arm KEIL

Abstract

This application note describes how to create new projects and debug applications for the Arm[®] Cortex[®]-M4 core of STMicroelectronics' heterogeneous multi-core device STM32MP1 in Arm Keil MDK. Two examples using different boot modes are explained step by step. The examples are present in the STM32MP1xx device family pack. Additionally, it is explained how to generate an OpenSTLinux device tree for customized peripheral assignments.

Contents

Using STM32MP1 Cortex-M with Keil MDK	1
Abstract	1
Prerequisites	2
Introduction	2
Hardware setup for the STM32MP157x-EV1 evaluation board	3
Boot in engineering mode	3
Boot in production mode	3
Debug connection	3
Hardware setup for the STM32MP157C-DK2 discovery kit	4
Boot in engineering mode	4
Boot in production mode	4
Debug connection	4
ST-Link	5
Engineering mode	5
Production mode	5
ULINK	6
Engineering mode	6
Production mode	6
Customizing the μ Vision Tools Menu	6
Creating a new Blinky project for engineering mode	7
Project creation step by step	7
Creating an OpenAMP project for production mode	14
Project creation step by step	14
Useful links	24
Creating a new Blinky project for engineering mode Project creation step by step Creating an OpenAMP project for production mode Project creation step by step Useful links	7 7 14 14 24

Prerequisites

MDK v5.27 provides support for creating and debugging applications for Arm Cortex-M based cores of heterogeneous multi-core devices, including STM32MP1 devices from STMicroelectronics.

To be able to use the examples provided in this application note, you need to have a valid MDK license (not MDK-Lite). You can use the <u>built-in seven days trial version</u> of MDK-Professional.

Using <u>PackInstaller</u>, make sure that the following software packs are available:

- ARM.CMSIS.5.5.1.pack (or higher)
- ARM.AMP.1.1.0.pack (or higher)
- Keil. STM32MP1xx.1.1.0.pack (or higher)

A Bash shell is required to run shell scripts. Install for example <u>Git for Windows</u> and make sure the installation folder (C:\Program Files\Git\bin) is added to the Windows path. In Windows, check **Control Panel – System – Advanced System Settings – Environment Variables – System Variables – Path**

Introduction

The STM32MP1 is a heterogeneous device based on a single- or dual-core Arm Cortex-A7 (CA7) and an Arm Cortex-M4 (CM4) core. The Arm Cortex-A7 core provides access to open-source operating systems (such as Linux or Android) while the Arm Cortex-M4 core leverages the STM32 MCU ecosystem and usually runs bare-metal code or a real-time operating system such as Keil RTX5.

This device offers two different boot modes: engineering and production.



- The *engineering* boot mode allows the user to connect a debugger to an opened chip, so that any program can be loaded on either the CA7 or the CM4. When the ROM code detects engineering boot mode, it reopens CA7 secure debug and starts the CM4 to run an infinite loop. In this boot mode, the CM4 application can be loaded directly with the debug adapter. This is the recommended setup to start a new CM4 project.
- In *production* boot mode, the master processor CA7 loads the application and starts the coprocessor CM4. In this case, system resources, such as clock and regulators, are managed by OpenSTLinux, which loads the executable file and launches the coprocessor through Linux remoteproc commands.

For the examples demonstrated in this tutorial, it's assumed you are familiar with the STM32MP157x-EV1 evaluation board or the STM32MP157C-DK2 discovery kit and you have already loaded and booted the OpenSTLinux image. Otherwise please follow the <u>Getting Started</u> section of the <u>STM32 MPU Wiki</u>, which is the entry point for many references in this document.

The examples are also part of the STM32MP1xx device family pack. They can be copied using the Pack Installer tool. Open it by clicking on the Pack Installer icon in the μ Vision toolbar:

Pack Installer - C:\KEIL_V5\ARM\PACK			
2 Device: STMicroelectronics - STM32MP1 Series			
Devices Boards Devices	Packs Examples		
Search: STM32MP1 • X	Show examples from installed Packs only		
Device 🛆 Summary	Example	Action	Description
E All Devices 8 Devices	Al_Character_Recognition (STM32MP157C-EV1)	🚸 Сору	Al Character Recognition example
STMicroelectronics 8 Devices		🚸 Сору	CMSIS-RTOS2 Blinky example
🖃 🏤 STM32MP1 Series 8 Devices	OpenAMP_TTY_echo (STM32MP157C-EV1)	🚸 Сору	OpenAMP TTY echo example

AN319 – Using STM32MP1 Cortex-M with Keil MDK

Copyright © 2019 Arm Ltd. All rights reserved www.keil.com/appnotes/docs/apnt_319.asp

Hardware setup for the STM32MP157x-EV1 evaluation board

The following connections must be made to be able to debug the hardware successfully:

- Connect the power supply to the CN1 jack.
- Using a Micro-USB cable, connect to the ST-Link (CN4); this can be used for debug and a serial terminal
- [Optional] Connect a ULINK debug adapter using an adapter board to the JTAG connector (CN14)
- Ethernet connection (network with DHCP server or fixed IP address)



Boot in engineering mode

For engineering mode, you need to set the boot pins in the following order:

- Boot 0: Off
- Boot 1: Off
- Boot 2: On

Boot in production mode

For production mode, you need to set the boot pins in the following order:

- Boot 0: On
- Boot 1: Off
- Boot 2: On



Debug connection

The STM32MP157x-EV1 evaluation board includes an on-board ST-Link debug adapter. While this is great to get started, the capabilities of the debug unit are limited. Using the 20-pin JTAG connector (with the optional adapter board HPI-0195C), you can connect any member of the <u>ULINK family of debug adapters</u> to get more debug visibility. The required connections and software settings are explained later.

AN319 - Using STM32MP1 Cortex-M with Keil MDK

Hardware setup for the STM32MP157C-DK2 discovery kit

The following connections must be made to be able to debug the hardware successfully:

- Connect the USB Type-C power supply to the CN6 connector.
- Using a Micro-USB cable, connect to the ST-Link (CN11); this can be used for debug and a serial terminal
- Ethernet connection (network with DHCP server or fixed IP address)



Boot in engineering mode

For engineering mode, you need to set the boot pins in the following order:

- Boot 0: Off
- Boot 2: On

Boot in production mode

For production mode, you need to set the boot pins in the following order:

- Boot 0: On
- Boot 2: On

Debug connection

The STM32MP157C-DK2 discovery kit has only an on-board ST-Link debug adapter. If you need more debug capabilities, use the STM32MP157x-EV1 evaluation board instead. The required connections and software settings are explained in the following page.





ST-Link

Engineering mode

Go to **Project – Options for Target**. On the **Debug** tab, select "ST-Link" and then open the "Settings" dialog. On the next dialog's **Debug** tab make sure that AP equals 2:

Cortex-M Target Driver Setup				×
Debug Trace Rash Download Pack	-SW Dev	vice		1
Unit: ST-LINK/V2-1 Shareable ST-Link	SWDIO	IDCODE 0x6BA02477	Device Name ARM CoreSight SW-DP	Move Up
Serial Number: 066EFF313732524E43012315 Version: HW: V2-1 FW: V2J33M25	© Aut C Ma	tomatic Detection nual Configuration	ID CODE:	Down
I ⊂ Check version on start	Add	DeleteU	pdate IR len:	AP: 2

On the **Trace** tab, make sure that "Trace Enable" is unchecked:

Cortex-M Target Driver Setup		×
Debug Trace Flash Download Pack	1	
Core Clock: 10.000000 MHz	Trace Enable	

On the **Pack** tab, make sure "Debug Description Enable" is unchecked:

Cortex-M Target Driver Setup	×
Debug Trace Flash Download Pack	
Debug Description Pack: Kel.STM32MP1xx_DFP.0.0.1-dev8	

Click OK.

On the Utilities tab, uncheck "Update Target before Debugging":

	Options for Target 'Target 1'	×
	Device Target Output Listing User C/C++ (AC6) Asm Linker Debug Utilities	
	Configure Flash Menu Command	
	Use Debug Driver Settings	
C	Click OK.	

Production mode

In production mode, in addition to the setting used above, make sure that you are not resetting the target when connecting to it. On the **Debug** tab, make sure that "Reset" equals VECTRESET and that "Reset after Connect" is not used:

Debug Connect & Reset Options Connect: Normal Reset after Connect i Stop arter Reset	Cache Options Cache Code Cache Code Cache Memory Cache Memory Cache Memory Cache Memory
	OK Cancel Apply

Note:

In production mode, the debug adapter cannot connect directly to the device. OpenSTLinux will establish the connection once the application for the Cortex-M4 is loaded. Thus, you need to have a binary image of the application ready and must follow the steps 15 – 17 on page 23.

ULINK

Engineering mode

Go to **Project – Options for Target**. On the **Debug** tab, for example, select "ULINK Pro Cortex Debugger" and then open the "Settings" dialog.

On the next dialog's **Trace** tab set the Core Clock to 133.25 and enable "Trace Enable". Use "Serial Wire Output – Manchester" as the trace port:



On the Pack tab, check that Debug Description is enabled:

ULINK Pro Cortex-M Target Driver Setup	×
Debug Trace Rash Download Pack	
Debug Description Pack: Keil STM32MP1xx DFP.0.0.1-dev8	
Enable Flash Sequences	
Log Sequences: C:\03_workspace\MDKv5\STM32\MP1_eng\eng_mode_Sequences_*Jog	
Configuration: \DebugConfig\Target_1_STM32MP157CAAx_Cortex-M4.dbgconf	Edit

Production mode

In production mode, use the same settings as above. In addition, click on the "Edit" button on the **Pack** tab (see screenshot above). The dbgconf configuration file opens in a new window. In Configuration Wizard mode, enable "Restart", "Stop and Remove", and "Load and Run":

These options will enable the ULINK debug adapters to automatically use the right debug sequences to communicate with the target (not available for the ST-Link debug adapter).

