Galil Widgets Quick Start

1 Widgets Overview

The Galil 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. Widgets are available for the following functions.

1. `GWComs`: Communications to Galil hardware including event-driven handling of asynchronous traffic.
2. `GWTerm`: A terminal for direct user interaction with the hardware.
3. `GWPoll`: A polling tool to display important data on screen.
4. `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.
5. `GWDatRec`: A data record visualization tool. Used to display controller status through user-configurable labels, "soft LEDs", and analog sliders.

2 Three Users

Galil Widgets has been designed to support three general user needs.

2.1 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 reguirement for writing code. This quick start

guide, and Microsoft Visual Studio Express is all that is needed to make a free application GUI with minimal effort.

2.2 The .Net developer, adding to preexisting 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. See the "galilwidgets_source_docs" documentation for detailed descriptions of each widget.

2.3 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. Review GalilWidgetsInterfaces.vb, GWComs.vb, and GWPoll.vb for an overview of how the Widgets work.

Note: Before opening the Designer for any Widget, build the project first. Some Widgets depend on the binaries of others.

3 Installation

3.1 gclib prerequisite

Galil Widgets requires the Galil gclib library to be installed. Download and run the executable installer, http://www.galil.com/sw/pub/all/doc/gclib/html/windows.html

Gclib should now be located at *C:\Program Files (x86)\Galil\gclib*

3.1.1 C++ Redistributable

gclib requires run-time components available in the Visual C++ Redistributable Packages for Visual Studio 2013. On machines that don't already have Visual Studio 2013 installed, the required files can be installed from Microsoft. Be sure to install the appropriate architecture (x86 or x64),
http://www.microsoft.com/en-us/download/details.aspx?id=40784

3.2 Galil Widgets files

The newest version of GalilWidgets can be downloaded here,
http://www.galil.com/sw/pub/win/galilwidgets/galilwidgets.html

See www.galil.com/sw/pub/all/rn/galilwidgets.html for the Galil Widgets release notes.

Unzip the GalilWidgets library into *C:\Program Files (x86)\Galil\galilwidgets*

The Galil Widgets directory should look something like...

```
C:\Program Files (x86)\Galil\galilwidgets>tree /a
Folder PATH listing for volume OS
Volume serial number is AE3F-6836
C:.
+---bin
|   +---x64
|   \---x86
+---doc
+---examples
|   +---cs
|   |   \---GWExample
|   |       \---GWExample
|   |           \---Properties
|   \---vb
|       \---GWExample
|           \---GWExample
|               \---My Project
+---source
|   +---GalilWidgets
|   |   \---My Project
|   \---TestForm
|       \---My Project
\---xml
    +---4000
    \---47100
```

4 Example details

The following step-by-step instructions use a DMC-4000 or DMC4103 as the hardware target. If using another controller, substitute the XML path for your controller. For example, for an RIO 471xx, see Screen 4. Contact Galil Applications Support or email softwaresupport@galilmc.com for help.

5 Step-By-Step, Visual Basic .Net

The following walkthough was performed on Microsoft Visual Studio 2012 Professional.

1. Open Microsoft Visual Studio 2012
2. Choose *File* | *New Project...*
3. Choose *Templates* | *Visual Basic* | *Windows*
4. Click *Windows Forms Application*
5. In the *Name:* field, type *GWExample*
6. Click *OK*
7. In the *Solution Explorer*, right click on *GWExample* and choose *Properties*
8. Click the *Compile* side tab. On *Target CPU* choose *x86*. Close the properties.
9. Save the project
10. Right click on Form1, choose *Properties*
11. In the *Text* field type *Galil Widgets Example*
12. In the *Size* field type *1000,850*
13. If the Toolbox is not visible, show it by choosing *View* | *Toolbox*

