ARM® Cortex®-M3 DesignStart™ Eval

Revision: r0p0

RTL and FPGA Quick Start Guide

ARM

ARM 100895_0000_00_en

ARM® Cortex®-M3 DesignStart™ Eval

RTL and FPGA Quick Start Guide

Release Information

Document History

Issue	Date	Confidentiality	Change
0000-00	14 June 2017	Non-Confidential	First release for r0p0

Non-Confidential Proprietary Notice

ARM Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20349

Confidentiality Status

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

Unrestricted Access is an ARM internal classification.

Product Status

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

Web Address

http://www.arm.com

Contents

ARM® Cortex®-M3 DesignStart™ Eval RTL and FPGA Quick Start Guide

Preface

This preface introduces the *ARM® Cortex®-M3 DesignStart™ Eval RTL and FPGA Quick Start Guide*.

It contains the following:

About this book

This book describes how to simulate a basic test in Cortex®-M3 DesignStart™ Eval, and load the design onto the *Versatile™ Express Cortex-M Prototyping System* (V2M-MPS2+) platform.

Product revision status

The r*m*p*n* identifier indicates the revision status of the product described in this book, for example, r*1*p*2*, where:

r*m* Identifies the major revision of the product, for example, r1.

p*n* Identifies the minor revision or modification status of the product, for example, p2.

Intended audience

This book is written for hardware engineers, software engineers, system integrators, and system designers, who might not have previous experience of ARM products, but want to run a complete example of a working Cortex®-M3 DesignStart™ Eval system.

Using this book

This book is organized into the following chapters:

Chapter 1 Introduction

This chapter introduces Cortex-M3 DesignStart Eval and describes the scope of this document.

Chapter 2 Quick start instructions

This chapter describes how to simulate the Cortex-M3 DesignStart Eval RTL with the processor cycle model, use the design with the MPS2+ platform, and use the mbed online toolchain to compile tests.

Glossary

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

See the *ARM® Glossary* for more information.

Typographic conventions

italic

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

bold

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

`monospace`

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

`monospace`

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

`monospace italic`

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

`monospace bold`

Denotes language keywords when used outside example code.

`<and>`

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

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

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

Timing diagrams

The following figure explains the components used in timing diagrams. Variations, when they occur, have clear labels. You must not assume any timing information that is not explicit in the diagrams.

Shaded bus and signal areas are undefined, so the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.

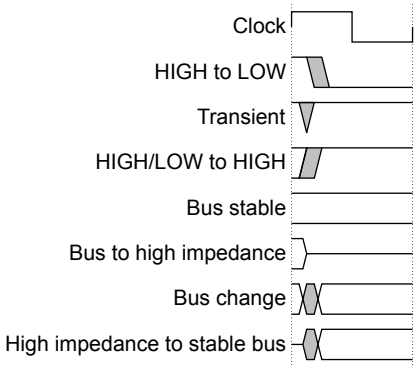

Figure 1 Key to timing diagram conventions

Signals

The signal conventions are:

Signal level
: The level of an asserted signal depends on whether the signal is active-HIGH or active-LOW. Asserted means:
- HIGH for active-HIGH signals.
- LOW for active-LOW signals.

Lowercase n
: At the start or end of a signal name denotes an active-LOW signal.

Additional reading

This book contains information that is specific to this product. See the following documents for other relevant information.

ARM publications

- Cortex-M3 DesignStart Eval publications:
  - *ARM® Cortex®-M3 DesignStart™ Eval RTL and Testbench User Guide* (ARM 100894).
  - *ARM® Cortex®-M3 DesignStart™ Eval FPGA User Guide* (ARM 100896).
  - *ARM® Cortex®-M3 DesignStart™ Eval Customization Guide* (ARM 100897).
- Other ARM publications:
  - *ARM® Cortex®-M System Design Kit Technical Reference Manual* (ARM DDI0479).
  - *ARM® TrustZone® TRNG True Random Number Generator Technical Reference Manual* (ARM 1009676).
  - *ARM® PrimeCell™ Real Time Clock (PL031) Technical Reference Manual* (ARM DDI 0224).
  - *ARM® PrimeCell® Synchronous Serial Port (PL022) Technical Reference Manual* (ARM DDI 0194).
  - *ARM® Versatile™ Express Cortex®-M Prototyping System (V2M-MPS2 and V2M-MPS2+) Technical Reference Manual* (ARM 100112).
  - *Application Note AN531 uSDCARD SPI Adapter for the Cortex-M Prototyping System (MPS2+)* (ARM DAI 0531).
  - *Application Note AN502 Adapter for Arduino for the Cortex-M Prototyping System (MPS2 and MPS2+)* (ARM DAI 0502).
  - *ARM® AMBA® 3 AHB-Lite Protocol Specification (v1.0)* (ARM IHI 0033).
  - *ARM® Architecture Reference Manual ARMv7, for ARMv7-M architecture profile* (ARM DDI0403).
  - *ARM® Cortex®-M3 Technical Reference Manual* (ARM 100165).
  - *ARM® Cortex®-M3 Devices Generic User Guide* (ARM DUI0552).

