008) //bank 5 is output
00974       output_bits(21, 41);
00975     else //bank 5 is input
00976       input_bits(11, 41);
00977
00978     if (co & 0x00000010) //bank 6 is output
00979       output_bits(22, 49);
00980     else //bank 6 is input
00981       input_bits(12, 49);
00982   }
00983
00984   map["_TC"] = Source(26, "UB", -1, "", "Error code");
00985
00986   //general status
00987   map["_EO"] = Source(27, "UB", 0, "Boolean", "Echo on");
00988   map["TR"] = Source(27, "UB", 1, "Boolean", "Trace on");
00989   map["IN"] = Source(27, "UB", 2, "Boolean", "IN waiting for user input");
00990   map["XQ"] = Source(27, "UB", 7, "Boolean", "Program running");
00991
00992   //S plane
00993   map["_CSS"] = Source(28, "UW", -1, "segments", "Axis S segment count");
00994   map["VDS"] = Source(30, "UB", 3, "Boolean", "Axis S final deceleration");
00995   map["STS"] = Source(30, "UB", 4, "Boolean", "Axis S stopping");
00996   map["VSS"] = Source(30, "UB", 5, "Boolean", "Axis S slewing");
00997   map["_BGS"] = Source(31, "UB", 7, "Boolean", "Axis S moving");
00998   map["_AVS"] = Source(32, "SL", -1, "counts", "Axis S length");
00999
01000   //T plane
01001   map["_CST"] = Source(36, "UW", -1, "segments", "Axis T segment count");
01002   map["VDT"] = Source(38, "UB", 3, "Boolean", "Axis T final deceleration");
01003   map["STT"] = Source(38, "UB", 4, "Boolean", "Axis T stopping");
01004   map["VST"] = Source(38, "UB", 5, "Boolean", "Axis T slewing");
01005   map["_BGT"] = Source(39, "UB", 7, "Boolean", "Axis T moving");
01006   map["_AVT"] = Source(40, "SL", -1, "counts", "Axis T length");
01007
01008   //per-axis data
01009   int base = 44; //start of A axis data
01010   for (int i = 0; i < axes; i++)
01011   {
01012     map[ax("_MO", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " motor off"));
01013     map[ax("_OE", i, "")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " off-on-error set"));
01014     map[ax("_AL", i, "")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " latch armed"));
01015     map[ax("DC", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " began deceleration"));
01016     map[ax("ST", i, "")] = Source(base, "UW", 4, "Boolean", ax("Axis ", i, " began stop"));
01017     map[ax("SP", i, "")] = Source(base, "UW", 5, "Boolean", ax("Axis ", i, " began slew"));
01018     map[ax("CM", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " in contour mode"));
01019     map[ax("JG", i, "-")] = Source(base, "UW", 7, "Boolean", ax("Axis ", i, " negative move"));
01020     ++base; //45
01021     map[ax("VM", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " in VM or LI mode"));
01022     map[ax("HM", i, "2")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " finding index"));
```

```
01023     map[ax("HM", i, "1")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " coming off home
      switch"));
01024     map[ax("HM", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " home command issued"));
01025     map[ax("FE", i, "")] = Source(base, "UW", 4, "Boolean", ax("Axis ", i, " find edge issued"));
01026     map[ax("PA", i, "")] = Source(base, "UW", 5, "Boolean", ax("Axis ", i, " in PA mode"));
01027     map[ax("PR", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " in PA or PR mode"));
01028     map[ax("_BG", i, "")] = Source(base, "UW", 7, "Boolean", ax("Axis ", i, " move in progress"));
01029     ++base; //46
01030     map[ax("SM", i, "")] = Source(base, "UW", 0, "Boolean", ax("Axis ", i, " stepper jumper
      installed"));
01031     map[ax("_HM", i, "")] = Source(base, "UW", 1, "Boolean", ax("Axis ", i, " home switch"));
01032     map[ax("_LR", i, "")] = Source(base, "UW", 2, "Boolean", ax("Axis ", i, " reverse limit switch"));
01033     map[ax("_LF", i, "")] = Source(base, "UW", 3, "Boolean", ax("Axis ", i, " forward limit switch"));
01034     //4 and 5 reserved
01035     map[ax("AL", i, "")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " latch digital input"));
01036     map[ax("_AL", i, "=0")] = Source(base, "UW", 6, "Boolean", ax("Axis ", i, " latch occurred"));
01037     ++base; //47
01038     map[ax("_SC", i, "")] = Source(base, "UB", -1, "", ax("Axis ", i, " stop code"));
01039     ++base; //48
01040     map[ax("_RP", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " reference position"));
01041     base += 4; //52
01042     map[ax("_TP", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " encoder position"));
01043     base += 4; //56
01044     map[ax("_TE", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " position error"));
01045     base += 4; //60
01046     map[ax("_TD", i, "")] = Source(base, "SL", -1, "counts", ax("Axis ", i, " dual encoder
      position"));
01047     base += 4; //64
01048     map[ax("_TV", i, "")] = Source(base, "SL", -1, "counts/s", ax("Axis ", i, " filtered velocity"),
      64);
01049     base += 4; //68
01050     map[ax("_TT", i, "")] = Source(base, "SW", -1, "V", ax("Axis ", i, " torque (DAC)"), 3255);
01051     base += 2; //70
01052
01053     if (db28040) //card has onboard Analog inputs
01054     {
01055       aq_analog(base, i + 1); //map in the analog
01056     }
01057     base += 2; //72
01058   } //for
01059 }
01060
01061 void GalilPrivate::InitRio(bool rio3)
01062 {
01063
01064
01065   //0-3 Header is ignored in GCL
01066
01067   map["TIME"] = Source(4, "UW", -1, "samples", "Sample counter");
01068   map["_TC"] = Source(6, "UB", -1, "", "Error code");
01069
01070   //general status
01071   map["_EO"] = Source(7, "UB", 0, "Boolean", "Echo on");
01072   map["TR"] = Source(7, "UB", 1, "Boolean", "Trace on");
01073   map["IN"] = Source(7, "UB", 2, "Boolean", "IN waiting for user input");
01074   map["XQ"] = Source(7, "UB", 7, "Boolean", "Program running");
01075
01076   bool aqdq = (q->command("ID").find("(AQ)") != string::npos); //progammable analog I/O
01077
01078   if (aqdq)
01079   {
01080     dq_analog(8, 0);
01081     dq_analog(10, 1);
01082     dq_analog(12, 2);
01083     dq_analog(14, 3);
01084     dq_analog(16, 4);
01085     dq_analog(18, 5);
01086     dq_analog(20, 6);
01087     dq_analog(22, 7);
01088   }
01089   else //fixed 0-5V
01090   {
01091     map["@AO[0]"] = Source(8, "UW", -1, "V", "Analog output 0", 13107.2, 0);
01092     map["@AO[1]"] = Source(10, "UW", -1, "V", "Analog output 1", 13107.2, 0);
01093     map["@AO[2]"] = Source(12, "UW", -1, "V", "Analog output 2", 13107.2, 0);
01094     map["@AO[3]"] = Source(14, "UW", -1, "V", "Analog output 3", 13107.2, 0);
01095     map["@AO[4]"] = Source(16, "UW", -1, "V", "Analog output 4", 13107.2, 0);
01096     map["@AO[5]"] = Source(18, "UW", -1, "V", "Analog output 5", 13107.2, 0);
01097     map["@AO[6]"] = Source(20, "UW", -1, "V", "Analog output 6", 13107.2, 0);
01098     map["@AO[7]"] = Source(22, "UW", -1, "V", "Analog output 7", 13107.2, 0);
01099   }
01100
01101
01102   if (aqdq)
01103   {
01104     aq_analog(24, 0);
01105     aq_analog(26, 1);
```

```
01106     aq_analog(28, 2);
01107     aq_analog(30, 3);
01108     aq_analog(32, 4);
01109     aq_analog(34, 5);
01110     aq_analog(36, 6);
01111     aq_analog(38, 7);
01112   }
01113   else  //fixed 0-5V
01114   {
01115     map["@AN[0]"] = Source(24, "UW", -1, "V", "Analog input 0", 13107.2, 0);
01116     map["@AN[1]"] = Source(26, "UW", -1, "V", "Analog input 1", 13107.2, 0);
01117     map["@AN[2]"] = Source(28, "UW", -1, "V", "Analog input 2", 13107.2, 0);
01118     map["@AN[3]"] = Source(30, "UW", -1, "V", "Analog input 3", 13107.2, 0);
01119     map["@AN[4]"] = Source(32, "UW", -1, "V", "Analog input 4", 13107.2, 0);
01120     map["@AN[5]"] = Source(34, "UW", -1, "V", "Analog input 5", 13107.2, 0);
01121     map["@AN[6]"] = Source(36, "UW", -1, "V", "Analog input 6", 13107.2, 0);
01122     map["@AN[7]"] = Source(38, "UW", -1, "V", "Analog input 7", 13107.2, 0);
01123   }
01124
01125   //Data record diverges here for RIO471/472 and RIO473
01126   int base = 40;
01127
01128   //outputs
01129   map["_OP0"] = Source(base, "UB", -1, "", "Digital ouputs 0-7");
01130   output_bits(base, 0);
01131   base++;
01132
01133   map["_OP1"] = Source(base, "UB", -1, "", "Digital outputs 8-15");
01134   output_bits(base, 8);
01135   base++;
01136
01137   if (rio3)
01138   {
01139     map["_OP2"] = Source(base, "UB", -1, "", "Digital outputs 16-23");
01140     output_bits(base, 16);
01141     base++;
01142     base++; //one more byte in IO space
01143   }
01144
01145   //inputs
01146   map["_TI0"] = Source(base, "UB", -1, "", "Digital inputs 0-7");
01147   input_bits(base, 0);
01148   base++;
01149
01150   map["_TI1"] = Source(base, "UB", -1, "", "Digital inputs 8-15");
01151   input_bits(base, 8);
01152   base++;
01153
01154   if (rio3)
01155   {
01156     map["_TI2"] = Source(base, "UB", -1, "", "Digital inputs 16-23");
01157     input_bits(base, 16);
01158     base++;
01159     base++; //one more byte in IO space
01160   }
01161
01162   //pulse counter
01163   map["_PC"] = Source(base, "UL", -1, "edges", "Pulse counter");
01164   base += 4;
01165
01166   //user vars
01167   map["_ZC"] = Source(base, "SL", -1, "", "1st user variable");
01168   base += 4;
01169   map["_ZD"] = Source(base, "SL", -1, "", "2nd user variable");
01170   base += 4;
01171 }
01172
01173 void GalilPrivate::InitRio3_24Ex()
01174 {
01175   //Extended I/O tacks 8 bytes on the end of the data record, three bytes of each of I/O, one reserved
      for each
01176   map["_OP3"] = Source(60, "UB", -1, "", "Digital outputs 24-31");
01177   output_bits(60, 24);
01178   map["_OP4"] = Source(61, "UB", -1, "", "Digital outputs 32-39");
01179   output_bits(61, 32);
01180   map["_OP5"] = Source(62, "UB", -1, "", "Digital outputs 40-47");
01181   output_bits(62, 40);
01182   //byte 63 is reserved
01183
01184   map["_TI3"] = Source(64, "UB", -1, "", "Digital inputs 24-31");
01185   input_bits(64, 24);
01186   map["_TI4"] = Source(65, "UB", -1, "", "Digital inputs 32-39");
01187   input_bits(65, 32);
01188   map["_TI5"] = Source(66, "UB", -1, "", "Digital inputs 40-47");
01189   input_bits(66, 40);
01190   //byte 67 is reserved
01191 }
```

```
01192
01193 void GalilPrivate::InitRioSer(bool rio3)
01194 {
01195   //SER tacks 4 longs on the end of the data record (4 encoders)
01196   int base = rio3 ? 60 : 56; //RIO 47300 base data record is longer than the other RIO products due to
      24 i/o standard
01197   map["_QE0"] = Source(base, "SL", -1, "counts", "encoder position"); base += 4;
01198   map["_QE1"] = Source(base, "SL", -1, "counts", "encoder position"); base += 4;
01199   map["_QE2"] = Source(base, "SL", -1, "counts", "encoder position"); base += 4;
01200   map["_QE3"] = Source(base, "SL", -1, "counts", "encoder position"); base += 4;
01201 }
```

10.189 gcl_galil.cpp File Reference

```
#include "gcl_galil.h"
```

Functions

- void ec (GReturn rc)
- string AddressConvert (const string &gcl_address)

  Takes a GCL address string and returns the equivalent gclib address string.

10.189.1 Function Documentation

10.189.1.1 AddressConvert()

```
string AddressConvert (
            const string & gcl_address )
```

Takes a GCL address string and returns the equivalent gclib address string.
Definition at line 80 of file gcl_galil.cpp.
References vector().

10.189.1.2 ec()

```
void ec (
            GReturn rc )
```

Definition at line 8 of file gcl_galil.cpp.

10.190 gcl_galil.cpp

Go to the documentation of this file.

```
00001
00004 #include "gcl_galil.h"
00005 using namespace std;
00006
00007
00008 void ec(GReturn rc)
00009 {
00010
00011   switch (rc)
00012   {
00013   case G_NO_ERROR:
00014     return;
00015
00016   case G_TIMEOUT:
00017     throw string("1010 TIMEOUT ERROR.  Galil::command() took longer than timeout to return\n");
00018     break;
00019
00020   case G_OPEN_ERROR:
00021     throw string("5002 OPEN ERROR.  Galil::Galil() failed to open device\n");
00022     break;
00023
00024   case G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND:
00025     throw string("7020 INVALID COMMAND ERROR.  DL, ED, and QD are not allowed from
      Galil::command()\n");
00026     break;
00027
00028   case G_DATA_RECORD_ERROR:
```

```
00029     throw string("6150 WRONG BUS ERROR.  Galil::record(\"DR\") not supported on RS-232.  Use Ethernet
      or Galil::record(\"QR\")\n");
00030     break;
00031
00032   case G_FIRMWARE_LOAD_NOT_SUPPORTED:
00033     throw string("6130 WRONG BUS ERROR.  Galil::firmwareDownloadFile() isn't allowed via Ethernet.
      Use RS-232\n"); //21x3 etc.
00034     break;
00035
00036   case G_ILLEGAL_DATA_IN_PROGRAM:
00037     throw string("7060 INVALID CHARACTER ERROR.  Galil::programDownload() can't download program with
      backslash \\ character.  Use {^92} in MG commands\n");
00038     break;
00039
00040   case G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT:
00041     throw string("7061 INVALID LENGTH ERROR.  Galil::programDownload() can't compress\n");
00042     break;
00043
00044   case G_BAD_RESPONSE_QUESTION_MARK:
00045     throw string("2010 COMMAND ERROR.  Galil::command() got ? instead of : response.\n");
00046     break;
00047
00048   case G_BAD_FILE:
00049     throw string("4000 FILE ERROR.  Galil::Galil() failed to open file\n");
00050     break;
00051
00052   case G_UNSUPPORTED_FUNCTION:
00053     throw string("6000 WRONG BUG ERROR. Function isn't allowed on this bus\n");
00054     break;
00055
00056
00057   /*
00058   case G_DATA_RECORD_ERROR:
00059   case G_BAD_ADDRESS:
00060   case G_BAD_LOST_DATA:
00061   case G_BAD_VALUE_RANGE:
00062   case G_BAD_FULL_MEMORY:
00063   case G_ARRAY_NOT_DIMENSIONED:
00064   case G_GCLIB_ERROR:
00065   case G_GCLIB_UTILITY_ERROR:
00066   case G_INVALID_PREPROCESSOR_OPTIONS:
00067   */
00068
00069   default: //couldn't find error map, make a gcl-like error from gclib error code
00070   {
00071     char buf[G_SMALL_BUFFER];
00072     GError(rc, buf, sizeof(buf)); //get the error message
00073     throw (to_string(rc) + " GCLIB ERROR. " + string(buf) + '\n');
00074   }
00075
00076   }//switch
00077 }
00078
00080 string AddressConvert(const string& gcl_address)
00081 {
00082   if (gcl_address.size() == 0) //this implementation does not present the user with a connections
      dialog if a nullstring is passed.
00083     throw string("5005 OPEN ERROR.  Null string specified in Galil::Galil()\n");
00084
00085   if (gcl_address.find("OFFLINE") != string::npos)
00086     throw string("5001 OPEN ERROR.  OFFLINE specified to Galil::Galil()\n");
00087
00088   vector<string> args;
00089   size_t start = 0;
00090   size_t i;
00091   for (i = 0; i < gcl_address.size(); i++) //split into tokens
00092   {
00093     if (gcl_address[i] == ' ')
00094     {
00095       if (start < i) //if not zero length
00096         args.push_back(gcl_address.substr(start, i - start));
00097
00098       start = i + 1; //jump over space
00099     }
00100   }
00101   if (start < i) //one token still remaining
00102     args.push_back(gcl_address.substr(start, i - start));
00103
00104   bool ei = true; //bools to remember if data streams should be subscribed to, gcl subscribes by
      default
00105   bool mg = true;
00106   bool dr = true;
00107   string address;
00108   args.push_back(""); //allow safe indexing one past the end of the tokens list
00109   for (i = 0; i < args.size() - 1; i++)
00110   {
00111     if (args[i] == "-p1") address.append("--p1 " + args[++i] + " ");
```

```
00112     else if (args[i] == "-p2") address.append("--p2 " + args[++i] + " ");
00113     else if (args[i] == "-udp") address.append("--command UDP "); //TCP is default in gclib
00114     else if ((args[i] == "-ei") && (args[++i] == "0")) ei = false;
00115     else if ((args[i] == "-mg") && (args[++i] == "0")) mg = false;
00116     else if ((args[i] == "-dr") && (args[++i] == "0")) dr = false;
00117     else if (args[i] == "-t") address.append("-t " + args[++i] + " ");
00118     else if (args[i] == "-s")
00119     {
00120       address.append("-s NONE ");
00121       ei = false;
00122       mg = false;
00123       dr = false;
00124     }
00125     else if (args[i] == "-d") ++i; //Debug not yet supported
00126     else if (args[i] == "-l") ++i; //Long-timeout not used
00127     else if (args[i].find("COM") != string::npos)
00128     {
00129       address.append(args[i] + " --baud " + args[i + 1] + " "); //COM1 --baud 115200
00130       i++;// jump over baud rate on next pass
00131     }
00132     else address.append(args[i] + " "); //keep unrecognized tokens, e.g. 192.168.0.123
00133   }
00134
00135   if (ei) address.append("-s EI ");
00136   if (mg) address.append("-s MG ");
00137   if (dr) address.append("-s DR ");
00138
00139   address.append("-d "); //add direct-connect switch
00140
00141   return address;
00142 }
00143
00144 string Galil::libraryVersion()
00145 {
00146   char buf[G_SMALL_BUFFER]; //function is static, so can't access GalilPrivate members
00147   ec(GVersion(buf, sizeof(buf)));
00148   return "Galil2.dll wrapper, gclib " + string(buf);
00149 }
00150
00151 vector<string> Galil::addresses()
00152 {
00153   vector<string> addresses; //this is the return collection
00154   char buf[1024];
00155   ec(GAddresses(buf, sizeof(buf)));
00156   //buf now holds the list, but the gcl only holds the first cell
00157
00158   short commas = 0; //counter for comma delimiters
00159   string address; //temp buffer for holding chars
00160   size_t len = strlen(buf); //don't call strlen every iteration
00161   for (size_t i = 0; i < len; i++)
00162   {
00163     if (buf[i] == '\n') //end of line
00164     {
00165       addresses.push_back(address);
00166       address.clear();
00167       commas = 0; //start counting commas from zero
00168       continue;
00169     }
00170
00171     if (commas) //already saw first comma, keep looking for end of line
00172       continue;
00173
00174     if (buf[i] == ',')
00175     {
00176       commas++; //count it
00177       continue;
00178     }
00179
00180     address.push_back(buf[i]); //keep the char
00181   }//for
00182
00183   return addresses;
00184 }
00185
00186 Galil::Galil(std::string address)
00187 {
00188   GCon g;
00189   timeout_ms = 500; //default value for gcl
00190   ec(GOpen(AddressConvert(address).c_str(), &g));
00191   d = new GalilPrivate(this, g);
00192   d->InitializeDataRecord();
00193 }
00194
00195 Galil::~Galil()
00196 {
00197   GClose(d->g); //close the connection in gclib
00198   delete d; //free memory
```

```
00199 }
00200
00201 string Galil::connection()
00202 {
00203   ec(GInfo(d->g, d->tbuf, sizeof(d->tbuf)));
00204   return string(d->tbuf);
00205 }
00206
00207 string Galil::command(const std::string& command, const std::string& terminator, const std::string&
      ack, bool trim)
00208 {
00209   /*
00210   *  Note: This wrapper ignores terminator and ack. GCommand does not require them to operate.
00211   *  If terminator and/or ack are desired, this function can be implemented with GRead and GWrite.
00212   *  Please contact softwaresupport@galil.com with questions/concerns.
00213   */
00214   ec(GTimeout(d->g, (short)timeout_ms)); //obey timeout_ms setting
00215   GSize bytes_read;
00216   char* response;
00217   if (trim)
00218     ec(GCmdT(d->g, command.c_str(), d->tbuf, sizeof(d->tbuf), &response));
00219   else
00220   {
00221     ec(GCommand(d->g, command.c_str(), d->tbuf, sizeof(d->tbuf), &bytes_read));
00222     response = d->tbuf;
00223   }
00224   ec(GTimeout(d->g, -1)); //replace timeout
00225
00226   return string(response);
00227 }
00228
00229 double Galil::commandValue(const std::string& command)
00230 {
00231   double value;
00232   ec(GTimeout(d->g, (short)timeout_ms));
00233   ec(GCmdD(d->g, command.c_str(), &value));
00234   ec(GTimeout(d->g, -1));
00235   return value;
00236 }
00237
00238 string Galil::message(int timeout_ms)
00239 {
00240   /*
00241   *  From GCL documentation, http://www.galil.com/sw/pub/all/doc/galiltools/html/library.html#message
00242   * "If a zero timeout is specified, no errors will be thrown; message() will simply return the
      waiting queue (even if it is empty, "").
00243   * A -1 timeout will cause message() to block until a message is received."
00244   */
00245   GReturn rc = G_NO_ERROR;
00246   short t = 5000; //nominal timeout, to be used if -1 is specified
00247   if (timeout_ms >= 0)
00248     t = (short) timeout_ms;
00249
00250   ec(GTimeout(d->g, t));
00251
00252   do
00253   {
00254       rc = GMessage(d->g, d->tbuf, sizeof(d->tbuf));
00255   } while(timeout_ms == -1 && rc == G_TIMEOUT);
00256
00257   ec(GTimeout(d->g, -1));
00258
00259   if (rc == G_GCLIB_NON_BLOCKING_READ_EMPTY)
00260     return "";
00261
00262   ec(rc); //check the other possible return codes
00263   return string(d->tbuf);
00264
00265 }
00266
00267 int Galil::interrupt(int timeout_ms)
00268 {
00269   ec(GTimeout(d->g, (short)timeout_ms));
00270   GStatus status;
00271   ec(GInterrupt(d->g, &status));
00272   ec(GTimeout(d->g, -1));
00273   return (int)status;
00274 }
00275
00276
00277 string Galil::programUpload()
00278 {
00279   ec(GTimeout(d->g, (short)timeout_ms));
00280   ec(GProgramUpload(d->g, d->tbuf, sizeof(d->tbuf)));
00281   ec(GTimeout(d->g, -1));
00282   return string(d->tbuf);
00283 }
```

```
00284
00285 void Galil::programDownload(const std::string& program)
00286 {
00287   ec(GTimeout(d->g, (short)timeout_ms));
00288   ec(GProgramDownload(d->g, program.c_str(), 0)); //no special preprocessor directives
00289   ec(GTimeout(d->g, -1));
00290 }
00291
00292 void Galil::programUploadFile(const std::string& file)
00293 {
00294   ec(GTimeout(d->g, (short)timeout_ms));
00295   ec(GProgramUploadFile(d->g, file.c_str()));
00296   ec(GTimeout(d->g, -1));
00297 }
00298
00299 void Galil::programDownloadFile(const std::string& file)
00300 {
00301   ec(GTimeout(d->g, (short)timeout_ms));
00302   ec(GProgramDownloadFile(d->g, file.c_str(), 0));  //no special preprocessor directives
00303   ec(GTimeout(d->g, -1));
00304 }
00305
00306 vector<double> Galil::arrayUpload(const std::string& name)
00307 {
00308   ec(GTimeout(d->g, (short)timeout_ms));
00309   ec(GArrayUpload(d->g, name.c_str(), G_BOUNDS, G_BOUNDS, G_CR, d->tbuf, sizeof(d->tbuf)));
00310   ec(GTimeout(d->g, -1));
00311
00312   vector<double> vals;
00313   int len = strlen(d->tbuf); //gclib null terminates array buf
00314   char* start = d->tbuf; //start hold pointer to begining of string to atof
00315   for (int i = 0; i < len; i++)
00316   {
00317     if (d->tbuf[i] == '\r')
00318     {
00319       vals.push_back(atof(start));
00320       start = d->tbuf + i;
00321     }
00322   }
00323   vals.push_back(atof(start)); //last number still left in tbuf
00324   return vals;
00325 }
00326
00327 void Galil::arrayDownload(const std::vector<double>& array, const std::string& name)
00328 {
00329   //GArrayDownload requires a cstring containing the data
00330   string array_str;
00331   for (size_t i = 0; i < array.size(); i++)
00332   {
00333     sprintf(d->tbuf, "%0.4f\r", array[i]);
00334     array_str.append(d->tbuf);
00335   }
00336   ec(GTimeout(d->g, (short)timeout_ms));
00337
00338   //Galil::arrayDownload auto-dimensions the array table
00339   string command = "DA " + name + "[]";
00340   ec(GCmd(d->g, command.c_str())); //Deallocate array
00341   command = "DM " + name + "[" + to_string(array.size()) + "]";
00342   ec(GCmd(d->g, command.c_str())); //Allocate array with correct dimension
00343
00344   ec(GArrayDownload(d->g, name.c_str(), G_BOUNDS, G_BOUNDS, array_str.c_str()));
00345   ec(GTimeout(d->g, -1));
00346 }
00347
00348 void Galil::arrayUploadFile(const std::string& file, const std::string& names)
00349 {
00350   ec(GTimeout(d->g, (short)timeout_ms));
00351   ec(GArrayUploadFile(d->g, file.c_str(), names.c_str()));
00352   ec(GTimeout(d->g, -1));
00353 }
00354
00355 void Galil::arrayDownloadFile(const std::string& file)
00356 {
00357   ec(GTimeout(d->g, (short)timeout_ms));
00358   ec(GArrayDownloadFile(d->g, file.c_str()));
00359   ec(GTimeout(d->g, -1));
00360 }
00361
00362 void Galil::firmwareDownloadFile(const std::string& file)
00363 {
00364   ec(GTimeout(d->g, (short)timeout_ms));
00365   ec(GFirmwareDownload(d->g, file.c_str()));
00366   ec(GTimeout(d->g, -1));
00367 }
00368
00369 int Galil::write(const std::string& bytes)
00370 {
```

```
00371   ec(GTimeout(d->g, (short)timeout_ms));
00372   ec(GWrite(d->g, bytes.data(), bytes.length()));
00373   ec(GTimeout(d->g, -1));
00374   return bytes.length();
00375 }
00376
00377 string Galil::read()
00378 {
00379   ec(GTimeout(d->g, (short)timeout_ms));
00380   GSize bytes_read;
00381   ec(GRead(d->g, d->tbuf, sizeof(d->tbuf), &bytes_read));
00382   ec(GTimeout(d->g, -1));
00383   return string(d->tbuf, bytes_read); //data is not null-terminatated, so construct string "from
      buffer"
00384 }
00385
```

10.191 gcl_galil.h File Reference

```
#include "gclibo.h"
#include <unordered_map>
#include <sstream>
#include <iomanip>
#include <algorithm>
#include "Galil.h"
```

Data Structures

- struct Source
- class GalilPrivate

Macros

- #define MAKEDLL
- #define TRAFFICBUF 4096

Functions

- void ec (GReturn rc)

10.191.1 Macro Definition Documentation

10.191.1.1 MAKEDLL

```
#define MAKEDLL
```

Definition at line 11 of file gcl_galil.h.

10.191.1.2 TRAFFICBUF

```
#define TRAFFICBUF 4096
```

Definition at line 13 of file gcl_galil.h.

10.191.2 Function Documentation

10.191.2.1 ec()

```
void ec (
            GReturn rc )
```

Definition at line 8 of file gcl_galil.cpp.

10.192 gcl_galil.h

Go to the documentation of this file.

