ARM® DS-5™

Version 5.5

Using Eclipse

ARM DUI 0480E (ID051311)

ARM DS-5

Using Eclipse

Release Information

The following changes have been made to this book.

Change History

Date	Issue	Confidentiality	Change
June 2010	A	Non-Confidential	First release for DS-5
September 2010	B	Non-Confidential	Update for DS-5 version 5.2
November 2010	C	Non-Confidential	Update for DS-5 version 5.3
January 2011	D	Non-Confidential	Update for DS-5 version 5.4
March 2011	E	Non-Confidential	Update for DS-5 version 5.5

Proprietary Notice

Words and logos marked with ® or ™ are registered trademarks or trademarks of ARM in the EU and other countries, except as otherwise stated below in this proprietary notice. Other brands and names mentioned herein may be the trademarks of their respective owners.

Neither the whole nor any part of the information contained in, or the product described in, this document may be adapted or reproduced in any material form except with the prior written permission of the copyright holder.

The product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by ARM in good faith. However, all warranties implied or expressed, including but not limited to implied warranties of merchantability, or fitness for purpose, are excluded.

This document is intended only to assist the reader in the use of the product. ARM shall not be liable for any loss or damage arising from the use of any information in this document, or any error or omission in such information, or any incorrect use of the product.

Where the term ARM is used it means “ARM or any of its subsidiaries as appropriate”.

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.

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 Using Eclipse

Chapter 1
Conventions and feedback

The following describes the typographical conventions and how to give feedback:

Typographical conventions

The following typographical conventions are used:

`monospace` Denotes text that can be entered at the keyboard, such as commands, file and program names, and source code.

`monospace` Denotes a permitted abbreviation for a command or option. The underlined text can be entered instead of the full command or option name.

`monospace italic`

Denotes arguments to commands and functions where the argument is to be replaced by a specific value.

`monospace bold`

Denotes language keywords when used outside example code.

italic Highlights important notes, introduces special terminology, denotes internal cross-references, and citations.

bold Highlights interface elements, such as menu names. Also used for emphasis in descriptive lists, where appropriate, and for ARM® processor signal names.

Feedback on this product

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

- your name and company

- the serial number of the product
- details of the release you are using
- details of the platform you are using, such as the hardware platform, operating system type and version
- a small standalone sample of code that reproduces the problem
- a clear explanation of what you expected to happen, and what actually happened
- the commands you used, including any command-line options
- sample output illustrating the problem
- the version string of the tools, including the version number and build numbers.

Feedback on content

If you have comments on content then send an e-mail to `errata@arm.com`. Give:

- the title
- the number, ARM DUI 0480E
- if viewing online, the topic names to which your comments apply
- if viewing a PDF version of a document, the page numbers to which your comments apply
- a concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

ARM periodically provides updates and corrections to its documentation on the ARM Information Center, together with knowledge articles and *Frequently Asked Questions* (FAQs).

Other information

- *ARM Information Center*, `http://infocenter.arm.com/help/index.jsp`
- *ARM Technical Support Knowledge Articles*, http://infocenter.arm.com/help/topic/com.arm.doc.faqs
- *Support and Maintenance*, http://www.arm.com/support/services/support-maintenance.php
- *ARM Glossary*, http://infocenter.arm.com/help/topic/com.arm.doc.aeg0014-/index.html.

Chapter 2
Introduction

The following topics give an overview of Eclipse, its main features and installation requirements:

Concepts

- *About Eclipse* on page 2-2
- *Integrating ARM plug-ins into a custom Eclipse environment* on page 2-3.

2.1 About Eclipse

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 also combines the GNU toolchain for ARM Linux targets. You can use it as a project manager to create, build, debug, monitor, and manage projects for ARM targets. It uses a single folder called a workspace to store files and folders related to specific projects.

2.1.1 See also

Reference

- Chapter 3 *Getting started with Eclipse*.

Other information

- *EclipseFoundation*, `http://eclipse.org`
- *GCC, the GNU Compiler Collection*, `http://gcc.gnu.org`.

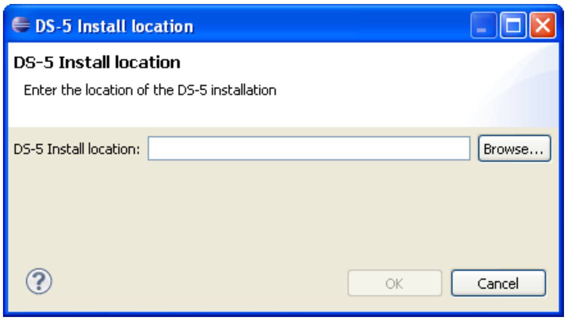
2.2 Integrating ARM plug-ins into a custom Eclipse environment

The following components are required to use Eclipse for DS-5. If you install DS-5, all the required components are installed for you including the Eclipse IDE.

If you have a custom installation of the Eclipse IDE then you must ensure that you have Eclipse v3.6.

Install the following components in the following order before using Eclipse for DS-5:

1. Download and install J2SE 6.0 or later from Sun Microsystems. You must use a 32-bit version of the JRE even when running on a 64-bit operating system.
2. The *C and C++ Development Tools* (CDT) is a set of plug-ins that integrate the C and C++ build environment into the IDE. You can install these plug-ins as part of the Eclipse+CDT bundle from the Eclipse website. CDT v7.0 is required.
3. The *Remote System Explorer* (RSE) is a plug-in that enables you to use remote log-in and file transfer facilities within the IDE. RSE v3.2.1 is required.
4. Install DS-5 in a separate location from your custom Eclipse installation. Search for the folder `install_directory\sw\eclipse` and take a copy of the `dropins` folder then place it under the `eclipse` folder in your custom installation.
5. If you want the custom installation of Eclipse to integrate with the GNU toolchain, you must install the latest GNU tools specific for ARM targets and then add the `.../bin` directory to the PATH environment variable.
6. If the path to the DS-5 installation is not configured, the DS-5 Install location dialog box opens to enable you to locate this folder. Alternatively, you can configure this later by changing the DS-5 settings in the **Window** → **Preferences** dialog box.



DS-5 Install location

DS-5 Install location

Enter the location of the DS-5 installation

DS-5 Install location:

Figure 2-1 DS-5 Install location dialog box

2.2.1 See also

Tasks

- *Launching the debugger from Eclipse* on page 2-6.

Reference

- *Preferences dialog box* on page 3-28.

Other information

- *Java from Sun Microsystems*, `http://www.java.com`
- *Eclipse*, `http://www.eclipse.org`
- *Sourcery G++ for ARM GNU/Linux*, `http://www.codesourcery.com`.

Chapter 3

Getting started with Eclipse

The following topics describe how to get started with Eclipse, the C/C++ perspective, the DS-5 Debug perspective, and associated features:

Tasks

Concepts

Reference

- *Toolbars* on page 3-24
- *Editing source code* on page 3-25
- *Configuring Eclipse* on page 3-27
- *Preferences dialog box* on page 3-28
- *Properties dialog box* on page 3-30
- *Importing and exporting options* on page 3-32
- *Restrictions of use* on page 3-39.

3.1 Launching Eclipse

To launch Eclipse:

- On Windows, select **Start** → **All Programs** → **ARM DS-5** → **Eclipse for DS-5**.
- On Linux:
  1. Add `install_directory/bin` to your `PATH` environment variable. If it is already configured then you can skip this step.
  2. Open Unix bash shell.
  3. Enter `eclipse` at the prompt.

3.1.1 Language packs

Some features of the workbench are translated into different languages.You can download language packs from the Eclipse website when available.

There are two ways to launch the workbench with a different language pack:

- If your operating system is running in the language that you want to use then the workbench automatically displays the translated features.
- If your operating system is not running in the language that you want to use then you must specify the `-nl` command-line argument when launching the workbench.

  For example, to use the Japanese language pack you can use:

  ```
  eclipse -nl ja
  ```

3.1.2 See also

Concepts

- *Using the welcome screen* on page 3-4
- *Overview of the workbench window* on page 3-6.

Reference

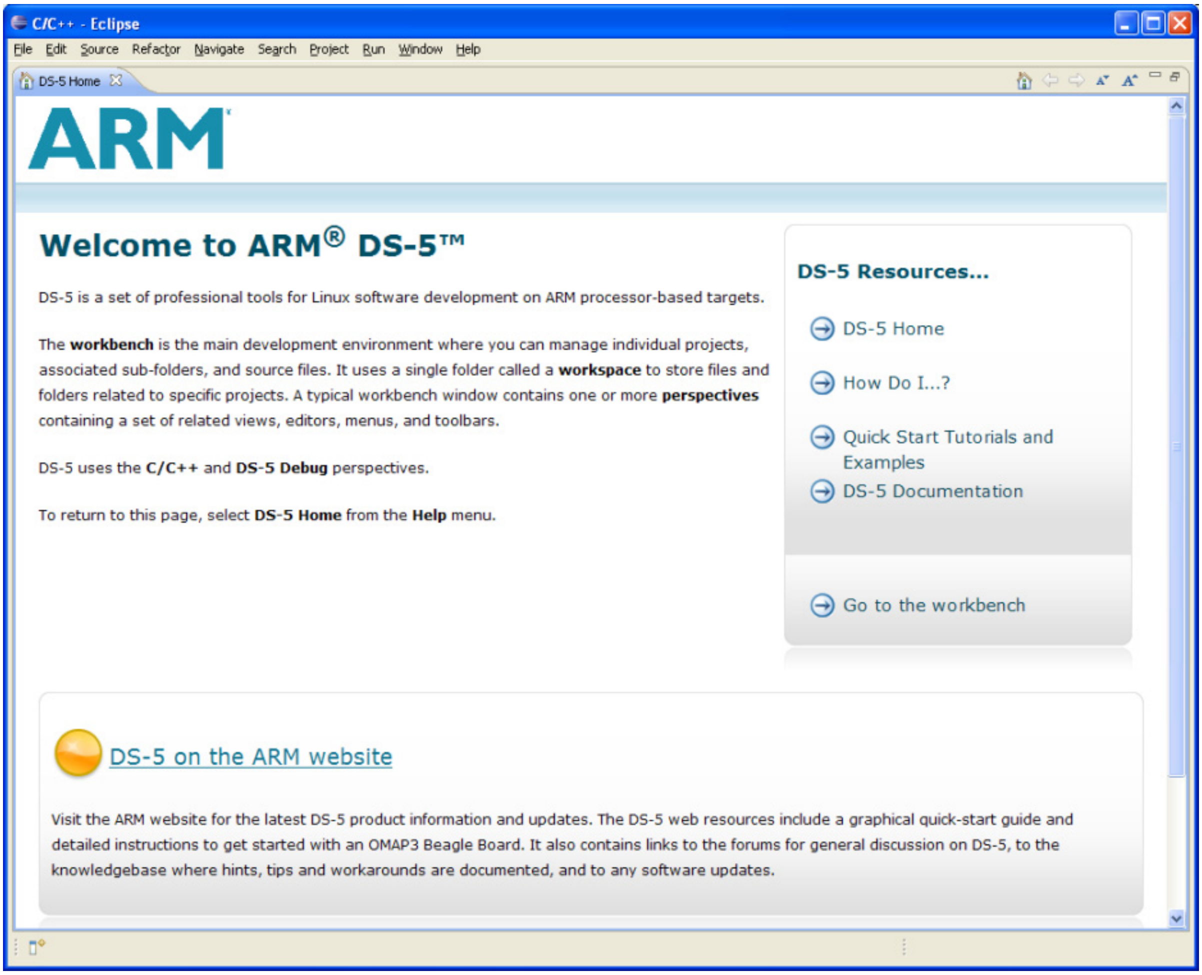
- *Workbench features* on page 3-7
- *Workspace* on page 3-8
- *Importing and exporting options* on page 3-32.

3.2 Using the welcome screen

When Eclipse launches for the first time, the Workspace Launcher dialog box opens enabling you to select your workspace, accept the default settings and click **OK**. You are then presented with the welcome screen containing an overview of the main development environment and links to other pages to help you get started.

Note

Some of the links require an active internet connection.


C/C++ - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
DS-5 Home
ARM

Welcome to ARM® DS-5™

DS-5 is a set of professional tools for Linux software development on ARM processor-based targets.

The **workbench** is the main development environment where you can manage individual projects, associated sub-folders, and source files. It uses a single folder called a **workspace** to store files and folders related to specific projects. A typical workbench window contains one or more **perspectives** containing a set of related views, editors, menus, and toolbars.

DS-5 uses the **C/C++** and **DS-5 Debug** perspectives.

To return to this page, select **DS-5 Home** from the **Help** menu.

DS-5 Resources...

- DS-5 Home
- How Do I...?
- Quick Start Tutorials and Examples
- DS-5 Documentation
- Go to the workbench

DS-5 on the ARM website

Visit the ARM website for the latest DS-5 product information and updates. The DS-5 web resources include a graphical quick-start guide and detailed instructions to get started with an OMAP3 Beagle Board. It also contains links to the forums for general discussion on DS-5, to the knowledgebase where hints, tips and workarounds are documented, and to any software updates.

Figure 3-1 Welcome screen

To access the main development environment click on the close icon in the **DS-5 Home** tab or click on the **Go to the Workbench** link on the welcome page. You can return to the welcome screen at any time by selecting **DS-5 Home** from the **Help** menu.

3.2.1 See also

Concepts

- *Overview of the workbench window* on page 3-6.

Reference

- *Workbench features* on page 3-7
- *Workspace* on page 3-8

- *Importing and exporting options* on page 3-32.

3.3 Overview of the workbench window

A typical workbench window contains a collections of views that are associated with a specific perspective. For example the Project Explorer view, C/C++ editor, Progress view, Console view, and Problems view are all associated with the C/C++ perspective. Each perspective automatically adds the most useful views when you select it.

DS-5 uses the C/C++ and DS-5 Debug perspectives.

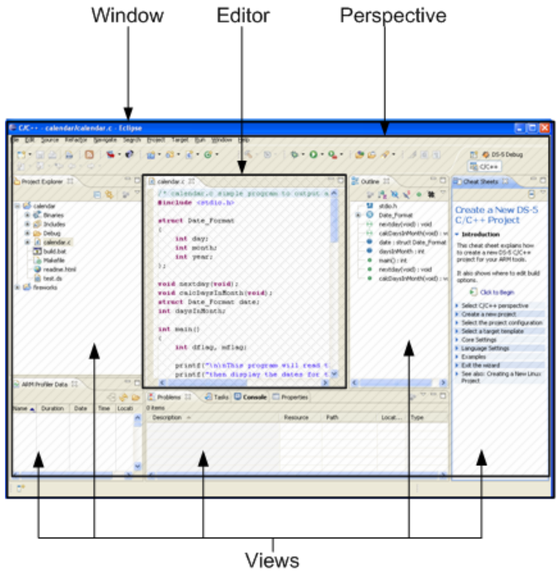


Figure 3-2 Typical workbench window

To close the workbench window and exit Eclipse, select **Exit** from the **File** menu or click on the close icon in the top corner of the window. On exit, Eclipse saves your settings automatically so that when you next open it, the window returns to the same perspectives and views.

3.3.1 See also

Tasks

- *Using the welcome screen* on page 3-4.

Reference

- *Workbench features* on page 3-7
- *Workspace* on page 3-8
- *Importing and exporting options* on page 3-32.

3.4 Workbench features

The workbench is the main development environment where you can manage individual projects, associated sub-folders, and source files. Each workbench window is linked to one workspace. If you want to use different workspaces at the same time, you can launch several workbench windows and link each one to a different workspace.

This section describes the main workbench features:

Editors Editors enable you to view and modify the content of a file, for example source files. The tabs in the editor area show files that are currently open for editing.

Menus and Toolbars

The main menu and toolbar are located at the top of the Eclipse window. Other toolbars associated with specific features are located at the top of each perspective or view.

Perspectives Perspectives define the layout of your selected views and editors in Eclipse. They also have their own associated menus and toolbars.

Resources Resources are projects, files, and folders that exist in your workbench.

Views Views provide related information corresponding to the active file in the editor. They also have their own associated menus and toolbars.

Workspace Workspace is an area designated on your file system to store files and folders related to your workbench projects and also your personal workbench settings.

3.4.1 See also

Tasks

- *Using the welcome screen* on page 3-4
- Chapter 5 *Working with editors*.

Concepts

- *Overview of the workbench window* on page 3-6.

Reference

- *Workspace* on page 3-8
- *Resources* on page 3-10
- *Perspectives and views* on page 3-19
- *Menus* on page 3-23
- *Toolbars* on page 3-24.

