gclib 280

C API for Galil controllers and PLCs

Galil Motion Control

Mon Oct 26 2015

Contents

Chapter 1

Getting Started

gclib is a C-compatible application programming interface (API) for communicating with Galil motion controllers and PLCs. 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 and on various platforms.

gclib will import virtually anywhere a dll/so/dylib can be imported. See Using gclib for details. Please contact softwaresupport@galil.com if the 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.

Technical Support

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

Chapter 2

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 2013* and can be extended to other Visual Studio versions.

Open *VS2013 x86 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 the source files. Note the quotes.

```
C:\temp>copy "%base%\source\gclibo\*.c" .
```

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. Note the quotes.

```
C:\temp>cl -c *.c -I "%base%\include" -DBUILDING_GCLIB
```

Link the source code. Note the quotes.

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

Copy

Copy back to the installation location from the file explorer.

- 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

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

Edit GOpen() call as necessary

Compile

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

Set Path to DLL

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

Execute

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

Linux

Copy files

```
$ mkdir test
$ cd test
$ tar -xvf /usr/share/doc/gclib/src/gclibo_164_src.tar.gz
gclibo.h
gclibo.c
arrays.c
makefile_gclibo
$ cp /usr/include/gclib*.h .
$ ls
arrays.c        gclib.h   gclibo.h        makefile_gclibo
gclib_errors.h  gclibo.c  gclib_record.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 -f makefile_gclibo
Open source component, libgclibo.so.0.0
  Compiling open source component.
gcc -c -Wall -Werror -fPIC -fvisibility=hidden -DBUILDING_GCLIB -DHAVE_VISIBILITY *.c
  Linking open source component into shared library.
gcc -shared -o libgclibo.so.0.0 *.o -Wl,-soname=libgclibo.so.0
strip --strip-unneeded libgclibo.so.0.0
  Cleaning up.
$ sudo make install -f makefile_gclibo
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_165_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_253_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_253_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
:
```

Chapter 3

Installation

gclib is available on the following operating systems.

- Microsoft Windows
  - 7 x64, x86
  - 8 x64, x86
  - 8.1 x64, x86
  - 10 x64, x86
- Apple OS X
  - Yosemite 10.10 x64
  - Mavericks 10.9 x64
- Ubuntu Linux
  - 14.04 LTS x64
  - 12.04 LTS x64
- Fedora Linux
  - fc22 x64
  - fc21 x64
- Red Hat & CentOS Linux
  - rhel7 x64
  - rhel6 x64
  - CentOs 7 x64
  - CentOs 6 x64

Don't see your OS? Please email `softwaresupport@galil.com`, or call `Galil Applications`.

3.1 Microsoft Windows

Tested versions

See the installation page for supported versions.

Installation

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 communications.

Download Installer

Recommended, all instructions and examples depend on installation paths.

Download Zip

For custom deployment or non-default file locations.

If an earlier version is required, see http://www.galil.com/sw/pub/win/gclib/ for a list of all builds.

Required third-party DLLs

gclib is built using MSVC2013 and requires run-time components available in the Visual C++ Redistributable Packages for Visual Studio 2013. On machines that don't already have Visual Studio 2013 installed, the required files can be installed from Microsoft. Be sure to install the appropriate architecture (x86 or x64).

- http://www.microsoft.com/en-us/download/details.aspx?id=40784

Uninstall gclib

- Run `uninstall.exe` in "C:\Program Files (x86)\Galil\gclib"

Installed Files

Installation from the executable installer looks like the following.

```
C:\Program Files (x86)\Galil\gclib>tree /a
Folder PATH listing for volume OS
Volume serial number is AE3F-6836
C:.
+---dll
|   +---x64
|   \---x86
+---doc
|   \---html
|       \---search
+---examples
|   +---cpp
|   +---cs
|   |   \---2013_12.0
|   |       \---gclib_example
|   |           \---gclib_example
|   |               \---Properties
```

```
|   +---gcc
|   +---mingw
|   +---msvc
|   |   \---2013_12.0
|   |       \---gclib_example
|   |           \---gclib_example
|   \---vb
|       \---2013_12.0
|           \---gclib_example
|               \---gclib_example
|                   \---My Project
+---include
+---lib
|   \---dynamic
|       +---x64
|       \---x86
\---source
    +---gclibo
    \---wrappers
        +---cs
        +---gcl
        \---vb
```

dll

The *dll* directory contains the binary *dynamic link libraries* (DLLs) for both x86 and 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. `gclib using MinGW`.

lib

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

source

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

3.2 Apple OS X

Tested versions

See the installation page for supported versions.

Installation

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

Download the gclib dmg

- Point a browser at http://www.galil.com/sw/pub/apple/osx/10.10/gclib/gclib_↩dmg.html and download the dmg file.
- Open the dmg file and drag the gclib directory to the Applications alias or another installation location.