14. Right click in the toolbox and choose *Add Tab*
15. Type *Galil Widgets* for the Tab Name
16. Right click the *Galil Widgets* tab and click *Choose Items*
17. Click *Browse...* and Open *GalilWidgets.dll* at *C:\Program Files (x86)\Galil\galilwidgets\bin\x86*
18. Uncheck *GWDatRecLabel, GWDatRecLed, GWDatRecSlider, and GWListView*
19. Click *OK* (See Screen 1)
20. From the *Galil Widgets* tab, click on *GWComs*, and then click on *Form1* to place the control.
21. Do the same for *GWPoll, GWSettings, GWTerm,* and twice for *GWDatRec.*
22. Use the mouse to move and resize the controls to fit the window (See Screen 2)
23. Double click the title bar in *Form1* to switch to the code editor. Input the following code.

```
Public Class Form1
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        GwComs1.GWRegisterWidget(GwTerm1)
        GwComs1.GWRegisterWidget(GwDatRec1)
        GwComs1.GWRegisterWidget(GwDatRec2)
        GwComs1.GWRegisterWidget(GwPoll1)
        GwComs1.GWRegisterWidget(GwSettings1)
    End Sub
End Class
```

24. Type F5 to run the application, The GUI will be blank at this point.
25. Click the Magnifing Glass icon in the GWComs control to populate the available controllers.
26. Select the desired controller and click *Open.* Connection details will print in the GWComs status window.
27. In the polling window, click the Gear icon and choose *Load GWPoll file*. Choose *GWPoll.xml* from *C:\Program Files (x86)\Galil\galilwidgets\xml\4000*
28. Likewise, in the upper right DatRec control, choose GWDatRec_IO.xml
29. In the middle-left DatRec control, choose *GWDatRec_Axis_A.xml*
30. Finally, in the GWSetup control, choose GWSettings.xml
31. Type *MG TIME* in the upper pane of the GWTerm control. Press *Enter.* The window should now be populated (See Screen 3).
32. In order to automatically populate the GUI items, close the app and add the following to the end of the code before the *End Sub* line.

```
Dim XmlPath As String = "C:\Program Files (x86)\Galil\galilwidgets\xml\4000\"
GwPoll1.GWLoadFile(XmlPath & "GWPoll.xml")
GwDatRec2.GWLoadFile(XmlPath & "GWDatRec_IO.xml") 'assuming #2 is upper right
GwDatRec1.GWLoadFile(XmlPath & "GWDatRec_Axis_A.xml")
GwSettings1.GWLoadFile(XmlPath & "GWSettings.xml")
GwComs1.GWOpen("192.168.127.12") 'put the correct connection address here
```

33. Type F5 to restart the application.
34. The full source for this example can be found in the Galil Widgets package in *examples\vb\GWExample*

6 Step-by-step, C#

The following walkthough was performed on Microsoft Visual Studio 2012 Professional.

1. Open Microsoft Visual Studio 2012
2. Choose *File* | *New Project...*
3. Choose *Templates* | *Other Languages* | *Visual C#*
4. Click *Windows Forms Application*
5. In the *Name:* field, type *GWExample*
6. Click *OK*
7. In the *Solution Explorer*, right click on *GWExample* and choose *Properties*
8. Click the *Build* side tab. On *Platform Target* choose *x86*. Close the properties.
9. Save the project
10. Right click on Form1, choose *Properties*
11. In the *Text* field type *Galil Widgets Example*
12. In the *Size* field type *1000,850*
13. If the Toolbox is not visible, show it by choosing *View* | *Toolbox*
14. Right click in the toolbox and choose *Add Tab*
15. Type *Galil Widgets* for the Tab Name
16. Right click the *Galil Widgets* tab and click *Choose Items*
17. Click *Browse...* and Open *GalilWidgets.dll* at *C:\Program Files (x86)\Galil\galilwidgets\bin\x86*
18. Uncheck *GWDatRecLabel, GWDatRecLed, GWDatRecSlider, and GWListView*
19. Click *OK* (See Screen 1)
20. From the *Galil Widgets* tab, click on *GWComs*, and then click on *Form1* to place the control.
21. Do the same for *GWPoll, GWSettings, GWTerm,* and twice for *GWDatRec.*
22. Use the mouse to move and resize the controls to fit the window (See Screen 2)
23. Double click the title bar in *Form1* to switch to the code editor. Input the following code into the subroutine Form1_Load()

```
1. GalilWidgets.GalilWidget wi; //for casting to Widget Interface
2. wi = (GalilWidgets.GalilWidget) gwTerm1; gwComs1.GWRegisterWidget(ref wi);
3. wi = (GalilWidgets.GalilWidget) gwDatRec1; gwComs1.GWRegisterWidget(ref wi);
4. wi = (GalilWidgets.GalilWidget) gwDatRec2; gwComs1.GWRegisterWidget(ref wi);
5. wi = (GalilWidgets.GalilWidget) gwPoll1; gwComs1.GWRegisterWidget(ref wi);
6. wi = (GalilWidgets.GalilWidget) gwSettings1; gwComs1.GWRegisterWidget(ref wi);
```

