Migration Guide
DS-5 to Arm Development Studio
Migration Guide

Version 1.1
Migration Guide

DS-5 to Arm Development Studio Migration Guide

Copyright © 2020 Arm Limited (or its affiliates). All rights reserved.

Release Information

Document History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Confidentiality</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>02 January 2020</td>
<td>Non-Confidential</td>
<td>First PDF release</td>
</tr>
<tr>
<td>1.1</td>
<td>18 August 2020</td>
<td>Non-Confidential</td>
<td>Update for Arm® Development Studio 2020.0</td>
</tr>
</tbody>
</table>

Non-Confidential Proprietary Notice

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

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

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

This document may include technical inaccuracies or typographical errors.

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

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

If any of the provisions contained in these terms conflict with any of the provisions of any click through or signed written agreement covering this document with Arm, then the click through or signed written agreement prevails over and supersedes the conflicting provisions of these terms. This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.
The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow Arm’s trademark usage guidelines at http://www.arm.com/company/policies/trademarks.

Copyright © 2020 Arm Limited (or its affiliates). All rights reserved.


110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20349

Confidentiality Status

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

Unrestricted Access is an Arm internal classification.

Product Status

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

Web Address

http://www.arm.com
Contents

1 Overview ......................................................................................................................................................................................... 5
2 License Manager ..................................................................................................................................................................................... 6
3 Default Workspace Location .................................................................................................................................................................... 8
4 Combined C/C++ and Debug Perspectives ........................................................................................................................................ 9
5 Migrating Existing DS-5 Projects .......................................................................................................................................................... 13
6 CMSIS Packs .......................................................................................................................................................................................... 16
7 Creating a new Hardware Connection .................................................................................................................................................. 21
8 Connecting to new or custom Hardware .......................................................................................................................................... 26
9 Creating a new Linux Application Connection .................................................................................................................................. 30
10 Creating a new Model Connection .................................................................................................................................................. 33
11 Creating to new or custom Models .................................................................................................................................................... 37
12 Miscellaneous .......................................................................................................................................................................................... 41
1 Overview

This guide is to help users of DS-5 Development Studio (DS-5) migrate to using the Arm Development Studio. Many aspects of the development environments remain the same, so many features and aspects will be familiar.

This guide will cover some of the differences a user will see when working with Arm Development Studio by prompting users to perform specific activities.
2 License Manager

Unlike DS-5, before the IDE opens, if no license file information exists, Arm Development Studio prompts you to get or enter a license file, and select a tools edition. This activity describes how to add an existing license server.

Add a license to Arm Development Studio:

1. Open Arm DS IDE <version>.
   If no product license information exists for Arm Development Studio, you see the following dialog:

   ![Add License Dialog]

   Note: You can skip adding a license to Arm Development Studio by clicking Skip. If you do so, some of the Arm Development Studio IDE functionality will be unavailable. You can add a license to Arm Development Studio in the future by navigating to the Arm License Manager (Window > Preferences > Arm DS > Product Licenses). This activity assumes adding a license to Arm Development Studio is not skipped.

2. Select Add product license and click Next.

3. Under License Server, enter the license server port and address.
4. Click Next.

5. Select the appropriate Arm Development Studio edition from the provided list. **Note:** Only editions enabled by the license file will be made available.

6. Click Next and then Finish.

Changes to Arm Development Studio license(s) are done through Help > Arm License Manager or Window > Preferences > Arm DS > Product Licenses.

**Note:** Consult the Arm Compiler Licensing Configuration page for details on how to configure your Arm Compiler license.
3 Default Workspace Location

When DS-5 is launched, you are prompted to choose a workspace, unless the Prompt for workspace on startup in the Preferences view was unticked. By contrast, Arm Development Studio opens without prompting you to provide the workspace location. It automatically opens the last used workspace. When Arm Development Studio is opened for the first time, it uses a default workspace in your home directory.

