

# Software IDE Quick Start Guide

# Table of Contents

1. Introduction .....	3
1.1. Software tools .....	3
1.2. References .....	3
2. Projects .....	4
2.1. Creating a new project .....	4
2.2. Building a project .....	7
3. Debugging an application .....	8
3.1. Preparing the remote target .....	8
3.2. Configuring a debug target .....	8
3.3. Restarting an application .....	9
4. FAQ / TROUBLESHOOTING .....	10
5. Support .....	12

## 1. Introduction

This document describes how to setup a software IDE together with the Cobham Gaisler tools to be able to compile applications and/or enable C/C++ level debugging of LEON applications.

### 1.1. Software tools

#### Eclipse IDE

Eclipse IDE is a popular development platform with support for many different programming languages and third-party plug-ins.

The guide is based on Eclipse IDE for C/C++ Developers 2019-09, but will most likely work with other versions as well as long as all the necessary plugins are available.

The following Eclipse plugins are used in this guide, which are included in the Eclipse IDE for C/C++ Developers package.

- Eclipse C/C++ Development Tools (org.eclipse.cdt)
- CDT GCC Cross Compiler Support (org.eclipse.cdt.build.crossgcc)
- Eclipse GDB Hardware Debugging Plug-in (org.eclipse.cdt.debug.gdbjtag)

#### GCC Toolchain for the LEON architecture

The GCC toolchain for the LEON architecture is required to be able to compile and link applications. Please see documentation of the toolchain for installation instructions.

The following toolchains are supported.

- Bare C cross-compiler (BCC 1.0.x, BCC 2.0.x)
- RTEMS LEON/ERC32 Cross-Compiler System (RCC 1.2.x, RCC 1.3.x)

#### GDB Server for the remote target

To be able to upload an application to a target system and run it, a GDB server compatible software must be available. Please see the user manual of the software for installation instructions.

The following programs are supported.

- GRMON 3 – a debug monitor for the LEON processor
- GRMON 2 – a debug monitor for the LEON processor
- TSIM 3 – a SPARC architecture simulator
- TSIM 2 – a SPARC architecture simulator

### 1.2. References

Eclipse website

[<https://www.eclipse.org/>]

GRMON download website

[<https://www.gaisler.com/index.php/downloads/debug-tools>]

TSIM download website

[<https://www.gaisler.com/index.php/downloads/simulators>]

BCC Download website

[<https://www.gaisler.com/index.php/downloads/compilers>]

RCC 1.2 (RTEMS 4.10) Download site

[<https://www.gaisler.com/index.php/downloads/compilers>]

RCC 1.3 (RTEMS 5) Download site

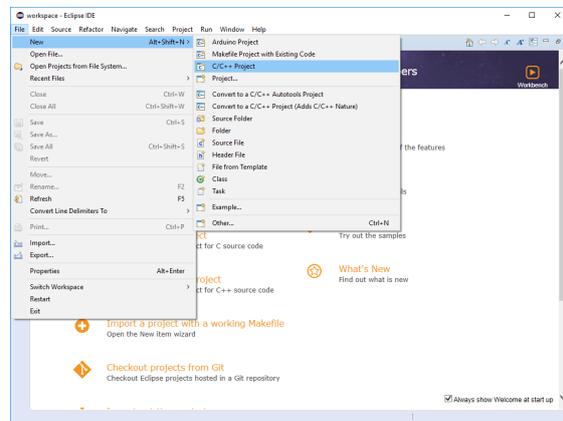
[<https://www.gaisler.com/index.php/downloads/compilers>]

## 2. Projects

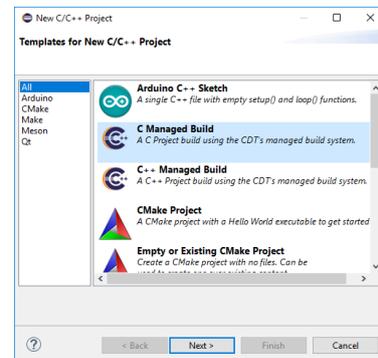
### 2.1. Creating a new project

This section describes the steps needed to create new project with a custom GCC based toolchain using the CDT GCC Cross Compiler plugin.