3.5 Workspace

The workspace is an area designated on your file system to store files and folders related to your workbench projects and also your personal workbench settings.

——— **Note** ———

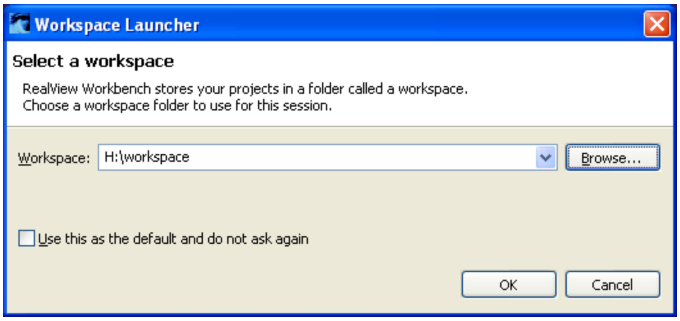
ARM recommends that you select a dedicated workspace folder for your workbench projects only. If you select an existing folder containing resources that are not related to workbench projects, you cannot access them in Eclipse. These resources might also cause a conflict later when you create and build projects.

Changes to the customized settings in the **Window** → **Preferences** dialog box are saved in your workspace. If you select a different workspace then these settings might be different.

———————

When Eclipse launches for the first time, the Workspace Launcher dialog box opens enabling you to select your workspace.

On subsequent launches the last saved workspace is shown as the default selection in the drop-down menu. You can select another workspace by clicking on the down arrow or the **Browse...** button.



Workspace Launcher

Select a workspace

RealView Workbench stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: H:\workspace

□ Use this as the default and do not ask again

Figure 3-3 Workspace launcher dialog box

If required, select the checkbox to disable the Workspace Launcher dialog box from opening on subsequent launches.

——— **Note** ———

You can change the default workspace at any time by selecting **Switch Workspace...** from the **File** menu.

———————

Alternatively, to open Eclipse and automatically link to a specific workspace you can use the `-data` command-line argument. For example:

```
eclipse -data h:\workspace
```

3.5.1 Editing files outside Eclipse

Project sub-folders and files can be edited even when Eclipse is not running. When you next launch Eclipse, the default preferences enable the relevant views to refresh and update. Alternatively if you change the default preferences to disable auto-refresh, you can click on the updated sub-folder or file in the Project Explorer view and select **Refresh** from the **File** menu.

3.5.2 See also

Tasks

- *Using the welcome screen* on page 3-4.
- Chapter 5 *Working with editors*.

Reference

- *Overview of the workbench window* on page 3-6
- *Workbench features* on page 3-7
- *Resources* on page 3-10
- *Perspectives and views* on page 3-19
- *Menus* on page 3-23
- *Toolbars* on page 3-24.

3.6 Resources

A resource is a generic term used to describe a project, file, folder or a combination of these. Resources exist in Eclipse but might not always exist in the workspace. There are three types of resources:

Project A project is displayed in the Project Explorer view and can be stored within the workspace folder or can be a linked resource.

A project must exist in Eclipse before other resources can be imported or linked to that project. The project creation process creates additional configuration files and folders, for example, build properties. These additional files and folders must not be edited or deleted.

Folder A folder is displayed in the Project Explorer view and can be located within the project folder or can be a linked resource.

File A file is displayed in the Project Explorer view and can be located within the project folder or can be a linked resource.

3.6.1 Linked resources

Resources can be shared between projects or they can exist in the file system outside of your selected workspace. To do this a link must be created within Eclipse.

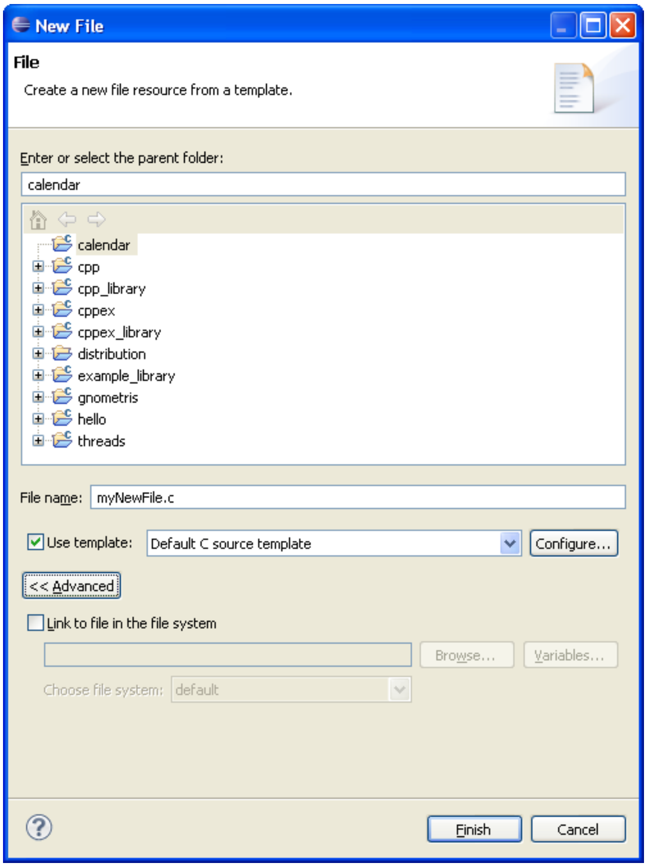
Note

A linked file or folder must have a project as its parent resource.

Deleting, moving or copying a linked resource only affects the link in your workspace and not the resource that it links to. However, deleting a child resource from within a linked folder also deletes it from the file system!

Linked file

To link an existing file to a project in your workspace instead of copying it, you can use the advanced options of the New File wizard. By default the advanced options are not visible, click on the **<<Advanced** button to reveal them. A path variable can also be used to reference a file. For more information on referencing with path variables, use the dynamic help.



New File

File

Create a new file resource from a template.

Enter or select the parent folder: calendar

- calendar
- cpp
- cpp_library
- cppex
- cppex_library
- distribution
- example_library
- gnometris
- hello
- threads

File name: myNewFile.c

☑ Use template: Default C source template

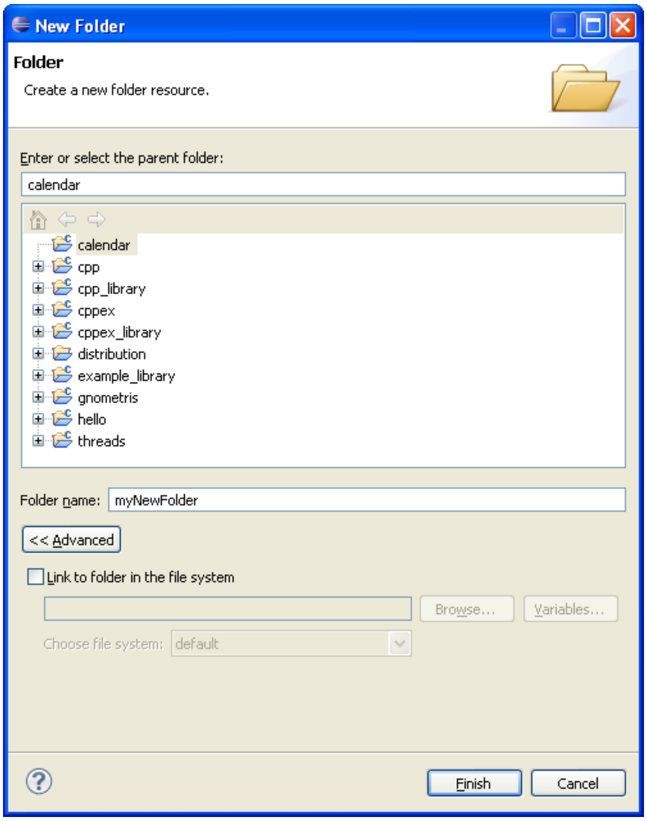
☐ Link to file in the file system

Choose file system: default

Figure 3-4 Linked file

Linked folder

To link an existing folder to a project in your workspace instead of copying it, you can use the advanced options of the New Folder wizard. By default the advanced options are not visible, click on the **<<Advanced** button to reveal them. A path variable can also be used to reference a file. For more information on referencing with path variables, use the dynamic help.



New Folder

Folder

Create a new folder resource.

Enter or select the parent folder: calendar

- calendar
- cpp
- cpp_library
- cppex
- cppex_library
- distribution
- example_library
- gnometris
- hello
- threads

Folder name: myNewFolder

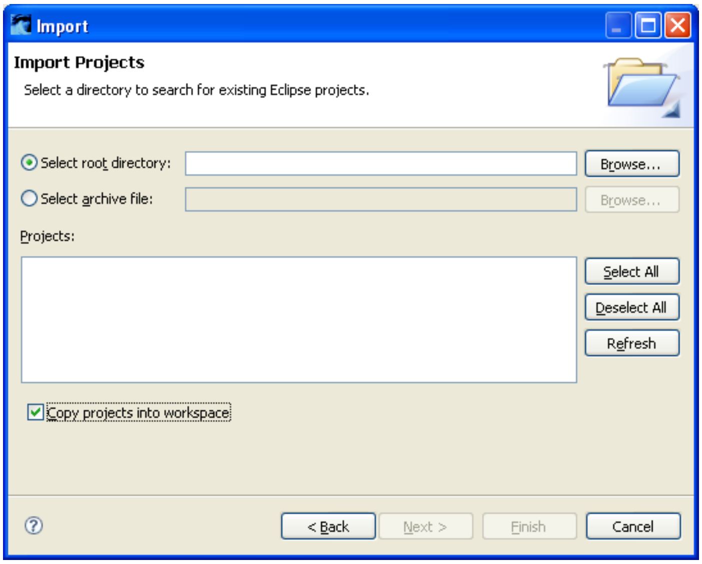
☐ Link to folder in the file system

Choose file system: default

Figure 3-5 Linked folder

Linked project

Eclipse uses the Import wizard to create a link to an existing project. This can be useful if you have a central folder with shared projects. By default, the **Copy projects into workspace** checkbox is selected to ensure that a copy of your project is placed in your current workspace. If you want to leave your project where it is and create a link to it, you must deselect this option.



Import

Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory:

○ Select archive file:

Projects:

☑ Copy projects into workspace

Figure 3-6 Linked project

Disabling the use of linked resources

You can disable the use of linked resources by changing the **General** → **Workspace** → **Linked Resources** settings in the Preferences dialog box.

3.6.2 See also

Tasks

- *Accessing the dynamic help* on page 3-35.

Reference

- *Perspectives and views* on page 3-19
- *Menus* on page 3-23
- *Toolbars* on page 3-24
- *Preferences dialog box* on page 3-28
- *Adding a new source file to your project* on page 4-13.

3.7 About working sets

The Project Explorer view usually displays a full list of all your projects associated with the current workspace. If you have a lot of projects it can be difficult to navigate through the list to find the project that you want to use. Creating a working set enables you to group projects together and display a smaller subset of projects. One or more working sets can be selected at the same time or you can use the view menu to switch between one set and another. To return to the original view you can deselect working sets in the view menu.

Working sets are also useful to refine the scope of a search or build projects in a specific working set.

3.7.1 See also

Tasks

- *Creating a working set* on page 3-15
- *Changing the top level element when displaying working sets* on page 3-17
- *Deselecting a working set* on page 3-18.

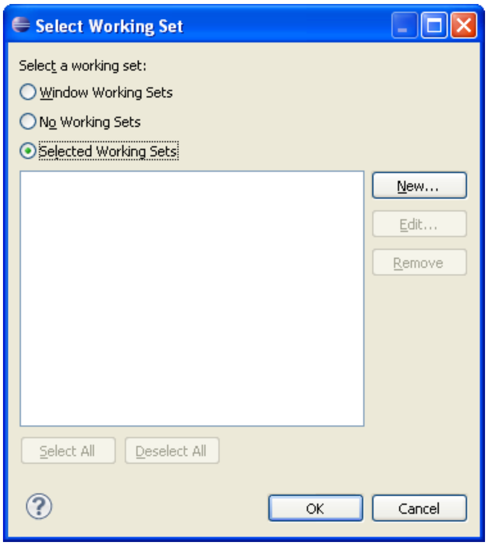
Reference

- *Perspectives and views* on page 3-19
- *Menus* on page 3-23
- *Toolbars* on page 3-24.

3.8 Creating a working set

To create a working set for related projects:

1. Click on the **View Menu** icon in the Project Explorer view toolbar.
2. Select **Select Working Set...** from the context menu.
3. Click **New...**.



Select a working set:

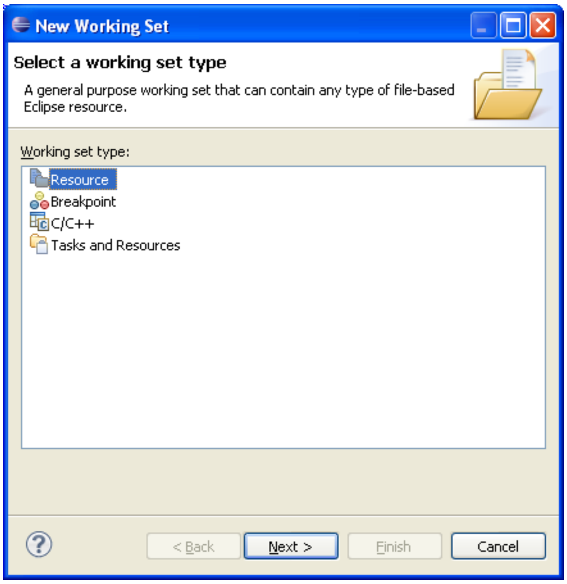
○ Window Working Sets

○ No Working Sets

● Selected Working Sets

Figure 3-7 Creating a new working set

4. In the working set type, select **Resource** to group related projects.
5. Click **Next**.



Select a working set type

A general purpose working set that can contain any type of file-based Eclipse resource.

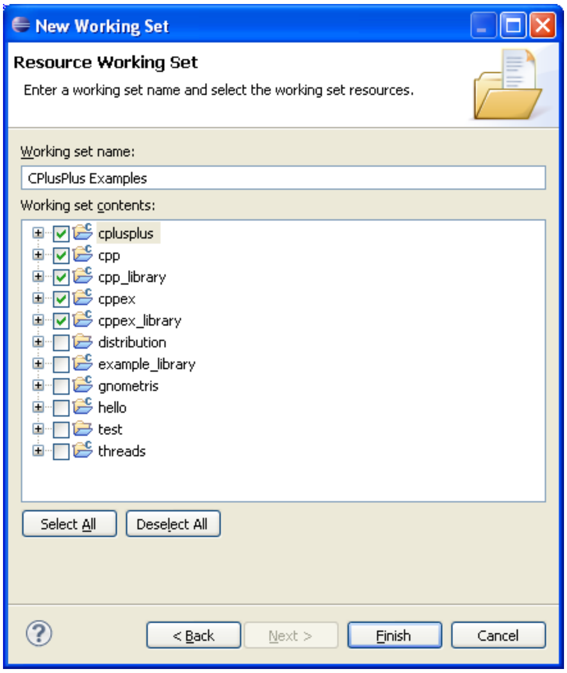
Working set type:

- Resource
- Breakpoint
- C/C++
- Tasks and Resources

Figure 3-8 Selecting the resource type for the new working set

6. In the Working set name field, enter a suitable name.
7. In the Working set contents panel, you can select existing projects that you want to associated with this working or you can return to the wizard later to add projects.

8. Click **Finish**.



New Working Set

Resource Working Set

Enter a working set name and select the working set resources.

Working set name:

CPlusPlus Examples

Working set contents:

- ☑ cplusplus
- ☑ cpp
- ☑ cpp_library
- ☑ cppex
- ☑ cppex_library
- □ distribution
- □ example_library
- □ gnometris
- □ hello
- □ test
- □ threads

Figure 3-9 Adding new resources to a working set

9. If required, repeat these steps to create more working sets.

10. In the Select Working Set dialog box, select the working sets that you want to display in the Project Explorer view.

11. Click **OK**.

The filtered list of projects are displayed in the Project Explorer view. Another feature of working sets that can help with navigation is the option to change the top level element in the Project Explorer view.

3.8.1 See also

Tasks

- *Changing the top level element when displaying working sets* on page 3-17
- *Deselecting a working set* on page 3-18.

Concepts

- *About working sets* on page 3-14.

Reference

- *Perspectives and views* on page 3-19
- *Menus* on page 3-23
- *Toolbars* on page 3-24.

