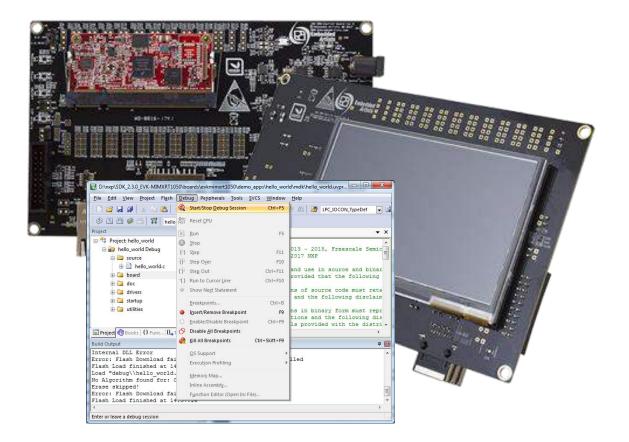
iMX RT1052/1062 Developer's Kit - Program Dev. Guide

Copyright 2022 © Embedded Artists AB

# iMX RT1052/1062 Developer's Kit Program Development Guide



Get Up-and-Running Quickly and Start Developing Your Application On Day 1!



### **Embedded Artists AB**

Rundelsgatan 14 211 36 Malmö Sweden

http://www.EmbeddedArtists.com

#### Copyright 2022 © Embedded Artists AB. All rights reserved.

No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual or otherwise, without the prior written permission of Embedded Artists AB.

### Disclaimer

Embedded Artists AB makes no representation or warranties with respect to the contents hereof and specifically disclaim any implied warranties or merchantability or fitness for any particular purpose. Information in this publication is subject to change without notice and does not represent a commitment on the part of Embedded Artists AB.

#### Feedback

We appreciate any feedback you may have for improvements on this document. Please send your comments to support@EmbeddedArtists.com.

### Trademarks

All brand and product names mentioned herein are trademarks, services marks, registered trademarks, or registered service marks of their respective owners and should be treated as such.

## **Table of Contents**

1	Document Revision History	5
2	Get Started with Program Development	6
2.1	Downloading the SDK	6
2.2	About the SDK	6
2.3	Getting Started with a Specific IDE	7
3	Getting Started with Keil uVision/MDK and SDK	8
3.1	Install the SDK	8
3.2	Install CMSIS Device Pack	8
3.3	Build an Example Application	9
3.4	Run an Example Application	10
3.5	Selecting Build Target	12
4	Getting Started with Keil uVision/MDK and RTE	13
4.1	Install CMSIS Device Pack	13
4.2	Build an Example Application	14
4.3	Run an Example Application	16
4.4	Selecting Build Target	18
4.5	Configuring Flash Algorithm	19
4.6	Working with RTE	20
5	Getting Started with IAR Embedded Workbench	21
5 5.1	Getting Started with IAR Embedded Workbench Install the SDK	21 21
-	•	
5.1	Install the SDK	21
5.1 5.2	Install the SDK Build an Example Application	21 21
5.1 5.2 5.3	Install the SDK Build an Example Application Run an Example Application	21 21 23
5.1 5.2 5.3 <b>6</b>	Install the SDK Build an Example Application Run an Example Application Getting Started with NXP MCUXpresso IDE	21 21 23 <b>24</b>
5.1 5.2 5.3 <b>6</b> 6.1	Install the SDK Build an Example Application Run an Example Application Getting Started with NXP MCUXpresso IDE Install the SDK	21 21 23 <b>24</b> 24
5.1 5.2 5.3 <b>6</b> 6.1 6.2	Install the SDK Build an Example Application Run an Example Application Getting Started with NXP MCUXpresso IDE Install the SDK Build an Example Application	21 21 23 <b>24</b> 24 24
5.1 5.2 5.3 <b>6</b> 6.1 6.2 6.3	Install the SDK Build an Example Application Run an Example Application Getting Started with NXP MCUXpresso IDE Install the SDK Build an Example Application Run an Example Application	21 21 23 <b>24</b> 24 24 24 24 27
5.1 5.2 5.3 <b>6</b> 6.1 6.2 6.3 6.4	Install the SDK Build an Example Application Run an Example Application Getting Started with NXP MCUXpresso IDE Install the SDK Build an Example Application Run an Example Application Target Memory	21 21 23 <b>24</b> 24 24 27 28
5.1 5.2 5.3 6 6.1 6.2 6.3 6.4 7 7.1 7.1.	Install the SDK Build an Example Application Run an Example Application Getting Started with NXP MCUXpresso IDE Install the SDK Build an Example Application Run an Example Application Target Memory Debug Interface J-LINK/J-TRACE Support 1 Install J-LINK Software	21 21 23 24 24 24 24 27 28 31 32 32
5.1 5.2 5.3 6 6.1 6.2 6.3 6.4 7 7.1 7.1. 7.1.	Install the SDK Build an Example Application Run an Example Application Getting Started with NXP MCUXpresso IDE Install the SDK Build an Example Application Run an Example Application Target Memory Debug Interface J-LINK/J-TRACE Support 1 Install J-LINK Software 2 MCUXpresso	21 21 23 24 24 24 24 27 28 31 32 32 32
5.1 5.2 5.3 6 6.1 6.2 6.3 6.4 7 7.1 7.1. 7.1. 7.1.	Install the SDK Build an Example Application Run an Example Application Getting Started with NXP MCUXpresso IDE Install the SDK Build an Example Application Run an Example Application Target Memory Debug Interface J-LINK/J-TRACE Support 1 Install J-LINK Software 2 MCUXpresso 3 Keil uVision	21 21 23 24 24 24 24 27 28 31 32 32 32 32
5.1 5.2 5.3 6 6.1 6.2 6.3 6.4 7 7.1 7.1. 7.1.	Install the SDK Build an Example Application Run an Example Application Getting Started with NXP MCUXpresso IDE Install the SDK Build an Example Application Run an Example Application Target Memory Debug Interface J-LINK/J-TRACE Support 1 Install J-LINK Software 2 MCUXpresso 3 Keil uVision	21 21 23 24 24 24 24 27 28 31 32 32 32
5.1 5.2 5.3 6 6.1 6.2 6.3 6.4 7 7.1 7.1. 7.1. 7.1. 7.2 8	Install the SDK Build an Example Application Run an Example Application Getting Started with NXP MCUXpresso IDE Install the SDK Build an Example Application Run an Example Application Target Memory Debug Interface J-LINK/J-TRACE Support 1 Install J-LINK Software 2 MCUXpresso 3 Keil uVision Debug Troubleshooting Standalone Program Download	21 21 23 24 24 24 24 27 28 31 32 32 32 32 32 32 34 35
5.1 5.2 5.3 6 6.1 6.2 6.3 6.4 7 7.1 7.1. 7.1. 7.1. 7.1. 7.2	Install the SDK Build an Example Application Run an Example Application Getting Started with NXP MCUXpresso IDE Install the SDK Build an Example Application Run an Example Application Target Memory Debug Interface J-LINK/J-TRACE Support 1 Install J-LINK Software 2 MCUXpresso 3 Keil uVision	21 21 23 24 24 24 27 28 31 32 32 32 32 34