Customizing the µVision Tools Menu

Alternatively, to the method above (and the only automated way when using the ST-Link), you can add shell scripts to the μ Vision Tools Menu. These shell scripts load, run, stop and remove the application in production mode. An additional one generates the device tree files.

The scripts are in the STM32MP1's pack subfolder 'Tools' and can be added to MDK menu:

Tools – Customize Tools Menu – Import... {PACK_FOLDER}\Tools\tools.cfg

(for example C:\KEIL_V5\ARM\PACK\Keil\ STM32MP1xx_DFP\0.0.1\Tools\tools.cfg)





Creating a new Blinky project for engineering mode

The ROM code detects *engineering* boot and reopens CA7 secure debug and starts CM4 to run an infinite loop, allowing the user to connect a debugger to an opened chip, so that any program can be loaded directly with the debug adapter.

The boot switches are described in the Hardware setup section.

In this example, Keil RTX5 threads are used to control LEDs and a user button. LD3 and LD4 blink as running lights, while blinking is paused if the UserPA13/B2 button is pressed.

Project creation step by step

- In μVision, select Project New μVision Project... and give a name to the new project.
- From the device list, select STMicroelectronics STM32MP157CAAx:Cortex-M4 device.
- 3. In the Manage Run-Time Environment window, select the following components:
 - Board Support:BSP
 - Board Support:Buttons
 - Board Support:LED
 - CMSIS:CORE
 - CMSIS:RTOS2 (API):Keil RTX5
 - (Variant Source)
 - Compiler:Event Recorder
 - Compiler:IO:STDOUT
 - (Variant ITM)
 - Device:Startup
 - Device:STM32Cube HAL:Common
 - Device:STM32Cube HAL:Cortex
 - Device:STM32Cube HAL:GPIO
 - Device:STM32Cube HAL:HSEM
 - Device:STM32Cube HAL:PWR
 - Device:STM32Cube HAL:RCC

Device			
	Software Packs	•	
Vendor:	STMicroelectronics		
Device:	STM32MP157CAAx:Cortex-M4		
Toolset:	ARM		
Search:			
]	Description	
.		Description.	
ia 🍄 S⊺	M32MP1 Series	Arm based dual Cortex-A7 650 MHz + Cortex-M4 MPU	1
ė- ~	STM32MP157	32-bit dual-core Arm Cortex-A7	
E	STM32MP157AAAx	Up to 650 MHz (Up to 4158 CoreMark)	
E	STM32MP157AABx	256 Kbyte unified level 2 cache	
	STM32MP157AACx	Arm NEON and Arm TrustZone	
ŧ	STM32MP157AADx	32-bit Arm Cortex-M4 with FPU/MPU	
Ė	STM32MP157CAAx	Up to 200 MHz (Up to 673 CoreMark)	
	STM32MP157CAAx	External DDR memory up to 1 Gbyte	
	STM32MP157CABx	708 Kbyte of internal SRAM: 256 KB of AXI	
	▶ ▼		~

