gclib 1.37.6

C API for Galil controllers and PLCs

Galil Motion Control

Tue Dec 13 2022

Contents

Chapter 1

Getting Started

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.

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

Contents

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.

Chapter 2

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.

2.1 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
```

2.2 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
```

2.3 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:
```

2.4 Jog Example

C++	C#	Visual Basic
Example	Example	Example

C++	C#	Visual Basic
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
```

2.5 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
```

2.6 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
```

2.7 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
```

2.8 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
```

2.9 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
```

2.10 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.

2.11 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
```

2.12 C/C++

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

- Microsoft Windows
- Linux

2.12.1 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.

2.12.2 Linux

Copy examples to a temporary directory

Create temporary directory:
```
mkdir ~/temp
```

Copy examples to ~/temp
```
cp /usr/share/doc/gclib/src/gclib_example_projects.tar.gz ~/temp
```

Navigate to ~/temp
```
cd ~/temp
```

Extract the examples with command:
```
tar -xzf gclib_example_projects.tar.gz
```

Run Make

To build all examples:
```
make
```

To build a single example:
```
make commands.o
```

To run an example:
```
./commands_example.out
```

2.13 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.

2.14 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 3

Installation

Supported Operating Systems

- Most versions are 64 bit (x64). A 32 bit (x86) version of gclib is also available on Windows.
- Older operating systems are listed along with the last Galil builds. On Linux, install these packages in this order `gclib -> gcapsd -> GDK`.

	Microsoft Windows				
	11				
	10				
	8.1, *No Galil Connect*				
	Older Versions	*gclib*	*gcaps*	*GDK*	
	7	245.331.480	1.0.0.151	1.0.24.655	
	Ubuntu				
	20.04				
	18.04				
	Older Versions	*gclib*	*gcaps*	*GDK*	
	16.04	200.252.396	1.0.0.130	1.0.7.353	
	14.04	189.224.370			
	12.04	130.114.278			
	Red Hat and CentOS				
	Red Hat 8 & CentOS 8 Linux				
	Red Hat 7 & CentOS 7 Linux				
	Older Versions	*gclib*	*gcaps*	*GDK*	
	Red Hat 6 & CentOS 6 Linux	147.185.319			
	Fedora				
	32				
	31				
	Older Versions	*gclib*	*gcaps*	*GDK*	
	30	267.339.528	1.0.0.210	1.0.25.696	
	29	239.316.448	1.0.0.149	1.0.18.601	
	28	222.301.423	1.0.0.143	1.0.13.526	
	27	200.252.396	1.0.0.130	1.0.7.353	
	26	195.232.387	1.0.0.128	1.0.5.307	
	25	194.232.386	1.0.0.128	1.0.5.305	
	24	189.224.370	1.0.0.124	1.0.4.276	
	23	154.190.329	1.0.0.68		
	22	130.114.278			
	21	130.114.278			

	Microsoft Windows **Raspberry Pi**				
	Buster, Raspberry Pi 4				
	Stretch, Raspberry Pi 3+				
	Older Versions	*gclib*	*gcaps*	*GDK*	
	Jessie, Raspberry Pi 2	194.232.386			
	Apple				
	Older Versions	*gclib*	*gcaps*	*GDK*	
	Yosemite	189.220.368			

3.1 Microsoft Windows

Tested versions

See the installation page for supported versions.

Install

On Windows, gclib is distributed in the following formats.

- An executable installer which will install the library in the proper location to work with the included examples and documentation. PCI users can optionally install the PCI driver from within this installer.
- A zip file containing the same set of files as the executable but in a zip archive. PCI users can use the stand-alone PCI driver installer.
  - A stand-alone PCI driver installer for PCI users (DMC-1806, 1800, 1802, 1417).

Note

The PCI driver is compatible with GalilTools but is enhanced for gclib/GDK communications.

Download Installer

Recommended. All instructions and examples depend on the installation paths.

Download Zip

For custom deployment or non-default file locations. Downloads are available on the release notes page.

Required third-party DLLs

gclib is built using **MSVC 2017** and requires run-time components available in the Microsoft Visual C++ Redistributable for Visual Studio 2017
The gclib installer will automatically install these prerequisites for both 32 bit (x86) and 64 bit (x64) builds.
If using the zip installation, the binaries must be downloaded and installed manually.

Silent Installation

For developers wishing to bundle gclib within their own installers, execute the gclib installer with the /S switch to run silently with defaults. If the Galil security certificate is not already trusted on the deployment target, a digital signature dialog may be presented.

Uninstall

Run `uninstall.exe` in "C:\Program Files (x86)\Galil\gclib"
Alternatively, the *Add or remove programs* Control Panel can be used to remove all Galil software by product.

Upgrading and Older Versions

The release notes page lists all available versions of gclib. Galil recommends uninstalling any current version of gclib before installing other versions.

Installed Directories

The installation is organized into several directories.

dll

The *dll* directory contains the binary *dynamic link libraries* (DLLs) for x86 and/or x64 architectures. **Dynamically linked executables must have the correct dlls in their path at runtime**.

doc

The *doc* directory contains this documentation and a printable, pdf version.

examples

The *examples* directory contains example projects for various compilers. The *cpp* directory contains *x_examples.h* and the implementation of the example files documented in this manual.

Warning

Before using the examples, copy the files to a user location such as *C:\Users\user\Documents*. Failing to do so may cause source files to be deleted upon gclib uninstallation.

include

The *include* directory contains header files needed for compiling code. The compiler will need to know where these files are at compile time.
See the compiler-specific directions for more information, e.g. mingw.

lib

The *lib* directory contains linker files (*gclib.lib* and *gclibo.lib*) for x86 and/or x64 architectures. The linker should include *gclib.lib* and *gclibo.lib*.

source

The *source* directory contains source files such as gclibo.c.

3.2 Ubuntu Linux

Tested versions

See the installation page for supported versions.

Installation

Note

Adding the `package repository` is a prerequisite to continue.

Install

Install gclib and gcapsd (recommended)

```
# apt install gclib gcapsd
```

Uninstall

If gclib is to be removed from the system, the following commands may be used.

```
# apt remove gclib gcapsd
```

Upgrading

To upgrade gclib to the newest release, use the following command.

```
# apt install --only-upgrade gclib gcapsd
```

List All Versions

Galil keeps older versions of gclib and gcapsd available for users. To list all versions use the following command.

```
$ apt-cache madison gclib gcapsd
```

Installing Older Versions

Warning

> When using gcaps, a compatible pairing of gcaps and gclib must be used. Galil maintains this compatibility with installations and upgrades. Installing GDK will also install compatible versions of gclib and gcaps. When installing older versions manually, it is the user's responsibility to ensure compatible versions.

Append the desired version's information after the package name.

```
# apt install gclib=<version> gcapsd=<version>
```

An Example

On the developer's machine, gclib is installed with the current version.

```
$ sudo apt install gclib gcapsd
```

After installation, the versions can be queried.

```
$ apt list --installed gclib gcapsd
Listing... Done
gcapsd/unknown,now 205-1 amd64 [installed]
gclib/unknown,now 517-1 amd64 [installed]
```

On the deployment machine, the precise versions can now be specified.

```
$ sudo apt install gclib=517-1 gcapsd=205-1
```

Serial Ports and USB

If access to the serial ports or USB (e.g. DMC-4103) is desired through gclib, the following will provide steps to join the correct access group. If using USB, be sure the controller is powered and the usb is plugged in before beginning.

Determine group with access

```
$ ls -l /dev/ttyUSB* /dev/ttyS*
crw-rw----. 1 root dialout   4, 64 Mar  3 16:39 /dev/ttyS0
crw-rw----. 1 root dialout   4, 65 Mar  3 16:39 /dev/ttyS1
crw-rw----. 1 root dialout   4, 66 Mar  3 16:39 /dev/ttyS2
crw-rw----. 1 root dialout   4, 67 Mar  3 16:39 /dev/ttyS3
crw-rw----. 1 root dialout 188,  0 Mar  6 11:08 /dev/ttyUSB0
```

In the above listing, **dialout** is the group that needs to be joined. **uucp** is another common group that may be listed.

Add the desired *username* to the group.

```
$ sudo gpasswd -a username dialout
[sudo] password for username:
Adding user username to group dialout
```

Log out and back in for change to take effect.

```
$ groups
username wheel dialout
```

gclib can now connect to serial and usb devices from user *username*.

PCI Controllers

If using a Galil PCI controller, the PCI driver must be installed.