8.3	Booting an Unsigned Image	36
8.4	Booting an Authenticated or Encrypted Image	37
9	Terminal Application Setup	38
9.1	UART-to-USB Bridge	38
9.2	Terminal Application on the PC	38
9.2.1	Tera Term Terminal Emulation Application	39
9.2.2	PuTTY terminal emulation application	40
10	Booting from External Memory	41
11	Things to Note	44
11.1	ESD Precaution	44
11.2	General Handling Care	44
11.3	OTP Fuse Programming	44
11.4	Further Information	45
12	Disclaimers	46
12.1	Definition of Document Status	47

# **1** Document Revision History

Revision	Date	Description
A	2020-01-31	Updated to match SDK 2.7.0. Added information about MCUXpresso Secure Provisioning Tools. Updated example descriptions and patches.
A1	2020-05-13	Updated information in section 2.3.1 regarding clocking and the new patched file.
В	2020-12-18	Document updated to match the new pre-patched SDK (2.8.6) that is provided.
B1	2022-11-18	Updated to include information about iMX RT1062 OEM board rev C1. Updated Secure Provisioning to handle v5 of the tool.

### 2 Get Started with Program Development

This chapter contains information about how to get started with program development on the *iMX RT1052 Developer's Kit* and the *iMX RT1062 Developer's Kit*. Since the kits are almost identical from a program development perspective, for the rest of this document the name *iMX RT Developer's Kit* will be used. Only when there is a difference will the individual *iMX RT Developer's Kit* be referenced.

The document also contains information about program download, i.e., how to flash the OctalSPI/QuadSPI flash memory.

To start program development, you need the following things, all of them:

- Patched version of MCUXpresso SDK this is a package of sample software from NXP that has been patched by Embedded Artists to work with the *iMX RT Developer's Kit*. The zip-file can be downloaded from http://imx.embeddedartists.com.
- 2. Integrated Development Environment (IDE)
  - a. NXP MCUXpresso, Keil uVision/MDK and IAR Embedded Workbench supports direct flash programming of the OctalSPI/QuadSPI. Update to latest version of respective IDE.
  - b. Programming of the OctalSPI/QuadSPI flash is also supported via NXP's MCUXpresso Secure Provisioning Tools standalone application but it is not a suitable tool to use during program development.
- 3. **JTAG probe** to be able to download the application to SRAM, SDRAM or the OctalSPI/QuadSPI flash memory and in general to be able to debug set breakpoints, inspect memory, etc.
  - The low-cost MCU-Link or LPC-Link2 are excellent choices. Keil ULINK2 and ULINKplus, as well as Segger JLINK, are also excellent debug probes.
  - b. Technically it is possible to program/flash the OctalSPI/QuadSPI without a JTAG probe (via NXP's MCUXpresso Secure Provisioning Tools application), but it is strongly recommended to use the proper tool for debugging - i.e., use a JTAG probe!
- 4. And of course, the *iMX RT Developer's Kit*!

#### 2.1 Downloading the SDK

Starting with SDK version 2.8.6 and going forward, Embedded Artists has published a version of the SDK that has already been patched to work with the *iMX RT Developer's Kit*. The file can be downloaded from http://imx.embeddedartists.com and will have a filename like

eaimxrt1052 sdk <version> freertos <date>.zip, Of

eaimxrt1062\_4mb\_sdk\_<version>\_<date>.zip

eaimxrt1062\_16mb\_sdk\_<version>\_<date>.zip

#### 2.2 About the SDK

NXP's SDK builder (https://mcuxpresso.nxp.com/en/) can be used to generate SDKs for:

- Windows, MAC or Linux
- FreeRTOS, AzureRTOS or bare metal (i.e., no RTOS)
- With all examples or a subset

 For all compilers/IDEs or for one of (MCUXpresso, ARM GCC, Keil uVision/MDK, IAR Embedded Workbench)

Depending on which options are selected it limits the number of examples that can be downloaded. The SDK with the broadest set of examples is the **Windows/All IDEs/All Examples/FreeRTOS** combination and that is the combination that was selected for the patched SDK.

This is what has been patched:

- Set CPU speed according to Commercial/Industrial CPU
- Correction of the VDD\_SOC\_IN voltage.
- LWIP projects added reading of the MAC address from the onboard I2C EEPROM
- Changed flash algorithm, the size of the flash and the header file for the flash
- Wi-Fi and Bluetooth projects
  - o added an I2C driver for the gpio expander and code to use it
  - modified pin muxing
  - o patched the WICED driver
- Adjusted the USB interface number (it is different for host and device examples)
- Changed reset pin for SD card examples
- Changed the mflash driver to work with the OctalSPI/QuadSPI flash

### 2.3 Getting Started with a Specific IDE

The following chapters will describe how to get started with several different IDEs.

