

# ARM® DS-5 Community Edition

Version 5.25

## Getting Started Guide



# ARM® DS-5 Community Edition

## Getting Started Guide

Copyright © 2016 ARM. All rights reserved.

### Release Information

### Document History

Issue	Date	Confidentiality	Change
A	15 March 2016	Non-Confidential	First release for DS-5 Community Edition
B	15 July 2016	Non-Confidential	Update for DS-5 Community Edition version 5.25

### Non-Confidential Proprietary Notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of ARM. **No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.**

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

THIS DOCUMENT IS PROVIDED “AS IS”. ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, ARM makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, third party patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word “partner” in reference to ARM’s customers is not intended to create or refer to any partnership relationship with any other company. ARM may make changes to this document at any time and without notice.

If any of the provisions contained in these terms conflict with any of the provisions of any signed written agreement covering this document with ARM, then the signed written agreement prevails over and supersedes the conflicting provisions of these terms. This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

Words and logos marked with ® or ™ are registered trademarks or trademarks of ARM Limited or its affiliates in the EU and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow ARM’s trademark usage guidelines at <http://www.arm.com/about/trademark-usage-guidelines.php>

Copyright © 2016, ARM Limited or its affiliates. All rights reserved.

ARM Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20349

### Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by ARM and the party that ARM delivered this document to.

Unrestricted Access is an ARM internal classification.

**Product Status**

The information in this document is Final, that is for a developed product.

**Web Address**

<http://www.arm.com>

# Contents

## ARM® DS-5 Community Edition Getting Started Guide

### **Preface**

<i>About this book</i> .....	9
------------------------------	---

### **Chapter 1**

#### **ARM® DS-5 Community Edition Product Overview**

1.1	<i>About DS-5 Community Edition</i> .....	1-12
1.2	<i>About Eclipse for DS-5</i> .....	1-13
1.3	<i>About DS-5 Debugger</i> .....	1-14
1.4	<i>About Fixed Virtual Platform (FVP)</i> .....	1-15
1.5	<i>About ARM® Streamline Performance Analyzer</i> .....	1-16
1.6	<i>Debug options supported by DS-5 Community Edition</i> .....	1-17

### **Chapter 2**

#### **ARM® DS-5 Community Edition installation and system requirements**

2.1	<i>System requirements</i> .....	2-19
2.2	<i>Installing DS-5 Community Edition</i> .....	2-20
2.3	<i>Installation directories</i> .....	2-21

### **Chapter 3**

#### **Working with ARM® DS-5 Community Edition**

3.1	<i>Documentation provided with DS-5</i> .....	3-23
3.2	<i>Examples provided with DS-5 Community Edition</i> .....	3-24
3.3	<i>Importing the example projects into Eclipse</i> .....	3-25
3.4	<i>Loading the Gnometriz application on to an ARM® Linux target</i> .....	3-26
3.5	<i>Configuring an RSE connection to work with an ARM® Linux target</i> .....	3-27

3.6	<i>Debugging Gnometris .....</i>	3-38
3.7	<i>Performance analysis of the threads application running on ARM® Linux .....</i>	3-39
3.8	<i>About registering a new compiler toolchain .....</i>	3-41
3.9	<i>Registering a compiler toolchain from the DS-5 command prompt .....</i>	3-42
3.10	<i>Configuring a compiler toolchain for the DS-5 command prompt .....</i>	3-46
3.11	<i>Registering a compiler toolchain from Eclipse .....</i>	3-47

# List of Figures

## ARM® DS-5 Community Edition Getting Started Guide

Figure 3-1	Cheat Sheet Selection .....	3-25
Figure 3-2	Selecting a connection type .....	3-27
Figure 3-3	Defining the connection information .....	3-28
Figure 3-4	Defining the file system .....	3-29
Figure 3-5	Defining the processes .....	3-30
Figure 3-6	Defining the shell services .....	3-31
Figure 3-7	Defining the terminal services .....	3-32
Figure 3-8	Modifying file properties from the Remote Systems view .....	3-33
Figure 3-9	Typical connection configuration for Linux application debug .....	3-35
Figure 3-10	Typical file selection for Linux application debug .....	3-36
Figure 3-11	Typical debugger settings for Linux application debug .....	3-37
Figure 3-12	Streamline Data view .....	3-39
Figure 3-13	Streamline analysis report for the threads application .....	3-40
Figure 3-14	Registering a new toolchain .....	3-42
Figure 3-15	Registering a new toolchain .....	3-43
Figure 3-16	Using a new toolchain for a new project .....	3-44
Figure 3-17	Changing the toolchain for a project .....	3-45
Figure 3-18	Configuring a default toolchain .....	3-46
Figure 3-19	Toolchains Preferences dialog .....	3-47
Figure 3-20	Properties for the new toolchain .....	3-48

List of Tables

**ARM® DS-5 Community Edition Getting Started Guide**

*Table 2-1*      *DS-5 default directories* ..... 2-21

# Preface

This preface introduces the *ARM® DS-5 Community Edition Getting Started Guide*.

It contains the following:

- [About this book](#) on page 9.



## About this book

This book gives an overview of ARM® DS-5 Community Edition. It describes the installation and system requirements. It also explains how to work with examples provided with DS-5 Community Edition.

## Using this book

This book is organized into the following chapters:

### **Chapter 1 ARM® DS-5 Community Edition Product Overview**

Gives an overview of the main features of ARM® DS-5 Community Edition.

### **Chapter 2 ARM® DS-5 Community Edition installation and system requirements**

This chapter provides information on the installation and system requirements for ARM DS-5 Community Edition.

### **Chapter 3 Working with ARM® DS-5 Community Edition**

This chapter explains how to run and debug applications using ARM DS-5 tools. It also provides information about the examples and documentation provided with DS-5 Community Edition

## Glossary

The ARM Glossary is a list of terms used in ARM documentation, together with definitions for those terms. The ARM Glossary does not contain terms that are industry standard unless the ARM meaning differs from the generally accepted meaning.

See the [ARM Glossary](#) for more information.

## Typographic conventions

*italic*

Introduces special terminology, denotes cross-references, and citations.

**bold**

Highlights interface elements, such as menu names. Denotes signal names. Also used for terms in descriptive lists, where appropriate.

monospace

Denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

monospace

Denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.

monospace *italic*

Denotes arguments to monospace text where the argument is to be replaced by a specific value.

monospace **bold**

Denotes language keywords when used outside example code.

<and>

Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example:

```
MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>
```

SMALL CAPITALS

Used in body text for a few terms that have specific technical meanings, that are defined in the *ARM glossary*. For example, IMPLEMENTATION DEFINED, IMPLEMENTATION SPECIFIC, UNKNOWN, and UNPREDICTABLE.

## Feedback

## Feedback on this product

If you have any comments or suggestions about this product, contact your supplier and give:

- The product name.
- The product revision or version.
- An explanation with as much information as you can provide. Include symptoms and diagnostic procedures if appropriate.

## Feedback on content

If you have comments on content then send an e-mail to [errata@arm.com](mailto:errata@arm.com). Give:

- The title *ARM® DS-5 Community Edition Getting Started Guide*.
- The number ARM DUI0962B.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

---

### Note

ARM tests the PDF only in Adobe Acrobat and Acrobat Reader, and cannot guarantee the quality of the represented document when used with any other PDF reader.

---

## Other information

- [ARM Information Center](#).
- [ARM Technical Support Knowledge Articles](#).
- [Support and Maintenance](#).
- [ARM Glossary](#).

# Chapter 1

## ARM® DS-5 Community Edition Product Overview

Gives an overview of the main features of ARM® DS-5 Community Edition.

It contains the following sections:

- [1.1 About DS-5 Community Edition on page 1-12.](#)
- [1.2 About Eclipse for DS-5 on page 1-13.](#)
- [1.3 About DS-5 Debugger on page 1-14.](#)
- [1.4 About Fixed Virtual Platform \(FVP\) on page 1-15.](#)
- [1.5 About ARM® Streamline Performance Analyzer on page 1-16.](#)
- [1.6 Debug options supported by DS-5 Community Edition on page 1-17.](#)

## 1.1 About DS-5 Community Edition

DS-5 Community Edition is a free software development solution based on our professional tool chain.

It provides support for:

- Linux-based application development on a target platform of your choice.
- Bare-metal or Linux kernel development on one of two Fixed Virtual Platform (FVP) targets, one based on the ARM ARMv7-A architecture and one on the ARMv8-A architecture.

It includes:

- Eclipse for DS-5, an *Integrated Development Environment* (IDE) that combines the Eclipse IDE from the Eclipse Foundation with the debug technology of the ARM tools.
- DS-5 Debugger, a graphical debugger supporting software development on ARM processor-based targets and *Fixed Virtual Platform* (FVP) targets.
- *Fixed Virtual Platform* (FVP) targets for architectures ARMv7-A and ARMv8-A, enabling development of software without the requirement for actual hardware.
- ARM Streamline, a graphical performance analysis tool.
- Dedicated examples, applications, and supporting documentation to help you get started with using the DS-5 tools.

Some third-party compilers are compatible with DS-5. For example, the GNU Compiler tools enable you to compile bare-metal, Linux kernel, and Linux applications for ARM targets.

### Related references

[2.1 System requirements on page 2-19.](#)

[2.2 Installing DS-5 Community Edition on page 2-20.](#)

[2.3 Installation directories on page 2-21.](#)

[3.1 Documentation provided with DS-5 on page 3-23.](#)

[3.2 Examples provided with DS-5 Community Edition on page 3-24.](#)

### Related information

[DS-5 Developer Resources.](#)

## 1.2 About Eclipse for DS-5

Eclipse for DS-5 is an *Integrated Development Environment* (IDE) that combines the Eclipse IDE from the Eclipse Foundation with the compilation and debug technology of the ARM tools..

It includes:

### **Project manager**

The project manager enables you to perform various project tasks such as adding or removing files and dependencies to projects, importing, exporting, or creating projects, and managing build options.

### **Editors**

Editors enables you read, write, or modify C/C++ or ARM assembly language source files.

### **Perspectives and views**

Perspectives provide customized views, menus, and toolbars to suit a particular type of environment. DS-5 uses the **C/C++**, **DS-5 Debug**, and **DS-5 Configuration** perspectives. To switch perspectives, from the main menu, select **Window > Open Perspective**.

### **Related tasks**

[3.3 Importing the example projects into Eclipse on page 3-25.](#)

[Building the gnometris project from Eclipse.](#)

### **Related information**

[Eclipse for ARM DS-5 User Guide.](#)

## 1.3 About DS-5 Debugger

DS-5 Debugger, a graphical debugger supporting software development on ARM processor-based targets and *Fixed Virtual Platform* (FVP) targets..

It makes it easy to debug bare-metal and Linux applications with comprehensive and intuitive views, including synchronized source and disassembly, call stack, memory, registers, expressions, variables, threads, and breakpoints.

Using the Debug Control view, you can single-step through applications at source-level or instruction-level and see the other views update as the code is executed. Setting breakpoints or watchpoints can assist you by stopping the application and enabling you to explore the behavior of the application.

You can also debug using the **DS-5 Command Prompt** command-line console.

### Related tasks

[3.6 Debugging Gnometriz on page 3-38.](#)

## 1.4 About *Fixed Virtual Platform* (FVP)

*Fixed Virtual Platform* (FVP) targets enable development of software without the requirement for actual hardware. The functional behavior of an *Fixed Virtual Platform* (FVP) is equivalent to real hardware from a programmers view.

When using an *Fixed Virtual Platform* (FVP), absolute timing accuracy is sacrificed to achieve fast simulated execution speed. This means that you can use a model for confirming software functionality, but you must not rely on the accuracy of cycle counts, low-level component interactions, or other hardware-specific behavior.