Advances:	oftware Component	Sel.	Variant	Version	Description
Beard Support ■ SSM ● SSM ■ STM22MP157C-EV1 10.0 StM22MP157C-EV1 Foundation Board ● SSM ■ 10.0 Beard specif: exiting for hardware initialization ● SSM ■ 10.0 Beard specif: exiting for hardware initialization ● LED ● LD ■ 10.0 ED Interface ● LD ● LD ● LD ● LD ● CORE ● LD ● LD ● LD ● CORE ● LD ● LD ● LD ● NNU bb ■ 1.10 Constance Interface Constance Interface Components ● CORE ● LD ● LD ● LD Constance Interface Components ● NNU bb ■ 1.10 Constance Interface Constance Interface ● STOS (APD) ■ 1.10 Constance Interface Constance Interface ● FreeROS (APD) ■ 1.10 Constance Interface Constance Interface ● CARS Brown Exit For KTX Son AMDI Source Interface Constance Interface ● String Constance Interface ■ Constance Interface Constance Interface Constance Interface ● C	AMP				Asymmetric Multiprocessing
● BSP ● Buttons (AP)0 ● Dattons Interfaces ● Buttons (AP)0 ● Dattons Interfaces ● Colleary ● Dattons Interfaces ● Colleary ● Dattons Interfaces ● Dattons ● Dattons Interfaces	🔸 Board Support		STM32MP157C-EV1	1.0.0	STM32MP157C-EV1 Evaluation Board
■ Buttons (AP) 10.0 Buttons Interáce ■ Buttons (AP) 10.0 Buttons Interáce ■ ED (AP) 10.0 ED Interáce ■ CD (AP) 10.0 ED Interáce ■ CD (AP) 10.0 ED Interáce ■ CLBS ■ 10.0 ED Interáce ■ CLBS ■ 10.0 ED Interáce ■ CLBS ■ 10.0 Contex Microcontroller Software Interáce Components ■ CLBS ■ 11.0 CMSS SOG SCOUL AND	BSP	v		1.0.0	Board specific settings for hardware initialization
● Butons F 1.0.0 Butons interface ● LED (P) 1.0.0 LED Interface 1.0.0 LED Interface ● LED (P) ● L.D.0 LED Interface 1.0.0 LED Interface ● CMSIS ● CORE ● S1.3 CAMSIS-CORE Functional End Science AL SCOOD. and SCIAO ● OSE ● S1.3 CAMSIS-CORE Functional End Science AL SCOOD. and SCIAO ● SCOOR and SCIAO ● KIRD SLAPI ■ 1.3.2 CAMSIS-CORE Functional Science AL SCOOD. and SCIAO ● SCOOR and SCIAO ● KIRD SLAPI ■ 1.3.2 CAMSIS-FINITAlexial Elevench Library ● SCOOR and SCIAO ● SCOOR and SCIAO ● KIRD SLAPI ■ 1.3.0 CAMSIS-FINITAlexial Elevench Library ● SCOOR and SCIAO ● SCOOR and SCIAO ● KIRD SLAPI ■ 2.3.3 CAMSIS-FINITAL Elevench Macronal SCIAO ● SCOOR and Annove ML ILD ● KIRD SCIAPI ■ UBAR ■ SCOOR and SCIAO ● SCOOR and Annove ML ILD ● KIRD SCIAPI ■ UBAR ■ SCOOR and Annove ML ILD ■ SCOOR and Annove ML ILD ● KIRD SCIAPIN ■ UBAR ■ SCOOR and Annove ML ILD ■ SCOOR and Annove ML ILD ● CMSIS SCINCS Validation Santet <td>🖃 💠 Buttons (API)</td> <td></td> <td></td> <td>1.0.0</td> <td>Buttons Interface</td>	🖃 💠 Buttons (API)			1.0.0	Buttons Interface
■ (bb (AP) 1.00 LED Interface ■ (Clamy 1.00 LED Interface ● (Clamy 1.01 Context MacGoometraller Software Interface Company ● (Clamy 1.01 Context MacGoometraller Software MacGoometraller MacGoometral	Buttons	✓		1.0.0	Buttons Interface
ED ELD interface ED ELD interface ELD interfa	🖃 💠 LED (API)			1.0.0	LED Interface
C Library Control Contro Control <thcontrol< th=""> <</thcontrol<>	LED	v		1.0.0	LED Interface
CMSS Context Micro-controller's oftware instructs. Components ● CORE ♥ \$ 1.3 CMSS-CORE for -Center-M. SCOD, sci 200, ARM-A. ARM-A. I.M. ● DP PN Lib ■ 1.3 CMSS-CORE for -Center-M. SCOD, and SCI00 ● ROS (APD) ■.1.0 CMSS-API of -Center-M. SCOD, and SCI00 ■ ● ROS (APD) ■.1.0 CMSS-RIDS API of -Center-M. SCOD, and SCI00 ■ ● ROS (APD) ■.1.1 CMSS-RIDS API of -Center-M. SCOD, and SCI00 ■ ● Kall FUSS RINS ANDMPU © Source 5.0 CMSS-RIDS Contex-M. SCOD, Cander AM-SCOD, Cander AM-SCO	🗈 💠 C Library				
CORE	🗈 💠 CMSIS				Cortex Microcontroller Software Interface Components
● DSP 1.3.2 CMSS-DSP Likery for Contex-M. SCOOL and SCOOL ● RTOS (AP) 1.0.0 CMSS-RTOS APL for Contex-M. SCOOL and SCOOL ● RTOS (AP) 1.0.0 CMSS-RTOS APL for Contex-M. SCOOL and SCOOL ● RTOS (AP) 1.0.0 CMSS-RTOS APL for Contex-M. SCOOL and SCOOL ● RTOS (AP) 2.1.3 CMSS-RTOS APL for Contex-M. SCOOL and SCOOL ● Kall FVS RTOS (AP) 5.0 CMSS-RTOS APL for Contex-M. SCOOL 2003 and AmmA MARK MRING SCOOL AND SCOOL 2003 and AmmA MARK MRING	CORE	v	~	5.1.3	CMSIS-CORE for Cortex-M, SC000, SC300, ARMv8-M, ARMv8.1-M
Image: NN Lib 1.1.0 CMSS-NN INeural Network Library Image: NN Lib 1.1.0 CMSS-NN INeural Network Library Image: NN Lib 1.0.0 CMSS-NN INeural Network Library Image: NN Lib 1.0.0 CMSS-NN INeural Network Library Image: NN Lib 1.0.0 CMSS-NN INeural Network Library Image: NN Lib No No NN Lib No NN Lib Image: NN Lib Source S.0.0 Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image: NN Lib No NN Lib No NN Lib Image:	DSP			1.5.2	CMSIS-DSP Library for Cortex-M, SC000, and SC300
Image: Note Status Interpretation CMSIS-RIOS API (or Contex-M. S2000, and SC300) Image: Note Status Note Status Note Status Note Status Image: Note Status Note Status Note Status Note Status Image: Note Status Note Status Note Status Note Status Image: Note Status Note Status Note Status Note Status Image: Note Status Note Status Note Status Note Status Image: Note Status Note Status Note Status Note Status Image: Note Status Note Status Note Status Note Status Image: Note Status Note Status Note Status Note Status Image: Note Note Note Note Note Note Note Note	NN Lib			1.1.0	CMSIS-NN Neural Network Library
■ R1052 (API) 2.1.3 CMSS-R105.API for Contrel-M.52000, and SC300 ● Keil FuSs R1X5 noMPU Source 5.5.0 CMSIS-R1052 Functional Saftey R1X5 for Contex-M4 without MPL ● Keil FuSs R1X5 noMPU ✓ Library ✓ 5.5.0 CMSIS-R1052 Functional Saftey R1X5 for Contex-M4 without MPL ● Keil R1X5 ✓ Library ✓ 5.5.0 CMSIS-R1052 Middition API ● CMSIS R1052 Validation API 1.0.0 Run API test for enabled drivers ● CMSIS R1052 Validation CMSIS-R1052 Validation Suite CMSIS-R1052 Validation Suite ● CMSIS R1052 Validation CMSIS-R1052 Validation Suite CMSIS-R1052 Validation Suite ● CMSIS R1052 Validation ARM Fuse Compiler ARM Fuse Compiler ● Free Recorder ✓ DAP 1.4.0 Completer With R118 System component ● Free Recorder ✓ DAP 1.2.0 Stop program execution at a breakpoint when using STDIN ● Free Recorder ✓ DAP 1.2.0 Stop program execution at a breakpoint when using STDIN ● Freekpoint 1.2.0 Stop program execution at a breakpoint when using STDIN ● Freekpoint 1.2.0 Stop program execution at a breakpoint when usin	🗉 🚸 RTOS (API)			1.0.0	CMSIS-RTOS API for Cortex-M, SC000, and SC300
FreeRtOS Keil Kusk ETXS noMPU Source Store Keil RTXS Gource Store	🖨 🚸 RTOS2 (API)			2.1.3	CMSIS-RTOS API for Cortex-M, SC000, and SC300
Kall FuSa RTX5 moMPU Kall FuSA STATUS FOR Carbon Marked	FreeRTOS			10.0.1	CMSIS-RTOS2 implementation for Cortex-M based on FreeRTOS
Keil RTX5	Keil FuSa RTX5 noMPU		Source	5.5.0	CMSIS-RTOS2 Functional Saftey RTX5 for Cortex-M4 without MPU support (Source)
Image: CMSB Driver Unified Device Driver completion to CMSB-Driver Specifications CMSB Driver Validation API 1.0.0 Rum API test for enabled drivers CMSB RTOS Validation CMSB RTOS Validation Suite API 1.0.0 Rum API test for enabled drivers CMSB RTOS Validation CMSB RTOS Validation Suite ARM FLSS Complit? 1.0.0 CMSB RTOS Validation Suite Complet Exern Recorder ✓ DAP 1.4.0 Event Recording and Component Viewer via Dabug Access Port I ✓ NO File File System Its Preskpoint 1.2.0 Store program execution at a breakpoint when using STDER ● Statup StoDUT Ø File System Breakpoint 1.2.0 Stop program execution at a breakpoint when using STDER ● Device Image: Statup V Image: Statup Statup <t< td=""><td>Keil RTX5</td><td>~</td><td>Library ~</td><td>5.5.0</td><td>CMSIS-RTOS2 RTX5 for Cortex-M, SC000, C300 and Armv8-M (Library)</td></t<>	Keil RTX5	~	Library ~	5.5.0	CMSIS-RTOS2 RTX5 for Cortex-M, SC000, C300 and Armv8-M (Library)
API 1.0.0 Run API test for enabled interes C KMSIS RTOS Validation CMSIS RTOS Validation Suite CMSIS RTOS Validation Suite C CMSIS RTOS Validation CMSIS RTOS Validation Suite CMSIS RTOS Validation Suite C CMSIS RTOS Validation CMSIS RTOS Validation Suite CMSIS RTOS Validation Suite C CMSIS RTOS Validation Suite CMSIS RTOS Validation Suite CMSIS RTOS Validation Suite C MSIS RTOS Validation Suite ARM FuSa Compile I 14.0 Creating and Component Viewers via Debug Access Port / Retarget Input/Output I File System T1.2.0 Use retargeting together with the file System component I STORT Breakpoint 12.0 Stop program execution at a breakpoint when using STDIN I STOUT I TIM 12.0 Stop program execution at a breakpoint when using STDIN I STOUT I TIM 12.0 Stop program execution at a breakpoint when using STDIN I STOUT I TIM 12.0 Stop program execution at a breakpoint when using STDIN I STOUT I TIM 12.0 Stop program execution at a breakpoint when using STDIN I STOUT I TIM 12.0 Stop program executin at a breakpoint when using STDIN	🛛 💠 CMSIS Driver				Unified Device Drivers compliant to CMSIS-Driver Specifications
CMSIS RTOS Validation Suite ARM FuSa Compiler Suite Suite CMSIS RTOS Validation VALID CMSIS RTOS VALIDATION VALIDATION VALIDATION VALIDATION	🛛 💠 CMSIS Driver Validation		API	1.0.0	Run API test for enabled drivers
CMSIS RT032 Validation CMSIS RT032 Validation Suite ARM FUSa Compile(ARM FUSa Compile(ARM FUSa Compile(Event Recorder Compiler Support Sup	CMSIS RTOS Validation				CMSIS-RTOS Validation Suite
Compiler ARM FuSa Compiler 16.0 Compiler Stand ARM Compiler 3 and ARM Compiler 5 Event Recorder VDAP 1.4.0 Exet Recording and Component Viewer via Debug Access Der (I VI VI Retriget input/Output Retriget input/Output VI File File Retriget input/Output VI Stop program execution at a breakpoint when using STDERR STDOUT TIM VI 1.2.0 Stop program execution at a breakpoint when using STDIN STDOUT TIM VI 2.0 Stop program execution at a breakpoint when using STDIN STDOUT TIM VI 2.0 Stop program execution at a breakpoint when using STDIN STDOUT TIM VI 1.0.0 System Startup for STMicroelectronics STM32MP157 Series STM32Cube HAL STM32Cube HAL STM32Cube HAL Startup System Setup VI CEC 1.0.0 Consumer Electronics Control (CEC) HAL driver CEC CCC 1.0.0 Consumer Electronics Control (CEC) HAL driver Cervp CCC 1.0.0 Consumer Electronics Control (CEC) HAL driver Cervp CCC Contex I.0.0 Contex HAL driver </td <td>CMSIS RTOS2 Validation</td> <td></td> <td></td> <td></td> <td>CMSIS-RTOS2 Validation Suite</td>	CMSIS RTOS2 Validation				CMSIS-RTOS2 Validation Suite
● Event Recorder ✓ DAP 1.4.0 Event Recording and Component Viewer via Debug Access Pent (Retarget Input/Output ● File File System 1.2.0 Stop program execution at a breakpoint when using STDERR ● STDER Breakpoint 1.2.0 Stop program execution at a breakpoint when using STDERR ● STDOUT ✓ ITM 1.2.0 Stop program execution at a breakpoint when using STDIN ● STDOUT ✓ ITM 1.2.0 Stop program execution at a breakpoint when using STDIN ● STDOUT ✓ ITM 1.2.0 Stop program execution at a breakpoint when using STDIN ● STDOUT ✓ ITM 1.2.0 Stop program execution at a breakpoint when using STDIN ● STDOUT ✓ ITM 1.2.0 Stop program execution at a breakpoint when using STDIN ● Starup I.0.0 System Starup for STMicroelectronics STM32MP157 Series STM32Cube HAL ● ADC I.0.0 Consumer Flectronics Control (ECC) HAL driver CCC ● CRC I.0.0 Consumer Flectronics Control (ECC) HAL driver ● CRC I.0.0 Consumer Flectronics Control (ECC) HAL driver ● CRC I.0.0 Consumer HAL driver ● CAC I.0.0 Digital Fiter for Sigma-Deta Modulators (DFSDM) HAL driver ● DCM I.0.0	🛛 💠 Compiler		ARM FuSa Compile ~	1.6.0	Compiler Extensions for ARM Compiler 5 and ARM Compiler 6
VO Retarget Input/Output File File System STDERR File System STDERR Breakpoint STDUN File System STDUN File System STDUN File System STDUN File System Stop program execution at a breakpoint when using STDIN STDUN File System Stop program execution at a breakpoint when using STDIN Stop program execution at a breakpoint when using STDIN TrY Breakpoint Startup Startup for Startup for Stifficrelectronics STM32MP157 Series STM32Cube HAL Stystem Startup for STM32MP157 Series STM32Cube HAL Stystem Startup for Stifficrelectronics STM32MP157 Series STM32Cube HAL Stop program execution at a breakpoint when using TTY CCC 1.0.0 CRC 1.0.0 Creptoraphic processor (CRP) HAL driver CCC 1.0.0 Cortex 1.0.0 Deck 1.0.0 Digital Filter for Sigma-Deta Modulators (DFSDM) HAL driver DCMI <	Event Recorder	v	DAP	1.4.0	Event Recording and Component Viewer via Debug Access Port (DAP)
 File STDERR Breakpoint V V Stop program execution at a breakpoint when using STDERR STDUT V TTV Breakpoint V V Stop program execution at a breakpoint when using STDIN STDUT V TTV Breakpoint V V Stop program execution at a breakpoint when using STDIN Statup. System Setup Statup. System Setup Statup. System Setup STM32Cube HAL STM32E2xx Hardware Abstraction Layer (HAL) Drivers Analog-to-digital converter (ADC) HAL driver CRC L0.0 Construct (ACC) HAL driver CRC CRC CRC CRC Common Contex Conte	⊡				Retarget Input/Output