3.9 Changing the top level element when displaying working sets

In the Project Explorer view, if you have more than one working set then you might want to display the projects in a hierarchical tree with the working set names as the top level element. This is not selected by default.

To change the top level element in the Project Explorer view:

1. Click on the **View Menu** icon in the Project Explorer view toolbar.
2. Select **Top Level Elements** from the context menu.
3. Select either **Projects** or **Working Sets**.

3.9.1 See also

Tasks

- *Creating a working set* on page 3-15
- *Deselecting a working set* on page 3-18.

Concepts

- *About working sets* on page 3-14.

Reference

- *Perspectives and views* on page 3-19
- *Menus* on page 3-23
- *Toolbars* on page 3-24.

3.10 Deselecting a working set

To change the display of projects in the Project Exlorer view and return to the full listing of all the projects in the workspace:

1. Click on the **View Menu** icon in the Project Explorer view toolbar.
2. Select **Deselect Working Set** from the context menu.

3.10.1 See also

Tasks

- *Creating a working set* on page 3-15
- *Changing the top level element when displaying working sets* on page 3-17.

Concepts

- *About working sets* on page 3-14.

Reference

- *Perspectives and views* on page 3-19
- *Menus* on page 3-23
- *Toolbars* on page 3-24.

3.11 Perspectives and views

The main workbench window contains one or more perspectives and each perspective contains one or more views.

3.11.1 Perspectives

Perspectives can be opened and customized using the options from the **Window** menu or you can use the perspective toolbar. By default a perspective opens in the same window, however, you can change the default settings in the Preferences dialog box.

Eclipse for DS-5 focuses on the C/C++ perspective and the DS-5 Debug perspective. You can change perspective by using the perspective toolbar or you can select **Window → Open perspective** from the main menu.

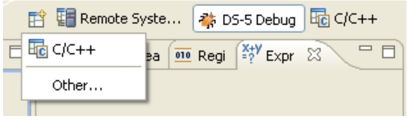


Figure 3-10 Changing perspective using the toolbar menu

3.11.2 Views

A view is a small visual component within Eclipse to navigate through resources or to display specific properties. An editor is a special type of view that enables viewing and editing of source files.

Views can be moved or docked as applicable by dragging and dropping them into position. Double-clicking on a view tab toggles the maximize/restore options for that view or you can use the relevant toolbar icons. To reset all the views in a perspective to the initial settings and layout you can click on **Reset Perspective...** in the **Window** menu.

The following views are useful when building and debugging C/C++ applications:

Editors Editors are associated with specific file types and open the related editor view automatically when you open an editable file from the Project Explorer view.

General views

Console Displays output messages when building projects and also execution messages when running or debugging an application.

Outline Displays a structured list of C/C++ elements in the active file. Clicking on an element changes the editor focus to the position of that element in the active file.

Problems Displays error messages encountered during a build. Selecting an error message opens the associated file and moves the focus to the line causing the problem.

Progress Displays the progress meter when a task is in progress.

Project Explorer Provides a hierarchical view of resources relevant to C/C++ project files. Right-clicking on a resource produces a context menu for specific tasks.

Properties Displays names and values for the selected item. For example, the last modified time/date for a file.

Help views

Help Contents Displays the installed documentation.

Dynamic Help Displays dynamic help for the selected feature.

Cheat Sheets Displays step by step instructions to help you complete a specific task. Use the view menu to access the list of cheat sheets.

DS-5 Debugger views

App Console Enables you to interact with the I/O capabilities provided by the semihosting implementation in the ARM® C libraries.

ARM Asm Info Enables you to view more information on an ARM® or Thumb® instruction or directive.

Breakpoints Displays all the breakpoints and the repsective locations in source code or target memory. Using this view you can add, edit, delete, enable or disable breakpoints.

Commands Provides access to the scripting functionality of the debugger. Most debugger actions are echoed here and you can also execute DS-5 Debugger commands.

Debug Control Contains a tree hierarchy showing debug connection states, processors, threads and stack frames as applicable.

Disassembly Displays disassembled code.

Expressions Enables you to view and edit expressions.

History Provides a historical log of all the executed debugger commands.

Memory Enables you to view and edit the contents of memory.

Modules Enables you to view shared libraries or loaded *Operating System* (OS) modules.

Registers Enables you to view and edit registers associated with the selected stack frame. When your application stops, modified values are highlighted.

Screen Enables you to display a block of memory as a graphical image. For example, a frame buffer.

Scripts Provides access to run and edit script files.

Target Displays a list of capabilities for the target device that is currently connected to the debugger.

Trace Displays the captured trace information from the debug agent.

Variables Enables you to view and edit variables associated with the selected stack frame.

Remote Systems views

Remote Scratchpad

Enables you to copy and paste or drag and drop local resources to an electronic clipboard for use at a later point in time.

Remote Systems Enables you to create connections and access resources on the host workstation and remote targets.

Terminals Enables you to enter shell commands directly on the target without launching any external application.

Other views can be added to a perspective but they might not be compatible with all perspectives. To add a view to a perspective:

1. Ensure that you are in the relevant perspective.
2. Select **Window** → **Show View** → **Other...** to open the Show View dialog box.
3. Select the required view.
4. Click **OK**.

For more information on the other views not listed here, use the dynamic help.

3.11.3 Trim bar

Mimimized views are placed in the trim bar at the edge of the current perspective. By default the trim bar is on the right-hand margin of the perspective but you can drag and drop it to another location if you prefer. In the trim bar, you can double click on a view to restore it or click on the restore icon to restore all the minimized views.

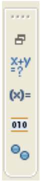


Figure 3-11 Minimized views in the trim area

3.11.4 Fast view bar

Fast views are hidden views that can be accessed quickly from the fast view bar but do not take up space in the current perspective. When you select a fast view it opens in the current perspective but hides again when it loses focus. By default the fast view bar is located at the bottom margin of the current perspective but you can drag and drop it to another location if you prefer.

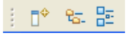


Figure 3-12 Fast view bar

To add a view to the fast view bar then right-click on a view tab and select **Fast View** from the context menu.

3.11.5 See also

Tasks

- *Accessing the dynamic help* on page 3-35
- Chapter 5 *Working with editors*.

Reference

- *Workbench features* on page 3-7
- *Workspace* on page 3-8
- *Menus* on page 3-23
- *Toolbars* on page 3-24
- *Preferences dialog box* on page 3-28

- *Perspective icons* on page 7-6
- *View markers* on page 7-6
- *ARM DS-5 Using the Debugger:*
  - *App Console view* on page 11-3
  - *ARM Asm Info view* on page 11-5
  - *ARM assembler editor* on page 11-6
  - *Breakpoints view* on page 11-8
  - *C/C++ editor* on page 11-12
  - *Commands view* on page 11-15
  - *Debug Control view* on page 11-18
  - *Disassembly view* on page 11-22
  - *Expressions view* on page 11-26
  - *History view* on page 11-29
  - *Memory view* on page 11-31
  - *Modules view* on page 11-34
  - *Registers view* on page 11-36
  - *Screen view* on page 11-39
  - *Scripts view* on page 11-41
  - *Target view* on page 11-43
  - *Trace view* on page 11-45
  - *Variables view* on page 11-48.
- *ARM DS-5 Using Eclipse:*
  - *Remote Systems view* on page 6-3
  - *Remote Scratchpad view* on page 6-5
  - *Terminals view* on page 6-6.

3.12 Menus

The main menu is located at the top of the Eclipse window and can be customized to your personal preferences. The contents might vary depending on the installed plug-ins and also the active perspective.

Eclipse supports the following options from the main menu:

File Enables you to create, save, close, print, import, and export resources. You can also manage project and file property settings.

Edit Enables you to cut, copy, paste, find, and replace text within a resource.

Navigate Enables you to navigate and quickly find specific resources.

Search Provides an advanced filter for searching through resources.

Project Enables you to manage project build configurations and perform specific builds. You can also customize the build settings for ARM® tools.

Run Enables you to run, send to, debug or configure external tools. You can also manage breakpoints and watchpoints.

Window Enables you to open, close, and customize perspectives, views and editors.

Help Provides documentation on Eclipse and the ARM tools, access to the ARM cheat sheets, and software updates. You can also access the menu option **About ARM DS-5** to open a dialog box that displays the version and build number for the current product.

Right-clicking on a resource produces a context menu for specific tasks. For more information on the other menu options not listed here, use the dynamic help.

3.12.1 See also

Tasks

- *Accessing the dynamic help* on page 3-35.

Reference

- *Perspectives and views* on page 3-19
- *Menus*
- *Toolbars* on page 3-24
- *Preferences dialog box* on page 3-28
- *Menu and toolbar icons* on page 7-5.

3.13 Toolbars

The main toolbar is located at the top of the Eclipse window and can be customized to your personal preferences. The toolbar icons vary depending on the installed plug-ins and also the active perspective.

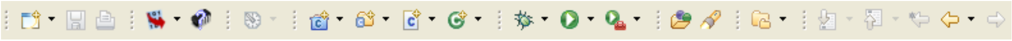


Figure 3-13 Workbench toolbar

Other toolbars associated with specific features are located at the top of each perspective or view.

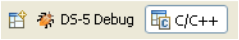


Figure 3-14 Perspective toolbar

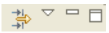


Figure 3-15 View toolbar

3.13.1 See also

Tasks

- *Accessing the dynamic help* on page 3-35.

Reference

- *Perspectives and views* on page 3-19
- *Menus* on page 3-23
- *Preferences dialog box* on page 3-28
- *Menu and toolbar icons* on page 7-5.

3.14 Editing source code

You can use the editors provided with Eclipse to edit your source code or you can use an external editor. If you work with an external editor you must refresh Eclipse to synchronize the views with the latest updates. To do this, in the Project Explorer view, you can click on the updated project, sub-folder, or file and select **Refresh** from the **File** menu. Alternatively you can enable the automatic refresh option by selecting **General** → **Workspace** → **Refresh automatically** in the Preferences dialog box.

When you open a file in Eclipse, a new editor tab appears with the name of the file. An edited file displays an asterisk (*) in the tab name to show that it has unsaved changes.

When you have two or more editor tabs open, you can tile them for side-by-side viewing by clicking on a tab and dragging it over an editor border.

In the left-hand margin of the editor tab you can find a vertical bar that displays markers relating to the active file.

3.14.1 Navigating

There are several ways to navigate to a specific resource within Eclipse. You can use the Project Explorer view to open a resource by browsing through the resource tree and double-clicking on a file. An alternative is to use the keyboard shortcuts or use the options from the **Navigate** menu.

3.14.2 Searching

To locate information or specific code contained within one or more files in Eclipse, you can use the options from the **Search** menu. Textual searching with pattern matching and filters to refine the search fields are provided in a customizable Search dialog box. You can also open this dialog box from the main workbench toolbar.

3.14.3 Content assist

The C/C++ editor, ARM® assembler editor, and the DS-5 Debugger Commands view provide content assistance at the cursor position to auto-complete the selected item. Using the Ctrl+Space keyboard shortcut produces a small dialog box with a list of valid options to choose from. You can shorten the list by partially typing a few characters before using the keyboard shortcut. From the list you can use the Arrow Keys to select the required item and then press the Enter key to insert it.

3.14.4 Bookmarks

Bookmarks can be used to mark a specific position in a file or mark an entire file so that you can return to it quickly. To create a bookmark, select a file or line of code that you want to mark and select **Add Bookmark** from the **Edit** menu. The Bookmarks view displays all the user defined bookmarks and can be accessed by selecting **Window** → **Show View** → **Bookmarks** from the main menu. If the Bookmarks view is not listed then select **Others...** for an extended list.

To delete a bookmark, open the Bookmarks view, click on the bookmark that you want to delete and select **Delete** from the **Edit** menu.

3.14.5 See also

Tasks

- *Accessing the dynamic help* on page 3-35.

Reference

- *Perspectives and views* on page 3-19
- *Menus* on page 3-23
- *Toolbars* on page 3-24
- *Preferences dialog box* on page 3-28
- *Keyboard shortcuts* on page 7-3
- *Menu and toolbar icons* on page 7-5.

3.15 Configuring Eclipse

Eclipse can be customized to your own settings by changing the layout, key bindings, file associations, and color schemes.

Projects and files can also be configured to use the build system in different ways by modifying the properties for the selected resource.

Perspectives can be opened and customized using the options from the **Window** menu or you can use the perspective toolbar. By default a perspective opens in the same window, however, you can change the default settings in the Preferences dialog box.

Views can be moved or docked as applicable by dragging and dropping them into position. Double-clicking on the title bar of a view toggles the maximize/restore options or you can use the relevant toolbar icons. If you have minimized a view you can restore it by clicking on the restore icon in the toolbar at the side of the perspective. By default this is on the right-hand side of the perspective but you can drag and drop it to another location if you prefer. To reset all the views in a perspective to the initial settings and layout you can click on **Reset Perspective...** in the **Window** menu.

3.15.1 See also

Reference

- *Perspectives and views* on page 3-19
- *Menus* on page 3-23
- *Toolbars* on page 3-24
- *Preferences dialog box* on page 3-28
- *Properties dialog box* on page 3-30
- *Keyboard shortcuts* on page 7-3
- *Menu and toolbar icons* on page 7-5.

3.16 Preferences dialog box

Workbench settings can be customized using the Preferences dialog box. You can access this dialog box by selecting **Preferences...** from the **Window** menu. Changes to these settings are saved in the current workspace. If you want to copy your workbench settings to another workspace, use the Export wizard.

The contents of the preferences hierarchy tree include the following groups:

General Controls the workspace, perspectives, editors, build order, linked resources, file associations, path variables, background operations, keyboard and mouse settings.

C/C++ Controls the C/C++ environment settings, CDT build variables, syntax formatting, and default project wizard settings.

DS-5 Controls the default DS-5 environment settings, presentation and formatting for DS-5 editors and views, target configuration database search locations, and the automatic checks for DS-5 product updates.

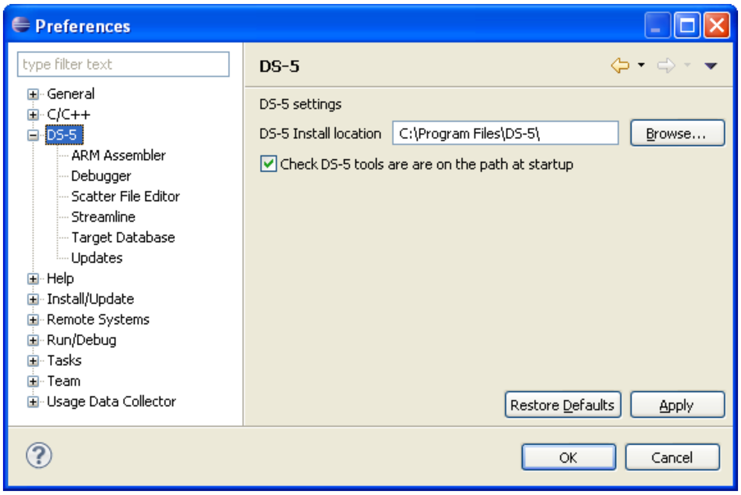
Help Controls how the context help is displayed.

Install/Update Controls the update history, scheduler, and policy.

Remote Systems Controls the settings used by the Remote System Explorer.

Run/Debug Controls the default perspectives, breakpoint, build, and launch settings before running and debugging.

For more information on the other options not listed here, use the dynamic help.



DS-5

DS-5 settings

DS-5 Install location C:\Program Files\DS-5\

☑ Check DS-5 tools are are on the path at startup

Figure 3-16 Window preferences dialog box

3.16.1 See also

Reference

- *Accessing the dynamic help* on page 3-35.

Reference

- *Menus* on page 3-23
- *Perspectives and views* on page 3-19
- *Toolbars* on page 3-24

- *Properties dialog box* on page 3-30
- *Importing and exporting options* on page 3-32
- *Keyboard shortcuts* on page 7-3
- *Menu and toolbar icons* on page 7-5.

3.17 Properties dialog box

Project settings can be customized using the Properties dialog box. You can access this dialog box by selecting a project and then selecting **Properties...** from the **Project** menu. Changes to the customized settings are saved in the project folder in your workspace. You can also customize the C/C++ properties for a single file for example, if you want to apply a specific compiler option to a file during the build.

——— **Note** ———

If you specify different options for a single file, it overrides the options specified in the project configuration panels that apply to all related source files.

The contents of the properties hierarchy tree for a project include the following:

Resource Displays the resource location, modification state, and file type.