DS-5 Community Edition provides a single-core Cortex®-A9 and a Foundation Platform model executable.

The executables are located in `tools_directory`. You can use them to run your applications from either the command-line or within Eclipse. See [2.3 \*Installation directories\* on page 2-21](#) for more information about various directories that are installed with DS-5.

### **Related tasks**

[Loading the Gnometris application on a Fixed Virtual Platform \(FVP\).](#)

## 1.5 About ARM® Streamline Performance Analyzer

ARM Streamline is a graphical performance analysis tool that enables you to transform sampling data and system trace into reports that present the data in both visual and statistical forms.

Streamline uses hardware performance counters with kernel metrics to provide an accurate representation of system resources.

### Related tasks

[3.7 Performance analysis of the threads application running on ARM® Linux](#) on page 3-39.



## 1.6 Debug options supported by DS-5 Community Edition

DS-5 Community Edition supports various debug options.

Debug adapters vary in complexity and capability but, combined with software debug agents, they provide high-level debug functionality for the target that is being debugged, for example:

- Reading/Writing registers.
- Setting breakpoints.
- Reading from memory.
- Writing to memory.

Supported debug connections include:

- CADI (debug interface for models).
- Ethernet to gdbserver.

## Chapter 2

# **ARM® DS-5 Community Edition installation and system requirements**

This chapter provides information on the installation and system requirements for ARM DS-5 Community Edition.

It contains the following sections:

- [2.1 System requirements on page 2-19.](#)
- [2.2 Installing DS-5 Community Edition on page 2-20.](#)
- [2.3 Installation directories on page 2-21.](#)

## 2.1 System requirements

To install and use DS-5 Community Edition, your workstation must have a minimum specification of a dual core 2GHz processor (or equivalent) and 2GB of RAM.

To improve performance when debugging large images, using models with large simulated memory maps, or when using ARM Streamline Performance Analyzer, 4GB of RAM, or more is recommended.

A full installation also requires approximately 1GB of hard disk space.

### Host platform requirements

DS-5 Community Edition is supported on the following host platforms and service packs.

- Windows 10 (64-bit only)
- Windows 7 (64-bit only) with Service Pack 1
- Red Hat Enterprise Linux 6 Workstation (64-bit only)
- Red Hat Enterprise Linux 7 Workstation (64-bit only)
- Ubuntu Desktop Edition 12.04 LTS (64-bit only)
- Ubuntu Desktop Edition 14.04 LTS (64-bit only)

### Debug system requirements

Linux application debug requires `gdbserver` on your target. The recommended version of `gdbserver` is 7.0 or later.

---

#### Note

DS-5 Debugger is unable to provide reliable multi-threaded debug support with `gdbserver` versions prior to 6.8.

---

DS-5 support for Linux application debug depends on infrastructure and features that are introduced in specific kernel versions:

- DS-5 Debugger supports debugging ARM Linux kernel versions 2.6.28 and later.
- Application debug on *Symmetric MultiProcessing* (SMP) systems requires ARM Linux kernel version 2.6.36 or later.
- Access to VFP and NEON registers requires ARM Linux kernel version 2.6.30 or later and `gdbserver` version 7.0 or later.
- ARM Streamline Performance Analyzer supports ARM Linux kernel versions 3.4 and later.

## 2.2 Installing DS-5 Community Edition

You can install DS-5 Community Edition on 64-bit Windows and Linux platforms.

### Installing on Linux

To install DS-5 on Linux, run (not source) `install.sh` and follow the on-screen instructions.

#### ————— Note —————

On Linux, you can use `suite_exec` to configure the environment variables correctly for DS-5. For example, run `DS-5_install_directory/bin/suite_exec <shell>` to open a shell with the PATH and other environment variables correctly configured. Run `suite_exec` with no arguments for more help.

### Installing on Windows

To install DS-5 on Windows, run `setup.exe` and follow the on-screen instructions.

### Command-line installation on Windows

Command-line installation and uninstallation are possible on Windows by opening a command prompt, with administrative privileges, and running Microsoft's installer, `msiexec.exe`. You must provide the location of the `.msi` file as an argument to `msiexec`. You can get a full list of options for using `msiexec` by running `msiexec /?` on the command-line. An example of how to install DS-5 using `msiexec` is:

```
msiexec.exe /i installer_location\data\install.msi EULA=1 /qn /l*v install.log
```

Where:

`/i`

This option is to perform the installation.

`installer_location\data\install.msi`

This specifies the full pathname of the `.msi` file to install.

`/EULA=1`

This is an ARM specific option. Setting EULA to 1 means you accept the End User License Agreement (EULA). You must read the EULA in the GUI installer before accepting it on the command-line.

`/qn`

This option specifies quiet mode, so that the installation does not require user interaction.

`/l*v install.log`

This option specifies the log file to log all output from the installation.

## 2.3 Installation directories

Various directories are installed with DS-5 that contain example code and documentation. The DS-5 documentation refers to these directories as required.

The main installation, examples, and documentation directories are identified in the following table. The *DS-5\_install\_directory* shown is the default installation directory. The DS-5 version number, *<version>*, is part of the default installation directory name. If you installed the product in a different directory, then the path names are relative to your chosen directory.

**Table 2-1 DS-5 default directories**

Directory	Windows	Linux
<i>DS-5_install_directory</i>	For 64-bit version of Windows with DS-5 Community Edition installed: C:\Program Files\DS-5 CE v<version>	~/DS-5_CE_v<version>
<i>examples_directory</i>	<i>DS-5_install_directory</i> \examples\...	<i>DS-5_install_directory</i> /examples/...
<i>tools_directory</i>	<i>DS-5_install_directory</i> \bin\...	<i>DS-5_install_directory</i> /bin/...

# Chapter 3

## Working with ARM® DS-5 Community Edition

This chapter explains how to run and debug applications using ARM DS-5 tools. It also provides information about the examples and documentation provided with DS-5 Community Edition

It contains the following sections:

- [3.1 Documentation provided with DS-5 on page 3-23.](#)
- [3.2 Examples provided with DS-5 Community Edition on page 3-24.](#)
- [3.3 Importing the example projects into Eclipse on page 3-25.](#)
- [3.4 Loading the Gnometris application on to an ARM® Linux target on page 3-26.](#)
- [3.5 Configuring an RSE connection to work with an ARM® Linux target on page 3-27.](#)
- [3.6 Debugging Gnometris on page 3-38.](#)
- [3.7 Performance analysis of the threads application running on ARM® Linux on page 3-39.](#)
- [3.8 About registering a new compiler toolchain on page 3-41.](#)
- [3.9 Registering a compiler toolchain from the DS-5 command prompt on page 3-42.](#)
- [3.10 Configuring a compiler toolchain for the DS-5 command prompt on page 3-46.](#)
- [3.11 Registering a compiler toolchain from Eclipse on page 3-47.](#)

## 3.1 Documentation provided with DS-5

DS-5 includes example projects and documentation.

To access the documentation from within DS-5, from the main menu, select **Help > Help Contents** and navigate to **ARM DS-5 Documentation**.

Documentation on using the examples is available in *DS-5\_install\_directory\examples\docs*.

### Related information

[\*DS-5 Developer Documentation\*](#).

## 3.2 Examples provided with DS-5 Community Edition

DS-5 Community Edition provides a selection of examples to help you get started:

- Bare-metal software development examples for ARMv7 and earlier that illustrate:
  - Compilation with GCC bare-metal compiler.
  - ARMv7 bare-metal debug.

The code is located in the archive file `<examples_directory>\Bare-metal_examples_ARMv7.zip`.

- Bare-metal software development examples for ARMv8 that illustrate:
  - Compilation with GCC bare-metal compiler.
  - ARMv8 bare-metal debug.

The code is located in the archive file `<examples_directory>\Bare-metal_examples_ARMv8.zip`.

- ARM Linux examples built with GCC Linux compiler that illustrate build, debug, and performance analysis of simple C/C++ console applications, shared libraries, and multi-threaded applications. These examples run on ARM Linux targets using gdbserver. The files are located in the archive file, `examples_directory\Linux_examples.zip`.

You can extract these examples to a working directory and build them from the command-line, or you can import them into Eclipse using the import wizard. All examples provided with DS-5 contain a preconfigured Eclipse launch script that enables you to easily load and debug example code on a target.

Each example provides instructions on how to build, run, and debug the example code. You can access the instructions from the main index, `examples_directory\docs\index.html`.



### 3.3 Importing the example projects into Eclipse

To use the example projects provided with DS-5, you must first import them.

#### Procedure

1. Launch Eclipse:
  - On Windows, select **Start > All Programs > ARM DS-5 > Eclipse for DS-5**.
  - On Linux, enter `eclipse` in the Unix bash shell.
2. ARM recommends that you create a new workspace for example projects so that they remain separate from your own projects. To do this you can either:
  - Create a new workspace directory during the start up of Eclipse.
  - If Eclipse is already open, select **File > Switch Workspace > Other** from the main menu.
3. In the main menu, select **Help > Cheat Sheets...**
4. Expand the **ARM Eclipse for DS-5** group.
5. Select **Automatically Importing the DS-5 Example Projects into the Current Workspace** from the list of ARM cheat sheets.

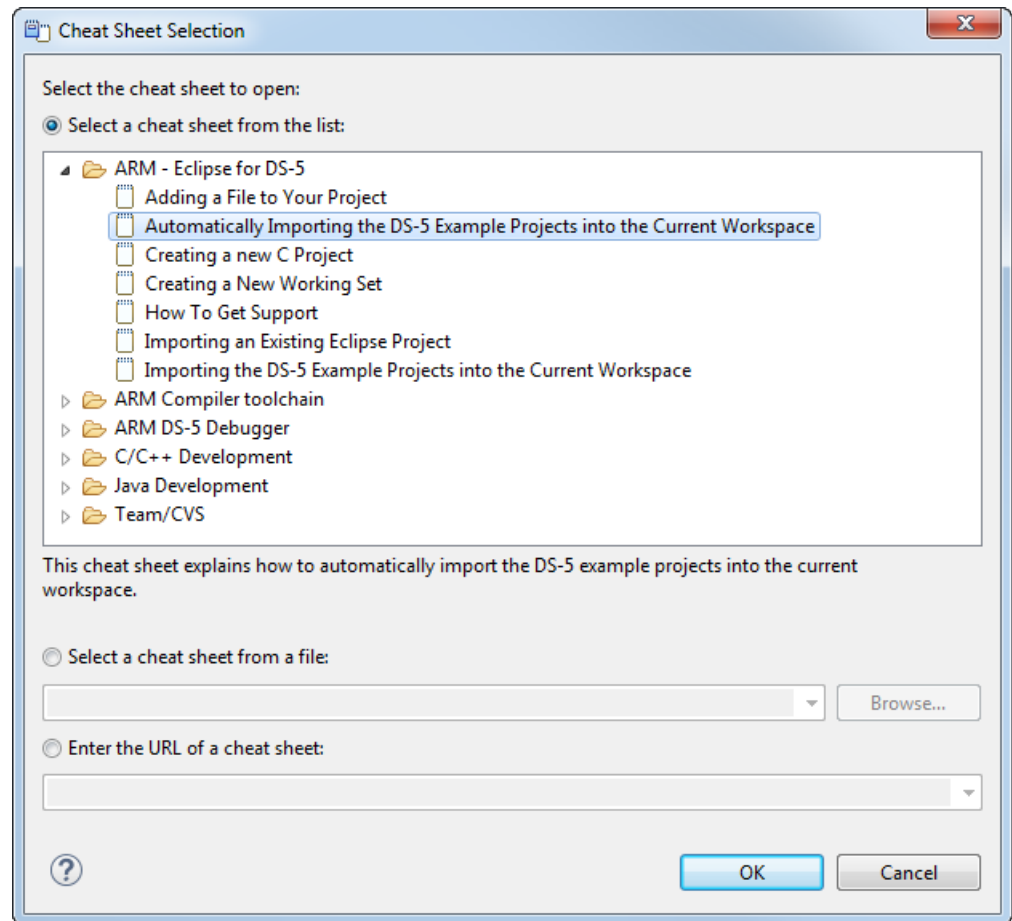


Figure 3-1 Cheat Sheet Selection

6. Click **OK**.
7. Follow the steps in the cheat sheet to import the DS-5 example projects into your workspace.  
Browse the imported examples in the Project Explorer.  
Each example contains a `readme.html` which explains how you can work with each example.