Chapter 3 describes Keil uVision and the SDK.

Chapter 4 describes Keil uVision and RTE.

Chapter 5 describes IAR Embedded Workbench.

Chapter 6 describes NXP MCUXpresso.

### 3 Getting Started with Keil uVision/MDK and SDK

This section is a guide to open, build and debug an example application using Keil uVision/MDK with the SDK that was downloaded in 2.1. It is assumed that you have this development environment installed on your computer.

Following the instructions in this section will make it possible to run all examples in the downloaded SDK. Note that none of the examples use the RTE (Run-Time Environment) feature of the Keil uVision/MDK. The use of RTE is described in chapter 4.

Make sure the version of uVision is v5.25.2, or later. Otherwise, there is no support for flash download/programming to the OctalSPI flash memory directly.

### 3.1 Install the SDK

Unpack the SDK zip file that was downloaded in 2.1 somewhere on the local file system. It is suggested to use a very short path as the SDK has a deep folder structure and a path that is too long can cause problems in Windows.

The folder that the SDK is unpacked into will be referred to as **<install\_dir>** in the following sections.

### 3.2 Install CMSIS Device Pack

After the MDK tools are installed, Cortex Microcontroller Software Interface Standard (CMSIS) device packs must be installed to fully support the device from a debug perspective. These packs include things such as memory map information, register definitions and flash programming algorithms. Follow these steps to install the IMXRT105x CMSIS pack or IMXRT106x CMSIS pack.

- 1. Start Keil uVision
- 2. Start the Pack Installer tool with this button on the toolbar:



 Browse to the MIMXRT1052 in the device tree (NXP→MIMXRT1052→MIMXRT1052xxxxB) or MIMXRT1062 in the device tree (NXP→MIMXRT1062→MIMXRT1062xxxxA) and then click the Install button next to the package with a "\_DFP" in the name. Note that the button in the image below has "Up to date" as the package has already been installed.

Pack Installer - C:\Keil_v5\ARM\PACK				
<u>File Packs Window H</u> elp				
Device: NXP - MIMXRT1052DVL6B				
1 Devices Boards	4	Packs Examples		
Search: mimxrt*2 - X	Ē	Pack	Action	Description
Device /	Summary	🖃 Device Specific	3 Packs	MIMXRT1052DVL6B selected
🖃 🍕 All Devices	15 Devices			Embedded Artists iMXRT1052
NXP	15 Devices	T-NYP-EV/CP IMAXPT1050	. 🚸 Up to date	Board Support Pack for EVKBI
i.MX RT Series	1 Device	■ NXP::MIMXRT1052_DFP	🥏 Up to date	Device Family Pack for MIMXR
MIMXRT1021	4 Devices	. ⊕ Generic	35 Packs	
MIMXRT1052	6 Devices			
🗄 🔧 MIMXRT1052xxxA	2 Devices			
MIMXRT1052xxxB	4 Devices			
MIMXRT1052CVJ5B	ARM Cortex-M7, 600 MHz			
MIMXRT1052CVL5B	ARM Cortex-M7, 600 MHz			
MIMXRT1052DV/6B	ARM Cortex-M7, 600 MHz			
MIMXRT1052DVL6B	ARM Cortex-M7, 600 MHz			
MIMXRT1062	4 Devices			
MIMXRT1062xxxA	4 Devices			
MIMXRT1062CVJ5A	ARM Cortex-M7, 600 MHz			
MIMXRT1062CVL5A	ARM Cortex-M7, 600 MHz			
MIMXRT1062DV/6A	ARM Cortex-M7, 600 MHz			
MIMXRT1062DVL6A	ARM Cortex-M7, 600 MHz			

### 3.3 Build an Example Application

The following steps will guide you through opening the hello\_world application. These steps may change slightly for other example applications as some of these applications may have additional layers of folders in their path.

1. If not already done, open the desired example application workspace in:

```
<install_dir>/boards/evkbimxrt1052/<example_type>/
<application_name>/mdk
The workspace file is named <demo_name>.uvmpw, so for this specific example, the actual
path is:
```

```
<install_dir>/boards/evkbimxrt1052/demo_apps/
hello world/mdk/hello world.uvmpw
```

or

```
<install_dir>/boards/evkmimxrt1062/<example_type>/
<application_name>/mdk
The workspace file is named <demo_name>.uvmpw, so for this specific example, the actual
path is:
<install_dir>/boards/evkmimxrt1062/demo_apps/
hello_world/mdk/hello_world.uvmpw
```

2. To build the demo project, select the "Rebuild" button, highlighted in red.

👔 🥸 🎬 🏙 😻 🕶 🤜 🔤 🙀 hello_world flexspi_no	-
--	---

- We X <u>File Edit View Project Flash Debug Peripherals Tools SVCS Window Help</u> 🍪 🕮 🕮 📽 📲 🙀 🕴 hello\_world debug 🛛 🔽 🐔 着 🛸 🗇 🆄 # 🛛 🚺 hello\_world.c Project **•** × 🖃 쓚 Project: hello\_world 1 1/ ~ \* Copyright (c) 2013 - 2015, Freescale Semiconductor, Inc. \* Copyright 2016-2017 NXP \* All rights reserved. 🗄 🔛 hello\_world debug 2 3 🗄 🦾 source 4 🗄 📄 hello\_world.c 5 🗄 🚞 board \* SPDX-License-Identifier: BSD-3-Clause 6 🗄 🛄 doc 7 \*/ 🗄 🚞 drivers 8 9 #include "fsl\_device\_registers.h"
  10 #include "fsl\_debug\_console.h"
  11 #include "board.h" 🗄 🛄 device 🕀 🚞 utilities 🗉 🛄 component-uart 12 🗄 🛄 component-serial\_manager 🗄 🛄 component-lists 🗄 🚞 startup 16 \* Definitions 🕀 🛄 xip 17 \*\*\*\*\*\*\*\*\*\*\* 18 🗄 🛅 CMSIS 19 21 \* Prototypes • 🔃 Project 🌍 Books | {} Func... | 0, Temp...| < Build Output **4** compiling generic list.c... ~ compiling evkbimxrt1050\_flexspi\_nor\_config.c... linking... Program Size: Code=11780 RO-data=2016 RW-data=8 ZI-data=264316 "debug/hello\_world.out" - 0 Error(s), 0 Warning(s). Build Time Elapsed: 00:00:01 CMSIS-DAP Debugg
- The build will complete without errors.