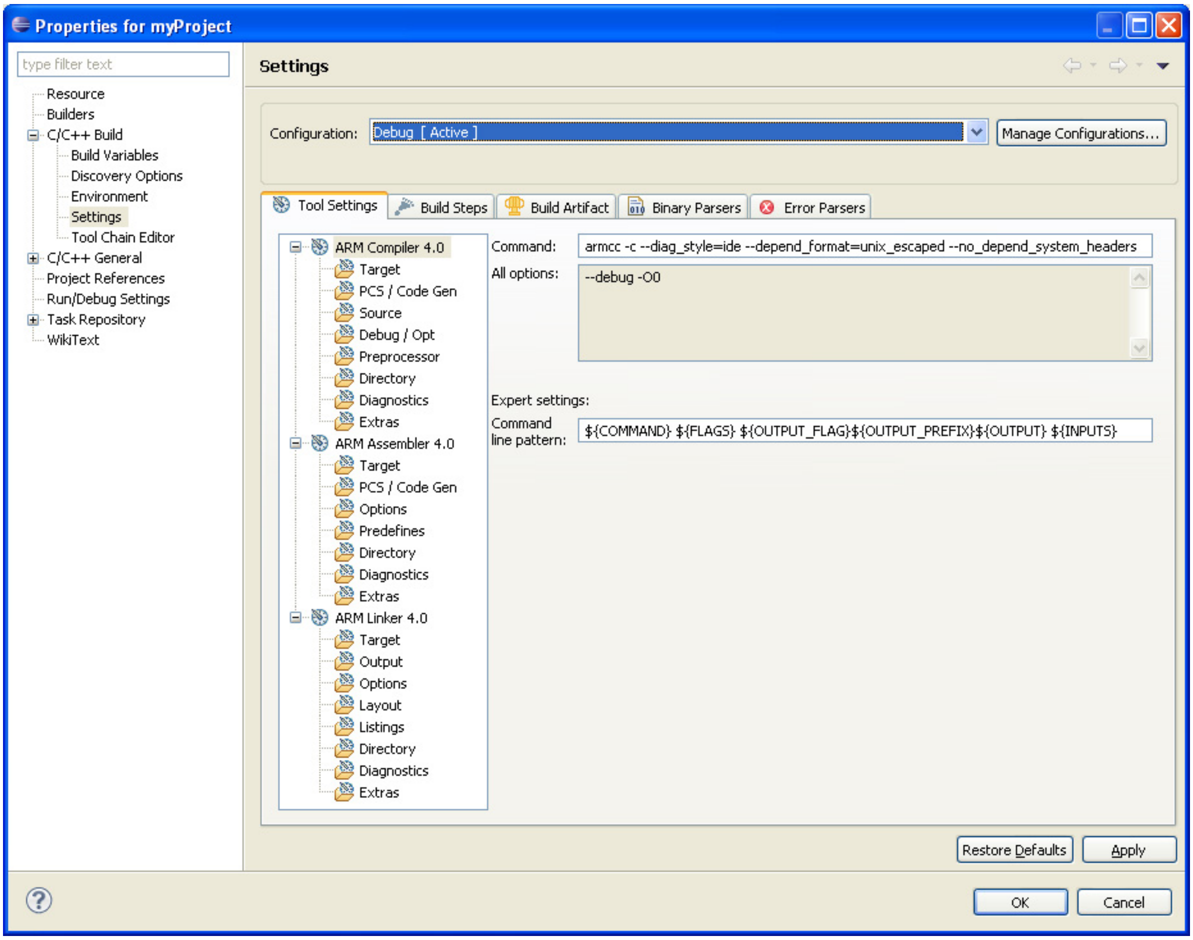
Builders Controls builders available for the selected project.

C/C++ Build Controls the environment, build, and tool chain settings for the active configuration.

C/C++ General Controls documentation, file types, indexer and path/symbol settings.

Project References Controls project dependencies.

For more information on the other options not listed here, use the dynamic help.



Configuration: Debug [Active]

Command: armcc -c --diag_style=ide --depend_format=unix_escaped --no_depend_system_headers

All options: --debug -O0

Expert settings:

Command line pattern: ${COMMAND} ${FLAGS} ${OUTPUT_FLAG}${OUTPUT_PREFIX}${OUTPUT} ${INPUTS}

Figure 3-17 Project properties dialog box

3.17.1 See also

Tasks

- *Accessing the dynamic help* on page 3-35

Reference

- *Menus* on page 3-23
- *Toolbars* on page 3-24
- *Perspectives and views* on page 3-19
- *Preferences dialog box* on page 3-28
- *Keyboard shortcuts* on page 7-3
- *Menu and toolbar icons* on page 7-5.

3.18 Importing and exporting options

A resource must exist in a project within Eclipse before you can use it in a build. If you want to use an existing resource from your file system in one of your projects, the recommended method is to use the Import wizard. To do this, select **Import...** from the **File** menu.

If you want to use a resource externally from Eclipse, the recommended method is to use the Export wizard. To do this, select **Export...** from the **File** menu.

There are several options available in the import and export wizards:

General This option enables you to import and export the following:

- files from an archive zip file
- complete projects
- selected source files and project sub-folders
- workbench preference settings.

C/C++ This option enables you to import the following:

- C/C++ executable files
- C/C++ project settings
- existing code as Makefile project.

You can also export C/C++ project settings and indexes.

Remote Systems

This option enables you to transfer files between the local host and the remote target.

Run/Debug This option enables you to import and export the following:

- breakpoint settings
- launch configurations.

Scatter File Editor

This option enables you to import the memory map from a BCD file and convert it into a scatter file for use in an existing project.

For information on the other options not listed here, use the dynamic help.

3.18.1 See also

Tasks

- *Using the import wizard* on page 3-33
- *Using the export wizard* on page 3-34
- *Accessing the dynamic help* on page 3-35.

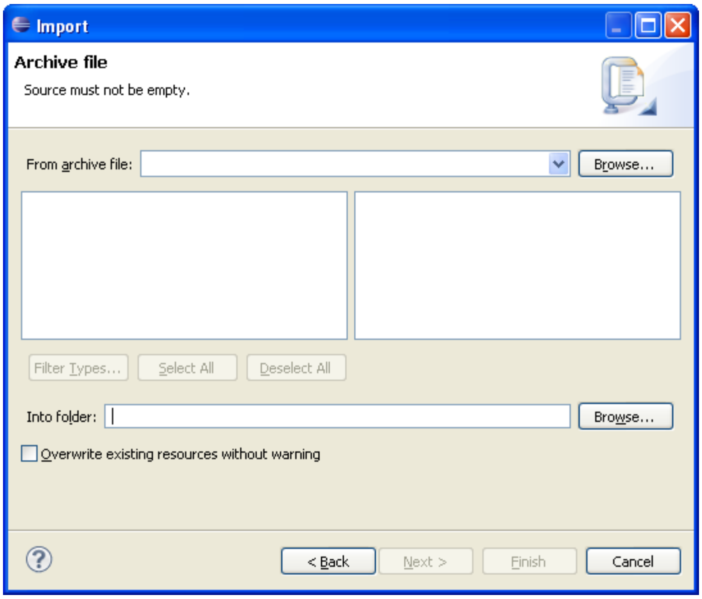
3.19 Using the import wizard

The Import wizard can be used to import complete projects, source files and, project sub-folders in addition to breakpoint and preference settings. Select **Import...** from the **File** menu.

This section focuses on projects, source files and project sub-folders.

To import a complete project either from an archive zip file or an external folder from your file system, you must use the Existing Projects into Workspace wizard. This ensures that the relevant workbench project files are also imported into your workspace.

Individual source files and project sub-folders can be imported using either the Archive File or File System wizard. Both options produce a dialog box similar to the following example. Using the options provided you can select the required resources and specify the relevant options, filename, and destination path.



Import

Archive file

Source must not be empty.

From archive file:

Into folder:

☐ Overwrite existing resources without warning

Figure 3-18 Typical example of the import wizard

With the exception of the Existing Projects into Workspace wizard, files and folders are copied into your workspace when you use the Import wizard. To create a link to an external file or project sub-folder you must use the New File or New Folder wizard.

3.19.1 See also

Tasks

- *Using the export wizard* on page 3-34
- *Accessing the dynamic help* on page 3-35.

Reference

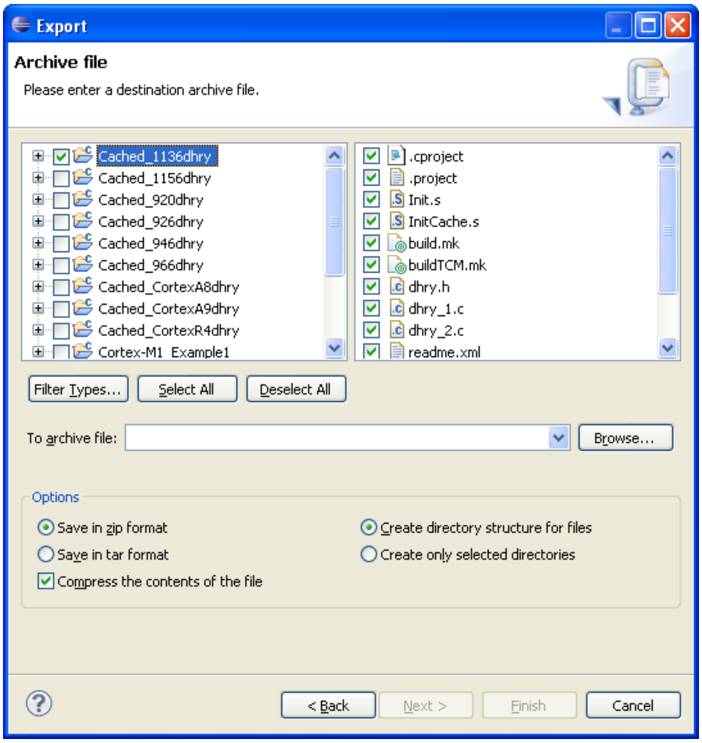
- *Linked resources* on page 3-10
- *Importing and exporting options* on page 3-32.

3.20 Using the export wizard

The Export wizard can be used to export complete projects, source files and, project sub-folders in addition to breakpoint and preference settings. Select **Export...** from the **File** menu.

This section focuses on projects, source files and project sub-folders.

Exporting a complete project, source file or project sub-folder uses the same process. If you want to create a zip file you can use the Archive File wizard, or alternatively you can use the File System wizard. Both options produce a dialog box similar to the example shown here. Using the options provided you can select the required resources and specify the relevant options, filename, and destination path.



Export

Archive file

Please enter a destination archive file.

☑ Cached_1136dhry
□ Cached_1156dhry
□ Cached_920dhry
□ Cached_926dhry
□ Cached_946dhry
□ Cached_966dhry
□ Cached_CortexA8dhry
□ Cached_CortexA9dhry
□ Cached_CortexR4dhry
□ Cortex-M1_Example1

☑ .cproject
☑ .project
☑ Init.s
☑ InitCache.s
☑ build.mk
☑ buildTCM.mk
☑ dhry.h
☑ dhry_1.c
☑ dhry_2.c
☑ readme.xml

To archive file:

Options

● Save in zip format
○ Save in tar format
☑ Compress the contents of the file
● Create directory structure for files
○ Create only selected directories

Figure 3-19 Typical example of the export wizard

3.20.1 See also

Tasks

- *Using the import wizard* on page 3-33
- *Accessing the dynamic help* on page 3-35.

Reference

- *Linked resources* on page 3-10
- *Importing and exporting options* on page 3-32.

3.21 Accessing the dynamic help

To access the dynamic help for a specific workbench feature you must:

1. Click on an editable field for the feature that you want to use.
2. Do one of the following:
   - Click on the question mark icon
   - Select **Dynamic Help** from the **Help** menu.
   - Use a keyboard shortcut:
     - On Windows, **F1** key
     - On Linux , **Shift+F1** key combination.

Dynamic help for the selected feature appears in the About panel at the top of the Related Topics view.

——— **Note** ———

Other possible search results are listed in the Dynamic Help panel at the bottom of the Related Topics view.

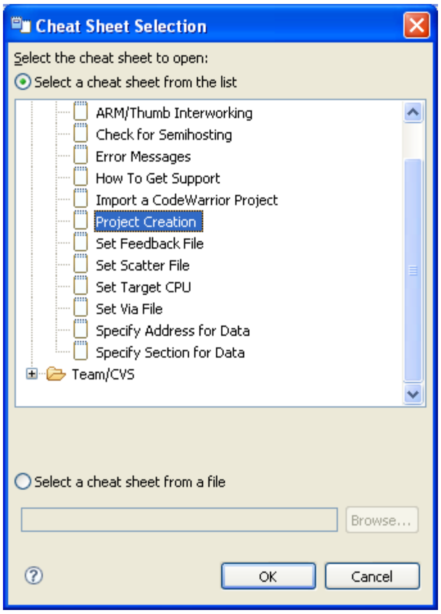
————————

3.22 Using ARM-specific cheat sheets

Cheat sheets are working examples that you can use to guide you through a specific task. Each step in the task is listed in the Cheat Sheets view and the current step is highlighted and expanded. You must perform each step in turn to complete the task.

To use a cheat sheet:

1. Select **Cheat Sheet...** from the **Help** menu.
2. Select a cheat sheet from the list or use **Browse...** to select from a file, see the following figure.



Cheat Sheet Selection

Select the cheat sheet to open:

● Select a cheat sheet from the list

- ARM/Thumb Interworking
- Check for Semihosting
- Error Messages
- How To Get Support
- Import a CodeWarrior Project
- Project Creation
- Set Feedback File
- Set Scatter File
- Set Target CPU
- Set Via File
- Specify Address for Data
- Specify Section for Data
- Team/CVS

○ Select a cheat sheet from a file

Figure 3-20 Selecting a cheat sheet

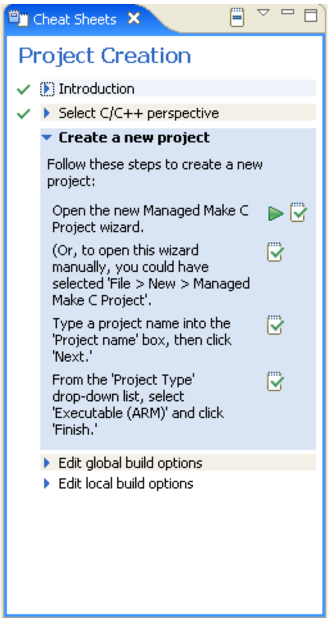
3. Click on **OK**.
4. Click on the **Click to Begin** link on the Introduction step to start the task. If you open a cheat sheet more than once, the link on the Introduction step changes text to **Click to Restart**.
5. Follow the instructions step by step. When you complete an instruction, click on the **Click to Complete** link to move on to the next instruction. Some of the instructions might have a **Click to Perform** link for you to use if you want that instruction to be performed automatically by the cheat sheet.

The task is complete when you have performed every step listed in the Cheat Sheets view. This is shown by the tick icon shown in the left-hand margin of the cheat sheet see Figure 3-21 on page 3-37.

Figure 3-21 on page 3-37 shows a typical example of a cheat sheet. Steps one and two are shown as complete, step three (the current step) is highlighted and expanded ready for use. Four instructions are listed in step three, complete each instruction in turn. When step three is fully complete, the cheat sheet moves on to populate and reveal the instructions in step four.

Note

Subsequent steps are not populated until you fully complete the previous step.



Project Creation

- ✓ Introduction
- ✓ Select C/C++ perspective
- Create a new project

Follow these steps to create a new project:

Open the new Managed Make C Project wizard.

(Or, to open this wizard manually, you could have selected 'File > New > Managed Make C Project'.

Type a project name into the 'Project name' box, then click 'Next.'

From the 'Project Type' drop-down list, select 'Executable (ARM)' and click 'Finish.'

- Edit global build options
- Edit local build options

Figure 3-21 Typical example of a cheat sheet

3.23 Installing new features

To install new features:

1. Select **Install New Software...** from the **Help** menu.
2. Select a site from the **Work with** drop-down list or click **Add...** to add a new site.
3. Select the required plug-in check boxes. Click on **Next >**.
4. Review the list of plug-ins that you are about to install. Click on **Next >**.
5. Read each of the license agreements and accept them. If you do not accept a license agreement, you cannot install that feature. Click on **Finish**.
6. Click on **Yes** to restart Eclipse and complete the installation.

To install updated features, select **Check for Updates** from the **Help** menu.