24. Type F5 to run the application, The GUI will be blank at this point.
25. Click the Magnifing Glass icon in the GWComs control to populate the available controllers.
26. Select the desired controller and click *Open.* Connection details will print in the GWComs status window.
27. In the polling window, click the Gear icon and choose *Load GWPoll file*. Choose *GWPoll.xml* from *C:\Program Files (x86)\Galil\galilwidgets\xml\4000*
28. Likewise, in the upper right DatRec control, choose GWDatRec_IO.xml
29. In the middle-left DatRec control, choose *GWDatRec_Axis_A.xml*
30. Finally, in the GWSetup control, choose GWSettings.xml

31. Type *MG TIME* in the upper pane of the GWTerm control. Press *Enter.* The window should now be populated (See Screen 3).
32. In order to automatically populate the GUI items, close the app and add the following to the end of the Form1_Load() subroutine.

```
1.  string xmlPath = "C:\\Program Files (x86)\\Galil\\galilwidgets\\xml\\4000\\";
2.  gwPoll1.GWLoadFile(xmlPath + "GWPoll.xml");
3.  gwDatRec2.GWLoadFile(xmlPath + "GWDatRec_IO.xml"); //assuming #2 is upper right
4.  gwDatRec1.GWLoadFile(xmlPath + "GWDatRec_Axis_A.xml");
5.  gwSettings1.GWLoadFile(xmlPath + "GWSettings.xml");
6.  gwComs1.GWOpen("192.168.127.12"); //put the correct connection address here
```

33. Type F5 to restart the application.
34. *The full source for this example can be found in the Galil Widgets package in examples\cs\GWExample*

7 Appendix, GWPoll XML Format

See the xml directory of the Galil Widgets package for examples. The following defines the meaning of each xml tag.

Tag	Example	Description
<galilwidgets></galilwidgets>		The root node. All Galil Widgets data are between these tags.
<version></version>	<version>0</version>	Version of the Galil Widgets format.
<gwpoll></gwpoll>		All GWPoll data are between these tags.
<commands></commands>		Command listing. All GWPoll commands are between these tags.
<command></command>	<command>var1=?</command>	A single command for the polling window. This defines one row of the table. The command text is displayed in the "Command" column and the controller's response to this command is displayed in the "Value" column.

8 Appendix, GWSettings XML Format

See the xml directory of the Galil Widgets package for examples. The following defines the meaning of each xml tag.

Tag	Example	Description
<galilwidgets></galilwidgets>		The root node. All Galil Widgets data are between these tags.

<version></version>	<version>0</version>	Version of the Galil Widgets format.
<gwsettings></gwsettings>		All GWSettings data are between these tags.
<settings></settings>		Settings listing. All settings are between these tags.
<setting></setting>		A single setting. Defines one row of the GWSettings table.
<name></name>	<name>KP</name>	The name to be displayed for the setting. Display only, does not need to be a Galil command. The name is displayed in the "Setting" column.
<group></group>	<group>Filter</group>	The group to display the setting in. If the group does not already exist, a new settings subtable will be created with this label. Display only.
<description></description>	<description>Proportional Constants</description>	A description of the setting. Display only. The description is displayed in the "Description" column.
<get></get>	<get>KP*=?</get>	The actual command to use to poll the value of the setting. The reponse is displayed in the "Value" column.
<set></set>	<set>KP </set>	The prefix string for the command to use to set the value of the setting. The value will be appended to this prefix and then sent to the controller when setting the value. See exceptions info below.
<value></value>	<value>6.00, 6.00</value>	The desired value of this setting. When downloading settings, these value are sent to the controller.

8.1 Exceptional Cases for <set> tag

Most settings are set by appending the new value string on the end of the *set* string. However, some settings require a new value to map to a unique string. These cases are supported in the GWSettings Widget with the backslash syntax. An example is used below to clarify.