Other publications

None.

Feedback

Feedback on this product

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

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

Feedback on content

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

- The title *ARM Cortex-M3 DesignStart Eval RTL and FPGA Quick Start Guide*.
- The number ARM 100895_0000_00_en.
- If applicable, the page number(s) to which your comments refer.
- A concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

Note

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

Chapter 1
Introduction

This chapter introduces Cortex-M3 DesignStart Eval and describes the scope of this document.

It contains the following sections:

1.1 What is Cortex®-M3 DesignStart™ Eval?

Cortex-M3 DesignStart Eval allows developers to easily develop and simulate SoC designs based on the ARM Cortex-M3 processor, and then perform hardware prototyping using an ARM *Versatile Express Cortex-M Prototyping System* (V2M-MPS2+) platform.

Cortex-M3 DesignStart Eval provides an example system that is built around the CoreLink™ SSE-050 subsystem, and the Cortex-M3 processor.

The example system includes all the standard components and peripherals for implementing a functioning ARM Cortex-M3 mbed™ OS endpoint device.

A fixed and preconfigured Cortex-M3 processor is delivered as obfuscated RTL, and a Cycle Model. Cortex-M3 DesignStart Eval does not support changing the configuration of the processor.

The obfuscated RTL is used to build an FPGA image when the Cortex-M3 DesignStart Eval system has been modified by the user. The obfuscated RTL only exposes a limited set of internal registers for debug purposes.

The processor Cycle Model is recommended for simulation and debug purposes. It includes visibility of the internal processor architectural registers.

The Cortex-M3 DesignStart Eval package includes:

The RTL

- Preconfigured Cortex-M3 processor (obfuscated but synthesizable).
- Modified CoreLink SSE-050 subsystem with debug and trace support.
- Memory subsystem.
- Peripherals for application use.
- Two timers for OS usage, and SPI interface.
- Reusable ARM *Advanced Microcontroller Bus Architectures* (AMBA) components.

Execution Testbench

- Processor Cycle Model with register visibility and execution tracing.
- Memory models that match the FPGA target.
- ARM CoreSight™ debug test engine preconfigured for single fixed debug and trace implementation.
- Integration tests for memories and internal peripherals.

FPGA Evaluation Flow

- FPGA synthesis of the simulation system that you can build and customize.
- Requires purchase of the MPS2+ platform.

For more details on how you are permitted to use these Cortex-M3 DesignStart Eval deliverables, see the *ARM® Cortex®-M3 DesignStart™ Eval RTL and Testbench User Guide*.

1.2 Scope of this document

This document demonstrates the basic steps to use the Cortex-M3 DesignStart Eval system. It helps you to learn about the different parts of the flow by:

1. Running the RTL simulation using the Cycle Model of the Cortex-M3 processor in Cortex-M3 DesignStart Eval.
2. Using Cortex-M3 DesignStart Eval with the MPS2+ platform.
3. Compiling code using the ARM mbed online toolchain.

This document only provides quick start instructions and does not cover each topic in detail. For detailed information about:

- System design and RTL simulation, see the *ARM® Cortex®-M3 DesignStart™ Eval RTL and Testbench User Guide*.
- Building your own FPGA image for the MPS2+ platform, see the *ARM® Cortex®-M3 DesignStart™ Eval FPGA User Guide*.
- Customization and integrating your own peripherals, see the *ARM® Cortex®-M3 DesignStart™ Eval Customization Guide*.

