Embedded IDE Link™ 4
User's Guide

For Use with Eclipse™

MATLAB®
& SIMULINK®

The MathWorks™
Accelerating the pace of engineering and science

How to Contact The MathWorks

🖥	`www.mathworks.com`	Web
	`comp.soft-sys.matlab`	Newsgroup
	`www.mathworks.com/contact_TS.html`	Technical Support
@	`suggest@mathworks.com`	Product enhancement suggestions
	`bugs@mathworks.com`	Bug reports
	`doc@mathworks.com`	Documentation error reports
	`service@mathworks.com`	Order status, license renewals, passcodes
	`info@mathworks.com`	Sales, pricing, and general information

☎ 508-647-7000 (Phone)

📠 508-647-7001 (Fax)

✉ The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

Embedded IDE Link™ User's Guide

Trademarks

Patents

Revision History

March 2010	Online only	New for Version 4.1 (Release 2010a)

Contents

1 Getting Started

2 Troubleshooting

Index

Getting Started

Note To use Embedded IDE Link™ software with Eclipse™ IDE, it is important to complete the steps in "Installing Third-Party Software" on page 1-4 and "Configuring Embedded IDE Link to Work With Eclipse" on page 1-12

Tested Software Versions

The MathWorks™ has tested Embedded IDE Link with the specific software versions listed in the following tables.

Required for all platforms	Tested Versions
Sun™ Java™ Runtime Environment (JRE)	JRE 6.0 (Java version 1.6.x)
Eclipse IDE for C/C++ Developers package, which includes the CDT feature	Ganymede (Eclipse 3.4)
CDT (If installed separately from Eclipse IDE for C/C++ Developers package, match CDT version with Eclipse version.)	CDT 5.0

Additional Software Required for Linux	Tested Versions
GNU GCC (compiler)	GCC 3.4.5
GNU as (assembler — part of the GNU binutils package)	as 2.18
GNU ar (archiver — part of the GNU binutils package)	ar 2.18
GNU GDB (debugger)	GDB 6.3.2
GNU make	make 3.81

Additional Software Required for Windows	Tested Versions
MinGW	5.1.x
GDB	GDB 6.3.2
MSYS	1.0.11

You can try untested versions and combinations of third-party software at your own risk.

For the most current information about using Embedded IDE Link software with Eclipse IDE, see:
www.mathworks.com/products/ide-link/eclipse-adaptor.html

Installing Third-Party Software

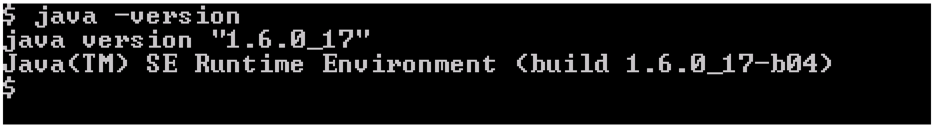
Installing Sun Java Runtime Environment (JRE)

To install the JRE, complete the following steps:

1 At your Windows® or Linux® command prompt, enter:

```
java -version
```

If Java is present, the command line responds with the version information. For example:



```
$ java -version
java version "1.6.0_17"
Java(TM) SE Runtime Environment (build 1.6.0_17-b04)
$
```

2 If Java is missing or the version is lower than 1.6.x, download and install JRE 6.0 from http://www.java.com.

3 Verify that Java is working by entering `java -version` again or by visiting http://www.java.com/en/download/help/testvm.xml.

Installing Eclipse IDE for C/C++ Developers

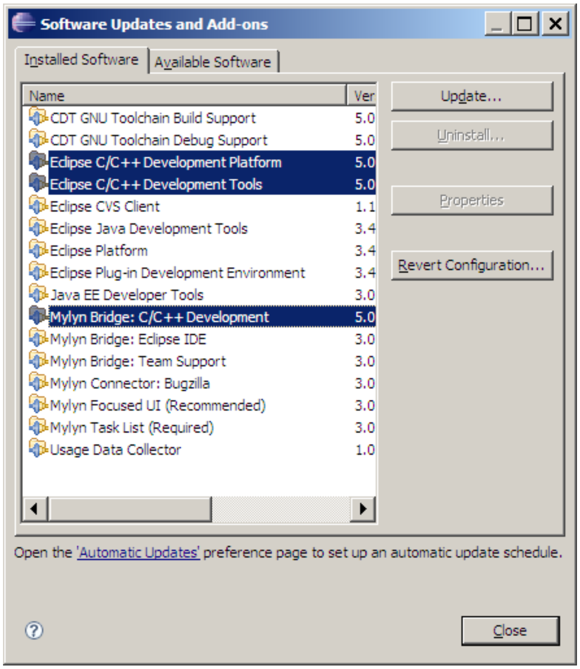
Note The following instructions are based on Eclipse 3.4 (Ganymede). More recent versions of the Eclipse IDE can appear different, and have different menus or software package names.