Extract source and build driver

```
$ tar -xf /usr/share/doc/gclib/src/gclib_pci.tar.gz
$ make
```

Copy module and add to kernel

```
$ sudo cp galilpci.ko /lib/modules/$(uname -r)
$ sudo depmod
$ sudo modprobe galilpci
```

Add galil group for access to PCI

```
$ sudo groupadd -f -K GID_MIN=100 -K GID_MAX=499 galil
$ sudo cp 90-galilpci.rules /etc/udev/rules.d/
$ sudo udevadm control --reload-rules
$ sudo udevadm trigger
$ sudo usermod -a -G galil username  #exchange "username" with actual user's name
```

Logout and back in. The PCI hardware is now available for access.

```
$ ls -l /dev/galil*
crw-rw---- 1 root galil 10, 56 Jun  9 11:07 /dev/galilpci0
$ echo -e "\x12\x16\r" > /dev/galilpci0
$ cat /dev/galilpci0
DMC1846 Rev 1.1a
:
```

Documentation

The documentation is left as a tarball to minimize disk usage. The latest release version of the user manual is available at the following link.

- http://www.galil.com/sw/pub/all/doc/gclib/html/

Offline html

The following allows viewing of the html docs from the installation.

```
$ tar -xzf /usr/share/doc/gclib/gclib_doc.tar.gz html
$ firefox html/index.html
```

3.3 Red Hat 8 & CentOS 8 Linux

Tested versions

See the installation page for supported versions.

Installation

Note

Adding the package repository is a prerequisite to continue.

Install

Install gclib and gcapsd (recommended)

```
# dnf install gclib gcapsd
```

Approve *Installed size* and *Importing GPG key*, if prompted.

Galil Connect

The nss-mdns package must be installed from the EPEL repository in order to use Galil Connect.
To use Galil Connect, follow the instructions linked below to enable EPEL:
https://docs.fedoraproject.org/en-US/epel/
Then use the following command to install nss-mdns:

```
$ sudo dnf install nss-mdns
```

Uninstall

If gclib is to be removed from the system, the following commands may be used.

```
# dnf remove gclib gcapsd
```

Upgrading

To upgrade gclib to the newest release, use the following command.

```
# dnf upgrade gclib gcapsd
```

List All Versions

Galil keeps older versions of gclib and gcapsd available for users. To list all versions use the following command.

```
$ dnf list gclib gcapsd --showduplicates
```

Installing Older Versions

Warning

When using gcaps, a compatible pairing of gcaps and gclib must be used. Galil maintains this compatibility with installations and upgrades. Installing GDK will also install compatible versions of gclib and gcaps. When installing older versions manually, it is the user's responsibility to ensure compatible versions.

Append the desired version's information after the package name.

```
# dnf install gclib-<version> gcapsd-<version>
```

An Example

On the developer's machine, gclib is installed with the current version.

```
$ sudo dnf install gclib gcapsd
```

After installation, the versions can be queried.

```
$ dnf list gclib gcapsd --installed
Installed Packages
gcapsd.x86_64                    194-1                     @galil
gclib.x86_64                     506-1                     @galil
```

On the deployment machine, the precise versions can now be specified.

```
$ sudo dnf install gclib-506-1 gcapsd-194-1
```

Serial Ports and USB

If access to the serial ports or USB (e.g. DMC-4103) is desired through gclib, the following will provide steps to join the correct access group. If using USB, be sure the controller is powered and the usb is plugged in before beginning.

Determine group with access

```
$ ls -l /dev/ttyUSB* /dev/ttyS*
crw-rw----. 1 root dialout   4, 64 Mar  3 16:39 /dev/ttyS0
crw-rw----. 1 root dialout   4, 65 Mar  3 16:39 /dev/ttyS1
crw-rw----. 1 root dialout   4, 66 Mar  3 16:39 /dev/ttyS2
crw-rw----. 1 root dialout   4, 67 Mar  3 16:39 /dev/ttyS3
crw-rw----. 1 root dialout 188,  0 Mar  6 11:08 /dev/ttyUSB0
```

In the above listing, **dialout** is the group that needs to be joined. **uucp** is another common group that may be listed.

Add the desired *username* to the group.

```
$ sudo gpasswd -a username dialout
[sudo] password for username:
Adding user username to group dialout
```

Log out and back in for change to take effect.

```
$ groups
username wheel dialout
```

gclib can now connect to serial and usb devices from user *username*.

PCI Controllers

If using a Galil PCI controller, the PCI driver must be installed.

Install prerequisites

```
# dnf update kernel
```

Reboot

```
# dnf install kernel-devel-$(uname -r)
# dnf install kernel-headers-$(uname -r)
# dnf install gcc
```

Extract source and build driver

```
$ tar -xf /usr/share/doc/gclib/src/gclib_pci.tar.gz
$ make
```

Copy module and add to kernel

```
# cp galilpci.ko /lib/modules/$(uname -r)
# depmod
# modprobe galilpci
```

Add galil group for access to PCI

```
# groupadd -f -K GID_MIN=100 -K GID_MAX=499 galil
# cp 90-galilpci.rules /etc/udev/rules.d/
# udevadm control --reload-rules
# udevadm trigger
# usermod -a -G galil username  #exchange "username" with actual user's name
```

Logout and back in. The PCI hardware is now available for access.

```
$ ls -l /dev/galil*
crw-rw---- 1 root galil 10, 56 Jun  9 11:07 /dev/galilpci0
$ echo -e "\x12\x16\r" > /dev/galilpci0
$ cat /dev/galilpci0
DMC1846 Rev 1.1a
:
```

Documentation

The documentation is left as a tarball to minimize disk usage. The latest release version of the user manual is available at the following link.

- http://www.galil.com/sw/pub/all/doc/gclib/html/

Offline html

The following allows viewing of the html docs from the installation.

```
$ tar -xzf /usr/share/doc/gclib/gclib_doc.tar.gz html
$ firefox html/index.html
```

3.4 Red Hat 7 & CentOS 7 Linux

Tested versions

See the installation page for supported versions.

Installation

Note

Adding the package repository is a prerequisite to continue.

Install

Install gclib and gcapsd (recommended)

```
# yum install gclib gcapsd
```

Approve *Installed size* and *Importing GPG key*, if prompted.

Galil Connect

The nss-mdns package must be installed from the EPEL repository in order to use Galil Connect.
To use Galil Connect, follow the instructions linked below to enable EPEL:
https://docs.fedoraproject.org/en-US/epel/
Then use the following command to install nss-mdns:

```
$ sudo yum install nss-mdns
```

Uninstall

If gclib is to be removed from the system, the following commands may be used.

```
# yum remove gclib gcapsd
```

Upgrading

To upgrade gclib to the newest release, use the following command.

```
# yum upgrade gclib gcapsd
```

List All Versions

Galil keeps older versions of gclib and gcapsd available for users. To list all versions use the following command.

```
$ yum list gclib gcapsd --showduplicates
```

Installing Older Versions

Warning

When using gcaps, a compatible pairing of gcaps and gclib must be used. Galil maintains this compatibility with installations and upgrades. Installing GDK will also install compatible versions of gclib and gcaps. When installing older versions manually, it is the user's responsibility to ensure compatible versions.

Append the desired version's information after the package name.

```
# yum install gclib-<version> gcapsd-<version>
```

An Example

On the developer's machine, gclib is installed with the current version.

```
# yum install gclib gcapsd
```

After installation, the versions can be queried.

```
$ yum list installed gclib gcapsd
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
* base: centos-distro.lgservers.com
* epel: sjc.edge.kernel.org
* extras: mirror.shastacoe.net
* updates: mirrors.oit.uci.edu
Installed Packages
gcapsd.x86_64                                          194-1                                          @galil
gclib.x86_64                                           506-1                                          @galil
```

On the deployment machine, the precise versions can now be specified.

```
# yum install gclib-506-1 gcapsd-194-1
```

If a newer version was previously installed, downgrade with the following.

```
# yum downgrade gclib-506-1 gcapsd-194-1
```

Serial Ports and USB

If access to the serial ports or USB (e.g. DMC-4103) is desired through gclib, the following will provide steps to join the correct access group. If using USB, be sure the controller is powered and the usb is plugged in before beginning.

Determine group with access

```
$ ls -l /dev/ttyUSB* /dev/ttyS*
crw-rw----. 1 root dialout   4, 64 Mar  3 16:39 /dev/ttyS0
crw-rw----. 1 root dialout   4, 65 Mar  3 16:39 /dev/ttyS1
crw-rw----. 1 root dialout   4, 66 Mar  3 16:39 /dev/ttyS2
crw-rw----. 1 root dialout   4, 67 Mar  3 16:39 /dev/ttyS3
crw-rw----. 1 root dialout 188,  0 Mar  6 11:08 /dev/ttyUSB0
```

In the above listing, **dialout** is the group that needs to be joined. **uucp** is another common group that may be listed.