- If an earlier version is required, see http://www.galil.com/sw/pub/apple/osx/10.↩10/gclib/ for a list of builds.

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_226_doc.tar.gz
- Example source tarball
  - /Applications/gclib/examples/gclib_226_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_229_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.

3.3 Ubuntu Linux

Tested versions

See the installation page for supported versions.

Installation

Create a temporary variable for Ubuntu version

```
uver=$(lsb_release -r | cut -f 2); echo $uver
14.04
```

Install Galil's public certificate

```
$ wget http://www.galil.com/sw/pub/ubuntu/$uver/GALIL-PUB-KEY
$ sudo apt-key add GALIL-PUB-KEY
```

Get Galil's apt sources list

```
$sudo wget http://www.galil.com/sw/pub/ubuntu/$uver/galil.list -O /etc/apt/sources.list.d/galil.list
$sudo apt-get update
```

Install Package

```
 $sudo apt-get install gclib
```

Uninstall Package

To uninstall gclib.

```
$sudo apt-get remove gclib
```

Installed files

- The gclib shared object files
  - /usr/lib/libgclibo.so
  - /usr/lib/libgclib.so
- The gclib header files

  - /usr/include/gclib_errors.h
  - /usr/include/gclibo.h
  - /usr/include/gclib.h
  - /usr/include/gclib_record.h
- gclib documentation tarball
  - /usr/share/doc/gclib/gclib_129_doc.tar.gz
- Example source tarball
  - /usr/share/doc/gclib/src/gclib_129_examples.tar.gz
- Source files to modify/rebuild libgclibo.so
  - /usr/share/doc/gclib/src/gclibo_129_src.tar.gz
- GalilTools Communication Library (gcl) wrapper
  - /usr/share/doc/gclib/src/gclib_129_gcl.tar.gz
- PCI driver files
  - /usr/share/doc/gclib/src/gclib_129_pci.tar.gz

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_202_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_132_doc.tar.gz html
$ firefox html/index.html
```

Offline pdf

There may be a pdf shipped in the package. The following allows viewing of the pdf docs from the installation.

```
$ tar -xzf /usr/share/doc/gclib/gclib_132_doc.tar.gz gclib_132.pdf
$ evince gclib_132.pdf
```

3.4 Fedora Linux

Tested versions

See the installation page for supported versions.

Installation

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

Download Galil's repository information

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

Install Package

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

```
$ sudo yum install gclib
```

Uninstall Package

To uninstall gclib.

```
$ sudo yum remove gclib
```

Installed files

- The gclib shared object files
  - /usr/lib/libgclibo.so
  - /usr/lib/libgclib.so
- The gclib header files
  - /usr/include/gclib_errors.h
  - /usr/include/gclibo.h
  - /usr/include/gclib.h
  - /usr/include/gclib_record.h
- gclib documentation tarball
  - /usr/share/doc/gclib/gclib_129_doc.tar.gz
- Example source tarball
  - /usr/share/doc/gclib/src/gclib_129_examples.tar.gz
- Source files to modify/rebuild libgclibo.so
  - /usr/share/doc/gclib/src/gclibo_129_src.tar.gz
- GalilTools Communication Library (gcl) wrapper
  - /usr/share/doc/gclib/src/gclib_129_gcl.tar.gz
- PCI driver files
  - /usr/share/doc/gclib/src/gclib_129_pci.tar.gz

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

Extract source and build driver

```
$ tar -xf /usr/share/doc/gclib/src/gclib_202_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 pdf

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

```
$ tar -xzf /usr/share/doc/gclib/gclib_132_doc.tar.gz gclib_132.pdf
$ evince gclib_132.pdf
```

Offline html

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

```
$ tar -xzf /usr/share/doc/gclib/gclib_132_doc.tar.gz html
$ firefox html/index.html
```

3.5 Red Hat & CentOS Linux

Tested versions

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.

Attention

- **Red Hat 7 and CentOS 7**
  - Point a browser at http://www.galil.com/sw/pub/rhel/7/galilrpm-2-1.noarch.rpm and install the rpm.
- **Red Hat 6 and CentOS 6**
  - 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.

```
$ sudo yum install gclib
```

Uninstall Package

To uninstall gclib.

```
$ sudo yum remove gclib
```

Installed files

- The gclib shared object files
  - /usr/lib/libgclibo.so
  - /usr/lib/libgclib.so
- The gclib header files
  - /usr/include/gclib_errors.h
  - /usr/include/gclibo.h
  - /usr/include/gclib.h
  - /usr/include/gclib_record.h
- gclib documentation tarball
  - /usr/share/doc/gclib/gclib_129_doc.tar.gz
- Example source tarball
  - /usr/share/doc/gclib/src/gclib_129_examples.tar.gz
- Source files to modify/rebuild libgclibo.so
  - /usr/share/doc/gclib/src/gclibo_129_src.tar.gz
- GalilTools Communication Library (gcl) wrapper
  - /usr/share/doc/gclib/src/gclib_129_gcl.tar.gz
- PCI driver files
  - /usr/share/doc/gclib/src/gclib_129_pci.tar.gz