Chapter 2
Quick start instructions

This chapter describes how to simulate the Cortex-M3 DesignStart Eval RTL with the processor cycle model, use the design with the MPS2+ platform, and use the mbed online toolchain to compile tests.

It contains the following sections:

2.1 Getting started

This section describes how to get started with your Cortex-M3 DesignStart Eval bundle, and the required installations and additional licenses.

2.1.1 Your Cortex®-M3 DesignStart™ Eval bundle

After completing the registration and downloading the Cortex-M3 DesignStart Eval bundle, unpack the bundle in your preferred installation directory (`<install_directory>`).

2.1.2 Installation and required licenses

Cortex-M3 DesignStart Eval requires you to install some tools for simulation. There are also optional tools and licenses that you can use, depending on the purpose of your task.

The minimum tools that you require to use Cortex-M3 DesignStart Eval are:

Simulation flow

Install all of the following:

- GNU Make version 3.80.
- GNU GCC version 4.7.2 or 4.8.3.
- GNU Binutils version 2.22.

Verilog simulator

Install one of the following:

- Mentor QuestaSim version 10.4e_1. You are required to set the environment variable `QUESTASIM_HOME` to reference your QuestaSim installation directory.
- Cadence IUS version 15.20.008.
- Synopsys VCS version 2016.06-SP2.

In addition to the minimum tools required, you can use:

ARM Cycle Model license

Only required if you are using the processor Cycle Model with your simulator. You can obtain the license from the ARM website using the information provided when you registered for access to the DesignStart product.

You must have a software license server available on your network.

ARM Cycle Model products use the FLEXNet License Manager and you can either install a *node locked* license or a *floating* license. For more information, see the following FAQ topics in the *ARM Technical Support Knowledge Articles*:

- *How do I install my node locked license?*
- *How do I install my floating license?*

Note

The RTL simulation that is demonstrated in this document uses the processor Cycle Model and therefore requires the ARM Cycle Model license.

C compiler

Only required if you want to modify or recompile the integration tests that are provided with Cortex-M3 DesignStart Eval. Install a minimum version of one of the following:

- ARM Keil® *Microcontroller Development Kit* (MDK) version 5.22.

  Note

  A limited term license for Keil MDK is provided with Cortex-M3 DesignStart Eval.

- *GNU Tools for* ARM *Embedded Processors* (ARM GCC) version 5-2016q2.
- ARM *Development Studio 5* (DS-5) version 5.06.409.

Intel Quartus

Only required if you want to build your own user bit file to load onto the MPS2+ FPGA platform. You can use the free Lite Edition of Quartus Prime. If you want more advanced features, you can install Intel Quartus Prime Standard Edition version 16.1 onwards.

Debugger software and license

Only required for debugging the test code running on the MPS2+ platform.

The Keil MDK includes the ARM Keil µVision® debugger software.

A full license of the Keil MDK is required to compile the test code. However, the debugger can be used with the evaluation version of MDK.

For more information, see *http://www.keil.com/mdk*.

2.2 Simulation

When running a simulation, you can either use the obfuscated RTL or the Cycle Model of the preconfigured processor in Cortex-M3 DesignStart Eval.

ARM recommends that you use the processor Cycle Model for simulation because this is easier to debug the design with.

Simulating the example system with the processor Cycle Model includes:

1. *2.2.1 Setting up the Cycle Model* on page 2-16
2. *2.2.2 Compiling the RTL* on page 2-16.
3. *2.2.3 Running the simulation* on page 2-17.

For more information on the example system, see the *ARM® Cortex®-M3 DesignStart™ Eval RTL and Testbench User Guide*.

2.2.1 Setting up the Cycle Model

To use the Cycle Model, obtain an ARM Cycle Model license file from the ARM website by following the instructions provided when you download the Cortex-M3 DesignStart Eval bundle.

By default, the Cycle Model is not used. To use it, you must specify `DSM=yes` in the simulation `make` commands, as shown in *2.2.2 Compiling the RTL* on page 2-16 and *2.2.3 Running the simulation* on page 2-17.

2.2.2 Compiling the RTL

To compile the RTL, follow these steps:

1. Navigate to the following directory:

   ```
   <install_dir>/m3designstart/logical/testbench/execution_tb
   ```

2. Clean any previous RTL compile by executing the following command:

   ```
   make clean
   ```

3. Compile the RTL with the processor Cycle Model (`DSM=yes`), and specify the simulator that you are using with one of the following options:

`mti`	Mentor QuestaSim
`ius`	Cadence IUS
`vcs`	Synopsys VCS

   For example, to compile using Synopsys VCS, execute the following command:

   ```
   make compile SIMULATOR=vcs DSM=yes
   ```

The compile log is written to `<sim>_compile.log`.