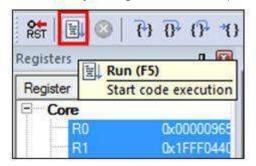
### 3.4 Run an Example Application

To download and run the application, perform these steps:

- 1. Connect the debug probe to the development platform to your PC via USB cable. See chapter 7 for details how to connect the debug probe.
- 2. Select the *Debug* menu and then *Start/Stop Debug Session*, or simply press Ctrl+F5. The application will then be downloaded into SRAM.

File Edit View Project Flash	Debug Peripherals Tools SV	CS Window	Help
🗋 🗃 🖬 🗿 🕉 🖧 🔤	Start/Stop Debug Session	Ctrl+F5	🗐 // 🗟 🖄 LPC_IOCON_TypeDef 🛛 🛃
🕸 🕮 🝘 🥔 🔜 🙀 🔤 hello	Reset CPU		
Project	E. Run	F5	- · ·
🖃 🔧 Project: hello_world	🙆 Stop		
🖻 羄 hello_world Xip	{+} Step	F11	013 - 2015, Freescale Semicond
🖨 🦾 source	Step Over	F10	2017 NXP
🕀 📄 hello_world.c	() Step Over () Step Out	Ctrl+F11	and use in source and binary f
🕀 🛄 board	<ul> <li>Run to Cursor Line</li> </ul>	Ctrl+F10	rovided that the following con
🗄 🛄 doc		Ctrl+F10	The second second second
🕀 🦾 drivers	Show Next Statement		ns of source code must retain and the following disclaimer.
🕀 🛄 startup	Breakpoints	Ctrl+B	and one torrowing drootsiner.
🕀 🧰 utilities	Insert/Remove Breakpoint	F9	ns in binary form must reprodu
🗄 🛄 FlashHeader	O Enable/Disable Breakpoint	Ctrl+F9	tions and the following discla ls provided with the distribut
	🔗 Disable All Breakpoints		is provided with the distribut
🛅 Project 🌍 Books   {} Func   🗛 T	Kill All Breakpoints	Ctrl+Skift+F9	•
Suild Output			ф.
compiling fsl_str.c	OS Support	1	2
compiling fsl_assert.c	Execution Profiling		
compiling fsl_debug_consc linking	Memory Map		
Program Size: Code=7828 F	Inline Assembly		84
"xip_txt_debug\hello_worl	Function Editor (Open Ini File	)	•
Build Time Elapsed: 00:0			_
٠			. Б.

- Open the terminal application on the PC, such as TeraTerm or PuTTY, and connect to the debug serial port number. Configure the terminal with 115200 baud, 8N1.
   You can alter the baud rate be searching for the reference BOARD\_DEBUG\_UART\_BAUDRATE variable in file: board.h
- 4. Run the code by clicking the "Run" button (or tress F5) to start the application.



5. The hello\_world application is now running and a banner is displayed on the terminal. If this is not true, check your terminal settings and connections.

File	Edit	<u>S</u> etup	Control	Window	<u>H</u> elp	
ello	uorld.					
						9

### 3.5 Selecting Build Target

A target is a variant of the project that for example can change compiler flags or dictate in what memory region the program should execute. All projects have one or more targets as shown in the picture below.

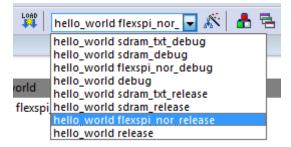


Figure 1 – Build Targets in Keil uVision/MDK

The target name starts with the project name, **hello\_world** in this case, and then the nomenclature as outlined in the table below is used. Release targets have a higher compiler optimization degree making the target less suitable for debugging.

<project> sdram_debug <project> sdram_release</project></project>	The application runs in internal SRAM but with data in external SDRAM
<project> sdram_txt_debug <project> sdram_txt_release</project></project>	The application runs in external SDRAM but with data in internal SRAM
<project> debug <project> release</project></project>	The application runs in internal SRAM
<project> flexspi_nor_debug <project> flexspi_nor_release</project></project>	The application runs in external flash, which must be programmed/flashed before use.

Note that each of the targets has its own set of options so changing for example include path in one target does not affect the other targets.

### 4 Getting Started with Keil uVision/MDK and RTE

This section is a guide to open, build and debug an example application using Keil uVision/MDK with the Board Support Package (BSP) installed using the Pack Installer and **not the SDK described in section 2.1**. It is assumed that you have this development environment installed on your computer.

The BSP contains a small subset of all the examples available in the SDK. The main difference between the SDK and the BSP examples is that the BSP examples use the Run-Time Environment (RTE) feature in Keil uVision/MDK and the SDK doesn't.

The RTE is a way to manage software components (e.g. drivers, RTOS, network stacks, USB stacks) to add/remove features in a project without having to add each source file and include path individually.

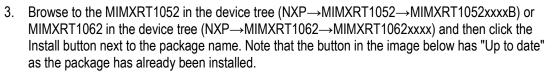
The recommendation is not to use RTE, but instead to use the SDK (as described in chapter 3) examples as a base for your programming. There are only a very limited set of examples are available for RTE.