——— **Note** ————

You can change the default setting for downloading and installing updated features by selecting **Install/Update** → **Automatic Updates** in the Preferences dialog box.

————————

3.23.1 See also

Reference

- *Preferences dialog box* on page 3-28.

3.24 Restrictions of use

This section lists the specific restrictions and peculiarities that apply when using Eclipse.

Organizing projects

The recommended structure for project source files is to create them in the project folder or sub-folder. If a source file is created in a folder that is higher than the project, an absolute link is created.

Opening an existing Eclipse project

You must use the import wizard to import an existing Eclipse project into your workspace.

Inter-project dependencies

Nested projects are not supported. Each project must be organized as a discrete entity. Inter-project dependencies can be set up by referencing other projects that reside in your workspace. Select **Project** → **Properties** → **Project References** from the main menu to manually add references.

Link order

Specifying the link order of your object files within the same project is not possible with Eclipse. As a workaround, if you split your object files into different projects, you can specify the project build order. Select **Window** → **Preferences...** → **General** → **Workspace** → **Build Order** from the main menu.

Restore Defaults

Restoring the defaults of a project discards all information that is not a part of the project type. All settings changed in the New Project Wizard are lost.

Chapter 4
Working with projects

You can use Eclipse to create projects for ARM® targets. Projects are top level folders in your workspace that contain related files and sub-folders. A project must exist in your workspace before a new file can be added or an existing file can be imported.

The following topics describe how to work with Eclipse projects:

Tasks

- *Creating a new C or C++ project* on page 4-4
- *Creating a new Makefile project for an ARM Linux target* on page 4-5
- *Importing an existing Eclipse project* on page 4-7
- *Setting up the compilation tools for a specific build configuration* on page 4-9
- *Configuring the C/C++ build behavior* on page 4-10
- *Updating a project to a new toolchain* on page 4-12
- *Adding a new source file to your project* on page 4-13.

Concepts

- *Project types* on page 4-2.

4.1 Project types

Different project types are provided with Eclipse, depending on the requirements of your project.

Note

Bare metal projects require a software license for the ARM Compiler to successfully build an ELF image.

Bare-metal Executable

Uses the ARM Compiler toolchain to build a bare-metal executable ELF image.

Bare-metal Static library

Uses the ARM Compiler toolchain to build a library of ELF object format members for a bare-metal project.

Note

It is not possible to debug or run a stand-alone library file until it is linked into an image.

Executable

Uses the GNU Compilation Tools to build a Linux executable ELF image.

Shared Library

Uses the GNU Compilation Tools to build a dynamic library for a Linux application.

Static library

Uses the GNU Compilation Tools to build a library of ELF object format members for a Linux application.

Note

It is not possible to debug or run a stand-alone library file until it is linked into an image.

Makefile project

Creates a project that requires a Makefile, that you can write yourself, to build the project.

Note

Eclipse does not modify Makefile projects.

4.1.1 Build configurations

By default, the new project wizard provides two separate build configurations:

Debug The debug target is configured to build output binaries that are fully debuggable, at the expense of optimization. It configures the compiler optimization setting to minimum (level 0), to provide an ideal debug view for code development.

Release The release target is configured to build output binaries that are highly optimized, at the expense of a poorer debug view. It configures the compiler optimization setting to high (level 3).

In all new projects, the `Debug` configuration is automatically set as the active configuration. This can be changed in the C/C++ Build Settings panel of the Project Properties dialog box.

4.1.2 See also

Tasks

- *Accessing the dynamic help* on page 3-35
- *Creating a new C or C++ project* on page 4-4
- *Importing an existing Eclipse project* on page 4-7.

Reference

- *Menus* on page 3-23
- *Toolbars* on page 3-24
- *Perspectives and views* on page 3-19
- *Preferences dialog box* on page 3-28
- *Keyboard shortcuts* on page 7-3
- *Menu and toolbar icons* on page 7-5
- *ARM® DS-5™ Getting Started with DS-5*:
  - *Licensing and product updates* on page 4-4.

4.2 Creating a new C or C++ project

To create a new C or C++ Project:

1. Select **File** → **New** → **Project...** from the main menu.
2. Expand the **C/C++** group.
3. Select either **C Project** or **C++ Project**.
4. Click on **Next**.
5. Enter a project name.
6. Leave the **Use default location** option selected so that the project is created in the default folder shown. Alternatively, deselect this option and browse to your preferred project folder.
7. Select the type of project that you want to create.
8. Click on **Finish** to create your new project.

   The project is visible in the Project Explorer view.

4.2.1 See also

Tasks

- *Accessing the dynamic help* on page 3-35
- *Setting up the compilation tools for a specific build configuration* on page 4-9
- *Configuring the C/C++ build behavior* on page 4-10
- *Adding a new source file to your project* on page 4-13.

Concepts

- *Project types* on page 4-2.

Reference

- *Menus* on page 3-23
- *Toolbars* on page 3-24
- *Perspectives and views* on page 3-19
- *Preferences dialog box* on page 3-28
- *Keyboard shortcuts* on page 7-3
- *Menu and toolbar icons* on page 7-5.

4.3 Creating a new Makefile project for an ARM Linux target

To create a new C or C++ Project for an ARM Linux target:

1. Select **File** → **New** → **Project...** from the main menu.
2. Expand the **C/C++** group.
3. Select either **C Project** or **C++ Project**.
4. Click on **Next**.
5. Enter a project name.
6. Leave the **Use default location** option selected so that the project is created in the default folder shown. Alternatively, deselect this option and browse to your preferred project folder.
7. Expand the **Makefile project** group.
8. Select **Empty project** in the Project type panel.
9. Select the toolchain that you want to use when building your project.
10. Click on **Finish** to create your new project. The project is visible in the Project Explorer view.
11. Before you can build the project, a `Makefile`, containing all the compilation tool settings, is required. The easiest way to create one is to copy the `Makefile` from the example project, `hello` and paste it into your new project. The `hello` project can be found in the Linux examples provided with DS-5.
12. Edit the `Makefile` as follows:
    a. Locate the line that contains `OBJS = hello.o`.
    b. Replace `hello.o` with the names of the object files corresponding to your source files.
    c. Locate the line that contains `TARGET = hello`.
    d. Replace `hello` with the name of the target image file corresponding to your source files.
    e. Save the file.
13. Add your C/C++ files to the project.

You can now build the new project:

1. In the Project Explorer view, select the project that you want to build.
2. Select **Project** → **Build Project** from the main menu.

4.3.1 See also

Tasks

- *Accessing the dynamic help* on page 3-35
- *Setting up the compilation tools for a specific build configuration* on page 4-9
- *Configuring the C/C++ build behavior* on page 4-10
- *Adding a new source file to your project* on page 4-13.

Concepts

- *Project types* on page 4-2.

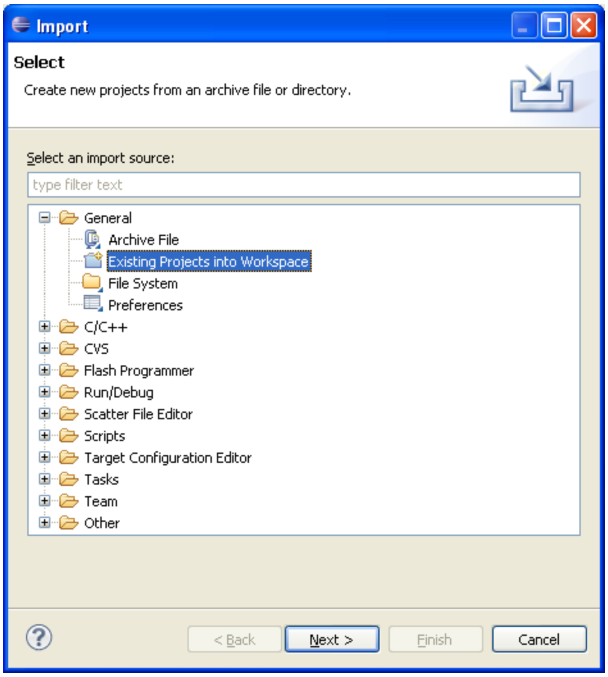
Reference

- *Menus* on page 3-23
- *Toolbars* on page 3-24
- *Perspectives and views* on page 3-19
- *Preferences dialog box* on page 3-28
- *Keyboard shortcuts* on page 7-3
- *Menu and toolbar icons* on page 7-5.

4.4 Importing an existing Eclipse project

To import an existing Eclipse project into your workspace:

1. Select **Import...** from the **File** menu.
2. To import an existing project, select **Existing Project into Workspace**. Click on **Next**.



Import

Select

Create new projects from an archive file or directory.

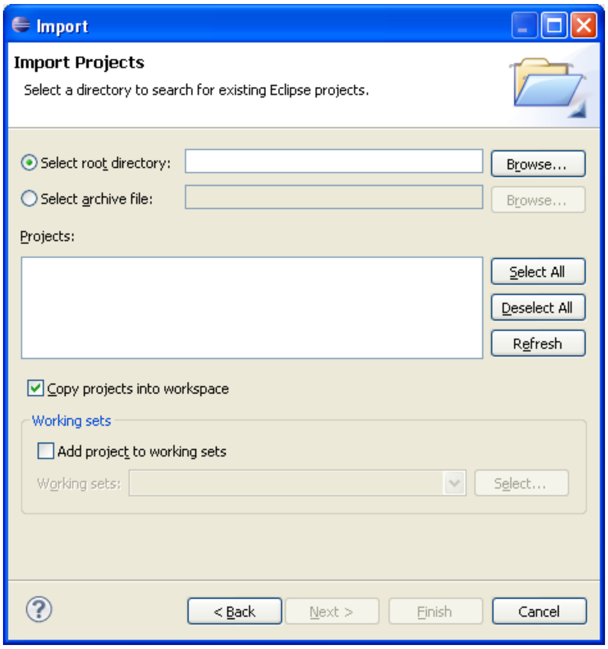
Select an import source:

type filter text

- General
  - Archive File
  - Existing Projects into Workspace
  - File System
  - Preferences
- C/C++
- CVS
- Flash Programmer
- Run/Debug
- Scatter File Editor
- Scripts
- Target Configuration Editor
- Tasks
- Team
- Other

Figure 4-1 Selecting the import source type

3. Click on **Browse** to select the import folder containg the project that you want to import.
4. In the Projects panel select the project that you want to import.
5. Select **Copy projects into workspace** if required or deselect to create links to your existing project(s) and associated files.
6. If you are not using working sets to group your projects then you can skip this step.
   a. Select **Add project to working sets**.
   b. Click on **Select...**.
   c. Select an existing working set or create a new one and then select it.
   d. Click **OK**.
7. Click on **Finish**.



Import Projects

Select a directory to search for existing Eclipse projects.

● Select root directory:

○ Select archive file:

Projects:

☑ Copy projects into workspace

☐ Add project to working sets

Working sets:

Figure 4-2 Selecting an existing Eclipse projects for import

Note

If your existing project contains project settings from an older version of the build system, you are given the option to update your project. Using the latest version means that you can access all the latest toolchain features.

The imported project is visible in the Project Explorer view.

4.4.1 See also

Tasks

- *Creating a working set* on page 3-15
- *Changing the top level element when displaying working sets* on page 3-17
- *Deselecting a working set* on page 3-18
- *Accessing the dynamic help* on page 3-35
- *Setting up the compilation tools for a specific build configuration* on page 4-9
- *Configuring the C/C++ build behavior* on page 4-10
- *Adding a new source file to your project* on page 4-13.

Concepts

- *About working sets* on page 3-14
- *Project types* on page 4-2.

Reference

- *Menus* on page 3-23
- *Toolbars* on page 3-24
- *Perspectives and views* on page 3-19
- *Preferences dialog box* on page 3-28
- *Keyboard shortcuts* on page 7-3
- *Menu and toolbar icons* on page 7-5.

4.5 Setting up the compilation tools for a specific build configuration

The C/C++ Build configuration panels enable you to set up the compilation tools for a specific build configuration, in your project. The settings in these panels determine how the compilation tools build an ARM executable image or library.

To access the build configuration panels:

1. Select the source file or project in the Project Explorer view.
2. Select **Properties** from the **Project** menu.
3. Expand **C/C++ Build** in the Properties dialog box.
4. Select **Settings**.
5. The Configuration panel shows the active configuration. If required, click **Manage Configurations...** from the Configuration panel to create a new build configuration or change the active setting.
6. The compilation tools available for the current project, and their respective build configuration panels are displayed in the **Tool Settings** tab. Click on this tab and configure the build as required.

   Note

   Makefile projects do not use these configuration panels. The Makefile must contain all the required compilation tool settings.

7. Click OK to save the settings.

4.5.1 See also

Tasks

- *Accessing the dynamic help* on page 3-35
- *Configuring the C/C++ build behavior* on page 4-10.

Concepts

- *Project types* on page 4-2.

Reference

- *Menus* on page 3-23
- *Toolbars* on page 3-24
- *Perspectives and views* on page 3-19
- *Preferences dialog box* on page 3-28
- *Keyboard shortcuts* on page 7-3
- *Menu and toolbar icons* on page 7-5.

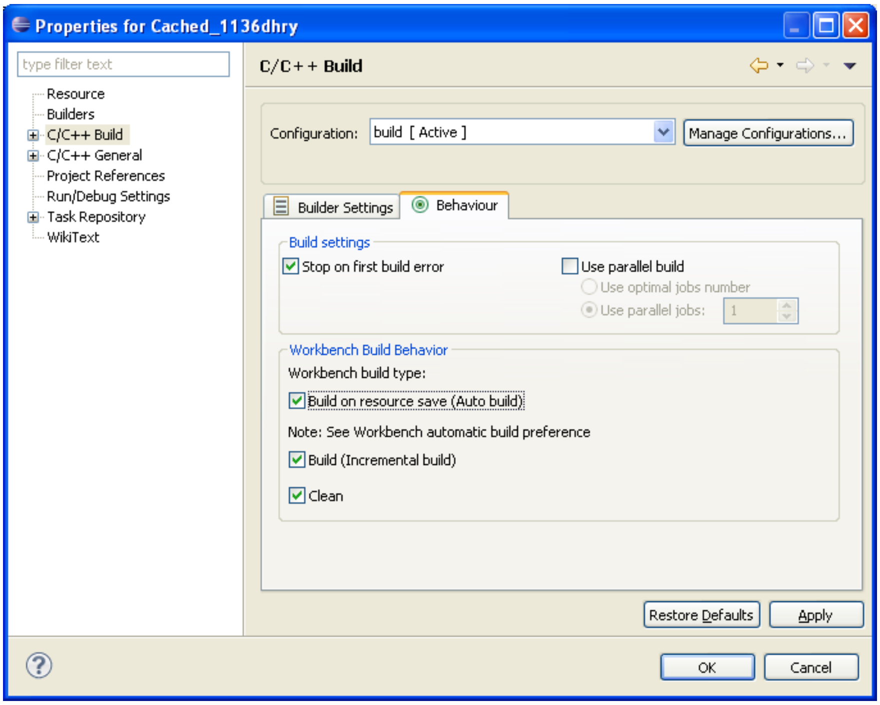
4.6 Configuring the C/C++ build behavior

A build is the process of compiling and linking source files to generate an output file. A build can be applied to either a specific set of projects or the entire workspace. It is not possible to build an individual file or sub-folder.

Eclipse provides an incremental build that applies the selected build configuration to resources that have changed since the last build. Another type of build is the Clean build that applies the selected build configuration to all resources, discarding any previous build states.

Automatic

This is an incremental build that operates over the entire workspace and can run automatically when a resource is saved. This setting must be enabled for each project by selecting **Build on resource save (Auto build)** in the **Behaviour** tab. By default, this behavior is not selected for any project.



Configuration: build [Active]

Build settings

☑ Stop on first build error

☐ Use parallel build

○ Use optimal jobs number

● Use parallel jobs: 1

Workbench Build Behavior

Workbench build type:

☑ Build on resource save (Auto build)

Note: See Workbench automatic build preference

☑ Build (Incremental build)

☑ Clean

Figure 4-3 Workbench build behavior

You must also ensure that **Build Automatically** is selected from the **Project** menu. By default, this menu option is selected.

Manual

This is an incremental build that operates over the entire workspace on projects with **Build (Incremental build)** selected. By default, this behavior is selected for all projects.

You can run an incremental build by selecting **Build All** or **Build Project** from the **Project** menu.