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

Extract source and build driver

```
$ tar -xf /usr/share/doc/gclib/src/gclib_202_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_132_doc.tar.gz html
$ firefox html/index.html
```

Offline pdf

There may be a pdf shipped in the package. The following allows viewing of the pdf docs from the installation.

```
$ tar -xzf /usr/share/doc/gclib/gclib_132_doc.tar.gz  gclib_132.pdf
$ evince gclib_132.pdf
```

Chapter 4

Legacy Compatibility

4.1 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_164_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
```

4.2 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, or 8.1, DMC32 OSU may be used on these O.S. upgrades.

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/in
  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/galil
  _dmc32_osu_exe.html

Chapter 5

Using gclib

A number of examples are provided to demonstrate how to use the library with various languages and on various platforms.

- C/C++
- Python
- .Net
- Galil Widgets

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

5.1 C/C++

- Microsoft Visual Studio
- MinGW
- Borland C++
- gcc (Linux)
- clang (OS X)

5.1.1 Microsoft Visual Studio

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.

5.1.2 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:
```

5.1.3 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:
```

5.1.4 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_131_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;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1
i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;i=i+1;EN

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

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

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


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

QR-based data record
36100
6000

DR-based data record
36204
```

```
36306
36408
36510
36612
36714
36816
36918
37020
37122
37224

QR-based data record with offsets
37224
37224

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

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

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

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

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


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

5.1.5 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_229_examples.tar.gz
$ cp /Applications/gclib/include/* .
$ cp /Applications/gclib/dylib/* .
$ ls
gclib.0.dylib   x_arrays.cpp    x_gmotioncomplete.cpp
gclib.h     x_examples.cpp    x_gread_gwrite.cpp
gclib_errors.h    x_examples.h    x_grecord.cpp
gclib_record.h    x_gcommand.cpp    x_nonblocking.cpp
gclibo.0.dylib    x_ginterrupt.cpp  x_programs.cpp
gclibo.h    x_gmessage.cpp    x_simple.c
```

x_simple.c

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

Compile

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

Run

```
$ ./simple
rc: 0
version: 126.108.229
rc: 0
rc: 0
info: 10.1.3.142, DMC4020 Rev 1.2a-BH, 291
rc: 0
response:  206676.0000
:
```

x_examples.cpp

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

Compile

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

Run

```
$ ./example
Library version: 126.108.229

10.1.3.142, DMC4020 Rev 1.2a-BH, 291

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

Read 1 byte(s)
```

```
:
Program test OK.

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

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

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

Receiving Binary Data
QR read 155 bytes

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

Modifying timeout
Burning program...OK

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

5.2 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.

```
C:\Users\username>cd Desktop
C:\Users\username\Desktop>mkdir python_temp
C:\Users\username\Desktop>cd python_temp
C:\Users\username\Desktop\python_temp>copy "c:\Program Files (x86)\Galil\gclib\source\wrappers\python\*" .
C:\Users\username\Desktop\python_temp>copy "c:\Program Files (x86)\Galil\gclib\examples\python\*" .
C:\Users\username\Desktop\python_temp>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 -> C:\Python34\Lib\site-packages
byte-compiling C:\Python34\Lib\site-packages\gclib.py to gclib.cpython-34.pyc
running install_egg_info
Writing C:\Python34\Lib\site-packages\gclib-1.0-py3.4.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 ~/python_temp
$ cd ~/python_temp/
$ tar -xvf /usr/share/doc/gclib/src/gclib_256_python.tar.gz
gclib.py
setup.py
$ tar -xvf /usr/share/doc/gclib/src/gclib_256_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_253_python.tar.gz
x gclib.py
x setup.py
$ tar -xvf /Applications/gclib/examples/gclib_253_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 --direct')
>>> 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.
    See Link GOpen() <http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h_aef4aec8a85630eed029b7a46aea7db5

>>> 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.
    See Link GCommand() <http://www.galil.com/sw/pub/all/doc/gclib/html/gclib_8h_a5ac031e76efc965affdd73a1bec0

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

5.3 .Net

- VB.NET
- C#.NET

5.3.1 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 -d") '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
```

- Hit *F5* to run the project

5.3.2 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 -d"); //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!
            }

        }
    }
}
```

- Hit *F5* to run the project

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)

Chapter 6

Data Structure Index

6.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 7

File Index

7.1 File List

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

Chapter 8

Data Structure Documentation

8.1 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 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.
- unsigned char byte_array [GALILDATARECORDMAXLENGTH]
  Generic byte array for offsets.

8.1.1 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 875 of file gclib_record.h.

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

- gclib_record.h

8.2 GDataRecord1802 Struct Reference

Data record struct for DMC-1802 controllers. Same as 2103 except no analog in axis data.