Make sure the version of uVision is v5.25.2, or later. Else there is no support for flash download/programming to the OctalSPI flash memory directly.

### 4.1 Install CMSIS Device Pack

After the MDK tools are installed, Cortex Microcontroller Software Interface Standard (CMSIS) device packs must be installed to fully support the device from a debug perspective. These packs include things such as memory map information, register definitions and flash programming algorithms. Follow these steps to install the IMXRT105x CMSIS pack or IMXRT106x CMSIS pack.

- 1. Start Keil uVision
- 2. Start the Pack Installer tool with this button on the toolbar:



Pack Installer - C:\Keil_v5\ARM\PACK				
<u>File Packs Window H</u> elp				
Device: NXP - MIMXRT1052DVL6B				
Devices Boards	4	Packs Examples		
Search: mimxrt*2 - X		Pack	Action	Description
Device 🔨	Summary	Device Specific	3 Packs	MIMXRT1052DVL6B selected
🖃 🏂 All Devices	15 Devices	EmbeddedArtists::iMX		
NXP	15 Devices	■ NXP::EVKB-IMXRT1050.		Board Support Pack for EVKBI
🗉 🔧 i.MX RT Series	1 Device	■ NXP::MIMXRT1052_DFP	😣 Up to date	Device Family Pack for MIMXR
🖃 🔧 MIMXRT1021	4 Devices	Generic	35 Packs	
MIMXRT1052	6 Devices			
MIMXRT1052xxxxA	2 Devices			
MIMXRT1052xxxxB	4 Devices			
MIMXRT1052CVJ5B	ARM Cortex-M7, 600 MHz			
MIMXRT1052CVL5B	ARM Cortex-M7, 600 MHz			
MIMXRT1052DVI6B	ARM Cortex-M7, 600 MHz			
MIMXRT1052DVL6B	RM Cortex-M7, 600 MHz			
	4 Devices			
MIMXRT1062xxxA	4 Devices			
MIMXRT1062CVJ5A	ARM Cortex-M7, 600 MHz			
MIMXRT1062CVL5A	ARM Cortex-M7, 600 MHz			
MIMXRT1062DVI6A	ARM Cortex-M7, 600 MHz			
MIMXRT1062DVL6A	ARM Cortex-M7, 600 MHz			

4. Next click the Install button for the EmbeddedArtists::iMXRT1052\_DevelopersKit\_BSP or EmbeddedArtists::iMXRT1062\_DevelopersKit\_BSP. Note that the button in the image below has "Up to date" as the package has already been installed

ile <u>P</u> acks <u>W</u> indow <u>H</u> elp				
Device: NXP - MIMXRT1052DVL6B     Devices Boards	Þ	Packs Examples		Þ
Search: mimxrt*2 • X		Pack	Action	Description
Device /	Summary	Device Specific	3 Packs	MIMXRT1052DVL6B se
= 🍄 All Devices	15 Devices	EmbeddedArtists::iMXRT1052_DevelopersKit_BSP	💠 Up to date	Embedded Artists iMX
NXP	15 Devices	NXP::EVKB-IMXRT1050_BSP	😻 Up to date	Board Support Pack fo
+ 🥸 i.MX RT Series	1 Device	NXP::MIMXRT1052_DFP	🚸 Up to date	Device Family Pack fo
HIMXRT1021	4 Devices		35 Packs	
E MIMXRT1052	6 Devices			
🗄 🔧 MIMXRT1052xxxA	2 Devices			
MIMXRT1052xxxxB	4 Devices			
MIMXRT1052CVJ5B	ARM Cortex-M7, 600 MHz			
MIMXRT1052CVL5B	ARM Cortex-M7, 600 MHz			
MIMIKRT1052DVI6P	ARM Cortex-M7, 600 MHz			
MIMXRT1052DVL6B	ABM Cortex-M7, 600 MHz			
H 🧐 MIMXR11062	4 Devices			

### 4.2 Build an Example Application

The following steps will guide you through opening the Hello World application. These steps may change slightly for other example applications as some of these applications may have additional layers of folders in their path.

- 1. Start Keil uVision
- 2. Start the Pack Installer tool with this button on the toolbar:



 Browse to the MIMXRT1052 in the device tree (NXP→MIMXRT1052→MIMXRT1052xxxxB) or MIMXRT1062 in the device tree (NXP→MIMXRT1062→MIMXRT1062xxxxB) and then select the Examples tab Click the Copy button next to the example to test it. In this case the Hello World example.

ile <u>P</u> acks <u>W</u> indow <u>H</u> elp Device: NXP - MIMXRT1052				
Devices Boards	Þ	4 Packs Examples		¢
Search: mimx • X	Ē	Show examples from installed Packs only		
Device /	Summary	Example	Action	Description
🖃 🍕 All Devices	30 Devices	CMSIS-RTOS Blinky (EA iMXRT1052 Develope	🚸 Сору	CMSIS-RTOS2 Blinky example
NXP	30 Devices	-GPIO LED output (EA iMXRT1052 Developers	🚸 Сору	GPIO LED output example
😐 🔧 i.MX RT Series	2 Devices	-GPIO input interrupt (EA iMXRT1052 Develop	Corry	GPIO input interrupt example
🗄 🍕 MIMX8MD6	2 Devices	Hello World (EA iMXRT1052 Developers Kit)	💠 Сору	Hello World demo application provides a sanity c
😐 🔧 MIMX8MD7	2 Devices	-LPUART eDMA transfer (EA iMXRT1052 Devel	😵 copy	LPUART eDMA transfer example
🗄 🤧 MIMX8MQ5	2 Devices	-LPUART interrupt transfer (EA iMXRT1052 De	🚸 Сору	LPUART interrupt transfer example
🕀 🏤 MIMX8MQ6	2 Devices	-QTMR timer (EA iMXRT1052 Developers Kit)	🚸 Сору	QTMR timer example
🗄 🦓 MIMX8MQ7	2 Devices	-WDOG (EA iMXRT1052 Developers Kit)	🚸 Сору	WDOG example
🕀 🏤 MIMXRT1021	4 Devices	eLCDIF LUT (EA iMXRT1052 Developers Kit)	🚸 Сору	eLCDIF LUT example
🗄 🤧 MIMXRT1051	3 Devices	eLCDIF RGB (EA iMXRT1052 Developers Kit)	🚸 Сору	eLCDIF RGB example
🖃 🍕 MIMXRT1052	4 Devices			
B 🕸 MIMXRT1052xxxxB	4 Devices			
MIMXRT1052CVJ5B	ARM Cortex-M7, 600 MHz			
MIMXRT1052CVL5B	ARM Cortex-M7, 600 MHz			
MIMXRT1052DVJ6B	ARM Cortex-M7, 600 MHz			
MIMXRT1052DVL6B	ARM Cortex-M7, 600 MHz			
	3 Devices			
🖃 🤧 MIMXRT1062	4 Devices			