```
00001
00003 #ifndef I_34E90AC4_EF44_4CB5_AF47_B9D334EE5FB7
00004 #define I_34E90AC4_EF44_4CB5_AF47_B9D334EE5FB7
00005
00006 #include "gclibo.h" //gclib and gclibo
00007 #include <unordered_map> //used for data record features
00008 #include <sstream> //format source keys, string stream
00009 #include <iomanip> //format source keys
00010 #include <algorithm> //sort sources
00011 #define MAKEDLL //Galil.h requires this for proper export declarations
00012 #include "Galil.h" //gcl header
00013 #define TRAFFICBUF 4096 //Memory used for buffering controller traffic. If program/array upload is
      getting truncated, make this larger.
00014
00015
00016 struct Source //each data record source key (e.g. "_RPA") maps to one of these, each of which
      describes the position and width of the variable within the binary data record
00017 {
00018   int byte; //byte offset within binary data record
00019   std::string type; //"SB", "UB", "SW", "UW", "SL", "UL".  Specifies width within binary data record
      and signed/unsigned.
00020   int bit; //-1 if not bit field (e.g. RPA).  >= 0 if bit field (e.g. _MOA)
00021   std::string units; //e.g. "counts"
00022   std::string description; //e.g. "analog input 1"
00023   double scale; //e.g. 32768, scale factor:  most sources are 1 except TV, TT, @AN, @AO etc.
00024   double offset; //needed for analog inputs and outputs
00025
00026   Source(int byte = 0, std::string type = "Ux", int bit = -1, std::string units = "", std::string
      description = "", double scale = 1, double offset = 0) :
00027     byte(byte), type(type), bit(bit), units(units), description(description), scale(scale),
      offset(offset)
00028   { /*ctor just initializes values*/ }
00029 };
00030
00031 // Use the gcl private class name to hold useful data for the wrapper
00032 class GalilPrivate
00033 {
00034 public:
00035   GalilPrivate(Galil* galil_ptr, GCon gclib_handle)
00036     : q(galil_ptr), g(gclib_handle)
00037   { /*ctor just initializes values*/ }
00038
00039   GCon g; //gclib handle for this class
00040   char tbuf[TRAFFICBUF]; //traffic buffer shared by member functions of Galil
00041   std::unordered_map<std::string, Source> map; //data structure for data record calls
00042   void InitializeDataRecord(); //called in Galil() ctor to create the data record map
00043
00044 private:
00045   Galil* q; //pointer to the Galil class so we can call Command, etc.
00046
00047   //functions for initializing the data record for various products
00048   void Init1806(int axes);
00049   void Init1800(int axes, bool dmc1802);
00050   void Init4000(int axes); //41x3 and 50000 too
00051   void Init30010(bool dmc31010);
00052   void Init2103(int axes);
00053   void InitRio(bool rio3); //rio3 indicates if this connection is to an RIO-47300 which has a
      different length than other RIOs
00054
00055   void InitRio3_24Ex(); //extended I/O option for RIO-47300
00056   void InitRioSer(bool rio3); //Serial encoder (quad/SSI/BISS) option for RIO
00057
00058   //Helper/convenience functions
00059   std::string ax(std::string prefix, int axis, std::string suffix); // Creates an axis string from a
      prefix, axis number (0-7), and suffix.
00060   void input_bits(int byte, int num); //Creates 8 digital input sources from an offset and a starting
      input number.
00061   void output_bits(int byte, int num); //Creates 8 digital ouput sources from an offset and a starting
      output number.
00062   void aq_analog(int byte, int input_num); //polls AQ for input_num and creates an analog input source
      with the proper AQ scalar
00063   void dq_analog(int byte, int input_num); //polls DQ for input_num and creates an analog output
      source with the proper DQ scalar
00064 };
00065
00066 void ec(GReturn rc); // Error Check. Takes a gclib error code and throws a GCL-like error string if
      there's a gclib error
00067
00068
00069
00070 #endif //I_34E90AC4_EF44_4CB5_AF47_B9D334EE5FB7
```

10.193 gcl_simple.cpp File Reference

```
#include "Galil.h"
#include <iostream>
```

Functions

- int main ()

10.193.1 Function Documentation

10.193.1.1 main()

```
int main ( )
```

Definition at line 11 of file gcl_simple.cpp.

10.194 gcl_simple.cpp

Go to the documentation of this file.

```
00001
00004 //g++ test.cpp -lgalil -lgclib -lgclibo -o test
00005
00006 #include "Galil.h"
00007 #include <iostream>
00008
00009 using namespace std;
00010
00011 int main()
00012 {
00013   try
00014   {
00015     cout « Galil::libraryVersion() « '\n';
00016     Galil g("10.1.3.169");
00017     cout « g.connection() « '\n';
00018     return 0;
00019   }
00020   catch(string s)
00021   {
00022     cout « s « endl;
00023     return 1;
00024   }
00025 }
```

10.195 main.cpp File Reference

```
#include "Galil.h"
#include <iostream>
#include <iomanip>
#include <Windows.h>
```

Functions

- int run (int argc, char *argv[])
- int main (int argc, char *argv[])

10.195.1 Function Documentation

10.195.1.1 main()

```
int main (
            int argc,
            char * argv[] )
```

Definition at line 109 of file main.cpp.

10.195.1.2 run()

```
int run (
            int argc,
            char * argv[] )
```

Definition at line 10 of file main.cpp.

10.196 main.cpp

Go to the documentation of this file.

```
00001
00004 #include "Galil.h"
00005 #include <iostream>
00006 #include <iomanip>
00007 #include <Windows.h> //Sleep
00008 using namespace std;
00009
00010 int run(int argc, char *argv[])
00011 {
00012   try
00013   {
00014     cout « Galil::libraryVersion() « '\n';
00015
00016     for (string s : Galil::addresses())
00017       cout « s « '\n';
00018
00019     //Galil g("GALILPCI1");
00020     Galil g("192.168.0.100");
00021
00022     try
00023     {
00024       g.command("xx"); //will throw string
00025     }
00026     catch (string s)
00027     {
00028       cout « s « endl;
00029     }
00030
00031     cout « g.connection() « '\n';
00032     cout « "MGTIME: [" « g.command("MG TIME") « "]\n"; //brackets to visually check trimming
00033     //g.command("UI 0");
00034     //cout « "Interrupt: " « g.interrupt() « '\n'; //30000 Bug 2047 - UDP interrupt payload byte count
      incorrect
00035     g.programDownload("WT 600\rMG \"Message from DMC code\", TIME\rEN");
00036     cout « '\n' « g.programUpload() « '\n' « '\n';
00037     g.command("XQ");
00038
00039     Sleep(1000);
00040     cout « g.message(700);
00041
00042
00043     vector<double> down;
00044     down.push_back(3.1415);
00045     down.push_back(2.7183);
00046     down.push_back(1.6180);
00047     down.push_back(42);
00048     g.arrayDownload(down, "mynums");
00049     g.arrayUploadFile(); //uses the default arrays.csv
00050     g.command("DA *[]");
00051     g.arrayDownloadFile(); //uses the default arrays.csv
00052     vector<double> up = g.arrayUpload("mynums");
00053     for (auto d : up) cout « d « " ";
00054     cout « '\n';
00055
00056     //
      g.firmwareDownloadFile("\\\\rd3\\applications2\\firmware\\hex2\\4000.dmc\\dmc-4000-r12b-bck.hex");
00057     // cout « g.connection() « '\n';
00058     // g.firmwareDownloadFile("\\\\rd3\\applications2\\firmware\\hex2\\4000.dmc\\dmc-4000-r12b.hex");
00059     // cout « g.connection() « '\n';
00060
00061     g.write("MG TIME \r");
00062     cout « g.read();
00063     cout « "\n\n";
00064
00065     //Data record
00066
00067
00068     vector<string> sources = g.sources();
00069     vector<char> r = g.record("QR");
00070     for (auto s : sources)
00071     {
00072       cout « left « setw(10)
```

```
00073         « g.sourceValue(r, s)
00074         « setw(10) « s
00075
00076         « g.source("Description", s)
00077         « ", " « g.source("Units", s)
00078         « ", S" « g.source("Scale", s)
00079         « " O" « g.source("Offset", s)
00080         « '\n';
00081     }
00082     cout « "\n\n";
00083
00084     //modify source properties
00085     /*g.setSource("Units", "TIME", "seconds");
00086     g.setSource("Description", "TIME", "Seconds counter");
00087     g.setSource("Scale", "TIME", "1024");
00088
00089     cout « "TIME" « ""
00090     « ", " « g.source("Description", "TIME")
00091     « ":" « g.sourceValue(g.record(), "TIME")
00092     « " " « g.source("Units", "TIME")
00093     « " S:" « g.source("Scale", "TIME")
00094     « " O:" « g.source("Offset", "TIME")
00095     « '\n';*/
00096
00097
00098     return 0;
00099   }
00100   catch (string s)
00101   {
00102     cout « s;
00103     return 1;
00104   }
00105
00106 }
00107
00108
00109 int main(int argc, char *argv[])
00110 {
00111   int rc = run(argc, argv);
00112 #ifdef _DEBUG
00113   cout « "\n\nPress enter to exit...\n";
00114   getchar();
00115 #endif
00116   return rc;
00117 }
```

10.197 GclibJava.java File Reference

```
import java.nio.charset.Charset;
```

Data Structures

- class GclibJava
- interface **GclibJava.Gclib**
- interface **GclibJava.Gclibo**

Packages

- package gclibjava

10.198 GclibJava.java

Go to the documentation of this file.

```
00001
00018 package gclibjava;
00019
00020 import java.nio.charset.Charset; //Charset for string conversions
00021
00022 //JNA imports
00023 import com.sun.jna.Library;
00024 import com.sun.jna.Native;
00025 import com.sun.jna.Pointer; //g
00026 import com.sun.jna.ptr.PointerByReference; //for GCon* in GOpen()
00027 import com.sun.jna.ptr.IntByReference; //for GSize* in GCommand()
00028 import com.sun.jna.ptr.ByteByReference; //for GStatus* in GInterrupt()
```

```
00029 import java.util.ArrayList;
00030 import java.util.List; //List<Double>
00031
00036 public class GclibJava {
00037
00038     Pointer gclibHandle; //handle for gclib's connection
00039     Boolean connected = false;  //we use a bool to indicate connection status
00040     byte[] trafficBuffer = new byte[524288]; //Most reads/writes to Galil hardware are small. This
      size will hold the largest array or program upload/download possible.
00041
00045     public GclibJava()
00046     {
00047         System.setProperty("jna.library.path", "C:\\Program Files (x86)\\Galil\\gclib\\dll\\x64");
00048     }
00049
00056     @Override
00057     protected void finalize() throws Throwable
00058     {
00059         try {
00060             if (connected)
00061                 GClose();
00062         } finally {
00063             super.finalize();
00064         }
00065     }
00066
00067     // -------------------------------------------------------------------------
00068     // JNA for gclib
00069     // -------------------------------------------------------------------------
00074     interface Gclib extends Library {
00075         Gclib INSTANCE = (Gclib)
00076                 Native.loadLibrary("gclib",
00077                         Gclib.class);
00078         /*
00079         Limit calls to one at a time
00080         Warning: gclibo library calls gclib. Therefore, calls to Gclib and
00081         Gclibo interfaces should not be concurrent.
00082         */
00083         Gclib SYNC_INSTANCE = (Gclib)
00084                 Native.synchronizedLibrary(INSTANCE);
00085
00086         int GArrayDownload(Pointer g, String arrayName, int first, int last, String buffer);
00087         int GArrayUpload(Pointer g, String arrayName, int first, int last, int delim, byte[] response,
      int len);
00088         int GCommand(Pointer g, String command, byte[] response, int len, IntByReference
      bytesReturned);
00089         int GClose(Pointer g);
00090         int GFirmwareDownload(Pointer g, String filePath);
00091         int GInterrupt(Pointer g, ByteByReference statusByte);
00092         int GMessage(Pointer g, byte[] response, int len);
00093         int GOpen(String address, PointerByReference g);
00094         int GProgramDownload(Pointer g, String program, String preprocessor);
00095         int GProgramUpload(Pointer g, byte[] response, int len);
00096     }
00097
00098     // -------------------------------------------------------------------------
00099     // gclib functions
00100     // -------------------------------------------------------------------------
00101
00111     public void GArrayDownload(String arrayName, List<Double> data) throws GclibJavaException
00112     {
00113         String buf = new String();
00114         buf = data.stream().map((d) -> d.toString() + ",").reduce(buf, String::concat);
00115
00116         ec(Gclib.SYNC_INSTANCE.GArrayDownload(gclibHandle, arrayName, -1, -1,
00117                 buf.substring(0, buf.length() - 1)));//remove trailing comma
00118     }
00119
00129     public List<Double> GArrayUpload(String arrayName) throws GclibJavaException
00130     {
00131         ec(Gclib.SYNC_INSTANCE.GArrayUpload(gclibHandle, arrayName, -1, -1, 1, trafficBuffer,
      trafficBuffer.length));
00132         String[] elements = cstringToString(trafficBuffer).split(", ");
00133         List<Double> doubleList = new ArrayList();
00134         for (String s : elements)
00135         {
00136             try
00137             {
00138                 doubleList.add(Double.parseDouble(s));
00139             }
00140             catch (NumberFormatException e)
00141             {
00142                 throw new GclibJavaException( -10002, e.getMessage()); //G_BAD_VALUE_RANGE
00143             }
00144         }
00145         return doubleList;
00146     }
```

```
00147
00151     public void GClose()
00152     {
00153         Gclib.SYNC_INSTANCE.GClose(gclibHandle);
00154         connected = false;
00155     }
00156
00167     public String GCommand(String command) throws GclibJavaException
00168     {
00169         IntByReference ptrInt = new IntByReference(); //for bytes read
00170         ec(Gclib.SYNC_INSTANCE.GCommand(gclibHandle, command, trafficBuffer, trafficBuffer.length,
      ptrInt));
00171         String response = cstringToString(trafficBuffer);
00172
00173         int index = response.lastIndexOf("\r\n:");
00174         if (index > 0)
00175             response = response.substring(0, index); //trim trailing crlf:
00176
00177         return response;
00178     }
00179
00190     public void GFirmwareDownload(String filePath) throws GclibJavaException
00191     {
00192         ec(Gclib.SYNC_INSTANCE.GFirmwareDownload(gclibHandle, filePath));
00193     }
00194
00207     public byte GInterrupt() throws GclibJavaException
00208     {
00209         ByteByReference statusByte = new ByteByReference();
00210         ec(Gclib.SYNC_INSTANCE.GInterrupt(gclibHandle, statusByte));
00211         return statusByte.getValue();
00212     }
00213
00235     public String GMessage() throws GclibJavaException
00236     {
00237         ec(Gclib.SYNC_INSTANCE.GMessage(gclibHandle, trafficBuffer, trafficBuffer.length));
00238         return cstringToString(trafficBuffer);
00239     }
00240
00250     public void GOpen(String address) throws GclibJavaException
00251     {
00252         if (connected)
00253             GClose();
00254
00255         PointerByReference ptrRef = new PointerByReference();
00256         ec(Gclib.SYNC_INSTANCE.GOpen(address, ptrRef));
00257         gclibHandle = ptrRef.getValue();
00258         connected = true;
00259     }
00260
00270     public void GProgramDownload(String program, String preprocessor) throws GclibJavaException
00271     {
00272         ec(Gclib.SYNC_INSTANCE.GProgramDownload(gclibHandle, program, preprocessor));
00273     }
00274
00282     public void GProgramDownload(String program) throws GclibJavaException
00283     {
00284         GProgramDownload(program, "");
00285     }
00286
00294     public String GProgramUpload() throws GclibJavaException
00295     {
00296         ec(Gclib.SYNC_INSTANCE.GProgramUpload(gclibHandle, trafficBuffer, trafficBuffer.length));
00297         return cstringToString(trafficBuffer);
00298     }
00299
00300     // ---------------------------------------------------------------------------
00301     // JNA for gclibo
00302     // ---------------------------------------------------------------------------
00303
00308     interface Gclibo extends Library {
00309         Gclibo INSTANCE = (Gclibo)
00310             Native.loadLibrary("gclibo",
00311                                Gclibo.class);
00312        /*
00313         Limit calls to one at a time
00314         Warning: gclibo library calls gclib. Therefore, calls to Gclib and
00315         Gclibo interfaces should not be concurrent.
00316         */
00317         Gclibo SYNC_INSTANCE = (Gclibo)
00318                 Native.synchronizedLibrary(INSTANCE);
00319
00320         int GAddresses(byte[] response, int len);
00321         int GArrayDownloadFile(Pointer g, String filePath);
00322         int GArrayUploadFile(Pointer g, String filePath, String names);
00323         int GAssign(String ip, String mac);
00324         void GError(int rc, byte[] response, int len);
```

```
00325         int GInfo(Pointer g, byte[] response, int len);
00326         int GIpRequests(byte[] response, int len);
00327         int GProgramDownloadFile(Pointer g, String filePath, String preprocessor);
00328         int GProgramUploadFile(Pointer g, String filePath);
00329         void GSleep(int timeout_ms);
00330         int GTimeout(Pointer g, short timeout_ms);
00331         int GVersion(byte[] response, int len);
00332         int GSetServer(String server_name);
00333         int GServerStatus(byte[] response, int len);
00334         int GListServers(byte[] response, int len);
00335         int GPublishServer(String server_name, int publish, int save);
00336         int GRemoteConnections(byte[] response, int len);
00337     }
00338
00339     // ------------------------------------------------------------------------
00340     // gclibo functions
00341     // ------------------------------------------------------------------------
00342
00359     public String GAddresses() throws GclibJavaException
00360     {
00361         ec(Gclibo.SYNC_INSTANCE.GAddresses(trafficBuffer, trafficBuffer.length));
00362         return cstringToString(trafficBuffer);
00363     }
00364
00373     public void GArrayDownloadFile(String filePath) throws GclibJavaException
00374     {
00375         ec(Gclibo.SYNC_INSTANCE.GArrayDownloadFile(gclibHandle, filePath));
00376     }
00377
00391     public void GArrayUploadFile(String filePath, String names) throws GclibJavaException
00392     {
00393         ec(Gclibo.SYNC_INSTANCE.GArrayUploadFile(gclibHandle, filePath, names));
00394     }
00395
00404     public void GArrayUploadFile(String filePath) throws GclibJavaException
00405     {
00406         GArrayUploadFile(filePath, "");
00407     }
00408
00419     public void GAssign(String ipAddress, String macAddress) throws GclibJavaException
00420     {
00421         ec(Gclibo.SYNC_INSTANCE.GAssign(ipAddress, macAddress));
00422     }
00423
00432     public String GInfo() throws GclibJavaException
00433     {
00434         ec(Gclibo.SYNC_INSTANCE.GInfo(gclibHandle, trafficBuffer, trafficBuffer.length));
00435         return cstringToString(trafficBuffer);
00436     }
00437
00449     public String GIpRequests() throws GclibJavaException
00450     {
00451         ec(Gclibo.SYNC_INSTANCE.GIpRequests(trafficBuffer, trafficBuffer.length));
00452         return cstringToString(trafficBuffer);
00453     }
00454
00464     public void GProgramDownloadFile(String filePath, String preprocessor) throws GclibJavaException
00465     {
00466         ec(Gclibo.SYNC_INSTANCE.GProgramDownloadFile(gclibHandle, filePath, preprocessor));
00467     }
00468
00476     public void GProgramDownloadFile(String filePath) throws GclibJavaException
00477     {
00478         GProgramDownloadFile(filePath, "");
00479     }
00480
00489     public void GProgramUploadFile(String filePath) throws GclibJavaException
00490     {
00491         ec(Gclibo.SYNC_INSTANCE.GProgramUploadFile(gclibHandle, filePath));
00492     }
00493
00502     public void GSleep(int timeout_ms)
00503     {
00504         Gclibo.SYNC_INSTANCE.GSleep(timeout_ms);
00505     }
00506
00515     public void GTimeout(short timeout_ms) throws GclibJavaException
00516     {
00517         ec(Gclibo.SYNC_INSTANCE.GTimeout(gclibHandle, timeout_ms));
00518     }
00519
00529     public String GVersion() throws GclibJavaException
00530     {
00531         ec(Gclibo.SYNC_INSTANCE.GVersion(trafficBuffer, trafficBuffer.length));
00532         return cstringToString(trafficBuffer);
00533     }
00534
```

```
00542     public void GSetServer(String server_name) throws GclibJavaException
00543     {
00544         ec(Gclibo.SYNC_INSTANCE.GSetServer(server_name));
00545     }
00546 
00555     public String GServerStatus() throws GclibJavaException
00556     {
00557         ec(Gclibo.SYNC_INSTANCE.GServerStatus(trafficBuffer, trafficBuffer.length));
00558         return cstringToString(trafficBuffer);
00559     }
00560 
00568     public String GListServers() throws GclibJavaException
00569     {
00570         ec(Gclibo.SYNC_INSTANCE.GListServers(trafficBuffer, trafficBuffer.length));
00571         return cstringToString(trafficBuffer);
00572     }
00573 
00583     public void GPublishServer(String server_name, int publish, int save) throws GclibJavaException
00584     {
00585         ec(Gclibo.SYNC_INSTANCE.GPublishServer(server_name, publish, save));
00586     }
00587 
00595     public String GRemoteConnections() throws GclibJavaException
00596     {
00597         ec(Gclibo.SYNC_INSTANCE.GRemoteConnections(trafficBuffer, trafficBuffer.length));
00598         return cstringToString(trafficBuffer);
00599     }
00600 
00601     // --------------------------------------------------------------------------
00602     // Helper functions
00603     // --------------------------------------------------------------------------
00604 
00605     //convert gclib's C strings to Java strings.
00606     String cstringToString(byte[] cbuf)
00607     {
00608         Charset charset = Charset.forName("UTF-8");
00609         int i;
00610         for (i = 0; i < cbuf.length && cbuf[i] != 0; i++){}//search for gclib's null terminator
00611         return new String(cbuf, 0, i, charset);
00612     }
00613 
00614     //Error checker for gclib return code
00615     void ec(int returnCode) throws GclibJavaException
00616     {
00617         if (returnCode != 0)
00618         {
00619             //lookup human-readable string
00620             Gclibo.SYNC_INSTANCE.GError(returnCode, trafficBuffer, trafficBuffer.length);
00621             throw new GclibJavaException(returnCode, cstringToString(trafficBuffer));
00622         }
00623     }
00624 
00625 }
```

10.199 GclibJavaException.java File Reference

Data Structures

- class GclibJavaException

Packages

- package gclibjava

10.200 GclibJavaException.java

Go to the documentation of this file.

```
00001 
00004 package gclibjava;
00005 
00006 public class GclibJavaException extends Exception {
00007     int myErrorCode = 0;
00008     public GclibJavaException(int errorCode, String message) {
00009         super(message);
00010         myErrorCode = errorCode;
00011     }
00012     public int getErrorCode()
00013     {
00014         return myErrorCode;
```

```
00015     }
00016 }
```

10.201 gclib.py File Reference

Data Structures

- class GclibError
- class py

Functions

- _rc (return_code)

10.201.1 Function Documentation

10.201.1.1 _rc()

```
_rc (
              return_code )  [protected]
```

Checks return codes from gclib and raises a python error if result is exceptional.

Definition at line 130 of file gclib.py.

10.202 gclib.py

Go to the documentation of this file.