Add the desired *username* to the group.

```
$ sudo gpasswd -a username dialout
[sudo] password for username:
Adding user username to group dialout
```

Log out and back in for change to take effect.

```
$ groups
username wheel dialout
```

gclib can now connect to serial and usb devices from user *username*.

PCI Controllers

If using a Galil PCI controller, the PCI driver must be installed.

Install prerequisites

```
# yum update kernel
```

Reboot

```
# yum install kernel-devel-$(uname -r)
# yum install kernel-headers-$(uname -r)
# yum install gcc
```

Extract source and build driver

```
$ tar -xf /usr/share/doc/gclib/src/gclib_pci.tar.gz
$ make
```

Copy module and add to kernel

```
# cp galilpci.ko /lib/modules/$(uname -r)
# depmod
# modprobe galilpci
```

Add galil group for access to PCI

```
# groupadd -f -K GID_MIN=100 -K GID_MAX=499 galil
# cp 90-galilpci.rules /etc/udev/rules.d/
# udevadm control --reload-rules
# udevadm trigger
# usermod -a -G galil username  #exchange "username" with actual user's name
```

Logout and back in. The PCI hardware is now available for access.

```
$ ls -l /dev/galil*
crw-rw---- 1 root galil 10, 56 Jun  9 11:07 /dev/galilpci0
$ echo -e "\x12\x16\r" > /dev/galilpci0
$ cat /dev/galilpci0
DMC1846 Rev 1.1a
:
```

Documentation

The documentation is left as a tarball to minimize disk usage. The latest release version of the user manual is available at the following link.

- http://www.galil.com/sw/pub/all/doc/gclib/html/

Offline html

The following allows viewing of the html docs from the installation.

```
$ tar -xzf /usr/share/doc/gclib/gclib_doc.tar.gz html
$ firefox html/index.html
```

3.5 Red Hat 6 & CentOS 6 Linux

Tested versions

This version of Linux has **x64/AMD64 Support Only**. Contact Galil if another version is required for an application. See the installation page for supported versions.

Installation

On Red Hat, gclib is distributed in an RPM repository. The following steps can be performed to install gclib.

Download Galil's repository information

This step installs Galil's RPM repositories and only needs to be done once.
Point a browser at http://www.galil.com/sw/pub/rhel/6/galilrpm-2-1.noarch.rpm and install the rpm.

Install Package

Install gclib package, approve "Installed size" and "Importing GPG key", if prompted.

```
# yum install gclib
```

Uninstall Package

To uninstall gclib.

```
# yum remove gclib
```

Serial Ports and USB

If access to the serial ports or USB (e.g. DMC-4103) is desired through gclib, the following will provide steps to join the correct access group. If using USB, be sure the controller is powered and the usb is plugged in before beginning.

Determine group with access

```
$ ls -l /dev/ttyUSB* /dev/ttyS*
crw-rw----. 1 root dialout   4, 64 Mar  3 16:39 /dev/ttyS0
crw-rw----. 1 root dialout   4, 65 Mar  3 16:39 /dev/ttyS1
crw-rw----. 1 root dialout   4, 66 Mar  3 16:39 /dev/ttyS2
crw-rw----. 1 root dialout   4, 67 Mar  3 16:39 /dev/ttyS3
crw-rw----. 1 root dialout 188,  0 Mar  6 11:08 /dev/ttyUSB0
```

In the above listing, **dialout** is the group that needs to be joined. **uucp** is another common group that may be listed.

Add the desired *username* to the group.

```
$ sudo gpasswd -a username dialout
[sudo] password for username:
Adding user username to group dialout
```

Log out and back in for change to take effect.

```
$ groups
username wheel dialout
```

gclib can now connect to serial and usb devices from user *username*.

PCI Controllers

If using a Galil PCI controller, the PCI driver must be installed.

Install prerequisites

```
# yum update kernel
```

Reboot

```
# yum install kernel-devel-$(uname -r)
# yum install kernel-headers-$(uname -r)
# yum install gcc
```

Extract source and build driver

```
$ tar -xf /usr/share/doc/gclib/src/gclib_pci.tar.gz
$ make
```

Copy module and add to kernel

```
# cp galilpci.ko /lib/modules/$(uname -r)
# depmod
# modprobe galilpci
```

Add galil group for access to PCI

```
# groupadd -f -K GID_MIN=100 -K GID_MAX=499 galil
# cp 90-galilpci.rules /etc/udev/rules.d/
# udevadm control --reload-rules
# udevadm trigger
# usermod -a -G galil username  #exchange "username" with actual user's name
```

Logout and back in. The PCI hardware is now available for access.

```
$ ls -l /dev/galil*
crw-rw---- 1 root galil 10, 56 Jun  9 11:07 /dev/galilpci0
$ echo -e "\x12\x16\r" > /dev/galilpci0
$ cat /dev/galilpci0
DMC1846 Rev 1.1a
:
```

Documentation

The documentation is left as a tarball to minimize disk usage. The latest release version of the user manual is available at the following link.

- http://www.galil.com/sw/pub/all/doc/gclib/html/

Offline html

The following allows viewing of the html docs from the installation.

```
$ tar -xzf /usr/share/doc/gclib/gclib_doc.tar.gz html
$ firefox html/index.html
```

3.6 Fedora Linux

Tested versions

See the installation page for supported versions.

Installation

Note

Adding the package repository is a prerequisite to continue.

Install

Install gclib and gcapsd (recommended)

```
# dnf install gclib gcapsd
```

Approve *Installed size* and *Importing GPG key*, if prompted.

Uninstall

If gclib is to be removed from the system, the following commands may be used.

```
# dnf remove gclib gcapsd
```

Upgrading

To upgrade gclib to the newest release, use the following command.

```
# dnf upgrade gclib gcapsd
```

List All Versions

Galil keeps older versions of gclib and gcapsd available for users. To list all versions use the following command.

```
$ dnf list gclib gcapsd --showduplicates
```

Installing Older Versions

Warning

When using gcaps, a compatible pairing of gcaps and gclib must be used. Galil maintains this compatibility with installations and upgrades. Installing GDK will also install compatible versions of gclib and gcaps. When installing older versions manually, it is the user's responsibility to ensure compatible versions.

Append the desired version's information after the package name.

```
# dnf install gclib-<version> gcapsd-<version>
```

An Example

On the developer's machine, gclib is installed with the current version.

```
$ sudo dnf install gclib gcapsd
```

After installation, the versions can be queried.

```
$ dnf list gclib gcapsd --installed
Installed Packages
gcapsd.x86_64                       194-1                       @galil-gcapsd
gclib.x86_64                        506-1                       @galil-gclib
```

On the deployment machine, the precise versions can now be specified.

```
$ sudo dnf install gclib-506-1 gcapsd-194-1
```

Serial Ports and USB

If access to the serial ports or USB (e.g. DMC-4103) is desired through gclib, the following will provide steps to join the correct access group. If using USB, be sure the controller is powered and the usb is plugged in before beginning.

Determine group with access

```
$ ls -l /dev/ttyUSB* /dev/ttyS*
crw-rw----. 1 root dialout   4, 64 Mar  3 16:39 /dev/ttyS0
crw-rw----. 1 root dialout   4, 65 Mar  3 16:39 /dev/ttyS1
crw-rw----. 1 root dialout   4, 66 Mar  3 16:39 /dev/ttyS2
crw-rw----. 1 root dialout   4, 67 Mar  3 16:39 /dev/ttyS3
crw-rw----. 1 root dialout 188,  0 Mar  6 11:08 /dev/ttyUSB0
```

In the above listing, **dialout** is the group that needs to be joined. **uucp** is another common group that may be listed.

Add the desired *username* to the group.

```
$ sudo gpasswd -a username dialout
[sudo] password for username:
Adding user username to group dialout
```

Log out and back in for change to take effect.

```
$ groups
username wheel dialout
```

gclib can now connect to serial and usb devices from user *username*.

PCI Controllers

If using a Galil PCI controller, the PCI driver must be installed.

Install prerequisites

```
$ sudo dnf install kernel-devel-$(uname -r)
$ sudo dnf install kernel-headers-$(uname -r)
$ sudo dnf install gcc
```

Extract source and build driver

```
$ tar -xf /usr/share/doc/gclib/src/gclib_pci.tar.gz
$ make
```

Copy module and add to kernel

```
$ sudo cp galilpci.ko /lib/modules/$(uname -r)
$ sudo depmod
$ sudo modprobe galilpci
```

Add galil group for access to PCI

```
$ sudo groupadd -f -K GID_MIN=100 -K GID_MAX=499 galil
$ sudo cp 90-galilpci.rules /etc/udev/rules.d/
$ sudo udevadm control --reload-rules
$ sudo udevadm trigger
$ sudo usermod -a -G galil username  #exchange "username" with actual user's name
```

Logout and back in. The PCI hardware is now available for access.

```
$ ls -l /dev/galil*
crw-rw---- 1 root galil 10, 56 Jun  9 11:07 /dev/galilpci0
$ echo -e "\x12\x16\r" > /dev/galilpci0
$ cat /dev/galilpci0
DMC1846 Rev 1.1a
:
```

Documentation

The documentation is left as a tarball to minimize disk usage. The latest release version of the user manual is available at the following link.