The Eclipse IDE for C/C++ Developers package includes the Eclipse IDE and the C/C++ Development Tools (CDT). To install Eclipse IDE for C/C++ Developers package, complete the following steps:

1 Download and install the Ganymede SR2 package, Eclipse IDE for C/C++ Developers, from http://www.eclipse.org/downloads/packages/release/ganymede/sr2.

2 When the installation process is complete, start Eclipse, and select **Help** > **Software Updates**.

3 Look under the **Installed Software** tab and verify that Eclipse has the following three CDT software packages:



Name	Ver
CDT GNU Toolchain Build Support	5.0
CDT GNU Toolchain Debug Support	5.0
Eclipse C/C++ Development Platform	5.0
Eclipse C/C++ Development Tools	5.0
Eclipse CVS Client	1.1
Eclipse Java Development Tools	3.4
Eclipse Platform	3.4
Eclipse Plug-in Development Environment	3.4
Java EE Developer Tools	3.0
Mylyn Bridge: C/C++ Development	5.0
Mylyn Bridge: Eclipse IDE	3.0
Mylyn Bridge: Team Support	3.0
Mylyn Connector: Bugzilla	3.0
Mylyn Focused UI (Recommended)	3.0
Mylyn Task List (Required)	3.0
Usage Data Collector	1.0

Open the 'Automatic Updates' preference page to set up an automatic update schedule.

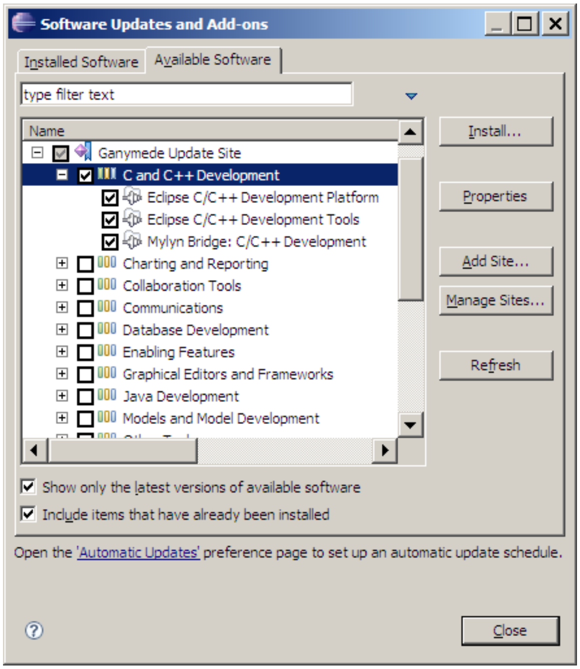
If you have a previous Eclipse installation that does not include CDT, complete the following steps:

1 In Eclipse, select **Help** > **Software Updates**.

2 Click the **Available Software** tab.

3 Click **Ganymede Update Site**.

4 Select **C and C++ Development** and click **Install**.



☑ Ganymede Update Site

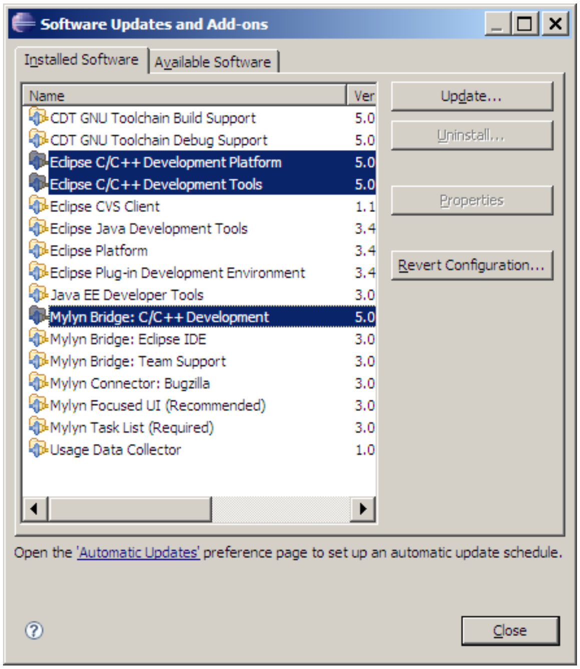
- ☑ C and C++ Development
  - ☑ Eclipse C/C++ Development Platform
  - ☑ Eclipse C/C++ Development Tools
  - ☑ Mylyn Bridge: C/C++ Development
- □ Charting and Reporting
- □ Collaboration Tools
- □ Communications
- □ Database Development
- □ Enabling Features
- □ Graphical Editors and Frameworks
- □ Java Development
- □ Models and Model Development