```
00001 "'! \file gclib.py
00002 "'
00003 """
00004 Python wrapper for Galil gclib.
00005 Contact softwaresupport@galil.com with questions, comments, and suggestions.
00006 """
00007
00011 import platform #for distinguishing 'Windows', 'Linux', 'Darwin'
00012 from ctypes import *
00013
00014 if platform.system() == 'Windows':
00015     if '64 bit' in platform.python_compiler():
00016         WinDLL(r'C:\Program Files (x86)\Galil\gclib\dll\x64\libcrypto-1_1-x64.dll')
00017         WinDLL(r'C:\Program Files (x86)\Galil\gclib\dll\x64\libssl-1_1-x64.dll')
00018         _gclib_path = r'C:\Program Files (x86)\Galil\gclib\dll\x64\gclib.dll'
00019         _gclibo_path = r'C:\Program Files (x86)\Galil\gclib\dll\x64\gclibo.dll'
00020         _gclib = WinDLL(_gclib_path)
00021         _gclibo = WinDLL(_gclibo_path)
00022     else:
00023         WinDLL(r'C:\Program Files (x86)\Galil\gclib\dll\x86\libcrypto-1_1.dll')
00024         WinDLL(r'C:\Program Files (x86)\Galil\gclib\dll\x86\libssl-1_1.dll')
00025         _gclib_path = r'C:\Program Files (x86)\Galil\gclib\dll\x86\gclib.dll'
00026         _gclibo_path = r'C:\Program Files (x86)\Galil\gclib\dll\x86\gclibo.dll'
00027         _gclib = WinDLL(_gclib_path)
00028         _gclibo = WinDLL(_gclibo_path)
00029         #Reassign symbol name, Python doesn't like @ in function names
00030         #gclib calls
00031         setattr(_gclib, 'GArrayDownload', getattr(_gclib, '_GArrayDownload@20'))
00032         setattr(_gclib, 'GArrayUpload', getattr(_gclib, '_GArrayUpload@28'))
00033         setattr(_gclib, 'GClose', getattr(_gclib, '_GClose@4'))
00034         setattr(_gclib, 'GCommand', getattr(_gclib, '_GCommand@20'))
00035         setattr(_gclib, 'GFirmwareDownload', getattr(_gclib, '_GFirmwareDownload@8'))
00036         setattr(_gclib, 'GInterrupt', getattr(_gclib, '_GInterrupt@8'))
00037         setattr(_gclib, 'GMessage', getattr(_gclib, '_GMessage@12'))
00038         setattr(_gclib, 'GOpen', getattr(_gclib, '_GOpen@8'))
00039         setattr(_gclib, 'GProgramDownload', getattr(_gclib, '_GProgramDownload@12'))
00040         setattr(_gclib, 'GProgramUpload', getattr(_gclib, '_GProgramUpload@12'))
00041         #gclibo calls (open source component/convenience functions)
00042         setattr(_gclibo, 'GAddresses', getattr(_gclibo, '_GAddresses@8'))
00043         setattr(_gclibo, 'GArrayDownloadFile', getattr(_gclibo, '_GArrayDownloadFile@8'))
00044         setattr(_gclibo, 'GArrayUploadFile', getattr(_gclibo, '_GArrayUploadFile@12'))
00045         setattr(_gclibo, 'GAssign', getattr(_gclibo, '_GAssign@8'))
00046         setattr(_gclibo, 'GError', getattr(_gclibo, '_GError@12'))
00047         setattr(_gclibo, 'GInfo', getattr(_gclibo, '_GInfo@12'))
00048         setattr(_gclibo, 'GIpRequests', getattr(_gclibo, '_GIpRequests@8'))
00049         setattr(_gclibo, 'GMotionComplete', getattr(_gclibo, '_GMotionComplete@8'))
```

```
00050         setattr(_gclibo, 'GProgramDownloadFile', getattr(_gclibo, '_GProgramDownloadFile@12'))
00051         setattr(_gclibo, 'GSleep', getattr(_gclibo, '_GSleep@4'))
00052         setattr(_gclibo, 'GProgramUploadFile', getattr(_gclibo, '_GProgramUploadFile@8'))
00053         setattr(_gclibo, 'GTimeout', getattr(_gclibo, '_GTimeout@8'))
00054         setattr(_gclibo, 'GVersion', getattr(_gclibo, '_GVersion@8'))
00055         setattr(_gclibo, 'GSetupDownloadFile', getattr(_gclibo, '_GSetupDownloadFile@20'))
00056         setattr(_gclibo, 'GServerStatus', getattr(_gclibo, '_GServerStatus@8'))
00057         setattr(_gclibo, 'GSetServer', getattr(_gclibo, '_GSetServer@4'))
00058         setattr(_gclibo, 'GListServers', getattr(_gclibo, '_GListServers@8'))
00059         setattr(_gclibo, 'GPublishServer', getattr(_gclibo, '_GPublishServer@12'))
00060         setattr(_gclibo, 'GRemoteConnections', getattr(_gclibo, '_GRemoteConnections@8'))
00061
00062 elif platform.system() == 'Linux':
00063     cdll.LoadLibrary("libgclib.so.2")
00064     _gclib = CDLL("libgclib.so.2")
00065     cdll.LoadLibrary("libgclibo.so.2")
00066     _gclibo = CDLL("libgclibo.so.2")
00067
00068 elif platform.system() == 'Darwin': #OSX
00069     _gclib_path = '/Applications/gclib/dylib/gclib.0.dylib'
00070     _gclibo_path = '/Applications/gclib/dylib/gclibo.0.dylib'
00071     cdll.LoadLibrary(_gclib_path)
00072     _gclib = CDLL(_gclib_path)
00073     cdll.LoadLibrary(_gclibo_path)
00074     _gclibo = CDLL(_gclibo_path)
00075
00076
00077
00078 # Python "typedefs"
00079 _GReturn = c_int #type for a return code
00080 _GCon = c_void_p #type for a Galil connection handle
00081 _GCon_ptr = POINTER(_GCon) #used for argtypes declaration
00082 _GSize = c_ulong #type for a Galil size variable
00083 _GSize_ptr = POINTER(_GSize) #used for argtypes declaration
00084 _GCStringIn = c_char_p #char*. In C it's const.
00085 _GCStringOut = c_char_p #char*
00086 _GOption = c_int #type for option variables, e.g.    GArrayDownload
00087 _GStatus = c_ubyte #type for interrupt status bytes
00088 _GStatus_ptr = POINTER(_GStatus) #used for argtypes declaration
00089
00090 #Define arguments and result type (if not C int type)
00091 #gclib calls
00092 _gclib.GArrayDownload.argtypes = [_GCon, _GCStringIn, _GOption, _GOption, _GCStringIn]
00093 _gclib.GArrayUpload.argtypes = [_GCon, _GCStringIn, _GOption, _GOption, _GOption, _GCStringOut,
      _GSize]
00094 _gclib.GClose.argtypes = [_GCon]
00095 _gclib.GCommand.argtypes = [_GCon, _GCStringIn, _GCStringOut, _GSize, _GSize_ptr]
00096 _gclib.GFirmwareDownload.argtypes = [_GCon, _GCStringIn]
00097 _gclib.GInterrupt.argtypes = [_GCon, _GStatus_ptr]
00098 _gclib.GMessage.argtypes = [_GCon, _GCStringOut, _GSize]
00099 _gclib.GOpen.argtypes = [_GCStringIn, _GCon_ptr]
00100 _gclib.GProgramDownload.argtypes = [_GCon, _GCStringIn, _GCStringIn]
00101 _gclib.GProgramUpload.argtypes = [_GCon, _GCStringOut, _GSize]
00102 #gclibo calls (open source component/convenience functions)
00103 _gclibo.GAddresses.argtypes = [_GCStringOut, _GSize]
00104 _gclibo.GArrayDownloadFile.argtypes = [_GCon, _GCStringIn]
00105 _gclibo.GArrayUploadFile.argtypes = [_GCon, _GCStringIn, _GCStringIn]
00106 _gclibo.GAssign.argtypes = [_GCStringIn, _GCStringIn]
00107 _gclibo.GError.argtypes = [_GReturn, _GCStringOut, _GSize]
00108 _gclibo.GError.restype    = None
00109 _gclibo.GError.argtypes = [_GCon, _GCStringOut, _GSize]
00110 _gclibo.GIpRequests.argtypes = [_GCStringOut, _GSize]
00111 _gclibo.GMotionComplete.argtypes = [_GCon, _GCStringIn]
00112 _gclibo.GProgramDownloadFile.argtypes = [_GCon, _GCStringIn, _GCStringIn]
00113 _gclibo.GSleep.argtypes = [c_uint]
00114 _gclibo.GSleep.restype    = None
00115 _gclibo.GProgramUploadFile.argtypes = [_GCon, _GCStringIn]
00116 _gclibo.GTimeout.argtypes = [_GCon, c_int]
00117 _gclibo.GVersion.argtypes = [_GCStringOut, _GSize]
00118 _gclibo.GServerStatus.argtypes = [_GCStringOut, _GSize]
00119 _gclibo.GSetServer.argtypes = [_GCStringIn]
00120 _gclibo.GListServers.argtypes = [_GCStringOut, _GSize]
00121 _gclibo.GPublishServer.argtypes = [_GCStringIn, _GOption, _GOption]
00122 _gclibo.GRemoteConnections.argtypes = [_GCStringOut, _GSize]
00123 _gclibo.GSetupDownloadFile.argtypes = [_GCon, _GCStringIn, _GOption, _GCStringOut, _GSize]
00124
00125 #Set up some constants
00126 _enc = "ASCII" #byte encoding for going between python strings and c strings.
00127 _buf_size = 500000 #size of response buffer. Big enough to fit entire 4000 program via UL/LS, or 24000
      elements of array data.
00128 _error_buf = create_string_buffer(128)    #buffer for retrieving error code descriptions.
00129
00130 def _rc(return_code):
00131     """Checks return codes from gclib and raises a python error if result is exceptional."""
00132     if return_code != 0:
00133         _gclibo.GError(return_code, _error_buf, 128) #Get the library's error description
00134         raise GclibError(str(_error_buf.value.decode(_enc)))
```

```
00135     return
00136
00137 class GclibError(Exception):
00138     """Error class for non-zero gclib return codes."""
00139     pass
00140
00141 class py:
00142     """Represents a single Python connection to a Galil Controller or PLC."""
00143
00144     def __init__(self):
00145         """Constructor for the Connection class. Initializes gclib's handle and read buffer."""
00146         self._gcon = _GCon(0) #handle to connection
00147         self._buf = create_string_buffer(_buf_size)
00148         self._timeout = 5000
00149         return
00150
00151     def __del__(self):
00152         """Destructor for the Connection class. Ensures close gets called to release Galil resource
      (Sockets, Kernel Driver, Com Port, etc)."""
00153         self.GClose()
00154         return
00155
00156     def _cc(self):
00157         """Checks if connection is established, throws error if not."""
00158         if self._gcon.value == None:
00159             _rc(-1201) #G_CONNECTION_NOT_ESTABLISHED
00160
00161     def GOpen(self, address):
00162         """
00163         Opens a connection a galil controller.
00164         See the gclib docs for address string formatting.
00165         """
00166         c_address = _GCStringIn(address.encode(_enc))
00167         _rc(_gclib.GOpen(c_address, byref(self._gcon)))
00168         return
00169
00170
00171     def GClose(self):
00172         """
00173         Closes a connection to a Galil Controller.
00174         """
00175         if self._gcon.value != None:
00176             _rc(_gclib.GClose(self._gcon))
00177             self._gcon = _GCon(0)
00178         return
00179
00180
00181     def GCommand(self, command):
00182         """
00183         Performs a command-and-response transaction on the connection.
00184         Trims the response.
00185         """
00186         self._cc()
00187         c_command = _GCStringIn(command.encode(_enc))
00188         _rc(_gclib.GCommand(self._gcon, c_command, self._buf, _buf_size, None))
00189         response = str(self._buf.value.decode(_enc))
00190         return response[:-3].strip() # trim trailing /r/n: and leading space
00191
00192
00193     def GSleep(self, val):
00194         """
00195         Provides a blocking sleep call which can be useful for timing-based chores.
00196         """
00197         _gclibo.GSleep(val)
00198         return
00199
00200
00201     def GVersion(self):
00202         """
00203         Provides the gclib version number. Please include the output of this function on all support
      cases.
00204         """
00205         _rc(_gclibo.GVersion(self._buf, _buf_size))
00206         return "py." + str(self._buf.value.decode(_enc))
00207
00208     def GServerStatus(self):
00209         _rc(_gclibo.GServerStatus(self._buf, _buf_size))
00210         return str(self._buf.value.decode(_enc))
00211
00212     def GSetServer(self, server_name):
00213         c_server_name = _GCStringIn(server_name.encode(_enc))
00214         _rc(_gclibo.GSetServer(c_server_name))
00215         return
00216
00217     def GListServers(self):
00218         _rc(_gclibo.GListServers(self._buf, _buf_size))
00219         return str(self._buf.value.decode(_enc))
```

```
00220
00221     def GPublishServer(self, server_name, publish, save):
00222         c_server_name = _GCStringIn(server_name.encode(_enc))
00223         _rc(_gclibo.GPublishServer(c_server_name, publish, save))
00224         return
00225
00226     def GRemoteConnections(self):
00227         _rc(_gclibo.GRemoteConnections(self._buf, _buf_size))
00228         return str(self._buf.value.decode(_enc))
00229
00230     def GInfo(self):
00231         """
00232         Provides a useful connection string. Please include the output of this function on all support
      cases.
00233         """
00234         _rc(_gclibo.GInfo(self._gcon, self._buf, _buf_size))
00235         return str(self._buf.value.decode(_enc))
00236
00237
00238     def GIpRequests(self):
00239         """
00240         Provides a dictionary of all Galil controllers requesting IP addresses via BOOT-P or DHCP.
00241
00242         Returns a dictionary mapping 'model-serial' --> 'mac address'
00243         e.g. {'DMC4000-783': '00:50:4c:20:03:0f', 'DMC4103-9998': '00:50:4c:38:27:0e'}
00244
00245         Linux/OS X users must be root to use GIpRequests() and have UDP access to bind and listen on
      port 67.
00246         """
00247         _rc(_gclibo.GIpRequests(self._buf, _buf_size)) #get the c string from gclib
00248         ip_req_dict = {}
00249         for line in str(self._buf.value.decode(_enc)).splitlines():
00250             line = line.replace(' ', '') #trim spaces throughout
00251             if (line == ""): continue
00252             fields = line.split(',')
00253             #fields go [model, serial number, mac]
00254             ip_req_dict[fields[0] + '-' + fields[1]] = fields[2] # e.g. DMC4000-783 maps to its MAC
      addr.
00255         return ip_req_dict
00256
00257
00258     def GAssign(self, ip, mac):
00259         """
00260         Assigns IP address over the Ethernet to a controller at a given MAC address.
00261         Linux/OS X users must be root to use GAssign() and have UDP access to send on port 68.
00262         """
00263         c_ip = _GCStringIn(ip.encode(_enc))
00264         c_mac = _GCStringIn(mac.encode(_enc))
00265         _rc(_gclibo.GAssign(c_ip, c_mac))
00266         return
00267
00268
00269     def GAddresses(self):
00270         """
00271         Provides a dictionary of all available connection addresses.
00272
00273         Returns a dictionary mapping 'address' -> 'revision reports', where possible
00274         e.g. {}
00275         """
00276         _rc(_gclibo.GAddresses(self._buf, _buf_size))
00277         addr_dict = {}
00278         for line in str(self._buf.value.decode(_enc)).splitlines():
00279             fields = line.split(',')
00280             if len(fields) >= 2:
00281                 addr_dict[fields[0]] = fields[1]
00282             else:
00283                 addr_dict[fields[0]] = ''
00284
00285         return addr_dict
00286
00287
00288     def GProgramDownload(self, program, preprocessor=""):
00289         """
00290         Downloads a program to the controller's program buffer.
00291         See the gclib docs for preprocessor options.
00292         """
00293         self._cc()
00294         c_prog = _GCStringIn(program.encode(_enc))
00295         c_pre = _GCStringIn(preprocessor.encode(_enc))
00296         _rc(_gclib.GProgramDownload(self._gcon, c_prog, c_pre))
00297         return
00298
00299
00300     def GProgramUpload(self):
00301         """
00302         Uploads a program from the controller's program buffer.
00303         """
```

```
00304         self._cc()
00305         _rc(_gclib.GProgramUpload(self._gcon, self._buf, _buf_size))
00306         return str(self._buf.value.decode(_enc))
00307 
00308 
00309     def GProgramDownloadFile(self, file_path, preprocessor=""):
00310         """
00311         Program download from file.
00312         See the gclib docs for preprocessor options.
00313         """
00314         self._cc()
00315         c_path = _GCStringIn(file_path.encode(_enc))
00316         c_pre = _GCStringIn(preprocessor.encode(_enc))
00317         _rc(_gclibo.GProgramDownloadFile(self._gcon, c_path, c_pre))
00318         return
00319 
00320     def GProgramUploadFile(self, file_path):
00321         """
00322         Program upload to file.
00323         """
00324         self._cc()
00325         c_path = _GCStringIn(file_path.encode(_enc))
00326         _rc(_gclibo.GProgramUploadFile(self._gcon, c_path))
00327         return
00328 
00329     def GArrayDownload(self, name, first, last, array_data):
00330         """
00331         Downloads array data to a pre-dimensioned array in the controller's array table.
00332         array_data should be a list of values (e.g. int or float)
00333         """
00334         self._cc()
00335         c_name = _GCStringIn(name.encode(_enc))
00336         array_string = ""
00337         for val in array_data:
00338             array_string += str(val) + ","
00339         c_data = _GCStringIn(array_string[:-1].encode(_enc)) #trim trailing command
00340         _rc(_gclib.GArrayDownload(self._gcon, c_name, first, last, c_data))
00341         return
00342 
00343 
00344     def GArrayUploadFile(self, file_path, names = []):
00345         """
00346         Uploads the entire controller array table or a subset and saves the data as a csv file
      specified by file_path.
00347         names is optional and should be a list of array names on the controller.
00348         """
00349         self._cc()
00350         c_path = _GCStringIn(file_path.encode(_enc))
00351         names_string = ”
00352         c_names = _GCStringIn(”.encode(_enc)) #in case empty list provided
00353         for name in names:
00354             names_string += name + ’ ’
00355 
00356         c_names = _GCStringIn(names_string[:-1].encode(_enc)) #trim trailing space
00357         _rc(_gclibo.GArrayUploadFile(self._gcon, c_path, c_names))
00358         return
00359 
00360 
00361     def GArrayDownloadFile(self, file_path):
00362         """
00363         Downloads a csv file containing array data at file_path.
00364         """
00365         self._cc()
00366         c_path = _GCStringIn(file_path.encode(_enc))
00367         _rc(_gclibo.GArrayDownloadFile(self._gcon, c_path))
00368         return
00369 
00370 
00371     def GArrayUpload(self, name, first, last):
00372         """
00373         Uploads array data from the controller's array table.
00374         """
00375         self._cc()
00376         c_name = _GCStringIn(name.encode(_enc))
00377         _rc(_gclib.GArrayUpload(self._gcon, c_name, first, last, 1, self._buf, _buf_size)) #1 is comma
      delimiter
00378         string_list = str(self._buf.value.decode(_enc)).split(’,’)
00379         float_list = []
00380         for s in string_list:
00381             float_list.append(float(s))
00382         return float_list
00383 
00384 
00385     def GTimeout(self, timeout):
00386         """
00387         Set the library timeout. Set to -1 to use the intitial library timeout, as specified in GOpen.
00388         """
```

```
00389         self._cc()
00390         _rc(_gclibo.GTimeout(self._gcon, timeout))
00391         self._timeout = timeout
00392         return
00393 
00394 
00395     @property
00396     def timeout(self):
00397         """
00398         Convenience property read access to timeout value. If -1, gclib uses the initial library timeout, as specified in GOpen.
00399         """
00400         return self._timeout
00401 
00402     @timeout.setter
00403     def timeout(self, timeout):
00404         """
00405         Convenience property write access to timeout value. Set to -1 to use the initial library timeout, as specified in GOpen.
00406         """
00407         self.GTimeout(timeout)
00408         return
00409 
00410 
00411     def GFirmwareDownload(self, file_path):
00412         """
00413         Upgrade firmware.
00414         """
00415         self._cc()
00416         c_path = _GCStringIn(file_path.encode(_enc))
00417         _rc(_gclib.GFirmwareDownload(self._gcon, c_path))
00418         return
00419 
00420 
00421     def GMessage(self):
00422         """
00423         Provides access to unsolicited messages from the controller.
00424         """
00425         self._cc()
00426         _rc(_gclib.GMessage(self._gcon, self._buf, _buf_size))
00427         return str(self._buf.value.decode(_enc))
00428 
00429 
00430     def GMotionComplete(self, axes):
00431         """
00432         Blocking call that returns once all axes specified have completed their motion.
00433         """
00434         self._cc()
00435         c_axes = _GCStringIn(axes.encode(_enc))
00436         _rc(_gclibo.GMotionComplete(self._gcon, c_axes))
00437         return
00438 
00439     def GInterrupt(self):
00440         """
00441         Provides access to PCI and UDP interrupts from the controller.
00442         """
00443         self._cc()
00444         status = _GStatus(0)
00445         _rc(_gclib.GInterrupt(self._gcon, byref(status)))
00446         return status.value
00447 
00448     def GSetupDownloadFile(self, file_path, options):
00449         """
00450         Downloads specified sectors from a Galil compressed backup (gcb) file to a controller.
00451 
00452         Returns a dictionary with the controller information stored in the gcb file.
00453         If options is specified as 0, an additional "options" key will be in the dictionary indicating the info sectors available in the gcb
00454         """
00455         self._cc()
00456         c_path = _GCStringIn(file_path.encode(_enc))
00457 
00458         rc = _gclibo.GSetupDownloadFile(self._gcon, c_path, options, self._buf, _buf_size)
00459         if (options != 0):
00460             _rc(rc)
00461 
00462         info_dict = {}
00463         for line in str(self._buf.value.decode(_enc)).split("\"\n"):
00464             fields = line.split(',',1)
00465 
00466             if (fields[0] == ""): continue
00467             elif len(fields) >= 2:
00468                 info_dict[fields[0].strip("\"\'")] = fields[1].strip("\"\'")
00469             else:
00470                 info_dict[fields[0].strip("\"\'")] = ''
00471 
00472         if (options == 0):
```

```
00473             info_dict["options"] = rc
00474
00475         return info_dict
```

10.203 setup.py File Reference

Variables

- name
- version
- description
- author
- author_email
- url
- py_modules

10.203.1 Variable Documentation

10.203.1.1 author

`author`
Definition at line 9 of file setup.py.

10.203.1.2 author_email

`author_email`
Definition at line 10 of file setup.py.

10.203.1.3 description

`description`
Definition at line 8 of file setup.py.

10.203.1.4 name

`name`
Definition at line 6 of file setup.py.

10.203.1.5 py_modules

`py_modules`
Definition at line 12 of file setup.py.

10.203.1.6 url

`url`
Definition at line 11 of file setup.py.

10.203.1.7 version

`version`
Definition at line 7 of file setup.py.

10.204 setup.py

Go to the documentation of this file.

```
00001 "'! \file setup.py
00002 "'
00003 #Setup script for gclib python wrapper
00004 # run 'python setup.py install' at console
```

```
00005 from setuptools import setup
00006 setup(name='gclib',
00007 version='1.0',
00008 description='Python wrapper for Galil gclib',
00009 author='Galil Motion Control',
00010 author_email='softwaresupport@galil.com',
00011 url='http://www.galil.com',
00012 py_modules=['gclib'],
00013 )
```

10.205 gclib.vb File Reference

Data Structures

- class Gclib

  Provides a class that binds to gclib's unmanaged dll. Wraps each call and provides a more user-friendly interface for use in Visual Basic.
- interface Gclib.GDataRecord
- struct Gclib.GDataRecord4000
- struct Gclib.GDataRecord52000
- struct Gclib.GDataRecord1806
- struct Gclib.GDataRecord2103
- struct Gclib.GDataRecord1802
- struct Gclib.GDataRecord30000
- struct Gclib.GDataRecord47000_ENC
- struct Gclib.GDataRecord47300_ENC
- struct Gclib.GDataRecord47300_24EX
- struct Gclib.GDataRecord47162

Typedefs

- using UB = System.Byte
- using UW = System.UInt16
- using SW = System.Int16
- using SL = System.Int32
- using UL = System.UInt32
- using GReturn = System.Int32
- using GCon = System.IntPtr
- using GSize = System.UInt32
- using GCStringOut = System.Text.StringBuilder
- using GCStringIn = System.String
- using GBufOut = System.Text.StringBuilder
- using GBufIn = System.String
- using GStatus = System.Byte

Variables

- using System
- Module LibraryPath
- const string GclibDllPath_ = "C:\Program Files[] \Galil\gclib\dll\x86\gclib.dll"
- const string GcliboDllPath_ = "C:\Program Files[] \Galil\gclib\dll\x86\gclibo.dll"

10.205.1 Typedef Documentation

10.205.1.1 SL

```
using SL = System.Int32
```

Definition at line 10 of file gclib.vb.

10.205.1.2 SW

```
using SW = System.Int16
```
Definition at line 9 of file gclib.vb.

10.205.1.3 UB

```
using UB = System.Byte
```
Definition at line 7 of file gclib.vb.

10.205.1.4 UL

```
using UL = System.UInt32
```
Definition at line 11 of file gclib.vb.

10.205.1.5 UW

```
using UW = System.UInt16
```
Definition at line 8 of file gclib.vb.

10.205.2 Variable Documentation

10.205.2.1 System

```
using System
```
Definition at line 3 of file gclib.vb.

10.206 gclib.vb

Go to the documentation of this file.