For axis A enabled state, the *get* command is *MG_MOA*. This returns a *1.0000* for motor disabled, and a *0.0000* for motor enabled. To set axis A to the disabled state, however, the command *MOA* must be sent. To enable, *SHA* is sent. This mapping is accomplished in GWSettings with the following setting XML element.

```
<setting>
  <name>MOA</name>
  <group>Filter/Motor</group>
  <description>A Axis Motor Off State</description>
  <get>MG_MOA{Z1.0}</get>
  <set>\1\MOA\0\SHA</set>
  <value>0, 0</value>
</setting>
```

The *{Z1.0}* in the get string formats the data as one digit only, e.g. removing the trailing decimal in 1.0000.

The *set* string is parsed as, "If the new value is 1, send "MOA", if the new value is 0, send "SHA". If no match is found, no command is sent.

9 Appendix, GWDatRec XML Format

See the xml directory of the Galil Widgets package for examples. The following defines the meaning of each xml tag.

All GWDatRec items require byte offsets to the data of interest. See the hardware user manual for a data record table, or see the byte offset comments in gclib_record.h in the *include* directory of the gclib installation.

Tag	Example	Description
`<galilwidgets></galilwidgets>`		The root node. All Galil Widgets data are between these tags.
`<version></version>`	`<version>0</version>`	Version of the Galil Widgets format.
`<gwdatrec></gwdatrec>`		All GWDatRec data are between these tags.
`<title></title>`	`<title>A Axis Info</title>`	The title to display in the groupbox around the data record items.
`<flow></flow>`	`<flow>vertical</flow>`	Items should flow top-down (vertical) or left-right (horizontal).
`<wrap></wrap>`	`<wrap>true</wrap>`	Wrap items (true) when the end of the control is reached, or continue flowing with a scrolbar (false).

<items></items>		Items listing. All GWDatRec items are between these tags.
<item></item>		A single data record item. See tables below for child elements.

9.1 Data record types

Labels and Sliders need to be told which data type to parse. The following table shows the data record indicator.

Type tag	Range	Example Galil data
<type>SignedLong</type>	-2147483648 to 2147483647	Encoder Position, TP
<type>SignedShort</type>	-32768 to 32767	+/-10 V, +/-5 V bipolar analog input, @AN[1]
<type>UnsignedShort</type>	0 to 65535	0 to 10 V, 0 to 5 V unipolar analog input, @AN[1], (see AQ)
<type>UnsignedByte</type>	0 to 255	Error code, TC

9.2 Label Item

Tag	Example	Description
<item></item>		Every GWDatRec item is in this tag.
<gui></gui>	<gui>label</gui>	Indicates the type of gui element and defines the expected tags.
<name></name>	<name>@AN[1]</name>	The name for the label. Display only.
<offset></offset>	<offset>110</offset>	The byte offset into the data record for this piece of information.
<type></type>	<type>SignedShort</type>	The data type to be parsed.

9.3 Soft Led Item

Tag	Example	Description
<item></item>		Every GWDatRec item is in this tag.
<gui></gui>	<gui>led</gui>	Indicates the type of gui element and defines the expected tags.
<name></name>	<name>@OUT[1]</name>	The name for the LED. Display only.
<offset></offset>	<offset>16</offset>	The byte offset into the data record for this piece of

		information.
<bit></bit>	<bit>0</bit>	The bit, 0-7 to extract from the offset byte.
<invert></invert>	<invert>false</invert>	Invert the display color. Default 0=Red, 1=Green.
<command></command>	<command>OB1,@COM[@OUT[1]]&1</command>	If non-null this command will be sent to the controller whenever the led is double clicked. The funny looking XML code at left translates to OB1,@COM[@OUT[1]]&1 Which toggles ouput 1.