`#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_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_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_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_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_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_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_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.

8.2.1 Detailed Description

Data record struct for DMC-1802 controllers. Same as 2103 except no analog in axis data.

Definition at line 536 of file gclib_record.h.

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

- gclib_record.h

8.3 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).

8.3.1 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 224 of file gclib_record.h.

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

- gclib_record.h

8.4 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.

8.4.1 Detailed Description

Data record struct for DMC-2103 controllers.

Definition at line 401 of file gclib_record.h.

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

- gclib_record.h

8.5 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).

8.5.1 Detailed Description

Data record struct for DMC-30010 controllers.

Definition at line 663 of file gclib_record.h.

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

- gclib_record.h

8.6 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).

8.6.1 Detailed Description

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

Definition at line 34 of file gclib_record.h.

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

- gclib_record.h

8.7 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.

8.7.1 Detailed Description

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

Definition at line 715 of file gclib_record.h.

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

- gclib_record.h

8.8 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.

8.8.1 Detailed Description

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

Definition at line 813 of file gclib_record.h.

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

- gclib_record.h

8.9 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.

8.9.1 Detailed Description

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

Definition at line 763 of file gclib_record.h.

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

- gclib_record.h

8.10 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**

8.10.1 Detailed Description

Structure to create a linked list for array data.

Definition at line 10 of file arrays.c.

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

- arrays.c

Chapter 9

File Documentation

9.1 arrays.c File Reference

```
#include "gclibo.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 downsteam 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)

  Uplaods 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)

  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 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.

9.1.1 Detailed Description

Function calls for uploading and downloading arrays with CSV files.

Definition in file arrays.c.

9.1.2 Function Documentation

9.1.2.1 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 251 of file arrays.c.

References G_BAD_FILE, G_NO_ERROR, H_ArrayAddElement(), H_CreateArrayNode(), H_DownloadArrays↩FromList(), H_FreeArrays(), and H_InitArrayNode().

9.1.2.2 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 326 of file arrays.c.

References G_NO_ERROR, GCmdT(), H_FreeArrays(), H_InitArrayNode(), H_UploadArrayToList(), and H_Write↩ArrayCsv().

9.1.2.3 GReturn H_DownloadArraysFromList (GCon *g*, ArrayNode * *head*)

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 126 of file arrays.c.

References G_BOUNDS, G_NO_ERROR, GArrayDownload(), and GCmd().

Referenced by GArrayDownloadFile().

9.2 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_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.

- #define G_SMALL_BUFFER 1024

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

  Most reads/writes to Galil hardware are small. This value will hold the largest array or program upload/download possible.

Typedefs

- typedef int GReturn

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

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

  Size of buffers, etc.
- typedef int GOption

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

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

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

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

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

  Interrupt status byte.
- typedef void * GMemory

  Pointer to untyped memory for use in GUtility().

Functions

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

  Open a connection to a Galil Controller.
- GCLIB_DLL_EXPORTED GReturn GCALL GClose (GCon g)

  Closes a connection to a Galil Controller.
- GCLIB_DLL_EXPORTED GReturn GCALL GRead (GCon g, GBufOut buffer, GSize buffer_len, GSize *bytes_read)

  Performs a read on the connection.
- GCLIB_DLL_EXPORTED GReturn GCALL GWrite (GCon g, GBufIn buffer, GSize buffer_len)

  Performs a write on the connection.
- GCLIB_DLL_EXPORTED GReturn GCALL GCommand (GCon g, GCStringIn command, GBufOut buffer, GSize buffer_len, GSize *bytes_returned)

  Performs a command-and-response transaction on the connection.
- GCLIB_DLL_EXPORTED GReturn GCALL GProgramDownload (GCon g, GCStringIn program, GCStringIn preprocessor)

  Downloads a program to the controller's program buffer.
- GCLIB_DLL_EXPORTED GReturn GCALL GProgramUpload (GCon g, GBufOut buffer, GSize buffer_len)

  Uploads a program from the controller's program buffer.
- GCLIB_DLL_EXPORTED GReturn GCALL GArrayDownload (GCon g, const GCStringIn array_name, G↩ Option 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.

9.2.1 Detailed Description

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

Definition in file gclib.h.

9.2.2 Function Documentation

9.2.2.1 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().

9.2.2.2 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().

9.2.2.3 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.

9.2.2.4 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 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 GCmd(), GCmdD(), GCmdI(), GCmdT(), and GMotionComplete().

9.2.2.5 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
```

9.2.2.6 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.

9.2.2.7 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_E↩ MPTY will be returned.

See x_gmessage.cpp for an example. See x_nonblocking.cpp for an example of non-blocking usage.