- http://www.galil.com/sw/pub/all/doc/gclib/html/

Offline html

The following allows viewing of the html docs from the installation.

```
$ tar -xzf /usr/share/doc/gclib/gclib_doc.tar.gz html
$ firefox html/index.html
```

3.7 Raspberry Pi

Tested versions

See the installation page for supported versions.

Installation

Note

Adding the `package repository` is a prerequisite to continue.

Install

Install gclib and gcapsd (recommended)

```
# apt install gclib gcapsd
```

Note

See the Remote Server Example to make your Galil controllers easily accessible to other computers on your network.

Uninstall

If gclib is to be removed from the system, the following commands may be used.

```
# apt remove gclib gcapsd
```

Upgrading

To upgrade gclib to the newest release, use the following command.

```
# apt install --only-upgrade gclib gcapsd
```

List All Versions

Galil keeps older versions of gclib and gcapsd available for users. To list all versions use the following command.

```
$ apt-cache madison gclib gcapsd
```

Installing Older Versions

Warning

When using gcaps, a compatible pairing of gcaps and gclib must be used. Galil maintains this compatibility with installations and upgrades. Installing GDK will also install compatible versions of gclib and gcaps. When installing older versions manually, it is the user's responsibility to ensure compatible versions.

Append the desired version's information after the package name.

```
# apt install gclib=<version> gcapsd=<version>
```

An Example

On the developer's machine, gclib is installed with the current version.

```
$ sudo apt install gclib gcapsd
```

After installation, the versions can be queried.

```
$ apt list --installed gclib gcapsd
Listing... Done
gcapsd/unknown,now 205-1 amd64 [installed]
gclib/unknown,now 517-1 amd64 [installed]
```

On the deployment machine, the precise versions can now be specified.

```
$ sudo apt install gclib=517-1 gcapsd=205-1
```

Serial Ports and USB

If access to the serial ports or USB (e.g. DMC-4103) is desired through gclib, the following will provide steps to join the correct access group. If using USB, be sure the controller is powered and the usb is plugged in before beginning.

Determine group with access

```
$ ls -l /dev/ttyUSB* /dev/ttyS*
crw-rw----. 1 root dialout   4, 64 Mar  3 16:39 /dev/ttyS0
crw-rw----. 1 root dialout   4, 65 Mar  3 16:39 /dev/ttyS1
crw-rw----. 1 root dialout   4, 66 Mar  3 16:39 /dev/ttyS2
crw-rw----. 1 root dialout   4, 67 Mar  3 16:39 /dev/ttyS3
crw-rw----. 1 root dialout 188,  0 Mar  6 11:08 /dev/ttyUSB0
```

In the above listing, **dialout** is the group that needs to be joined. **uucp** is another common group that may be listed.

Check the user's group

The default *pi* username is already a member of dialout.

```
$ groups
pi adm dialout cdrom sudo audio video plugdev games users input netdev gpio i2c spi
```

If needed, add the desired *username* to the group.

```
$ sudo gpasswd -a username dialout
[sudo] password for username:
Adding user username to group dialout
```

Log out and back in for change to take effect.

```
$ groups
username wheel dialout
```

gclib can now connect to serial and usb devices from user *username*.

Documentation

The documentation is left as a tarball to minimize disk usage. The latest release version of the user manual is available at the following link.

- http://www.galil.com/sw/pub/all/doc/gclib/html/

Offline html

The following allows viewing of the html docs from the installation, in the GUI mode.

```
$ tar -xzf /usr/share/doc/gclib/gclib_doc.tar.gz html
$ epiphany html/index.html
```

3.8 Apple OS X

Tested versions

See the installation page for supported versions.

Warning

The latest Apple support is for *OSX 10.10, Yosemite*. Please contact Galil if your application requires a current version of Apple software.

Installation

On OS X, gclib is distributed in a dmg image. The following steps can be performed to install gclib.

Download the gclib dmg

- Open the dmg file and drag the gclib directory to the Applications alias or another installation location.

Create Environment Variable (Optional)

- To provide maximum functionality, e.g. usage of the Python wrapper, add to the *DYLD_LIBRARY_PATH* by typing the following at a Terminal prompt.

```
$ echo "export DYLD_LIBRARY_PATH=/Applications/gclib/dylib/:\$DYLD_LIBRARY_PATH" >> ~/.profile
```

- Log Out and back in to set the environment variable.

Make links for usb devices

If using the DMC4103 or another Galil USB product, symbolic links may be created so GAddresses() can list the controllers.
Make a link from the Terminal.

```
user-mac:~ user$ #plug in DMC4103 usb cable
user-mac:~ user$ ls /dev/tty.usb*
/dev/tty.usbserial-A402L6KG
user-mac:~ user$ #make a symbolic link so gclib can list it
user-mac:~ user$ sudo ln -s /dev/tty.usbserial-A402L6KG /dev/tty.usbserial0
user-mac:~ user$ #gclib searches start at 0
user-mac:~ user$ #GAddresses() will now list this device
```

Demonstrating with Python.

```
user-mac:~ user$ python
Python 2.7.10 (default, Jul 14 2015, 19:46:27)
[GCC 4.2.1 Compatible Apple LLVM 6.0 (clang-600.0.39)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import gclib
>>> g = gclib.py()
>>> g.GAddresses()
{'/dev/tty.usbserial0': ''}
>>> g.GOpen("/dev/tty.usbserial0 -d")
>>> print(g.GInfo())
/dev/tty.usbserial0, DMC4143 Rev 1.2b, 9998
>>> g.GClose()
>>> exit()
user-mac:~ user$
```

Installed files

- The gclib shared object files
  - /Applications/gclib/dylib/gclib.0.dylib
  - /Applications/gclib/dylib/gclibo.0.dylib
- The gclib header files
  - /Applications/gclib/include/gclib_errors.h
  - /Applications/gclib/include/gclibo.h
  - /Applications/gclib/include/gclib.h

  - /Applications/gclib/include/gclib_record.h

- gclib documentation tarball
  - /Applications/gclib/doc/gclib_doc.tar.gz
- Example source tarball
  - /Applications/gclib/examples/gclib_examples.tar.gz
- Source files to modify/rebuild libgclibo.so
  - /Applications/gclib/source/gclibo_229_src.tar.gz
- GalilTools Communication Library (gcl) wrapper
  - /Applications/gclib/source/gclib_gcl.tar.gz

Documentation

The documentation is left as a tarball to minimize disk usage. The latest release version of the user manual is available at the following link.

- http://www.galil.com/sw/pub/all/doc/gclib/html/

Offline pdf

The following allows viewing of the pdf docs from the installation.

- Browse in the Finder to Applications/gclib/doc.
- Double-click the tar.gz file to extact it.
- Open the resultant directory.
- Open the pdf.

Offline html

The following allows viewing of the html docs from the installation.

- Browse in the Finder to Applications/gclib/doc.
- Double-click the tar.gz file to extact it.
- Open the resultant directory.
- Open the html directory.
- Double-click index.html to open the help.

Chapter 4

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.

4.1 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)

4.1.1 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.

4.1.2 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.

4.1.3 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.

4.1.4 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.

4.1.5 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;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:
```

4.1.6 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:
```

4.1.7 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

```
$ mkdir test
$ cd test
$ tar -xzf /usr/share/doc/gclib/src/gclib_examples.tar.gz
$ 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;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:

4.1.8 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;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:
```

4.2 Python

Install gclib

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
$ tar -xvf /usr/share/doc/gclib/src/gclib_python.tar.gz
gclib.py
setup.py
$ tar -xvf /usr/share/doc/gclib/src/gclib_python_examples.tar.gz
example.py
$ 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;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)'

4.3 .Net

- VB.NET
- C#.NET

4.3.1 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

4.3.2 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

4.4 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. Java hackers with recommendations on how to make this library better are encouraged to email support@galil.com. Somebody has to teach those Galil Java noobs what's what.

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.

4.5 LabVIEW

This document provides step-by-step instructions on how to access the Galil Communication Library (gclib) in LabVIEW on Windows 10. The Galil Communication Library (gclib) is the current generation communication library for Galil Motion Controllers and PLCs. By combining the versatility of gclib with the simplicity of programming in LabVIEW, a professional looking graphical user interface can be created for a wide range of motion control applications.