If the compilation is successful, then the following message is displayed:

```
>> Testbench compile with vcs and DSM=yes completed successfully, log in vcs_compile.log
```

If the compilation fails, then the following is displayed:

```
>> ERROR: Testbench compile failed, check vcs_compile.log
```

Note

If you want to compile the processor obfuscated RTL instead of the Cycle Model, then use `DSM=no`.

2.2.3 Running the simulation

Cortex-M3 DesignStart Eval provides a precompiled binary file for each integration test that you can run during simulation. These files can be used out of the box without any special configuration.

To run the simulation, execute the following command:

1. Ensure that you are in the following directory:

   ```
   <install_directory>/m3designstart/logical/testbench/execution_tb
   ```

2. Run the simulation with an integration test. For example, to start the simulation with the `hello` test, execute the following command:

   ```
   make run TESTNAME=hello SIMULATOR=vcs DSM=yes
   ```

Note

If values of any of the configuration options `SIMULATOR`, `DSM`, or `SIM_64BIT` were specified in the previous `make compile` command, then the same set of values must be used for the `make run` command.

For a complete list of the integration tests and the full set of makefile options, see the *ARM® Cortex®-M3 DesignStart™ Eval RTL and Testbench User Guide*.

Running the tests only takes a few minutes. As delivered, all tests should pass with the following display message:

```
** TEST PASSED **
```

If any of the tests fails to display the `** TEST PASSED **` message, then rerun the `hello` test since it is the most basic test that can help identify the problem. You can also refer to the `<sim>_<testname>_run.log` file in the `execution_tb` directory for troubleshooting.

For more suggestions on troubleshooting or how to recompile the integration tests, see the *ARM® Cortex®-M3 DesignStart™ Eval RTL and Testbench User Guide*.

2.3 Using Cortex®-M3 DesignStart™ Eval on the MPS2+ FPGA platform

Cortex-M3 DesignStart Eval is designed to be used on the *Versatile Express Cortex-M Prototyping System* (V2M-MPS2+) FPGA platform. ARM supplies an encrypted prebuilt image file of the Cortex-M3 DesignStart Eval example system, which you can load onto the MPS2+ FPGA platform.

Note

The MPS2+ platform is not included with the Cortex-M3 DesignStart Eval package, and requires a separate purchase. If you already have an MPS2+ platform and want to use the mbed OS, then you will need the uSDCARD SPI adapter, which is also available for purchase. For further information on this platform and how to purchase, visit *www.arm.com/mps*.

2.3.1 Setting up your MPS2+ platform

To set up your MPS2+ platform, follow these steps:

1. Connect the 12V power adapter to the 12V DC power jack on the MPS2+ platform.
2. Connect a USB lead from your computer to the USB 2.0 port on the MPS2+ platform. Your computer should recognize the MPS2+ platform as an external USB drive, named `V2M_MPS2`.

   Note

   Optionally, you can use the supplied RS232 to USB converter to connect an RS232 connector to the UART General Purpose connector. This connector allows for terminal access to the self-test program. If you want to run the self-test program, see the *ARM® Cortex®-M3 DesignStart™ Eval FPGA User Guide*.

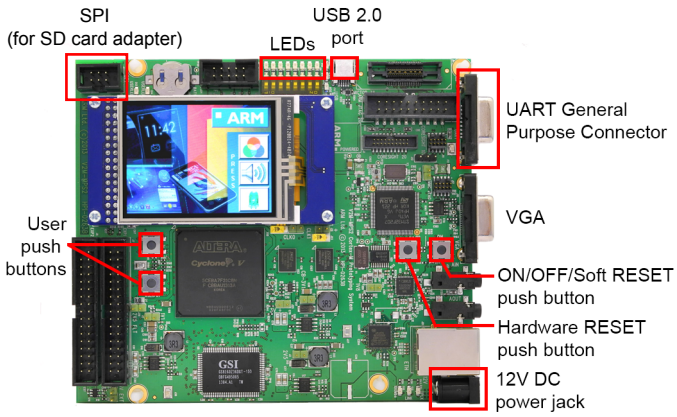

Figure 2-1 MPS2+ platform

For more information on the MPS2+ platform specifications, see the *ARM® Versatile™ Express Cortex®-M Prototyping System (V2M-MPS2 and V2M-MPS2+) Technical Reference Manual*.