The default location is:

- Windows: <user directory>\Development Studio Workspace
- Linux: <home>/Development Studio Workspace

To change to a different workspace directory in Arm Development Studio, select File > Switch Workspace.

The previous DS-5 behaviour, where you can specify a workspace on startup, can be done in Arm Development Studio by ticking Prompt for workspace on startup in Window > Preferences > General > Startup and Shutdown > Workspaces.
4 Combined C/C++ and Debug Perspectives

Arm Development Studio comes with a new IDE perspective, called Development Studio. This is the default view when Arm Development Studio opens.

This perspective combines the views of the DS-5 C/C++ and the DS-5 Debugger perspectives.

Explore the Development Studio perspective:

1. Click on the Import Project icon in the Project Explorer.
   
   Note: The Import Project icon is only present if no projects are listed in the Project Explorer view.
2. Expand **Arm Development Studio**.
   Arm Development Studio allows you to convert and import existing uVision projects. There is more information on this topic later.

   ![Image of Import wizard]

3. Click **Cancel** to exit the **Import Project** view.

4. Click **View Menu** (three line icon) at the top of the **Project Explorer** view.
   **Note:** To display menu items for the view, Arm Development Studio uses a three line icon instead of the inverted triangle used in DS-5.

   ![Image of Project Explorer]

5. Click anywhere outside the **View Menu** list to exit the menu.

6. Click the **New Debug Connection** icon in the **Debug Control** view.
   **Note:** The **New Debug Connection** icon is only present if no launch configurations exist in Arm Development Studio workspace.
   Arm Development Studio provides new methods to create Hardware, Linux Application, and Model connections. These new methods are discussed in the connection creation sections of this guide.
7. Click **Cancel** to exit the New Debug Connection view.

8. Click the **Connect with Existing Config** icon.
   
   **Note**: The Connect to Existing Config icon is only present if no launch configurations exist in Arm Development Studio workspace.
   
   The Connect with Existing Config view is empty, unless you have already created launch configurations with Arm Development Studio.

9. Click on the **Cancel** button to exit the Connect with Existing Config view.

   Click the ‘+’ to add other views to the Development Studio perspective, or select **Window > Show View** and choose your view.
Unlike DS-5, Arm Development Studio has **Builds the selected project** (hammer) and **Cleans the selected project** (broom) icons in the **Project Explorer** view to build and clean the selected project respectively.
5 Migrating Existing DS-5 Projects

Existing DS-5 projects and launch configurations (.launch files) can be imported into Arm Development Studio. The projects and launch configurations imported into Arm Development Studio are not backward-compatible with DS-5.

Import existing DS-5 projects into Arm Development Studio:

1. Choose one of the project import methods below:
   a. Click on the Import Project icon in the Project Explorer view.
      Note: The Import Project icon will only appear if no projects exist.
   b. Select File > Import... 

2. Select General > Existing Projects into Workspace.
3. Click **Next**.

4. Click **Browse**... to locate the existing DS-5 projects to import, and ensure the boxes are ticked next to the appropriate projects.

5. Click **Finish**.
6. Click **Select All** to import all the existing DS-5 projects or tick the box next to the project(s) to import.

![Migrate DS-5 Projects dialog box](image)

- Selected projects will be migrated to Arm DS projects. Migrated projects are not compatible with DS-5.

- calendar_Armv8-A_AC6
- fireworks_Cortex-A55_Cortex-A76
- Armv8-M_security

- [Select All](#)
- [Deselect All](#)

1. Click **OK**.

The imported projects will appear in the **Project Explorer** view.

**Note:** Consult the **Arm Compiler Licensing Configuration** page for details on how to configure your Arm Compiler license.
6 CMSIS Packs

Arm Development Studio includes support for CMSIS Packs. “CMSIS” stands for “Cortex Microcontroller Software Interface Standard”. The Packs offer the user a quick and easy way to create, build, and debug embedded software applications for Arm Cortex-M based microcontrollers and Cortex-A5/A7/A9 based processors.