```
00001 Imports System
00002 Imports System.Text 'StringBuilder
00003 Imports System.Runtime.InteropServices 'DLL import
00004 Imports System.IO 'File.Exists
00005
00006 Imports UB = System.Byte
00007 Imports UW = System.UInt16
00008 Imports SW = System.Int16
00009 Imports SL = System.Int32
00010 Imports UL = System.UInt32
00011
00012 #If PLATFORM = "x86" Then
00013 Imports GReturn = System.Int32
00014 Imports GCon = System.IntPtr
00015 Imports GSize = System.UInt32
00016 Imports GOption = System.Int32
00017 Imports GCStringOut = System.Text.StringBuilder
00018 Imports GCStringIn = System.String
00019 Imports GBufOut = System.Text.StringBuilder
00020 Imports GBufIn = System.String
00021 Imports GStatus = System.Byte
00022 ' IMPORTANT! Be sure that the paths below are correct
00023 Public Module LibraryPath
00024     Public Const GclibDllPath_ As String = "C:\Program Files (x86)\Galil\gclib\dll\x86\gclib.dll"
00025     Public Const GcliboDllPath_ As String = "C:\Program Files (x86)\Galil\gclib\dll\x86\gclibo.dll"
00026 End Module
00027
00028 #ElseIf PLATFORM = "x64" Then
00029 Imports GReturn = System.Int32
00030 Imports GCon = System.IntPtr
00031 Imports GSize = System.UInt32
00032 Imports GOption = System.Int32
00033 Imports GCStringOut = System.Text.StringBuilder
00034 Imports GCStringIn = System.String
00035 Imports GBufOut = System.Text.StringBuilder
00036 Imports GBufIn = System.String
00037 Imports GStatus = System.Byte
00038 ' IMPORTANT! Be sure that the paths below are correct
00039 Public Module LibraryPath
00040     Public Const GclibDllPath_ As String = "C:\Program Files (x86)\Galil\gclib\dll\x64\gclib.dll"
00041     Public Const GcliboDllPath_ As String = "C:\Program Files (x86)\Galil\gclib\dll\x64\gclibo.dll"
00042 End Module
00043 #End If
```

```
00044
00045 ''' <summary>
00046 ''' Provides a class that binds to gclib's unmanaged dll. Wraps each call and provides a more
      user-friendly interface for use in Visual Basic.
00047 ''' </summary>
00048 ''' <remarks>
00049 ''' The Gclib class assumes the default installation of gclib, "C:\Program Files (x86)\Galil\gclib\".
00050 ''' If the dlls are elsewhere, change the path strings GclibDllPath_, and GcliboDllPath_.
00051 ''' </remarks>
00052 Public Class Gclib
00053
00054 #Region "VB wrappers of gclib C calls"
00055
00056 #Region "Private properties"
00057     Private Const BufferSize_ As Integer = 500000 'size of "char *" buffer. Big enough to fit entire
      4000 program via UL/LS, or 24000 elements of array data.
00058     Private Buffer_ As New System.Text.StringBuilder(BufferSize_) 'used to pass a "char *" to gclib.
00059     Private ByteArray_(512) As Byte 'byte array to hold data record and response to GRead
00060     Private ConnectionHandle_ As GCon 'keep track of the gclib connection handle.
00061     Private ConnectionStatus_ As Boolean 'keep track of the status of gclib's connection.
00062 #End Region
00063
00064     ''' <summary>
00065     ''' Constructor of the gclib wrapper class.
00066     ''' </summary>
00067     ''' <remarks>Checks to ensure gclib dlls are in the correct location.</remarks>
00068     ''' <exception cref="System.Exception">Will throw an exception if either dll isn't
      found.</exception>
00069     Public Sub New()
00070         If Not File.Exists(GclibDllPath_) Then
00071             Throw New System.Exception("Could not find gclib dll at " & GclibDllPath_)
00072         End If
00073
00074         If Not File.Exists(GcliboDllPath_) Then
00075             Throw New System.Exception("Could not find gclibo dll at " & GcliboDllPath_)
00076         End If
00077     End Sub
00078
00079     ''' <summary>
00080     ''' Return a string array of available connection addresses.
00081     ''' </summary>
00082     ''' <returns>String array containing all available Galil Ethernet controllers, PCI controllers, and
      COM ports.</returns>
00083     ''' <remarks>Wrapper around gclib GAddresses(),
00084     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a6a6114683ed5749519b64f19512c24d6
00085     ''' An empty array is returned on error.</remarks>
00086     Public Function GAddresses() As String()
00087         Dim rc As GReturn = DllGAddresses(Buffer_, BufferSize_)
00088         If rc = G_NO_ERROR Then
00089             Return Buffer_.ToString().Split({vbCr, vbLf},
      System.StringSplitOptions.RemoveEmptyEntries)
00090         Else
00091             Return New String() {}
00092         End If
00093     End Function
00094
00095     ''' <summary>
00096     ''' Downloads array data to a pre-dimensioned array in the controller's array table.
00097     ''' </summary>
00098     ''' <param name="array_name">String containing the name of the array to download. Must match the
      array name used in DM.</param>
00099     ''' <param name="data">A list of doubles, to be downloaded.</param>
00100     ''' <param name="first">The first element of the array for sub-array downloads.</param>
00101     ''' <param name="last">The last element of the array for sub-array downloads.</param>
00102     ''' <remarks>Wrapper around gclib GArrayDownload(),
00103     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a6ea5ae6d167675e4c27ccfaf2f240f8a
00104     ''' The array must already exist on the controller, see DM and LA.</remarks>
00105     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00106     Public Sub GArrayDownload(array_name As String, ByRef data As List(Of Double), Optional first As
      Int16 = -1, Optional last As Int16 = -1)
00107         Dim ArrayData As New System.Text.StringBuilder(BufferSize_) 'for converting to ASCII
00108         Dim len As Integer = data.Count()
00109         For i As Integer = 0 To len - 1
00110             ArrayData.Append(data(i).ToString("F4")) 'format to fixed point
00111             If i < len - 1 Then
00112                 ArrayData.Append(",") 'delimiter
00113             End If
00114         Next
00115         Dim rc As GReturn = DllGArrayDownload(ConnectionHandle_, array_name, first, last,
      ArrayData.ToString())
00116         If Not rc = G_NO_ERROR Then
00117             Throw New System.Exception(GError(rc))
00118         End If
00119     End Sub
00120
00121     ''' <summary>
```

```
00122     ''' Allows downloading of a program array file to the controller.
00123     ''' </summary>
00124     ''' <param name="Path">The full filepath of the array csv file.</param>
00125     ''' <remarks>Wrapper around gclib GArrayDownload(),
00126     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a14b448ab8c7e6cf495865af301be398e
00127     ''' </remarks>
00128     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00129     Public Sub GArrayDownloadFile(Path As String)
00130         Dim rc As GReturn = DllGArrayDownloadFile(ConnectionHandle_, Path)
00131         If rc <> G_NO_ERROR Then
00132             Throw New System.Exception(GError(rc))
00133         End If
00134     End Sub
00135
00136     ''' <summary>
00137     ''' Uploads array data from the controller's array table.
00138     ''' </summary>
00139     ''' <param name="array_name">String containing the name of the array to upload.</param>
00140     ''' <param name="first">The first element of the array for sub-array uploads.</param>
00141     ''' <param name="last">The last element of the array for sub-array uploads.</param>
00142     ''' <returns>The desired array as a list of doubles.</returns>
00143     ''' <remarks>Wrapper around gclib GArrayUpload(),
00144     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#af215806ec26ba06ed3f174ebeeafa7a7
00145     ''' </remarks>
00146     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00147     Public Function GArrayUpload(array_name As String, Optional first As Int16 = -1, Optional last As
      Int16 = -1) As List(Of Double)
00148         Dim array As New List(Of Double)
00149         Dim rc As GReturn = DllGArrayUpload(ConnectionHandle_, array_name, first, last, 1, Buffer_,
      BufferSize_) '1 = comma delim
00150         If Not rc = G_NO_ERROR Then
00151             Throw New System.Exception(GError(rc))
00152         End If
00153         Dim tokens As String() = Buffer_.ToString.Split({","},
      System.StringSplitOptions.RemoveEmptyEntries)
00154         Dim value As Double
00155         For Each s As String In tokens
00156             If Not Double.TryParse(s, value) Then
00157                 Throw New System.Exception("Could not parse " & s & " into double")
00158             End If
00159             array.Add(value)
00160         Next
00161         Return array
00162     End Function
00163
00164     ''' <summary>
00165     ''' Allows uploading of a program array file from the controller to an array CSV file.
00166     ''' </summary>
00167     ''' <param name="Path">The full filepath of the array csv file to save.</param>
00168     ''' <param name="Names">A space separated list of the array names to upload. A null string uploads
      all arrays in the array table (LA).</param>
00169     ''' <remarks>Wrapper around gclib GArrayUpload().
00170     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#af215806ec26ba06ed3f174ebeeafa7a7
00171     ''' </remarks>
00172     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00173     Public Sub GArrayUploadFile(Path As String, Names As String)
00174         Dim rc As GReturn = DllGArrayUploadFile(ConnectionHandle_, Path, Names)
00175         If rc <> G_NO_ERROR Then
00176             Throw New System.Exception(GError(rc))
00177         End If
00178     End Sub
00179
00180     ''' <summary>
00181     ''' Assigns IP address over the Ethernet to a controller at a given MAC address.
00182     ''' </summary>
00183     ''' <param name="ip">The ip address to assign. The hardware should not yet have an IP address.
      </param>
00184     ''' <param name="mac">The MAC address of the hardware.</param>
00185     ''' <remarks>Wrapper around gclib GAssign(),
00186     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#acc996b7c22cfed8e5573d096ef1ab759
00187     ''' </remarks>
00188     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00189     Public Sub GAssign(ip As String, mac As String)
00190         Dim rc As GReturn = DllGAssign(ip, mac)
00191         If Not rc = G_NO_ERROR Then
00192             Throw New System.Exception(GError(rc))
00193         End If
00194     End Sub
00195
00196     ''' <summary>
00197     ''' Used to close a connection to Galil hardware.
00198     ''' </summary>
00199     ''' <remarks>Wrapper around gclib GClose(),
```

```
00200     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a24a437bcde9637b0db4b94176563a052
00201     ''' Be sure to call GClose() whenever a connection is finished.</remarks>
00202     Public Sub GClose()
00203         If ConnectionStatus_ Then
00204             DllGClose(ConnectionHandle_)
00205         End If
00206
00207         ConnectionStatus_ = False
00208     End Sub
00209
00210     ''' <summary>
00211     ''' Used for command-and-response transactions.
00212     ''' </summary>
00213     ''' <param name="Command">The command to send to the controller. Do not append a carriage return.
      Use only ASCII-based commmands.</param>
00214     ''' <param name="Trim">If true, the response will be trimmed of the trailing colon and any leading
      or trailing whitespace.</param>
00215     ''' <returns>The command's response.</returns>
00216     ''' <remarks>Wrapper around gclib GCommand(),
00217     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5ac031e76efc965affdd73a1bec084a8
00218     ''' </remarks>
00219     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00220     Public Function GCommand(Command As String, Optional Trim As Boolean = True) As String
00221         Dim rc As GReturn = DllGCommand(ConnectionHandle_, Command, Buffer_, BufferSize_, 0)
00222         If rc <> G_NO_ERROR Then
00223             Throw New System.Exception(GError(rc))
00224         End If
00225         If Trim Then
00226             Dim r As String = Buffer_.ToString()
00227             If r(r.Count() - 1) = ":" Then
00228                 r = r.Substring(0, r.Count() - 1)
00229             End If
00230             Return r.Trim()
00231         Else
00232             Return Buffer_.ToString()
00233         End If
00234     End Function
00235
00236
00237     ''' <summary>
00238     ''' Used for command-And-response transactions.
00239     ''' </summary>
00240     ''' <param name="Command">The command to send to the controller. Do Not append a carriage return.
      Use only ASCII-based commmands.</param>
00241     ''' <returns>The command's response parsed as an integer.</returns>
00242     ''' <remarks>Wrapper around gclib GCmdI(),
00243     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5ac031e76efc965affdd73a1bec084a8
00244     ''' </remarks>
00245     Public Function GCmdI(Command As String) As Int16
00246         Return Convert.ToInt16(Convert.ToDouble(GCommand(Command)))
00247     End Function
00248
00249     ''' <summary>
00250     ''' Used for command-And-response transactions.
00251     ''' </summary>
00252     ''' <param name="Command">The command to send to the controller. Do Not append a carriage return.
      Use only ASCII-based commmands.</param>
00253     ''' <returns>The command's response parsed as a double.</returns>
00254     ''' <remarks>Wrapper around gclib GCmdD(),
00255     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5ac031e76efc965affdd73a1bec084a8
00256     ''' </remarks>
00257     Public Function GCmdD(Command As String) As Double
00258         Return Convert.ToDouble(GCommand(Command))
00259     End Function
00260
00261     ''' <summary>
00262     ''' Provides a human-readable error message from a gclib error code.
00263     ''' </summary>
00264     ''' <param name="ErrorCode">The gclib error code, as returned from a call to the gclib.</param>
00265     ''' <returns>Error message string.</returns>
00266     ''' <remarks>
00267     ''' Wrapper around gclib GError(),
00268     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#afef1bed615bd72134f3df6d3a5723cba
00269     '''  This function is private, all public calls that throw errors use this command for setting the
      exception message.
00270     ''' </remarks>
00271     Private Function GError(ErrorCode As GReturn) As String
00272         DllGError(ErrorCode, Buffer_, BufferSize_)
00273         Return ErrorCode.ToString & " " & Buffer_.ToString() & vbCrLf
00274     End Function
00275
00276     ''' <summary>
00277     ''' Upgrade firmware.
00278     ''' </summary>
00279     ''' <param name="filepath ">The full filepath of the firmware hex file.</param>
00280     ''' <remarks>Wrapper around gclib GFirmwareDownload(),
```

```
00281     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a1878a2285ff17897fa4fb20182ba6fdf
00282     ''' </remarks>
00283     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00284     Public Sub GFirmwareDownload(filepath As String)
00285         Dim rc As GReturn = DllGFirmwareDownload(ConnectionHandle_, filepath)
00286         If rc <> G_NO_ERROR Then
00287             Throw New System.Exception(GError(rc))
00288         End If
00289     End Sub
00290 
00291     ''' <summary>Provides a useful connection string.</summary>
00292     ''' <remarks>Wrapper around gclib GInfo(),
00293     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a08abfcff8a1a85a01987859473167518
00294     ''' </remarks>
00295     ''' <returns>String containing connection information, e.g. "192.168.0.43, DMC4020 Rev 1.2c, 291".
      A null string indicates an error was returned from the library.</returns>
00296     Public Function GInfo() As String
00297         Dim rc As GReturn = DllGInfo(ConnectionHandle_, Buffer_, BufferSize_)
00298         If rc = G_NO_ERROR Then
00299             Return Buffer_.ToString()
00300         Else
00301             Return ""
00302         End If
00303     End Function
00304 
00305     ''' <summary>
00306     ''' Provides access to PCI and UDP interrupts from the controller.
00307     ''' </summary>
00308     ''' <returns>The status byte from the controller. Zero will be returned if a status byte is not
      read.</returns>
00309     ''' <remarks>Wrapper around gclib GInterrupt(),
00310     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a5bcf802404a96343e7593d247b67f132
00311     ''' -s ALL or -s EI must be specified in the address argument of GOpen() to receive
      interrupts.</remarks>
00312     Public Function GInterrupt() As Byte
00313         Dim StatusByte As Byte = 0
00314         Dim rc As GReturn = DllGInterrupt(ConnectionHandle_, StatusByte)
00315         If rc = G_NO_ERROR Then
00316             Return StatusByte
00317         Else
00318             Return 0
00319         End If
00320     End Function
00321 
00322     ''' <summary>
00323     ''' Provides a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.
00324     ''' </summary>
00325     ''' <returns>Each line of the returned data will be of the form "model, serial_number, mac".
      </returns>
00326     ''' <remarks>Wrapper around gclib GIpRequests(),
00327     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a0afb4c82642a4ef86f997c39a5518952
00328     ''' An empty array is returned on error.
00329     ''' Call will take roughly 5 seconds to return.</remarks>
00330     Public Function GIpRequests() As String()
00331         Dim rc As GReturn = DllGIpRequests(Buffer_, BufferSize_)
00332         If rc = G_NO_ERROR Then
00333             Return Buffer_.ToString().Split({vbCr, vbLf},
      System.StringSplitOptions.RemoveEmptyEntries)
00334         Else
00335             Return New String() {}
00336         End If
00337     End Function
00338 
00339     ''' <summary>
00340     ''' Provides access to unsolicited messages.
00341     ''' </summary>
00342     ''' <returns>String containing all messages received by controller.</returns>
00343     ''' <remarks>Wrapper around gclib GMessage(),
00344     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#aabc5eaa09ddeca55ab8ee048b916cbcd
00345     '''An empty string is returned on error.
00346     ''' -s ALL or -s MG must be specified in the address argument of GOpen() to receive
      messages.</remarks>
00347     Public Function GMessage() As String
00348         Dim rc As GReturn = DllGMessage(ConnectionHandle_, Buffer_, BufferSize_)
00349         If rc = G_NO_ERROR Then
00350             Return Buffer_.ToString
00351         Else
00352             Return ""
00353         End If
00354     End Function
00355 
00356     ''' <summary>
00357     ''' Blocking call that returns once all axes specified have completed their motion.
00358     ''' </summary>
00359     ''' <param name="axes">A string containing a multiple-axes mask. Every character in the string
      should be a valid argument to MG_BGm, i.e. XYZWABCDEFGHST.</param>
```

```
00360     ''' <remarks>Wrapper around gclib GMotionComplete(),
00361     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a19c22087944298797070644197f397a
00362     ''' </remarks>
00363     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00364     Public Sub GMotionComplete(axes As String)
00365         Dim rc As GReturn = DllGMotionComplete(ConnectionHandle_, axes)
00366         If Not rc = G_NO_ERROR Then
00367             Throw New System.Exception(GError(rc))
00368         End If
00369     End Sub
00370 
00371     ''' <summary>
00372     ''' Used to open a connection to Galil hardware.
00373     ''' </summary>
00374     ''' <param name="address">Address string including any connection switches. See gclib documentation
      for GOpen().</param>
00375     ''' <remarks>Wrapper around gclib GOpen(),
00376     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#aef4aec8a85630eed029b7a46aea7db54
00377     ''' </remarks>
00378     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00379     Public Sub GOpen(address As String)
00380         Dim rc As GReturn = DllGOpen(address, ConnectionHandle_)
00381         If rc <> G_NO_ERROR Then
00382             Throw New System.Exception(GError(rc))
00383         Else
00384             ConnectionStatus_ = True
00385         End If
00386     End Sub
00387 
00388     ''' <summary>
00389     ''' Allows downloading of a DMC program from a string buffer.
00390     ''' </summary>
00391     ''' <param name="program">The program to download.</param>
00392     ''' <param name="preprocessor">Preprocessor directives. Use nullstring for none.</param>
00393     ''' <remarks>Wrapper around gclib GProgramDownload(),
00394     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#acafe19b2dd0537ff458e3c8afe3acfeb
00395     ''' </remarks>
00396     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00397     Public Sub GProgramDownload(ByRef program As String, Optional preprocessor As String = "")
00398         Dim rc As GReturn = DllGProgramDownload(ConnectionHandle_, program, preprocessor)
00399         If rc <> G_NO_ERROR Then
00400             Throw New System.Exception(GError(rc))
00401         End If
00402     End Sub
00403 
00404     ''' <summary>
00405     ''' Allows downloading of a DMC program from file.
00406     ''' </summary>
00407     ''' <param name="file_path">The full filepath of the DMC file.</param>
00408     ''' <param name="preprocessor">Preprocessor directives. Use nullstring for none.</param>
00409     ''' <remarks>Wrapper around gclib GProgramDownloadFile(),
00410     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a8e44e2e321df9e7b8c538bf2d640633f
00411     ''' </remarks>
00412     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00413     Public Sub GProgramDownloadFile(file_path As String, Optional preprocessor As String = "")
00414         Dim rc As GReturn = DllGProgramDownloadFile(ConnectionHandle_, file_path, preprocessor)
00415         If rc <> G_NO_ERROR Then
00416             Throw New System.Exception(GError(rc))
00417         End If
00418     End Sub
00419 
00420     ''' <summary>
00421     ''' Allows uploading of a DMC program to a string.
00422     ''' </summary>
00423     ''' <remarks>Wrapper around gclib GProgramUpload(),
00424     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a80a653ce387a2bd16bde2793c6de77e9
00425     ''' </remarks>
00426     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00427     Public Function GProgramUpload() As String
00428         Dim rc As GReturn = DllGProgramUpload(ConnectionHandle_, Buffer_, BufferSize_)
00429         If rc <> G_NO_ERROR Then
00430             Throw New System.Exception(GError(rc))
00431         Else
00432             Return Buffer_.ToString()
00433         End If
00434     End Function
00435 
00436     ''' <summary>
00437     ''' Allows uploading of a DMC program to a file.
00438     ''' </summary>
00439     ''' <param name="file_path">The full filepath of the DMC file to save.</param>
00440     ''' <remarks>Wrapper around gclib GProgramUploadFile(),
```

```
00441     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a38c5565afc11762fa19d37fbaa3c9aa3
00442     ''' </remarks>
00443     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00444     Public Sub GProgramUploadFile(file_path As String)
00445         Dim rc As GReturn = DllGProgramUploadFile(ConnectionHandle_, file_path)
00446         If rc <> G_NO_ERROR Then
00447             Throw New System.Exception(GError(rc))
00448         End If
00449     End Sub
00450 
00451     ''' <summary>
00452     ''' Performs a read on the connection.
00453     ''' </summary>
00454     ''' <returns>String containing the read data, or a nullstring if nothing was read or an error
      occured.</returns>
00455     ''' <remarks>Wrapper around gclib GRead(),
00456     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#adab6ec79b7e1bc7f0266684dd3434923
00457     ''' </remarks>
00458     Public Function GRead() As Byte()
00459         Dim read As UInteger
00460         Dim rc As GReturn = DllGRead(ConnectionHandle_, ByteArray_, ByteArray_.Length, read)
00461         If rc = G_NO_ERROR Then
00462             Dim ReturnData(read - 1) As Byte 'create an array of the correct size
00463             For i As Integer = 0 To read - 1
00464                 ReturnData(i) = ByteArray_(i) 'copy over the data
00465             Next
00466             Return ReturnData
00467         Else
00468             Return Nothing
00469         End If
00470     End Function
00471 
00472     ''' <summary>
00473     ''' Used for retrieving data records from the controller.
00474     ''' </summary>
00475     ''' <returns>A struct containing the information of the retrieved data record.</returns>
00476     ''' <param name="async">False to user QR, True to use DR.</param>
00477     ''' <remarks>Wrapper around gclib GRecord(),
00478     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#a1f39cd57dcfa55d065c972a020b1f8ee
00479     ''' To use async, -s ALL or -s DR must be specified in the address argument of GOpen(),
00480     ''' and the records must be started via DR or RecordRate().
00481     ''' </remarks>
00482     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00483     Public Function GRecord(Of T As GDataRecord)(async As Boolean) As T
00484         Dim method As UShort = 0 'QR mode
00485         If async Then
00486             method = 1 'DR mode
00487         End If
00488 
00489         Dim rc As GReturn = DllGRecord(ConnectionHandle_, ByteArray_, method)
00490         If rc <> G_NO_ERROR Then
00491             Throw New System.Exception(GError(rc))
00492         End If
00493         Return ByteArrayToDataRecord(Of T)(ByteArray_)
00494     End Function
00495 
00496     ''' <summary>
00497     ''' Sets the asynchronous data record to a user-specified period via DR.
00498     ''' </summary>
00499     ''' <param name="period_ms">Period, in milliseconds, to set up for the asynchronous data
      record.</param>
00500     ''' <remarks>Wrapper around gclib GRecordRate(),
00501     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#ada86dc9d33ac961412583881963a1b8a
00502     ''' Takes TM and product type into account and sets the DR period to the period requested by the
      user, if possible.</remarks>
00503     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00504     Public Sub GRecordRate(period_ms As Double)
00505         Dim rc As GReturn = DllGRecordRate(ConnectionHandle_, period_ms)
00506         If Not rc = G_NO_ERROR Then
00507             Throw New System.Exception(GError(rc))
00508         End If
00509     End Sub
00510 
00511     ''' <summary>
00512     ''' Set the timeout of communication transactions. Use -1 to set the original timeout from GOpen().
00513     ''' </summary>
00514     ''' <param name="timeout_ms ">New timeout in miliseconds.</param>
00515     ''' <remarks>Wrapper around gclib GTimeout(),
00516     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a179aa2d1b8e2227944cc06a7ceaf5640
00517     ''' </remarks>
00518     Public Sub GTimeout(timeout_ms As Int16)
00519         DllGTimeout(ConnectionHandle_, timeout_ms)
00520     End Sub
00521 
```

```
00522     ''' <summary>Used to get the gclib version.</summary>
00523     ''' <returns>The library version, e.g. "104.73.179". A null string indicates an error was returned
      from the library.</returns>
00524     ''' <remarks>Wrapper around gclib GVersion(),
00525     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclibo_8h.html#a1784b39416b77af20efc98a05f8ce475
00526     ''' </remarks>
00527     Public Function GVersion() As String
00528         Dim rc As GReturn = DllGVersion(Buffer_, BufferSize_)
00529         If rc = G_NO_ERROR Then
00530             Return Buffer_.ToString()
00531         Else
00532             Return ""
00533         End If
00534     End Function
00535 
00536     ''' <summary>
00537     ''' Performs a write on the connection.
00538     ''' </summary>
00539     ''' <param name="buffer">The user's write buffer. To send a Galil command, a terminating carriage
      return is usually required. </param>
00540     ''' <remarks>Wrapper around gclib GWrite(),
00541     ''' http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h.html#abe28ebaecd5b3940adf4e145d40e5456
00542     ''' </remarks>
00543     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00544     Public Sub GWrite(ByRef buffer As String)
00545         Dim rc As GReturn = DllGWrite(ConnectionHandle_, buffer, buffer.Length())
00546         If Not rc = G_NO_ERROR Then
00547             Throw New System.Exception(GError(rc))
00548         End If
00549     End Sub
00550 
00551     ''' <summary>
00552     ''' Allows downloading of a Galil compressed backup (gcb) file to the controller.
00553     ''' </summary>
00554     ''' <param name="Path">The full filepath of the gcb file.</param>
00555     ''' <param name="Options">A bit mask indicating which sectors of the gcb file to restore to the
      controller.</param>
00556     ''' <returns>The controller information stored in the gcb file.</returns>
00557     ''' <remarks>Wrapper around gclib GSetupDownloadFile(),
00558     '''
00559     ''' If options is specified as 0, the return string will have a number appended corresponding to a
      bit mask of the available gcb sectors
00560     ''' </remarks>
00561     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR is
      received from gclib.</exception>
00562     Public Function GSetupDownloadFile(Path As String, Options As Int32) As String()
00563         'Dim _info As New System.Text.StringBuilder(BufferSize_)
00564         Dim rc As GReturn = DllGSetupDownloadFile(ConnectionHandle_, Path, Options, Buffer_,
      BufferSize_)
00565         Dim ret_buf As String = Buffer_.ToString()
00566         ret_buf = Replace(ret_buf, vbCrLf, ", ")
00567 
00568         If Not Options = 0 Then
00569             If rc <> G_NO_ERROR Then
00570                 Throw New System.Exception(GError(rc))
00571             End If
00572         Else
00573             ret_buf += """options""" + "," + rc.ToString() + vbLf
00574         End If
00575 
00576         Return ret_buf.Split({vbLf}, System.StringSplitOptions.RemoveEmptyEntries)
00577     End Function
00578 
00579     ''' <summary>
00580     ''' Connects gclib to a New gcaps server
00581     ''' </summary>
00582     ''' <param name="server_name">Name of the server to connect.</param>
00583     ''' <remarks>Wrapper around gclib GSetServer(),
00584     ''' Call GSetServer("Local") to connect gclib back to local gcaps server
00585     ''' </remarks>
00586     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR Is
      received from gclib.</exception>
00587     Public Sub GSetServer(server_name As String)
00588         Dim rc As GReturn = DllGSetServer(server_name)
00589 
00590         If Not rc = G_NO_ERROR Then
00591             Throw New System.Exception(GError(rc))
00592         End If
00593     End Sub
00594 
00595     ''' <summary>
00596     ''' Retrieves the name of your local gcaps server And whether Or Not it Is currently published
00597     ''' </summary>
00598     ''' <returns>A string in the form "<server_name>, <isPublished>"</returns>
00599     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR Is
      received from gclib.</exception>
```

```
00600     Public Function GServerStatus() As String
00601         Dim rc As GReturn = DllGServerStatus(Buffer_, BufferSize_)
00602 
00603         If Not rc = G_NO_ERROR Then
00604             Throw New System.Exception(GError(rc))
00605         End If
00606 
00607         Return Buffer_.ToString()
00608     End Function
00609 
00610     ''' <summary>
00611     ''' Retrieves a list of gcaps servers that are advertising themselves on the local network
00612     ''' </summary>
00613     ''' <returns>A list of available gcaps server names</returns>
00614     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR Is
      received from gclib.</exception>
00615     Public Function GListServers() As String()
00616         Dim rc As GReturn = DllGListServers(Buffer_, BufferSize_)
00617 
00618         If Not rc = G_NO_ERROR Then
00619             Throw New System.Exception(GError(rc))
00620         End If
00621 
00622         Dim delimiters As Char() = New Char() {vbLf, vbNewLine}
00623         Return Buffer_.ToString().Split(delimiters, System.StringSplitOptions.RemoveEmptyEntries)
00624     End Function
00625 
00626     ''' <summary>
00627     ''' Publishes Or removes local gcaps server from the network
00628     ''' </summary>
00629     ''' <param name="server_name">Name to publish server under.</param>
00630     ''' <param name="publish">True=publish server, False=remove server.</param>
00631     ''' <param name="save">Save this configuration for future server reboots.</param>
00632     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR Is
      received from gclib.</exception>
00633     Public Sub GPublishServer(server_name As String, publish As Boolean, save As Boolean)
00634         Dim rc As GReturn = DllGPublishServer(server_name, Convert.ToInt16(publish),
      Convert.ToInt16(save))
00635 
00636         If Not rc = G_NO_ERROR Then
00637             Throw New System.Exception(GError(rc))
00638         End If
00639     End Sub
00640 
00641     ''' <summary>
00642     ''' Returns a list of IP Addresses that currently have an open connection to your hardware.
00643     ''' </summary>
00644     ''' <returns>Returns a list of IP Addresses that currently have an open connection to your
      hardware.</returns>
00645     ''' <exception cref="System.Exception">Will throw an exception if anything other than G_NO_ERROR Is
      received from gclib.</exception>
00646     Public Function GRemoteConnections() As String()
00647         Dim rc As GReturn = DllGRemoteConnections(Buffer_, BufferSize_)
00648 
00649         If Not rc = G_NO_ERROR Then
00650             Throw New System.Exception(GError(rc))
00651         End If
00652 
00653         Dim delimiters As Char() = New Char() {vbLf, vbNewLine}
00654         Return Buffer_.ToString().Split(delimiters, System.StringSplitOptions.RemoveEmptyEntries)
00655     End Function
00656 
00657 #End Region
00658 
00659 #Region "DLL Imports"
00660 
00661     'Import declarations for gclib functions. Functions are private to this class and are prefixed
      with "Dll" to distinguish from VB functions.
00662 
00663 #Region "Error Codes"
00664     ''' <summary>Functions are checked for G_NO_ERROR.</summary>
00665     ''' <remarks>Some functions throw exceptions if G_NO_ERROR is not returned.</remarks>
00666     Private Const G_NO_ERROR As Integer = 0
00667 #End Region
00668 
00669     <DllImport(GcliboDllPath_, EntryPoint:="GAddresses", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00670     Private Shared Function DllGAddresses(addresses As GCStringOut, addresses_len As GSize) As GReturn
00671     End Function
00672 
00673     <DllImport(GclibDllPath_, EntryPoint:="GArrayDownload", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00674     Private Shared Function DllGArrayDownload(g As GCon, array_name As GCStringIn, first As GOption,
00675                                               last As GOption, buffer As GCStringIn) As GReturn
00676     End Function
00677 
00678     <DllImport(GcliboDllPath_, EntryPoint:="GArrayDownloadFile", CharSet:=CharSet.Ansi,
```