Installation

To install the LabVIEW language support, run the latest version of the gclib installer and select the "LabVIEW Integration" option.

Accessing gclib with LabVIEW

In LabVIEW, create a new VI and right click on the Block Diagram. At the bottom of the Functions palette, Galil palette set will be available. Functions are sorted in to three categories: Configuration, Data, and Utility.

Examples for gclib with LabVIEW

Full Examples

Full Examples will be installed to C:\Program Files (x86)\National Instruments\LabVIEW 2020\vi.lib\Galil\examples\full_examples.

1. Analog_IO_Example.vi – This example will display and store the value of a single analog input. The most current analog input value will be displayed on the Front Panel of the VI. All analog input values will be stored in a .txt file in a location specified by the user.
2. Digital_IO_Example.vi – This example allows the user to set digital outputs using the 'OP' command and read digital inputs using the 'TI' command. It will also pol the state of Digital Output 1. The most current digital I/O information will be displayed on the Front Panel of the VI in the form of a bit mask.
3. Data_Record_Example.vi – This example uses GRecord() to capture data record packets and then parses those packets into valuable information such as sample time (TIME), reference position (RP), and actual encoder position (TP). The most current values of each will be displayed on the Front Panel of the VI. Note, this example was written for the DMC-4000.

VI Examples

Individual VI Examples will be installed to C:\Program Files (x86)\National Instruments\LabVIEW 2020\vi.lib\Galil\examples\vi_examples.

1. example_GArrayDownload().vi
   - This example uses GArrayDownload() to download array elements to a Galil controller. The VI Front Panel gives users access to which array elements are downloaded to the controller.
2. example_GArrayUpload().vi

- This example uses GArrayUpload() to upload array elements from a Galil controller. The VI Front Panel will display all array elements uploaded from the controller.

3. example_GCommand().vi
   - This example uses GCommand() to send 'MG TIME' to query the value of the internal free running, real time clock. Users can send other DMC commands by altering the Front Panel of the example VI.
4. example_GInterrupt().vi
   - This example uses GInterrupt() to pause code execution until a specified event has occurred. In this case, GInterrupt() pauses code execution until profiled motion is complete on Axis A. The user can define which event interrupts will hold up code execution using the EI/UI commands.
5. example_GMessage().vi
   - This example shows users how to access unsolicited messages from a Galil controller using the GMessage() function. Unsolicited messages are messages that are generated by a DMC program running on the controller.
6. example_GProgramDownload().vi
   - This example uses GProgramDownload() to download a DMC program to a Galil controller. The VI Front Panel gives allows users to define the DMC program before downloading it to the controller.
7. example_GProgramUpload().vi
   - This example uses GProgramUpload() to upload a DMC program from a Galil controller. The VI Front Panel will display the DMC program uploaded from the controller.
8. example_GRecord().vi
   - This example shows users how to access the data record from a Galil controller using the GRecord() function. This example gathers data record packets and then divides the packet into valuable information. For more information on the format of the data record packets from your controller, please check the controller's user manual.
9. example_GArrayDownloadFile().vi
   - This example uses GArrayDownloadFile() to download array elements from a file to a Galil controller. The users should specify the path to the file in the VI Front Panel.
10. example_GArrayUploadFile().vi
    - This example uses GArrayUploadFile() to upload array elements to a file from a Galil controller. The users should specify the path to the file in the VI Front Panel.
11. example_GMotionComplete().vi
    - This example uses GMotionComplete to block program execution until motion is complete on the A axis. Once code execution resumes, another command is sent to the controller to confirm that the sequence has been completed.
12. example_GProgramDownloadFile().vi
    - This example uses GProgramDownloadFile() to download a DMC program from a file location to a Galil controller. The VI Front Panel allows users to define the file path to the DMC program before before downloading to the controller.
13. example_GProgramUploadFile().vi
    - This example uses GProgramUploadFile() to upload a DMC program from a Galil controller. The VI Front Panel allows users to define the file path to the DMC program before uploading from the controller.

14. example_GRecordRate().vi

    - This example uses GRecordRate() to set the asynchronous data record rate (in milliseconds) for the DR method of gathering data records. Once that is accomplished, this example gathers data record packets and then divides the packet into valuable information. For more information on the format of the data record packets from your controller, please check the controller's user manual.

15. example_GTimeout().vi

    - This example uses GTimeout() to set the timeout for for GInterrupt(). Once that is accomplished, This example uses GInterrupt() to pause code execution until profiled motion is complete on Axis A. The user can define which event interrupts will hold up code execution using the EI/UI commands.

gclib/LabVIEW Support

If there questions or requests, please contact Galil Support at support@galil.com or call 916-626-0101 to speak with an Applications Engineer.

Chapter 5

Using gclib

5.1 gcaps

gcaps 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.
Incidentally, the name *gcaps* is an acronym for the improbable name *Galil Controller Asynchronous Proxy Server*. Yet another tidbit to impress friends at parties.

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.

5.2 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.

5.3 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.

5.4 Galil Widgets

Note

gclib provides the communications foundation for the Galil Widgets project. 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.

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)

5.5 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

Recompiling gclibo requires the source code for the open source compression library **zlib**. Make will automatically download the needed project from the zlib website: **https://www.zlib.net/**.

Copy files

```
$ mkdir test
$ cd test
$ tar -xzf /usr/share/doc/gclib/src/gclibo_src.tar.gz
$ 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
Acquiring zlib
wget -q http://zlib.net/zlib-1.2.11.tar.gz
shasum -a 256 -c zlib.sha256
zlib-1.2.11.tar.gz: OK
tar -xzf zlib-1.2.11.tar.gz
Compiling zlib
gcc -c -Wall -w -fPIC -fvisibility=hidden -DBUILDING_GCLIB -DHAVE_VISIBILITY zlib-1.2.11/*.c
Compiling open source component, libgclibo.so.0.0
gcc -Izlib-1.2.11 -c -Wall -Werror -fPIC -fvisibility=hidden -DBUILDING_GCLIB -DHAVE_VISIBILITY gclibo.c array
Linking open source component into shared library.
gcc -shared -o libgclibo.so.0.0 *.o -lgclib -L. -Wl,-rpath=/usr/lib -Wl,-soname=libgclibo.so.0
```

```
strip --strip-unneeded libgclibo.so.0.0
Installing libgclibo.so.0.0
install -m 755 libgclibo.so.0.0 /usr/lib
ldconfig
# make clean -f makefile_gclibo
Cleaning project...
```

Test

Extract simple example

```
$ tar -xzf /usr/share/doc/gclib/src/gclib_examples.tar.gz 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
:
```

5.6 Software Licenses

For purposes of licensing, gclib is broken into two categories.

1. The closed source portion is covered under the Closed Source License. This covers the binaries created for the gclib.h interface.
2. The open source portion and all examples and wrappers are covered under the Open Source License.

The informal software licenses are provided to describe the user's responsibilities in a format that is easy to read and understand. If more legal verbiage is required or if any questions arise regarding the following details, please contact Galil for clarification.

5.6.1 Closed Source License

Copyright (c) 2016, Galil Motion Control

Galil C Library (gclib) Closed Source Software License - January 4, 2016

This informal software license is provided to describe the user's responsibilities in a format that is easy to read and understand. If more legal verbiage is required or if any questions arise regarding the following details, please contact Galil for clarification.
This license includes the closed source portion of the gclib, defined by the header file gclib.h as outlined in the accompanying documentation (the docs).
Galil hereby grants the following:
The closed source portion may be freely copied and used for any Galil application, commercial and non-commercial in an unmodified format, unless such modifications are detailed in the docs.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE

DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Galil Motion Control
270 Technology Way
Rocklin, CA 95765, USA
support@galil.com
1 (800) 377-6329
1 (916) 626-0102
www.galil.com

5.6.2 Open Source License

Galil C Library (gclib) Open Source Software License - January 4, 2016

This informal software license is provided to describe the user's responsibilities in a format that is easy to read and understand. If more legal verbiage is required or if any questions arise regarding the following details, please contact Galil for clarification.
This license includes the open source portions of the gclib, as well as examples and wrappers as outlined in the accompanying documentation (the docs).
Galil hereby grants the following:
The open source portions, examples, and wrappers may be freely copied, used, and modified for any Galil application, commercial and non-commercial provided that:
(1) This license appears with all copies, including modifications, of the software source code.
(2) Modified files must be commented at the top of the file with the original Galil version number and the author of the modifications.
(3) This license NEED NOT appear with binaries or executables if the source code is not included.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Galil Motion Control
270 Technology Way
Rocklin, CA 95765, USA
support@galil.com
1 (800) 377-6329
1 (916) 626-0102
www.galil.com

5.7 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.