 STDERR Breakpoint 1.2.0 Stop program execution at a breakpoint when using STDERR STDN STDOUT TTV Breakpoint 1.2.0 Stop program execution at a breakpoint when using STDIN TTV Breakpoint 1.2.0 Stop program execution at a breakpoint when using TTV TV Breakpoint 1.2.0 Stop program execution at a breakpoint when using TTV Startup Startup Startup Startup System Startup for STMicroelectronics STM32AP157 Series STM32Cube HAL STM32Cube HAL CCC CCC ADC I.0.0 Analog-to-digital converter (ADC) HAL driver CCC CRC I.0.0 Common CCC Contex I.0.0 Digital filter for Sigma-Delta Modulators (DFSDM) HAL driver DCMI DL0.0 Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DCMI DDMA I.0.0 Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DCMI DCAN Extri DCAN Extra DAC Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DCAN Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DCAN Extra DAC Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DCAN Extra DCAN Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DCAN DCAN Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DCAN DCAN DAC DAC	🖌 🖗 File		File System	1.2.0	Use retargeting together with the File System component
STDIN STDIN Breakpoint I.2.0 Stop program execution at a breakpoint when using STDIN STDUT TTM I.2.0 Redirect STDUT to a debug output window using ITM TY Breakpoint I.2.0 Stop program execution at a breakpoint when using TTV Breakpoint I.2.0 Stop program execution at a breakpoint when using TTV Breakpoint I.2.0 Stop program execution at a breakpoint when using TTV Breakpoint I.2.0 Stop program execution at a breakpoint when using TTV Breakpoint TV Breakpoint I.2.0 Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program execution at a breakpoint when using TTV Breakpoint Stop program program execution at a breakpoint when using TTV Breakpoint Stop protempore breakpoint at the processor (CRYP) HAL driver Drive	STDERR		Breakpoint 🗸	1.2.0	Stop program execution at a breakpoint when using STDERR
STDUT TTV TTM TTA	STDIN	Ē	Breakpoint 🗸	1.2.0	Stop program execution at a breakpoint when using STDIN
TTY Breakpoint Startup Startup Startup Startup STM32Cube HAL ADC CEC 1.0.0 CRC 1.0.0 CRC Common Cortex Contex Contex DAC DAC Consumer Electronics Control (CEC) HAL driver CCC Consumer Electronics Control (CEC) HAL driver CCC Consumer Electronics Control (CEC) HAL driver Consumer Electronics Control (CEC) HAL driver CCC Consumer Electronics Control (CEC) HAL driver Consumer Electronics Control (CEC) HAL driver Consumer Electronics Control (CEC) HAL driver Contex Contex DAC DAC <td>STDOUT</td> <td>v</td> <td>ITM V</td> <td>1.2.0</td> <td>Redirect STDOUT to a debug output window using ITM</td>	STDOUT	v	ITM V	1.2.0	Redirect STDOUT to a debug output window using ITM
 Device Startup Star			Breakpoint 🗸	1.2.0	Stop program execution at a breakpoint when using TTY
Startup 1.0.0 System Startup for STMicroelectronics STM32MP157 Series STM32Cube HAL STM32F2x Hardware Abstraction Layer (HAL) Drivers ADC 1.0.0 Analog-to-digital converter (ADC) HAL driver CEC 1.0.0 Consumer Electronics Control (CEC) HAL driver CRC 1.0.0 CRC calculation unit (CRC) HAL driver Common 1.0.0 CRC common HAL driver Contex 1.0.0 Corptographic processor (CRYP) HAL driver DAC 1.0.0 Cortex HAL driver DAC 1.0.0 Digital-to-analog converter (DAC) HAL driver DCMI 1.0.0 Digital camera interface (DCMI) HAL driver DCMI 1.0.0 Digital filter for Sigma-Delta Modulators (DFSDM) HAL driver DFSDM 1.0.0 Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DMA 1.0.0 Extranal Interrupts (EXTI) HAL driver CFC 1.0.0 General-purpose I/O (GPIO) HAL driver OFCAN 1.0.0 General-purpose I/O (GPIO) HAL driver HASH 1.0.0 Hardware semaphore (HSSM) HAL driver HASH 1.0.0 Hardware semaphore (HSSM) HAL driver OFC 1.0.0 Interrupts cost Communication I/O (IPCC) HAL driver OFC 1.0.0 Management data input/output (MDIOS) slave HAL driver OV 1.0.0 Kargered ata input/output (MDIOS) slave HAL driver OV 1.0.0 Naser Communication I/O (IPCC) HAL driver OV 1.0.0 Naser Communication I/O (IPCC) HAL driver OV 1.	🛛 🚸 Device				Startup, System Setup
STM32Cube HAL STM32Cu	Startup	~		1.0.0	System Startup for STMicroelectronics STM32MP157 Series
ADC 1.0.0 Analog-to-digital converter (ADC) HAL driver CEC 1.0.0 Consumer Electronics Control (CEC) HAL driver CRVP 1.0.0 CCryptographic processor (CRVP) HAL driver Common 1.0.0 Corport (CRVP) HAL driver Common 1.0.0 Corport (CRVP) HAL driver Contex 1.0.0 Corport (CRVP) HAL driver Contex 1.0.0 Corport (CRVP) HAL driver Contex 1.0.0 Cortex HAL driver DCMI 1.0.0 Digital-to-analog converter (DAC) HAL driver DCMI 1.0.0 Digital converter (DAC) HAL driver DFSDM 1.0.0 Digital Filter for Sigma-Detta Modulators (DFSDM) HAL driver DFSDM 1.0.0 Digital Filter for Sigma-Detta Modulators (DFSDM) HAL driver EXTI 1.0.0 Digital converter (DAC) HAL driver EXTI 1.0.0 Digital Converter (DAC) HAL driver GPIO 1.0.0 External Interrupts (EXTI) HAL driver HASH 1.0.0 Flexible DataRate Controller Area Network (FDCAN) HAL driver HASH 1.0.0 Hash processor (HASH) HAL driver HASH 1.0.0 Hardware semaphore (HSEM) HAL driver EVEC 1.0.0 Inter-integrated circuit (C) interface HAL driver HSEM 1.0.0	STM32Cube HAI				STM32E2xx Hardware Abstraction Laver (HAL) Drivers
CEC CC CCC CC CCC CC C	ADC			1.0.0	Analog-to-digital converter (ADC) HAL driver
CRC CRC CRC CRC CRC CRC calculation unit (CRC) HAL driver CRC pyper 1.0.0 CRC calculation unit (CRC) HAL driver CRC pyper 1.0.0 Correx HAL driver Correx Correx Correx Correx Correx Correx DAC 10.0 Digital comera interface (DCMI) HAL driver DAC DAC DAC DAC DAC DIgital comera interface (DCMI) HAL driver DAC DAC DAC DAC DAC DAC DAC DIgital comera interface (DCMI) HAL driver DAC DOM Digital comera interface (DCMI) HAL driver DFSDM DAC	¢ CEC			100	Consumer Electronics Control (CEC) HAL driver
CRYP CRYP Common Cryptographic processor (CRYP) HAL driver Common Cortex	¢ CRC			100	CRC calculation unit (CRC) HAL driver
Common Common Common Common Contex				100	Contographic processor (CRVP) HAL driver
Allow 10.0 Contract HAL driver DAC 10.0 Digital-to-analog converter (DAC) HAL driver DAC 10.0 Digital-to-analog converter (DAC) HAL driver DCMI 10.0 Digital camera interface (DCMI) HAL driver DFSDM 10.0 Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DMA 10.0 Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DMA 10.0 DMA controller (DMA) HAL driver EXTI 10.0 External Interrupts (EXTI) HAL driver FDCAN 10.0 Flexible DataRate Controller Area Network (FDCAN) HAL driver GPIO V 10.0 General-purpose I/O (GPIO) HAL driver HASH 10.0 Hash processor (HASH) HAL driver HASH 10.0 Inter-integrated circuit (2C) interface HAL driver IPCC 10.0 Inter-integrated circuit (2C) HAL driver IPCC 10.0 Interprocessor Communication I/O (IPCC) HAL driver MDIOS 10.0 Master DMA controller (MDMA) HAL driver WR 10.0 Reset and clock control (RCC) HAL driver WR 10.0 Reset and clock control (RCC) HAL driver	Common			100	Common HAL driver
Allow 10.0 Digital-to-analog converter (DAC) HAL driver DAC 10.0 Digital-to-analog converter (DAC) HAL driver DCMI 10.0 Digital-to-analog converter (DAC) HAL driver DFSDM 10.0 Digital armera interface (DCMI) HAL driver DMA 10.0 Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DMA 10.0 DMA controller (DMA) HAL driver EXTI 10.0 External Interrupts (EXTI) HAL driver FDCAN 10.0 Flexible DataRate Controller Area Network (FDCAN) HAL driver GPIO Interrupts (EXTI) HAL driver HASH HASH 10.0 General-purpose I/O (GPIO) HAL driver HASH 10.0 Hardware semaphore (HSEM) HAL driver HASH 10.0 Hardware semaphore (HSEM) HAL driver HASH 10.0 Inter-integrated circuit (I2C) interface HAL driver IPCC 10.0 Inter-integrated circuit (I2C) HAL driver MDIOS 10.0 Maagement data input/output (MDIOS) slave HAL driver MDMA 10.0 Master DMA controller (PWR) HAL driver WR 10.0 Reset and clock control (RCC) HAL driver WR 10.0 Reset and clock control (RCC) HAL driver	Cortex			1.0.0	Cortex HAL driver
DAC DCM DCM DCM DCM DCM DCM DCM DCM DFSDM DFSDM DDS Digital Camera interface (DCM) HAL driver DFSDM DFSDM DDGM Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DMA DDMA DDMA DDGM DMA DDMA DDM				1.0.0	Digital to applog converter (DAC) HAL driver
DESDM DESDM 10.0 Digital Carrier an Interface (DCM) FAL driver DFSDM 10.0 Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver DMA 10.0 DMA controller (DMA) HAL driver EXTI 10.0 External Interrupts (EXTI) HAL driver FDCAN 10.0 Flexible DataRate Controller Area Network (FDCAN) HAL driver GPI0 FDCAN 10.0 General-purpose I/O (GPIO) HAL driver HASH 10.0 Hash processor (HASH) HAL driver HASH 10.0 Inter-integrated circuit (I2C) interface HAL driver IPCC Inter-integrated circuit (I2C) interface HAL driver IPCC Inter-integrated circuit (I2C) interface HAL driver MDIOS Inter-integrated circuit (IDIM) HAL driver MDMA Inter Owner Timer (LPTIM) HAL driver MDMA Inter Owner Timer (LPTIM) HAL driver WR Interface (QSPI) Interface (QSPI) HAL driver QSPI Interface Interface (QSPI) HAL driver Interface (QSPI) HAL dri				1.0.0	Digital company interface (DCMI) HAL driver
				1.0.0	Digital Camera Interface (DCMI) HAL driver
DMA DMA DMA DMA DMA DMA Controller (DMA) HAL driver EXTI DMA				1.0.0	Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver
EXII 10.0 External Interrupts (EXII) HAL driver FDCAN 10.0 Flexible DataRate Controller Area Network (FDCAN) HAL driver GPIO 10.0 General-purpose I/O (GPIO) HAL driver HASH 10.0 Hash processor (HASH) HAL driver HSEM 10.0 Hardware semaphore (HSEM) HAL driver I2C 10.0 Inter-integrated circuit (I2C) interface HAL driver IPCC 10.0 Interprocessor Communication I/O (IPCC) HAL driver WDIOS 10.0 Management data input/output (MDIOS) slave HAL driver WMMA 10.0 Master DMA controller (MDMA) HAL driver WR 10.0 Quad Serial peripheral interface (QSPI) HAL driver RCC 10.0 Reset and clock control (RCC) HAL driver	V DMA			1.0.0	DMA controller (DMA) HAL driver
PDCAN 10.0 Hexable DataRate Controller Area Network (FUCAN) HAL driver GPIO 10.0 General-purpose I/O (GPIO) HAL driver HASH 10.0 Hash processor (HASH) HAL driver HSEM 10.0 Hardware semaphore (HSEM) HAL driver IZC 10.0 Inter-integrated circuit (I2C) interface HAL driver IPCC 10.0 Interprocessor Communication I/O (IPCC) HAL driver UPTIM 10.0 Low Power Timer (LPTIM) HAL driver MDIOS 10.0 Master DMA controller (MDMA) HAL driver WR 10.0 Master DMA controller (PWR) HAL driver QSPI 10.0 Reset and clock control (RCC) HAL driver	EXII			1.0.0	External Interrupts (EXTI) HAL driver
GPIO Image: Constraint of the second secon	FDCAN			1.0.0	Hexible DataRate Controller Area Network (FDCAN) HAL driver
HASH I 1.0.0 Hash processor (HASH) HAL driver HSEM I 1.0.0 Hardware semaphore (HSEM) HAL driver I2C 1.0.0 Inter-integrated circuit (I2C) interface HAL driver IPCC 1.0.0 Interprocessor Communication I/O (IPCC) HAL driver MDIOS 1.0.0 Low Power Timer (LPTIM) HAL driver MDIOS 1.0.0 Management data input/output (MDIOS) slave HAL driver WR 1.0.0 Master DMA controller (MDMA) HAL driver QSPI 1.0.0 Quad Serial peripheral interface (QSPI) HAL driver Kalidation Output Description	GPIO			1.0.0	General-purpose I/O (GPIO) HAL driver
HSEM Image: Mark State S	V HASH			1.0.0	Hash processor (HASH) HAL driver
IZC 1.0.0 Inter-integrated circuit (I2C) interface HAL driver IPCC 1.0.0 Interprocessor Communication I/O (IPCC) HAL driver LPTIM 1.0.0 Interprocessor Communication I/O (IPCC) HAL driver MDIOS 1.0.0 Management data input/output (MDIOS) slave HAL driver MDMA 1.0.0 Maser DMA controller (MDMA) HAL driver PWR 1.0.0 Power controller (PWR) HAL driver QSPI 1.0.0 Quad Serial peripheral interface (QSPI) HAL driver RCC 1.0.0 Reset and clock control (RCC) HAL driver	HSEM			1.0.0	Hardware semaphore (HSEM) HAL driver
IPCC 1.0.0 Interprocessor Communication I/O (IPCC) HAL driver LPTIM 1.0.0 Low Power Timer (LPTIM) HAL driver MDIOS 1.0.0 Management data input/output (MDIOS) slave HAL driver MDMA 1.0.0 Master DMA controller (MDMA) HAL driver PWR 1.0.0 Power controller (MDMA) HAL driver QSPI 1.0.0 Quad Serial peripheral interface (QSPI) HAL driver RCC 1.0.0 Reset and clock control (RCC) HAL driver	₩ I2C			1.0.0	Inter-integrated circuit (I2C) interface HAL driver
LPTIM L L L L L C L PTIM L L L L C L PTIM L L L C L Pti L L C L Pti L L C C	IPCC			1.0.0	Interprocessor Communication I/O (IPCC) HAL driver
MDIOS MDMA 10.0 Maagement data input/output (MDIOS) slave HAL driver MDMA 10.0 Master DMA controller (MDMA) HAL driver PWR QSPI 10.0 Quad Serial peripheral interface (QSPI) HAL driver RCC RCC Description				1.0.0	Low Power Timer (LPTIM) HAL driver
MDMA 1.0.0 Master DMA controller (MDMA) HAL driver PWR 1.0.0 Power controller (PWR) HAL driver QSPI 1.0.0 Quad Serial peripheral interface (QSPI) HAL driver RCC 1.0.0 Reset and clock control (RCC) HAL driver	MDIOS			1.0.0	Management data input/output (MDIOS) slave HAL driver
PWR I.0.0 Power controller (PWR) HAL driver QSPI 1.0.0 Quad Serial peripheral interface (QSPI) HAL driver RCC I.0.0 Reset and clock control (RCC) HAL driver Validation Output Description	MDMA			1.0.0	Master DMA controller (MDMA) HAL driver
Image: QSPI 1.0.0 Quad Serial peripheral interface (QSPI) HAL driver RCC 1.0.0 Reset and clock control (RCC) HAL driver	PWR	v		1.0.0	Power controller (PWR) HAL driver
alidation Output Description	QSPI			1.0.0	Quad Serial peripheral interface (QSPI) HAL driver
alidation Output Description	RCC	v		1.0.0	Reset and clock control (RCC) HAL driver
/alidation Output Description					
/alidation Output Description					
	/alidation Output		Description		