CMSIS Packs are a delivery mechanism for software components, device parameters, and board support. A Pack is a file collection that might include:

- Source code, header files, and software libraries – for example for an RTOS, DSP libraries, or generic middleware.
- Device parameters (such as the memory layout or debug settings) along with startup code and Flash programming algorithms.
- Board support (drivers, board parameters, and debug connections).
- Documentation and source code templates.
- Example projects that show how components can be assembled into complete working systems.

Packs are developed by various silicon and software vendors, covering thousands of different boards and devices. They can also be used to enable life-cycle management of in-house software components. For more information see CMSIS-Pack Documentation.

A CMSIS Pack Manager perspective is provided in Arm Development Studio for loading and managing Packs, and the New Project wizard allows you to easily create a new project based on selected Pack(s). To create a new Pack-based project, first install the Packs needed for your target board/device from the Pack Manager, then use File > New > Project to create a new CMSIS C project. For more details see Arm Development Studio Documentation page.

Note:
If you already have some CMSIS Packs installed in a specific directory, such as the one created by MDK’s Pack Installer, the CMSIS Pack Manager can be redirected to the existing CMSIS Packs by setting Window > Preferences > CMSIS Packs > CMSIS Pack root folder to the location of the installation folder.

Work with the CMSIS Pack Manager:

1. Open the CMSIS Pack Manager perspective.
2. Install the desired CMSIS pack(s); select the device manufacturer and board in the *Devices* view, and click the appropriate *Install(+) in the Packs* view.

*Note:* Boards/devices which have CMSIS pack(s) installed are available as selectable targets when creating a *New Project* or a *Hardware Connection*. Creating a Hardware Connection with Arm Development Studio is described later.
3. Copy an example into the current Workspace; select the desired board in the **Boards** view and click the appropriate **Import** in the **Examples** view.

*Note:* Only examples for installed CMSIS Packs are visible by default. Untick **Only show examples from installed packs** to see examples from packs which have not been installed.

The pack manager shows a uVision icon if the example is a uVision project and requires conversion. This will help you to identify whether the example are affected by the limitations listed below.
Below is a set of limitations that apply to imported uVision projects:

- uVision project settings affecting debugging of the target are not migrated to Arm Development Studio debug configurations but are limited to the project build.

- uVision projects can have multiple project targets with variations in the Run Time Environment (RTE) setting. In Arm Development Studio, each project is limited to exactly one RTE configuration and Eclipse's C/C++ Development Tooling (CDT) concept of project "configurations" is not supported. Therefore, each uVision project target is imported as an individual Arm Development Studio project with its own copy of the project files.

- The conversion and import of uVision projects always results in a copy of the project files.

- uVision Multi-Project-Workspaces files (*.uvmpw) are not supported. Instead, the projects included in the workspace need to be imported individually and project interdependencies setup manually.

- uVision project targets which have a fixed version of the compiler configured cannot be imported if the specified version is not installed (see Register a compiler toolchain). This is because the compiler version is configured per project target.

- User commands in uVision projects are not converted into the corresponding Arm Development Studio Build Steps located in the project's Properties > C/C++ Build > Settings > Build Steps.

- uVision Key Sequence for Tool Parameters need to be translated to corresponding variables in Arm Development Studio.

- Signature Creator for NXP Cortex-M Devices (elfdwt) needs to be manually setup as an Arm Development Studio Post-build step in the project's Properties > C/C++ Build > Settings > Build Steps. The elfdwt utility is not included in the Arm Development Studio installation.
• The fcarm utility is not included in the Arm Development Studio installation.

• The conversion and import of uVision projects requires that all files and include folders specified in the project are in the same folder as the project file or in a subdirectory structure. Files outside of the project folder are required to be copied into the project folder and relative dependencies need to be manually resolved.

• In uVision source groups, software components and individual files can have specific assignments to memory regions which are evaluated when the linker script is generated by the tools. Since this feature is not available in Arm Development Studio, you need to manually edit the linker script file.