☑ Show only the latest versions of available software

☑ Include items that have already been installed

Open the 'Automatic Updates' preference page to set up an automatic update schedule.

5 When the installation process is complete, click the **Installed Software** tab and verify that you have CDT.



Name	Ver
CDT GNU Toolchain Build Support	5.0
CDT GNU Toolchain Debug Support	5.0
Eclipse C/C++ Development Platform	5.0
Eclipse C/C++ Development Tools	5.0
Eclipse CVS Client	1.1
Eclipse Java Development Tools	3.4
Eclipse Platform	3.4
Eclipse Plug-in Development Environment	3.4
Java EE Developer Tools	3.0
Mylyn Bridge: C/C++ Development	5.0
Mylyn Bridge: Eclipse IDE	3.0
Mylyn Bridge: Team Support	3.0
Mylyn Connector: Bugzilla	3.0
Mylyn Focused UI (Recommended)	3.0
Mylyn Task List (Required)	3.0
Usage Data Collector	1.0

Open the 'Automatic Updates' preference page to set up an automatic update schedule.

Verifying the GNU Tool Chain on Linux

Most Linux distributions include the following GNU C/C++ development tools, which Eclipse and CDT require to compile code, build projects, and debug applications:

- Assembler (as)
- Archiver (ar)
- compiler and linker (gcc)
- debugger (gdb)
- build utility (make)

Verify that the GNU tools are present and set the tool chain path:

1 On the Linux command line, enter:

   - gcc --version

- gdb --version
- as --version
- ar --version
- make --version

2 Compare the version of each tool with the following list of tested versions:

- gcc 3.4.5
- as 2.18
- ar 2.18
- gdb 6.3.2
- make 3.81

Embedded IDE Link also supports MontaVista™ Linux 4.0.1. If you are using MontaVista Linux 4.0.1, disregard the preceding list of GNU tool versions.

You can use untested versions of the GNU tools at your own risk.

To install a missing tool or to change the version of the tool, use the software installation manager that comes with your Linux distribution.

Alternatively, visit http://directory.fsf.org/GNU/ for more information about individual tools. Source files for the tools are available from:

- binutils (includes as and ar), http://ftp.gnu.org/gnu/binutils/
- gcc, http://ftp.gnu.org/gnu/gcc/
- gdb, http://ftp.gnu.org/gnu/gdb/
- make, http://ftp.gnu.org/gnu/make/

3 Set the path.

If you are using a Bash shell prompt, enter:

```
PATH=my_tool_path:$PATH
```

Where `my_tool_path` is the path to the GNU tool binaries. For example:

```
PATH=/bin:$PATH
```

If you are using a C shell prompt, enter:

```
setenv PATH my_tool_path:$PATH
```

Where my_tool_path is the path to the GNU tool binaries. For example:

```
setenv PATH /bin:$PATH
```

Installing the GNU Tool Chain on Windows

Windows typically does not include the GNU C/C++ development tools Eclipse and CDT require to compile code, build projects, and debug applications.

Provide a GNU tool chain for Windows by installing MinGW:

1 Open http://sourceforge.net/projects/mingw/files/.

2 Under “Automated MinGW Installer", download and run MinGW-5.1.x.exe.

   For **Select type of install**, choose **Full**. Otherwise, accept all of the default settings.

   Wait for the installation process to download and install additional files from the Internet.

   This software provides the assembler, archiver, linker, and compiler tools.

3 Under “GNU Source-Level Debugger”, download and run gdb-6.3.x.exe. Accept the default settings. This software provides the debugger tool.