——— **Note** ———

Manual builds do not save before running so you must save all related files before selecting this option! To save automatically before building, you can change your default settings by selecting **Preferences...** → **General** → **Workspace** from the **Window** menu.

Clean

This option discards any previous build states including object files and images from the selected projects. The next automatic or manual build after a clean, applies the selected build configuration to all resources.

You can run a clean build on either the entire workspace or specific projects by selecting **Clean...** from the **Project** menu. You must also ensure that **Clean** is selected in the **C/C++ Build** → **Behaviour** tab of the Preferences dialog box. By default, this behavior is selected for all projects.

Build order is a feature where inter-project dependencies are created and a specific build order is defined. For example, an image might require several object files to be built in a specific order. To do this, you must split your object files into separate smaller projects, reference them within a larger project to ensure they are built before the larger project. Build order can also be applied to the referenced projects.

4.6.1 See also

Tasks

- *Accessing the dynamic help* on page 3-35
- *Creating a new C or C++ project* on page 4-4
- *Importing an existing Eclipse project* on page 4-7
- *Setting up the compilation tools for a specific build configuration* on page 4-9.
- *Configuring the C/C++ build behavior* on page 4-10.

Concepts

- *Project types* on page 4-2.

Reference

- *Menus* on page 3-23
- *Toolbars* on page 3-24
- *Perspectives and views* on page 3-19
- *Preferences dialog box* on page 3-28
- *Keyboard shortcuts* on page 7-3
- *Menu and toolbar icons* on page 7-5.

4.7 Updating a project to a new toolchain

If you have several products installed, only the latest toolchain is listed in the New Project wizard. Therefore, if you have projects that use an older toolchain, you must update them to the latest toolchain.

To update them to use the new toolchain:

1. Right-click on the project in thr Project Explorer view.
2. Select **Convert To...** from the context menu to display the Project Converters dialog box.
3. Select the required toolchain.

   Note

   Beware that if you convert back to an earlier version you might lose toolchain functionality.

4. Click **OK**.

4.7.1 See also

Tasks

- *Accessing the dynamic help* on page 3-35
- *Creating a new C or C++ project* on page 4-4
- *Importing an existing Eclipse project* on page 4-7.

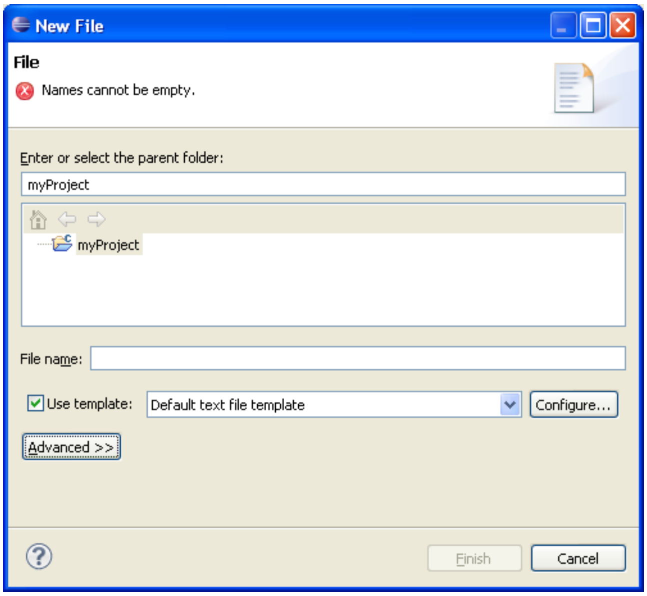
Concepts

- *Project types* on page 4-2.

4.8 Adding a new source file to your project

To add a new source file to your project:

1. Select **File** → **New** → **File from Template** from the main menu.
2. Select your project folder or sub-folder from the New File dialog box.



New File

File

Names cannot be empty.

Enter or select the parent folder: myProject

File name:

☑ Use template: Default text file template

Figure 4-4 Naming your file

3. Enter a filename with the relevant extension in the **File name** field.
4. Accept the default template or click **Configure...** if you want to change it.
5. Click on **Finish**.

   The new source file is visible in the Project Explorer view.

Note

- You can also create files, or drag and drop files directly into the project folder, using the file system. To update the views in Eclipse, click on the relevant project in the Project Explorer view and select **Refresh** from the **File** menu.
- You can also drag and drop files directly into a project folder, in the Project Explorer view of Eclipse.

Chapter 5
Working with editors

The following topics describe how to use the editors when developing a project for an ARM® target:

Concepts

Reference

5.1 Overview of the C/C++ editor

The standard C/C++ editor is provided by the CDT plug-in that provides C and C++ extensions to Eclipse. It provides syntax highlighting, formatting of code and content assistance when editing C/C++ code. Embedded assembler in C/C++ files is supported by the ARM Compiler but this editor does not support it and so an error is displayed. This type of code is ARM-specific and accepted Eclipse behavior so you can ignore the syntax error.

If this is not the default editor, right-click on a source file in the Project Explorer view and select **Open With → C/C++ Editor** from the context menu.

See the *C/C++ Development User Guide* for more information. Select **Help → Help Contents** from the main menu.

5.2 Overview of the ARM assembler editor

The ARM® assembler editor provides syntax highlighting, formatting of code and content assistance for labels in ARM assembly language source files. You can change the default settings in the Preferences dialog box.

If this is not the default editor, right-click on your source file in the Project Explorer view and select **Open With → ARM Assembler Editor** from the context menu.

The following shortcuts are also available for use:

Table 5-1 ARM assembler editor shortcuts

Content assist	Content assist provides auto-completion on labels existing in the active file. When entering a label for a branch instruction, Partially type the label and then use the keyboard shortcut Ctrl+Space to display a list of valid auto-complete options. Use the Arrow Keys to select the required label and press Enter to complete the term. Continue typing to ignore the auto-complete list.
Editor focus	The following options change the editor focus: • Outline View provides a list of all areas and labels in the active file. Click on an area or label to move the focus of the editor to the position of the selected item. • Select a label from a branch instruction and press F3 to move the focus of the editor to the position of the selected label.
Formatter activation	Press Ctrl+Shift+F to activate the formatter settings.
Block comments	Block comments are enabled or disabled by using Ctrl+Semicolon. Select a block of code and apply the keyboard shortcut to change the commenting state.

5.2.1 See also

Concepts

- *Preferences dialog box* on page 3-28.

5.3 Overview of the ELF content editor

The ELF content editor creates forms for the selected ELF file. You can use this editor to view the contents of image files and object files. The editor is read-only and cannot be used to modify the contents of any files.

If this is not the default editor, right-click on your source file in the Project Explorer view and select **Open With → ELF Content Editor** from the context menu.

The ELF content editor displays one or more of the following tabs depending on the selected file type:

Overview An image or object file shows header information and section details.

Symbol Table

Tabular view showing the breakdown of all symbols.

Disassembly

Textual view with syntax highlighting.

5.3.1 See also

Reference

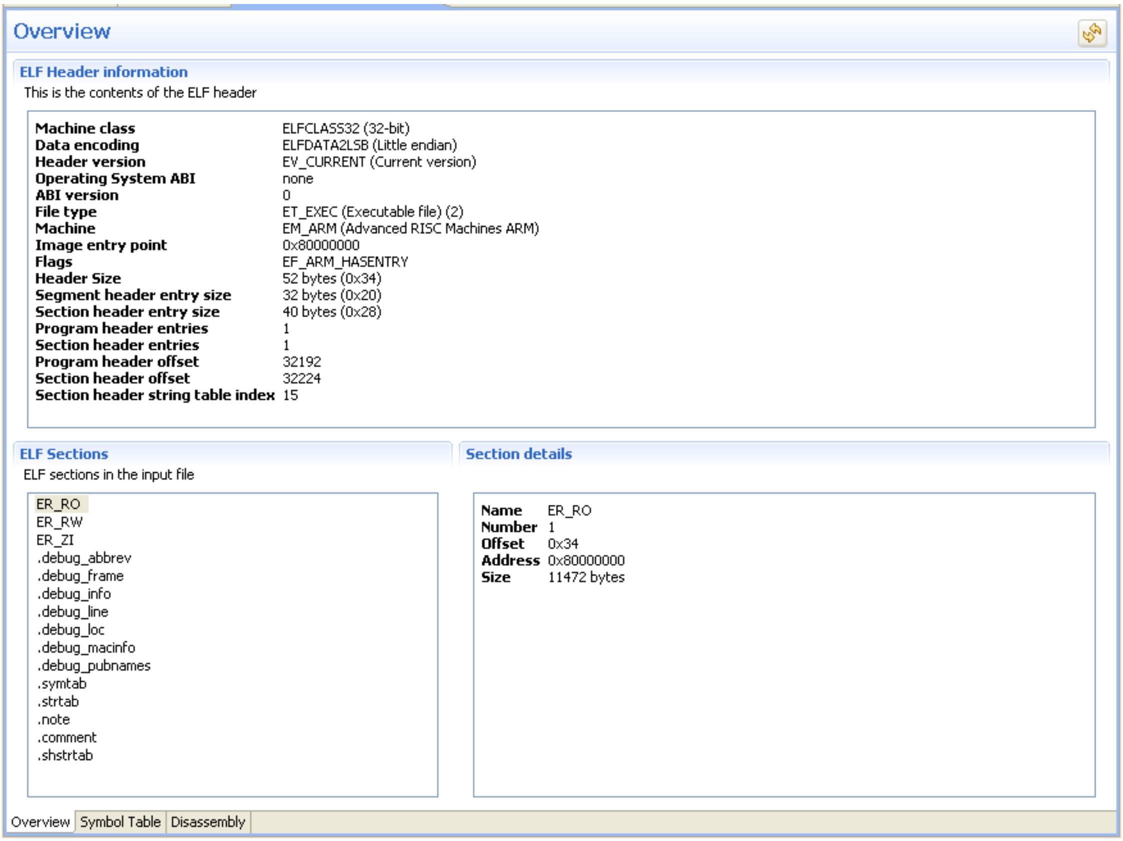
- *ELF content editor - Overview tab* on page 5-5
- *ELF content editor - Symbol Table tab* on page 5-6
- *ELF content editor - Disassembly tab* on page 5-7.

5.4 ELF content editor - Overview tab

The ELF content editor **Overview** tab displays different information depending on the selected file type.

5.4.1 Image or object file

The **Overview** tab provides a form view of the ELF header and section information.



Overview

ELF Header information

This is the contents of the ELF header

Machine class	ELFCLASS32 (32-bit)
Data encoding	ELFDATA2LSB (Little endian)
Header version	EV_CURRENT (Current version)
Operating System ABI	none
ABI version	0
File type	ET_EXEC (Executable file) (2)
Machine	EM_ARM (Advanced RISC Machines ARM)
Image entry point	0x80000000
Flags	EF_ARM_HASENTRY
Header Size	52 bytes (0x34)
Segment header entry size	32 bytes (0x20)
Section header entry size	40 bytes (0x28)
Program header entries	1
Section header entries	1
Program header offset	32192
Section header offset	32224
Section header string table index	15

ELF Sections

ELF sections in the input file

ER_RO
ER_RW
ER_ZI
.debug_abbrev
.debug_frame
.debug_info
.debug_line
.debug_loc
.debug_macinfo
.debug_pubnames
.symtab
.strtab
.note
.comment
.shstrtab

Section details

Name	ER_RO
Number	1
Offset	0x34
Address	0x80000000
Size	11472 bytes

Figure 5-1 Overview tab

5.4.2 See also

Concepts

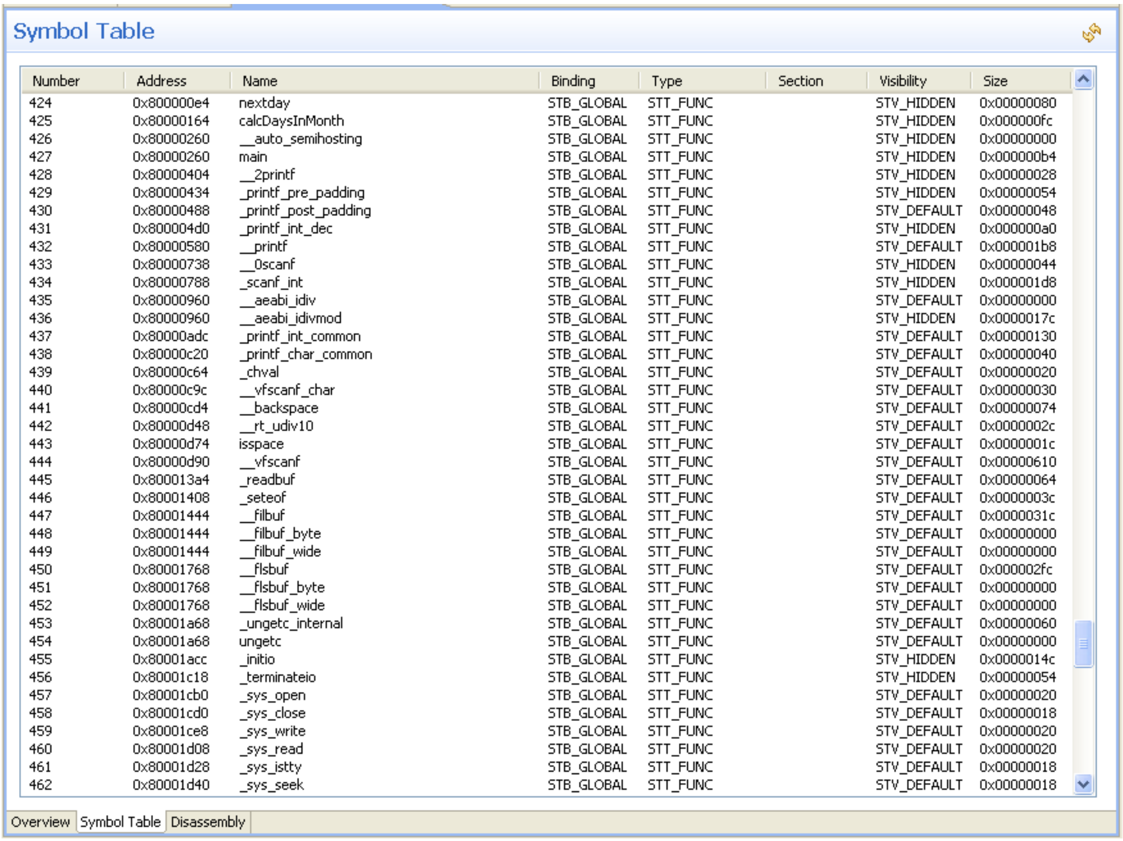
- *Overview of the ELF content editor* on page 5-4.

Reference

- *ELF content editor - Symbol Table tab* on page 5-6
- *ELF content editor - Disassembly tab* on page 5-7.

5.5 ELF content editor - Symbol Table tab

The **Symbol Table** tab provides a tabular view of the symbols.



Symbol Table

Number	Address	Name	Binding	Type	Section	Visibility	Size
424	0x800000e4	nextday	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x00000080
425	0x80000164	calcDaysInMonth	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x000000fc
426	0x80000260	__auto_semihosting	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x00000000
427	0x80000260	main	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x000000b4
428	0x80000404	__2printf	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x00000028
429	0x80000434	_printf_pre_padding	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x00000054
430	0x80000488	_printf_post_padding	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000048
431	0x800004d0	_printf_int_dec	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x000000a0
432	0x80000580	__printf	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x000001b8
433	0x80000738	__0scanf	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x00000044
434	0x80000788	_scanf_int	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x000001d8
435	0x80000960	__aeabi_idiv	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000000
436	0x80000960	__aeabi_idivmod	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x0000017c
437	0x80000adc	_printf_int_common	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000130
438	0x80000c20	_printf_char_common	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000040
439	0x80000c64	_chval	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000020
440	0x80000c9c	__vfscanf_char	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000030
441	0x80000cd4	__backspace	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000074
442	0x80000d48	__rt_udiv10	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x0000002c
443	0x80000d74	isspace	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x0000001c
444	0x80000d90	__vfscanf	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000610
445	0x800013a4	_readbuf	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000064
446	0x80001408	_seteof	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x0000003c
447	0x80001444	__filbuf	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x0000031c
448	0x80001444	__filbuf_byte	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000000
449	0x80001444	__filbuf_wide	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000000
450	0x80001768	__flsbuf	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x000002fc
451	0x80001768	__flsbuf_byte	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000000
452	0x80001768	__flsbuf_wide	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000000
453	0x80001a68	_ungetc_internal	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000060
454	0x80001a68	ungetc	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000000
455	0x80001acc	_initio	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x0000014c
456	0x80001c18	_terminateio	STB_GLOBAL	STT_FUNC		STV_HIDDEN	0x00000054
457	0x80001cb0	_sys_open	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000020
458	0x80001cd0	_sys_close	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000018
459	0x80001ce8	_sys_write	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000020
460	0x80001d08	_sys_read	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000020
461	0x80001d28	_sys_istty	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000018
462	0x80001d40	_sys_seek	STB_GLOBAL	STT_FUNC		STV_DEFAULT	0x00000018

Figure 5-2 Symbol Table tab

5.5.1 See also

Concepts

- *Overview of the ELF content editor* on page 5-4.