## 3.4 Loading the Gnometris application on to an ARM® Linux target

You can load the Gnometris application on to a target that is running ARM Linux. DS-5 provides preconfigured target connection settings that connect the debugger to gdbserver running on supported ARM architecture-based platforms.

### Procedure

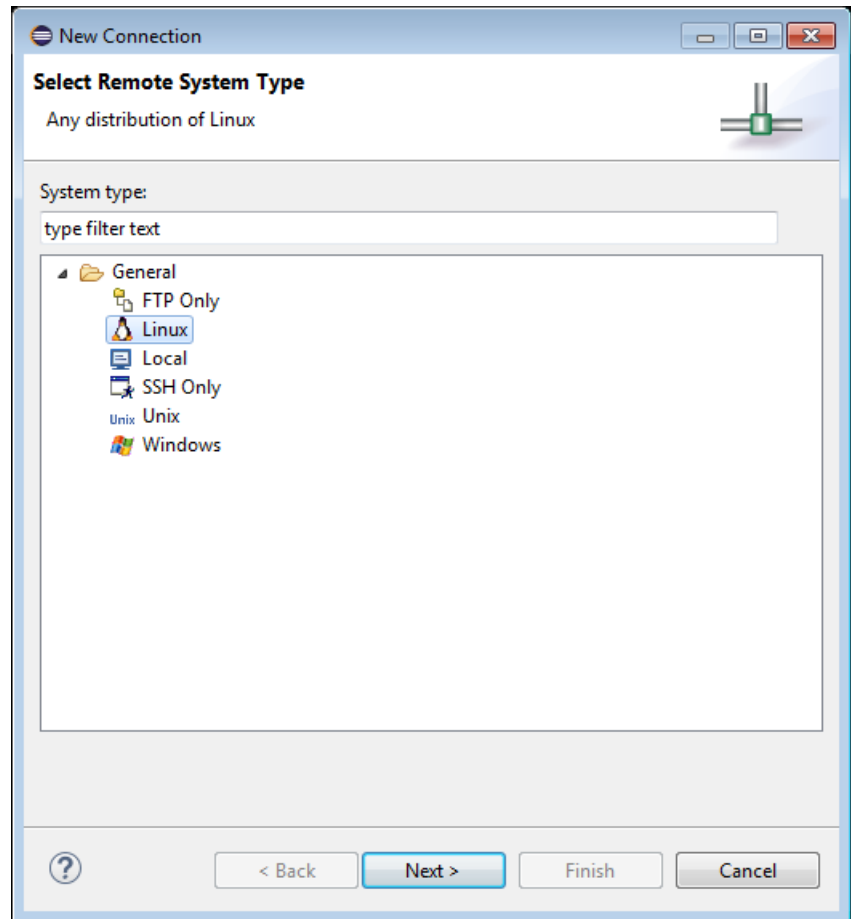
1. Obtain the IP address of the target. You can use the `ifconfig` application in a Linux console. The IP address is denoted by the **inet addr**.
2. Boot the appropriate Linux distribution on the target.
3. Launch Eclipse.
4. Transfer the application and related files to the ARM Linux target, run the application, and then connect the debugger. There are several ways to do this:
  - Use a *Secure SHell* (SSH) connection with the *Remote System Explorer* (RSE) provided with DS-5 to set up the target and run the application. When the application is running you can then connect the debugger to the running target.
  - Use an external file transfer utility such as PuTTY.

## 3.5 Configuring an RSE connection to work with an ARM® Linux target

On some targets you can use a *Secure SHell* (SSH) connection with the *Remote System Explorer* (RSE) provided with DS-5.

### Procedure

1. In the Remote Systems view, click on the **Define a connection to remote system** option on the toolbar.
2. In the Select Remote System Type dialog box, expand the **General** group and select **Linux**.



**Figure 3-2** Selecting a connection type

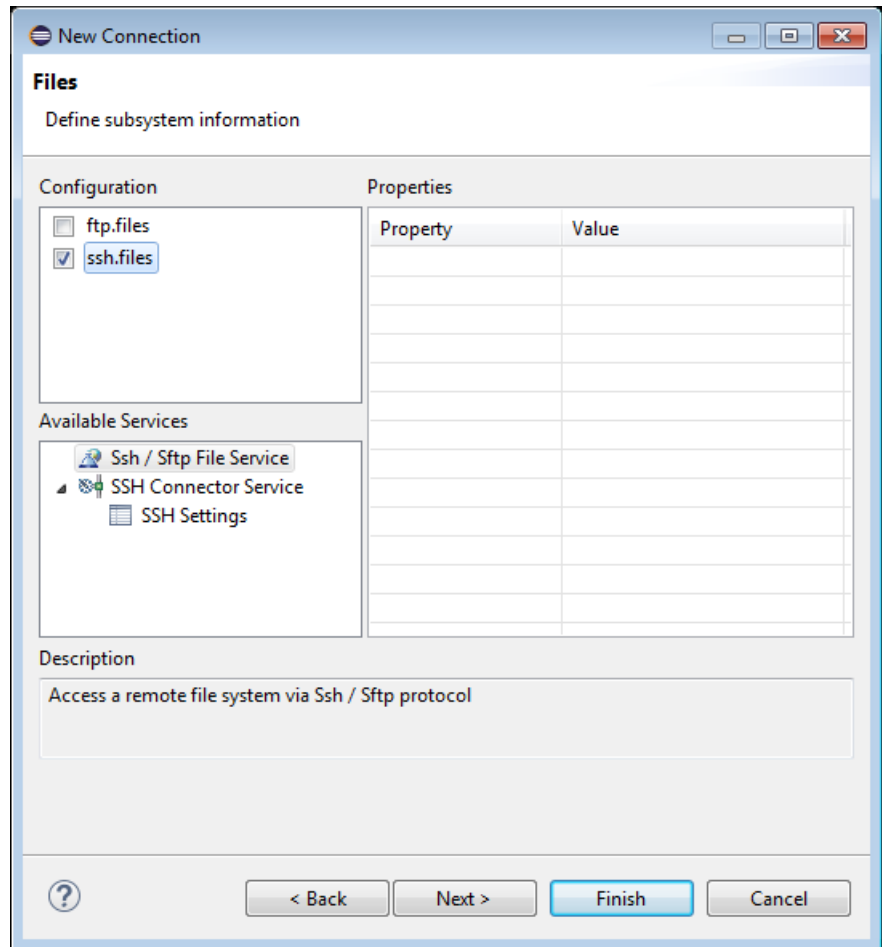
3. Click **Next**.
4. In Remote Linux System Connection, enter the remote target IP address or name in the **Host name** field.

The screenshot shows a 'New Connection' dialog box with the title 'Remote Linux System Connection'. Below the title is the instruction 'Define connection information'. The dialog contains the following fields and controls:

- Parent profile:** A dropdown menu with 'E106738' selected.
- Host name:** A text field containing '10.1.204.180'.
- Connection name:** A text field containing 'Zebra2'.
- Description:** An empty text field.
- Verify host name:** A checked checkbox.
- Configure proxy settings:** A blue hyperlink.
- Navigation buttons:** At the bottom, there is a help icon (?), a '< Back' button, a 'Next >' button, a highlighted 'Finish' button, and a 'Cancel' button.

**Figure 3-3** Defining the connection information

5. Click **Next**.
6. Select SSH protocol file access.



**Figure 3-4** Defining the file system

7. Click **Next**.
8. Select the shell processes for Linux systems.

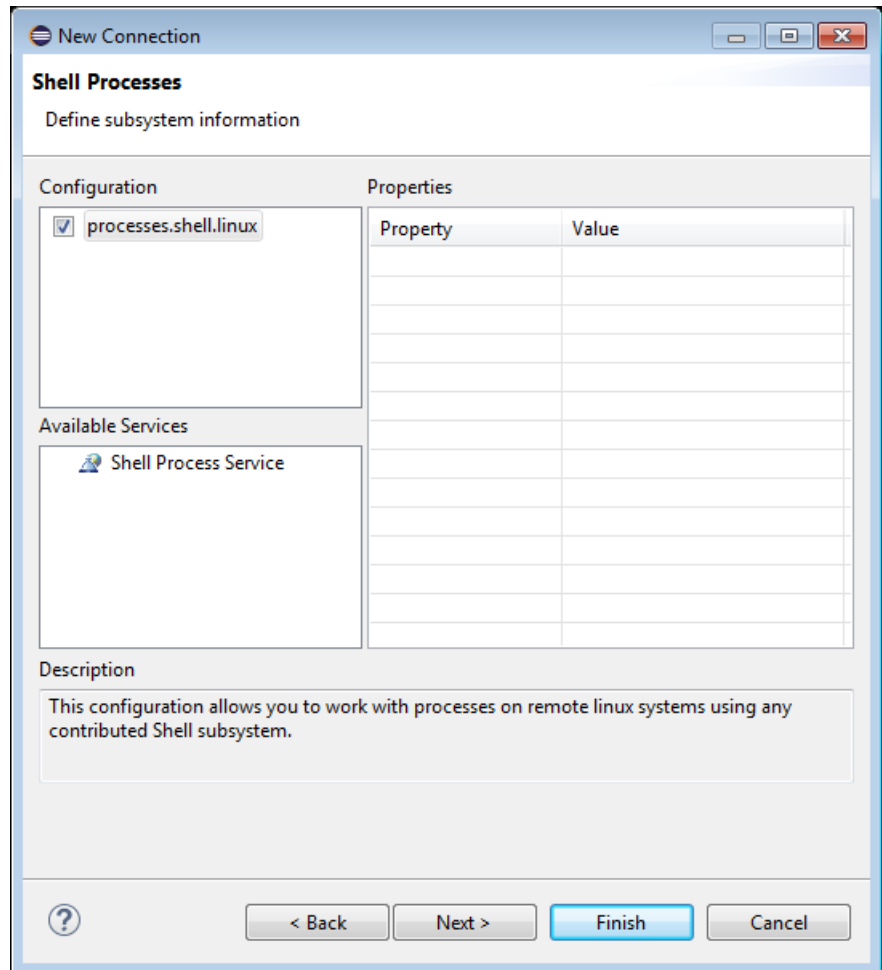
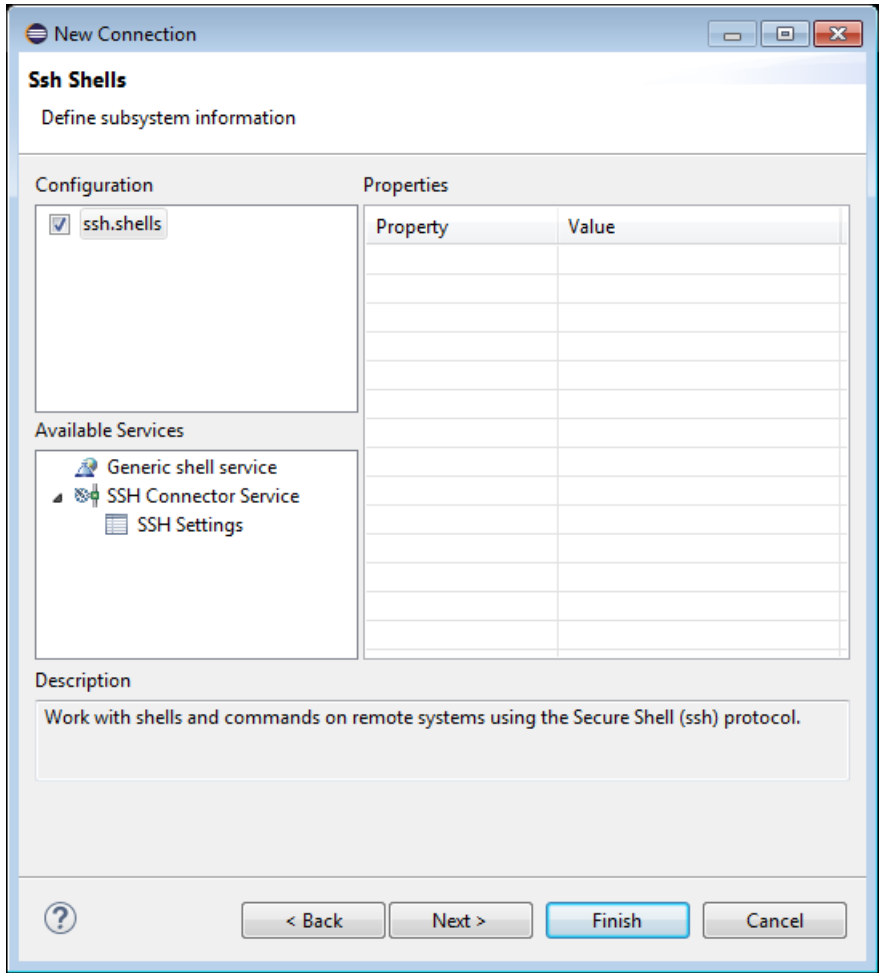