9.4 Slider Item

The slider provides two sets of coefficients, m and b, following the linear function

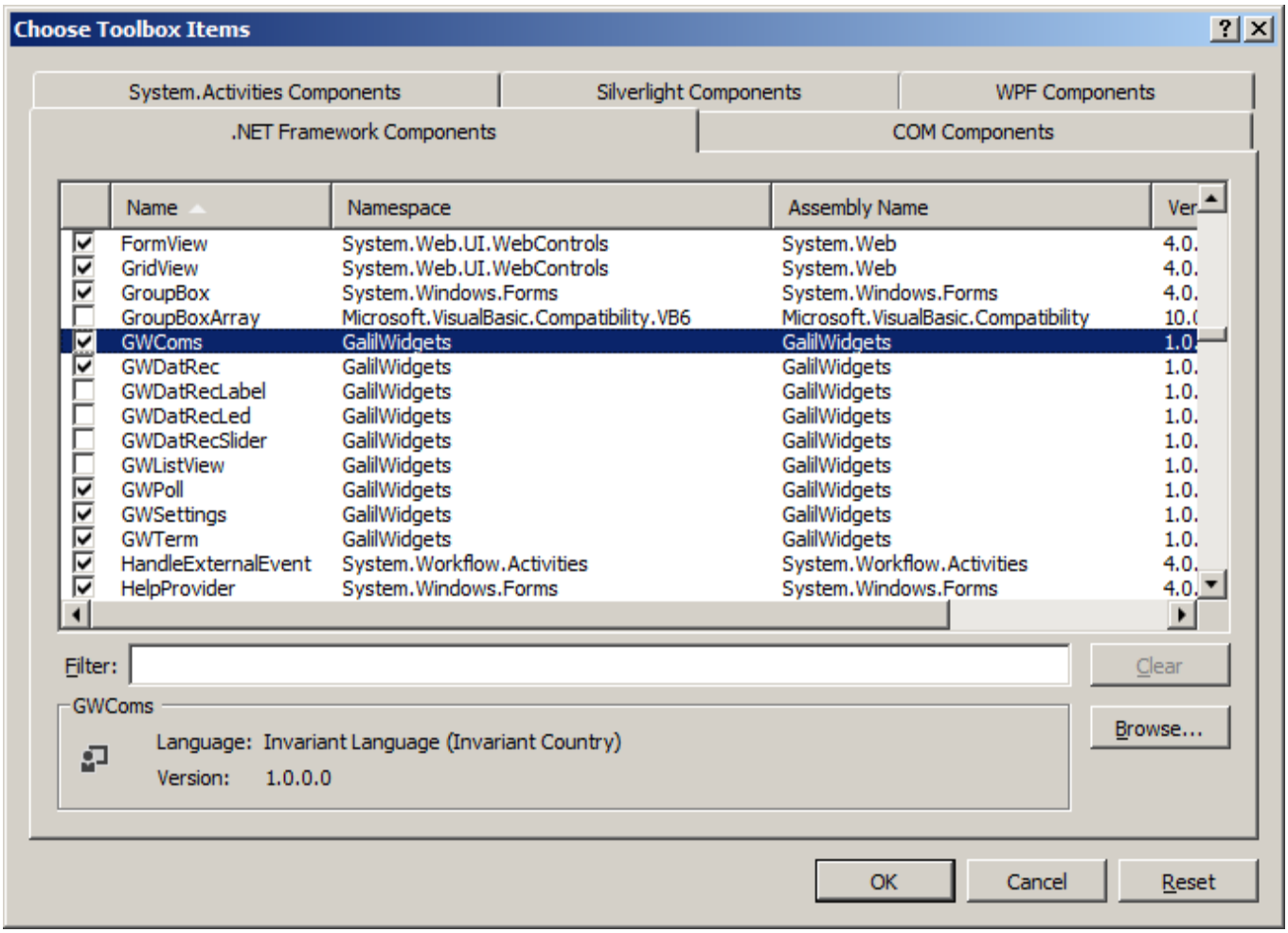
$$y = mx + b$$

- The Display pair provides for scaling the data out of the data record for displaying on the slider control. For example, on a typical +/-10v, signed 16 bit analog input, using a Display M of 0.0003052 will provide units of volts on the slider display.
- The Set pair provides for scaling the data set on the slider to a string that can be appended to the set command for sending to the controller. For example, on a typical +/-10v, 16 bit analog output, using a Set M of 0.0003052 and a Set B of -10 will provide units of volts to the AO command.