4 Under “MSYS Base System”, download and run MSYS-1.0.11.exe. Accept the default settings. This software provides the make tool.

5 When the installer finishes, it launches a post installer configuration script. Answer y to the questions, and use forward slashes when you give the path for MinGW. For example:

```
Where is your MinGW installation? c:/mingw
```

Note Do not use paths that contains spaces.

Set the PATH environment variable:

1. In Windows, right-click the **My Computer** icon, and choose **Properties**.
2. In Properties, select the **Advanced** tab, and click **Environment Variables**.
3. Under **System variables**, scroll down to the **Path** variable.
4. Select **Path**, and click **Edit**.
5. Add the paths of the MinGW and MSYS bin folders to the beginning of the **Variable value**. Doing this ensures that the operating system calls these tools when multiple paths are present. Use semicolons to separate the paths. For example, `C:\msys\1.0\bin;C:\mingw\bin;`
6. Reboot your system.
7. After Windows reboots, verify that you installed the tools correctly. Open the Windows command line (click **Start** > **Run**, and enter `cmd`) and enter the following commands:

   - gcc --version
   - gdb --version
   - as --version
   - ar --version
   - make --version

   The command line responds to each command with version information for each tool. If not, double check that you completed each step in this installation and configuration procedure.
8. Compare the version of each tool with the following list of tested versions:

   - gcc 3.4.5
   - as 2.19

- ar 2.19
- gdb 6.3
- make 3.81

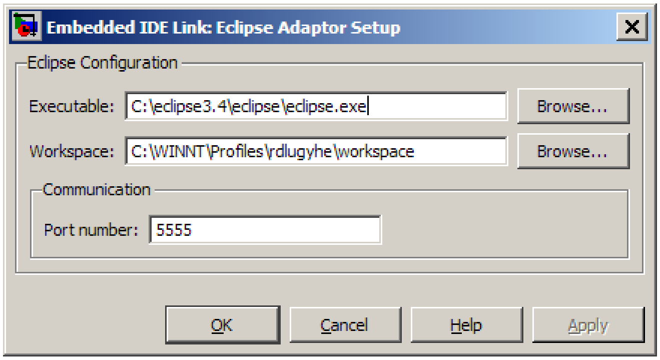
You can use untested versions of the GNU tools at your own risk.

Configuring Embedded IDE Link to Work With Eclipse

Having installed the third-party software, configure Embedded IDE Link to work with Eclipse:

1. To avoid build errors later on, close Eclipse IDE before you run `eclipseidesetup`. For more information, see "Build Errors" on page 2-4.

2. At the MATLAB® command line, enter `eclipseidesetup`. Embedded IDE Link displays the following dialog box:



Eclipse Configuration

Executable: C:\eclipse3.4\eclipse\eclipse.exe

Workspace: C:\WINNT\Profiles\rdlugyhe\workspace

Communication

Port number: 5555

3. Update **Executable** with the location and file name of the Eclipse application file. For example, `C:\Program Files\eclipse\eclipse.exe`.

4. Update **Workspace** with the default location where Eclipse creates and saves new project files. For example: `C:\WINNT\Profiles\username\workspace`.

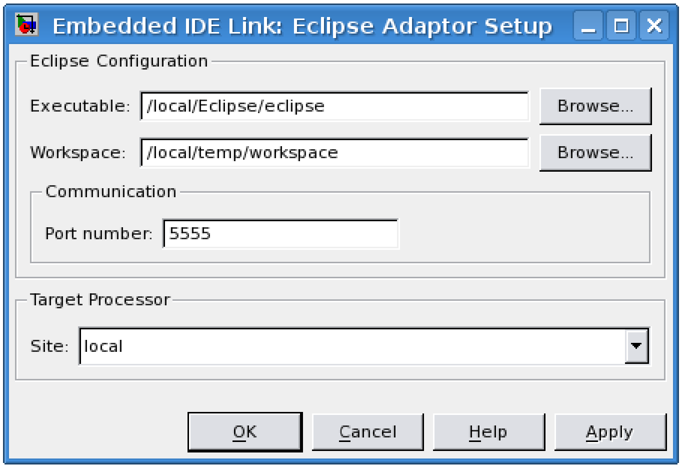
Note To find the current Eclipse workspace, select **File** > **Switch Workspace** > **Other**.

5. For **Port number**, enter a valid IP port number. For example, 5555.

6. (Linux Platform Only) Use **Site** to identify where the Embedded IDE Link uploads and runs the executable file upon completing the build process.