Figure 3-5 Defining the processes

9. Click **Next**.
10. Select **SSH shells**.



### Figure 3-6 Defining the shell services

11. Click **Next**.
12. Select **SSH terminals**.

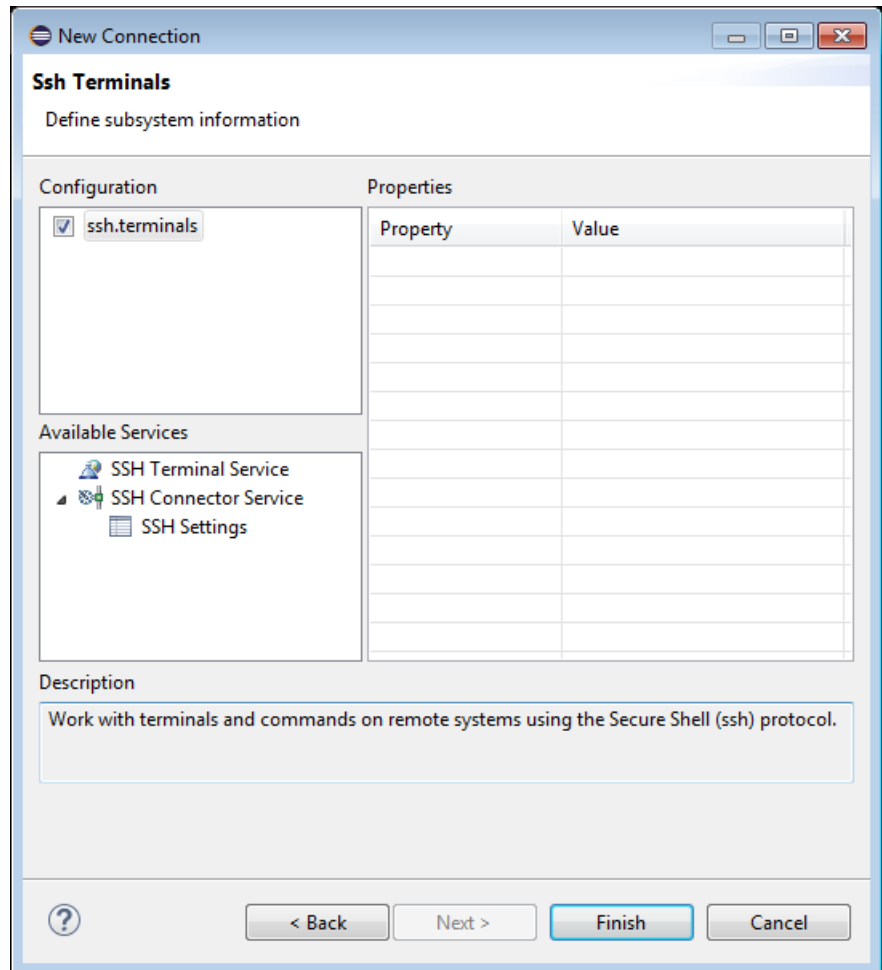


Figure 3-7 Defining the terminal services

13. Click **Finish**.

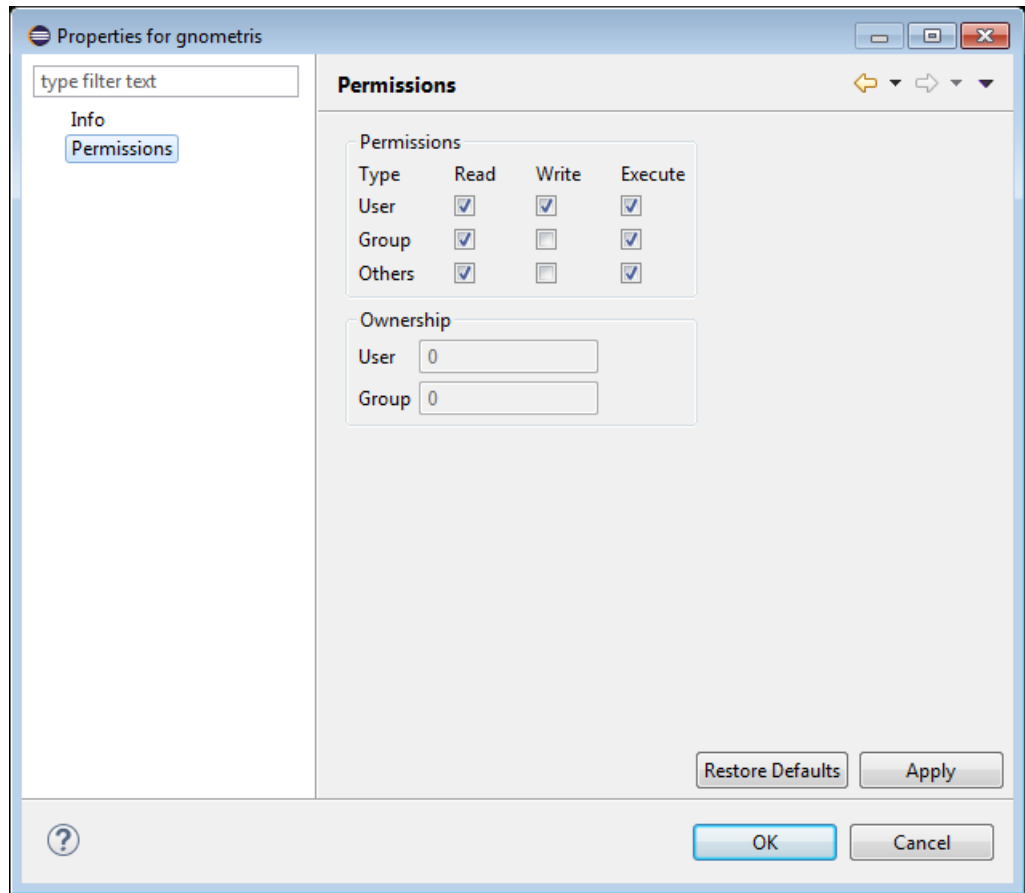
14. In the Remote Systems view:

- Right-click on the Linux target and select **Connect** from the context menu.
- In the Enter Password dialog box, enter a **UserID** and **Password** if required.
- Click **OK** to close the dialog box.
- Copy the required files from the local file system on to the target file system. You can do this by dragging and dropping the relevant files in the Remote Systems view.

This example uses Gnometriz which requires copying the stripped version of the Gnometriz application, `gnometris`, and the `libgames-support.so` library.

- Ensure that the files on the target have execute permissions. To do this, right-click on each file, select **Properties** from the context menu and select the checkboxes as required.





**Figure 3-8 Modifying file properties from the Remote Systems view**

15. Open a terminal shell that is connected to the target and launch gdbserver with the application:
  - a. In the Remote Systems view, right-click on **Ssh Terminals**.
  - b. Select **Launch Terminal** to open a terminal shell.
  - c. In the terminal shell, navigate to the directory where you copied the application, then execute the required commands.

For example, to launch Gnometriz:

```
export DISPLAY=ip:0.0
gdbserver :port gnometriz
```

where:

ip

is the IP address of the host to display the Gnometriz game

port

is the connection port between gdbserver and the application, for example 5000.

**Note**

If the target has a display connected to it then you do not need to use the export DISPLAY command.

This section contains the following subsections:

- [3.5.1 Launching gdbserver with an application on page 3-34.](#)
- [3.5.2 Connecting to the Gnometriz application that is already running on an ARM® Linux target on page 3-34.](#)

### 3.5.1 Launching gdbserver with an application

To launch `gdbserver` with the application:

#### Procedure

1. Open a terminal shell that is connected to the target.
2. In the Remote Systems view, right-click on **Ssh Terminals**.
3. Select **Launch Terminal** to open a terminal shell.
4. In the terminal shell, navigate to the directory where you copied the application, then execute the required commands.

For example, to launch Gnometris:

```
export DISPLAY=ip:0.0  
gdbserver :port gnometris
```

where:

`ip`

is the IP address of the host to display the Gnometris game

`port`

is the connection port between `gdbserver` and the application, for example 5000.

#### ————— Note —————

If the target has a display connected to it then you do not need to use the `export DISPLAY` command.