```
    CallingConvention:=CallingConvention.StdCall)>
00679     Private Shared Function DllGArrayDownloadFile(g As GCon, path As GCStringIn) As GReturn
00680     End Function
00681
00682     <DllImport(GclibDllPath_, EntryPoint:="GArrayUpload", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00683     Private Shared Function DllGArrayUpload(g As GCon, array_name As GCStringIn, first As GOption,
00684                                             last As GOption, delim As GOption, buffer As
    GCStringOut, bufferLength As GSize) As GReturn
00685     End Function
00686
00687     <DllImport(GcliboDllPath_, EntryPoint:="GArrayUploadFile", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00688     Private Shared Function DllGArrayUploadFile(g As GCon, path As GCStringIn, names As GCStringIn) As
    GReturn
00689     End Function
00690
00691     <DllImport(GcliboDllPath_, EntryPoint:="GAssign", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00692     Private Shared Function DllGAssign(ip As GCStringIn, mac As GCStringIn) As GReturn
00693     End Function
00694
00695     <DllImport(GclibDllPath_, EntryPoint:="GClose", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00696     Private Shared Function DllGClose(g As GCon) As GReturn
00697     End Function
00698
00699     <DllImport(GclibDllPath_, EntryPoint:="GCommand", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00700     Private Shared Function DllGCommand(g As GCon, command As GCStringIn, buffer As GCStringOut,
00701                                         bufferLength As GSize, ByRef bytesReturned As
    GSize) As GReturn
00702     End Function
00703
00704     <DllImport(GcliboDllPath_, EntryPoint:="GError", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00705     Private Shared Sub DllGError(error_code As GReturn, errorbuf As GCStringOut, error_len As GSize)
00706     End Sub
00707
00708     <DllImport(GclibDllPath_, EntryPoint:="GFirmwareDownload", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00709     Private Shared Function DllGFirmwareDownload(g As GCon, path As GCStringIn) As GReturn
00710     End Function
00711
00712     <DllImport(GcliboDllPath_, EntryPoint:="GInfo", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00713     Private Shared Function DllGInfo(g As GCon, info As GCStringOut, infoLength As GSize) As GReturn
00714     End Function
00715
00716     <DllImport(GclibDllPath_, EntryPoint:="GInterrupt", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00717     Private Shared Function DllGInterrupt(g As GCon, ByRef status_byte As GStatus) As GReturn
00718     End Function
00719
00720     <DllImport(GcliboDllPath_, EntryPoint:="GIpRequests", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00721     Private Shared Function DllGIpRequests(requests As GCStringOut, requests_len As GSize) As GReturn
00722     End Function
00723
00724     <DllImport(GclibDllPath_, EntryPoint:="GMessage", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00725     Private Shared Function DllGMessage(g As GCon, buffer As GCStringOut, bufferLength As GSize) As
    GReturn
00726     End Function
00727
00728     <DllImport(GcliboDllPath_, EntryPoint:="GMotionComplete", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00729     Private Shared Function DllGMotionComplete(g As GCon, axes As GCStringIn) As GReturn
00730     End Function
00731
00732     <DllImport(GclibDllPath_, EntryPoint:="GOpen", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00733     Private Shared Function DllGOpen(address As GCStringIn, ByRef g As GCon) As GReturn
00734     End Function
00735
00736     <DllImport(GclibDllPath_, EntryPoint:="GProgramDownload", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00737     Private Shared Function DllGProgramDownload(g As GCon, program As GCStringIn, preprocessor As
    GCStringIn) As GReturn
00738     End Function
00739
00740     <DllImport(GcliboDllPath_, EntryPoint:="GProgramDownloadFile", CharSet:=CharSet.Ansi,
    CallingConvention:=CallingConvention.StdCall)>
00741     Private Shared Function DllGProgramDownloadFile(g As GCon, path As GCStringIn, preprocessor As
    GCStringIn) As GReturn
00742     End Function
00743
```

```
00744     <DllImport(GclibDllPath_, EntryPoint:="GProgramUpload", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00745     Private Shared Function DllGProgramUpload(g As GCon, buffer As GCStringOut, bufferLength As GSize)
      As GReturn
00746     End Function
00747
00748     <DllImport(GcliboDllPath_, EntryPoint:="GProgramUploadFile", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00749     Private Shared Function DllGProgramUploadFile(g As GCon, path As GCStringIn) As GReturn
00750     End Function
00751
00752     <DllImport(GclibDllPath_, EntryPoint:="GRead", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00753     Private Shared Function DllGRead(g As GCon, buffer As Byte(), buffer_len As GSize, ByRef
      bytes_read As GSize) As GReturn
00754     End Function
00755
00756     <DllImport(GclibDllPath_, EntryPoint:="GRecord", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00757     Private Shared Function DllGRecord(g As GCon, record As Byte(), method As GOption) As GReturn
00758     End Function
00759
00760     <DllImport(GcliboDllPath_, EntryPoint:="GRecordRate", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00761     Private Shared Function DllGRecordRate(g As GCon, period_ms As Double) As GReturn
00762     End Function
00763
00764     <DllImport(GcliboDllPath_, EntryPoint:="GTimeout", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00765     Private Shared Sub DllGTimeout(g As GCon, timeoutMs As Short)
00766     End Sub
00767
00768     <DllImport(GcliboDllPath_, EntryPoint:="GVersion", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00769     Private Shared Function DllGVersion(ver As GCStringOut, ver_len As GSize) As GReturn
00770     End Function
00771
00772     <DllImport(GclibDllPath_, EntryPoint:="GWrite", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00773     Private Shared Function DllGWrite(g As GCon, buffer As GCStringIn, buffer_len As GSize) As GReturn
00774     End Function
00775
00776     <DllImport(GcliboDllPath_, EntryPoint:="GSetupDownloadFile", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00777     Private Shared Function DllGSetupDownloadFile(g As GCon, path As GCStringIn, options As GOption,
      info As GCStringOut, info_len As GSize) As GReturn
00778     End Function
00779
00780     <DllImport(GcliboDllPath_, EntryPoint:="GSetServer", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00781     Private Shared Function DllGSetServer(server_name As GCStringIn) As GReturn
00782     End Function
00783
00784     <DllImport(GcliboDllPath_, EntryPoint:="GServerStatus", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00785     Private Shared Function DllGServerStatus(status As GCStringOut, status_len As GSize) As GReturn
00786     End Function
00787
00788     <DllImport(GcliboDllPath_, EntryPoint:="GListServers", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00789     Private Shared Function DllGListServers(servers As GCStringOut, servers_len As GSize) As GReturn
00790     End Function
00791
00792     <DllImport(GcliboDllPath_, EntryPoint:="GPublishServer", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00793     Private Shared Function DllGPublishServer(server_name As GCStringIn, publish As GOption, save As
      GOption) As GReturn
00794     End Function
00795
00796     <DllImport(GcliboDllPath_, EntryPoint:="GRemoteConnections", CharSet:=CharSet.Ansi,
      CallingConvention:=CallingConvention.StdCall)>
00797     Private Shared Function DllGRemoteConnections(connections As GCStringOut, connections_len As
      GSize) As GReturn
00798     End Function
00799
00800 #End Region
00801
00802 #Region "Data Record"
00803
00804     Private Function ByteArrayToDataRecord(Of T As GDataRecord)(array As Byte()) As T
00805         Dim handle As GCHandle = GCHandle.Alloc(array, GCHandleType.Pinned)
00806         Try
00807             Return Marshal.PtrToStructure(Of T)(handle.AddrOfPinnedObject())
00808         Finally
00809             handle.Free()
00810         End Try
00811     End Function
```

```
00812
00813     Public Interface GDataRecord
00814         Function byte_array() As Byte()
00815     End Interface
00816
00817     Private Shared Function StructToByteArray(record As GDataRecord)
00818         Dim size As Integer = Marshal.SizeOf(record)
00819         Dim arr(size) As Byte
00820
00821         Dim ptr As IntPtr = Marshal.AllocHGlobal(size)
00822         Marshal.StructureToPtr(record, ptr, True)
00823         Marshal.Copy(ptr, arr, 0, size)
00824         Marshal.FreeHGlobal(ptr)
00825         Return arr
00826     End Function
00827
00828     ' Data record struct for DMC-4000 controllers, including 4000, 4200, 4103, And 500x0.
00829     <StructLayout(LayoutKind.Sequential, Pack:=1)>
00830     Public Structure GDataRecord4000
00831         Implements GDataRecord
00832
00833         Public header_0 As UB                   '/*00*/       1st Byte of Header.
00834         Public header_1 As UB                   '/*01*/       2nd Byte of Header.
00835         Public header_2 As UB                   '/*02*/       3rd Byte of Header.
00836         Public header_3 As UB                   '/*03*/       4th Byte of Header.
00837
00838         Public sample_number As UW              '/*04-05*/    sample number.
00839
00840         Public input_bank_0 As UB               '/*06*/       general input bank 0 (inputs 1-8).
00841         Public input_bank_1 As UB               '/*07*/       general input bank 1 (inputs 9-16).
00842         Public input_bank_2 As UB               '/*08*/       general input bank 2 (inputs 17-24).
00843         Public input_bank_3 As UB               '/*09*/       general input bank 3 (inputs 25-32).
00844         Public input_bank_4 As UB               '/*10*/       general input bank 4 (inputs 33-40).
00845         Public input_bank_5 As UB               '/*11*/       general input bank 5 (inputs 41-48).
00846         Public input_bank_6 As UB               '/*12*/       general input bank 6 (inputs 49-56).
00847         Public input_bank_7 As UB               '/*13*/       general input bank 7 (inputs 57-64).
00848         Public input_bank_8 As UB               '/*14*/       general input bank 8 (inputs 65-72).
00849         Public input_bank_9 As UB               '/*15*/       general input bank 9 (inputs 73-80).
00850
00851         Public output_bank_0 As UB              '/*16*/       general output bank 0 (outputs 1-8).
00852         Public output_bank_1 As UB              '/*17*/       general output bank 1 (outputs 9-16).
00853         Public output_bank_2 As UB              '/*18*/       general output bank 2 (outputs 17-24).
00854         Public output_bank_3 As UB              '/*19*/       general output bank 3 (outputs 25-32).
00855         Public output_bank_4 As UB              '/*20*/       general output bank 4 (outputs 33-40).
00856         Public output_bank_5 As UB              '/*21*/       general output bank 5 (outputs 41-48).
00857         Public output_bank_6 As UB              '/*22*/       general output bank 6 (outputs 49-56).
00858         Public output_bank_7 As UB              '/*23*/       general output bank 7 (outputs 57-64).
00859         Public output_bank_8 As UB              '/*24*/       general output bank 8 (outputs 65-72).
00860         Public output_bank_9 As UB              '/*25*/       general output bank 9 (outputs 73-80).
00861
00862         Public reserved_0 As SW                 '/*26-27*/    Reserved.
00863         Public reserved_2 As SW                 '/*28-29*/    Reserved.
00864         Public reserved_4 As SW                 '/*30-31*/    Reserved.
00865         Public reserved_6 As SW                 '/*32-33*/    Reserved.
00866         Public reserved_8 As SW                 '/*34-35*/    Reserved.
00867         Public reserved_10 As SW                '/*36-37*/    Reserved.
00868         Public reserved_12 As SW                '/*38-39*/    Reserved.
00869         Public reserved_14 As SW                '/*40-41*/    Reserved.
00870
00871         Public ethernet_status_a As UB          '/*42*/       Ethernet Handle A Status.
00872         Public ethernet_status_b As UB          '/*43*/       Ethernet Handle B Status.
00873         Public ethernet_status_c As UB          '/*44*/       Ethernet Handle C Status.
00874         Public ethernet_status_d As UB          '/*45*/       Ethernet Handle D Status.
00875         Public ethernet_status_e As UB          '/*46*/       Ethernet Handle E Status.
00876         Public ethernet_status_f As UB          '/*47*/       Ethernet Handle F Status.
00877         Public ethernet_status_g As UB          '/*48*/       Ethernet Handle G Status.
00878         Public ethernet_status_h As UB          '/*49*/       Ethernet Handle H Status.
00879
00880         Public error_code As UB                 '/*50*/       error code.
00881         Public thread_status As UB              '/*51*/       thread status
00882         Public amplifier_status As UL           '/*52-55*/    Amplifier Status.
00883
00884         Public contour_segment_count As UL      '/*56-59*/    Segment Count for Contour Mode.
00885         Public contour_buffer_available As UW   '/*60-61*/    Buffer space remaining, Contour Mode.
00886
00887         Public s_plane_segment_count As UW      '/*62-63*/    segment count of coordinated move for S
      plane.
00888         Public s_plane_move_status As UW        '/*64-65*/    coordinated move status for S plane.
00889         Public s_distance As SL                 '/*66-69*/    distance traveled in coordinated move
      for S plane.
00890         Public s_plane_buffer_available As UW   '/*70-71*/    Buffer space remaining, S Plane.
00891
00892         Public t_plane_segment_count As UW      '/*72-73*/    segment count of coordinated move for T
      plane.
00893         Public t_plane_move_status As UW        '/*74-75*/    Coordinated move status for T plane.
00894         Public t_distance As SL                 '/*76-79*/    distance traveled in coordinated move
      for T plane.
```

```
00895         Public t_plane_buffer_available As UW   '/*80-81*/    Buffer space remaining, T Plane.
00896
00897         Public axis_a_status As UW              '/*82-83*/    A axis status.
00898         Public axis_a_switches As UB            '/*84*/       A axis switches.
00899         Public axis_a_stop_code As UB           '/*85*/       A axis stop code.
00900         Public axis_a_reference_position As SL  '/*86-89*/    A axis reference position.
00901         Public axis_a_motor_position As SL      '/*90-93*/    A axis motor position.
00902         Public axis_a_position_error As SL      '/*94-97*/    A axis position error.
00903         Public axis_a_aux_position As SL        '/*98-101*/   A axis auxiliary position.
00904         Public axis_a_velocity As SL            '/*102-105*/  A axis velocity.
00905         Public axis_a_torque As SL              '/*106-109*/  A axis torque.
00906         Public axis_a_analog_in As UW           '/*110-111*/  A axis analog input.
00907         Public axis_a_halls As UB               '/*112*/      A Hall Input Status.
00908         Public axis_a_reserved As UB            '/*113*/      Reserved.
00909         Public axis_a_variable As SL            '/*114-117*/  A User-defined variable (ZA).
00910
00911         Public axis_b_status As UW              '/*118-119*/  B axis status.
00912         Public axis_b_switches As UB            '/*120*/      B axis switches.
00913         Public axis_b_stop_code As UB           '/*121*/      B axis stop code.
00914         Public axis_b_reference_position As SL  '/*122-125*/  B axis reference position.
00915         Public axis_b_motor_position As SL      '/*126-129*/  B axis motor position.
00916         Public axis_b_position_error As SL      '/*130-133*/  B axis position error.
00917         Public axis_b_aux_position As SL        '/*134-137*/  B axis auxiliary position.
00918         Public axis_b_velocity As SL            '/*138-141*/  B axis velocity.
00919         Public axis_b_torque As SL              '/*142-145*/  B axis torque.
00920         Public axis_b_analog_in As UW           '/*146-147*/  B axis analog input.
00921         Public axis_b_halls As UB               '/*148*/      B Hall Input Status.
00922         Public axis_b_reserved As UB            '/*149*/      Reserved.
00923         Public axis_b_variable As SL            '/*150-153*/  B User-defined variable (ZA).
00924
00925         Public axis_c_status As UW              '/*154-155*/  C axis status.
00926         Public axis_c_switches As UB            '/*156*/      C axis switches.
00927         Public axis_c_stop_code As UB           '/*157*/      C axis stop code.
00928         Public axis_c_reference_position As SL  '/*158-161*/  C axis reference position.
00929         Public axis_c_motor_position As SL      '/*162-165*/  C axis motor position.
00930         Public axis_c_position_error As SL      '/*166-169*/  C axis position error.
00931         Public axis_c_aux_position As SL        '/*170-173*/  C axis auxiliary position.
00932         Public axis_c_velocity As SL            '/*174-177*/  C axis velocity.
00933         Public axis_c_torque As SL              '/*178-181*/  C axis torque.
00934         Public axis_c_analog_in As UW           '/*182-183*/  C axis analog input.
00935         Public axis_c_halls As UB               '/*184*/      C Hall Input Status.
00936         Public axis_c_reserved As UB            '/*185*/      Reserved.
00937         Public axis_c_variable As SL            '/*186-189*/  C User-defined variable (ZA).
00938
00939         Public axis_d_status As UW              '/*190-191*/  D axis status.
00940         Public axis_d_switches As UB            '/*192*/      D axis switches.
00941         Public axis_d_stop_code As UB           '/*193*/      D axis stop code.
00942         Public axis_d_reference_position As SL  '/*194-197*/  D axis reference position.
00943         Public axis_d_motor_position As SL      '/*198-201*/  D axis motor position.
00944         Public axis_d_position_error As SL      '/*202-205*/  D axis position error.
00945         Public axis_d_aux_position As SL        '/*206-209*/  D axis auxiliary position.
00946         Public axis_d_velocity As SL            '/*210-213*/  D axis velocity.
00947         Public axis_d_torque As SL              '/*214-217*/  D axis torque.
00948         Public axis_d_analog_in As UW           '/*218-219*/  D axis analog input.
00949         Public axis_d_halls As UB               '/*220*/      D Hall Input Status.
00950         Public axis_d_reserved As UB            '/*221*/      Reserved.
00951         Public axis_d_variable As SL            '/*222-225*/  D User-defined variable (ZA).
00952
00953         Public axis_e_status As UW              '/*226-227*/  E axis status.
00954         Public axis_e_switches As UB            '/*228*/      E axis switches.
00955         Public axis_e_stop_code As UB           '/*229*/      E axis stop code.
00956         Public axis_e_reference_position As SL  '/*230-233*/  E axis reference position.
00957         Public axis_e_motor_position As SL      '/*234-237*/  E axis motor position.
00958         Public axis_e_position_error As SL      '/*238-241*/  E axis position error.
00959         Public axis_e_aux_position As SL        '/*242-245*/  E axis auxiliary position.
00960         Public axis_e_velocity As SL            '/*246-249*/  E axis velocity.
00961         Public axis_e_torque As SL              '/*250-253*/  E axis torque.
00962         Public axis_e_analog_in As UW           '/*254-255*/  E axis analog input.
00963         Public axis_e_halls As UB               '/*256*/      E Hall Input Status.
00964         Public axis_e_reserved As UB            '/*257*/      Reserved.
00965         Public axis_e_variable As SL            '/*258-261*/  E User-defined variable (ZA).
00966
00967         Public axis_f_status As UW              '/*262-263*/  F axis status.
00968         Public axis_f_switches As UB            '/*264*/      F axis switches.
00969         Public axis_f_stop_code As UB           '/*265*/      F axis stop code.
00970         Public axis_f_reference_position As SL  '/*266-269*/  F axis reference position.
00971         Public axis_f_motor_position As SL      '/*270-273*/  F axis motor position.
00972         Public axis_f_position_error As SL      '/*274-277*/  F axis position error.
00973         Public axis_f_aux_position As SL        '/*278-281*/  F axis auxiliary position.
00974         Public axis_f_velocity As SL            '/*282-285*/  F axis velocity.
00975         Public axis_f_torque As SL              '/*286-289*/  F axis torque.
00976         Public axis_f_analog_in As UW           '/*290-291*/  F axis analog input.
00977         Public axis_f_halls As UB               '/*292*/      F Hall Input Status.
00978         Public axis_f_reserved As UB            '/*293*/      Reserved.
00979         Public axis_f_variable As SL            '/*294-297*/  F User-defined variable (ZA).
00980
00981         Public axis_g_status As UW              '/*298-299*/  G axis status.
```

```
00982         Public axis_g_switches As UB              '/*300*/      G axis switches.
00983         Public axis_g_stop_code As UB             '/*301*/      G axis stop code.
00984         Public axis_g_reference_position As SL    '/*302-305*/  G axis reference position.
00985         Public axis_g_motor_position As SL        '/*306-309*/  G axis motor position.
00986         Public axis_g_position_error As SL        '/*310-313*/  G axis position error.
00987         Public axis_g_aux_position As SL          '/*314-317*/  G axis auxiliary position.
00988         Public axis_g_velocity As SL              '/*318-321*/  G axis velocity.
00989         Public axis_g_torque As SL                '/*322-325*/  G axis torque.
00990         Public axis_g_analog_in As UW             '/*326-327*/  G axis analog input.
00991         Public axis_g_halls As UB                 '/*328*/      G Hall Input Status.
00992         Public axis_g_reserved As UB              '/*329*/      Reserved.
00993         Public axis_g_variable As SL              '/*330-333*/  G User-defined variable (ZA).
00994
00995         Public axis_h_status As UW                '/*334-335*/  H axis status.
00996         Public axis_h_switches As UB              '/*336*/      H axis switches.
00997         Public axis_h_stop_code As UB             '/*337*/      H axis stop code.
00998         Public axis_h_reference_position As SL    '/*338-341*/  H axis reference position.
00999         Public axis_h_motor_position As SL        '/*342-345*/  H axis motor position.
01000         Public axis_h_position_error As SL        '/*346-349*/  H axis position error.
01001         Public axis_h_aux_position As SL          '/*350-353*/  H axis auxiliary position.
01002         Public axis_h_velocity As SL              '/*354-357*/  H axis velocity.
01003         Public axis_h_torque As SL                '/*358-361*/  H axis torque.
01004         Public axis_h_analog_in As UW             '/*362-363*/  H axis analog input.
01005         Public axis_h_halls As UB                 '/*364*/      H Hall Input Status.
01006         Public axis_h_reserved As UB              '/*365*/      Reserved.
01007         Public axis_h_variable As SL              '/*366-369*/  H User-defined variable (ZA).
01008
01009         Public Function byte_array() As Byte() Implements GDataRecord.byte_array
01010             Return StructToByteArray(Me)
01011         End Function
01012     End Structure
01013
01014     ' Data record struct for DMC-52000 controller. Same as DMC-4000, with bank indicator added at byte
      40.
01015     <StructLayout(LayoutKind.Sequential, Pack:=1)>
01016     Public Structure GDataRecord52000
01017         Implements GDataRecord
01018
01019         Public header_0 As UB                     '/*00*/       1st Byte of Header.
01020         Public header_1 As UB                     '/*01*/       2nd Byte of Header.
01021         Public header_2 As UB                     '/*02*/       3rd Byte of Header.
01022         Public header_3 As UB                     '/*03*/       4th Byte of Header.
01023
01024         Public sample_number As UW                '/*04-05*/    sample number.
01025
01026         Public input_bank_0 As UB                 '/*06*/       general input bank 0 (inputs 1-8).
01027         Public input_bank_1 As UB                 '/*07*/       general input bank 1 (inputs 9-16).
01028         Public input_bank_2 As UB                 '/*08*/       general input bank 2 (inputs 17-24).
01029         Public input_bank_3 As UB                 '/*09*/       general input bank 3 (inputs 25-32).
01030         Public input_bank_4 As UB                 '/*10*/       general input bank 4 (inputs 33-40).
01031         Public input_bank_5 As UB                 '/*11*/       general input bank 5 (inputs 41-48).
01032         Public input_bank_6 As UB                 '/*12*/       general input bank 6 (inputs 49-56).
01033         Public input_bank_7 As UB                 '/*13*/       general input bank 7 (inputs 57-64).
01034         Public input_bank_8 As UB                 '/*14*/       general input bank 8 (inputs 65-72).
01035         Public input_bank_9 As UB                 '/*15*/       general input bank 9 (inputs 73-80).
01036
01037         Public output_bank_0 As UB                '/*16*/       general output bank 0 (outputs 1-8).
01038         Public output_bank_1 As UB                '/*17*/       general output bank 1 (outputs 9-16).
01039         Public output_bank_2 As UB                '/*18*/       general output bank 2 (outputs 17-24).
01040         Public output_bank_3 As UB                '/*19*/       general output bank 3 (outputs 25-32).
01041         Public output_bank_4 As UB                '/*20*/       general output bank 4 (outputs 33-40).
01042         Public output_bank_5 As UB                '/*21*/       general output bank 5 (outputs 41-48).
01043         Public output_bank_6 As UB                '/*22*/       general output bank 6 (outputs 49-56).
01044         Public output_bank_7 As UB                '/*23*/       general output bank 7 (outputs 57-64).
01045         Public output_bank_8 As UB                '/*24*/       general output bank 8 (outputs 65-72).
01046         Public output_bank_9 As UB                '/*25*/       general output bank 9 (outputs 73-80).
01047
01048         Public reserved_0 As SW                   '/*26-27*/    Reserved.
01049         Public reserved_2 As SW                   '/*28-29*/    Reserved.
01050         Public reserved_4 As SW                   '/*30-31*/    Reserved.
01051         Public reserved_6 As SW                   '/*32-33*/    Reserved.
01052         Public reserved_8 As SW                   '/*34-35*/    Reserved.
01053         Public reserved_10 As SW                  '/*36-37*/    Reserved.
01054         Public reserved_12 As SW                  '/*38-39*/    Reserved.
01055         Public ethercat_bank As UB                '/*40*/        EtherCAT Bank Indicator.
01056         Public reserved_14 As UB                  '/*41*/       Reserved.
01057
01058         Public ethernet_status_a As UB            '/*42*/       Ethernet Handle A Status.
01059         Public ethernet_status_b As UB            '/*43*/       Ethernet Handle B Status.
01060         Public ethernet_status_c As UB            '/*44*/       Ethernet Handle C Status.
01061         Public ethernet_status_d As UB            '/*45*/       Ethernet Handle D Status.
01062         Public ethernet_status_e As UB            '/*46*/       Ethernet Handle E Status.
01063         Public ethernet_status_f As UB            '/*47*/       Ethernet Handle F Status.
01064         Public ethernet_status_g As UB            '/*48*/       Ethernet Handle G Status.
01065         Public ethernet_status_h As UB            '/*49*/       Ethernet Handle H Status.
01066
01067         Public error_code As UB                   '/*50*/       error code.
```