 Go to Project – Options for Target (Alt+F7). On the Target tab, set "Use default compiler version 6" as the ARM compiler.

🕅 Options for Target 'Target 1'	
Device Target Output Listing User C/C++ (AC6) A	sm Linker Debug Utilities
STMicroelectronics STM32MP157CAAx:Cortex-M4	Code Generation
<u>X</u> tal (MHz): 12.0	ARM Compiler: Use default compiler version 6
Operating system: None	Use Cross-Module Optimization
System Viewer File:	🔽 Use MicroLIB 🔽 Big Endian
STM32MP1.svd	Floating Point Hardware: Single Precision 💌
Use Custom File	
Read/Only Memory Areas	Read/Write Memory Areas
default off-chip Start Size Startup	default off-chip Start Size NoInit
□ ROM1: ○ ○	□ RAM1: □ □
□ ROM2: ○	□ RAM2: □ □
□ ROM3: □ C	□ RAM3: □ □
on-chip	on-chip
IROM1: 0x10000000 0x20000 €	IRAM1: 0x10020000 0x20000 □
□ IROM2: ○	□ IRAM2: 0x10040000 0x20000 □
OK Ca	ncel Defaults Help

Device Target Output Listing User C/C++	(AC6) Asm Linker Debug Utilities
Lyse Memory Layout from Target Dialog Make RW Sections Position Independent Make RO Sections Position Independent Don't Search Standard Libraries Report 'might fail' Conditions as Errors	X/O Base:
Scatter	Cortex-M4\stm32mp15xx_m4_sram.sct 💽 Edit

On the Linker tab, uncheck "Use Memory Layout from Target Dialog", click on the ellipsis button "..." and select as Scatter File: .\RTE\Device\STM32MP157CAAx_Cortex-M4\stm32mp15xx_m4_sram.sct

On the **Debug** tab, select your debug adapter. Refer to the section Debug connection for more information.