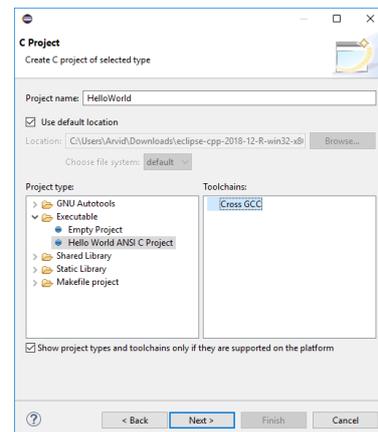
1. In the menu, select **File -> New -> C/C++ Project**



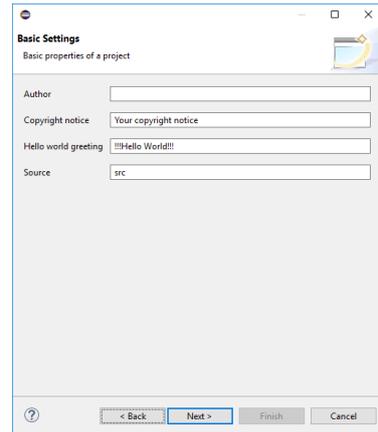
2. Select the **C Managed Build** template.  
Note that there are other valid templates at this stage, but they are not covered by this guide.
3. Press **Next** to continue.



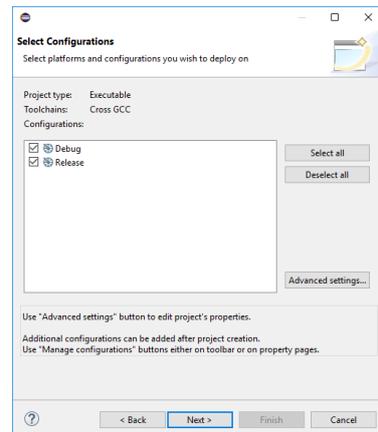
4. Enter your the name of your project at the **Project name** field.
5. Under **Project type**, select **Hello World ANSI C Project**
6. Under Toolchains, select **Cross GCC**.
7. Press **Next** to continue.



8. Customize any inputs as you please.
9. Press **Next** to continue.



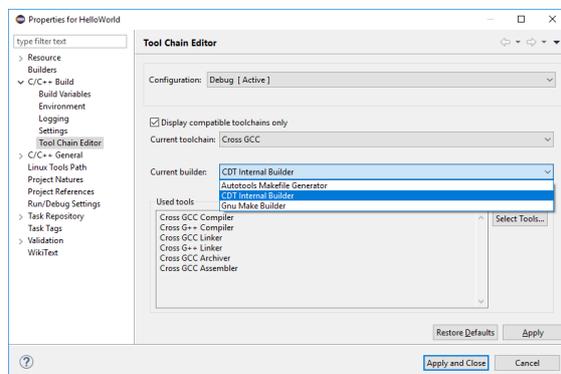
10. Press **Advanced settings...**



The following steps are optional. They are recommended for Windows users and other systems where the program make is not available.

Jump to step 16 to skip these steps.

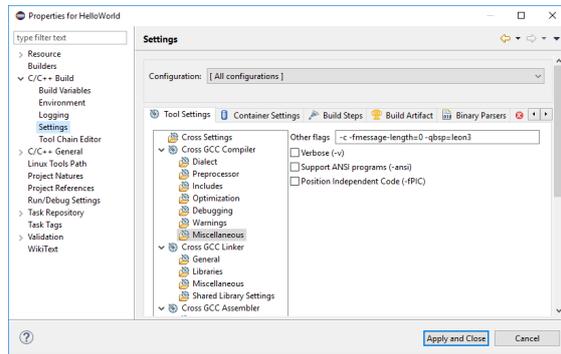
11. In the settings tree, select **C/C++ Build -> Tool Chain Editor**
12. Set the **Configuration** to **Debug**.
13. At the **Current Builder** option, select **CDT Internal Builder**.
14. Set the **Configuration** to **Release**.
15. At the **Current Builder** option, select **CDT Internal Builder**.