Reference

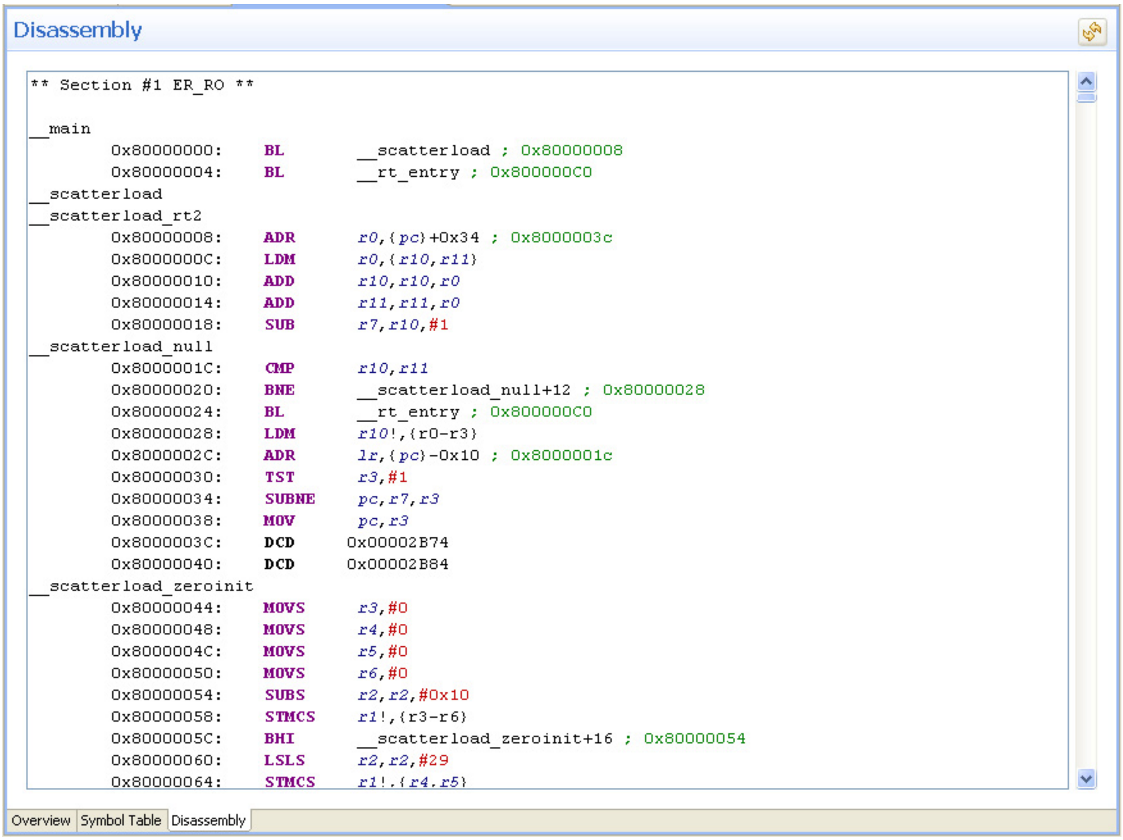
- *ELF content editor - Overview tab* on page 5-5
- *ELF content editor - Disassembly tab* on page 5-7.

5.6 ELF content editor - Disassembly tab

The **Disassembly** tab displays the output with syntax highlighting. The color schemes and syntax preferences use the same settings as the ARM® assembler editor. There are several keyboard combinations that can be used to navigate around the output:

- Use Ctrl+F to open the Find dialog box to search the output
- Use Ctrl+Home to move the focus to the beginning of the output
- Use Ctrl+End to move the focus to the end of the output.
- Use Page Up and Page Down to navigate through the output one page at a time.

You can also use the **Copy** and **Find** options in the context menu by right-clicking in the Disassembly view.



```
** Section #1 ER_RO **

__main
        0x80000000:    BL       __scatterload ; 0x80000008
        0x80000004:    BL       __rt_entry ; 0x800000C0
__scatterload
__scatterload_rt2
        0x80000008:    ADR      r0,{pc}+0x34 ; 0x8000003c
        0x8000000C:    LDM      r0,{r10,r11}
        0x80000010:    ADD      r10,r10,r0
        0x80000014:    ADD      r11,r11,r0
        0x80000018:    SUB      r7,r10,#1
__scatterload_null
        0x8000001C:    CMP      r10,r11
        0x80000020:    BNE      __scatterload_null+12 ; 0x80000028
        0x80000024:    BL       __rt_entry ; 0x800000C0
        0x80000028:    LDM      r10!,{r0-r3}
        0x8000002C:    ADR      lr,{pc}-0x10 ; 0x8000001c
        0x80000030:    TST      r3,#1
        0x80000034:    SUBNE    pc,r7,r3
        0x80000038:    MOV      pc,r3
        0x8000003C:    DCD     0x00002B74
        0x80000040:    DCD     0x00002B84
__scatterload_zeroinit
        0x80000044:    MOVS     r3,#0
        0x80000048:    MOVS     r4,#0
        0x8000004C:    MOVS     r5,#0
        0x80000050:    MOVS     r6,#0
        0x80000054:    SUBS     r2,r2,#0x10
        0x80000058:    STMCS    r1!,{r3-r6}
        0x8000005C:    BHI      __scatterload_zeroinit+16 ; 0x80000054
        0x80000060:    LSLS     r2,r2,#29
        0x80000064:    STMCS    r1!,{r4,r5}
```

Figure 5-3 Disassembly tab

5.6.1 See also

Concepts

- *Overview of the ELF content editor* on page 5-4
- *Overview of the ARM assembler editor* on page 5-3.

Reference

- *ELF content editor - Overview tab* on page 5-5
- *ELF content editor - Symbol Table tab* on page 5-6.

5.7 Overview of the scatter file editor

The scatter file editor enables you to easily create and edit scatter files for use with the ARM linker to construct the memory map of an image. It provides a text editor, a hierarchical tree and a graphical view of the regions and output sections of an image. You can change the default syntax formatting and color schemes in the Preferences dialog box.

If this is not the default editor, right-click on your source file in the Project Explorer view and select **Open With → Scatter File Editor** from the context menu.

The scatter file editor displays the following tabs:

Source Textual view of the source code with syntax highlighting and formatting.

Regions/Sections

A graphical view showing load and execute memory maps. These are not editable, however, you can select a load region to show the related memory blocks in the execution regions.

The scatter file editor also provides a hierarchical tree with associated toolbar and context menus using the Outline view. Clicking on a region or section in the Outline view moves the focus of the editor to the relevant position in your code. If this view is not visible, select **Show View → Outline** from the Window menu.

——— **Note** ———

The linker documentation for the ARM Compiler toolchain describes in more detail how to use scatter files.

Before you can use a scatter file you must add the `--scatter=file` option to the project within the **C/C++ Build → Settings → Tool settings → ARM Linker → Image Layout** panel of the Properties dialog box.

5.7.1 Example of a scatter file

To create a scatter file:

1. Use an existing project or create a new project.

2. Create a new project and add a new empty text file with the extension `.scat`. for example `scatter.scat`.

3. Using the Outline view, click on the toolbar icon or right-click and select **Add load region** from the context menu.

   The **Add load region** dialog box is displayed.

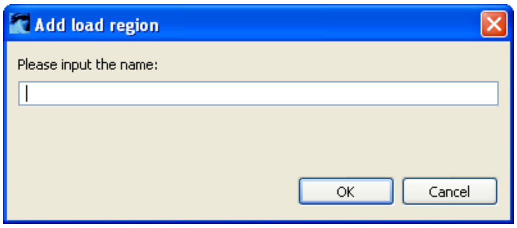


Figure 5-4 Add load region name

4. Enter a load region name, for example, `LR1`.

5. Click **OK**.

6. Modify the load region as shown in the following example.

Example 5-1 Simple scatter file

```
LR1 0x0000 0x8000
{
    LR1_er1 0x0000 0x8000
    {
        * (+RO)
    }

    LR1_er2 0x10000 0x6000
    {
        * (+RW,+ZI)
    }
}
```

This example shows a simple scatter file

7. Save your changes.

8. To set up a more complex scatter file:

    a. Repeat steps 3 to 5 to add a second load region `LR2`.

    b. Modify the LR1 and LR2 load regions as shown in the following example.

Example 5-2 Complex scatter file

```
LR1 0x0
{
    LR1_er1 0x0
    {
        program.o (+RO)
    }

    LR1_er2 0x18000 0x8000
    {
        program1.o (+RW,+ZI)
    }
}

LR2 0x4000
{
    LR2_er1 0x4000
    {
        program2.o (+RO)
    }

    LR2_er2 0x8000 0x8000
    {
        program2.0 (+RW,+ZI)
    }
}
```

    c. Save your changes.

5.7.2 See also

Tasks

- *Creating a new C or C++ project* on page 4-4
- *Adding a new source file to your project* on page 4-13
- *Importing a memory map from a BCD file* on page 5-11.

Reference

- *Preferences dialog box* on page 3-28
- *Properties dialog box* on page 3-30
- *ARM® Compiler toolchain Using the Linker*, `http://arminfo.emea.arm.com/help/topic/com.arm.doc.dui0474-`
- *ARM® Compiler toolchain Linker Reference*, `http://arminfo.emea.arm.com/help/topic/com.arm.doc.dui0493-`.

5.8 Importing a memory map from a BCD file

To import a memory map from a BCD file:

1. Select **Import** from the **File** menu.

   Select **Scatter File Editor** → **Memory from a BCD File**.

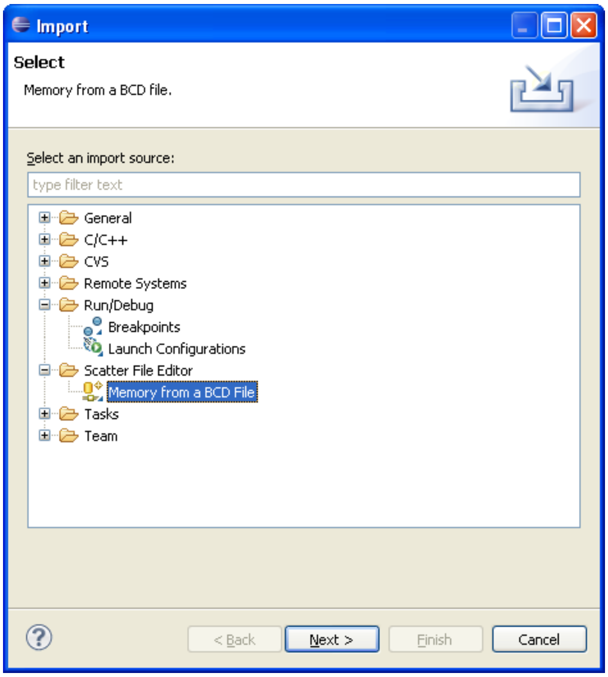
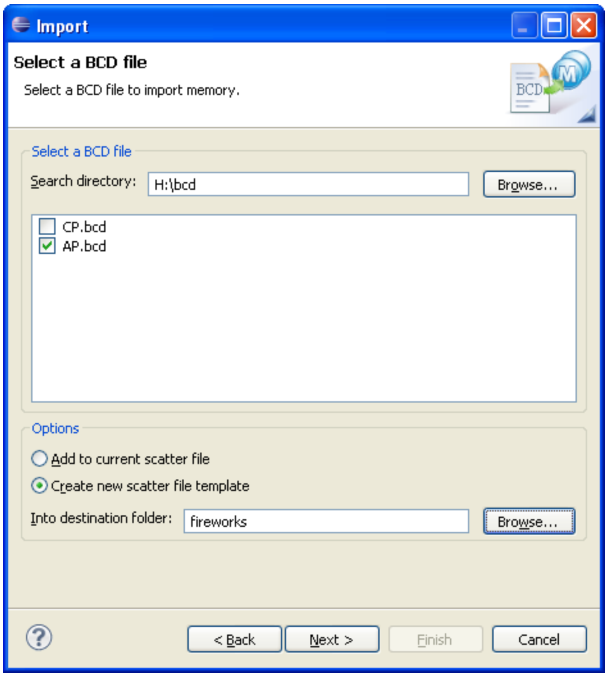


Figure 5-5 Import scatter file editor selection

2. Enter the location of the BCD file or click on **Browse...** to select the folder.

3. Select the required file containing the memory map that you want to import.



Search directory: H:\bcd

☐ CP.bcd
☑ AP.bcd

○ Add to current scatter file
● Create new scatter file template

Into destination folder: fireworks

Figure 5-6 BCD file selection for the scatter file editor

4. Select **Add to current scatter file** if you want to add specific memory regions to an existing scatter file.

   ——— **Note** ———

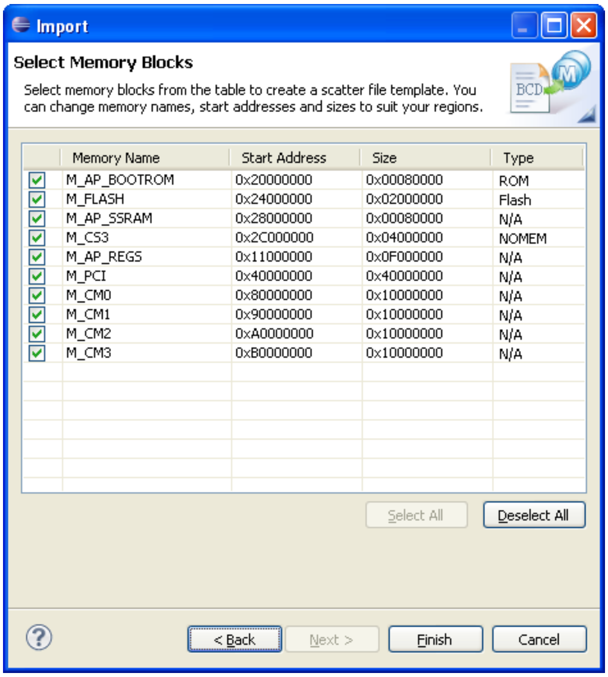
   The scatter file must be open and active in the editor view before you can use this option.

   ————————

5. Select **Create new scatter file template** if you want the wizard to create a new file with the same name as the BCD file but with a `.scat` file extension.

6. Select the destination project folder.

7. By default, all the memory regions are selected. Modify the selections and table content as required.



Select Memory Blocks

Select memory blocks from the table to create a scatter file template. You can change memory names, start addresses and sizes to suit your regions.

	Memory Name	Start Address	Size	Type
☑	M_AP_BOOTROM	0x20000000	0x00080000	ROM
☑	M_FLASH	0x24000000	0x02000000	Flash
☑	M_AP_SSRAM	0x28000000	0x00080000	N/A
☑	M_CS3	0x2C000000	0x04000000	NOMEM
☑	M_AP_REGS	0x11000000	0x0F000000	N/A
☑	M_PCI	0x40000000	0x40000000	N/A
☑	M_CM0	0x80000000	0x10000000	N/A
☑	M_CM1	0x90000000	0x10000000	N/A
☑	M_CM2	0xA0000000	0x10000000	N/A
☑	M_CM3	0xB0000000	0x10000000	N/A

Figure 5-7 Memory map selection for the scatter file editor

8. Click **Finish** to complete the scatter file.

Chapter 6
Working with Remote System Explorer

The following topics describe the Remote System Explorer and how you can use it to access resources on the host workstation and remote targets:

Concepts

- *Overview of the Remote System Explorer* on page 6-2.

Reference

- *Remote Systems view* on page 6-3
- *Remote System Details view* on page 6-4
- *Remote Scratchpad view* on page 6-5
- *Terminals view* on page 6-6.

6.1 Overview of the Remote System Explorer

The *Remote System Explorer* (RSE) enables you to:

- set up Linux SSH connections to remote targets using TCP/IP
- create, copy, delete, and rename resources
- set the read, write, and execute permissions for resources
- edit files by double-clicking to open in the C/C++ editor
- execute commands on the remote target
- view and kill running processes
- transfer files between the host workstation and remote targets
- launch terminal views.