9.2.2.8 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. REQUIRED for this version of gclib.**		`-a GALILPCI2 --direct`
`-d`	shorthand for `--direct`		`GALILPCI2 -d`
`--handshake`	**Serial Handshake mode**	(`HARDWARE`), `NONE`	`COM1 --handshake NONE`
`--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`
`--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`), `TCP`, `NONE`	`192.168.0.42 --unsolicited TCP`

-u	shorthand for --unsolicited		192.168.1.42 -u NONE

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.

9.2.2.9 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. – Maximum compression, only if needed, to fit the program. – Code downloads at start of buffer. • Compression options – `--max` *n* provides preprocessing up to and including level *n*. Only the necessary preprocessing will be performed up to level *n*, as listed below. * Level 0 (mandatory) 1. Remove lines starting with REM. 2. Error on \ in buffer. 3. Comment blank lines with '. 4. Remove white space (space/tab) in front of # (label declarations). 5. Remove white space after commands. 6. Remove trailing semicolons. 7. Line ends changed to carriage return. 8. Replace leading tabs with double space. 9. Replace non-leading tabs with single space. * 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) too. * 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. – `--min` *n* will preprocess at least up to and including *n*. *n* defined as with `--max` above. • Code insertion – `--insert` *arg* invokes the insert option of the firmware's *DL* command. *arg* can be one of the following. * Line number, e.g. `100`. Program insertion will occur on the line after the line specified. * Variable name, e.g. `myvar`. Program insertion will occur on the line after the line equal to the value of the variable. * Label callout, e.g. `#mylabel`. Program insertion will occur on the line after the label. * A lone # symbol. Program insertion will occur on the line after the last line in the program buffer. – **Important 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. – Compression directives `--max` and `--min` are followed. – All original code following the point of insertion is cleared.
gclib 280	– Not all products support the `--insert` operation, e.g. DMC-30010. See the DL command for support.

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().

9.2.2.10 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().

9.2.2.11 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.

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.

9.2.2.12 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_NO↩ N_BLOCKING_READ_EMPTY will be returned.

See x_grecord.cpp for an example. See x_nonblocking.cpp for an example of non-blocking usage.

9.2.2.13 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 a pointer to 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 pointer to a short which overrides 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 pointer to 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, e.g. g may be null. – `memory1` is output, and must be a `GCStringOut`. Data will be null terminated, even if the data must be truncated to do so. – `memory2` is input and must be a pointer to a `GSize` 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 `GCStringOut`. Data will be null terminated, even if the data must be truncated to do so. – `memory2` is input and must be a pointer to a `GSize` holding the length of the buffer in `memory1`. • `G_UTIL_SLEEP` See GSleep(). Platform-independent sleep. A valid connection (g) is not necessary, i.e. g may be null. – `memory1` is input and must be a pointer to an unsigned int, units are milliseconds. – `memory2` is ignored, use null. • `G_UTIL_ADDRESSES` 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. – `memory1` is output and must be a `GCStringOut`. Data will be null terminated, even if the data must be truncated to do so. – `memory2` is input and must be a pointer to a `GSize` holding the length of the buffer in `memory1`. • `G_UTIL_IPREQUEST` 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 `GCStringOut`. Data will be null terminated, even if the data must be truncated to do so. – `memory2` is input and must be a pointer to a `GSize` holding the length of the buffer in `memory1`. • `G_UTIL_ASSIGN` Provides a method to assign an IP address given a Galil MAC address and optionally a host NIC IP address. A valid connection (g) is not necessary, i.e. g may be null. The address to assign will be pinged to ensure its availability before the assign packet is sent. `G_GCLIB_UTILITY_IP_TAKEN` will be returned by GUtility() if the ping returns a response.

memory2	An untyped pointer to data type required for request.

Returns

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

See source of gclibo.c for examples of G_UTIL_TIMEOUT, G_UTIL_TIMEOUT_OVERRIDE, G_UTIL_VER↩SION, G_UTIL_INFO, G_UTIL_SLEEP.

Except for serial ports, each line from G_UTIL_ADDRESSES will be of the form *address, revision report (^R^V)*.

```
1 10.1.3.168, DMC30010 Rev 1.2d
2 GALILPCI1, DMC1826 Rev 1.1a
3 COM7
4 COM8
```

See GAddresses() for an example of G_UTIL_ADDRESSES.

Note

Linux/OS X users must be root to use G_UTIL_IPREQUEST and have UDP access to bind and listen on port 67.

Each line from G_UTIL_IPREQUEST will be of the form *model, serial_number, mac*.

```
1 DMC4000, 291, 00:50:4c:20:01:23
2 DMC30000, 4184, 00:50:4c:40:10:58
```

```
//example for getting controllers requesting IPs
char buf[1024];
GSize len = sizeof(buf);
GUtility(0, G_UTIL_IPREQUEST, buf, &len);
cout << buf << "\n";
```

Note

Linux users must be root to use G_UTIL_ASSIGN and have UDP access to send on port 68.

```
//example of assigning an IP address.
GUtility(0, G_UTIL_ASSIGN, "10.1.3.178", "00:50:4c:40:10:58"); //Assign 10.1.3.178 TO
       00:50:4c:40:10:58
```

Referenced by GAddresses(), GAssign(), GInfo(), GIpRequests(), GSleep(), GTimeout(), and GVersion().