### 3.5.2 Connecting to the Gnometris application that is already running on an ARM® Linux target

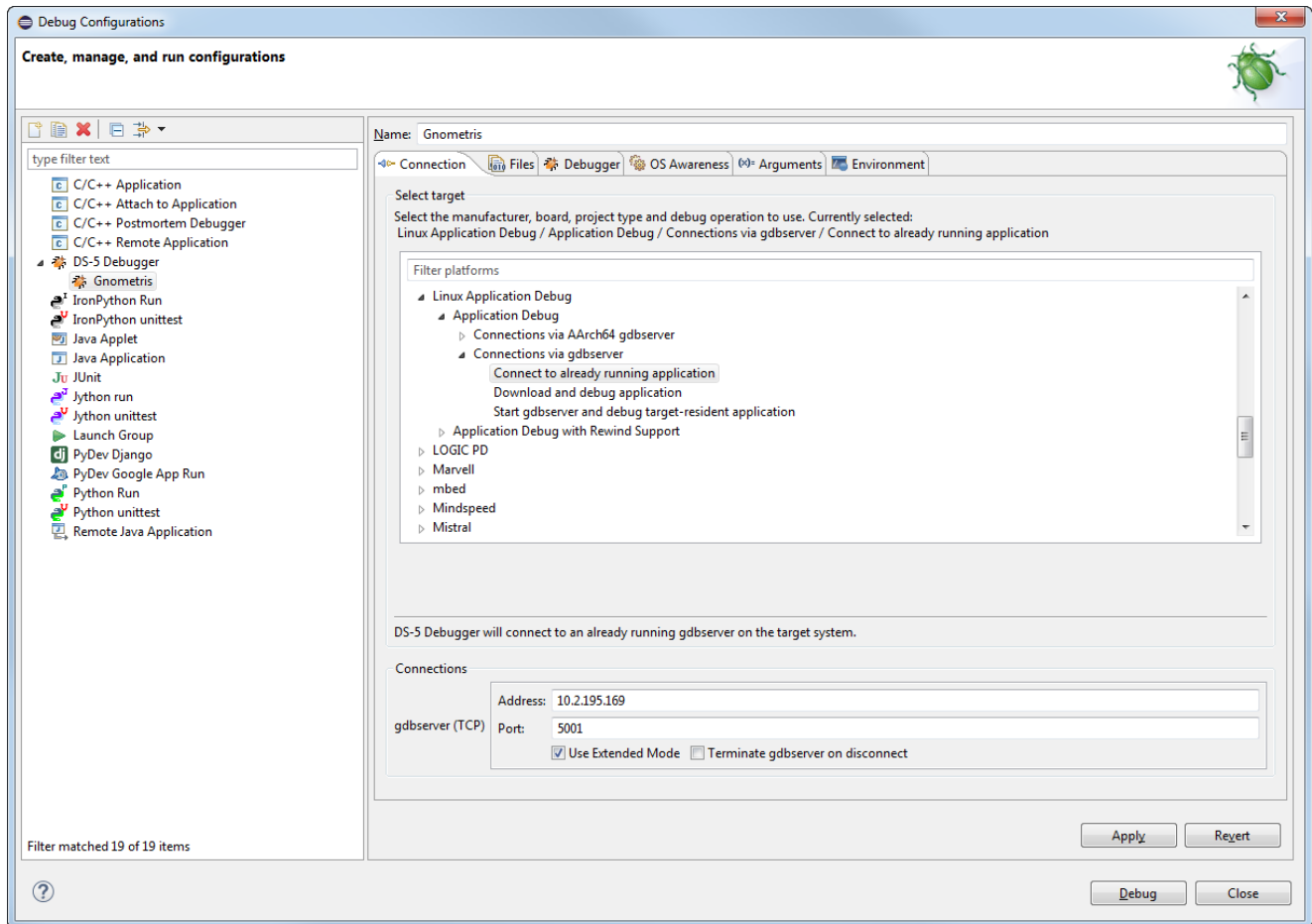
Describes how to connect to the Gnometris application that is already running on a ARM Linux target.

#### Prerequisites

- `gdbserver` and the Gnometris application running on the target and awaiting a connection on the appropriate port.
- The Gnometris application files available in your host workspace.

#### Procedure

1. Select **Debug Configurations...** from the **Run** menu.
2. Select **DS-5 Debugger** from the configuration tree and then click on **New** to create a new configuration. Alternatively you can select an existing DS-5 Debugger configuration and then click on **Duplicate** from the toolbar.
3. In the **Name** field, enter a suitable name for the new configuration.
4. Click on the **Connection** tab and:
  - a. In the **Select target** panel, browse and select **Linux Application Debug > Connections via gdbserver > Connect to already running application**.
  - b. In the **Connections** panel, enter the **TCP Address** and **Port** details of the `gdbserver` running on the target system.



**Figure 3-9 Typical connection configuration for Linux application debug**

5. Click on the **Files** tab and:
  - a. Select **Load symbols from file** and then select the application image containing debug information. For example: H:\workspace\gnometris\gnometris.
  - b. Click **Add a new resource to the list** to add another file entry.
  - c. Select **Load symbols from file** and then select the shared library that is required by the Gnometris application. For example: H:\workspace\gnometris\libgames-support.so.

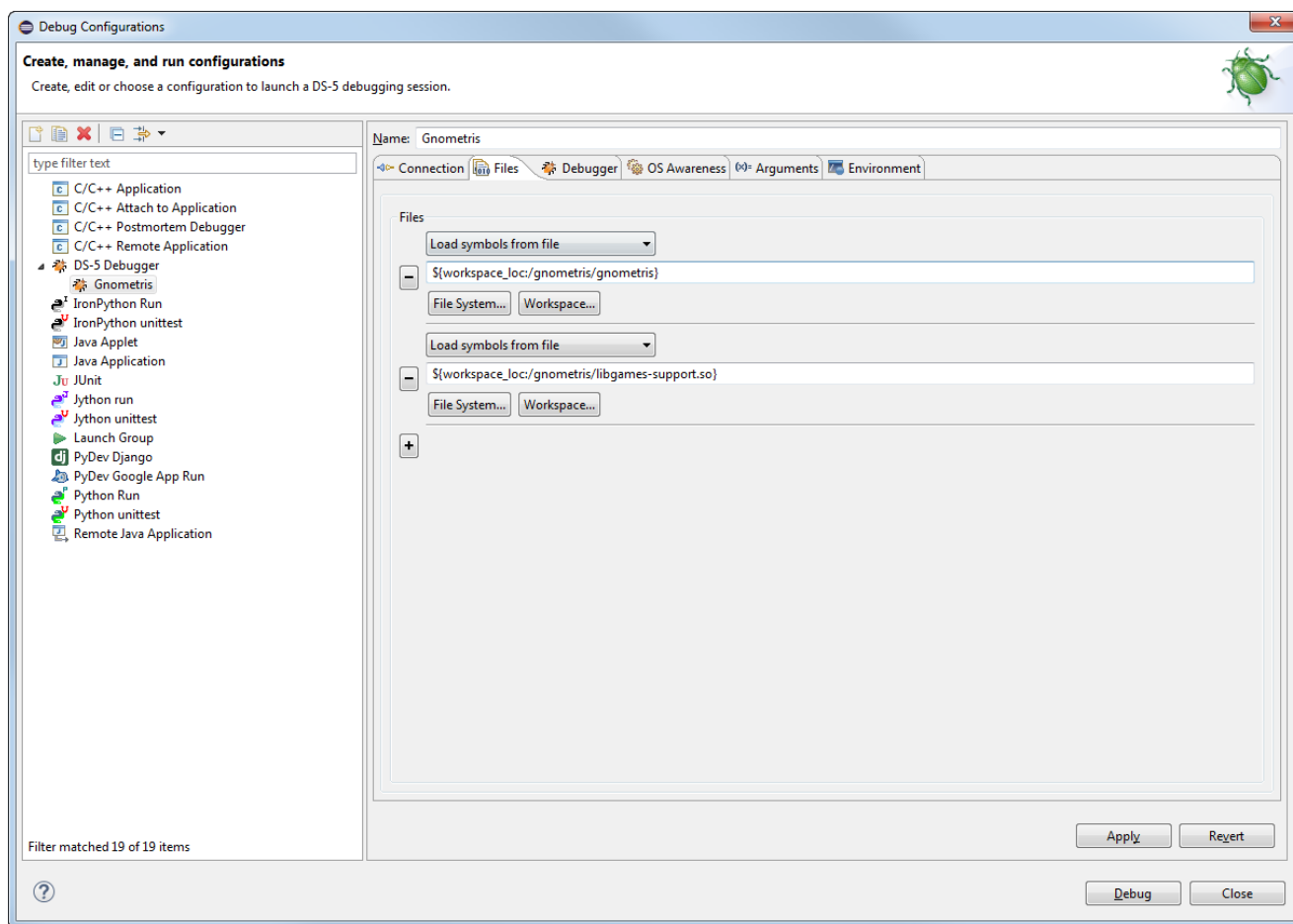
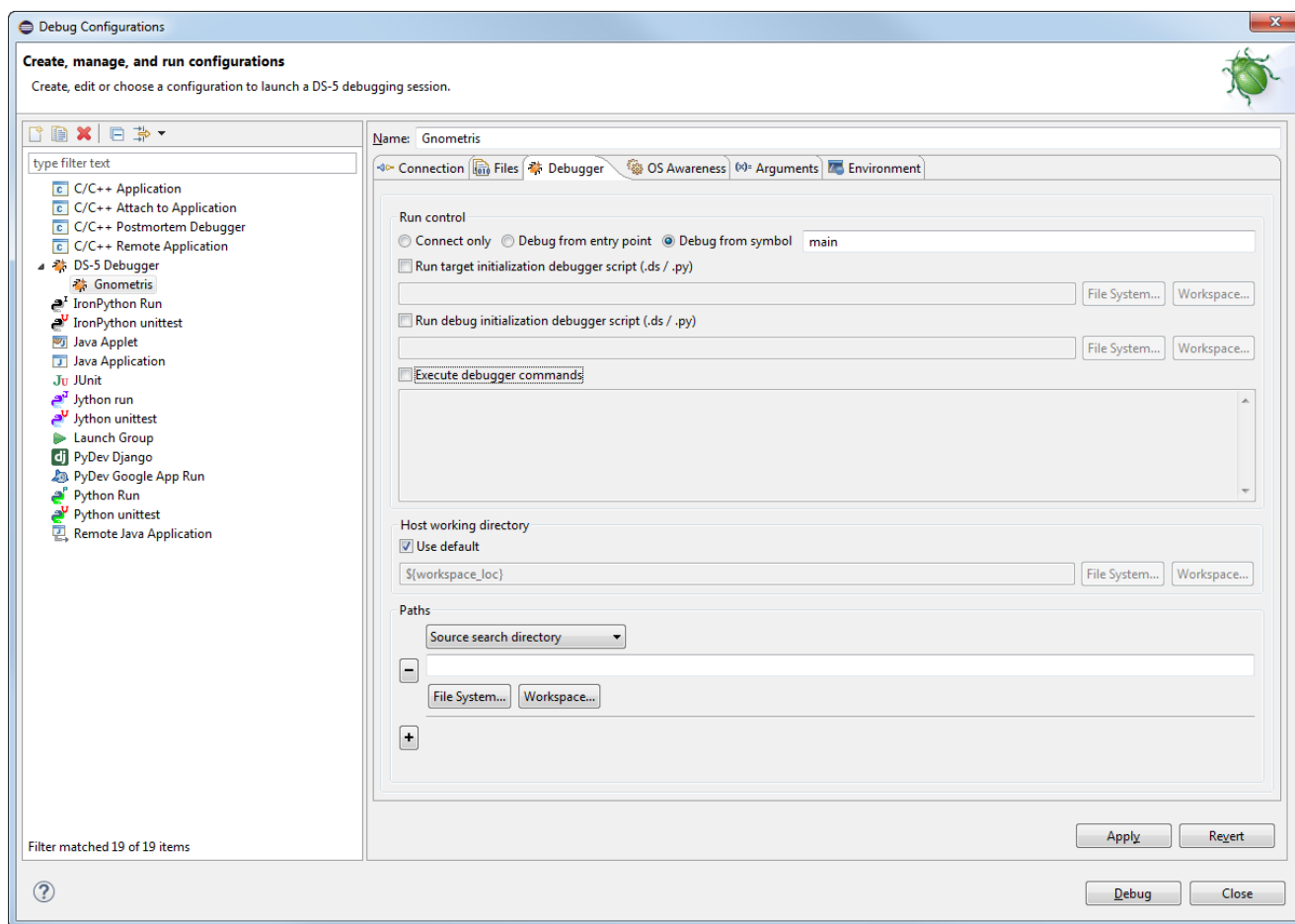


Figure 3-10 Typical file selection for Linux application debug

6. Click on the **Debugger** tab, and:
  - a. In the **Run control** panel, select **Debug from symbol**.
  - b. Enter **main** in the field provided.
7. In the **Host working directory** panel, select **Use default**.



**Figure 3-11 Typical debugger settings for Linux application debug**

8. Click on **Debug** to start the debugger and run to the `main()` function.
9. Debugging requires the DS-5 Debug perspective. If the Confirm Perspective Switch dialog box opens, click **Yes** to switch perspective.

## 3.6 Debugging Gnometris

Debugging the Gnometris application using the example project containing the image binaries and libraries provided with DS-5.

