Using the ARM Fast Model-based Cortex-M33 IoT Kit FVP



MDK Version 5

AN302, Summer 2017, V 1.1

Ken Havens

ken.havens@arm.com

Contents

Introduction	1
Prerequisites	1
Using the ARM Fast Model based Cortex-M33 IoT Kit FVP with MDK	1
Verify the Pack Installation	1
Copy, Update the Target and Run the Example Application	3
Adding the ARM Fast Model based Cortex-M33 IoT Kit FVP as a debug target	4
Let's build the projects and debug the FVP target	7
Some Notes	11

Introduction

This document describes a step by step process on how to use the ARM Fast Model based Cortex-M33 IoT Kit FVP with the MDK toolchain. While we've tested these implementations, there will be updates to the tools and FPGA images. Expect differences between these and follow-on implementations.

Prerequisites

All the ARMv8-M support is now in our standard MDK product as of version 5.23 and later. Just perform the normal MDK installation.

To build and run the examples, you'll need the "CMSIS.5.0.1 (2017-02-03)" pack and "V2M-MPS2_IOTKit_BSP 1.3.0 (2017-03-10) packs available via the pack installer in MDK. These packs have the latest support for the MPS2+ (V2M-MPS2-0318C) board running the Cortex-M33 IoT FPGA image.

The example project introduced in this document requires the Windows built-in telnet.exe to be activated. For some Windows version, such as Windows 10, telnet.exe is not activated by default. After enabling the Windows built-in telnet.exe, it can be found via the path C:\Windows\System32\telnet.exe.

Using the ARM Fast Model based Cortex-M33 IoT Kit FVP with MDK

Verify the Pack Installation

1. Let's double check that the CMSIS.5.0.1" and "V2M-MPS2_IOTKit_BSP 1.3.0" packs are both installed properly. We'll use the pack installer from MDK. Click on the pack installer icon from the icon bar...



2. Then make sure you have the latest packs loaded, CMSIS.5.0.1-dev5.pack and V2M-MPS2_IOTKit_BSP.1.3.0.pack packs by checking the versions listed and that the "Up to date" button is shown next to each one...

Image: Packs Examples		
Pack	Action	Description
E. Pce Specific	1 Pack	IOTKit_CM33_FP selected
E Keil::V2M-MPS2_IOTKit_BSP	💠 Up to date	AF M V2M-MPS2 Device Family Pack for IOT-Kit devices
1.3.0 (2017-03-10)	💥 Remove	ARM V2M-MPS2 Device Family Pack for IOT-Kit devices
	🔉 Remove	ARM V2M-MPS2 Device Family Pack for IOT-Kit devices
1.1.0 (2017-01-12)	💥 Remove	ARM V2M-MPS2 Device Family Pack for IOT-Kit devices
+Previous		Keil::V2M-MPS2_IOTKit_BSP - Previous Pack Versions
Generic	10 Parts	
- ARM::CMSIS	💠 Up to date	CivelS (Cortex Microcontroller Software Interface Standard)
5.0.1 (2017-02-03)	💥 Remove	Childer Standard)
	Remove	CMSIS (Cortex Microcontroller Software Interface Standard)
4.5.0 (2015-10-28)	💥 Remove	CMSIS (Cortex Microcontroller Software Interface Standard)
ARM::CMSIS-Driver Validation	♦ Install	CMSIS-Driver Validation

Copy, Update the Target and Run the Example Application

Once you have these packs loaded you can go to the examples tab and export the latest "TrustZone for ARMv8-M RTOS" (μ Vision Simulator)" example...

- 1. On the left side, click the "Devices" tab, then select "ARM" then "ARM Cortex-M33" then "IOTKit_CM33" as the device.
- 2. On the right side, switch to the "Examples" tab. If needed, scroll down till you see the "IOT-Kit CM33 Secure/Non-Secure (V2M-MPS2 (IoT))" example...



Click the "Copy" button next to the "IOT-Kit CM33 Secure/Non-Secure (V2M-MPS2 (IoT))" example project...