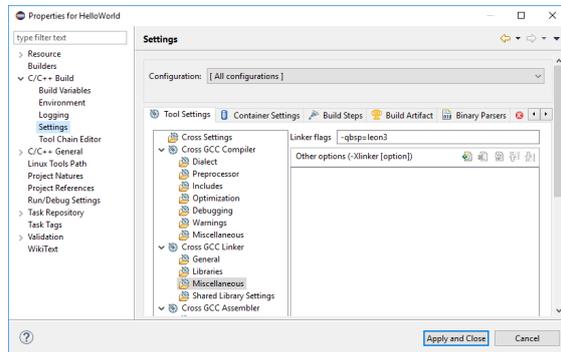
The following steps are optional. They are needed to setup the compiler and linker flags for the target hardware that are recommended by the toolchain documentation, i.e. `-qbsp`, `-mfix-*`, `-mcpu` etc. See toolchain documentation for more information.

Jump to step 21 to skip these steps.

16. In the settings tree, select **C/C++ Build -> Settings**
17. Set the **Configuration** to **All configurations**.
18. To add compiler flags, select **Cross GCC Compiler -> Miscellaneous** and the flags in the field **Other flags**.

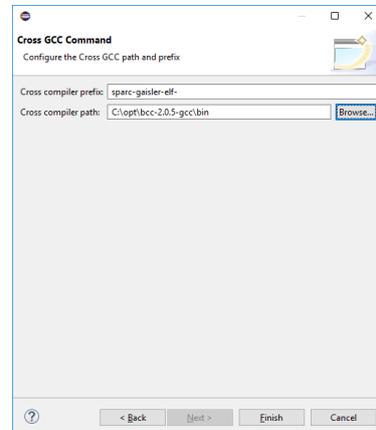


19. To add linker specific flags, select **Cross GCC Linker -> Miscellaneous** and the flags in the field **Linker flags**.



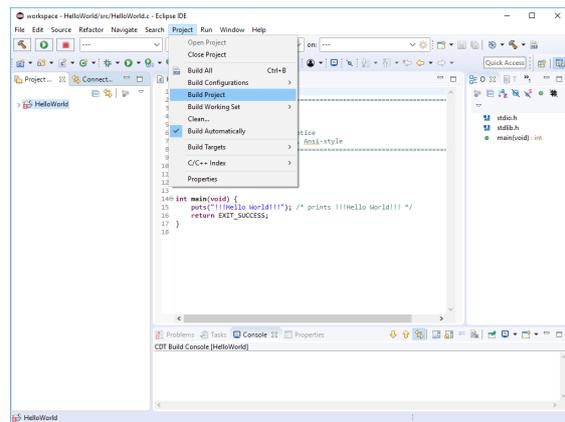
20. Press **Apply and Close** to close the advanced settings dialog.
21. Press **Next** to continue.

22. At the field **Cross compiler prefix**, enter the prefix of the toolchain, including the trailing hyphen. The prefix is the string that is in front of all the executable names in the `bin` folder of the toolchain.
23. Enter the **Cross compiler path** by pressing the **Browse**. Locate and choose the `bin` folder of the toolchain.
24. Press **Finish** to continue.



## 2.2. Building a project

To build the project, select menu **Project -> Build Project**.



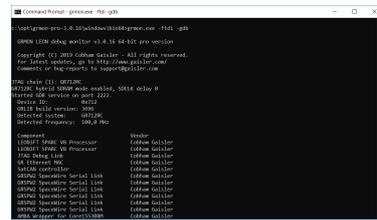
### 3. Debugging an application

#### 3.1. Preparing the remote target

Before the debugging session in Eclipse is started, you must start TSM/GRMON and enable it to act as a GDB server. This is achieved by adding `-gdb [port]` to the command line, in addition to your normal command line options.

If you want to forward the UART input/output from the system you need to start GRMON3 with the commandline option `-ucli` \*. GRMON will setup it's terminal to handle stdin/stdout, therefore the GRMON prompt will not be shown, nor will any commands be accepted.