• uVision projects that are not contained in a CMSIS Pack can be converted and imported using File > Import > Arm Development Studio > uVision Project. The same limitations listed above apply.
7 Creating a new Hardware Connection

Arm Development Studio provides a method to connect to and debug hardware targets. Unlike DS-5, Arm Development Studio provides connection via CMSIS Packs as well as Configuration Database platform configurations.

Arm Development Studio adds a new Hardware Connection wizard to help you create connections to hardware targets. This activity describes how to connect to a hardware target with the new Hardware Connection wizard using a CMSIS Pack. Adding a new configuration database hardware platform using the same wizard is described later.

Establish a Hardware Connection:

1. Click on the **New Debug Connection** icon in the **Debug Control** view, in the **View Menu** listing of the **Debug Control** view, or at the top of the **Development Studio** perspective and select **Hardware Connection**. Note: The **New Debug Connection** icon only appears in the **Debug Control** view if no launch configurations have been used.
The Hardware Connection wizard can also be opened by selecting File > New > Hardware Connection.

2. Enter a connection name in the Debug connection name field. 
   Note: A new Hardware connection can be associated with an existing project by ticking Associate debug connection with an existing project and selecting a project from the list provided. This option is not used in this activity.

3. Click Next.

4. Select a hardware target to connect to from the available list.
   Note: The Location entry of the selected target tells you whether the device support comes from a CMSIS Pack or a
5. Click **Finish**.

   **Note:** If the selected target uses a CMSIS Pack that has not been installed, the below message appears. Confirm the pack installation by clicking **OK**.

   ![Confirm pack install dialog](image)

   If the device support for the selected target comes from a Configuration Database, the Hardware Connection wizard opens an Arm Development Studio **Edit Configuration** view which functions the same as the DS-5 **Debug Configurations** view.
If the device support for the selected target comes from a CMSIS Pack, the Arm Development Studio Edit Configuration view is different from the DS-5 Debug Configurations view. The below activity will setup and make a connection to a target with this version of the Edit Configuration view.

Setup and connect to a target using the Edit Configuration view:

1. In the Connection tab:
   a. Select a debug adapter from the Connection Type drop-down list.
   b. Select a debug connection method (JTAG or SWD) from the Debug Port drop-down list.
   c. Enter a connection address for the debug adapter or browse for the debug adapter; click Browse.
   d. Click Target Configuration... to set the trace connection options.
      
      Note: If the CMSIS-Pack does not contain trace information, Platform Configuration Editor (PCE) autodetection runs to determine the trace devices and topology. Once autodetect is complete, the Arm Development Studio Debug and Trace Services Layer (DTSL) Configuration view has the same function as DS-5's DTSL Configuration view.

2. In the Advanced tab:
   a. If appropriate, add an image file to download to the target; click Add an image under File Settings.
   b. Select the appropriate Run Control. Select and reset. Reset control. and Scripts options.
3. In the **Flash** tab, if appropriate, add a flash programming algorithm to the connection by clicking **Add a flash programming algorithm** under **Programming Algorithms**.
   **Note:** A flash programming algorithm might already be present if the CMSIS Pack includes one.

4. In the **OS Awareness** tab, if appropriate, select an OS from the **Select OS awareness** drop-down list.

5. In the **Connection** tab, click **Apply** and then click **Debug**.

The debug connection status appears in the **Debug Control** view and the created launch configuration appears in the **Project Explorer** view.
Connecting to new or custom Hardware

Like DS-5, Arm Development Studio provides a method to add new hardware target configurations for connection and debug purposes. The Arm Development Studio provides a different way to add hardware configurations to the configuration database. In DS-5, hardware configurations were added using a separate perspective, Platform Configuration Editor (PCE). In Arm Development Studio, adding a hardware configuration is part of the new Hardware Connection wizard.