Example	Action	Description
IOT-Kit ARMv8MBL Secure/Non-Secure (V2M	🚸 Сору 🛛 🗍	IOT-Kit ARMv8MBL Secure/Non-Secure example
IOT-Kit ARMv8MML Blinky (V2M-MPS2 (IoT))	💠 Сору 🛛	Blinky example
IOT-Kit ARMv8MML Secure/Non-Secure (V2	💠 Copy 🛛	IOT-Kit ARMv8MML Secure/Non-Secure example
IOT-Kit CM23 Secure/Non-Secure (V2M-MPS	🗢 Сору 🛛 🗍	IOT-Kit CM23 Secure/Non-Secure example
IOT-Kit CM33 Secure/Non-Secure (V2M-10PS	💠 Сору 🛛	IOT Xit CM33 Secure/Non-Secure example
	Board: V2M-I Device: IOTKit Pack: Keil::V2 ARM V2M-ME	MPS2 (IoT) (ARM) _ARMv8MML (ARM) M-MPS2_IOTKit_BSP, Version 1.2.0 (2017-02-08) 252 Device Family Park for IOT.Kit devices

3. ... for running the example application, it is more convenient to check the box to 'Launch μ Vision' with the copied project (Note the destination path)...

C	opy Example		×
	Destination Folder		Browse
	✓ Use Pack Folder Structure	✓ Launch µVision	
		ОК	Cancel

4. Close the Pack Installer and, if you didn't select the 'Launch µVision' check box, open the newly downloaded project in MDK by double clicking the "IOTKit_CM33_s_ns.uvmpw" multi-project file in the folder located in the following directory path noted above...
 <load point>\proj\Boards\ARM\V2M-MPS2\IOTKit_CM33\IOTKit_CM33_S_NS\

Adding the ARM Fast Model based Cortex-M33 IoT Kit FVP as a debug target

1. First, check to make sure secure project, "IOTKIT_CM33_s", is the current active project by right clicking the project name and selecting "Set as Active Project" button...



2. To add additional targets to the project, click the "Manage Project Items" icon near the top of the μVision window...



3. In the "Manage Project Items" dialog box, add a new target by clicking the "New (Insert)" button...

Manage Project Items		
Project Items Folders	/Extensions Book	s
Project Targets:		Groups:
V2M-MPS2+		Source Group 1

4. This will add a blank line to fill in the name of the new target. In the example below I typed "FVP" then enter. Once the new name is highlighted, click the "Set as Current Target" button at the bottom of the dialog and then the "OK" button...



5. Now the FVP target should be seen as the current target in the project tab as below. As the MPS2 settings were copied over when we created the new target, we need to update the debug settings to accommodate using the FVP. Right click on the "FVP" target in the Project window and select "Options for Target for IOTKit CM33 s – Target FVP" as shown below...



6. In the "Options for Target FVP" dialog, click the "Debug" tab to open the debug settings view...



7. In the drop down selection for the debugger, select "Models ARMv8-M Debugger", then click the "Settings" button next to it...

sm Link	er Debug	Utilities		
⊙ <u>U</u> se:	Models ARI	Mv8-M Debugger	- Settin	gs
6	ULINK Pro	ARMv8-M Debug Mv8-M Debugger	ger	\leq
🗌 Load	OLINK2/ME CMSIS-DAF	ARMv8-M Debu ARMv8-M Debu	gger gger	
Initializatio	on File:			

8. You should have the "Models ARMv8-M Target Driver Setup" dialog box open now. Here we need to make a few entries. Keep in mind your paths may vary based on your system preferences. First let's point to the actual FVP executable, "FVP_MPS2_Cortex-M33_MDK.exe" by clicking the ellipsis button is to browse to the file location in the MDK install directories similar to the path below...

Mod	dels ARMv8-M Target Driver Setup	
De	ebug	
	- 🔍 Use: Launch Simulation	
	Command: C:\Keil_v5\ARM\FVP\MPS2_Cortex-M\FVP_MPS2_Cortex-M33_MDK.exe	\$
	Arguments:	'

9. Next we need to update the "Target:" box with the CPU we want to connect to which in this case is "cpu0". Click on the ellipsis button at the right of the "Target:" box to display the available CPU selections available and select "cpu0" then click "OK"...