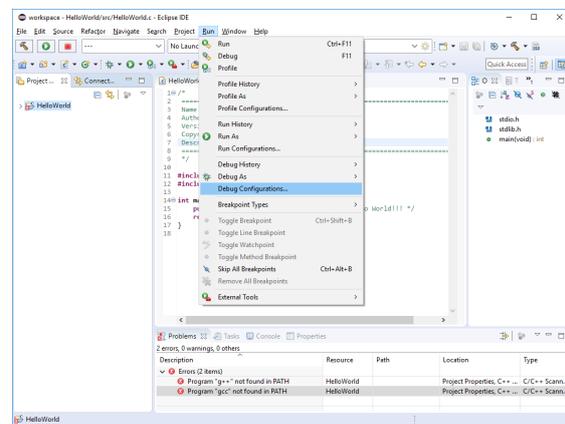
\*The `-ucli` was added in GRMON 3.1.2. See FAQ Q4 for more information.



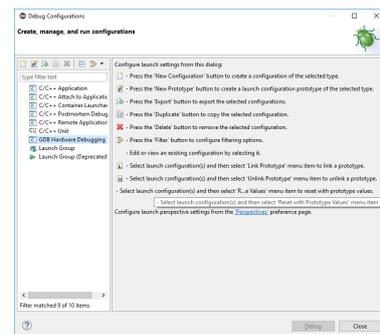
#### 3.2. Configuring a debug target

This section describes how to setup a new debug configuration for an application, which includes connecting and uploading the application to the target board, using the Eclipse GDB Hardware Debugging Plug-in.

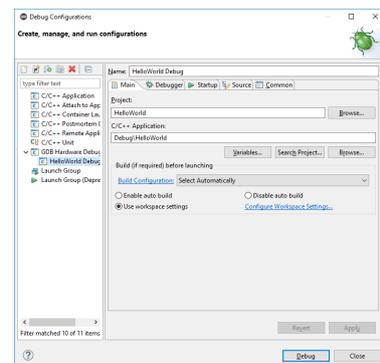
1. In the menu, select **Run -> Debug Configurations...**



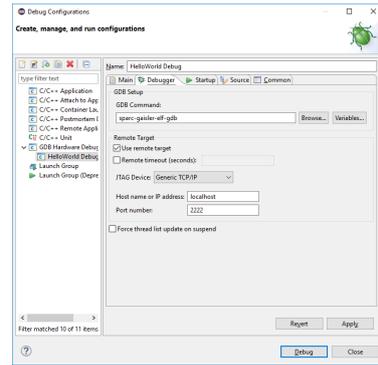
2. Double-click on **GDB Hardware Debugging** to create a new configuration



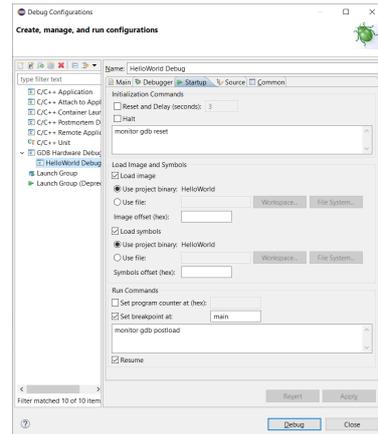
3. Verify that the **Project** name is correct.
4. Verify that the **C/C++ Application** path is set to the application that you want to debug.



5. Open the **Debugger** tab.
6. Setup the path to the GDB executable at **GDB command**. If the environment variable **PATH** is setup for the toolchain, or if the project was created using the Cross GCC plug-in, then it should be sufficient to enter the name of the GDB executable. Otherwise press **Browse** and locate the GDB executable in the toolchain **bin** folder.
7. The drop-down list **JTAG Device** should be set to **Generic TCP/IP**.
8. Enter the **Port number** that GRMON/TSIM has opened.