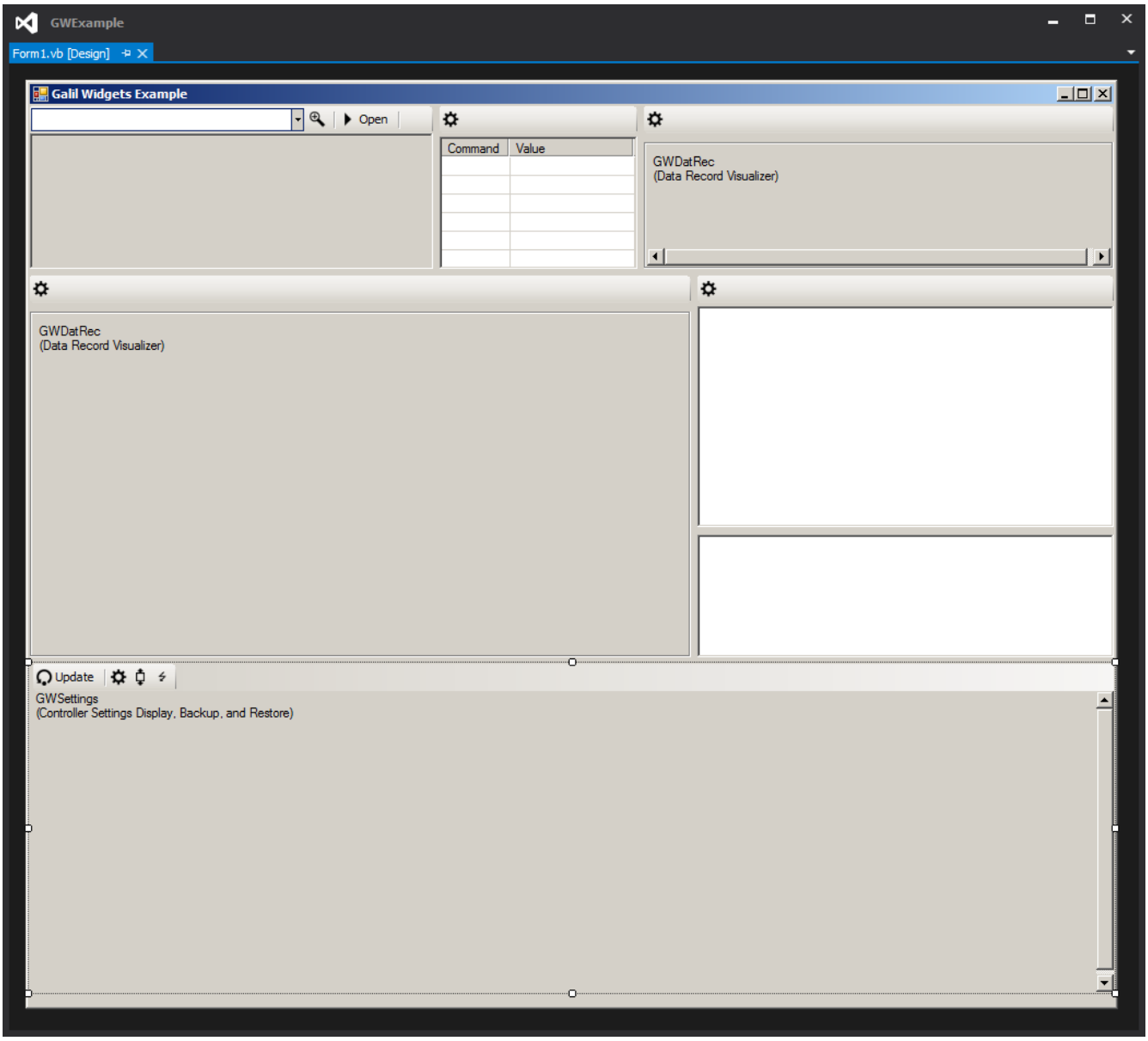
Tag	Example	Description
<item></item>		Every GWDatRec item is in this tag.
<gui></gui>	<gui>slider</gui>	Indicates the type of gui element and defines the expected tags.
<name></name>	<name>@AN[1]</name>	The name for the Slider. Display only.
<offset></offset>	<offset>110</offset>	The byte offset into the data record for this piece of information.
<type></type>	<type>SignedShort</type>	The data type to be parsed.
<dispm></dispm>	<dispm>0.0003052</dispm>	The slope of the display linear function.
<dispb></dispb>	<dispb>0</dispb>	The Y-intercept of the display

		linear function.
`<command></command>`	`<command></command>`	If this tag is not empty, the slider will be movable via mouse drag and the contents of this tag are the command prefix to be used to send the slider's value to the controller. The set M and B are applied to the slider value before being appended to the prefix.[1]
`<setm></setm>`	`<setm>1</setm>`	The slope of the set linear function.
`<setb></setb>`	`<setb>0</setb>`	The Y-intercept of the set linear function.
`<vertical></vertical>`	`<vertical>true</vertical>`	Display slider vertical (true) or horizontal (false).