Select Ta	get	×
Target	CPU	
<pre></pre>	ARM_Cortex-M33	
Cpu1	ARM_Cortex-M33	
	OK	Cancel

10. In the "Configuration File" box, select the "IOTKit_CM33_FP_config.txt" configuration file from the same location of the file FVP_MPS2_Cortex-M33_MDK.exe mentioned above, which is similar to the below...

Models ARMv8-M Target Driver Setup	\times
Debug Image: Connection Timeout Command: C:\Keil_v5\ARM\FVP\MPS2_Cortex-M\FVP_MPS2_Cortex-M33_MDK.exe Image: C:\Keil_v5\ARM\FVP\MPS2_Cortex-M\FVP_MPS2_Cortex-M33_MDK.exe	
Arguments:	
Configuration Hie C:\Keil_v5\ARM\FVP\MPS2_Cortex-M\IOTKit_CM33_FP_config.txt Edit Generate	
C Use: Running Simulation	
Shut Down Simulation Update List	
OK Cancel Help	

If your setup box is similar to the above, click "OK" on both dialog boxes to save the changes and close them out.

11. Repeat this same procedure for the non-secure project, "IOTKit_CM33_ns", by right clicking the project name and selecting "Set as Active Project" button on the non-secure project...



12. Follow the same process as in steps 2 through 10 above. Note that both projects can use the same configuration file in step 10.

Let's build the projects and debug the FVP target

1. Click the "Batch Build" button on the icon bar near the top left of the MDK window to open the "Batch Build" dialog box. In the below example I've deselected the "V2M-MPS2+" projects to save on build time. This is optional and only affects the amount of build time. In this simple project the build time is minimal but, can become more significant as your project becomes more complex. In this case make sure the "FVP" targets are selected for both the secure and non-secure targets. Then click "Rebuild"...

Batch Build	X
Select Project Targets:	
□ IOTKit_CM33_s	Build
V2M-MPS2+ 	Rebuild
⊡-IOTKit_CM33_ns V2M-MPS2+	Clean
FVP	
	Select All
	Deselect All
	Help
	Close
,	

2. In the "Build Output Window at the bottom you should see both projects built...

```
Build Output

compiling system_IOTKit_CM33.c...

linking...

Program Size: Code=844 RO-data=588 RW-data=32 ZI-data=4200

".\Objects\IOTKit_CM33_ns.axf" - 0 Error(s), 0 Warning(s).

Build Time Elapsed: 00:00:01

Batch-Build summary: 2 succeeded, 0 failed, 2 skipped - Time Elapsed: 00:00:09
```

3. Now let's start a debug session using the FVP. First make sure you start the debug session from the secure project. This is important in our example as the secure project initializes the memory and security settings. Again, right click the secure project, "IOTKIT_CM33_s", and make it the active project by clicking the "Set as Active Project" button...



5. At this point you should see the FVP open in another window on your machine similar to the below screen capture...



6. Back in the debug session, click run... File Edit View Project Flash



7. You should see the FVP window's LCD screen update, the LED lights begin to blink and another Telnet window providing messages via a UART connection from the FVP...

CLCD Cortex-M33 MPS2	al Teinet localhost
Per Libra not national and an	#10 Upp1d (non-score) #110 Upp1d (non-score)
VGA display	Rello Warld (unm-secure) Rello Warld (unm-secure)
KO and Touch Screen VCM 14/932 District Signature 4 (Non - Stanting WWW Kint 1 - Spon	A DE VALL AUTORISTICA Tello V

8. Press stop...



9. As an example let's walk through the process of going from non-secure to secure and back, by placing a break point at or near line 97 in the main_ns.c file then press "Run" to stop at that breakpoint...



10. From this point, click into the "Disassembly" window and single step (F11) through the code. After around five steps you'll see the "SG" instruction which is the secure gate instruction. After executing that instruction you should see the debugger update from "Non-Secure Thread" to "Secure Thread" in the register window...