 Add user code templates: Right click on "Source Group 1" and then "Add new Item to Source Group 1".
 Add "Lock Resource" from Device:Startup component.

Add New Item to	Group 'Source Gro	oup 1'		×
		Add template file(s) to the	project.	
C+++ File (.c	:pp)	Component	Name	
A Asm File (.s	3)	Device	Exception Handlers and Peripheral IRQ	
h Header File	e (.h)	Startup	Lock Resource	
Text File (+	vt)	Startup	MCU Support Package	
Image File	(.*) Template			
Туре:	User Code Templa	ite		
Name:	lock_resource.c			
Location:	C:\Temp\Projects	\Blinky		
		Add	Close	Help

 Add main file: Right click on "Source Group 1" and then "Add new Item to Source Group 1". Add "Main file" from Device:Startup component.

Add New Iter	m to Group 'Source G	roup 1'		
	(c)	Add template file(s) to th	e project.	
		Component	Name	
C++ H	le (.cpp)	🕀 🗇 CMSIS		
A Asm Fi	le (.s)	🖃 💎 Device		
		Startup	Exception Handlers and Peripheral IR	RQ.
h Heade	r File (.h)	Startup	Lock Resource	
B		Startup	MCU Support Package	
lext H	ile (.txt)	Startup	Main file	
Image	File (.*)			
User C	ode Template			
Type:	User Code Temp	late		
Name:	main.c main.h			
Location:	C:\Temp\Project	slBlinky		
		Add	Close	Help

In the main.c file add the following lines into the respective sections:

```
Includes
#ifdef _RTE_
#include "RTE_Components.h" // Component selection
#endif
#ifdef RTE_CMSIS_RTOS2 // when RTE component CMSIS RTOS2 is
used
#include "cmsis_os2.h" // :CMSIS:RTOS2
#endif
```

Function prototypes

extern void app main (void *arg);

Kernel initialization, before while loop in the main function

```
#ifdef RTE_CMSIS_RTOS2
    /* Initialize CMSIS-RTOS2 */
    osKernelInitialize ();
    /* Create application main thread */
    osThreadNew(app_main, NULL, NULL);
    /* Start thread execution */
    osKernelStart();
#endif
```

7. Add a new Blinky.c file with the following content:

```
* Name:
         Blinky.c
 * Purpose: LED Flasher
 * This file is part of the uVision/ARM development tools.
 * This software may only be used under the terms of a valid, current,
 * end user licence from KEIL for a compatible version of KEIL software
 * development tools. Nothing else gives you the right to use this software.
 * This software is supplied "AS IS" without warranties of any kind.
 *
 * Copyright (c) 2019 Keil - An ARM Company. All rights reserved.
 *-----*/
#include <stdio.h>
#include "main.h"
#include "Board_LED.h" /* :Board Support:LED */
#include "Board_Buttons.h" /* :Board Support:Buttons */
#include "cmsis_os2.h" /* :CMSIS:RTOS2 */
#include "RTE_Components.h"
                                  /* Component selection */
#include CMSIS device header
#ifdef RTE Compiler EventRecorder
#include "EventRecorder.h"
#endif
static osThreadId_t tid_thrLED; /* Thread id of thread: LED */
static osThreadId_t tid_thrBUT; /* Thread id of thread: BUT */
/*_____
 thrLED: blink LED
 *-----*/
 NO RETURN static void thrLED(void *argument) {
uint32_t led_max = LED_GetCount();
uint32_t led_num = OU;
 (void) argument;
 printf("thrLED is running\n");
 for (;;) {
   osThreadFlagsWait(0x0001U, osFlagsWaitAny ,osWaitForever);
   LED On (led num);
                                         /* Turn specified LED on */
   printf("Blink LED number %d\n", led num);
   osThreadFlagsWait(0x0001U, osFlagsWaitAny ,osWaitForever);
   LED Off(led num);
                                         /* Turn specified LED off */
   led num++;
                                        /* Change LED number */
   if (led num >= led max) {
     led num = 0U;
                                        /* Restart with first LED */
   }
 }
}
```

```
AN319 – Using STM32MP1 Cortex-M with Keil MDK
```

```
*_____
 thrBUT: check button state
                       _____
                                        _____*/
 NO RETURN static void thrBUT (void *argument) {
 uint32 t button msk = (1U << Buttons GetCount()) - 1U;
 (void) argument;
 printf("thrBUT is running\n");
 for (;;) {
   osDelay(100U);
                                        /* Wait */
   if (Buttons GetState() & (button msk)) printf("Button is pressed\n");
   while (Buttons GetState() & (button msk)); /* Wait while holding user
button */
   osThreadFlagsSet(tid thrLED, 0x0001U);
 }
}
                    -----
/*___
 * Application main thread
                     ------
 NO RETURN void app main (void *argument) {
 (void) argument;
                                        /* initialize LEDs */
 LED Initialize();
 Buttons Initialize();
                                        /* initialize Buttons */
 tid thrBUT = osThreadNew (thrBUT, NULL, NULL); /* create BUT thread */
 if (tid thrBUT == NULL) { /* add error handling */ }
 tid thrLED = osThreadNew (thrLED, NULL, NULL); /* create LED thread */
 if (tid thrLED == NULL) { /* add error handling */ }
 for (;;) {}
```

 To see threads switching in the System Analyzer window, enable Event Recorder for RTX5. In the **Project** window, under CMSIS, double-click the "RTX_Config.h" file.

In Configuration Wizard mode, open "Event Recorder Configuration" and enable "Global Initlialization":

9. Build the target (F7). The program is compiled without errors or warnings.



 Start debugging: if you have not yet selected your debug adapter in your μVision project, refer to the section Debug connection how to do so. When done, go to Debug – Start/Stop Debug Session or press Ctrl+F5.

The application is loaded by the debug adapter into the RAM memory space specified in the linker scatter file. Run and debug as usual, using μ Vision features like System Analyzer and Event Recorder as illustrated here:



Creating an OpenAMP project for production mode

In *production mode*, the OpenSTLinux runs on the CA7 as master and is responsible to load the application and to start the CM4 as coprocessor, using the remoteproc framework. MDK integrates such remote coprocessor commands in the debug engine, so the developer can load, run and debug the CM4 application as usual.

This example demonstrates the inter-processor communication between the CM4 and the CA7 cores. It uses a STMicroelectronics variant of the OpenAMP distribution.

What the example does:

- CM4 initializes the OpenAMP component, configures IPCC peripheral through HAL and sets up the openamp-rpmsg framework infrastructure, which is the first level of communication between CA7 and CM4.
- CM4 creates two rpmsg channels for two virtual UART instances.
- CM4 waits for messages from the master core CA7 on these two channels.
- When the CM4 receives a message on one virtual UART instance, it sends the message back to CA7 on the same virtual UART instance.

For more info about STM32MPU coprocessor handling, resource management, RPMsg and remoteproc framework, please refer to <u>Coprocessor management Linux</u> and <u>Coprocessor management STM32Cube</u>.

Project creation step by step

- In μVision, select Project New μVision Project... and give a name to the new project.
- From the device list, select STMicroelectronics STM32MP157CAAx:Cortex-M4 device.

Software Packs Vendor: STMicroelectronics Device: STM32MP157CAAx:Cortex-M4 Toolset: ARM Search:	Device			
Vendor: STM32MP157CAAx:Cotex-M4 Device: STM32MP157CAAx:Cotex-M4 Toolset: ARM Search: Description: Image: STM32MP157 Image: STM32MP157 Image: STM32MP157 Image: STM32MP157AAx Image: STM32MP157AABx Image: STM32MP157AAAx Image: STM32MP157AADx Image: STM32MP157AADx Image: STM32MP157CAAx Image: STM32MP157CAAx Image: STM32MP157CABx Image: STM32MP157CAAx </td <td></td> <td>Software Packs</td> <td>T</td> <td></td>		Software Packs	T	
Desgription: STM32MP1 Series STM32MP157 STM32MP157 STM32MP157 STM32MP157AAAx STM32MP157AAAx STM32MP157AAAx STM32MP157AAAx STM32MP157AAAx STM32MP157AAAx STM32MP157AAAx STM32MP157AAAx STM32MP157AAAx STM32MP157CAAx STM32MP157CAAx STM32MP157CAAx STM32MP157CAAx STM32MP157CAAx STM32MP157CAAx STM32MP157CAAx STM32MP157CAAx STM32MP157CAAx	Vendor: Device: Toolset: Search:	STMicroelectronics STM32MP157CAAx:Cortex-M4 ARM		
Image: STM32MP1 Series Image: STM32MP157 Image: STM32MP157 STM32MP157AAx Image: STM32MP157AAx STM32MP157AAx Image: STM32MP157AAx Image: STM32MP157AAx Image: STM32MP157CAAx Image: STM32MP157CAAx Ima		,	Des <u>c</u> ription:	
	*** ST ***** * * * *	M32MP1 Series STM32MP157 STM32MP157 STM32MP157AAAx STM32MP157AABx STM32MP157AACx STM32MP157AADx STM32MP157CAAx STM32MP157CAAx STM32MP157CABx V	Am based dual Cortex-A7 650 MHz + Cortex-M4 MPU 32-bit dual-core Am Cortex-A7 Up to 650 MHz (Up to 4158 CoreMark) L1 32 Kbyte I / 32 Kbyte D for each core 256 Kbyte unified level 2 cache Am NEON and Am TrustZone 32-bit Am Cortex-M4 with FPU/MPU Up to 200 MHz (Up to 673 CoreMark) External DDR memory up to 1 Gbyte 708 Kbyte of internal SRAM: 256 KB of AXI	