   - Choose `local` to run the executable on your Linux workstation.

- Choose `remote` to download the executable to a remote target processor running Linux operating system over a network connection. For example, to an embedded system connected to the Ethernet port on your workstation. If you select `remote`, see the following topic for more information: "Additional Configuration Steps to Run Your Executable on a Remote Linux Target" on page 1-14


Embedded IDE Link: Eclipse Adaptor Setup

Eclipse Configuration

Executable: /local/Eclipse/eclipse

Workspace: /local/temp/workspace

Communication

Port number: 5555

Target Processor

Site: local

7 When you click **OK** or **Apply**, Embedded IDE Link:

- Verifies the locations of the Executable and Workspace in the Eclipse Adaptor Setup dialog box.
- Verifies that the required third-party software is present.
- Installs Embedded IDE Link plug-ins in the Eclipse `plugins` folder, `C:\Program Files\eclipse\plugins\`.
- Saves configuration information to the `mwidelink.ini` file, located in the Eclipse `plugins` folder.

Note When Eclipse starts, it loads the Embedded IDE Link plug-in. The Embedded IDE Link plug-in loads the port number from `mwidelink.ini`. To resolve a port number conflict, change the port number by running `eclipseidesetup` again. Do not edit `mwidelink.ini`.

8 To verify that the configuration process is complete, create a handle object for the Eclipse IDE. Enter the following command in MATLAB

```
IDE_Obj = eclipseide
```

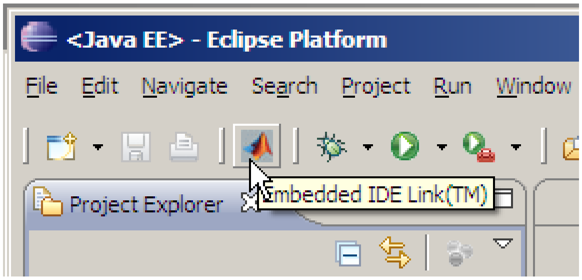
This command, starts Eclipse IDE if it is not already running, and creates a handle object:

```
Starting Eclipse(TM) IDE...

ECLIPSEIDE Object:
  Default timeout   : 10.00 secs
  Eclipse folder    : C:\eclipse3.4\eclipse
  Eclipse workspace: C:\WINNT\Profiles\rdlugyhe\workspace
  Port number       : 5555
  Processor site    : local
```

.

9 In Eclipse, find the following Embedded IDE Link plug-in icon:

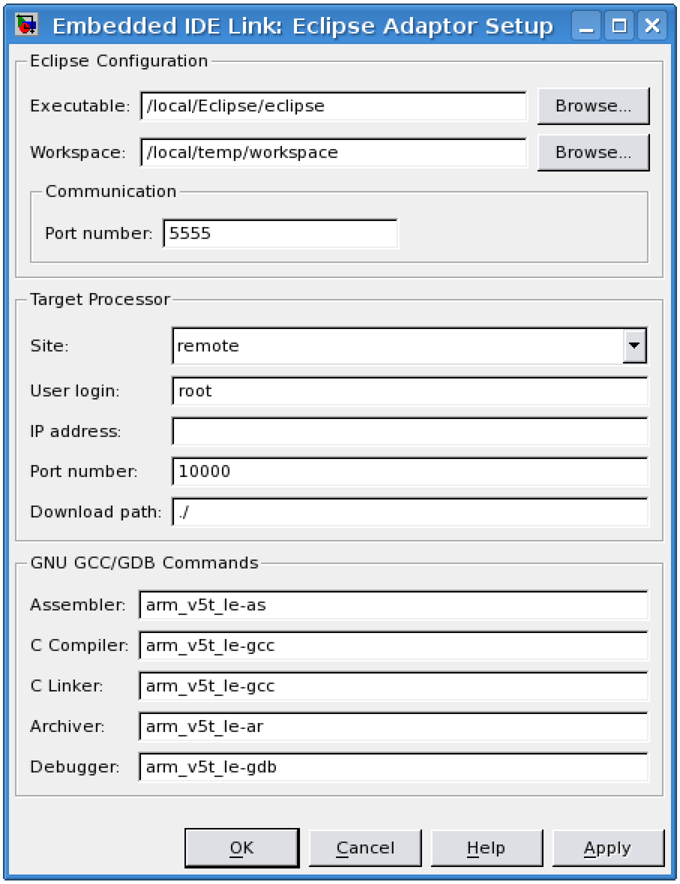


Additional Configuration Steps to Run Your Executable on a Remote Linux Target

On Linux development systems, you can configure Embedded IDE Link to download and run an executable on a remote target processor running Linux.