```
01068         Public thread_status As UB              '/*51*/       thread status
01069         Public amplifier_status As UL           '/*52-55*/    Amplifier Status.
01070
01071         Public contour_segment_count As UL      '/*56-59*/    Segment Count for Contour Mode.
01072         Public contour_buffer_available As UW   '/*60-61*/    Buffer space remaining, Contour Mode.
01073
01074         Public s_plane_segment_count As UW      '/*62-63*/    segment count of coordinated move for S
      plane.
01075         Public s_plane_move_status As UW        '/*64-65*/    coordinated move status for S plane.
01076         Public s_distance As SL                 '/*66-69*/    distance traveled in coordinated move
      for S plane.
01077         Public s_plane_buffer_available As UW   '/*70-71*/    Buffer space remaining, S Plane.
01078
01079         Public t_plane_segment_count As UW      '/*72-73*/    segment count of coordinated move for T
      plane.
01080         Public t_plane_move_status As UW        '/*74-75*/    Coordinated move status for T plane.
01081         Public t_distance As SL                 '/*76-79*/    distance traveled in coordinated move
      for T plane.
01082         Public t_plane_buffer_available As UW   '/*80-81*/    Buffer space remaining, T Plane.
01083
01084         Public axis_a_status As UW              '/*82-83*/    A axis status.
01085         Public axis_a_switches As UB            '/*84*/       A axis switches.
01086         Public axis_a_stop_code As UB           '/*85*/       A axis stop code.
01087         Public axis_a_reference_position As SL  '/*86-89*/    A axis reference position.
01088         Public axis_a_motor_position As SL      '/*90-93*/    A axis motor position.
01089         Public axis_a_position_error As SL      '/*94-97*/    A axis position error.
01090         Public axis_a_aux_position As SL        '/*98-101*/   A axis auxiliary position.
01091         Public axis_a_velocity As SL            '/*102-105*/  A axis velocity.
01092         Public axis_a_torque As SL              '/*106-109*/  A axis torque.
01093         Public axis_a_analog_in As UW           '/*110-111*/  A axis analog input.
01094         Public axis_a_halls As UB               '/*112*/      A Hall Input Status.
01095         Public axis_a_reserved As UB            '/*113*/      Reserved.
01096         Public axis_a_variable As SL            '/*114-117*/  A User-defined variable (ZA).
01097
01098         Public axis_b_status As UW              '/*118-119*/  B axis status.
01099         Public axis_b_switches As UB            '/*120*/      B axis switches.
01100         Public axis_b_stop_code As UB           '/*121*/      B axis stop code.
01101         Public axis_b_reference_position As SL  '/*122-125*/  B axis reference position.
01102         Public axis_b_motor_position As SL      '/*126-129*/  B axis motor position.
01103         Public axis_b_position_error As SL      '/*130-133*/  B axis position error.
01104         Public axis_b_aux_position As SL        '/*134-137*/  B axis auxiliary position.
01105         Public axis_b_velocity As SL            '/*138-141*/  B axis velocity.
01106         Public axis_b_torque As SL              '/*142-145*/  B axis torque.
01107         Public axis_b_analog_in As UW           '/*146-147*/  B axis analog input.
01108         Public axis_b_halls As UB               '/*148*/      B Hall Input Status.
01109         Public axis_b_reserved As UB            '/*149*/      Reserved.
01110         Public axis_b_variable As SL            '/*150-153*/  B User-defined variable (ZA).
01111
01112         Public axis_c_status As UW              '/*154-155*/  C axis status.
01113         Public axis_c_switches As UB            '/*156*/      C axis switches.
01114         Public axis_c_stop_code As UB           '/*157*/      C axis stop code.
01115         Public axis_c_reference_position As SL  '/*158-161*/  C axis reference position.
01116         Public axis_c_motor_position As SL      '/*162-165*/  C axis motor position.
01117         Public axis_c_position_error As SL      '/*166-169*/  C axis position error.
01118         Public axis_c_aux_position As SL        '/*170-173*/  C axis auxiliary position.
01119         Public axis_c_velocity As SL            '/*174-177*/  C axis velocity.
01120         Public axis_c_torque As SL              '/*178-181*/  C axis torque.
01121         Public axis_c_analog_in As UW           '/*182-183*/  C axis analog input.
01122         Public axis_c_halls As UB               '/*184*/      C Hall Input Status.
01123         Public axis_c_reserved As UB            '/*185*/      Reserved.
01124         Public axis_c_variable As SL            '/*186-189*/  C User-defined variable (ZA).
01125
01126         Public axis_d_status As UW              '/*190-191*/  D axis status.
01127         Public axis_d_switches As UB            '/*192*/      D axis switches.
01128         Public axis_d_stop_code As UB           '/*193*/      D axis stop code.
01129         Public axis_d_reference_position As SL  '/*194-197*/  D axis reference position.
01130         Public axis_d_motor_position As SL      '/*198-201*/  D axis motor position.
01131         Public axis_d_position_error As SL      '/*202-205*/  D axis position error.
01132         Public axis_d_aux_position As SL        '/*206-209*/  D axis auxiliary position.
01133         Public axis_d_velocity As SL            '/*210-213*/  D axis velocity.
01134         Public axis_d_torque As SL              '/*214-217*/  D axis torque.
01135         Public axis_d_analog_in As UW           '/*218-219*/  D axis analog input.
01136         Public axis_d_halls As UB               '/*220*/      D Hall Input Status.
01137         Public axis_d_reserved As UB            '/*221*/      Reserved.
01138         Public axis_d_variable As SL            '/*222-225*/  D User-defined variable (ZA).
01139
01140         Public axis_e_status As UW              '/*226-227*/  E axis status.
01141         Public axis_e_switches As UB            '/*228*/      E axis switches.
01142         Public axis_e_stop_code As UB           '/*229*/      E axis stop code.
01143         Public axis_e_reference_position As SL  '/*230-233*/  E axis reference position.
01144         Public axis_e_motor_position As SL      '/*234-237*/  E axis motor position.
01145         Public axis_e_position_error As SL      '/*238-241*/  E axis position error.
01146         Public axis_e_aux_position As SL        '/*242-245*/  E axis auxiliary position.
01147         Public axis_e_velocity As SL            '/*246-249*/  E axis velocity.
01148         Public axis_e_torque As SL              '/*250-253*/  E axis torque.
01149         Public axis_e_analog_in As UW           '/*254-255*/  E axis analog input.
01150         Public axis_e_halls As UB               '/*256*/      E Hall Input Status.
```

```
01151         Public axis_e_reserved As UB            '/*257*/      Reserved.
01152         Public axis_e_variable As SL            '/*258-261*/  E User-defined variable (ZA).
01153
01154         Public axis_f_status As UW              '/*262-263*/  F axis status.
01155         Public axis_f_switches As UB            '/*264*/      F axis switches.
01156         Public axis_f_stop_code As UB           '/*265*/      F axis stop code.
01157         Public axis_f_reference_position As SL  '/*266-269*/  F axis reference position.
01158         Public axis_f_motor_position As SL      '/*270-273*/  F axis motor position.
01159         Public axis_f_position_error As SL      '/*274-277*/  F axis position error.
01160         Public axis_f_aux_position As SL        '/*278-281*/  F axis auxiliary position.
01161         Public axis_f_velocity As SL            '/*282-285*/  F axis velocity.
01162         Public axis_f_torque As SL              '/*286-289*/  F axis torque.
01163         Public axis_f_analog_in As UW           '/*290-291*/  F axis analog input.
01164         Public axis_f_halls As UB               '/*292*/      F Hall Input Status.
01165         Public axis_f_reserved As UB            '/*293*/      Reserved.
01166         Public axis_f_variable As SL            '/*294-297*/  F User-defined variable (ZA).
01167
01168         Public axis_g_status As UW              '/*298-299*/  G axis status.
01169         Public axis_g_switches As UB            '/*300*/      G axis switches.
01170         Public axis_g_stop_code As UB           '/*301*/      G axis stop code.
01171         Public axis_g_reference_position As SL  '/*302-305*/  G axis reference position.
01172         Public axis_g_motor_position As SL      '/*306-309*/  G axis motor position.
01173         Public axis_g_position_error As SL      '/*310-313*/  G axis position error.
01174         Public axis_g_aux_position As SL        '/*314-317*/  G axis auxiliary position.
01175         Public axis_g_velocity As SL            '/*318-321*/  G axis velocity.
01176         Public axis_g_torque As SL              '/*322-325*/  G axis torque.
01177         Public axis_g_analog_in As UW           '/*326-327*/  G axis analog input.
01178         Public axis_g_halls As UB               '/*328*/      G Hall Input Status.
01179         Public axis_g_reserved As UB            '/*329*/      Reserved.
01180         Public axis_g_variable As SL            '/*330-333*/  G User-defined variable (ZA).
01181
01182         Public axis_h_status As UW              '/*334-335*/  H axis status.
01183         Public axis_h_switches As UB            '/*336*/      H axis switches.
01184         Public axis_h_stop_code As UB           '/*337*/      H axis stop code.
01185         Public axis_h_reference_position As SL  '/*338-341*/  H axis reference position.
01186         Public axis_h_motor_position As SL      '/*342-345*/  H axis motor position.
01187         Public axis_h_position_error As SL      '/*346-349*/  H axis position error.
01188         Public axis_h_aux_position As SL        '/*350-353*/  H axis auxiliary position.
01189         Public axis_h_velocity As SL            '/*354-357*/  H axis velocity.
01190         Public axis_h_torque As SL              '/*358-361*/  H axis torque.
01191         Public axis_h_analog_in As UW           '/*362-363*/  H axis analog input.
01192         Public axis_h_halls As UB               '/*364*/      H Hall Input Status.
01193         Public axis_h_reserved As UB            '/*365*/      Reserved.
01194         Public axis_h_variable As SL            '/*366-369*/  H User-defined variable (ZA).
01195
01196         Public Function byte_array() As Byte() Implements GDataRecord.byte_array
01197             Return StructToByteArray(Me)
01198         End Function
01199     End Structure
01200
01201
01202     'Data record struct for DMC-1806 controller.
01203     '
01204     'The 18x6 Data record Is the same as 4000 except the following.
01205     '-# No header bytes. Firmware strips it in DR. Software removes it from QR.
01206     '-# No Ethernet status (bytes 42-49).
01207     '-# No amplfifier status (bytes 52-55).
01208     '-# No axis-specific hall input status.
01209     <StructLayout(LayoutKind.Sequential, Pack:=1)>
01210     Public Structure GDataRecord1806
01211         Implements GDataRecord
01212
01213         Public sample_number As UW              '/*00-01*/    sample number.
01214
01215         Public input_bank_0 As UB               '/*02*/       general input bank 0 (inputs 1-8).
01216         Public input_bank_1 As UB               '/*03*/       general input bank 1 (inputs 9-16).
01217         Public input_bank_2 As UB               '/*04*/       general input bank 2 (inputs 17-24).
01218         Public input_bank_3 As UB               '/*05*/       general input bank 3 (inputs 25-32).
01219         Public input_bank_4 As UB               '/*06*/       general input bank 4 (inputs 33-40).
01220         Public input_bank_5 As UB               '/*07*/       general input bank 5 (inputs 41-48).
01221         Public input_bank_6 As UB               '/*08*/       general input bank 6 (inputs 49-56).
01222         Public input_bank_7 As UB               '/*09*/       general input bank 7 (inputs 57-64).
01223         Public input_bank_8 As UB               '/*10*/       general input bank 8 (inputs 65-72).
01224         Public input_bank_9 As UB               '/*11*/       general input bank 9 (inputs 73-80).
01225
01226         Public output_bank_0 As UB              '/*12*/       general output bank 0 (outputs 1-8).
01227         Public output_bank_1 As UB              '/*13*/       general output bank 1 (outputs 9-16).
01228         Public output_bank_2 As UB              '/*14*/       general output bank 2 (outputs 17-24).
01229         Public output_bank_3 As UB              '/*15*/       general output bank 3 (outputs 25-32).
01230         Public output_bank_4 As UB              '/*16*/       general output bank 4 (outputs 33-40).
01231         Public output_bank_5 As UB              '/*17*/       general output bank 5 (outputs 41-48).
01232         Public output_bank_6 As UB              '/*18*/       general output bank 6 (outputs 49-56).
01233         Public output_bank_7 As UB              '/*19*/       general output bank 7 (outputs 57-64).
01234         Public output_bank_8 As UB              '/*20*/       general output bank 8 (outputs 65-72).
01235         Public output_bank_9 As UB              '/*21*/       general output bank 9 (outputs 73-80).
01236
01237         Public reserved_0 As SW                 '/*22-23*/    Reserved.
```

```
01238         Public reserved_2 As SW                 '/*24-25*/    Reserved.
01239         Public reserved_4 As SW                 '/*26-27*/    Reserved.
01240         Public reserved_6 As SW                 '/*28-29*/    Reserved.
01241         Public reserved_8 As SW                 '/*30-31*/    Reserved.
01242         Public reserved_10 As SW                '/*32-33*/    Reserved.
01243         Public reserved_12 As SW                '/*34-35*/    Reserved.
01244         Public reserved_14 As SW                '/*36-37*/    Reserved.
01245
01246         Public reserved_16 As UB                '/*38*/       Reserved.
01247         Public reserved_17 As UB                '/*39*/       Reserved.
01248         Public reserved_18 As UB                '/*40*/       Reserved.
01249         Public reserved_19 As UB                '/*41*/       Reserved.
01250         Public reserved_20 As UB                '/*42*/       Reserved.
01251         Public reserved_21 As UB                '/*43*/       Reserved.
01252         Public reserved_22 As UB                '/*44*/       Reserved.
01253         Public reserved_23 As UB                '/*45*/       Reserved.
01254
01255         Public error_code As UB                 '/*46*/       error code.
01256         Public thread_status As UB              '/*47*/       thread status.
01257         Public reserved_24 As UL                '/*48-51*/    Reserved.
01258
01259         Public contour_segment_count As UL      '/*52-55*/    Segment Count for Contour Mode.
01260         Public contour_buffer_available As UW   '/*56-57*/    Buffer space remaining, Contour Mode.
01261
01262         Public s_plane_segment_count As UW      '/*58-59*/    segment count of coordinated move for S
    plane.
01263         Public s_plane_move_status As UW        '/*60-61*/    coordinated move status for S plane.
01264         Public s_distance As SL                 '/*62-65*/    distance traveled in coordinated move
    for S plane.
01265         Public s_plane_buffer_available As UW   '/*66-67*/    Buffer space remaining, S Plane.
01266
01267         Public t_plane_segment_count As UW      '/*68-69*/    segment count of coordinated move for T
    plane.
01268         Public t_plane_move_status As UW        '/*70-71*/    Coordinated move status for T plane.
01269         Public t_distance As SL                 '/*72-75*/    distance traveled in coordinated move
    for T plane.
01270         Public t_plane_buffer_available As UW   '/*76-77*/    Buffer space remaining, T Plane.
01271
01272         Public axis_a_status As UW              '/*78-79*/    A axis status.
01273         Public axis_a_switches As UB            '/*80*/       A axis switches.
01274         Public axis_a_stop_code As UB           '/*81*/       A axis stop code.
01275         Public axis_a_reference_position As SL  '/*82-85*/    A axis reference position.
01276         Public axis_a_motor_position As SL      '/*86-89*/    A axis motor position.
01277         Public axis_a_position_error As SL      '/*90-93*/    A axis position error.
01278         Public axis_a_aux_position As SL        '/*94-97*/    A axis auxiliary position.
01279         Public axis_a_velocity As SL            '/*98-101*/   A axis velocity.
01280         Public axis_a_torque As SL              '/*102-105*/  A axis torque.
01281         Public axis_a_analog_in As UW           '/*106-107*/  A axis analog input.
01282         Public axis_a_reserved_0 As UB          '/*108*/      Reserved.
01283         Public axis_a_reserved_1 As UB          '/*109*/      Reserved.
01284         Public axis_a_variable As SL            '/*110-113*/  A User-defined variable (ZA).
01285
01286         Public axis_b_status As UW              '/*114-115*/  B axis status.
01287         Public axis_b_switches As UB            '/*116*/      B axis switches.
01288         Public axis_b_stop_code As UB           '/*117*/      B axis stop code.
01289         Public axis_b_reference_position As SL  '/*118-121*/  B axis reference position.
01290         Public axis_b_motor_position As SL      '/*122-125*/  B axis motor position.
01291         Public axis_b_position_error As SL      '/*126-129*/  B axis position error.
01292         Public axis_b_aux_position As SL        '/*130-133*/  B axis auxiliary position.
01293         Public axis_b_velocity As SL            '/*134-137*/  B axis velocity.
01294         Public axis_b_torque As SL              '/*138-141*/  B axis torque.
01295         Public axis_b_analog_in As UW           '/*142-143*/  B axis analog input.
01296         Public axis_b_reserved_0 As UB          '/*144*/      Reserved.
01297         Public axis_b_reserved_1 As UB          '/*145*/      Reserved.
01298         Public axis_b_variable As SL            '/*146-149*/  B User-defined variable (ZA).
01299
01300         Public axis_c_status As UW              '/*150-151*/  C axis status.
01301         Public axis_c_switches As UB            '/*152*/      C axis switches.
01302         Public axis_c_stop_code As UB           '/*153*/      C axis stop code.
01303         Public axis_c_reference_position As SL  '/*154-157*/  C axis reference position.
01304         Public axis_c_motor_position As SL      '/*158-161*/  C axis motor position.
01305         Public axis_c_position_error As SL      '/*162-165*/  C axis position error.
01306         Public axis_c_aux_position As SL        '/*166-169*/  C axis auxiliary position.
01307         Public axis_c_velocity As SL            '/*170-173*/  C axis velocity.
01308         Public axis_c_torque As SL              '/*174-177*/  C axis torque.
01309         Public axis_c_analog_in As UW           '/*178-179*/  C axis analog input.
01310         Public axis_c_reserved_0 As UB          '/*180*/      Reserved.
01311         Public axis_c_reserved_1 As UB          '/*181*/      Reserved.
01312         Public axis_c_variable As SL            '/*182-185*/  C User-defined variable (ZA).
01313
01314         Public axis_d_status As UW              '/*186-187*/  D axis status.
01315         Public axis_d_switches As UB            '/*188*/      D axis switches.
01316         Public axis_d_stop_code As UB           '/*189*/      D axis stop code.
01317         Public axis_d_reference_position As SL  '/*190-193*/  D axis reference position.
01318         Public axis_d_motor_position As SL      '/*194-197*/  D axis motor position.
01319         Public axis_d_position_error As SL      '/*198-201*/  D axis position error.
01320         Public axis_d_aux_position As SL        '/*202-205*/  D axis auxiliary position.
```

```
01321         Public axis_d_velocity As SL              '/*206-209*/  D axis velocity.
01322         Public axis_d_torque As SL                '/*210-213*/  D axis torque.
01323         Public axis_d_analog_in As UW             '/*214-215*/  D axis analog input.
01324         Public axis_d_reserved_0 As UB            '/*216*/      Reserved.
01325         Public axis_d_reserved_1 As UB            '/*217*/      Reserved.
01326         Public axis_d_variable As SL              '/*218-221*/  D User-defined variable (ZA).
01327
01328         Public axis_e_status As UW                '/*222-223*/  E axis status.
01329         Public axis_e_switches As UB              '/*224*/      E axis switches.
01330         Public axis_e_stop_code As UB             '/*225*/      E axis stop code.
01331         Public axis_e_reference_position As SL    '/*226-229*/  E axis reference position.
01332         Public axis_e_motor_position As SL        '/*230-233*/  E axis motor position.
01333         Public axis_e_position_error As SL        '/*234-237*/  E axis position error.
01334         Public axis_e_aux_position As SL          '/*238-241*/  E axis auxiliary position.
01335         Public axis_e_velocity As SL              '/*242-245*/  E axis velocity.
01336         Public axis_e_torque As SL                '/*256-249*/  E axis torque.
01337         Public axis_e_analog_in As UW             '/*250-251*/  E axis analog input.
01338         Public axis_e_reserved_0 As UB            '/*252*/      Reserved.
01339         Public axis_e_reserved_1 As UB            '/*253*/      Reserved.
01340         Public axis_e_variable As SL              '/*254-257*/  E User-defined variable (ZA).
01341
01342         Public axis_f_status As UW                '/*258-259*/  F axis status.
01343         Public axis_f_switches As UB              '/*260*/      F axis switches.
01344         Public axis_f_stop_code As UB             '/*261*/      F axis stop code.
01345         Public axis_f_reference_position As SL    '/*262-265*/  F axis reference position.
01346         Public axis_f_motor_position As SL        '/*266-269*/  F axis motor position.
01347         Public axis_f_position_error As SL        '/*270-273*/  F axis position error.
01348         Public axis_f_aux_position As SL          '/*274-277*/  F axis auxiliary position.
01349         Public axis_f_velocity As SL              '/*278-281*/  F axis velocity.
01350         Public axis_f_torque As SL                '/*282-285*/  F axis torque.
01351         Public axis_f_analog_in As UW             '/*286-287*/  F axis analog input.
01352         Public axis_f_reserved_0 As UB            '/*288*/      Reserved.
01353         Public axis_f_reserved_1 As UB            '/*289*/      Reserved.
01354         Public axis_f_variable As SL              '/*290-293*/  F User-defined variable (ZA).
01355
01356         Public axis_g_status As UW                '/*294-295*/  G axis status.
01357         Public axis_g_switches As UB              '/*296*/      G axis switches.
01358         Public axis_g_stop_code As UB             '/*297*/      G axis stop code.
01359         Public axis_g_reference_position As SL    '/*298-301*/  G axis reference position.
01360         Public axis_g_motor_position As SL        '/*302-305*/  G axis motor position.
01361         Public axis_g_position_error As SL        '/*306-309*/  G axis position error.
01362         Public axis_g_aux_position As SL          '/*310-313*/  G axis auxiliary position.
01363         Public axis_g_velocity As SL              '/*314-317*/  G axis velocity.
01364         Public axis_g_torque As SL                '/*318-321*/  G axis torque.
01365         Public axis_g_analog_in As UW             '/*322-323*/  G axis analog input.
01366         Public axis_g_reserved_0 As UB            '/*324*/      Reserved.
01367         Public axis_g_reserved_1 As UB            '/*325*/      Reserved.
01368         Public axis_g_variable As SL              '/*326-329*/  G User-defined variable (ZA).
01369
01370         Public axis_h_status As UW                '/*330-331*/  H axis status.
01371         Public axis_h_switches As UB              '/*332*/      H axis switches.
01372         Public axis_h_stop_code As UB             '/*333*/      H axis stop code.
01373         Public axis_h_reference_position As SL    '/*334-337*/  H axis reference position.
01374         Public axis_h_motor_position As SL        '/*338-341*/  H axis motor position.
01375         Public axis_h_position_error As SL        '/*342-345*/  H axis position error.
01376         Public axis_h_aux_position As SL          '/*346-349*/  H axis auxiliary position.
01377         Public axis_h_velocity As SL              '/*350-353*/  H axis velocity.
01378         Public axis_h_torque As SL                '/*354-357*/  H axis torque.
01379         Public axis_h_analog_in As UW             '/*358-359*/  H axis analog input.
01380         Public axis_h_reserved_0 As UB            '/*360*/      Reserved.
01381         Public axis_h_reserved_1 As UB            '/*361*/      Reserved.
01382         Public axis_h_variable As SL              '/*362-365*/  H User-defined variable (ZA).
01383
01384         Public Function byte_array() As Byte() Implements GDataRecord.byte_array
01385             Return StructToByteArray(Me)
01386         End Function
01387     End Structure
01388
01389     ' Data record struct for DMC-2103 controllers.
01390     <StructLayout(LayoutKind.Sequential, Pack:=1)>
01391     Public Structure GDataRecord2103
01392         Implements GDataRecord
01393
01394         Public header_0 As UB                     '/*00*/       1st Byte of Header.
01395         Public header_1 As UB                     '/*01*/       2nd Byte of Header.
01396         Public header_2 As UB                     '/*02*/       3rd Byte of Header.
01397         Public header_3 As UB                     '/*03*/       4th Byte of Header.
01398
01399         Public sample_number As UW                '/*04-05*/    sample number.
01400
01401         Public input_bank_0 As UB                 '/*06*/       general input bank 0 (inputs 1-8).
01402         Public input_bank_1 As UB                 '/*07*/       general input bank 1 (inputs 9-16).
01403         Public input_bank_2 As UB                 '/*08*/       general input bank 2 (inputs 17-24).
01404         Public input_bank_3 As UB                 '/*09*/       general input bank 3 (inputs 25-32).
01405         Public input_bank_4 As UB                 '/*10*/       general input bank 4 (inputs 33-40).
01406         Public input_bank_5 As UB                 '/*11*/       general input bank 5 (inputs 41-48).
01407         Public input_bank_6 As UB                 '/*12*/       general input bank 6 (inputs 49-56).
```