### Procedure

1. Ensure that you are connected to the target, Gnometris is running, and the debugger is waiting at the `main()` function.
2. In the Project Explorer view, open the Gnometris directory to see a list of all the source files.
3. Double-click on the file `blockops-noclutter.cpp` to open the file.
4. In the `blockops-noclutter.c` file, find the line `BlockOps::rotateBlock()`, and double click in the vertical bar on the left-hand side of the C/C++ editor to add a breakpoint. A marker is placed in the vertical bar of the editor and the Breakpoints view updates to display the new information.
5. Click on **Continue** in the Debug Control view to continue running the program.
6. Start a new Gnometris game on the target. When a block arrives, press the up cursor key to hit the breakpoint.
7. Select the Registers view to see the values of the registers.
8. Select the Disassembly view to see the disassembly instructions. You can also double click in the vertical bar on the left-hand side of this view to set breakpoints on individual instructions.
9. In the Debug Control view, click on **Step Over Source Line** to move to the next line in the sourcefile. All the views update as you step through the source code.
10. Select the History view to see a list of all the debugger commands generated during the current debug session. You can select one or more commands and then click on **Exports the selected lines as a script** to create a script file for future use.

## 3.7 Performance analysis of the threads application running on ARM® Linux

ARM Streamline is a graphical performance analysis tool. It captures a wide variety of statistics about code running on the target and uses them to generate analysis reports. You can use these to identify problem areas at system, process, and thread level, in addition to hot spots in the code.

### Prerequisites


This tutorial uses the `threads` example application to show how to use Streamline to capture and analyze profiling data from a Linux target. `threads` is one of the ARMv7 Linux application examples that are provided with DS-5. Before capturing the data, ensure that:

1. You have built the `threads` application.
2. You know the IP address or network name of the target. To find the IP address, you can use the `ifconfig` application in a Linux console. The IP address is denoted by the **inet addr**.
3. The Linux kernel on the target is configured to work with Streamline.
4. The gator daemon, `gator`, is running on the target. If not, the simplest way to install and run `gator` on the target is to use the **Setup Target...** button in the **Connection Browser** dialog in the Streamline Data view.
5. SSH and `gdbserver` are running on the target.

#### Note

- For more information about building and running the `threads` application on a Linux target see the `readme.html` supplied in the same directory as the source code for the example.
- For more information about how to configure your target for Streamline, see the Streamline User Guide.

### Procedure

1. Launch Eclipse for DS-5 and open the DS-5 Debug perspective.
2. In the Remote Systems view, define a connection to the target using the **Define a connection to remote system** button .
3. Launch the Streamline application.
4. Specify the IP address or network name of the target in the **Address** field. Alternatively, use the **Browse for a target** button, as shown in the following screenshot:

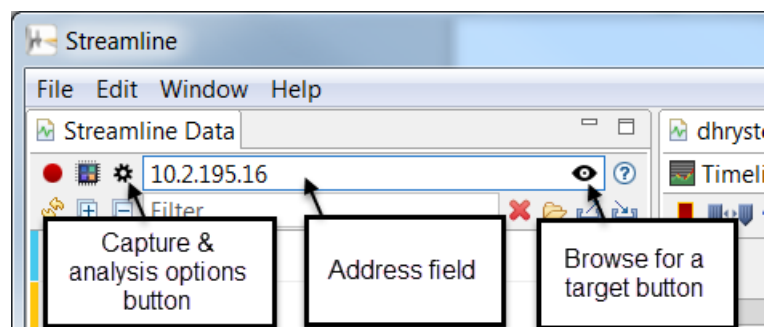




Figure 3-12 Streamline Data view

5. Click the **Capture & analysis options** button. In the Program Images section, select the `threads` image from the workspace, then select **Save**.
6. In Eclipse for DS-5, select **Run > Debug configurations...** then select the `threads-gdbserver-example` debug configuration. This configuration downloads the application to the target, starts `gdbserver` on the target and starts executing the application, stopping at `main()`.
7. Connect to the target either by clicking on **Debug** in the Debug Configurations dialog, or by right clicking on the connection in the Debug Control and selecting **Connect to target**.

8. The program stops at `main()`. To start capturing data, switch to the Streamline application and click the **Start capture** button . Give the capture file a unique name. The Live view opens in Streamline, displaying the capture data in real time.
9. In Eclipse for DS-5, press **Continue** to continue executing the code.
10. When the application terminates, stop the capture in Streamline by clicking the **Stop capture and analyze** button .

Streamline automatically analyzes the capture data and produces a report, which it displays in the Timeline view, as shown in the following screenshot:

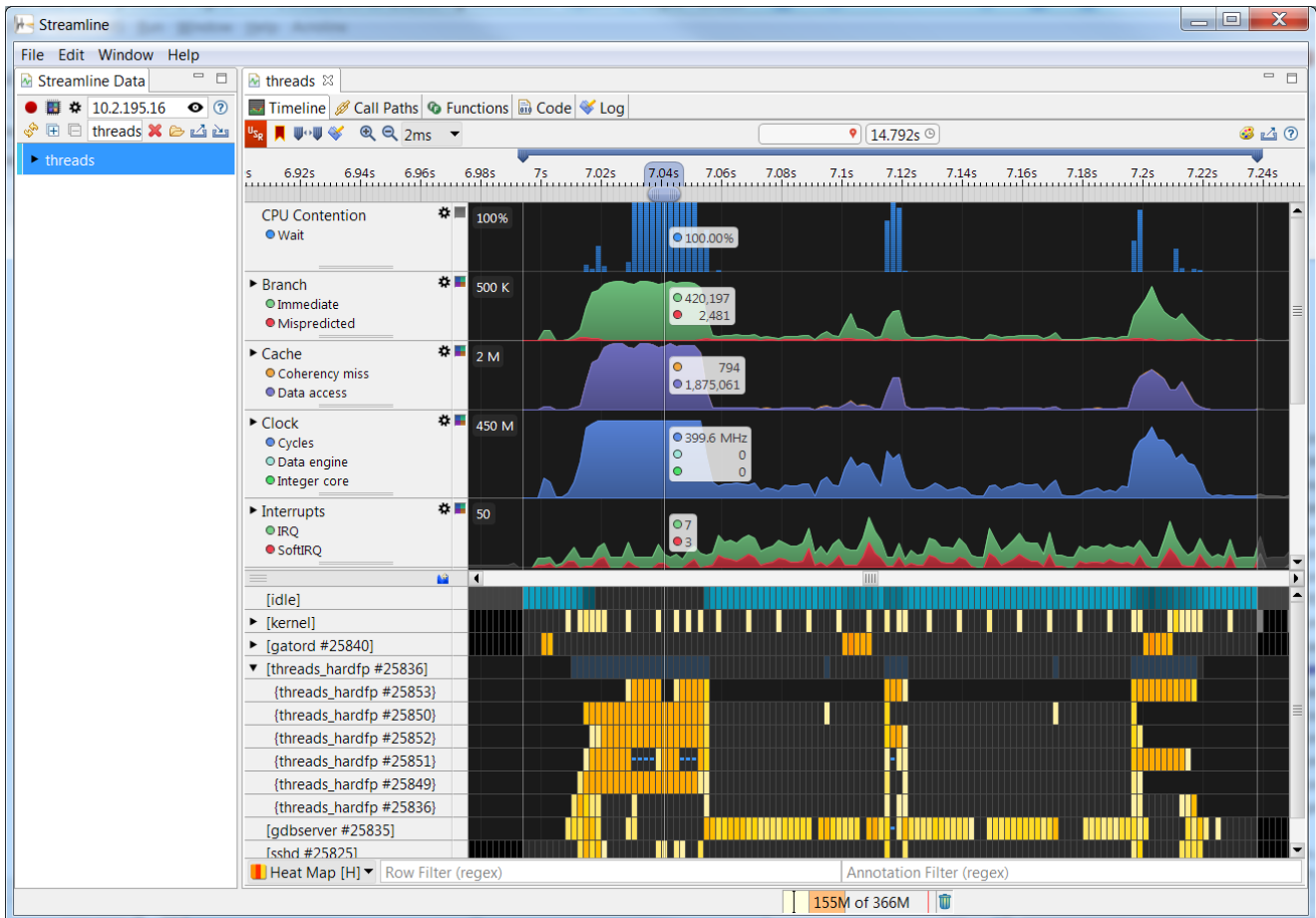


Figure 3-13 Streamline analysis report for the threads application



## 3.8 About registering a new compiler toolchain

If you want to build projects using a toolchain that is not installed with DS-5, you must first register the toolchain you want to use. You can register toolchains:

- Using the **Preferences** dialog in Eclipse for DS-5.
- Using the `add_toolchain` utility from the DS-5 Command Prompt.

You might want to register a compiler toolchain if:

- You upgrade your version of DS-5 but you want to use an earlier version of the toolchain that was previously installed.
- You install a newer version or older version of the toolchain without re-installing DS-5.

When you register a toolchain, the toolchain is be available for new and existing projects in DS-5.

---

**Note**

You can only register ARM or GCC toolchains.

---

## 3.9 Registering a compiler toolchain from the DS-5 command prompt

Use the `add_toolchain` utility from the command prompt to register a new toolchain.

To register a toolchain using the DS-5 command prompt:

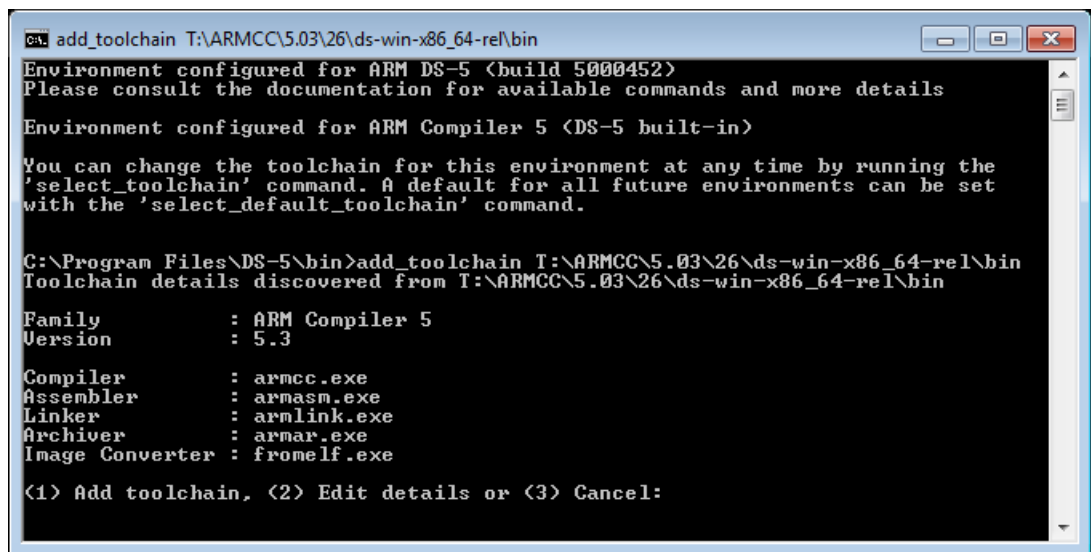
### Procedure

1. Enter `add_toolchain path`, where *path* is the directory containing the toolchain binaries. The utility automatically detects the toolchain properties.

#### ————— Note —————

By default, the `add_toolchain` utility is an interactive tool. To use the `add_toolchain` utility as a non-interactive tool, add the `--non-interactive` option to the command.