9.2.2.14 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.

See x_gread_gwrite.cpp for an example.

9.3 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
- #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_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_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_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"

9.3.1 Detailed Description

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

Definition in file gclib_errors.h.

9.4 gclib_record.h File Reference

Data Structures

- struct GDataRecord4000

  Data record struct for DMC-4000 controllers, including 4000, 4200, 4103, and 500x0.
- struct GDataRecord1806

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

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

  Data record struct for DMC-1802 controllers. Same as 2103 except no analog in axis data.
- 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.
- 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 unsigned char **UB**
- typedef unsigned short **UW**
- typedef short **SW**
- typedef int **SL**
- typedef unsigned int **UL**

9.4.1 Detailed Description

Defines a union for data records. Each supported controller has a struct member in the union with named record types. Offsets into the data record can also be used by referencing the member `byte_array`.

Definition in file gclib_record.h.

9.5 gclibo.c File Reference

```
#include "gclibo.h"
```

Functions

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

  Uses GUtility() and `G_UTIL_VERSION` to provide the library version number.
- GReturn GCALL GInfo (GCon g, GCStringOut info, GSize info_len)

  Uses GUtility() and `G_UTIL_INFO` to provide a useful connection string.
- GReturn GCALL GAddresses (GCStringOut addresses, GSize addresses_len)

  Uses GUtility() and `G_UTIL_ADDRESSES` to provide a listing of all available connection addresses.
- GReturn GCALL GTimeout (GCon g, short timeout_ms)

  Uses GUtility() and `G_UTIL_TIMEOUT_OVERRIDE` to set the library timeout.
- GReturn GCALL GAssign (char ∗ip, char ∗mac)

  Assigns IP address over the Ethernet to a controller at a given MAC address.
- GReturn GCALL GIpRequests (GCStringOut requests, GSize requests_len)

  Provides a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.
- GReturn GCALL GCmd (GCon g, GCStringIn command)

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

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

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

  Wrapper around GCommand that provides the return value of a command parsed into a double.
- GReturn GCALL GMotionComplete (GCon g, GCStringIn axes)

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

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

  Program download from file.

- GReturn GCALL GProgramUploadFile (GCon g, GCStringIn file_path)
  Program upload to file.
- void GCALL GError (GReturn rc, GCStringOut error, GSize error_len)
  Provides a human-readable description string for return codes.

9.5.1 Detailed Description

Partial implementation of gclibo.h

Definition in file gclibo.c.

9.5.2 Function Documentation

9.5.2.1 GReturn GCALL GAddresses (GCStringOut *addresses*, GSize *addresses_len*)

Uses GUtility() and `G_UTIL_ADDRESSES` to provide a listing of all available connection addresses.

Note

Serial ports are listed, e.g. COM1. It may be necesary 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.
addresses_len	Length of buffer.

Returns

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

Definition at line 24 of file gclibo.c.

References G_UTIL_ADDRESSES, and GUtility().

9.5.2.2 GReturn GCALL GAssign (char * *ip*, char * *mac*)

Assigns 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. The desired IP address will be pinged prior to the assigment. If the ping is returned, GAssign() will return G_GCLIB_UTILI↩ TY_IP_TAKEN.

Note

Linux/OS X users must be root to use GAssign() and have UDP access to send on port 68.

```
//example of assigning an IP address.
GAssign("10.1.3.178", "00:50:4c:40:10:58"); //Assign 10.1.3.178 to 00:50:4c:40:10:58
```

Definition at line 34 of file gclibo.c.

References G_UTIL_ASSIGN, and GUtility().

9.5.2.3 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 44 of file gclibo.c.

References G_SMALL_BUFFER, and GCommand().

Referenced by GRecordRate(), and H_DownloadArraysFromList().

9.5.2.4 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 97 of file gclibo.c.

References G_NO_ERROR, G_SMALL_BUFFER, and GCommand().

Referenced by GRecordRate().

9.5.2.5 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 86 of file gclibo.c.

References G_NO_ERROR, G_SMALL_BUFFER, and GCommand().

9.5.2.6 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 51 of file gclibo.c.

References G_NO_ERROR, and GCommand().

Referenced by GArrayUploadFile(), and GRecordRate().

9.5.2.7 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 251 of file gclibo.c.