During the "Configuring Embedded IDE Link to Work With Eclipse" on page 1-12 process, complete these additional steps:

1 Set **Site** to `remote`. The dialog box displays additional **Target Processor** and **GNU GCC/GDB Commands** parameters.



Embedded IDE Link: Eclipse Adaptor Setup

Eclipse Configuration

Executable: /local/Eclipse/eclipse

Workspace: /local/temp/workspace

Communication

Port number: 5555

Target Processor

Site: remote

User login: root

IP address:

Port number: 10000

Download path: ./

GNU GCC/GDB Commands

Assembler: arm_v5t_le-as

C Compiler: arm_v5t_le-gcc

C Linker: arm_v5t_le-gcc

Archiver: arm_v5t_le-ar

Debugger: arm_v5t_le-gdb

2 Under **Target Processor**, enter the values Embedded IDE Link uses to connect to the target processor over the network:

- **User login**: Supply a user name that has trusted "r-" access to the remote system. The user name must be present in the /etc/hosts.equiv or $HOME/.rhosts files on the remote system.

- **IP address**: Enter the IP address of the remote system. To test the software on your local system instead of the remote system, enter `localhost`.
- **Port number**: Enter the IP port number for communications between the two systems. For example, `10000`.
- **Download path**: Enter the path on the remote system that receives the compiled executable and related files. For example, `./` sends the files to the home directory of the user login.

3 Under **GNU GCC/GDB Commands**, enter the toolchain commands and optional arguments Embedded IDE Link uses to build executable for the target processor.

For example, if you are using the generic GNU toolchain to build an executable for a target processor running Linux, enter:

- **Assembler**: `as`
- **C Compiler**: `gcc`
- **C Linker**: `gcc`
- **Archiver**: `ar`
- **Debugger**: `gdb`

For example, if you are using the MontaVista Linux toolchain to build an executable for an ARM® processor running Embedded Linux, enter:

- **Assembler**: `arm_v5t_le-as`
- **C Compiler**: `arm_v5t_le-gcc`
- **C Linker**: `arm_v5t_le-gcc`
- **Archiver**: `arm_v5t_le-ar`
- **Debugger**: `arm_v5t_le-gdb`

4 Click **OK** to complete the Eclipse Adaptor Setup process.

Also see: Preparing Models to Run on Windows or Linux

Troubleshooting

SIGSEGV Segmentation Fault for GDB

If you use Comodo Internet Security (CIS) software on your development system, CIS causes a SIGSEGV segmentation fault for GDB. For example:

```
Debugger name and version: GNU gdb (GDB) 7.0
Program received signal SIGSEGV, Segmentation fault.
In ntdll!RtlpWaitForCriticalSection () (C:\WINDOWS\system32\ntdll.dll)
Continuing...
Program received signal SIGSEGV, Segmentation fault.
In ?? () (C:\WINDOWS\system32\guard32.dll)
```

If you get this message, click **OK** and **Continue** when the message appears.

For more information, see http://forums.comodo.com/defense_bugs/guard32dll_causes_sigsegv_segmentation_fault_fo

GDB Stops on Each Semaphore Post

If you use gdb to debug a target application for Linux or Windows, gdb stops on each semaphore post. You can override this expected behavior adding the following text to `.gdbinit`, the GDB init file:

```
handle SIG34 nostop noprint pass
handle SIG35 nostop noprint pass
```

On Linux, the default location for `.gdbinit` is your home directory. On Windows, the environment variable `HOME` defines the home directory. For more information creating `.gdbinit` and configuring gdb to find it, consult the *GDB User Manual*, available from http://www.gnu.org/software/gdb/documentation/.

Build Errors

If you use eclipseidesetup without closing Eclipse IDE, you may get build errors similar to the following ones:

```
The call to idelink_ert_make_rtw_hook, during the exit hook generated the
Error while creating the project.

The build process will terminate as a result.
===
Error while creating the project.
===
Error creating a new project.
===
An exception occurred while performing this operation. 0
```

To solve this problem, close and restart Eclipse IDE.

Index