For example: `add_toolchain C:\ARMCC\5.03\26\ds-win-x86_64-rel\bin --non-interactive`



```
cmd: add_toolchain T:\ARMCC\5.03\26\ds-win-x86_64-rel\bin
Environment configured for ARM DS-5 <build 5000452>
Please consult the documentation for available commands and more details

Environment configured for ARM Compiler 5 <DS-5 built-in>

You can change the toolchain for this environment at any time by running the
'select_toolchain' command. A default for all future environments can be set
with the 'select_default_toolchain' command.

C:\Program Files\DS-5\bin>add_toolchain T:\ARMCC\5.03\26\ds-win-x86_64-rel\bin
Toolchain details discovered from T:\ARMCC\5.03\26\ds-win-x86_64-rel\bin

Family      : ARM Compiler 5
Version     : 5.3

Compiler    : armcc.exe
Assembler   : armasm.exe
Linker      : armlink.exe
Archiver    : armar.exe
Image Converter : fromelf.exe

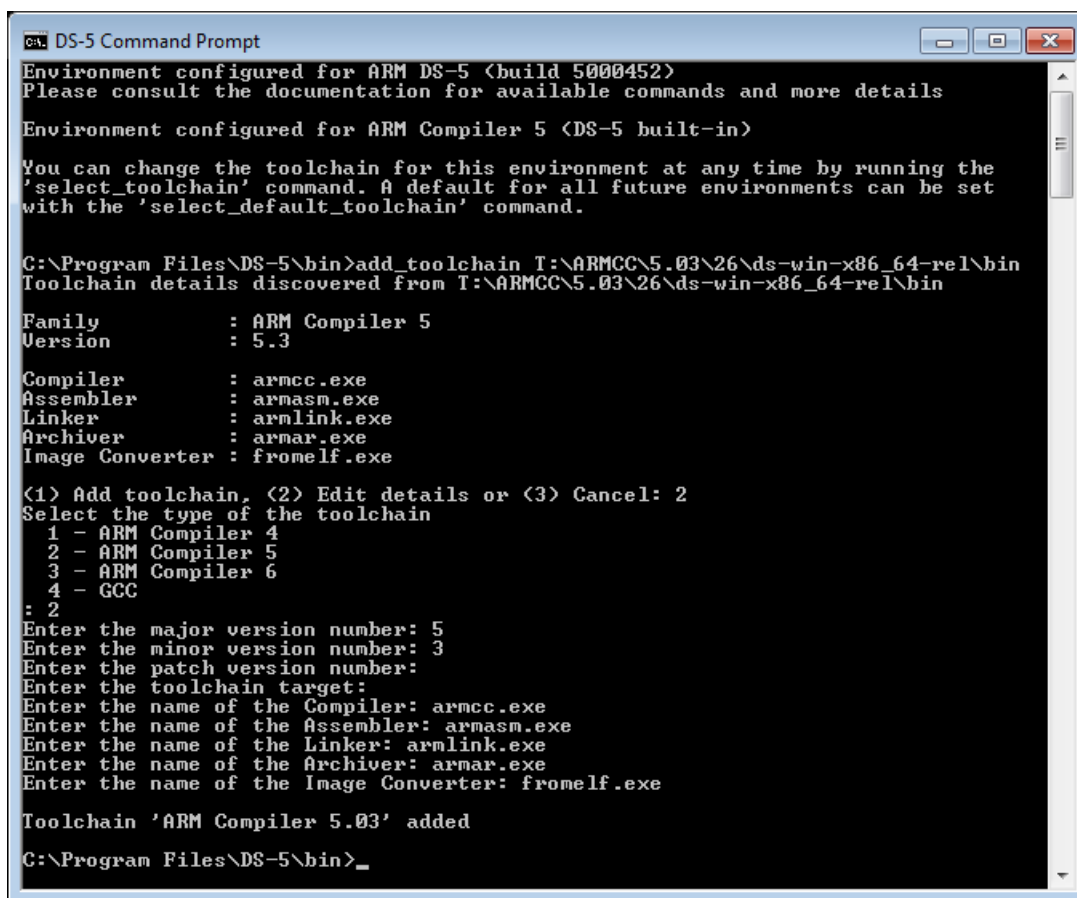
<1> Add toolchain, <2> Edit details or <3> Cancel:
```

Figure 3-14 Registering a new toolchain

2. The utility prompts whether you want to register the toolchain with the details it has detected. If you want to change the details, the utility prompts for the details of the toolchain.

#### ————— Note —————

- The toolchain type must be one of ARM Compiler 4, ARM Compiler 5, ARM Compiler 6, or GCC.
- The toolchain target only applies to GCC toolchains. It indicates what target platform the GCC toolchain builds for. For example, if your compiler toolchain binary is named `arm-linux-gnueabi-hf-gcc`, then the target name is the prefix `arm-linux-gnueabi-hf`. The target field allows DS-5 to distinguish different toolchains that otherwise have the same version.



```
DS-5 Command Prompt
Environment configured for ARM DS-5 (build 5000452)
Please consult the documentation for available commands and more details

Environment configured for ARM Compiler 5 (DS-5 built-in)

You can change the toolchain for this environment at any time by running the
'select_toolchain' command. A default for all future environments can be set
with the 'select_default_toolchain' command.

C:\Program Files\DS-5\bin>add_toolchain T:\ARMCC\5.03\26\ds-win-x86_64-rel\bin
Toolchain details discovered from T:\ARMCC\5.03\26\ds-win-x86_64-rel\bin

Family           : ARM Compiler 5
Version          : 5.3

Compiler         : armcc.exe
Assembler        : armasm.exe
Linker           : armlink.exe
Archiver         : armar.exe
Image Converter  : fromelf.exe

<1> Add toolchain, <2> Edit details or <3> Cancel: 2
Select the type of the toolchain
  1 - ARM Compiler 4
  2 - ARM Compiler 5
  3 - ARM Compiler 6
  4 - GCC
: 2
Enter the major version number: 5
Enter the minor version number: 3
Enter the patch version number:
Enter the toolchain target:
Enter the name of the Compiler: armcc.exe
Enter the name of the Assembler: armasm.exe
Enter the name of the Linker: armlink.exe
Enter the name of the Archiver: armar.exe
Enter the name of the Image Converter: fromelf.exe

Toolchain 'ARM Compiler 5.03' added
C:\Program Files\DS-5\bin>
```

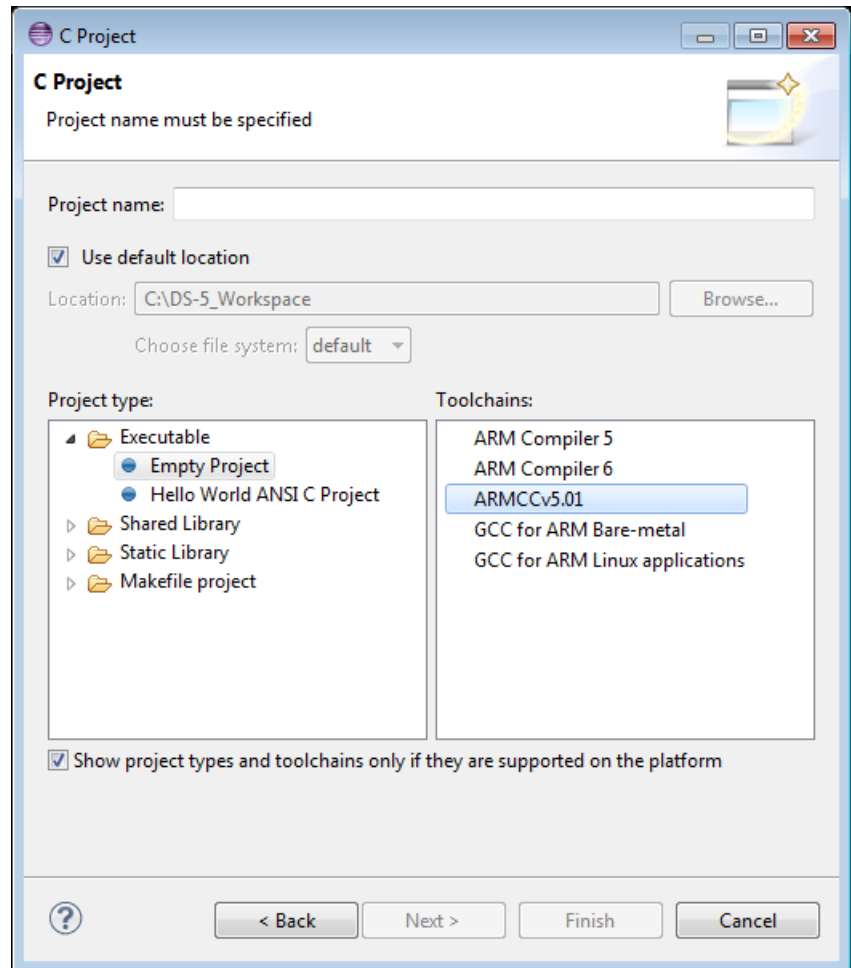
Figure 3-15 Registering a new toolchain

**Note**

You must manually enter the toolchain properties if:

- The toolchain properties were not autodetected.
- The type, major version, and minor version of the new toolchain are identical to a toolchain that DS-5 already knows about.

3. After you register a new toolchain, you must restart DS-5 before you can use the toolchain in the DS-5 environment.
4. When you create a new project, DS-5 shows the new toolchain in the available list of toolchains. In this example, ARMCCv5.01 is the newly registered toolchain.



**Figure 3-16 Using a new toolchain for a new project**

For an existing project, if you want to change the toolchain to the newly registered toolchain, use the **Tool Chain Editor** dialog.

- Right-click the project and select **Properties** to show the **Properties** dialog.
- Select **C/C++ Build > Tool Chain Editor**

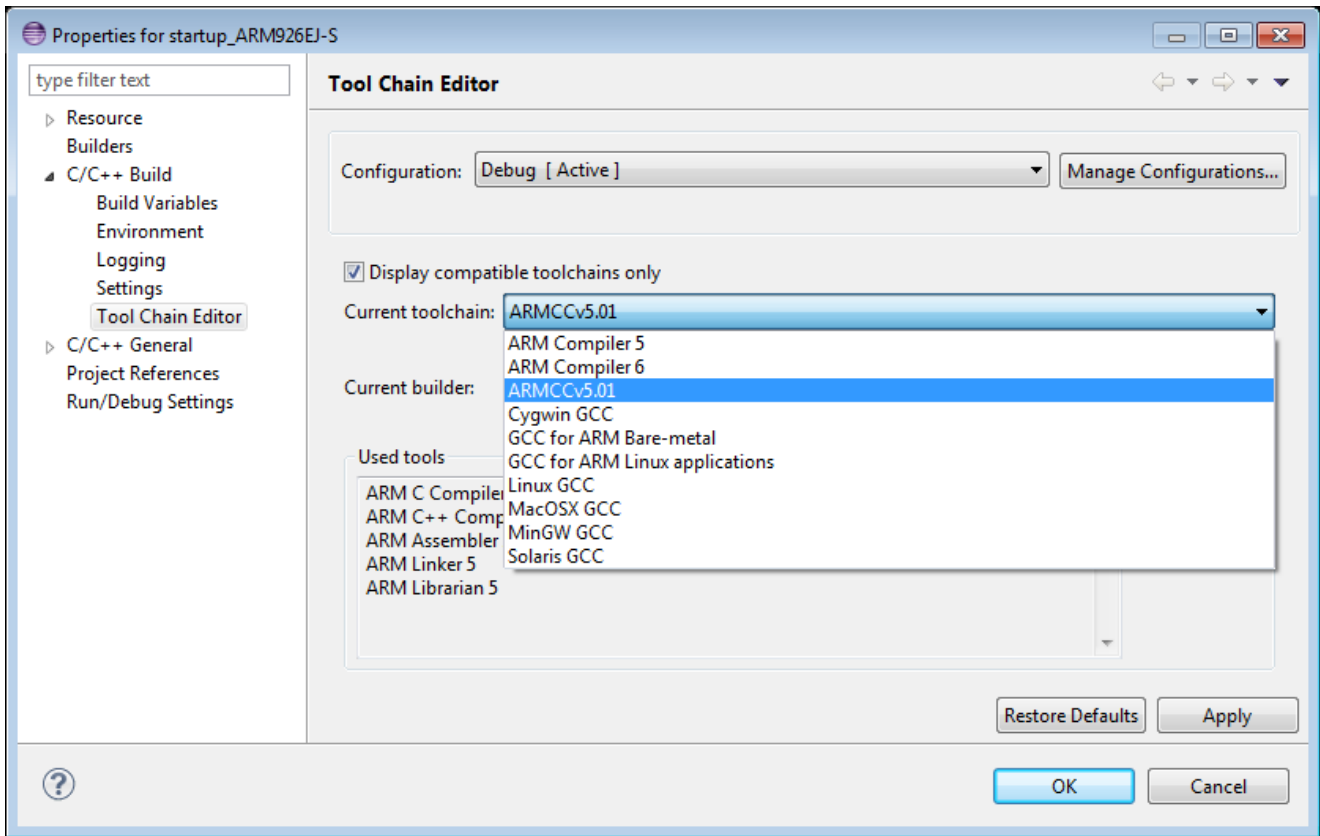


Figure 3-17 Changing the toolchain for a project

### 3.10 Configuring a compiler toolchain for the DS-5 command prompt

When you want to compile or build from the DS-5 command prompt, you can select the compiler toolchain you want to use. You can set this as the default toolchain so that you do not need to select a toolchain every time you start the DS-5 command prompt.