References G_ARRAY_NOT_DIMENSIONED, G_BAD_ADDRESS, G_BAD_FILE, G_BAD_FULL_MEMORY, G_BAD_LOST_DATA, G_BAD_RESPONSE_QUESTION_MARK, G_BAD_VALUE_RANGE, G_COMMAND_CALLED_WITH_ILLEGAL_COMMAND, G_DATA_RECORD_ERROR, G_FIRMWARE_LOAD_NOT_SUPPORTED, G_GCLIB_ERROR, G_GCLIB_NON_BLOCKING_READ_EMPTY, G_GCLIB_UTILITY_ERROR, G_ILLEGAL_DATA_IN_PROGRAM, G_INVALID_PREPROCESSOR_OPTIONS, G_NO_ERROR, G_OPEN_ERROR, G_TIMEOUT, G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT, and G_UNSUPPORTED_FUNCTION.

9.5.2.8 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.

See x_examples.cpp for an example.

Definition at line 19 of file gclibo.c.

References G_UTIL_INFO, and GUtility().

9.5.2.9 GReturn GCALL GIpRequests (GCStringOut *requests*, GSize *requests_len*)

Provides 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.
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 about 5 seconds while listening for requests.

Note

Linux/OS X users must be root to use GIpRequests() and have UDP access to bind and listen on port 67.

```
//example of listening for controllers needing IP addresses
GIpRequests(listen_buf, sizeof(listen_buf));
cout << listen_buf << '\n';
```

Each line of the returned data will be of the form *model, serial_number, mac*.

```
1 DMC4000, 291, 00:50:4c:20:01:23
2 DMC30000, 4184, 00:50:4c:40:10:58
```

Definition at line 39 of file gclibo.c.

References G_UTIL_IPREQUEST, and GUtility().

9.5.2.10 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 108 of file gclibo.c.

References G_NO_ERROR, G_SMALL_BUFFER, GCommand(), and GSleep().

9.5.2.11 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 179 of file gclibo.c.

References G_BAD_FILE, G_BAD_FULL_MEMORY, G_NO_ERROR, and GProgramDownload().

9.5.2.12 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 222 of file gclibo.c.

References G_BAD_FILE, G_BAD_FULL_MEMORY, G_NO_ERROR, and GProgramUpload().

9.5.2.13 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 134 of file gclibo.c.

References G_NO_ERROR, G_SMALL_BUFFER, GCmd(), GCmdD(), and GCmdT().

9.5.2.14 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 GMotionComplete() for an example.

Definition at line 9 of file gclibo.c.

References G_UTIL_SLEEP, and GUtility().

Referenced by GMotionComplete().

9.5.2.15 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 29 of file gclibo.c.

References G_UTIL_TIMEOUT_OVERRIDE, and GUtility().

9.5.2.16 GReturn GCALL GVersion (GCStringOut *ver*, GSize *ver_len*)

Uses GUtility() and `G_UTIL_VERSION` to provide the library version number.

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.

See x_examples.cpp for an example.

Definition at line 14 of file gclibo.c.

References G_UTIL_VERSION, and GUtility().

9.6 gclibo.h File Reference

```
#include "gclib.h"
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include <math.h>
```

Macros

- #define **GCLIB_DLL_EXPORTED**
- #define **_CRT_SECURE_NO_WARNINGS**
- #define **GCALL** __stdcall
- #define **MALLOCBUF** G_HUGE_BUFFER
- #define **MAXPROG** MALLOCBUF
- #define **MAXARRAY** MALLOCBUF
- #define **POLLINGINTERVAL** 100

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() and `G_UTIL_VERSION` to provide the library version number.
- GCLIB_DLL_EXPORTED GReturn GCALL GAddresses (GCStringOut addresses, GSize addresses_len)

  Uses GUtility() and `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 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)

  Provides a list of all Galil controllers requesting IP addresses via BOOT-P or DHCP.
- GCLIB_DLL_EXPORTED GReturn GCALL GAssign (char *ip, char *mac)

  Assigns 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.

9.6.1 Detailed Description

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

Definition in file gclibo.h.

9.6.2 Function Documentation

9.6.2.1 GCLIB_DLL_EXPORTED GReturn GCALL GAddresses (GCStringOut *addresses*, GSize *addresses_len*)

Uses GUtility() and G_UTIL_ADDRESSES to provide a listing of all available connection addresses.

Note

Serial ports are listed, e.g. COM1. It may be necesary 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.
addresses_len	Length of buffer.

Returns

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

Definition at line 24 of file gclibo.c.

References G_UTIL_ADDRESSES, and GUtility().

9.6.2.2 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 251 of file arrays.c.

References G_BAD_FILE, G_NO_ERROR, H_ArrayAddElement(), H_CreateArrayNode(), H_DownloadArrays↩FromList(), H_FreeArrays(), and H_InitArrayNode().

9.6.2.3 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 326 of file arrays.c.

References G_NO_ERROR, GCmdT(), H_FreeArrays(), H_InitArrayNode(), H_UploadArrayToList(), and H_Write↩ArrayCsv().

9.6.2.4 GCLIB_DLL_EXPORTED GReturn GCALL GAssign (char * *ip*, char * *mac*)

Assigns 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. The desired IP address will be pinged prior to the assigment. If the ping is returned, GAssign() will return G_GCLIB_UTILI↩TY_IP_TAKEN.