5.7.1 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

```
$ tar -xzf /usr/share/doc/gclib/src/gclib_gcl.tar.gz
$ 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
```

5.7.2 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

Chapter 6

Module Index

6.1 Modules

Here is a list of all modules:

Chapter 7

Namespace Index

7.1 Namespace List

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

Chapter 8

Data Structure Index

8.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 9

File Index

9.1 File List

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

Chapter 10

Module Documentation

10.1 API

10.1.1 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_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 GLost (GCon g)

  Checks for a lost connection.
- 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.

10.1.2 Function Documentation

10.1.2.1 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, and GUtility().

10.1.2.2 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().

10.1.2.3 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, and H_ArrayDownloadFromMemory().

10.1.2.4 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.
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(), and write_array_to_file().

10.1.2.5 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(), and H_Write↩ ArrayCsv().

10.1.2.6 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, and GUtility().

10.1.2.7 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.

10.1.2.8 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, and GCommand().
Referenced by check_interrupts(), commands(), contour(), GRecordRate(), H_DownloadArraysFromList(), H_↩ DownloadData(), jog(), load_buf(), load_buffer(), message(), motion_complete(), position_tracking(), record_↩ position(), and vector().

10.1.2.9 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, and GCommand().
Referenced by commands(), and GRecordRate().

10.1.2.10 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, and GCommand().
Referenced by commands(), record_position(), and vector().

10.1.2.11 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, and GCommand().
Referenced by commands(), GArrayUploadFile(), GRecordRate(), and motion_complete().

10.1.2.12 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(), GCmdI(), GCmdT(), GWaitForBool(), and motion_↩ complete().

10.1.2.13 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 G_NO_ERROR.
Referenced by error().

10.1.2.14 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
```

10.1.2.15 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, and GUtility().

10.1.2.16 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(), and motion_complete().

10.1.2.17 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(), and GUtility().
Referenced by ip_assigner().

10.1.2.18 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, and GUtility().

10.1.2.19 GLost()

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

Checks for a lost connection.

Attention

`GLost()` checks if a device has been lost. Devices are considered "lost" when the suddenly disapear which can happen when a controller is powered off or the ethernet cord is disconnected. GLost should be called periodically and the results handled accordingly.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function.

See x_examples.cpp for an example.

10.1.2.20 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().

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().

10.1.2.21 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, and GWaitForBool().
Referenced by contour(), jog(), position_tracking(), and vector().

10.1.2.22 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`

address switch	Meaning	Arguments (default), other options	Examples
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.

10.1.2.23 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(), and record_position().

10.1.2.24 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, and GProgramDownload().

10.1.2.25 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().

10.1.2.26 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(), and MAXPROG.

10.1.2.27 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, and GUtility().
Referenced by remote_server().

10.1.2.28 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.

10.1.2.29 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.

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.

10.1.2.30 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(), and GCmdT().

10.1.2.31 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, and GUtility().

10.1.2.32 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, and GUtility().

10.1.2.33 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, and GUtility().

10.1.2.34 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(), and H_FindSector().

10.1.2.35 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 GIpRequests(), GWaitForBool(), record_position(), and vector().

10.1.2.36 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().

10.1.2.37 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(), GPublish↩Server(), GRemoteConnections(), GServerStatus(), GSetServer(), GSleep(), GTimeout(), GVersion(), and message().

10.1.2.38 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, and GUtility().

10.1.2.39 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(), and POLLINGINTERVAL.
Referenced by GMotionComplete().

10.1.2.40 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.

10.2 C#/VB examples

10.2.1 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.

10.2.2 Function Documentation

10.2.2.1 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.
Referenced by Commands_Example.Main().

10.2.2.2 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, and Examples.Record_↩Position().
Referenced by Contour_Example.Main().

10.2.2.3 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, and Examples.GALIL_EXAMPLE_OK.
Referenced by IP_Assigner_Example.Main().

10.2.2.4 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.
Referenced by Jog_Example.Main().

10.2.2.5 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, and message().
Referenced by Message_Example.Main().

10.2.2.6 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, and Examples.GALIL_EXAMPLE_OK.
Referenced by Motion_Complete_Example.Main().

10.2.2.7 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.
Referenced by Position_Tracking_Example.Main().

10.2.2.8 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.
Referenced by Examples.Contour(), and Record_Position_Example.Main().

10.2.2.9 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 Examples.GALIL_EXAMPLE_OK.
Referenced by Remote_Client_Example.Main().

10.2.2.10 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.
Referenced by Remote_Server_Example.Main().

10.2.2.11 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.
Referenced by Vector_Mode_Example.Main().

10.3 C++ examples

10.3.1 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_server_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)

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().

10.3.2.2 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(), GMotionComplete(), and record_position().

10.3.2.3 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.
Referenced by check_interrupts(), commands(), contour(), H_ArrayDownloadFromMemory(), ip_assigner(), jog(), load_buf(), load_buffer(), message(), motion_complete(), position_tracking(), record_position(), remote_server(), vector(), and write_array_to_file().

10.3.2.4 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, GIpRequests(), and string_split().

10.3.2.5 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(), and GMotionComplete().

10.3.2.6 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(), and GCmd().
Referenced by vector().

10.3.2.7 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 G_SMALL_BUFFER, and pause().

10.3.2.8 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(), and GUtility().
Referenced by Examples::Message().

10.3.2.9 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(), and GTimeout().

10.3.2.10 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(), and GMotionComplete().

10.3.2.11 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(), and write_array_to_file().
Referenced by contour().

10.3.2.12 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 G_SMALL_BUFFER.

10.3.2.13 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, and GPublishServer().

10.3.2.14 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(), and load_buffer().

Chapter 11

Namespace Documentation

11.1 examples Namespace Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.
- 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 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.

Chapter 12

Data Structure Documentation

12.1 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.

12.1.1 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.

12.1.2 Member Function Documentation

12.1.2.1 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, and Examples.PrintError().
The documentation for this class was generated from the following file:

- commands_example.cs

12.2 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.

12.2.1 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.

12.2.2 Member Function Documentation

12.2.2.1 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, and Examples.PrintError().
The documentation for this class was generated from the following file:

- contour_example.cs

12.3 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.

12.3.1 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.

12.3.2 Member Function Documentation

12.3.2.1 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.
Referenced by Commands_Example.Main().

12.3.2.2 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, and Examples.Record_↩Position().
Referenced by Contour_Example.Main().

12.3.2.3 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, and Examples.GALIL_EXAMPLE_OK.
Referenced by IP_Assigner_Example.Main().

12.3.2.4 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.
Referenced by Jog_Example.Main().

12.3.2.5 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, and message().
Referenced by Message_Example.Main().

12.3.2.6 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, and Examples.GALIL_EXAMPLE_OK.
Referenced by Motion_Complete_Example.Main().

12.3.2.7 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.
Referenced by Position_Tracking_Example.Main().

12.3.2.8 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.
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().

12.3.2.9 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.
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.
Referenced by Examples.Contour(), and Record_Position_Example.Main().

12.3.2.10 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 Examples.GALIL_EXAMPLE_OK.
Referenced by Remote_Client_Example.Main().

12.3.2.11 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.
Referenced by Remote_Server_Example.Main().

12.3.2.12 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.
Referenced by Vector_Mode_Example.Main().
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

12.4 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.

12.4.1 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.
The documentation for this union was generated from the following file:

- gclib_record.h

12.5 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.

12.5.1 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.
The documentation for this struct was generated from the following file:

- gclib_record.h

12.6 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).

12.6.1 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.
The documentation for this struct was generated from the following file:

- gclib_record.h

12.7 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.

12.7.1 Detailed Description

Data record struct for DMC-2103 controllers.
Definition at line 586 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

12.8 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).

12.8.1 Detailed Description

Data record struct for DMC-30010 controllers.
Definition at line 818 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

12.9 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).

12.9.1 Detailed Description

Data record struct for DMC-4000 controllers, including 4000, 4200, 4103, and 500x0.
Definition at line 35 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

12.10 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.

12.10.1 Detailed Description

Data record struct for RIO-471xx and RIO-472xx PLCs. Includes encoder fields.
Definition at line 870 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

12.11 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.

12.11.1 Detailed Description

Data record struct for RIO-47162.
Definition at line 1019 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

12.12 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.

12.12.1 Detailed Description

Data record struct for RIO-47300 with 24EX I/O daughter board.
Definition at line 968 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

12.13 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.

12.13.1 Detailed Description

Data record struct for RIO-47300. Includes encoder fields.
Definition at line 918 of file gclib_record.h.
The documentation for this struct was generated from the following file:

- gclib_record.h

12.14 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).

12.14.1 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.
The documentation for this struct was generated from the following file:

- gclib_record.h

12.15 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**

12.15.1 Detailed Description

Structure to create a linked list for array data.
Definition at line 20 of file arrays.c.
The documentation for this struct was generated from the following file:

- arrays.c

12.16 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.

12.16.1 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.

12.16.2 Member Function Documentation

12.16.2.1 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, Examples.IP_Assigner(), and Examples.PrintError().
The documentation for this class was generated from the following file:

- ip_assigner_example.cs

12.17 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.

12.17.1 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.

12.17.2 Member Function Documentation

12.17.2.1 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, Examples.Jog(), and Examples.PrintError().
The documentation for this class was generated from the following file:

- jog_example.cs

12.18 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.

12.18.1 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.

12.18.2 Member Function Documentation

12.18.2.1 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, Examples.Message(), and Examples.PrintError().
The documentation for this class was generated from the following file:

- message_example.cs

12.19 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.

12.19.1 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.

12.19.2 Member Function Documentation

12.19.2.1 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, Examples.Motion_↩ Complete(), and Examples.PrintError().
The documentation for this class was generated from the following file:

- motion_complete_example.cs

12.20 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.

12.20.1 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.

12.20.2 Member Function Documentation

12.20.2.1 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, Examples.Position_↩ Tracking(), and Examples.PrintError().
The documentation for this class was generated from the following file:

- position_tracking_example.cs

12.21 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.

12.21.1 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.

12.21.2 Member Function Documentation

12.21.2.1 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, Examples.PrintError(), and Examples.Record_Position().
The documentation for this class was generated from the following file:

- record_position_example.cs

12.22 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.

12.22.1 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.

12.22.2 Member Function Documentation

12.22.2.1 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, and Examples.Remote_↩ Client().
The documentation for this class was generated from the following file:

- remote_client_example.cs

12.23 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.

12.23.1 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.

12.23.2 Member Function Documentation

12.23.2.1 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, and Examples.Remote_↩ Server().
The documentation for this class was generated from the following file:

- remote_server_example.cs

12.24 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.

12.24.1 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.

12.24.2 Member Function Documentation

12.24.2.1 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, Examples.PrintError(), and Examples.Vector_Mode().
The documentation for this class was generated from the following file:

- vector_mode_example.cs

Chapter 13

File Documentation

13.1 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.

13.1.1 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

13.1.2 Function Documentation

13.1.2.1 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, and H_ArrayDownloadFromMemory().

13.1.2.2 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(), and H_WriteArrayCsv().

13.1.2.3 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

Bit	Value	Function	Description
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(), and H_FindSector().

13.1.2.4 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(), and GCmd().
Referenced by H_ArrayDownloadFromMemory().

13.2 commands.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- GReturn commands (GCon g)

  Demonstrates various uses of GCommand() and GUtility().

13.2.1 Detailed Description

Function calls for the Command Example Project.

13.2.2 Function Documentation

13.2.2.1 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().

13.3 commands.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.3.1 Detailed Description

Function calls for the Command Example Project.

For VB.NET, see definition in file commands.vb

13.4 commands_example.cpp File Reference