---

**Note**

By default, the DS-5 command prompt is not configured with a compiler toolchain.

---

On Linux, run `suite_exec` with the `--toolchain name` option to configure a compiler toolchain for the DS-5 environment. Run `suite_exec` with no arguments for the list of available toolchains.

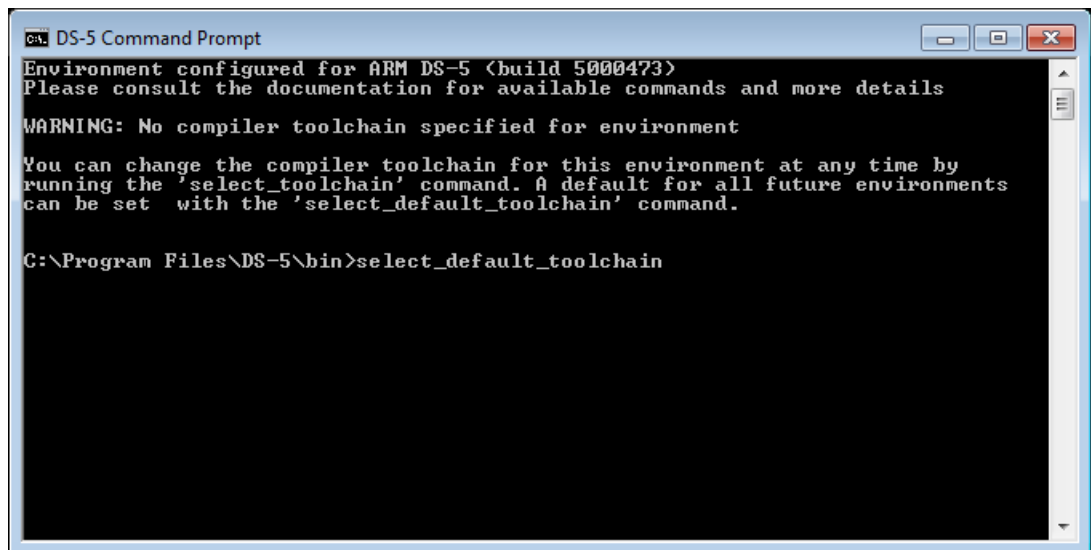
For example, to use the ARM Compiler 5 toolchain included in DS-5, run:

```
DS-5_install_directory/bin/suite_exec --toolchain "ARM Compiler 5 (DS-5 built-in)"  
bash --norc
```

The following procedure describes the steps for using `select_default_toolchain` on Windows.

#### Procedure

1. Open the DS-5 command prompt, by selecting **Start > All Programs > DS-5 Command Prompt**.
2. Enter `select_default_toolchain` on the DS-5 command prompt. This lists the available compiler toolchains.



**Figure 3-18** Configuring a default toolchain

3. Select your default compiler toolchain. For example, enter 1 to select the ARM Compiler 5 that is built-in with DS-5. This configures the DS-5 command prompt for the selected toolchain. When you open a new DS-5 command prompt, the environment is still configured for your selected toolchain.

---

**Note**

To configure a compiler toolchain for the current DS-5 command prompt, without changing the default toolchain, use the `select_toolchain` command.

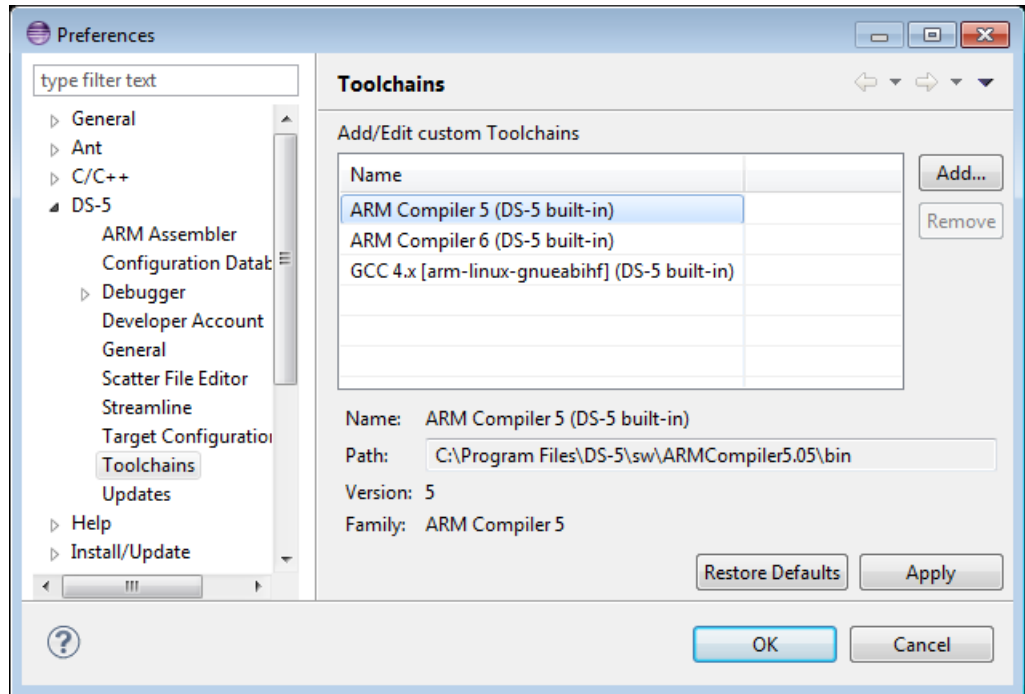
---

## 3.11 Registering a compiler toolchain from Eclipse

You can register compiler toolchains using the **Preferences** dialog in Eclipse for DS-5.

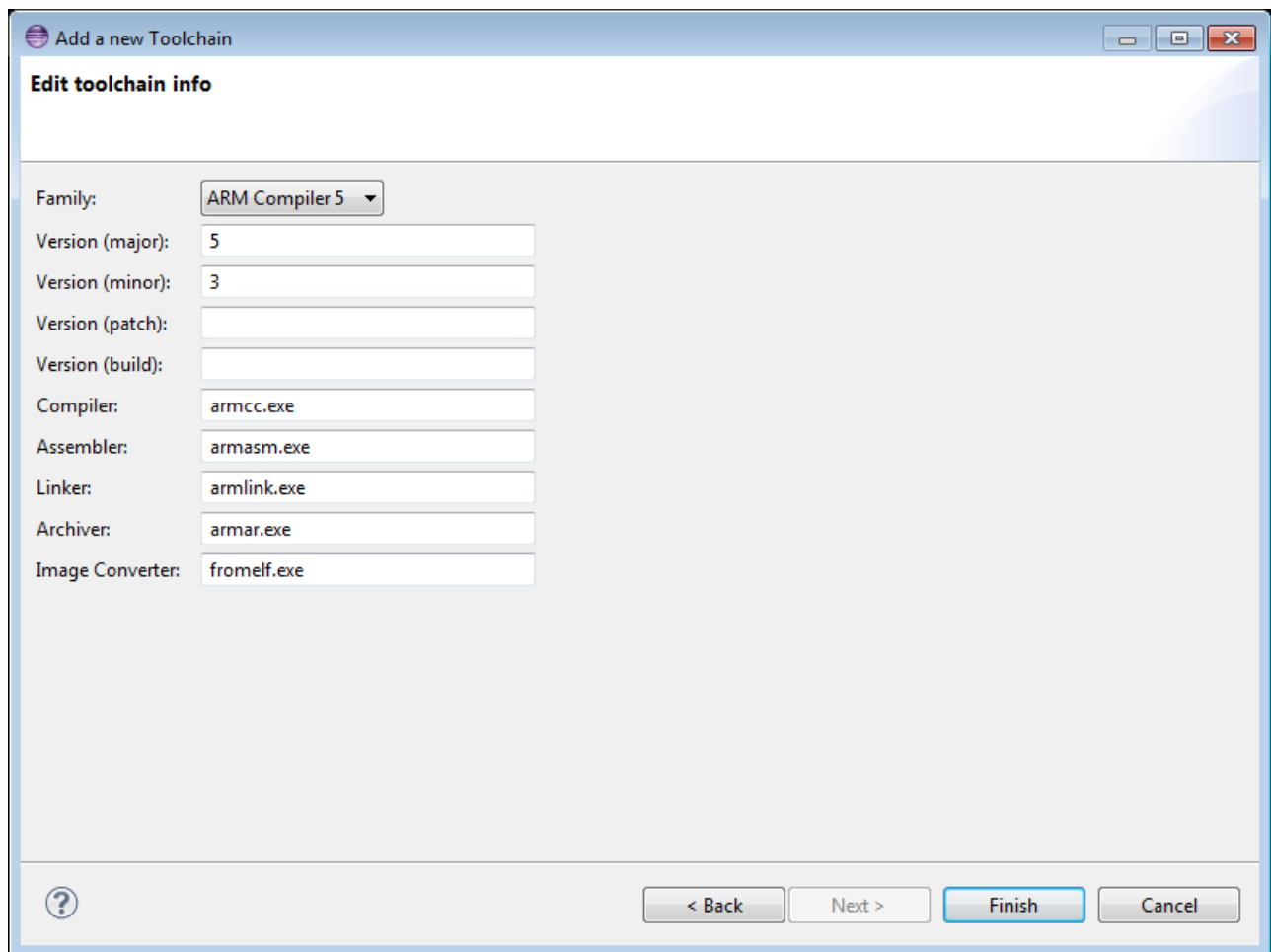
### Procedure

1. To view the compiler toolchains that DS-5 currently knows about, select **Windows > Preferences**. And then select **DS-5 > Toolchains**.



**Figure 3-19 Toolchains Preferences dialog**

2. To add a toolchain, select **Add**. This displays the **Add a new Toolchain** dialog.
3. Enter the path to the toolchain binaries that you want to use. Then click **Next** to autodetect the toolchain properties.
4. When the toolchain properties have been autodetected, you can select **Finish** to register the toolchain. Alternatively, select **Next** to manually enter or change the toolchain properties, and then select **Finish**.



**Figure 3-20** Properties for the new toolchain

————— **Note** —————

You must manually enter the toolchain properties if:

- The toolchain properties were not autodetected.
- The family, major version, and minor version of the new toolchain are identical to a toolchain that DS-5 already knows about.

5. Select **Apply** from the **Toolchains** preferences dialog. The new toolchain has now been registered into DS-5. You must restart DS-5 before you can use the new toolchain in the DS-5 environment.