9. Open the **Startup** tab.
10. Add "**monitor gdb reset**" to the first large text box <sup>\*</sup>
11. Verify that **Load Image** is checked.
12. Verify that **Load Symbols** is checked.
13. Check **Set breakpoint at** and enter **main** in the field.
14. Add "**monitor gdb postload**" to the second large text box <sup>†</sup>
15. Check **Resume**
16. Press **Debug** to start the application.

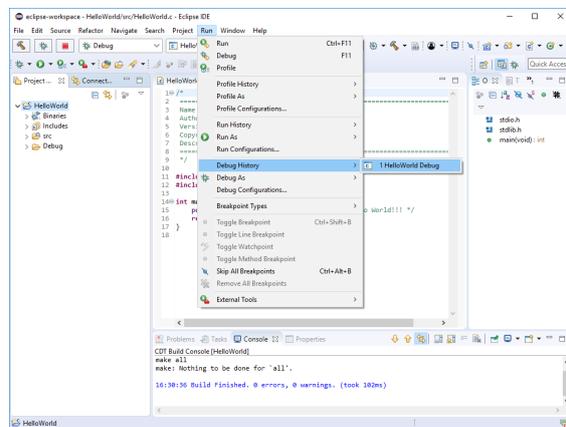


<sup>\*</sup>This command was added in GRMON 3.1.2, TSIM 2.0.65. For older versions. See FAQ Q6 for more information.

<sup>†</sup>This command was added in GRMON 3.1.2, TSIM 2.0.65. For older versions, See FAQ Q7 for more information.

### 3.3. Restarting an application

After the debug configuration has been created and run once, it can be started again by selecting it under the menu **Run -> Debug History**.



## 4. FAQ / TROUBLESHOOTING

**Q1.** After I have created my Cross GCC project, I get two errors in the Problems view: Program "g++" not found in PATH and Program "gcc" not found in PATH.

**A** This is because Eclipse defaults to using gcc for the indexing and the errors show up because you don't have any native GCC toolchain installed on your computer. These errors are harmless, as far as we know, but might cause Eclipse to report false positive problems.

Follow the steps below to remove the errors.

1. Open the menu **Project -> Properties**
2. Select **C/C++ General -> Preprocessor Include Paths, Macros etc.**
3. Open the tab **Providers**.
4. Uncheck **CDT Cross GCC Built-in Compiler settings**
5. Press **Apply**
6. Re-check **CDT Cross GCC Built-in Compiler settings**
7. Press **Apply and close**

**Q2.** When I build my application I get the error message Error: Program "make" not found in PATH.

**A** The Cross GCC plugins uses the GNU Make builder to build the application. The GNU Make builder requires the program make to be installed on your computer. Make can be downloaded from from the GnuWin32 project [<http://gnuwin32.sourceforge.net/packages/make.htm>].

Another solution is to switch to the CDT Internal builder instead, which is independant of make. Follow the steps below to switch build system.

1. Open the menu **Project -> Properties**
2. Select **C/C++ Build -> Tool Chain Editor**
3. At **Current Builder** select **CDT Internal Builder** from the drop down list.
4. Repeat the previous step for each **configuration**.
5. Press the button **Apply and Close** to apply the changes and close the dialog.

**Q3.** Why don't I get any output in Eclipse when I start GRMON with "-u"?

**A** GRMON does forward all output to Eclipse, but Eclipse does not print it. A workaround is to start GRMON with "-ucli" instead, which print all the output in the GRMON console instead. See Q4 for additional information.

**Q4.** Why do I get the error message "ERROR! Unknown option (-ucli)" when starting GRMON?

**A** This option was added in GRMON 3.1.2. Earlier versions of GRMON does not support I/O forwarding when using Eclipse.

**Q5.** When launching an debug configuration I get the error message: "Failed to execute MI command: -gdb-set auto-solib-add on "Error message from debugger back end: No symbol table is loaded. Use the "file" command. What am I doing wrong?"

**A** Your Eclipse version is not compatible with the GDB 6.8 version that you are using. Either you need to downgrade you Eclipse version or switch to GDB 8. GDB 8 is distributed in BCC 2.1.0 but requires GRMON3 or TSIM3.