Add a hardware configuration:

1. Click on the New Debug Connection icon in the Debug Control view, in the View Menu listing of the Debug Control view, or at the top of the Development Studio perspective and select Hardware Connection.
2. Enter a connection name in the Debug connection name field.
3. Click Next.
4. Click Add a new platform....

If it is not already present, a configuration database, ExtensionDB, is created automatically to store the new hardware configurations.

5. Select the appropriate debug adapter connection in the Connection Type drop-down list.

The Debug Adapter Connection view automatically discovers any debug adapters of the selected type. Unlike DS-5, Arm Development Studio can use ULINK devices for autodetection purposes.

If the debug adapter is not discovered, enter the debug adapter connection information in the Connection Address field.
6. Click **Next**.
   The hardware target autodetection process starts to create a platform configuration for the attached target. You may be prompted to update the debug adaptor firmware before the autodetection process begins. The debug adaptor firmware update process is the same as it is for DS-5.

7. When the autodetection process completes, choose the type of platform configuration to create.
To exit the Hardware Connection wizard and enter Arm Development Studio's Platform Configuration Editor (PCE) perspective, select the **Edit Platform in Arm DS Platform Configuration Editor** option. Arm Development Studio's PCE functions the same as DS-5's PCE.

Select either the **Save a Debug-only Arm DS Platform Configuration** or **Save a Debug and Trace Arm DS Platform Configuration** options to continue using the Hardware Connection wizard. This route is taken for this activity.

8. Click **Next**.

9. Enter the target information into the **Platform** fields.

10. Click **Finish**.

11. Select the new hardware configuration in the Hardware Connection **Target Selection** view.

12. Click **Finish**.

The new hardware target configuration appears in the **Edit Configuration** view.
Connecting to new or custom Hardware

Edit Configuration

Edit configuration and launch.

Name: Example New Hardware Connection

Connection

Select target
This debug configuration is associated with Imported / Example Platform. Select which debug operation to use.
Currently selected: Bare Metal Debug / Cortex-M4

- Imported
  - Example Platform
  - Bare Metal Debug
    - Cortex-M4

Target Connect: DSTREAM Family

Arm Debugger will connect to a DSTREAM to debug a bare metal application.

Connections

- Bare Metal Debug: Connection [USB:002888]
  
DTSOL Options

Configure DSTREAM trace or other target options. Using "default" configuration options

Revert  Apply

Debug  Close
9 Creating a new Linux Application Connection

Like DS-5, Arm Development Studio provides a method to connect to and debug Linux applications using gdbserver. Arm Development Studio adds a new Linux Application Connection wizard to help you create connections to a Linux application running on a target. This activity describes how to connect to a Linux application using the new Linux Application Connection wizard.

Establish a Linux Application Connection:

1. Click on the **New Debug Connection** icon in the **Debug Control** view, in the **View Menu** listing of the **Debug Control** view, or at the top of the **Development Studio** perspective and select **Linux Application Connection**.
   
   **Note:** The **New Debug Connection** icon only appears in the **Debug Control** view if no launch configurations have been used.

   ![New Debug Connection Icon](image)

   OR

   ![Create a new debug connection](image)

   OR

   ![Development Studio Workspace - Development Studio - Debug Control](image)
Creating a new Linux Application Connection

The Linux Application Connection wizard can also be opened by selecting **File > New > Linux Application Connection**.

2. Enter a connection name in the **Debug connection name** field.

   **Note**: A new Linux Application connection can be associated with an existing project by ticking **Associate debug connection with an existing project** and selecting a project from the list provided. This option is not used in this activity.

3. Click **Finish**.
The Arm Development Studio Edit Configuration view provides the same functions as the DS-5 Debug Configurations view. The main difference is that the Arm Development Studio Edit Configuration view only shows the Linux Application Debug options under the Connection Select target field.

The remaining steps to establish the Linux Application Connection are the same as they are in DS-5.