- 3. In the Manage Run-Time Environment window, select the following components:
 - AMP:RPMSG
 - CMSIS:CORE
 - CMSIS:RTOS2 (API):Keil RTX5 (Variant Source)
 - Compiler:Event Recorder
 - Compiler:IO:STDOUT (Variant ITM)
 - Device:Startup
 - Device:STM32Cube HAL:Common
 - Device:STM32Cube HAL:Cortex
 - Device:STM32Cube HAL:GPIO
 - Device:STM32Cube HAL:HSEM
 - Device:STM32Cube HAL:IPCC
 - Device:STM32Cube HAL:PWR
 - Device:STM32Cube HAL:RCC

oftware Component	Sel.	variant	V	version	Description	
AMP					Asymmetric Multiprocessing	
RPMSG	v	OpenAMP	1.	.0.0	Remote Processor Messaging - OpenAMP	
💠 Board Support		STM32MP157C-E	V1 1	1.0.0	STM32MP157C-EV1 Evaluation Board	
💠 C Library						
CMSIS					Cortex Microcontroller Software Interface Components	
CORE	~		~ 5	5.1.3	CMSIS-CORE for Cortex-M, SC000, SC300, ARMv8-M, ARMv8.1-M	
DSP	Ē		1	1.5.2	CMSIS-DSP Library for Cortex-M, SC000, and SC300	
NN Lib	Ē		1.	1.1.0	CMSIS-NN Neural Network Library	
🗉 🚸 RTOS (API)			1.	1.0.0	CMSIS-RTOS API for Cortex-M, SC000, and SC300	
🗉 🚸 RTOS2 (API)			2	2.1.3	CMSIS-RTOS API for Cortex-M, SC000, and SC300	
FreeRTOS			1	0.0.1	CMSIS-RTOS2 implementation for Cortex-M based on FreeRTOS	
Keil FuSa RTX5 noMPU	<u> </u>	Source	5	5.5.0	CMSIS-RTOS2 Functional Saftey RTX5 for Cortex-M4 without MPU support (Source)	
Keil RTX5		Source	~ 5	5.5.0	CMSIS-RTOS2 RTX5 for Cortex-M. SC000, C300 and Army8-M (Source)	
CMSIS Driver			-		Unified Device Drivers compliant to CMSIS-Driver Specifications	
CMSIS Driver Validation		ΔΡΙ	1	0.0	Run API test for enabled drivers	
CMSIS BTOS Validation					CMSIS-RTOS Validation Suite	
CMSIS RTOS2 Validation					CMSIS-RTOS2 Validation Suite	-
Compiler		ARM Compiler	× 1	.6.0	Compiler Extensions for ARM Compiler 5 and ARM Compiler 6	-
Event Recorder	V		1	40	Event Recording and Component Viewer via Debug Access Port (DAP)	
	1.	DAP			Potarget Input/Output	
		Eile Sustem	1	20	Use retargeting together with the File System component	
		Prie System Deselvasiat	1	1.2.0	Ose retargeting together with the File System component	
		Breakpoint	× 1.	1.2.0	Stop program execution at a breakpoint when using STDERK	
		breakpoint	× 1.	1.2.0	Stop program execution at a breakpoint when using STDIN	
	V	IIM D. L. S.	~ 1	1.2.0	Redirect STDOUT to a debug output window using ITM	
······································		Breakpoint	\sim	1.2.0	Stop program execution at a breakpoint when using 11Y	
Vevice	_				Startup, System Setup	
Startup	V		1.	1.0.0	System Startup for STMicroelectronics STM32MP157 Series	
STM32Cube HAL	_				STM32F2xx Hardware Abstraction Layer (HAL) Drivers	
ADC			1.	1.0.0	Analog-to-digital converter (ADC) HAL driver	
CEC			1.	1.0.0	Consumer Electronics Control (CEC) HAL driver	
CRC			1.	1.0.0	CRC calculation unit (CRC) HAL driver	
CRYP			1.	1.0.0	Cryptographic processor (CRYP) HAL driver	
Common	~		1.	1.0.0	Common HAL driver	
Cortex	~		1.	1.0.0	Cortex HAL driver	
DAC			1.	1.0.0	Digital-to-analog converter (DAC) HAL driver	
DCMI			1.	1.0.0	Digital camera interface (DCMI) HAL driver	
DFSDM			1	1.0.0	Digital Filter for Sigma-Delta Modulators (DFSDM) HAL driver	
DMA			1	1.0.0	DMA controller (DMA) HAL driver	
exti			1.	1.0.0	External Interrupts (EXTI) HAL driver	
FDCAN			1.	.0.0	Flexible DataRate Controller Area Network (FDCAN) HAL driver	
GPIO	v		1.	.0.0	General-purpose I/O (GPIO) HAL driver	
ASH			1.	.0.0	Hash processor (HASH) HAL driver	
HSEM	~		1.	.0.0	Hardware semaphore (HSEM) HAL driver	
Ø 12C			1.	.0.0	Inter-integrated circuit (I2C) interface HAL driver	
PCC	~		1	.0.0	Interprocessor Communication I/O (IPCC) HAL driver	
PTIM			1	.0.0	Low Power Timer (LPTIM) HAL driver	
MDIOS			1	.0.0	Management data input/output (MDIOS) slave HAL driver	
MDMA			1.	1.0.0	Master DMA controller (MDMA) HAL driver	-
	~		1.	1.0.0	Power controller (PWR) HAL driver	-
QSPI			1.	1.0.0	Quad Serial peripheral interface (QSPI) HAL driver	_
RCC	~		1	.0.0	Reset and clock control (RCC) HAL driver	
RNG			1	1.0.0	Random number generator (RNG) HAL driver	
lidation Output		Description				T
indation Output		Description				

4. Configure RTX5: open the RTXConfig.h file under CMSIS component, expand the "Thread Configuration" option and change the "Default Thread Stack size" to 512. Expand Event Recorder Configuration and enable "Global Initialization":



5. Open the menu Project – Options for Target (Alt+F7). On the **Target** tab, set "Use default compiler version 6" as the ARM Compiler:

Options for Target 'Target 1'	×
Device Target Output Listing User C/C++ (AC6) A	am Linker Debug Utilities
STMicroelectronics STM32MP157CAAx:Cortex-M4	Code Generation
Xtal (MHz): 12.0	ARM Compiler: Use default compiler version 6

On the C/C++ (AC6) tab, add the following preprocessor defines:
 _LOG_TRACE_IO_NO_ATOMIC_64_SUPPORT METAL_INTERNAL METAL_MAX_DEVICE_REGIONS=2



7. On the **Linker** tab, uncheck "Use Memory Layout from Target Dialog", click on the ellipsis button "..." and select as Scatter File:

\RTE\Device\STM32MP157CAAx_	Cortex-M4\stm	32mp15xx_m4	_sram.sct

vice Targ	et Output Listing User C/C++ (AC6) A	sm Linker De	bug Utilities	
Use Men	nory Layout from Target Dialog RW Sections Position Independent RO Sections Position Independent Search Standard Libraries	X/O Base: R/O Base: R/W Base	0x10000000	
Repo	rt 'might fail' Conditions as Errors	disable warnings:	1	
Scatter File	rt 'might fail' Conditions as Errors	oisable vvamings: 4\stm32mp15xx_m4	l_sram.sct ▼	Edit
Repo	rt 'might fail' Conditions as Errors	aisabie warnings: 4\stm32mp15xx_m4	I_sram.sct ▼	Edit

Copyright © 2019 Arm Ltd. All rights reserved www.keil.com/appnotes/docs/apnt_319.asp

AN319 – Using STM32MP1 Cortex-M with Keil MDK

8. On the **Debug** tab: select your debug adapter, uncheck "Load Application at Startup", click on the ellipsis button "..." and type "debug.ini" to create a new Initialization File:



Click on "Edit..." to open the debug.ini file and add the following line:

```
LOAD %L NOCODE INCREMENTAL
```

9. Edit the linker scatter file to add Resource Table, OpenAMP, and Event Recorder buffer sections:

```
LR VECTORS 0x0000000 0x00000400 {
                                            ; Vector table
  .isr vector +0 {
   startup*.o (RESET, +First)
  }
}
LR IROM1 0x1000000 0x00060000 {
                                            ; SRAM
  ER IROM1 0x10000000 0x00020000 {
                                           ; Code
    *(InRoot$$Sections)
    .ANY (+RO)
    .ANY (+XO)
  }
 RW IRAM1 0x10020000 0x00020000 {
                                           ; Data
    .ANY (+RW +ZI)
  }
  .resource table +0 ALIGN 4 {
                                            ; Resource Table
   rsc table.o (+RW +ZI)
  }
 RW OPENAMP 0x10040000 EMPTY 0x00010000 { ; OpenAMP Buffer
  }
 RW ER 0x10050000 UNINIT 0x00010000 { ; Event Recorder
   EventRecorder.o (+ZI)
  }
}
```

10. Add user code templates: Right click on "Source Group 1" and then "Add new Item to Source Group 1":



Add the all available templates from the Device:Startup component.