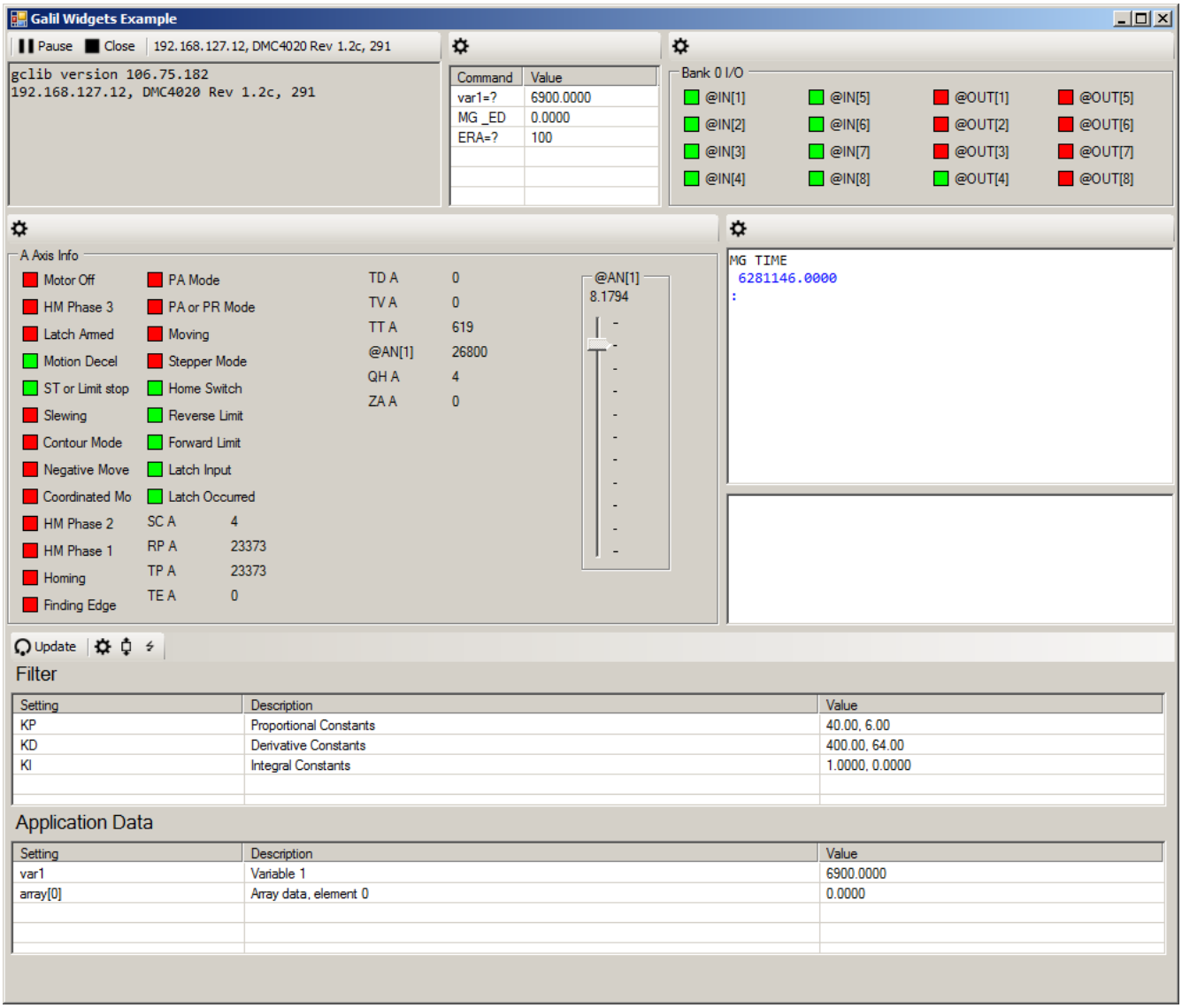
10 Appendix, Screen Shots



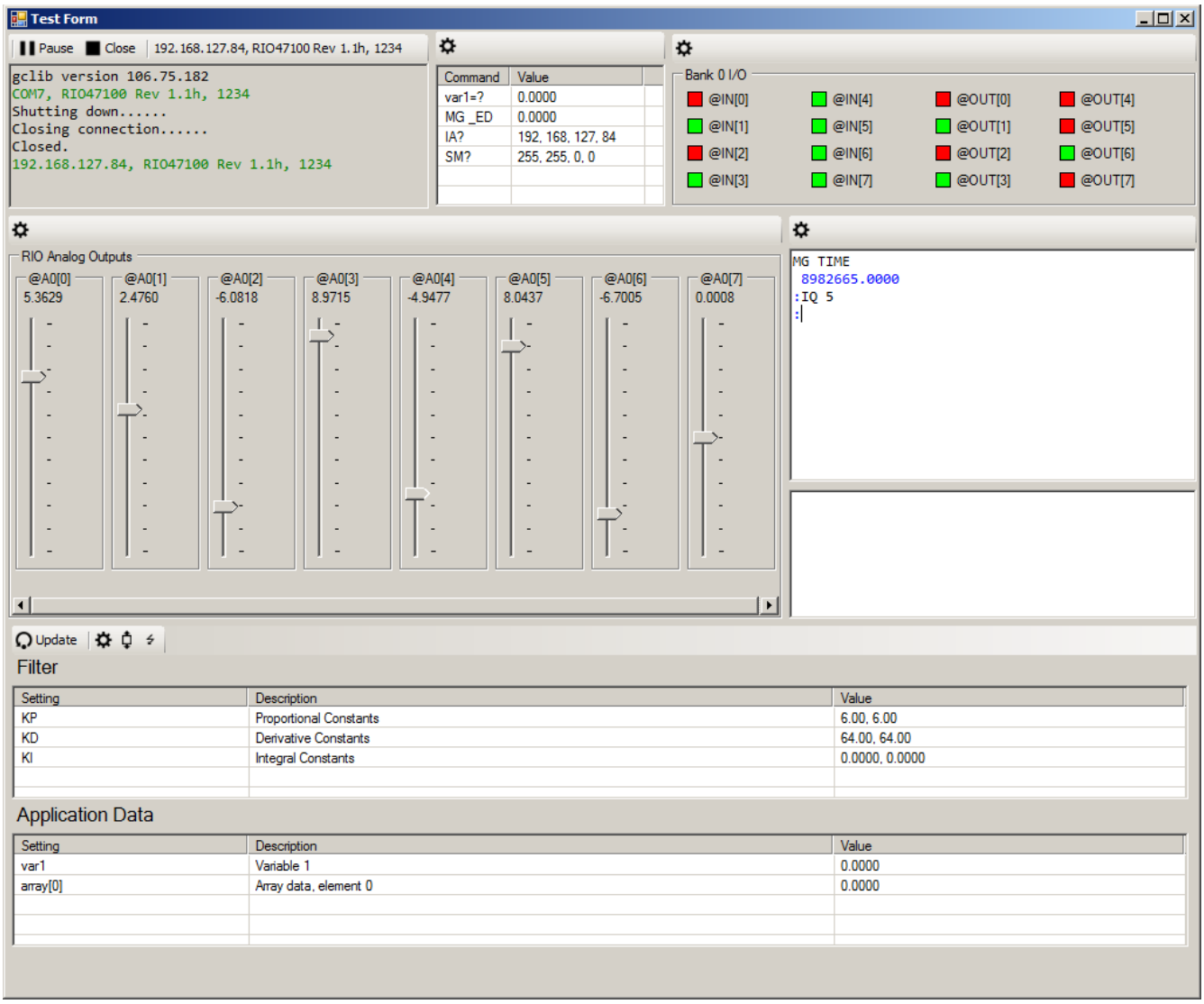
Screen 1: Selecting Widgets for the Toolbox



Screen 2: GUI Layout of Form1



Screen 3: GUI Populated



Screen 4: Example UI using RIO XML Settings