```
#include "examples.h"
```

Functions

- int main (int argc, char ∗argv[])

  Main function for Commands Example.

13.4.1 Detailed Description

See commands() for implementation of logic

13.4.2 Function Documentation

13.4.2.1 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 G_SMALL_BUFFER, and pause().

13.5 commands_example.cs File Reference

Data Structures

- class Commands_Example

  Demonstrates various uses of GCommand() and basic controller queries.

13.5.1 Detailed Description

See Commands() for implementation of logic

For VB.NET, see definition in file commands_example.vb

13.6 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.

13.6.1 Detailed Description

Function calls for the Contour Example project

13.6.2 Function Documentation

13.6.2.1 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(), GMotionComplete(), and record_position().

13.7 contour.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.7.1 Detailed Description

Function calls for the Contour Example Project.

For VB.NET, see definition in file contour.vb

13.8 contour_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char ∗argv[])

  Main function for Commands Example.

13.8.1 Detailed Description

See contour() for implementation of logic

13.8.2 Function Documentation

13.8.2.1 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 G_SMALL_BUFFER, and pause().

13.9 contour_example.cs File Reference

Data Structures

- class Contour_Example

  Record user's training and plays back training through contour mode.

13.9.1 Detailed Description

See Contour() for implementation of logic

For VB.NET, see definition in file contour_example.vb

13.10 examples.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.10.1 Detailed Description

Shared methods and constants for gclib example projects.

For VB.NET, see definition in file examples.vb

13.11 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.

13.11.1 Detailed Description

Header file for Galil gclib example projects.

13.11.2 Function Documentation

13.11.2.1 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().

13.11.2.2 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(), GMotionComplete(), and record_position().

13.11.2.3 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.
Referenced by check_interrupts(), commands(), contour(), H_ArrayDownloadFromMemory(), ip_assigner(), jog(), load_buf(), load_buffer(), message(), motion_complete(), position_tracking(), record_position(), remote_server(), vector(), and write_array_to_file().

13.11.2.4 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, GIpRequests(), and string_split().

13.11.2.5 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(), and GMotionComplete().

13.11.2.6 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(), and GUtility().
Referenced by Examples::Message().

13.11.2.7 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(), and GTimeout().

13.11.2.8 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(), and GMotionComplete().

13.11.2.9 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(), and write_array_to_file().
Referenced by contour().

13.11.2.10 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 G_SMALL_BUFFER.

13.11.2.11 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, and GPublishServer().

13.11.2.12 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(), and load_buffer().

13.12 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 GLost (GCon g)

  Checks for a lost connection.

- 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.

13.12.1 Detailed Description

Defines the interface for the Galil C Library (GCLIB).

13.12.2 Function Documentation

13.12.2.1 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().

13.12.2.2 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.
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(), and write_array_to_file().

13.12.2.3 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.

13.12.2.4 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.
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(), GCmdI(), GCmdT(), GWaitForBool(), and motion_↩ complete().

13.12.2.5 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
```

13.12.2.6 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(), and motion_complete().

13.12.2.7 GLost()

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

Checks for a lost connection.

Attention

GLost() checks if a device has been lost. Devices are considered "lost" when the suddenly disapear which can happen when a controller is powered off or the ethernet cord is disconnected. GLost should be called periodically and the results handled accordingly.

Parameters

g	Connection's handle.

Returns

The success status or error code of the function.

See x_examples.cpp for an example.

13.12.2.8 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().

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().

13.12.2.9 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`

address switch	Meaning	Arguments (default), other options	Examples
-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.

13.12.2.10 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(), and record_position().

13.12.2.11 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().

13.12.2.12 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.

13.12.2.13 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.

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.

13.12.2.14 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().

13.12.2.15 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.

13.13 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_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()"

13.13.1 Detailed Description

Defines values for the Galil C Library return codes and error strings.

13.14 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**

13.14.1 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`.

13.15 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.

13.15.1 Detailed Description

Partial implementation of gclibo.h

13.15.2 Function Documentation

13.15.2.1 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, and GUtility().

13.15.2.2 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, and GUtility().

13.15.2.3 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, and GCommand().
Referenced by check_interrupts(), commands(), contour(), GRecordRate(), H_DownloadArraysFromList(), H_↩DownloadData(), jog(), load_buf(), load_buffer(), message(), motion_complete(), position_tracking(), record_↩position(), and vector().

13.15.2.4 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, and GCommand().
Referenced by commands(), and GRecordRate().

13.15.2.5 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, and GCommand().
Referenced by commands(), record_position(), and vector().

13.15.2.6 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, and GCommand().
Referenced by commands(), GArrayUploadFile(), GRecordRate(), and motion_complete().

13.15.2.7 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 G_NO_ERROR.
Referenced by error().

13.15.2.8 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.

Parameters

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, and GUtility().

13.15.2.9 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(), and GUtility().
Referenced by ip_assigner().

13.15.2.10 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, and GUtility().

13.15.2.11 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, and GWaitForBool().
Referenced by contour(), jog(), position_tracking(), and vector().

13.15.2.12 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, and GProgramDownload().

13.15.2.13 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(), and MAXPROG.

13.15.2.14 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, and GUtility().
Referenced by remote_server().

13.15.2.15 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(), and GCmdT().

13.15.2.16 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, and GUtility().

13.15.2.17 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, and GUtility().

13.15.2.18 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, and GUtility().

13.15.2.19 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 GIpRequests(), GWaitForBool(), record_position(), and vector().

13.15.2.20 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().

13.15.2.21 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, and GUtility().

13.15.2.22 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(), and POLLINGINTERVAL.
Referenced by GMotionComplete().

13.16 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.

13.16.1 Detailed Description

Open-source convenience functions for Galil C Lib. Please email softwarefeedback@galil.com with suggestions for useful/missing functions.