Useful RSE views that can be added to the DS-5 Debug perspective are:

- Remote Systems
- Remote System Details
- Remote Scratchpad
- Terminals.

To add a view to the DS-5 Debug perspective:

1. Ensure that you are in the DS-5 perspective. You can change perspective by using the perspective toolbar or you can select **Window** → **Open perspective** from the main menu.
2. Select **Window** → **Show View** → **Other...** to open the Show View dialog box.
3. Select the required view from the **Remote Systems** group.
4. Click **OK**.

6.1.1 See also

Tasks

- *Using an SSH connection to set up and run Gnometris on an ARM Linux target* on page 3-8.

Reference

- *Remote Systems view* on page 6-3
- *Remote System Details view* on page 6-4
- *Remote Scratchpad view* on page 6-5
- *Terminals view* on page 6-6
- *Miscellaneous icons* on page 7-8.

6.2 Remote Systems view

This hierarchical tree view enables you to:

- set up a Linux connection to a remote target using the *Secure SHell* (SSH) protocol
- access resources on the host workstation and remote targets
- display a selected file in the C/C++ editor
- open the Remote System Details view and show the selected connection configuration details in a table
- open the Remote Monitor view and show the selected connection configuration details
- import and export the selected connection configuration details
- connect to the selected target
- delete all passwords for the selected connection
- open the Properties dialog box and display the current connection details for the selected target.

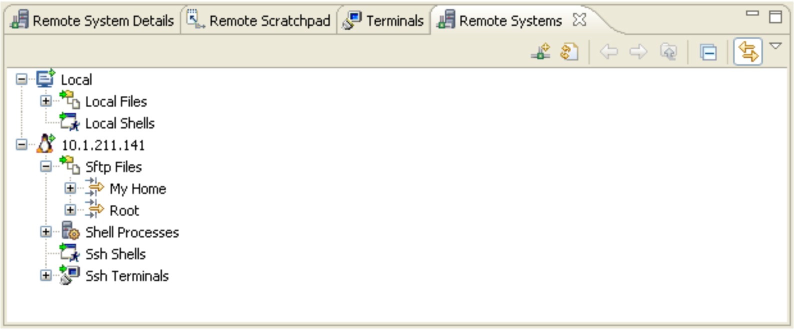


Figure 6-1 Remote Systems view

The Remote Systems view might not visible by default. To add this view:

1. Select **Window** → **Show View** → **Other...** to open the Show View dialog box.
2. Expand the **Remote Systems** group and select **Remote Systems**.
3. Click **OK**.

To configure a connection to the target, click on the toolbar icon **Define a connection to a remote system**.

6.2.1 See also

Tasks

- *Using an SSH connection to set up and run Gnometris on an ARM Linux target* on page 3-8.

Concepts

- *Overview of the Remote System Explorer* on page 6-2.

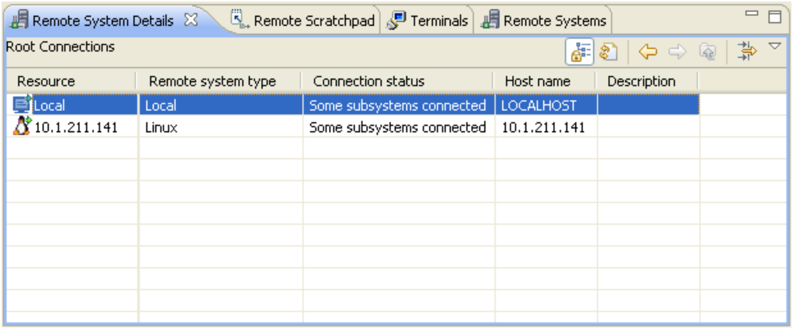
Reference

- *Remote System Details view* on page 6-4
- *Remote Scratchpad view* on page 6-5
- *Terminals view* on page 6-6
- *Miscellaneous icons* on page 7-8.

6.3 Remote System Details view

This tabular view enables you to:

- set up a Linux connection to a remote target using the *Secure SHell* (SSH) protocol
- access resources on the host workstation and remote targets
- display a selected file in the C/C++ editor
- open the Remote Systems view and show the selected connection configuration details in a hierarchical tree
- open the Remote Monitor view and show the selected connection configuration details
- import and export the selected connection configuration details
- connect to the selected target
- delete all passwords for the selected connection
- open the Properties dialog box and display the current connection details for the selected target.



Resource	Remote system type	Connection status	Host name	Description
Local	Local	Some subsystems connected	LOCALHOST	
10.1.211.141	Linux	Some subsystems connected	10.1.211.141	

Figure 6-2 Remote System Details view

The Remote System Details view is not visible by default. To add this view:

1. Select **Window** → **Show View** → **Other...** to open the Show View dialog box.
2. Expand the **Remote Systems** group and select **Remote System Details**.
3. Click **OK**.

6.3.1 See also

Tasks

- *Using an SSH connection to set up and run Gnometris on an ARM Linux target* on page 3-8.

Concepts

- *Overview of the Remote System Explorer* on page 6-2.

Reference

- *Remote Systems view* on page 6-3
- *Remote Scratchpad view* on page 6-5
- *Terminals view* on page 6-6
- *Miscellaneous icons* on page 7-8.

6.4 Remote Scratchpad view

The Remote Scratchpad view is an electronic clipboard where you can copy and paste or drag and drop useful files and folders into this view for use at a later point in time. This enables you to keep a list of resources from any connection in one place.

Note

Be aware that although the scratchpad only shows links, any changes made to a linked resource also changes it in the original file system.

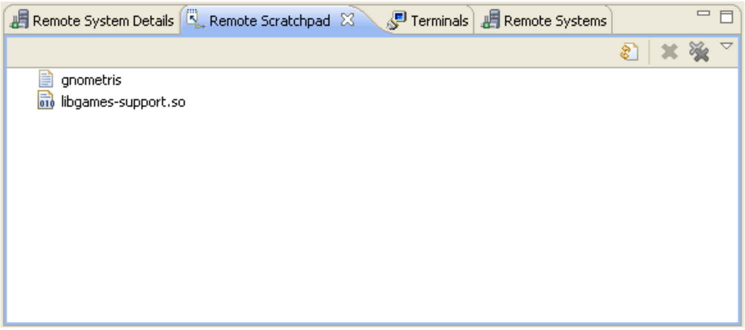


Figure 6-3 Remote Scratchpad

The Remote Scratchpad view is not visible by default. To add this view:

1. Select **Window** → **Show View** → **Other...** to open the Show View dialog box.
2. Expand the **Remote Systems** group and select **Remote Scratchpad**.
3. Click **OK**.

6.4.1 See also

Tasks

- *Using an SSH connection to set up and run Gnometris on an ARM Linux target* on page 3-8.

Concepts

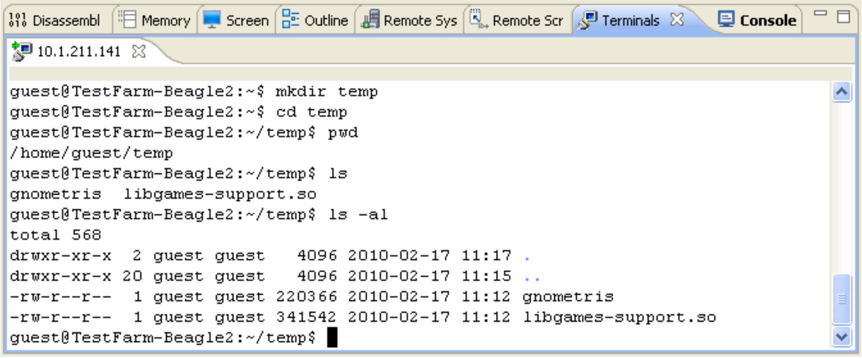
- *Overview of the Remote System Explorer* on page 6-2.

Reference

- *Remote Systems view* on page 6-3
- *Remote System Details view* on page 6-4
- *Terminals view* on page 6-6
- *Miscellaneous icons* on page 7-8.

6.5 Terminals view

The Terminals view enables you to enter shell commands directly on the target without launching any external application. For example you can browse remote files and folders by entering the `ls` or `pwd` commands in the same way as you would on a Linux terminal.



```
guest@TestFarm-Beagle2:~$ mkdir temp
guest@TestFarm-Beagle2:~$ cd temp
guest@TestFarm-Beagle2:~/temp$ pwd
/home/guest/temp
guest@TestFarm-Beagle2:~/temp$ ls
gnometris  libgames-support.so
guest@TestFarm-Beagle2:~/temp$ ls -al
total 568
drwxr-xr-x  2 guest guest   4096 2010-02-17 11:17 .
drwxr-xr-x 20 guest guest   4096 2010-02-17 11:15 ..
-rw-r--r--  1 guest guest 220366 2010-02-17 11:12 gnometris
-rw-r--r--  1 guest guest 341542 2010-02-17 11:12 libgames-support.so
guest@TestFarm-Beagle2:~/temp$ 
```

Figure 6-4 Terminals view

The Terminals view is not visible by default. To add this view:

1. Select **Window** → **Show View** → **Other...** to open the Show View dialog box.
2. Expand the **Remote Systems** group and select **Remote Systems**.
3. Click **OK**.
4. In the Remote Systems view, click on the toolbar icon **Define a connection to a remote system** and configure an *Secure SHell* (SSH) connection to the target.
5. Right-click on the connection and select **Connect** from the context menu.
6. Enter the User ID and password in the relevant fields.
7. Click **OK** to connect to the target.
8. In the Remote Systems view, right-click on **Ssh Terminals**.
9. Select **Launch Terminal** to open a terminal shell that is connected to the target.

6.5.1 See also

Tasks

- *Using an SSH connection to set up and run Gnometris on an ARM Linux target* on page 3-8.

Concepts

- *Overview of the Remote System Explorer* on page 6-2.

Reference

- *Remote Systems view* on page 6-3
- *Remote System Details view* on page 6-4
- *Remote Scratchpad view* on page 6-5
- *Miscellaneous icons* on page 7-8.

Chapter 7
Terminology, shortcuts and icons

This following topics describe some of the terminology used in *Eclipse for DS-5 Using Eclipse*, useful keyboard shortcuts, and menu and toolbar icons:

Reference

- *Terminology* on page 7-2
- *Keyboard shortcuts* on page 7-3
- *Menu and toolbar icons* on page 7-5.

7.1 Terminology

The following terminology is used in this document:

Block A small sub-division of a flash device that can be programmed.

Device A component on a target containing the application that you want to debug.

Dialog box A small page containing tabs, panels and editable fields prompting you to enter information.

Editor A view that controls the visual aspects of source code for a specific file type.

Erase A feature of a flash device where memory cells are reset to a known value.

Flash device A set of flash memory that has a single command interface.

Panel A small area in a dialog box or tab to group editable fields.

Perspective A page within the Eclipse window containing a set of related views, editors, menus, and toolbars.

Program A term used to describe the storing of data on a flash device.

Project A group of related files and folders in Eclipse.

Resource A generic term used to describe a project, file, folder, or a combination of these.

Send To A term used to describe sending a file to a target.

Tab A small overlay page containing panels and editable fields within a dialog box to group related information. Clicking on a tab brings it to the top.

Target A development platform on a printed circuit board or a software model that emulates the expected behavior of ARM® hardware.

View A small page to display related information for a specific function.

Width The smallest number of bits (8, 16 or 32) that can be natively accessed by a flash device.

Wizard A group of dialog boxs to guide you through a common tasks, for example, creating new files and projects.

Workbench A window containing perspectives, menus, and toolbars.

Workspace An area designated on your file system to store files and folders related to your projects.

7.2 Keyboard shortcuts

The most common keyboard shortcuts available for use with Eclipse are:

F3 Click on an assembler label from a branch instruction or a C/C++ calling function and press F3 to move the editor focus to the position of the selected item.

F10 Use in conjunction with the arrow keys to access the main menu.

Alt+F4 Exit Eclipse.

Alt+Left arrow

Go back in navigation history.

Alt+Right arrow

Go forward in navigation history.

Ctrl+Semicolon

Provided with the ARM® assembler editor to add comment markers to a selected block of code in the active file.

Ctrl+End Moves the editor focus to the end of the code.

Ctrl+B Build all projects in the workspace that have changed since the last build.

Ctrl+Home Moves the editor focus to the beginning of the code.

Ctrl+F Opens the Find or Find/Replace dialog box to search through the code in the active editor. Some editors are read-only and therefore disable this functionality.

Ctrl+F4 Close the active file in the editor view.

Ctrl+F6 Cycles through open files in the editor view.

Ctrl+F7 Cycles through available views.

Ctrl+F8 Cycles through available perspectives.

Ctrl+F10 Use in conjunction with the arrow keys to access the drop-down menu.

Ctrl+L Move to a specified line in the active file.

Ctrl+Q Move to the last edited position in the active file.

Ctrl+Space Provides auto-completion on selected functions in editors.

Shift+F10 use in conjunction with the arrow keys to access the context menu.

Ctrl+Shift+F

Activates the code style settings in the Preferences dialog box and apply them to the active file.

Ctrl+Shift+L

Opens a small page with a list of all keyboard shortcuts.

Ctrl+Shift+R

Opens the Open resource dialog box.

Ctrl+Shift+T

Opens the Open Type dialog box.

Ctrl+Shift+/

Provided with the C/C++ editor to add comment markers to the start and end of a selected block of code in the active file.

7.3 Menu and toolbar icons

These tables list the most common menu and toolbar icons available for use with Eclipse. For information on icons, markers, and buttons not listed in the following tables, see the standard *Workbench User Guide* or the *C/C++ Development User Guide* in the Help Contents.

If you leave the mouse pointer positioned on a toolbar icon for a few seconds without clicking, a tooltip appears informing you of the purpose of the icon.

7.3.1 Perspective icons

Table 7-1 Perspective icons

Icon	Description	Icon	Description
	Open new perspective	C	C/C++
	Fast view bar		

7.3.2 View icons

Table 7-2 View icons

Icon	Description	Icon	Description
▽	Display drop-down menu	⇆	Synchronize view contents
▭	Minimize	□	Maximize
⧉	Restore	✕	Close

7.3.3 View markers

Table 7-3 View markers

Icon	Description	Icon	Description
🔖	Bookmark	i	Information
☑	Task	⇨	Search result
⊗	Error	⚠	Warning

7.3.4 Editor icons

Table 7-4 Editor icons

Icon	Description	Icon	Description
💾	Save the active file		Save all files
🖨	Print the active file	✕	Close view
	Create new configuration		Duplicate selected configuration
✖	Delete selected configuration	⊟	Collapse configuration tree

7.3.5 Configuration icons

Table 7-5 Configuration icons

Icon	Description	Icon	Description
▶	Open Run Configurations dialog box	🐞	Open Debug Configurations dialog box

7.3.6 Outline icons

Table 7-6 Outline icons

Icon	Description	Icon	Description
⊘	Hide fields	S	Hide static members
●	Hide non-public members	↓az	Sort alphabetically
C	Class	⊞	Namespace
#	Macro definition	E	Enum
▪	Enumerator	▲	Variable
◇	Protected field	□	Private field
○	Public field	▟	Include
◆	Protected method	■	Private method
●	Public method	S	Struct
T	Type definition	U	Union
●	Function	⚙	Target configuration file
F	Include file	M	Memory
P	Peripheral	R	Register
B	Bitfield	E	Enumeration definition
L	Map rules	T	Tools information

7.3.7 Miscellaneous icons

Table 7-7 Miscellaneous icons

Icon	Description	Icon	Description
	Open a new resource wizard		Open new project wizard
	Open new folder wizard		Open new file wizard
	open search dialog box	?	Display context-sensitive help
✚L	Add load region	✚E	Add execution region
✚S	Add section	✖	Delete selected item
	Focus on active task		Displays product updates

7.3.8 Navigation icons

Table 7-8 Navigation icons

Icon	Description	Icon	Description
⇦	Navigate back	⇨	Navigate forwards
	Navigate up one level		Print active page
	Open help instruction page		Synchronize TOC with active page
✚	Bookmark active page		

7.3.9 Help Contents icons

Table 7-9 Help Contents icons

Icon	Description	Icon	Description
	Display list of all documents		Display list of documents in last search
	Display list of context-sensitive help links to related topics		Display list of all bookmarks
□	Maximize frame	❐	Restore frame
	Synchronize TOC with active page		