5. Select a location for the project files in the dialog that appears and then click OK.

c:\temp\my_pack\		▼ Browse
✓ Use Pack Folder Structure	✓ Launch µVision	

- 6. A new uVision window will appear after a few seconds with the newly created project
- 7. To build the demo project, select the "Rebuild" button, highlighted in red.



- 👄 🗖 🗖 📈 🗠 C:\temp\my\_pack\examples\hello\_world\hello\_world.uvprojx - µVision <u>File Edit View Project Flash Debug Peripherals Tools SVCS Window Help</u> 🕞 🗟 🥐 🔍 • 💊 💿 🔗 🔞 09\_0d\_bs 🖄 🕌 🐇 🖓 🖉 🗢 🗠 🛛 🗢 🖌 🖓 🕵 🐘 🎼 🕼 🖓 🧼 🕮 🕮 🔹 - 🔜 🙀 flexspi\_nor\_debug 💿 🐷 👗 📥 🍫 🐡 🎰 # 🖬 📄 hello\_world.c Project - × 🖃 🍕 Project: hello\_world 1 -1/\* \* The Clear BSD License
   Copyright (c) 2013 - 2015, Freescale Semiconductor, Inc.
   Copyright 2016-2017 NXP 😑 🚂 flexspi\_nor\_debug 2 3 🗄 🧀 source 4 🗄 📄 hello\_world.c \* All rights reserved. 5 🕀 🛄 doc 6 7 🗄 🚸 Board Support \* Redistribution and use in source and binary forms, with or with 🚸 CMSIS 8 \* are permitted (subject to the limitations in the disclaimer belo that the following conditions are met: 🗄 🚸 Device 10 11 \* o Redistributions of source code must retain the above copyright 12 of conditions and the following disclaimer. 13 14 15 \* o Redistributions in binary form must reproduce the above copyr:
   \* list of conditions and the following disclaimer in the document 16 other materials provided with the distribution. 17 \* o Neither the name of the copyright holder nor the names of its \* contributors may be used to endorse or promote products derive 18 19 20 software without specific prior written permission. 21 \* NO EXPRESS OR IMPLIED LICENSES TO ANY PARTY'S PATENT RIGHTS ARE 22 23 \* THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBU: \* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED T( + 24 🖻 Project 🌏 Books | {} Func... | 🗓 Temp...| . ..... Build Output д 🔛 compiling fsl\_log.c... linking... Program Size: Code=8380 RO-data=3224 RW-data=24 ZI-data=2084 After Build - User command #1: fromelf --bincombined --output "C:\temp\my pack\examples\hello world\flexspi ".\flexspi\_nor\_debug\hello\_world.out" - 0 Error(s), 0 Warning(s). Build Time Elapsed: 00:00:25 CMSIS-DAP Debugg
- The build will complete without errors.

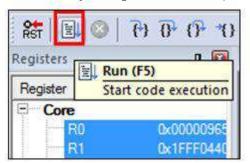
### 4.3 Run an Example Application

To download and run the application, perform these steps:

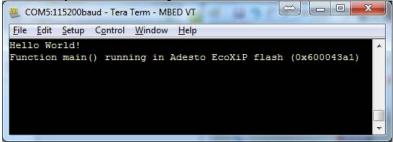
- 1. Connect the debug probe to the development platform to your PC via USB cable. See chapter 7 for details how to connect the debug probe.
- 2. Select the *Debug* menu and then *Start/Stop Debug Session*, or simply press Ctrl+F5. The application will then be downloaded into flash and started.

<u>File Edit View Project Fla</u> sh	Debug Peripherals Tools SVCS Window	Help
n 🗃 🖬 🗿 🐒 🕰 🖪	Start/Stop Debug Session Ctrl+F5	📔 //🗧 🖄 ad_b0_09 🛛 🕞 🔜 🥐 🔍 🗸 🐠 🔿 🔗
🕸 🕮 🕮 🥔 - 📇 🙀 📊	Energy Measurement without Debug	
Project	Reset CPU	F ·
<ul> <li>Project: hello_world</li> <li>Flexspi_nor_debug</li> <li>source</li> <li>hello_world.c</li> <li>doc</li> <li>8 Board Support</li> <li>CMSIS</li> <li>9 Device</li> </ul>	Bun       F5         Stop       F1         Step       F11         Step Ower       F10         Step Out       Ctrl+F11         Step Out       Ctrl+F10         Show Negt Statement       Breakpoints         Breakpoints       Ctrl+B         Insert/Remove Breakpoint       F9         Enable/Disable Breakpoints       Ctrl+F9         Disable All Breakpoints       Ctrl+Skift+F9	icense 013 - 2015, Freescale Semiconductor, Inc. 2017 NXP rved. and use in source and binary forms, with or withor subject to the limitations in the disclaimer belowing wing conditions are met: ns of source code must retain the above copyright and the following disclaimer.
E Project 《 Books ] {} Func [0.]	Memory Map Inline Assembly Function Editor (Open Ini File)	ame of the copyright holder nor the names of its may be used to endorse or promote products derive out specific prior written permission. MPLIED LICENSES TO ANY PARTY'S PATENT RIGHTS ARE S PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBU: IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO
Build Output	t.	<b>p</b>
After Build - User comma	lo_world.out" - 0 Error(s), 0 War	tput "C:\temp\my_pack\examples\hello_world\flexspi
•		