13.16.2 Function Documentation

13.16.2.1 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, and GUtility().

13.16.2.2 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, and H_ArrayDownloadFromMemory().

13.16.2.3 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(), and H_Write↩ ArrayCsv().

13.16.2.4 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, and GUtility().

13.16.2.5 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, and GCommand().
Referenced by check_interrupts(), commands(), contour(), GRecordRate(), H_DownloadArraysFromList(), H_↩DownloadData(), jog(), load_buf(), load_buffer(), message(), motion_complete(), position_tracking(), record_↩position(), and vector().

13.16.2.6 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, and GCommand().
Referenced by commands(), and GRecordRate().

13.16.2.7 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, and GCommand().
Referenced by commands(), record_position(), and vector().

13.16.2.8 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, and GCommand().
Referenced by commands(), GArrayUploadFile(), GRecordRate(), and motion_complete().

13.16.2.9 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 G_NO_ERROR.
Referenced by error().

13.16.2.10 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, and GUtility().

13.16.2.11 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(), and GUtility().
Referenced by ip_assigner().

13.16.2.12 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, and GUtility().

13.16.2.13 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, and GWaitForBool().
Referenced by contour(), jog(), position_tracking(), and vector().

13.16.2.14 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, and GProgramDownload().

13.16.2.15 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(), and MAXPROG.

13.16.2.16 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, and GUtility().
Referenced by remote_server().

13.16.2.17 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(), and GCmdT().

13.16.2.18 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, and GUtility().

13.16.2.19 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, and GUtility().

13.16.2.20 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, and GUtility().

13.16.2.21 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(), and H_FindSector().

13.16.2.22 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 GIpRequests(), GWaitForBool(), record_position(), and vector().

13.16.2.23 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().

13.16.2.24 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, and GUtility().

13.16.2.25 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(), and POLLINGINTERVAL.
Referenced by GMotionComplete().

13.17 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.

13.17.1 Detailed Description

Function calls for the IP Assigner Example Project.

13.17.2 Function Documentation

13.17.2.1 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, GIpRequests(), and string_split().

13.18 ip_assigner.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.18.1 Detailed Description

Function calls for the IP Assigner Example Project.

For VB.NET, see definition in file ip_assigner.vb

13.19 ip_assigner_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char ∗argv[])

  Main function for Commands Example.

13.19.1 Detailed Description

See ip_assigner() for implementation of logic

13.19.2 Function Documentation

13.19.2.1 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 G_SMALL_BUFFER, and pause().

13.20 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.

13.20.1 Detailed Description

See IP_Assigner() for implementation of logic

For VB.NET, see definition in file ip_assigner_example.vb

13.21 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.

13.21.1 Detailed Description

Function calls for the Jog Example Project.

13.21.2 Function Documentation

13.21.2.1 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(), and GMotionComplete().

13.22 jog.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.22.1 Detailed Description

Function calls for the Jog Example Project.

For VB.NET, see definition in file jog.vb

13.23 jog_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char ∗argv[])

  Main function for Commands Example.

13.23.1 Detailed Description

See jog() for implementation of logic

13.23.2 Function Documentation

13.23.2.1 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 G_SMALL_BUFFER, and pause().

13.24 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.

13.24.1 Detailed Description

See Jog() for implementation of logic

For VB.NET, see definition in file jog_example.vb

13.25 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.

13.25.1 Detailed Description

Function calls for the Message Example project

13.25.2 Function Documentation

13.25.2.1 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(), and GUtility().
Referenced by Examples::Message().

13.26 message.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.26.1 Detailed Description

Function calls for the Message Example Project.

For VB.NET, see definition in file message.vb

13.27 message_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

13.27.1 Detailed Description

See message() for implementation of logic

13.27.2 Function Documentation

13.27.2.1 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 G_SMALL_BUFFER, and pause().

13.28 message_example.cs File Reference

Data Structures

- class Message_Example

  Demonstrates how to handle and interpret messages from the controller.

13.28.1 Detailed Description

See Message() for implementation of logic

For VB.NET, see definition in file message_example.vb

13.29 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.

13.29.1 Detailed Description

Function calls for the Motion Complete Example Project.

13.29.2 Function Documentation

13.29.2.1 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(), and GTimeout().

13.30 motion_complete.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.30.1 Detailed Description

Function calls for the Motion Complete Example Project.

For VB.NET, see definition in file motion_complete.vb

13.31 motion_complete_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char *argv[])

  Main function for Commands Example.

13.31.1 Detailed Description

See motion_complete() for implementation of logic

13.31.2 Function Documentation

13.31.2.1 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 G_SMALL_BUFFER, and pause().

13.32 motion_complete_example.cs File Reference

Data Structures

- class Motion_Complete_Example

  Uses controller interrupts to detect when motion is complete.

13.32.1 Detailed Description

See Motion_Complete() for implementation of logic

For VB.NET, see definition in file motion_complete_example.vb

13.33 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.

13.33.1 Detailed Description

Function calls for the Position Tracking Example Project.

13.33.2 Function Documentation

13.33.2.1 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(), and GMotionComplete().

13.34 position_tracking.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.34.1 Detailed Description

Function calls for the Position Tracking Example Project.

For VB.NET, see definition in file position_tracking.vb

13.35 position_tracking_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char ∗argv[])

  Main function for Commands Example.

13.35.1 Detailed Description

See position_tracking() for implementation of logic

13.35.2 Function Documentation

13.35.2.1 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 G_SMALL_BUFFER, and pause().

13.36 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.

13.36.1 Detailed Description

See Position_Tracking() for implementation of logic

For VB.NET, see definition in file position_tracking_example.vb

13.37 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.

13.37.1 Detailed Description

Function calls for the Record Position Example project

13.37.2 Function Documentation

13.37.2.1 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(), and write_array_to_file().
Referenced by contour().

13.38 record_position.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.38.1 Detailed Description

Function calls for the Record Position Example Project.

For VB.NET, see definition in file record_position.vb

13.39 record_position_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char ∗argv[])

  Main function for Commands Example.

13.39.1 Detailed Description

See record_position() for implementation of logic

13.39.2 Function Documentation

13.39.2.1 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 G_SMALL_BUFFER, and pause().

13.40 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.

13.40.1 Detailed Description

See Record_Position() for implementation of logic

For VB.NET, see definition in file record_position_example.vb

13.41 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.

13.41.1 Detailed Description

Function calls for the Remote Client Example Project.

13.41.2 Function Documentation

13.41.2.1 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 G_SMALL_BUFFER.

13.42 Remote_Client.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.42.1 Detailed Description

Function calls for the Remote Client Example Project.

For VB.NET, see definition in file remote_client.vb

13.43 remote_client_example.cs File Reference

Data Structures

- class Remote_Client_Example

  Demonstrates various uses of GListServers() and GSetServer()

13.43.1 Detailed Description

See Remote_Client() for implementation of logic

For VB.NET, see definition in file remote_client_example.vb

13.44 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.

13.44.1 Detailed Description

Function calls for the Remote Server Example Project.

13.44.2 Function Documentation

13.44.2.1 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, and GPublishServer().

13.45 Remote_Server.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.45.1 Detailed Description

Function calls for the Remote Server Example Project.

For VB.NET, see definition in file remote_server.vb

13.46 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.

13.46.1 Detailed Description

See remote_server() for implementation of logic

13.46.2 Function Documentation

13.46.2.1 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 G_SMALL_BUFFER, and pause().

13.47 remote_server_example.cs File Reference

Data Structures

- class Remote_Server_Example

  Demonstrates various uses of GPublishServer()

13.47.1 Detailed Description

See Remote_Server() for implementation of logic

For VB.NET, see definition in file remote_server_example.vb

13.48 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.

13.48.1 Detailed Description

Function calls the Vector Mode Example Project.

13.48.2 Function Documentation

13.48.2.1 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(), and GCmd().
Referenced by vector().

13.48.2.2 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(), and load_buffer().

13.49 vector_example.cpp File Reference

```
#include "examples.h"
#include <iostream>
```

Functions

- int main (int argc, char ∗argv[])

  Main function for Commands Example.

13.49.1 Detailed Description

See vector() for implementation of logic

13.49.2 Function Documentation

13.49.2.1 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 G_SMALL_BUFFER, and pause().

13.50 vector_mode.cs File Reference

Data Structures

- class Examples

  Provides a class of shared constants and methods for gclib's example projects.

13.50.1 Detailed Description

Function calls for the Vector Mode Example Project.

For VB.NET, see definition in file vector_mode.vb

13.51 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.

13.51.1 Detailed Description

See Vector_Mode() for implementation of logic

For VB.NET, see definition in file vector_mode_example.vb

Index