```
01408         Public input_bank_7 As UB                 '/*13*/       general input bank 7 (inputs 57-64).
01409         Public input_bank_8 As UB                 '/*14*/       general input bank 8 (inputs 65-72).
01410         Public input_bank_9 As UB                 '/*15*/       general input bank 9 (inputs 73-80).
01411
01412         Public output_bank_0 As UB                '/*16*/       general output bank 0 (outputs 1-8).
01413         Public output_bank_1 As UB                '/*17*/       general output bank 1 (outputs 9-16).
01414         Public output_bank_2 As UB                '/*18*/       general output bank 2 (outputs 17-24).
01415         Public output_bank_3 As UB                '/*19*/       general output bank 3 (outputs 25-32).
01416         Public output_bank_4 As UB                '/*20*/       general output bank 4 (outputs 33-40).
01417         Public output_bank_5 As UB                '/*21*/       general output bank 5 (outputs 41-48).
01418         Public output_bank_6 As UB                '/*22*/       general output bank 6 (outputs 49-56).
01419         Public output_bank_7 As UB                '/*23*/       general output bank 7 (outputs 57-64).
01420         Public output_bank_8 As UB                '/*24*/       general output bank 8 (outputs 65-72).
01421         Public output_bank_9 As UB                '/*25*/       general output bank 9 (outputs 73-80).
01422
01423         Public error_code As UB                   '/*26*/       error code.
01424         Public general_status As UB               '/*27*/       general status
01425
01426         Public s_plane_segment_count As UW        '/*28-29*/    segment count of coordinated move for S
      plane.
01427         Public s_plane_move_status As UW          '/*30-31*/    coordinated move status for S plane.
01428         Public s_distance As SL                   '/*32-35*/    distance traveled in coordinated move
      for S plane.
01429
01430         Public t_plane_segment_count As UW        '/*36-37*/    segment count of coordinated move for T
      plane.
01431         Public t_plane_move_status As UW          '/*38-39*/    Coordinated move status for T plane.
01432         Public t_distance As SL                   '/*40-43*/    distance traveled in coordinated move
      for T plane.
01433
01434         Public axis_a_status As UW                '/*44-45*/    A axis status.
01435         Public axis_a_switches As UB              '/*46*/       A axis switches.
01436         Public axis_a_stop_code As UB             '/*47*/       A axis stop code.
01437         Public axis_a_reference_position As SL    '/*48-51*/    A axis reference position.
01438         Public axis_a_motor_position As SL        '/*52-55*/    A axis motor position.
01439         Public axis_a_position_error As SL        '/*56-59*/    A axis position error.
01440         Public axis_a_aux_position As SL          '/*60-63*/    A axis auxiliary position.
01441         Public axis_a_velocity As SL              '/*64-67*/    A axis velocity.
01442         Public axis_a_torque As SW                '/*68-69*/    A axis torque.
01443         Public axis_a_analog_in As UW             '/*70-71*/    A axis analog input.
01444
01445         Public axis_b_status As UW                '/*72-73*/    B axis status.
01446         Public axis_b_switches As UB              '/*74*/       B axis switches.
01447         Public axis_b_stop_code As UB             '/*75*/       B axis stop code.
01448         Public axis_b_reference_position As SL    '/*76-79*/    B axis reference position.
01449         Public axis_b_motor_position As SL        '/*80-83*/    B axis motor position.
01450         Public axis_b_position_error As SL        '/*84-87*/    B axis position error.
01451         Public axis_b_aux_position As SL          '/*88-91*/    B axis auxiliary position.
01452         Public axis_b_velocity As SL              '/*92-95*/    B axis velocity.
01453         Public axis_b_torque As SW                '/*96-97*/    B axis torque.
01454         Public axis_b_analog_in As UW             '/*98-99*/    B axis analog input.
01455
01456         Public axis_c_status As UW                '/*100-101*/  C axis status.
01457         Public axis_c_switches As UB              '/*102*/      C axis switches.
01458         Public axis_c_stop_code As UB             '/*103*/      C axis stop code.
01459         Public axis_c_reference_position As SL    '/*104-107*/  C axis reference position.
01460         Public axis_c_motor_position As SL        '/*108-111*/  C axis motor position.
01461         Public axis_c_position_error As SL        '/*112-115*/  C axis position error.
01462         Public axis_c_aux_position As SL          '/*116-119*/  C axis auxiliary position.
01463         Public axis_c_velocity As SL              '/*120-123*/  C axis velocity.
01464         Public axis_c_torque As SW                '/*124-125*/  C axis torque.
01465         Public axis_c_analog_in As UW             '/*126-127*/  C axis analog input.
01466
01467         Public axis_d_status As UW                '/*128-129*/  D axis status.
01468         Public axis_d_switches As UB              '/*130*/      D axis switches.
01469         Public axis_d_stop_code As UB             '/*131*/      D axis stop code.
01470         Public axis_d_reference_position As SL    '/*132-135*/  D axis reference position.
01471         Public axis_d_motor_position As SL        '/*136-139*/  D axis motor position.
01472         Public axis_d_position_error As SL        '/*140-143*/  D axis position error.
01473         Public axis_d_aux_position As SL          '/*144-147*/  D axis auxiliary position.
01474         Public axis_d_velocity As SL              '/*148-151*/  D axis velocity.
01475         Public axis_d_torque As SW                '/*152-153*/  D axis torque.
01476         Public axis_d_analog_in As UW             '/*154-155*/  D axis analog input.
01477
01478         Public axis_e_status As UW                '/*156-157*/  E axis status.
01479         Public axis_e_switches As UB              '/*158*/      E axis switches.
01480         Public axis_e_stop_code As UB             '/*159*/      E axis stop code.
01481         Public axis_e_reference_position As SL    '/*160-163*/  E axis reference position.
01482         Public axis_e_motor_position As SL        '/*164-167*/  E axis motor position.
01483         Public axis_e_position_error As SL        '/*168-171*/  E axis position error.
01484         Public axis_e_aux_position As SL          '/*172-175*/  E axis auxiliary position.
01485         Public axis_e_velocity As SL              '/*176-179*/  E axis velocity.
01486         Public axis_e_torque As SW                '/*180-181*/  E axis torque.
01487         Public axis_e_analog_in As UW             '/*182-183*/  E axis analog input.
01488
01489         Public axis_f_status As UW                '/*184-185*/  F axis status.
01490         Public axis_f_switches As UB              '/*186*/      F axis switches.
```

```
01491         Public axis_f_stop_code As UB           '/*187*/      F axis stop code.
01492         Public axis_f_reference_position As SL  '/*188-191*/  F axis reference position.
01493         Public axis_f_motor_position As SL      '/*192-195*/  F axis motor position.
01494         Public axis_f_position_error As SL      '/*196-199*/  F axis position error.
01495         Public axis_f_aux_position As SL        '/*200-203*/  F axis auxiliary position.
01496         Public axis_f_velocity As SL            '/*204-207*/  F axis velocity.
01497         Public axis_f_torque As SW              '/*208-209*/  F axis torque.
01498         Public axis_f_analog_in As UW           '/*210-211*/  F axis analog input.
01499
01500         Public axis_g_status As UW              '/*212-213*/  G axis status.
01501         Public axis_g_switches As UB            '/*214*/      G axis switches.
01502         Public axis_g_stop_code As UB           '/*215*/      G axis stop code.
01503         Public axis_g_reference_position As SL  '/*216-219*/  G axis reference position.
01504         Public axis_g_motor_position As SL      '/*220-223*/  G axis motor position.
01505         Public axis_g_position_error As SL      '/*224-227*/  G axis position error.
01506         Public axis_g_aux_position As SL        '/*228-231*/  G axis auxiliary position.
01507         Public axis_g_velocity As SL            '/*232-235*/  G axis velocity.
01508         Public axis_g_torque As SW              '/*236-237*/  G axis torque.
01509         Public axis_g_analog_in As UW           '/*238-239*/  G axis analog input.
01510
01511         Public axis_h_status As UW              '/*240-241*/  H axis status.
01512         Public axis_h_switches As UB            '/*242*/      H axis switches.
01513         Public axis_h_stop_code As UB           '/*243*/      H axis stop code.
01514         Public axis_h_reference_position As SL  '/*244-247*/  H axis reference position.
01515         Public axis_h_motor_position As SL      '/*248-251*/  H axis motor position.
01516         Public axis_h_position_error As SL      '/*252-255*/  H axis position error.
01517         Public axis_h_aux_position As SL        '/*256-259*/  H axis auxiliary position.
01518         Public axis_h_velocity As SL            '/*260-263*/  H axis velocity.
01519         Public axis_h_torque As SW              '/*264-265*/  H axis torque.
01520         Public axis_h_analog_in As UW           '/*266-267*/  H axis analog input.
01521
01522         Public Function byte_array() As Byte() Implements GDataRecord.byte_array
01523             Return StructToByteArray(Me)
01524         End Function
01525     End Structure
01526
01527     ' Data record struct for DMC-1802 controllers.
01528     '
01529     'The 18x2 Data record Is the Same as 2103 except the following.
01530     '-# No header bytes. Software removes it from QR.
01531     '-# No analog in axis data.
01532     <StructLayout(LayoutKind.Sequential, Pack:=1)>
01533     Public Structure GDataRecord1802
01534         Implements GDataRecord
01535
01536         Public sample_number As UW              '/*00-01*/    sample number.
01537
01538         Public input_bank_0 As UB               '/*02*/       general input bank 0 (inputs 1-8).
01539         Public input_bank_1 As UB               '/*03*/       general input bank 1 (inputs 9-16).
01540         Public input_bank_2 As UB               '/*04*/       general input bank 2 (inputs 17-24).
01541         Public input_bank_3 As UB               '/*05*/       general input bank 3 (inputs 25-32).
01542         Public input_bank_4 As UB               '/*06*/       general input bank 4 (inputs 33-40).
01543         Public input_bank_5 As UB               '/*07*/       general input bank 5 (inputs 41-48).
01544         Public input_bank_6 As UB               '/*08*/       general input bank 6 (inputs 49-56).
01545         Public input_bank_7 As UB               '/*09*/       general input bank 7 (inputs 57-64).
01546         Public input_bank_8 As UB               '/*10*/       general input bank 8 (inputs 65-72).
01547         Public input_bank_9 As UB               '/*11*/       general input bank 9 (inputs 73-80).
01548
01549         Public output_bank_0 As UB              '/*12*/       general output bank 0 (outputs 1-8).
01550         Public output_bank_1 As UB              '/*13*/       general output bank 1 (outputs 9-16).
01551         Public output_bank_2 As UB              '/*14*/       general output bank 2 (outputs 17-24).
01552         Public output_bank_3 As UB              '/*15*/       general output bank 3 (outputs 25-32).
01553         Public output_bank_4 As UB              '/*16*/       general output bank 4 (outputs 33-40).
01554         Public output_bank_5 As UB              '/*17*/       general output bank 5 (outputs 41-48).
01555         Public output_bank_6 As UB              '/*18*/       general output bank 6 (outputs 49-56).
01556         Public output_bank_7 As UB              '/*19*/       general output bank 7 (outputs 57-64).
01557         Public output_bank_8 As UB              '/*20*/       general output bank 8 (outputs 65-72).
01558         Public output_bank_9 As UB              '/*21*/       general output bank 9 (outputs 73-80).
01559
01560         Public error_code As UB                 '/*22*/       error code.
01561         Public general_status As UB             '/*23*/       general status
01562
01563         Public s_plane_segment_count As UW      '/*24-25*/    segment count of coordinated move for S
      plane.
01564         Public s_plane_move_status As UW        '/*26-27*/    coordinated move status for S plane.
01565         Public s_distance As SL                 '/*28-31*/    distance traveled in coordinated move
      for S plane.
01566
01567         Public t_plane_segment_count As UW      '/*32-33*/    segment count of coordinated move for T
      plane.
01568         Public t_plane_move_status As UW        '/*34-35*/    Coordinated move status for T plane.
01569         Public t_distance As SL                 '/*36-39*/    distance traveled in coordinated move
      for T plane.
01570
01571         Public axis_a_status As UW              '/*40-41*/    A axis status.
01572         Public axis_a_switches As UB            '/*42*/       A axis switches.
01573         Public axis_a_stop_code As UB           '/*43*/       A axis stop code.
```

```
01574         Public axis_a_reference_position As SL  '/*44-47*/    A axis reference position.
01575         Public axis_a_motor_position As SL      '/*48-51*/    A axis motor position.
01576         Public axis_a_position_error As SL      '/*52-55*/    A axis position error.
01577         Public axis_a_aux_position As SL        '/*56-59*/    A axis auxiliary position.
01578         Public axis_a_velocity As SL            '/*60-63*/    A axis velocity.
01579         Public axis_a_torque As SW              '/*64-65*/    A axis torque.
01580         Public axis_a_reserved_0 As UB          '/*66*/       Reserved.
01581         Public axis_a_reserved_1 As UB          '/*67*/       Reserved.
01582
01583         Public axis_b_status As UW              '/*68-69*/    B axis status.
01584         Public axis_b_switches As UB            '/*70*/       B axis switches.
01585         Public axis_b_stop_code As UB           '/*71*/       B axis stop code.
01586         Public axis_b_reference_position As SL  '/*72-75*/    B axis reference position.
01587         Public axis_b_motor_position As SL      '/*76-79*/    B axis motor position.
01588         Public axis_b_position_error As SL      '/*80-83*/    B axis position error.
01589         Public axis_b_aux_position As SL        '/*84-87*/    B axis auxiliary position.
01590         Public axis_b_velocity As SL            '/*88-91*/    B axis velocity.
01591         Public axis_b_torque As SW              '/*92-93*/    B axis torque.
01592         Public axis_b_reserved_0 As UB          '/*94*/       Reserved.
01593         Public axis_b_reserved_1 As UB          '/*95*/       Reserved.
01594
01595         Public axis_c_status As UW              '/*96-97*/    C axis status.
01596         Public axis_c_switches As UB            '/*98*/       C axis switches.
01597         Public axis_c_stop_code As UB           '/*99*/       C axis stop code.
01598         Public axis_c_reference_position As SL  '/*100-103*/  C axis reference position.
01599         Public axis_c_motor_position As SL      '/*104-107*/  C axis motor position.
01600         Public axis_c_position_error As SL      '/*108-111*/  C axis position error.
01601         Public axis_c_aux_position As SL        '/*112-115*/  C axis auxiliary position.
01602         Public axis_c_velocity As SL            '/*116-119*/  C axis velocity.
01603         Public axis_c_torque As SW              '/*120-121*/  C axis torque.
01604         Public axis_c_reserved_0 As UB          '/*122*/      Reserved.
01605         Public axis_c_reserved_1 As UB          '/*123*/      Reserved.
01606
01607         Public axis_d_status As UW              '/*124-125*/  D axis status.
01608         Public axis_d_switches As UB            '/*126*/      D axis switches.
01609         Public axis_d_stop_code As UB           '/*127*/      D axis stop code.
01610         Public axis_d_reference_position As SL  '/*128-131*/  D axis reference position.
01611         Public axis_d_motor_position As SL      '/*132-135*/  D axis motor position.
01612         Public axis_d_position_error As SL      '/*136-139*/  D axis position error.
01613         Public axis_d_aux_position As SL        '/*140-143*/  D axis auxiliary position.
01614         Public axis_d_velocity As SL            '/*144-147*/  D axis velocity.
01615         Public axis_d_torque As SW              '/*148-149*/  D axis torque.
01616         Public axis_d_reserved_0 As UB          '/*150*/      Reserved.
01617         Public axis_d_reserved_1 As UB          '/*151*/      Reserved.
01618
01619         Public Function byte_array() As Byte() Implements GDataRecord.byte_array
01620             Return StructToByteArray(Me)
01621         End Function
01622     End Structure
01623
01624     ' Data record struct for DMC-30010 controllers.
01625     <StructLayout(LayoutKind.Sequential, Pack:=1)>
01626     Public Structure GDataRecord30000
01627         Implements GDataRecord
01628
01629         Public header_0 As UB                   '/*00*/       1st Byte of Header.
01630         Public header_1 As UB                   '/*01*/       2nd Byte of Header.
01631         Public header_2 As UB                   '/*02*/       3rd Byte of Header.
01632         Public header_3 As UB                   '/*03*/       4th Byte of Header.
01633
01634         Public sample_number As UW              '/*04-05*/    sample number.
01635
01636         Public input_bank_0 As UB               '/*06*/       general input bank 0 (inputs 1-8).
01637         Public input_bank_1 As UB               '/*07*/       general input bank 1 (inputs 9-16).
01638
01639         Public output_bank_0 As UB              '/*08*/       general output bank 0 (outputs 1-8).
01640         Public output_bank_1 As UB              '/*09*/       general output bank 1 (outputs 9-16).
01641
01642         Public error_code As UB                 '/*10*/       error code.
01643         Public thread_status As UB              '/*11*/       thread status.
01644
01645         Public input_analog_2 As UW             '/*12-13*/     Analog input 2. 1 is in axis data, see
      axis_a_analog_in.
01646
01647         Public output_analog_1 As UW            '/*14-15*/     Analog output 1.
01648         Public output_analog_2 As UW            '/*16-17*/     Analog output 2.
01649
01650         Public amplifier_status As UL           '/*18-21*/    Amplifier Status.
01651
01652         Public contour_segment_count As UL      '/*22-25*/    Segment Count for Contour Mode.
01653         Public contour_buffer_available As UW   '/*26-27*/    Buffer space remaining, Contour Mode.
01654
01655         Public s_plane_segment_count As UW      '/*28-29*/    segment count of coordinated move for S
      plane.
01656         Public s_plane_move_status As UW        '/*30-31*/    coordinated move status for S plane.
01657         Public s_distance As SL                 '/*32-35*/    distance traveled in coordinated move
      for S plane.
```

```
01658         Public s_plane_buffer_available As UW   '/*36-37*/    Buffer space remaining, S Plane.
01659
01660         Public axis_a_status As UW              '/*38-39*/    A axis status.
01661         Public axis_a_switches As UB            '/*40*/       A axis switches.
01662         Public axis_a_stop_code As UB           '/*41*/       A axis stop code.
01663         Public axis_a_reference_position As SL  '/*42-45*/    A axis reference position.
01664         Public axis_a_motor_position As SL      '/*46-49*/    A axis motor position.
01665         Public axis_a_position_error As SL      '/*50-53*/    A axis position error.
01666         Public axis_a_aux_position As SL        '/*54-57*/    A axis auxiliary position.
01667         Public axis_a_velocity As SL            '/*58-61*/    A axis velocity.
01668         Public axis_a_torque As SL              '/*62-65*/    A axis torque.
01669         Public axis_a_analog_in As UW           '/*66-67*/    A axis analog input.
01670         Public axis_a_halls As UB               '/*68*/       A Hall Input Status.
01671         Public axis_a_reserved As UB            '/*69*/       Reserved.
01672         Public axis_a_variable As SL            '/*70-73*/    A User-defined variable (ZA).
01673
01674         Public Function byte_array() As Byte() Implements GDataRecord.byte_array
01675             Return StructToByteArray(Me)
01676         End Function
01677     End Structure
01678
01679     ' Data record struct for RIO-471xx and RIO-472xx PLCs. Includes encoder fields.
01680     <StructLayout(LayoutKind.Sequential, Pack:=1)>
01681     Public Structure GDataRecord47000_ENC
01682         Implements GDataRecord
01683
01684         Public header_0 As UB                   '/*00*/       1st Byte of Header.
01685         Public header_1 As UB                   '/*01*/       2nd Byte of Header.
01686         Public header_2 As UB                   '/*02*/       3rd Byte of Header.
01687         Public header_3 As UB                   '/*03*/       4th Byte of Header.
01688
01689         Public sample_number As UW              '/*04-05*/    Sample number.
01690         Public error_code As UB                 '/*06*/       Error code.
01691         Public general_status As UB             '/*07*/       General status.
01692
01693         Public output_analog_0 As UW            '/*08-09*/    Analog output 0.
01694         Public output_analog_1 As UW            '/*10-11*/    Analog output 1.
01695         Public output_analog_2 As UW            '/*12-13*/    Analog output 2.
01696         Public output_analog_3 As UW            '/*14-15*/    Analog output 3.
01697         Public output_analog_4 As UW            '/*16-17*/    Analog output 4.
01698         Public output_analog_5 As UW            '/*18-19*/    Analog output 5.
01699         Public output_analog_6 As UW            '/*20-21*/    Analog output 6.
01700         Public output_analog_7 As UW            '/*22-23*/    Analog output 7.
01701
01702         Public input_analog_0 As UW             '/*24-25*/    Analog input 0.
01703         Public input_analog_1 As UW             '/*26-27*/    Analog input 1.
01704         Public input_analog_2 As UW             '/*28-29*/    Analog input 2.
01705         Public input_analog_3 As UW             '/*30-31*/    Analog input 3.
01706         Public input_analog_4 As UW             '/*32-33*/    Analog input 4.
01707         Public input_analog_5 As UW             '/*34-35*/    Analog input 5.
01708         Public input_analog_6 As UW             '/*36-37*/    Analog input 6.
01709         Public input_analog_7 As UW             '/*38-39*/    Analog input 7.
01710
01711         Public output_bank_0 As UW              '/*40-41*/    Digital outputs 0-15;
01712
01713         Public input_bank_0 As UW               '/*42-43*/    Digital inputs 0-15;
01714
01715         Public pulse_count_0 As UL              '/*44-47*/    Pulse counter (see PC).
01716         Public zc_variable As SL                '/*48-51*/    ZC User-defined variable (see ZC).
01717         Public zd_variable As SL                '/*52-55*/    ZD User-defined variable (see ZD).
01718
01719         Public encoder_0 As SL                  '/*56-59*/    Encoder channel 0. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01720         Public encoder_1 As SL                  '/*60-63*/    Encoder channel 1. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01721         Public encoder_2 As SL                  '/*64-67*/    Encoder channel 2. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01722         Public encoder_3 As SL                  '/*68-71*/    Encoder channel 3. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01723
01724         Public Function byte_array() As Byte() Implements GDataRecord.byte_array
01725             Return StructToByteArray(Me)
01726         End Function
01727     End Structure
01728
01729     ' Data record struct for RIO-47300. Includes encoder fields.
01730     <StructLayout(LayoutKind.Sequential, Pack:=1)>
01731     Public Structure GDataRecord47300_ENC
01732         Implements GDataRecord
01733
01734         Public header_0 As UB                   '/*00*/       1st Byte of Header.
01735         Public header_1 As UB                   '/*01*/       2nd Byte of Header.
01736         Public header_2 As UB                   '/*02*/       3rd Byte of Header.
01737         Public header_3 As UB                   '/*03*/       4th Byte of Header.
01738
01739         Public sample_number As UW              '/*04-05*/    Sample number.
01740         Public error_code As UB                 '/*06*/       Error code.
```

```
01741         Public general_status As UB             '/*07*/        General status.
01742
01743         Public output_analog_0 As UW            '/*08-09*/     Analog output 0.
01744         Public output_analog_1 As UW            '/*10-11*/     Analog output 1.
01745         Public output_analog_2 As UW            '/*12-13*/     Analog output 2.
01746         Public output_analog_3 As UW            '/*14-15*/     Analog output 3.
01747         Public output_analog_4 As UW            '/*16-17*/     Analog output 4.
01748         Public output_analog_5 As UW            '/*18-19*/     Analog output 5.
01749         Public output_analog_6 As UW            '/*20-21*/     Analog output 6.
01750         Public output_analog_7 As UW            '/*22-23*/     Analog output 7.
01751
01752         Public input_analog_0 As UW             '/*24-25*/     Analog input 0.
01753         Public input_analog_1 As UW             '/*26-27*/     Analog input 1.
01754         Public input_analog_2 As UW             '/*28-29*/     Analog input 2.
01755         Public input_analog_3 As UW             '/*30-31*/     Analog input 3.
01756         Public input_analog_4 As UW             '/*32-33*/     Analog input 4.
01757         Public input_analog_5 As UW             '/*34-35*/     Analog input 5.
01758         Public input_analog_6 As UW             '/*36-37*/     Analog input 6.
01759         Public input_analog_7 As UW             '/*38-39*/     Analog input 7.
01760
01761         Public output_bank_0 As UW              '/*40-41*/     Digital outputs 0-15;
01762         Public output_bank_1 As UW              '/*42-43*/     Digital outputs 16-23;
01763
01764         Public input_bank_0 As UW               '/*44-45*/     Digital inputs 0-15;
01765         Public input_bank_1 As UW               '/*46-47*/     Digital inputs 16-23;
01766
01767         Public pulse_count_0 As UL              '/*48-51*/     Pulse counter (see PC).
01768         Public zc_variable As SL                '/*52-55*/     ZC User-defined variable (see ZC).
01769         Public zd_variable As SL                '/*56-59*/     ZD User-defined variable (see ZD).
01770
01771         Public encoder_0 As SL                  '/*60-63*/     Encoder channel 0. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01772         Public encoder_1 As SL                  '/*64-67*/     Encoder channel 1. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01773         Public encoder_2 As SL                  '/*68-71*/     Encoder channel 2. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01774         Public encoder_3 As SL                  '/*72-75*/     Encoder channel 3. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01775
01776         Public Function byte_array() As Byte() Implements GDataRecord.byte_array
01777             Return StructToByteArray(Me)
01778         End Function
01779     End Structure
01780
01781     ' Data record struct for RIO-47300 with 24EX I/O daughter board.
01782     <StructLayout(LayoutKind.Sequential, Pack:=1)>
01783     Public Structure GDataRecord47300_24EX
01784         Implements GDataRecord
01785
01786         Public header_0 As UB                   '/*00*/        1st Byte of Header.
01787         Public header_1 As UB                   '/*01*/        2nd Byte of Header.
01788         Public header_2 As UB                   '/*02*/        3rd Byte of Header.
01789         Public header_3 As UB                   '/*03*/        4th Byte of Header.
01790
01791         Public sample_number As UW              '/*04-05*/     Sample number.
01792         Public error_code As UB                 '/*06*/        Error code.
01793         Public general_status As UB             '/*07*/        General status.
01794
01795         Public output_analog_0 As UW            '/*08-09*/     Analog output 0.
01796         Public output_analog_1 As UW            '/*10-11*/     Analog output 1.
01797         Public output_analog_2 As UW            '/*12-13*/     Analog output 2.
01798         Public output_analog_3 As UW            '/*14-15*/     Analog output 3.
01799         Public output_analog_4 As UW            '/*16-17*/     Analog output 4.
01800         Public output_analog_5 As UW            '/*18-19*/     Analog output 5.
01801         Public output_analog_6 As UW            '/*20-21*/     Analog output 6.
01802         Public output_analog_7 As UW            '/*22-23*/     Analog output 7.
01803
01804         Public input_analog_0 As UW             '/*24-25*/     Analog input 0.
01805         Public input_analog_1 As UW             '/*26-27*/     Analog input 1.
01806         Public input_analog_2 As UW             '/*28-29*/     Analog input 2.
01807         Public input_analog_3 As UW             '/*30-31*/     Analog input 3.
01808         Public input_analog_4 As UW             '/*32-33*/     Analog input 4.
01809         Public input_analog_5 As UW             '/*34-35*/     Analog input 5.
01810         Public input_analog_6 As UW             '/*36-37*/     Analog input 6.
01811         Public input_analog_7 As UW             '/*38-39*/     Analog input 7.
01812
01813         Public output_bank_0 As UW              '/*40-41*/     Digital outputs 0-15.
01814         Public output_bank_1 As UW              '/*42-43*/     Digital outputs 16-23.
01815
01816         Public input_bank_0 As UW               '/*44-45*/     Digital inputs 0-15.
01817         Public input_bank_1 As UW               '/*46-47*/     Digital inputs 16-23.
01818
01819         Public pulse_count_0 As UL              '/*48-51*/     Pulse counter (see PC)8.
01820         Public zc_variable As SL                '/*52-55*/     ZC User-defined variable (see ZC).
01821         Public zd_variable As SL                '/*56-59*/     ZD User-defined variable (see ZD).
01822
01823         Public output_bank_2 As UW              '/*60-61*/     Digital outputs 24-39. Data only valid
```