2.3.2 Loading a prebuilt image onto the MPS2+ platform

To load the prebuilt Cortex-M3 DesignStart Eval image, follow these steps:

1. Copy all the contents in the following directory and paste them in the root directory of the attached `V2M_MPS2` drive:

```
<install_dir>/boards/Recovery
```

Note

You might want to manually modify and merge the contents for certain configuration files. Alternatively, you can restore the existing configuration files from the `/Recovery` directory. The affected configuration files are:

- `<install_dir>/boards/Recovery/config.txt`
- `<install_dir>/boards/Recovery/MB/HBI0263C/board.txt`

2. Eject the `V2M_MPS2` volume from your computer to unmount the drive.
3. Power up the MPS2+ platform using the ON/OFF/Soft RESET push button shown in *Figure 2-1 MPS2+ platform* on page 2-18. The LEDs flash in sequence to indicate that the new BIOS is being downloaded (this only occurs the first time when the BIOS is updated), and that the prebuilt image is being loaded onto the platform. Then, the color LCD touch screen shows an 'ARM IoT Subsystem for Cortex-M' splash screen. Simultaneously, if you have configured the UART to run a self-test program, then the debug UART terminal shows the self-test menu for Application Note AN511.

If you want to run the self-test program, see the *ARM® Cortex®-M3 DesignStart™ Eval FPGA User Guide*.

2.3.3 Connecting to a debugger

Cortex-M3 DesignStart Eval supports debug through the CMSIS-DAP protocol, using the MPS2+ platform in-built USB port.

To connect to a debugger, follow these steps:

1. Ensure that you have installed ARM Keil µVision debugger software. For more information, see *2.1.2 Installation and required licenses* on page 2-14.
2. Connect your computer to the USB 2.0 port on the MPS2+ platform.
3. After loading the prebuilt image onto the platform (see *2.3.2 Loading a prebuilt image onto the MPS2+ platform* on page 2-18), navigate to:

```
<install_directory>/m3designstart/software/self_test/Build_keil
```

4. Open the following file with the Keil µVision debugger:

```
selftest_mpb.uvprojx
```

5. Select **Accept** in the prompts to install legacy support in Keil µVision. This is only necessary if you want to use the self-test software. You can also develop your own software using 'Packs' as normal.

The following steps are optional, but they confirm whether the debug connection is valid:

1. When the project file opens in Keil µVision debugger, select **Flash** > **Configure Flash Tools**. This gives you the following dialog:

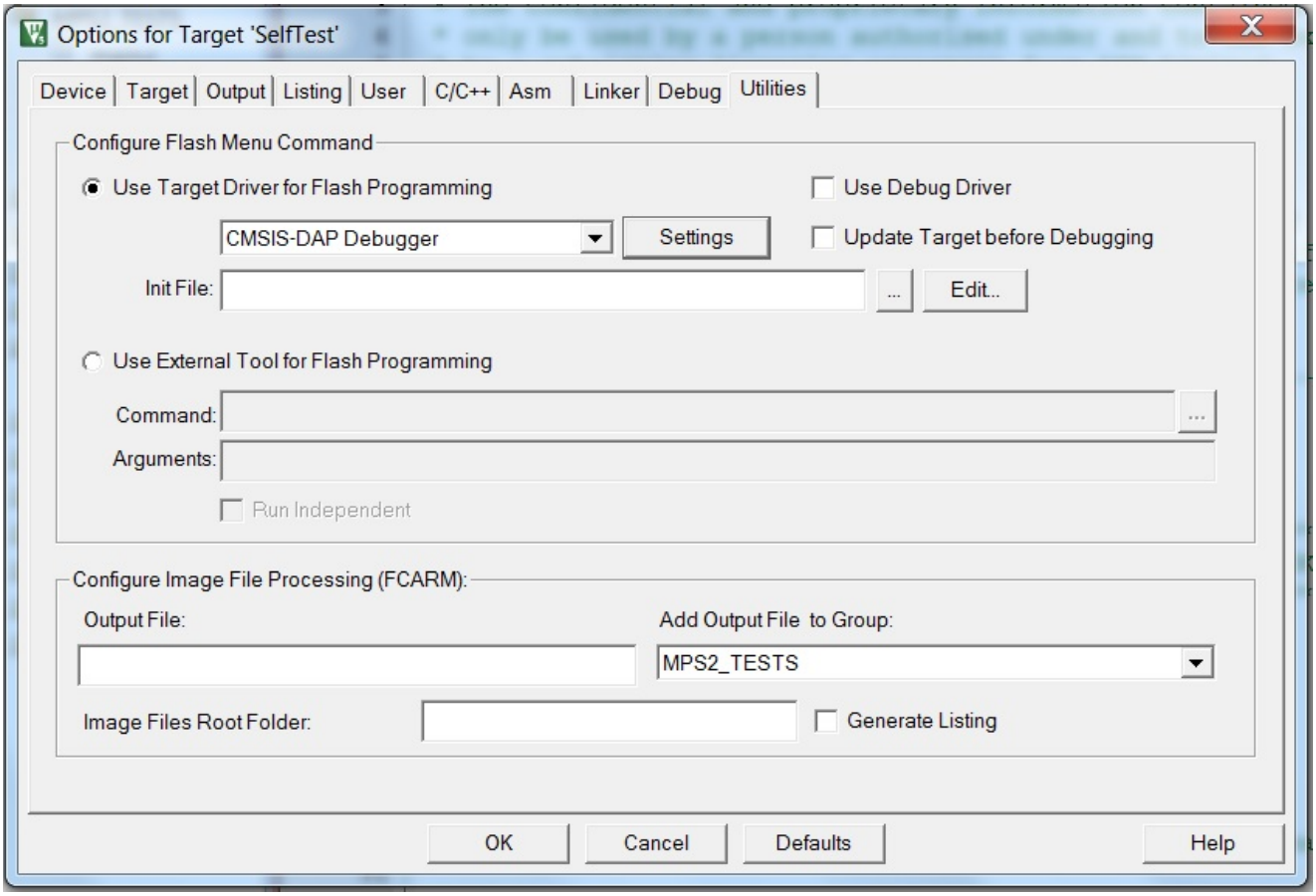

Figure 2-2 Keil µVision debugger options

2. Select **Use Target Driver for Flash Programming**.
3. Select **Settings** > **Debug**. This gives you the following dialog:

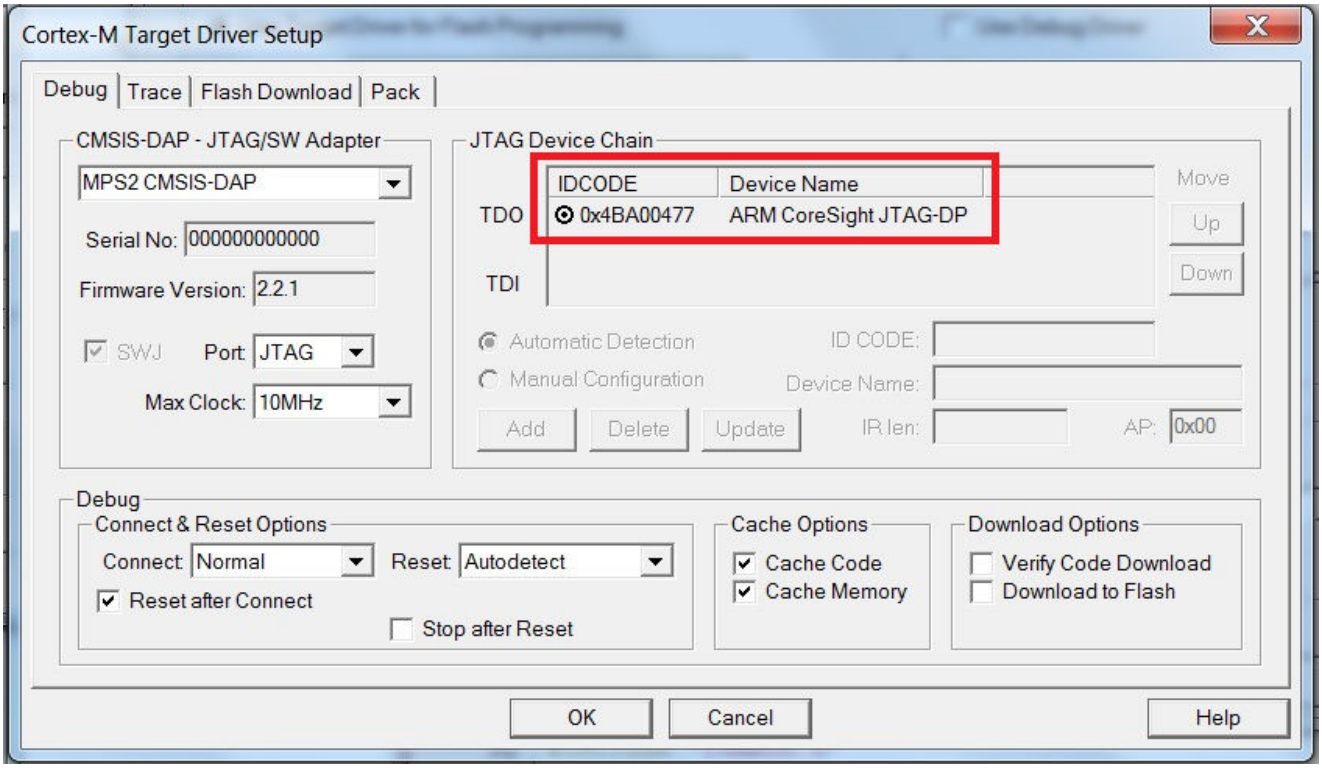

Figure 2-3 Debug options

4. Ensure that the IDCODE and Device Name is the same as shown in *Figure 2-3 Debug options on page 2-21*.
5. Select **OK** > **OK**.

The self-test code can now be dynamically debugged using the **Start/Stop Debug Session** on the toolbar.

2.3.4 Building your own image file

If you have your own RTL design modifications, then you are able to build your own user bit file using the free Lite Edition of Quartus Prime or Intel Quartus version 16.1 onwards.

For more detailed information on how to build an FPGA bit file, and load the bit file onto the MPS2+ platform, see the *ARM® Cortex®-M3 DesignStart™ Eval FPGA User Guide*.

2.4 Compiling code using ARM mbed™ online toolchain

The mbed online toolchain allows you to develop applications using the mbed OS.

Before using the mbed online compiler:

1. Go to the mbed platform site at *https://developer.mbed.org/platforms/*.
2. Search and select **ARM Cortex-M3 DesignStart**.
3. In the ARM Cortex-M3 DesignStart platform site, select **Add to your mbed Compiler**. Alternatively, open the `<install_directory>/boards/Recovery/mbed.htm` file in a web browser.

To compile and run a basic LED blinking program on the MPS2+ platform using the online mbed compiler, follow these steps:

1. Go to the mbed compiler site at *http://developer.mbed.org/compiler/*.
2. Import the program and libraries into your workspace by selecting **New program**. This gives a dialog for you to select the following:

Platform	ARM Cortex-M3 DesignStart
Template	Blinky LED Hello World
Program name	`mbed_blinky`

3. Select **Compile** to compile the program and produce a `.bin` file, which will be downloaded automatically.
4. Rename the `.bin` file to follow an 8:3 character format (for example, `mbed_bl.bin`).
5. Ensure that your MPS2+ platform is powered up and connected to the computer. Your computer should recognize the MPS2+ platform as an external USB drive, named `V2M_MPS2`.
6. Copy the `.bin` file to the following MPS2+ platform drive:

   ```
   V2M_MPS2:\SOFTWARE
   ```

7. Edit the following file to reference the `.bin` file.

   ```
   V2M_MPS2:\MB\HBI0263C\AN511\images.txt
   ```

   For example:

   ```
   TITLE: Versatile Express Images Configuration File

   [IMAGES]
   TOTALIMAGES: 1                    ;Number of Images (Max: 32)

   IMAGE0ADDRESS: 0x00000000         ;Please select the required executable program
   ;IMAGE0FILE: \SOFTWARE\st_m3ds.axf ; - M3 DesignStart selftest
   IMAGE0FILE: \SOFTWARE\mbed_bl.bin  ; Compiled on mbed.org
   ```

Note

If you are using the mbed command-line interface, use **cm3ds_mps2** as the platform name.