11. Add the IPCC functions to the stm32mp1xx_hal_msp.c file as reported below.

```
/**
* @brief IPCC MSP Initialization
* This function configures the hardware resources used in this example
* @param hipcc: IPCC handle pointer
* @retval None
*/
void HAL IPCC MspInit(IPCC HandleTypeDef* hipcc)
{
  if (hipcc->Instance==IPCC)
  /* USER CODE BEGIN IPCC MspInit 0 */
  /* USER CODE END IPCC MspInit 0 */
    /* Peripheral clock enable */
     HAL RCC IPCC CLK ENABLE();
  /* IPCC interrupt Init */
   HAL NVIC SetPriority (IPCC RX1 IRQn, 0, 0);
   HAL NVIC EnableIRQ(IPCC RX1 IRQn);
  /* USER CODE BEGIN IPCC MspInit 1 */
  /* USER CODE END IPCC MspInit 1 */
}
}
/**
* @brief IPCC MSP De-Initialization
* This function freeze the hardware resources used in this example
* @param hipcc: IPCC handle pointer
* @retval None
*/
void HAL IPCC MspDeInit(IPCC HandleTypeDef* hipcc)
{
  if(hipcc->Instance==IPCC)
  /* USER CODE BEGIN IPCC MspDeInit 0 */
  /* USER CODE END IPCC MspDeInit 0 */
   /* Peripheral clock disable */
    /* IPCC interrupt DeInit */
   HAL NVIC DisableIRQ(IPCC RX1 IRQn);
  /* USER CODE BEGIN IPCC MspDeInit 1 */
  /* USER CODE END IPCC MspDeInit 1 */
  }
}
```

```
AN319 – Using STM32MP1 Cortex-M with Keil MDK
```

12. The interrupt handlers SVC_Handler, PendSV_Handler and SysTick_Handler are part of the RTX5 implementation and shall be removed from the stm32mp1xx_it.c file. Add the IPCC handlers as reported below.

```
extern IPCC_HandleTypeDef hipcc;
void IPCC_RX1_IRQHandler(void)
{
    HAL_IPCC_RX_IRQHandler(&hipcc);
}
void IPCC_TX1_IRQHandler(void)
{
    HAL_IPCC_TX_IRQHandler(&hipcc);
}
```

13. In the main.c file add the following lines to the respective sections:

Includes

```
#ifdef _RTE_
#include "RTE_Components.h" // Component selection
#endif
#ifdef RTE_CMSIS_RTOS2 // when RTE component CMSIS RTOS2 is
used
#include "cmsis_os2.h" // :CMSIS:RTOS2
#endif
#include "virt uart.h" // ARM:AMP:RPMSG
```

Defines

#define MAX_BUFFER_SIZE RPMSG_BUFFER_SIZE

Variables

```
IPCC_HandleTypeDef hipcc;
VIRT_UART_HandleTypeDef huart0;
VIRT_UART_HandleTypeDef huart1;
____IO FlagStatus VirtUart0RxMsg = RESET;
uint8_t VirtUart0ChannelBuffRx[MAX_BUFFER_SIZE];
uint16_t VirtUart0ChannelRxSize = 0;
____IO FlagStatus VirtUart1RxMsg = RESET;
uint8_t VirtUart1ChannelBuffRx[MAX_BUFFER_SIZE];
uint16_t VirtUart1ChannelBuffRx[MAX_BUFFER_SIZE];
```

Function prototypes

```
extern void app_main (void *arg);
static void MX_IPCC_Init(void);
void VIRT_UART0_RxCpltCallback(VIRT_UART_HandleTypeDef *huart);
void VIRT_UART1_RxCpltCallback(VIRT_UART_HandleTypeDef *huart);
```

Kernel initialization, before while loop in the main function

```
#ifdef RTE_CMSIS_RTOS2
    /* Initialize CMSIS-RTOS2 */
    osKernelInitialize ();
    /* Create application main thread */
    osThreadNew(app_main, NULL, NULL);
    /* Start thread execution */
    osKernelStart();
#endif
```

Main thread

```
/*-----
 * Application main thread
 *-----*/
void app main (void *argument) {
 /* IPCC initialisation */
 MX IPCC Init();
 /* OpenAmp initialisation */
 MX OPENAMP Init (RPMSG REMOTE, NULL);
 /*
  * Create Virtual UART device
  * defined by a rpmsg channel attached to the remote device
  */
 log info("Virtual UART0 OpenAMP-rpmsg channel creation\r\n");
 if (VIRT UART Init(&huart0) != VIRT UART OK) {
   log err("VIRT UART Init UART0 failed.\r\n");
   Error Handler();
 }
 log info("Virtual UART1 OpenAMP-rpmsg channel creation\r\n");
 if (VIRT UART Init(&huart1) != VIRT UART OK) {
   log err("VIRT UART Init UART1 failed.\r\n");
   Error Handler();
 }
 /*Need to register callback for message reception by channels*/
 if(VIRT UART RegisterCallback(&huart0, VIRT UART RXCPLT CB ID,
VIRT UARTO RxCpltCallback) != VIRT UART OK)
 {
  Error Handler();
 }
 if (VIRT UART RegisterCallback (& huart1, VIRT UART RXCPLT CB ID,
VIRT UART1 RxCpltCallback) != VIRT UART OK)
 {
   Error Handler();
 }
 /* Infinite loop */
 while (1)
```

AN319 – Using STM32MP1 Cortex-M with Keil MDK

```
{
    OPENAMP_check_for_message();
    if (VirtUart0RxMsg) {
        VirtUart0RxMsg = RESET;
        VIRT_UART_Transmit(&huart0, VirtUart0ChannelBuffRx,
VirtUart0ChannelRxSize);
    }
    if (VirtUart1RxMsg) {
        VirtUart1RxMsg = RESET;
        VIRT_UART_Transmit(&huart1, VirtUart1ChannelBuffRx,
VirtUart1ChannelRxSize);
    }
    osDelay(100);
    }
}
```

IPCC Initialization and Virtual UART callbacks

```
/**
  * @brief IPPC Initialization Function
  * @param None
  * @retval None
  */
static void MX IPCC Init(void)
{
 hipcc.Instance = IPCC;
 if (HAL IPCC Init(&hipcc) != HAL OK)
  ſ
    Error Handler();
  }
}
void VIRT UART0 RxCpltCallback (VIRT UART HandleTypeDef *huart)
{
    log info("Msg received on VIRTUAL UART0 channel: s \n\r", (char *)
huart->pRxBuffPtr);
    /* copy received msg in a variable to sent it back to master processor
in main infinite loop*/
    VirtUart0ChannelRxSize = huart->RxXferSize < MAX BUFFER SIZE? huart-
>RxXferSize : MAX BUFFER SIZE-1;
    memcpy(VirtUart0ChannelBuffRx, huart->pRxBuffPtr,
VirtUart0ChannelRxSize);
   VirtUart0RxMsg = SET;
}
void VIRT UART1 RxCpltCallback (VIRT UART HandleTypeDef *huart)
{
    log info("Msg received on VIRTUAL UART1 channel: s \n\r", (char *)
huart->pRxBuffPtr);
    /* copy received msg in a variable to sent it back to master processor
in main infinite loop*/
    VirtUart1ChannelRxSize = huart->RxXferSize < MAX BUFFER SIZE? huart-
>RxXferSize : MAX BUFFER SIZE-1;
   memcpy(VirtUart1ChannelBuffRx, huart->pRxBuffPtr,
VirtUart1ChannelRxSize);
   VirtUart1RxMsg = SET;
}
```

14. Build the target (F7). The program compiles without errors or warnings.

15. Before connecting to the target, open a terminal viewer (such as PuTTY or TeraTerm) and connect to the COM port that is provided by the ST-LINK (check Windows Device Manager), using 115200 Baud. Once Linux is up and running, use the command ifconfig to get the IP address of the device:

<pre>root@stm32mp1:~# ifconfig</pre>
eth0 Link encap:Ethernet HWaddr 00:80:FF:FF:FF:A5
inet addr:192.168.0.10 Bcast:192.168.0.255 Mask:255.255.255.0
<pre>inet6 addr: fe80:ff0:e1ff:feef:45a5/64 Scope:Link</pre>
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:35 errors:0 dropped:0 overruns:0 frame:0
TX packets:52 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:7081 (6.9 KiB) TX bytes:8135 (7.9 KiB)
<pre>Interrupt:63 Base address:0xc000</pre>

You'll need this address in the next step, to communicate with the target.

16. Configure the production.config file: set BOARD_HOSTNAME with the board's hostname or IP address.



17. Start debugging: if you have not yet selected your debug adapter in your μVision project, refer to the section Debug connection how to do so.

If you are using ST-Link, go to Tools – STM32MP1 Load and Run.

For ULINKpro, ULINKplus, ULINK2 and CMSIS-DAP debug adapters, this command is run using debug sequences, as configured earlier. Go to **Debug – Start/Stop Debug Session** or press Ctrl+F5. The Linux console should show the following message:

remoteproc remoteproc0: remote processor m4 is now up

At this point, the image has been loaded to the embedded Linux file system. The remoteproc commands load the application into the specified memory location (as stated in the scatter file) and start the CM4. The debug symbols have been loaded as well and the debug adapter is connected to the target, so that you can run and debug as usual.

Run the following script in Linux to send messages every second:

```
if [[ $(ls -A /dev/ttyRPMSG*) ]]; then
   stty -onlcr -echo -F /dev/ttyRPMSG0
   cat /dev/ttyRPMSG0 &
   stty -onlcr -echo -F /dev/ttyRPMSG1
   cat /dev/ttyRPMSG1 &
fi

while [[ $(ls -A /dev/ttyRPMSG*) ]]; do
   echo "Hello Virtual UART0" >/dev/ttyRPMSG0
   echo "Hello Virtual UART1" >/dev/ttyRPMSG1
   sleep 1
done
```

Use System Analyzer and Event Recorder to keep track of RTX events and OpenAMP messages:



Useful links

- <u>STM32 Arm[®] Cortex[®]-based MPUs user guide</u>
- <u>Reference Manual STM32MP157</u>
- Programming Manual STM32
- Datasheet STM32MP157C
- Errata STM32MP151x/3x/7x
- OpenAMP workgroup