Note

Linux/OS X users must be root to use GAssign() and have UDP access to send on port 68.

```
//example of assigning an IP address.
GAssign("10.1.3.178", "00:50:4c:40:10:58"); //Assign 10.1.3.178 to 00:50:4c:40:10:58
```

Definition at line 34 of file gclibo.c.

References G_UTIL_ASSIGN, and GUtility().

9.6.2.5 GCLIB_DLL_EXPORTED 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 44 of file gclibo.c.

References G_SMALL_BUFFER, and GCommand().

Referenced by GRecordRate(), and H_DownloadArraysFromList().

9.6.2.6 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.

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 97 of file gclibo.c.

References G_NO_ERROR, G_SMALL_BUFFER, and GCommand().

Referenced by GRecordRate().

9.6.2.7 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.

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 86 of file gclibo.c.

References G_NO_ERROR, G_SMALL_BUFFER, and GCommand().

9.6.2.8 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.

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 51 of file gclibo.c.

References G_NO_ERROR, and GCommand().

Referenced by GArrayUploadFile(), and GRecordRate().

9.6.2.9 GCLIB_DLL_EXPORTED 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 251 of file gclibo.c.

References G_ARRAY_NOT_DIMENSIONED, G_BAD_ADDRESS, G_BAD_FILE, G_BAD_FULL_MEMORY, G_↩ BAD_LOST_DATA, G_BAD_RESPONSE_QUESTION_MARK, G_BAD_VALUE_RANGE, G_COMMAND_CALL↩ ED_WITH_ILLEGAL_COMMAND, G_DATA_RECORD_ERROR, G_FIRMWARE_LOAD_NOT_SUPPORTED, G↩ _GCLIB_ERROR, G_GCLIB_NON_BLOCKING_READ_EMPTY, G_GCLIB_UTILITY_ERROR, G_ILLEGAL_DA↩ TA_IN_PROGRAM, G_INVALID_PREPROCESSOR_OPTIONS, G_NO_ERROR, G_OPEN_ERROR, G_TIMEO↩ UT, G_UNABLE_TO_COMPRESS_PROGRAM_TO_FIT, and G_UNSUPPORTED_FUNCTION.

9.6.2.10 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.

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.

See x_examples.cpp for an example.

Definition at line 19 of file gclibo.c.

References G_UTIL_INFO, and GUtility().

9.6.2.11 GCLIB_DLL_EXPORTED GReturn GCALL GIpRequests (GCStringOut *requests*, GSize *requests_len*)

Provides 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.
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 about 5 seconds while listening for requests.

Note

Linux/OS X users must be root to use GIpRequests() and have UDP access to bind and listen on port 67.

```
//example of listening for controllers needing IP addresses
GIpRequests(listen_buf, sizeof(listen_buf));
cout << listen_buf << '\n';
```

Each line of the returned data will be of the form *model, serial_number, mac*.

```
1 DMC4000, 291, 00:50:4c:20:01:23
2 DMC30000, 4184, 00:50:4c:40:10:58
```

Definition at line 39 of file gclibo.c.

References G_UTIL_IPREQUEST, and GUtility().

9.6.2.12 GCLIB_DLL_EXPORTED 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 108 of file gclibo.c.

References G_NO_ERROR, G_SMALL_BUFFER, GCommand(), and GSleep().

9.6.2.13 GCLIB_DLL_EXPORTED 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 179 of file gclibo.c.

References G_BAD_FILE, G_BAD_FULL_MEMORY, G_NO_ERROR, and GProgramDownload().

9.6.2.14 GCLIB_DLL_EXPORTED 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 222 of file gclibo.c.

References G_BAD_FILE, G_BAD_FULL_MEMORY, G_NO_ERROR, and GProgramUpload().

9.6.2.15 GCLIB_DLL_EXPORTED 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 134 of file gclibo.c.

References G_NO_ERROR, G_SMALL_BUFFER, GCmd(), GCmdD(), and GCmdT().

9.6.2.16 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.

Parameters

timeout_ms	The timeout, in milliseconds, to block before returning.

See GMotionComplete() for an example.

Definition at line 9 of file gclibo.c.

References G_UTIL_SLEEP, and GUtility().

Referenced by GMotionComplete().

9.6.2.17 GCLIB_DLL_EXPORTED 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 29 of file gclibo.c.

References G_UTIL_TIMEOUT_OVERRIDE, and GUtility().

9.6.2.18 GCLIB_DLL_EXPORTED GReturn GCALL GVersion (GCStringOut *ver*, GSize *ver_len*)

Uses GUtility() and `G_UTIL_VERSION` to provide the library version number.

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.

See x_examples.cpp for an example.

Definition at line 14 of file gclibo.c.

References G_UTIL_VERSION, and GUtility().