Registers 🗜 🔝 Disassembly Register 0x1 Value COx10001DB4 F7FFBAEA B Secure_LED_On (0x1000138C) R0 0x00000004 0x10001DBC F7FFBB20 B Secure LED On callback (0x10001400) 0x0000001 R1 0x10001DC0 E97FE97F SG 0x40300004 R2 0x10001DC4 F7FFBB54 в Secure_printf (0x10001470) R3 0x00000020 0x10001DC8 0000 MOVS r0,r0 R4 0x28200004 0x10001DCA 0000 MOVS r0, r0 R5 0x28200020 R6 0x00000000 R7 0x00000000 Abstract.txt startup_IOTKit_CM33.s main_ns.c 0x00000000 R8 62 R9 0x00200200 63 R10 0x00200598 64 0x00000000 R11 nt main (void) 65 R12 0x10001DB1 66 uint32_t i; 67 68 69 volatile unt32_t fpuType = SCG_GetFPUType(); volatile float x1 = 12.4567f; volatile float x2 = 0.6637967f; volatile float x3 = 24.111118f; xPSR 0x21000000 70 71 пкес Secure 72 73 74 75 Internal 76 77 Privilege Privileged xercise some core register from Non Secure Mode */ MSP Stack 78 79 _get_MSP(); _get_PSP(); 242531863 States Sec 24 25318630 80 E-FPU 81 Secure_LED_On_callback(NonSecure_LED_On); Secure_LED_Off_callback(NonSecure_LED_Off); 82 83

11. Continuing the single step in the disassembly window for approximately 23 steps and you should see the branch back to non-secure instruction "BXNS" which when executed will take you back to the "Non-Secure Three d" state in the new secure main measurem.

Secure Thread" state in the non-secure main program

After the "SG" Instruction...

Registers	÷ 🔟	
Register	Value	0x100013F0 4671 MOV r1,1r
register □ Core □ R0 □ R1 □ R2 □ R3 □ R4 □ R5 □ R6	legister Value → Core R0 0x0000000 R1 0x002004EA R2 0x002004EA R3 0x002004EA R4 0x28200004 R5 0x082000000 R5 0x08200000000 R5 0x0800000000 R5 0x08000000000 R5 0x08000000000 R5 0x08000000000000 R5 0x08000000000000000 R5 0x080000000000000000000000000000000000	0x100013F2 4672 MOV r2,1r 0x100013F4 4673 MOV r3,1r 0x100013F6 46F4 MOV r12,1r 0x600013F6 45F4 MOV r12,1r 0x600013F6 4774 BXNS 1r 0x100013FC 4774 BXNS 1r 0x100013FE 0000 MOVS r0,r0 62: {
R7 0x0000000 R8 0x0000000 R9 0x002002CC R10 0x002002CS R11 0x0000000 R12 0x002004EA R13 (SP) 0x38001118 R14 (LR) 0x02004EA R15 (PC) 0x100013FC	0x0000000 0x0000000 0x002002CC 0x00200538 0x00000000 0x002004EA 0x38001118 0x002004EA 0x100013FC 0x000010	Abstract.tt startup_IOTKit_CM33.s main_ns.c E LED_V2M-MP52.c 57 /*
Banked Soure Non-Secure Internal		67 _7 67 int32_t Secure_LED_Off_callback(NonSecure_fpParam_callback)attribute((cmse_nonsec 68 int32_t Secure_LED_Off_callback(NonSecure_fpParam_callback) 69 ⊟{

12. More information about CMSIS-Core for ARMv8-M can be found in the included documentation in the Manage RTE component "CMSIS-CORE for Cortex-M, SC000, SC300, ARMv8-M" or by browsing to it in your installation at...

<install point> /ARM/PACK/ARM/CMSIS/5.0.1/CMSIS/Documentation/Core/html/index.html

13. For more information about the Cortex-M33 IoTKit FPV look in the following directories of the Cortex-M33_IoT_kit_2_0.zip file...

\Cortex-M33_IoT_kit_2_0\app_notes\AN505\docs\ DAI0505A_example_iot_kit_subsystem_for_v2m_mps2.pdf \Cortex-M33_IoT_kit_2_0\boards\Docs\ARMv8-M_IoT_Kit_UG.pdf

While these both are for the MPS2 FPGA image, they are also applicable to the FVP

This is a very simple example but I hope it helps you understand Secure and Non-secure operations better.

Some Notes

1. Note that the MDK tools do two incremental loads before starting, one each for the secure and non-secure domains.

This is a very simple example but I hope it helps you understand Secure and Non-secure operations better.