- Open the terminal application on the PC, such as TeraTerm or PuTTY, and connect to the debug serial port number. Configure the terminal with 115200 baud, 8N1.
   You can alter the baud rate be searching for the reference BOARD\_DEBUG\_UART\_BAUDRATE variable in file: board.h
- 4. Run the code by clicking the "Run" button (or tress F5) to start the application.



5. The Hello World application is now running and a banner is displayed on the terminal. If this is not true, check your terminal settings and connections.



### 4.4 Selecting Build Target

A target is a variant of the project that for example can change compiler flags or dictate in what memory region the program should execute. All projects have one or more targets as shown in the picture below.

1		
LOAD 44	flexspi_nor_debug 🛛 🚽	
	debug	
	flexspi nor debug	•
ld	sdram_debug	÷
bug	sdram_txt_debug	÷
bug	release	
	flexspi_nor_release	7
	sdram_release	ł
world.c	sdram_txt_release	÷

Figure 2 – Build Targets in Keil uVision/MDK

The default target is **flexspi\_nor\_debug**. The table below shows each of the targets. Release targets have a higher compiler optimization degree making the target less suitable for debugging.

sdram_debug sdram_release	The application runs in internal SRAM but with data in external SDRAM
sdram_txt_debug sdram_txt_release	The application runs in external SDRAM but with data in internal SRAM
debug release	The application runs in internal SRAM
flexspi_nor_debug flexspi_nor_release	The application runs in external flash, which must be programmed/flashed before use. Need extra configuration, see section 4.5.

Note that each of the targets has its own set of options so changing for example include path in one target does not affect the other targets.

### 4.5 Configuring Flash Algorithm

If you have an iMX RT1052/62 OEM board up to, and including, rev B2 you must setup projects for storing/executing the application from the OctalSPI flash. Select one of the flexspi\_nor\_\* targets, right click on the project and select *Options for Target <name>* (or press Alt+F7). Select the *Debug tab* and select *Settings* for the debugger. Then select the *Flash Download* tab. Replace the existing driver with the OctalSPI flash driver as shown below.

This is **NOT** applicable for iMX RT1062 boards of **revision C1 or higher** as those use the same flash as the NXP EVK so it is already correct.

Download Function C Erase Full Chip Erase Sectors C Do not Erase	<ul> <li>✓ Program</li> <li>✓ Verify</li> <li>✓ Reset and R</li> </ul>		Agorithm 0x200000000 Size: 0x00008000
Programming Algorithm	T		
Description	Device Size	Device Type	Address Range
MIMXRT105x EcoXiP Flash	4M	Ext. Flash SPI	60000000H - 603FFFFFH
annanonalissono d	4M		
annanonalissono d	4M		60000000H - 603FFFFFH

Figure 3 - Keil uVision/MDK Flash Algorithm Settings

### 4.6 Working with RTE

The Manage Run-Time Environment tool is started with a button on the toolbar:

File Edit View	Project Flash Debug P	eripherals Tools	SVCS Wind	ow Help	
🗋 🚰 🗟 🔊	8 10 10 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	1 1 9 10	同様	🗏 //= //= 🎽 🍱	ad_b0_09
🧇 🖭 🖽 🥔 •	🖽 🕌 👫 flexspi_nor_del	oug 💽 💰	*	ጵ 🛞	
Project	<b>д </b> 🛛			Manage Run-Tir	ne Environment
⊟ <sup>4</sup> S Project: hello	<u>.</u>			Select Software	Components for the

The example projects are already configured correctly, and it is possible to add more software components by ticking the corresponding checkboxes.

Software Component	Sel.	Variant	Versi	on Description
⊕ 🚸 AMP				Asymmetric Multiprocessing
😑 💠 Board Support		Embedded Artist	1.0.1	This is the bundle for the EA iMX RT1052 Developer's Kit.
🖃 🚸 Drivers				
🖉 wdog			1.0.1	Watchdog Driver
Adesto EcoXiP flash (ATXP032)	~		1.0.1	Adesto EcoXiP flash (ATXP032)
🖃 🚸 Project Template	-			
project_template	~		1.0.1	
🗄 🚸 CMSIS				Cortex Microcontroller Software Interface Components
🗄 🚸 CMSIS Driver				Unified Device Drivers compliant to CMSIS-Driver Specifications
🗄 🚸 CMSIS Driver Validation		API	1.0.0	Run API test for enabled drivers
🗄 💠 Compiler		ARM Compiler	1.5.0	Compiler Extensions for ARM Compiler 5 and ARM Compiler 6
🗄 🚸 Device				Startup, System Setup
Startup	~		1.0.0	
🗄 🚸 SDK Drivers				NXP MCUXpresso SDK Peripheral Drivers
🗄 🚸 SDK Project Template				NXP MCUXpresso SDK RTE Device Project Template
🕀 🚸 SDK Utilities				NXP MCUXpresso SDK Utilities
🗄 💠 File System		MDK-Plus	6.10.1	File Access on various storage devices
🗄 🚸 Graphics		MDK-Plus	5.46.5	User Interface on graphical LCD displays
🗄 🚸 Network		MDK-Plus	7.9.0	IPv4 Networking using Ethernet or Serial protocols
🗄 💠 RTOS		FreeRTOS	10.0.1	FreeRTOS Real Time Kernel
🗄 🚸 USB		MDK-Plus	6.12.8	USB Communication with various device classes

There are a couple of very important things to note:

- 1. The Board Support must be "Embedded Artists" to get the drivers and templates as seen above.
- If you have an iMX RT1052/62 OEM board with revision up to, and including, rev B2 then the Adesto EcoXip flash (ATXP032) must be selected otherwise applications will not start after being flashed.
- If the watchdog driver (wdog) is used it must be selected from Board Support->Drivers->wdog and not the Device->SDK Drivers->wdog. The reason is that the one in Board Support is patched to work with the hardware.