```
      for parts with 24EXOUT.
01824         Public output_back_3 As UW             '/*62-63*/     Digital outputs 40-47. Data only valid
      for parts with 24EXOUT.
01825
01826         Public input_bank_2 As UW              '/*64-65*/     Digital inputs 24-39. Data only valid
      for parts with 24EXIN.
01827         Public input_bank_3 As UW              '/*66-67*/     Digital inputs 40-47. Data only valid
      for parts with 24EXIN.
01828
01829         Public Function byte_array() As Byte() Implements GDataRecord.byte_array
01830             Return StructToByteArray(Me)
01831         End Function
01832     End Structure
01833
01834     'Data record struct for RIO-47162.
01835     <StructLayout(LayoutKind.Sequential, Pack:=1)>
01836     Public Structure GDataRecord47162
01837         Implements GDataRecord
01838
01839         Public header_0 As UB                  '/*00*/        1st Byte of Header.
01840         Public header_1 As UB                  '/*01*/        2nd Byte of Header.
01841         Public header_2 As UB                  '/*02*/        3rd Byte of Header.
01842         Public header_3 As UB                  '/*03*/        4th Byte of Header.
01843
01844         Public sample_number As UW             '/*04-05*/     Sample number.
01845         Public error_code As UB                '/*06*/        Error code.
01846         Public general_status As UB            '/*07*/        General status.
01847
01848         Public output_analog_0 As UW           '/*08-09*/     Analog output 0.
01849         Public output_analog_1 As UW           '/*10-11*/     Analog output 1.
01850         Public output_analog_2 As UW           '/*12-13*/     Analog output 2.
01851         Public output_analog_3 As UW           '/*14-15*/     Analog output 3.
01852         Public output_analog_4 As UW           '/*16-17*/     Analog output 4.
01853         Public output_analog_5 As UW           '/*18-19*/     Analog output 5.
01854         Public output_analog_6 As UW           '/*20-21*/     Analog output 6.
01855         Public output_analog_7 As UW           '/*22-23*/     Analog output 7.
01856
01857         Public input_analog_0 As UW            '/*24-25*/     Analog input 0.
01858         Public input_analog_1 As UW            '/*26-27*/     Analog input 1.
01859         Public input_analog_2 As UW            '/*28-29*/     Analog input 2.
01860         Public input_analog_3 As UW            '/*30-31*/     Analog input 3.
01861         Public input_analog_4 As UW            '/*32-33*/     Analog input 4.
01862         Public input_analog_5 As UW            '/*34-35*/     Analog input 5.
01863         Public input_analog_6 As UW            '/*36-37*/     Analog input 6.
01864         Public input_analog_7 As UW            '/*38-39*/     Analog input 7.
01865
01866         Public output_byte_0 As UB             '/*40*/        Digital outputs 0-7.
01867         Public output_byte_1 As UB             '/*41*/        Digital outputs 8-15.
01868         Public output_byte_2 As UB             '/*42*/        Digital outputs 16-23.
01869
01870         Public input_byte_0 As UB              '/*43*/        Digital inputs 0-7.
01871         Public input_byte_1 As UB              '/*44*/        Digital inputs 8-15.
01872         Public input_byte_2 As UB              '/*45*/        Digital inputs 16-23.
01873         Public input_byte_3 As UB              '/*46*/        Digital inputs 24-31.
01874         Public input_byte_4 As UB              '/*47*/        Digital inputs 32-39.
01875
01876         Public pulse_count_0 As UL             '/*48-51*/     Pulse counter (see PC).
01877         Public zc_variable As SL               '/*52-55*/     ZC User-defined variable (see ZC).
01878         Public zd_variable As SL               '/*56-59*/     ZD User-defined variable (see ZD).
01879
01880         Public encoder_0 As SL                 '/*60-63*/     Encoder channel 0. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01881         Public encoder_1 As SL                 '/*64-67*/     Encoder channel 1. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01882         Public encoder_2 As SL                 '/*68-71*/     Encoder channel 2. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01883         Public encoder_3 As SL                 '/*72-75*/     Encoder channel 3. Data only valid for
      parts with -BISS, -QUAD, or -SSI.
01884
01885         Public Function byte_array() As Byte() Implements GDataRecord.byte_array
01886             Return StructToByteArray(Me)
01887         End Function
01888     End Structure
01889 #End Region
01890
01891 End Class
```

10.207 gclib_errors.h File Reference

Macros

- #define G_NO_ERROR 0

  Return value if function succeeded.

- #define G_NO_ERROR_S "no error"
- #define G_GCLIB_ERROR -1

  General library error. Indicates internal API caught an unexpected error. Contact Galil support if this error is returned, softwaresupport@galil.com.
- #define G_GCLIB_ERROR_S "gclib unexpected error"
- #define G_GCLIB_UTILITY_ERROR -2

  An invalid request value was specified to GUtility.
- #define G_GCLIB_UTILITY_ERROR_S "invalid request value or bad arguments were specified to GUtility()"
- #define G_GCLIB_UTILITY_IP_TAKEN -3

  The IP cannot be assigned because ping returned a reply.
- #define G_GCLIB_UTILITY_IP_TAKEN_S "ip address is already taken by a device on the network"
- #define G_GCLIB_NON_BLOCKING_READ_EMPTY -4

  GMessage, GInterrupt, and GRecord can be called with a zero timeout. If there wasn't data waiting in memory, this error is returned.
- #define G_GCLIB_NON_BLOCKING_READ_EMPTY_S "data was not waiting for a zero-timeout read"
- #define G_GCLIB_POLLING_FAILED -5

  GWaitForBool out of polling trials.
- #define G_GCLIB_POLLING_FAILED_S "exit condition not met in specified polling period"
- #define G_TIMEOUT -1100

  Operation timed out. Timeout is set by the –timeout option in GOpen() and can be overriden by GSetting().
- #define G_TIMEOUT_S "device timed out"
- #define G_OPEN_ERROR -1101

  Device could not be opened. E.G. Serial port or PCI device already open.
- #define G_OPEN_ERROR_S "device failed to open"
- #define G_ALREADY_OPEN -1111

  Serial or PCI file has a flock placed on it, presumably by another gclib connection.
- #define G_ALREADY_OPEN_S "Serial or PCI port already open"
- #define G_READ_ERROR -1103

  Device read failed. E.G. Socket was closed by remote host. See G_UTIL_GCAPS_KEEPALIVE.
- #define G_READ_ERROR_S "device read error"
- #define G_WRITE_ERROR -1104

  Device write failed. E.G. Socket was closed by remote host. See G_UTIL_GCAPS_KEEPALIVE.
- #define G_WRITE_ERROR_S "device write error"
- #define G_INVALID_PREPROCESSOR_OPTIONS -1204

  GProgramDownload was called with a bad preprocessor directive.
- #define G_INVALID_PREPROCESSOR_OPTIONS_S "preprocessor did not recognize options"
- #define G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND -1106

  GCommand() was called with an illegal command, e.g. ED, DL or QD.
- #define G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND_S "illegal command passed to command call"
- #define G_DATA_RECORD_ERROR -1107

  Data record error, e.g. DR attempted on serial connection.
- #define G_DATA_RECORD_ERROR_S "data record error"
- #define G_UNSUPPORTED_FUNCTION -1109

  Function cannot be called on this bus. E.G. GInterrupt() on serial.
- #define G_UNSUPPORTED_FUNCTION_S "function not supported on this communication bus"
- #define G_FIRMWARE_LOAD_NOT_SUPPORTED -1110

  Firmware is not supported on this bus, e.g. Ethernet for the DMC-21x3 series.
- #define G_FIRMWARE_LOAD_NOT_SUPPORTED_S "firmware cannot be loaded on this communication bus to this hardware"
- #define G_ARRAY_NOT_DIMENSIONED -1200

  Array operation was called on an array that was not in the controller's array table, see LA command.

- #define G_ARRAY_NOT_DIMENSIONED_S "array not dimensioned on controller or wrong size"
- #define G_CONNECTION_NOT_ESTABLISHED -1201

  Function was called with no connection.
- #define G_CONNECTION_NOT_ESTABLISHED_S "connection to hardware not established"
- #define G_ILLEGAL_DATA_IN_PROGRAM -1202

  Data to download not valid, e.g. \ in data.
- #define G_ILLEGAL_DATA_IN_PROGRAM_S "illegal ASCII character in program"
- #define G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT -1203

  Program preprocessor could not compress the program within the user's constraints.
- #define G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT_S "program cannot be compressed to fit on the controller"
- #define G_BAD_RESPONSE_QUESTION_MARK -10000

  Operation received a ?, indicating controller has a TC error.
- #define G_BAD_RESPONSE_QUESTION_MARK_S "question mark returned by controller"
- #define G_BAD_VALUE_RANGE -10002

  Bad value or range, e.g. GCon g variable passed to function was bad.
- #define G_BAD_VALUE_RANGE_S "value passed to function was bad or out of range"
- #define G_BAD_FULL_MEMORY -10003

  Not enough memory for an operation, e.g. all connections allowed for a process already taken.
- #define G_BAD_FULL_MEMORY_S "operation could not complete because of a memory error"
- #define G_BAD_LOST_DATA -10004

  Lost data, e.g. GCommand() response buffer was too small for the controller's response.
- #define G_BAD_LOST_DATA_S "data was lost due to buffer or fifo limitations"
- #define G_BAD_FILE -10005

  Bad file path, bad file contents, or bad write.
- #define G_BAD_FILE_S "file was not found, contents are invalid, or write failed"
- #define G_BAD_ADDRESS -10006

  Bad address.
- #define G_BAD_ADDRESS_S "a bad address was specified in open"
- #define G_BAD_FIRMWARE_LOAD -10008

  Bad firmware upgrade.
- #define G_BAD_FIRMWARE_LOAD_S "Firmware upgrade failed"
- #define G_GCAPS_OPEN_ERROR -20000

  gcaps connection couldn't open. Server is not running or is not reachable.
- #define G_GCAPS_OPEN_ERROR_S "gcaps connection could not be opened"
- #define G_GCAPS_SUBSCRIPTION_ERROR -20002

  GMessage(), GRecord(), GInterrupt() called on a connection without –subscribe switch.
- #define G_GCAPS_SUBSCRIPTION_ERROR_S "function requires subscription not specified in GOpen()"

10.207.1 Detailed Description

Defines values for the Galil C Library return codes and error strings.
Definition in file gclib_errors.h.

10.207.2 Macro Definition Documentation

10.207.2.1 G_ALREADY_OPEN

```
#define G_ALREADY_OPEN -1111
```

Serial or PCI file has a flock placed on it, presumably by another gclib connection.
Definition at line 37 of file gclib_errors.h.

10.207.2.2 G_ALREADY_OPEN_S

```
#define G_ALREADY_OPEN_S "Serial or PCI port already open"
```
Definition at line 38 of file gclib_errors.h.

10.207.2.3 G_ARRAY_NOT_DIMENSIONED

```
#define G_ARRAY_NOT_DIMENSIONED -1200
```
Array operation was called on an array that was not in the controller's array table, see LA command.
Definition at line 61 of file gclib_errors.h.

10.207.2.4 G_ARRAY_NOT_DIMENSIONED_S

```
#define G_ARRAY_NOT_DIMENSIONED_S "array not dimensioned on controller or wrong size"
```
Definition at line 62 of file gclib_errors.h.

10.207.2.5 G_BAD_ADDRESS

```
#define G_BAD_ADDRESS -10006
```
Bad address.
Definition at line 88 of file gclib_errors.h.

10.207.2.6 G_BAD_ADDRESS_S

```
#define G_BAD_ADDRESS_S "a bad address was specified in open"
```
Definition at line 89 of file gclib_errors.h.

10.207.2.7 G_BAD_FILE

```
#define G_BAD_FILE -10005
```
Bad file path, bad file contents, or bad write.
Definition at line 85 of file gclib_errors.h.

10.207.2.8 G_BAD_FILE_S

```
#define G_BAD_FILE_S "file was not found, contents are invalid, or write failed"
```
Definition at line 86 of file gclib_errors.h.

10.207.2.9 G_BAD_FIRMWARE_LOAD

```
#define G_BAD_FIRMWARE_LOAD -10008
```
Bad firmware upgrade.
Definition at line 91 of file gclib_errors.h.

10.207.2.10 G_BAD_FIRMWARE_LOAD_S

```
#define G_BAD_FIRMWARE_LOAD_S "Firmware upgrade failed"
```
Definition at line 92 of file gclib_errors.h.

10.207.2.11 G_BAD_FULL_MEMORY

```
#define G_BAD_FULL_MEMORY -10003
```
Not enough memory for an operation, e.g. all connections allowed for a process already taken.
Definition at line 79 of file gclib_errors.h.

10.207.2.12 G_BAD_FULL_MEMORY_S

```
#define G_BAD_FULL_MEMORY_S "operation could not complete because of a memory error"
```
Definition at line 80 of file gclib_errors.h.

10.207.2.13 G_BAD_LOST_DATA

```
#define G_BAD_LOST_DATA -10004
```
Lost data, e.g. GCommand() response buffer was too small for the controller's response.
Definition at line 82 of file gclib_errors.h.

10.207.2.14 G_BAD_LOST_DATA_S

```
#define G_BAD_LOST_DATA_S "data was lost due to buffer or fifo limitations"
```
Definition at line 83 of file gclib_errors.h.

10.207.2.15 G_BAD_RESPONSE_QUESTION_MARK

```
#define G_BAD_RESPONSE_QUESTION_MARK -10000
```
Operation received a ?, indicating controller has a TC error.
Definition at line 73 of file gclib_errors.h.

10.207.2.16 G_BAD_RESPONSE_QUESTION_MARK_S

```
#define G_BAD_RESPONSE_QUESTION_MARK_S "question mark returned by controller"
```
Definition at line 74 of file gclib_errors.h.

10.207.2.17 G_BAD_VALUE_RANGE

```
#define G_BAD_VALUE_RANGE -10002
```
Bad value or range, e.g. GCon *g* variable passed to function was bad.
Definition at line 76 of file gclib_errors.h.

10.207.2.18 G_BAD_VALUE_RANGE_S

```
#define G_BAD_VALUE_RANGE_S "value passed to function was bad or out of range"
```
Definition at line 77 of file gclib_errors.h.

10.207.2.19 G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND

```
#define G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND -1106
```
GCommand() was called with an illegal command, e.g. ED, DL or QD.
Definition at line 49 of file gclib_errors.h.

10.207.2.20 G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND_S

```
#define G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND_S "illegal command passed to command call"
```
Definition at line 50 of file gclib_errors.h.

10.207.2.21 G_CONNECTION_NOT_ESTABLISHED

```
#define G_CONNECTION_NOT_ESTABLISHED -1201
```
Function was called with no connection.
Definition at line 64 of file gclib_errors.h.

10.207.2.22 G_CONNECTION_NOT_ESTABLISHED_S

```
#define G_CONNECTION_NOT_ESTABLISHED_S "connection to hardware not established"
```
Definition at line 65 of file gclib_errors.h.

10.207.2.23 G_DATA_RECORD_ERROR

```
#define G_DATA_RECORD_ERROR -1107
```
Data record error, e.g. DR attempted on serial connection.
Definition at line 52 of file gclib_errors.h.

10.207.2.24 G_DATA_RECORD_ERROR_S

```
#define G_DATA_RECORD_ERROR_S "data record error"
```
Definition at line 53 of file gclib_errors.h.

10.207.2.25 G_FIRMWARE_LOAD_NOT_SUPPORTED

```
#define G_FIRMWARE_LOAD_NOT_SUPPORTED -1110
```
Firmware is not supported on this bus, e.g. Ethernet for the DMC-21x3 series.
Definition at line 58 of file gclib_errors.h.

10.207.2.26 G_FIRMWARE_LOAD_NOT_SUPPORTED_S

```
#define G_FIRMWARE_LOAD_NOT_SUPPORTED_S "firmware cannot be loaded on this communication bus
to this hardware"
```
Definition at line 59 of file gclib_errors.h.

10.207.2.27 G_GCAPS_OPEN_ERROR

```
#define G_GCAPS_OPEN_ERROR -20000
```
gcaps connection couldn't open. Server is not running or is not reachable.
Definition at line 94 of file gclib_errors.h.

10.207.2.28 G_GCAPS_OPEN_ERROR_S

```
#define G_GCAPS_OPEN_ERROR_S "gcaps connection could not be opened"
```
Definition at line 95 of file gclib_errors.h.

10.207.2.29 G_GCAPS_SUBSCRIPTION_ERROR

```
#define G_GCAPS_SUBSCRIPTION_ERROR -20002
```
GMessage(), GRecord(), GInterrupt() called on a connection without –subscribe switch.
Definition at line 97 of file gclib_errors.h.

10.207.2.30 G_GCAPS_SUBSCRIPTION_ERROR_S

```
#define G_GCAPS_SUBSCRIPTION_ERROR_S "function requires subscription not specified in GOpen()"
```
Definition at line 98 of file gclib_errors.h.

10.207.2.31 G_GCLIB_ERROR

```
#define G_GCLIB_ERROR -1
```
General library error. Indicates internal API caught an unexpected error. Contact Galil support if this error is returned, softwaresupport@galil.com.
Definition at line 16 of file gclib_errors.h.

10.207.2.32 G_GCLIB_ERROR_S

```
#define G_GCLIB_ERROR_S "gclib unexpected error"
```
Definition at line 17 of file gclib_errors.h.

10.207.2.33 G_GCLIB_NON_BLOCKING_READ_EMPTY

```
#define G_GCLIB_NON_BLOCKING_READ_EMPTY -4
```
GMessage, GInterrupt, and GRecord can be called with a zero timeout. If there wasn't data waiting in memory, this error is returned.
Definition at line 25 of file gclib_errors.h.

10.207.2.34 G_GCLIB_NON_BLOCKING_READ_EMPTY_S

```
#define G_GCLIB_NON_BLOCKING_READ_EMPTY_S "data was not waiting for a zero-timeout read"
```

Definition at line 26 of file gclib_errors.h.

10.207.2.35 G_GCLIB_POLLING_FAILED

```
#define G_GCLIB_POLLING_FAILED -5
```

GWaitForBool out of polling trials.
Definition at line 28 of file gclib_errors.h.

10.207.2.36 G_GCLIB_POLLING_FAILED_S

```
#define G_GCLIB_POLLING_FAILED_S "exit condition not met in specified polling period"
```

Definition at line 29 of file gclib_errors.h.

10.207.2.37 G_GCLIB_UTILITY_ERROR

```
#define G_GCLIB_UTILITY_ERROR -2
```

An invalid request value was specified to GUtility.
Definition at line 19 of file gclib_errors.h.

10.207.2.38 G_GCLIB_UTILITY_ERROR_S

```
#define G_GCLIB_UTILITY_ERROR_S "invalid request value or bad arguments were specified to
GUtility()"
```

Definition at line 20 of file gclib_errors.h.

10.207.2.39 G_GCLIB_UTILITY_IP_TAKEN

```
#define G_GCLIB_UTILITY_IP_TAKEN -3
```

The IP cannot be assigned because ping returned a reply.
Definition at line 22 of file gclib_errors.h.

10.207.2.40 G_GCLIB_UTILITY_IP_TAKEN_S

```
#define G_GCLIB_UTILITY_IP_TAKEN_S "ip address is already taken by a device on the network"
```

Definition at line 23 of file gclib_errors.h.

10.207.2.41 G_ILLEGAL_DATA_IN_PROGRAM

```
#define G_ILLEGAL_DATA_IN_PROGRAM -1202
```

Data to download not valid, e.g. \ in data.
Definition at line 67 of file gclib_errors.h.

10.207.2.42 G_ILLEGAL_DATA_IN_PROGRAM_S

```
#define G_ILLEGAL_DATA_IN_PROGRAM_S "illegal ASCII character in program"
```

Definition at line 68 of file gclib_errors.h.

10.207.2.43 G_INVALID_PREPROCESSOR_OPTIONS

```
#define G_INVALID_PREPROCESSOR_OPTIONS -1204
```

GProgramDownload was called with a bad preprocessor directive.
Definition at line 46 of file gclib_errors.h.

10.207.2.44 G_INVALID_PREPROCESSOR_OPTIONS_S

```
#define G_INVALID_PREPROCESSOR_OPTIONS_S "preprocessor did not recognize options"
```

Definition at line 47 of file gclib_errors.h.

10.207.2.45 G_NO_ERROR

```
#define G_NO_ERROR 0
```

Return value if function succeeded.
Definition at line 13 of file gclib_errors.h.

10.207.2.46 G_NO_ERROR_S

```
#define G_NO_ERROR_S "no error"
```

Definition at line 14 of file gclib_errors.h.

10.207.2.47 G_OPEN_ERROR

```
#define G_OPEN_ERROR -1101
```

Device could not be opened. E.G. Serial port or PCI device already open.
Definition at line 34 of file gclib_errors.h.

10.207.2.48 G_OPEN_ERROR_S

```
#define G_OPEN_ERROR_S "device failed to open"
```

Definition at line 35 of file gclib_errors.h.

10.207.2.49 G_READ_ERROR

```
#define G_READ_ERROR -1103
```

Device read failed. E.G. Socket was closed by remote host. See G_UTIL_GCAPS_KEEPALIVE.
Definition at line 40 of file gclib_errors.h.

10.207.2.50 G_READ_ERROR_S

```
#define G_READ_ERROR_S "device read error"
```

Definition at line 41 of file gclib_errors.h.

10.207.2.51 G_TIMEOUT

```
#define G_TIMEOUT -1100
```

Operation timed out. Timeout is set by the –timeout option in GOpen() and can be overriden by GSetting().
Definition at line 31 of file gclib_errors.h.

10.207.2.52 G_TIMEOUT_S

```
#define G_TIMEOUT_S "device timed out"
```

Definition at line 32 of file gclib_errors.h.

10.207.2.53 G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT

```
#define G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT -1203
```

Program preprocessor could not compress the program within the user's constraints.
Definition at line 70 of file gclib_errors.h.

10.207.2.54 G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT_S

```
#define G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT_S "program cannot be compressed to fit on the
controller"
```

Definition at line 71 of file gclib_errors.h.

10.207.2.55 G_UNSUPPORTED_FUNCTION

```
#define G_UNSUPPORTED_FUNCTION -1109
```

Function cannot be called on this bus. E.G. GInterrupt() on serial.
Definition at line 55 of file gclib_errors.h.

10.207.2.56 G_UNSUPPORTED_FUNCTION_S

```
#define G_UNSUPPORTED_FUNCTION_S "function not supported on this communication bus"
```
Definition at line 56 of file gclib_errors.h.

10.207.2.57 G_WRITE_ERROR

```
#define G_WRITE_ERROR -1104
```
Device write failed. E.G. Socket was closed by remote host. See G_UTIL_GCAPS_KEEPALIVE.
Definition at line 43 of file gclib_errors.h.

10.207.2.58 G_WRITE_ERROR_S

```
#define G_WRITE_ERROR_S "device write error"
```
Definition at line 44 of file gclib_errors.h.

10.208 gclib_errors.h

Go to the documentation of this file.

```
00001
00010 #ifndef I_0DD3687F_47D0_454B_ADB2_CBAB0ED46FCE
00011 #define I_0DD3687F_47D0_454B_ADB2_CBAB0ED46FCE
00012
00013 #define G_NO_ERROR 0
00014 #define G_NO_ERROR_S "no error"
00015
00016 #define G_GCLIB_ERROR -1
00017 #define G_GCLIB_ERROR_S "gclib unexpected error"
00018
00019 #define G_GCLIB_UTILITY_ERROR -2
00020 #define G_GCLIB_UTILITY_ERROR_S "invalid request value or bad arguments were specified to GUtility()"
00021
00022 #define G_GCLIB_UTILITY_IP_TAKEN -3
00023 #define G_GCLIB_UTILITY_IP_TAKEN_S "ip address is already taken by a device on the network"
00024
00025 #define G_GCLIB_NON_BLOCKING_READ_EMPTY -4
00026 #define G_GCLIB_NON_BLOCKING_READ_EMPTY_S "data was not waiting for a zero-timeout read"
00027
00028 #define G_GCLIB_POLLING_FAILED -5
00029 #define G_GCLIB_POLLING_FAILED_S "exit condition not met in specified polling period"
00030
00031 #define G_TIMEOUT -1100
00032 #define G_TIMEOUT_S "device timed out"
00033
00034 #define G_OPEN_ERROR -1101
00035 #define G_OPEN_ERROR_S "device failed to open"
00036
00037 #define G_ALREADY_OPEN -1111
00038 #define G_ALREADY_OPEN_S "Serial or PCI port already open"
00039
00040 #define G_READ_ERROR -1103
00041 #define G_READ_ERROR_S "device read error"
00042
00043 #define G_WRITE_ERROR -1104
00044 #define G_WRITE_ERROR_S "device write error"
00045
00046 #define G_INVALID_PREPROCESSOR_OPTIONS -1204
00047 #define G_INVALID_PREPROCESSOR_OPTIONS_S "preprocessor did not recognize options"
00048
00049 #define G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND -1106
00050 #define G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND_S "illegal command passed to command call"
00051
00052 #define G_DATA_RECORD_ERROR -1107
00053 #define G_DATA_RECORD_ERROR_S "data record error"
00054
00055 #define G_UNSUPPORTED_FUNCTION -1109
00056 #define G_UNSUPPORTED_FUNCTION_S "function not supported on this communication bus"
00057
00058 #define G_FIRMWARE_LOAD_NOT_SUPPORTED -1110
00059 #define G_FIRMWARE_LOAD_NOT_SUPPORTED_S "firmware cannot be loaded on this communication bus to this
     hardware"
00060
00061 #define G_ARRAY_NOT_DIMENSIONED -1200
00062 #define G_ARRAY_NOT_DIMENSIONED_S "array not dimensioned on controller or wrong size"
00063
00064 #define G_CONNECTION_NOT_ESTABLISHED -1201
00065 #define G_CONNECTION_NOT_ESTABLISHED_S "connection to hardware not established"
00066
```

```
00067 #define G_ILLEGAL_DATA_IN_PROGRAM -1202
00068 #define G_ILLEGAL_DATA_IN_PROGRAM_S "illegal ASCII character in program"
00069
00070 #define G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT -1203
00071 #define G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT_S "program cannot be compressed to fit on the controller"
00072
00073 #define G_BAD_RESPONSE_QUESTION_MARK -10000
00074 #define G_BAD_RESPONSE_QUESTION_MARK_S "question mark returned by controller"
00075
00076 #define G_BAD_VALUE_RANGE -10002
00077 #define G_BAD_VALUE_RANGE_S "value passed to function was bad or out of range"
00078
00079 #define G_BAD_FULL_MEMORY -10003
00080 #define G_BAD_FULL_MEMORY_S "operation could not complete because of a memory error"
00081
00082 #define G_BAD_LOST_DATA -10004
00083 #define G_BAD_LOST_DATA_S "data was lost due to buffer or fifo limitations"
00084
00085 #define G_BAD_FILE -10005
00086 #define G_BAD_FILE_S "file was not found, contents are invalid, or write failed"
00087
00088 #define G_BAD_ADDRESS -10006
00089 #define G_BAD_ADDRESS_S "a bad address was specified in open"
00090
00091 #define G_BAD_FIRMWARE_LOAD -10008
00092 #define G_BAD_FIRMWARE_LOAD_S "Firmware upgrade failed"
00093
00094 #define G_GCAPS_OPEN_ERROR -20000
00095 #define G_GCAPS_OPEN_ERROR_S "gcaps connection could not be opened"
00096
00097 #define G_GCAPS_SUBSCRIPTION_ERROR -20002
00098 #define G_GCAPS_SUBSCRIPTION_ERROR_S "function requires subscription not specified in GOpen()"
00099
00100 #endif //I_0DD3687F_47D0_454B_ADB2_CBAB0ED46FCE
00101
```

Index