The created launch configuration will appear in the Project Explorer view.
10 Creating a new Model Connection

Like DS-5, Arm Development Studio provides a method to connect to and debug models. Arm Development Studio adds a new Model Connection wizard to help you create connections to models. This activity describes how to connect to a model that is shipped with Arm Development Studio, using the new Model Connection wizard. Adding a new model configuration using the same wizard is described later.

Establish a Model Connection:

1. Click on the New Debug Connection icon in the Debug Control view, in the View Menu listing of the Debug Control view, or at the top of the Development Studio perspective and select Model Connection. 
   
   Note: The New Debug Connection icon only appears in the Debug Control view if no launch configurations have been used.
The Model Connection wizard can also be opened by selecting File > New > Model Connection.

2. Enter a connection name in the **Debug connection name** field.
   **Note:** A new Model connection can be associated with an existing project by ticking **Associate debug connection with an existing project** and selecting a project from the list provided. This option is not used in this activity.

3. Click **Next**.

4. Select a model to connect to from the available list or click **Add a new model...** to add a new model configuration to Arm Development Studio.
Adding a new model configuration is described later.

5. Click **Finish**.

The Arm Development Studio **Edit Configuration** dialog has the same function as the DS-5 **Debug Configurations** view. The main difference is that the Arm Development Studio **Edit Configuration** view only shows the configuration details for the selected model under the **Connection Select target** field.
The remaining steps to establish the Model Connection are the same as they are in DS-5.

The created launch configuration will appear in the **Project Explorer** view.
11 Creating to new or custom Models

Like DS-5, Arm Development Studio provides a method to add new model configurations for connection and debug purposes. The Arm Development Studio provides a different way to add model configurations to the configuration database. In DS-5, model configurations are added using a separate perspective, Platform Configuration Editor (PCE). Arm Development Studio makes adding a model configuration part of the new Model Connection wizard.

Add a model configuration:

1. Click on the New Debug Connection icon in the Debug Control view, in the View Menu listing of the Debug Control view, or at the top of the Development Studio perspective and select Model Connection.

2. Enter a connection name in the Debug connection name field.

3. Click Next.

4. Click Add a new model...

5. There are two model connection methods available: Launch and connect to specific model and Browse for model running on local host.
These are the same model connecting methods provided by DS-5. The `Browse for model running on local host` method is selected for illustrative purposes.

6. Click **Next**.

7. Browse for the model by clicking **Browse...**

8. Click **Finish**.

If it is not already present, a configuration database, `ExtensionDB`, is created automatically to store the new model configuration. A new Model configuration appears in the `Model Connection Target Selection` view.
9. Click Finish.

The new model configuration will appear in the Edit Configuration view.
Migration Guide DS-5 to Arm Development Studio Migration Guide

Creating to new or custom Models

Copyright © 2020 Arm Limited (or its affiliates). All rights reserved.
Non-Confidential
12 Miscellaneous

This section covers miscellaneous small differences between DS-5 and Arm Development Studio.

- The Linaro GCC 4.8-2014.04 [arm-linux-gnueabihf] compiler is not supplied. The old Linaro Linux GCC 4.8 previously supplied in DS-5 has been removed from Arm Development Studio. To use GCC (Linux or bare-metal) within Arm Development Studio, download the version you require from e.g. Linaro or Arm Developer, then add it as a toolchain (see Register a compiler toolchain).

- The Arm Development Studio IDE executable in the installation's bin directory is armds_ide. In DS-5, the IDE executable is eclipse.

- Arm Development Studio’s default Run control option is Connect only in the Edit Configuration view’s Debugger tab. DS-5’s default Run control option is Debug from symbol set to main.

- The command-line debugger is renamed from debugger in DS-5 to armdbg in Arm Development Studio.

- Unlike DS-5, the Platform Configuration Editor (PCE) is integrated into the new Arm Development Studio Connection Wizard (see Connecting to new or custom Hardware). In DS-5, PCE is a separate “Configuration” perspective.

- Ensure Properties > C/C++ Build > Environment > Append variables to native environment is selected for any project. If Replace native environment with specified one is selected instead, the project might not build successfully.