**Q6.** When launching an debug configuration I get the error message: "Failed to execute MI command: monitor gdb reset". What am I doing wrong?

**A** This command is available in GRMON 3.1.2 or later and TSM 2.0.65 or later. A workaround for earlier versions of GRMON is to add "monitor reset" and for earlier versions of TSIM2 you can leave it empty. These workarounds works in most cases.

**Q7.** When launching an debug configuration I get the error message: "Failed to execute MI command: monitor gdb postload". What am I doing wrong?

**A** This command is only available in GRMON 3.1.2 or later and TSM 2.0.65 or later. For earlier versions that are running a single core application you can leave this field empty. It should work in most cases.

For an SMP application, when using earlier versions of GRMON, you should add this line instead:  
silent for {set c 1} {\$c < [llength [lindex [cpu] 1]]} {incr c} {reg pc [reg pc cpu0] npc [reg npc cpu0] cpu[set c]}

- Q8.** How can I get the output from my application in a console inside Eclipse?
- A** Open a Local Terminal, or a Command Shell console, in Eclipse and start GRMON/TSIM from there. To open a Local Terminal:
1. Select the Terminal view
  2. Click on the **Open a Terminal** icon on the Terminal view toolbar
  3. In the **Choose terminal** drop-down list, select **Local Terminal**
  4. Press **OK**
- Q9.** How do I issue a TSIM/GRMON monitor command from within Eclipse?
- A** In Eclipse open the GDB debugger console through the console dropdown. Issue a “monitor <command>”. The output of the monitor command will appear in the console of the application being debugged.
- Q10.** I try to run TSIM/GRMON command with the monitor command, but it only works the first time.
- A** Eclipse sends all output from the target to a separate console window. When you run the GDB monitor command the output goes to that console window instead of the GDB console and Eclipse will automatically select it. To run a new command select the GDB console again by clicking on the GDB process or using the console selection button.
- Q11.** I get the error message: "error creating session localhost:2222: Bad file descriptor" . What am I doing wrong?
- A** This error message is usually caused by GRMON/TSIM not being able to launch correctly. Make sure that your launch configuration settings are correct and that GRMON/TSIM is launched with the right settings.
- Q12.** When the debug sessions start the following messages are printed (or similar): No symbol "auto" in current context. monitor symbol C:\workspace\rtms-tasks\Debug\rtms-tasks Previous frame identical to this frame (corrupt stack?) The target endianness is set automatically (currently big endian)
- A** These are just informational messages from GDB and can be ignored.

## 5. Support

Questions related to Eclipse usage is not covered by the Cobham Gaisler support. Please contact the Eclipse Community Forums or other online resources.

For support related to Cobham Gaisler tools, e.g. GRMON, TSIM or this guide itself, contact the Cobham Gaisler support team at [support@gaisler.com](mailto:support@gaisler.com).

When contacting support, please identify yourself in full, including company affiliation and site name and address. Please identify exactly what product that is used, specifying if it is an IP core (with full name of the library distribution archive file), component, software version, compiler version, operating system version, debug tool version, simulator tool version, board version, etc.

The support service is only for paying customers with a support contract.

**Cobham Gaisler AB**  
Kungsgatan 12  
411 19 Gothenburg  
Sweden  
[www.cobham.com/gaisler](http://www.cobham.com/gaisler)  
[sales@gaisler.com](mailto:sales@gaisler.com)  
T: +46 31 7758650  
F: +46 31 421407

Cobham Gaisler AB, reserves the right to make changes to any products and services described herein at any time without notice. Consult Cobham or an authorized sales representative to verify that the information in this document is current before using this product. Cobham does not assume any responsibility or liability arising out of the application or use of any product or service described herein, except as expressly agreed to in writing by Cobham; nor does the purchase, lease, or use of a product or service from Cobham convey a license under any patent rights, copyrights, trademark rights, or any other of the intellectual rights of Cobham or of third parties. All information is provided as is. There is no warranty that it is correct or suitable for any purpose, neither implicit nor explicit.

Copyright © 2020 Cobham Gaisler AB