The system for managing the RTE is described by Keil here: http://www.keil.com/support/man/docs/uv4/uv4\_ca\_rtemanager.htm .

### **5 Getting Started with IAR Embedded Workbench**

This section is a guide to open, build and debug an example application using IAR Embedded Workbench. It is assumed that you have this development environment installed on your computer.

Make sure the version of Embedded Workbench is 8.30.1, or later. Else there is no support for flash download/programming to the OctalSPI/QuadSPI flash memory directly.

### 5.1 Install the SDK

Unpack the SDK zip file that was downloaded (as described in section 2.1) somewhere on the local file system. It is suggested to use a very short path as the SDK has a deep folder structure and a path that is too long can cause problems in Windows.

The folder that the SDK is unpacked into will be referred to as <install\_dir> in the following sections.

#### 5.2 Build an Example Application

The following steps will guide you through opening the hello\_world application. These steps may change slightly for other example applications as some of these applications may have additional layers of folders in their path.

The workspace file is named <demo\_name>.eww, so for this specific example, the actual path is:

```
<install_dir>/boards/evkmimxrt1060/demo_apps/
hello world/iar/hello world.eww
```

 Select the desired build target from the drop-down. For this example, select the "flexspi\_nor\_debug" target.

File Edit View Project	CMSIS-DAP Tools Win	dow Help				
🖞 🖸 🖬 🗿 🔒 🗆 🗶	D 0 0 0		- <	Q>	5 × E	<
Workspace	<b>→</b> ‡ ×					
debug	•					
debug						
flexspi_nor_debug sdram_debug						
sdram_txt_debug release						
release flexspi_nor_release						
sdram_release						
sdram_txt_release						
⊞ III startup						
🕀 💻 utilities						
—⊞ <b>≣</b> xip						
└─⊞ 📹 Output						

The target name indicates from where the application is executing, see table below. The name also indicates if it is a release or debug target. Release targets have a higher compiler optimization degree making the target less suitable for debugging.

sdram_debug sdram_release	The application runs in internal SRAM but with data in external SDRAM
sdram_txt_debug sdram_txt_release	The application runs in external SDRAM but with data in internal SRAM
debug release	The application runs in internal SRAM
flexspi_nor_debug flexspi_nor_release	The application runs in external flash, which must be programmed/flashed before use.

- 3. Note that the target has its own set of options so changing, for example, the include path in one target does not affect the other targets.
- 4. To build the demo application, click the "Make" button, highlighted in red below.

<u>F</u> ile <u>E</u> dit	<u>V</u> iew	<u>P</u> roject	<u>C</u> MSIS-DAP	<u>T</u> ools	<u>W</u> indow	<u>H</u> el	р									
1 🗅 🗎		<u>لا</u> ا		D C			<b>*</b>	< Q	> 4	► E	< 😳	>	< >	0	•==	0 🕨
Workspace					🔺 t	×										
[flexspi_nor_de	ebug					-										
Files					\$	•										
🗆 🌒 hello		l - flexs	pi_nor_d	ebug	<ul> <li>✓</li> </ul>											
-⊞ 🖬 bo																
do 🗐 🕀 🕂																
📙 🕂 🕀 🖬 driv																
🖬 🛋 sou	urce															
🛛 🗕 🖽 🖬 sta	rtup															
📙 🕂 🕀 🖬 utili	ties															
⊞ 🛋 xip																
L-⊞ <b>≡</b> Our	tput															

5. The build completes without errors.

### 5.3 Run an Example Application

To download and run the application, perform these steps:

- Connect the debug probe to the development platform to your PC via USB cable. See chapter 7 for details how to connect the debug probe.
- Open the terminal application on the PC, such as TeraTerm or PuTTY, and connect to the debug serial port number. Configure the terminal with 115200 baud, 8N1.
   You can alter the baud rate be searching for the reference BOARD\_DEBUG\_UART\_BAUDRATE variable in file: board.h
- 3. Click the "Download and Debug" button to download the application to the target.



- 4. The application is then downloaded to the target and automatically runs to the main() function.
- 5. Run the code by clicking the "Go" button to start the application.



6. The hello\_world application is now running and a banner is displayed on the terminal. If this is not true, check your terminal settings and connections.



### 6 Getting Started with NXP MCUXpresso IDE

This section is a guide to open, build and debug an example application using NXP MCUXpresso IDE. It is assumed that you have this development environment installed on your computer.

### 6.1 Install the SDK

MCUXpresso requires the SDK to be installed before it can be used. To do that start MCUXpresso and then drag-n-drop the SDK zip file that was downloaded (as described in section 2.1) to the "Installed SDKs" tab in MCUXpresso:

🎁 Installed SDKs 🐹 🔲 Properties  🖹	Problems 📃 Console 🔬	🖵 Terminal 🛛 📄 Im	nage Info 🛛 🙀 Debugger Console 🛛 🛞
🕅 Installed SDKs			
To install an SDK, simply drag and drop an	SDK (zip file/folder) into th	ne 'Installed SDKs' v	iew. [Common 'mcuxpresso' folder]
Installed SDKs Available Boards Availa	ble Devices		
Installed SDKs Available Boards Availa Name	ble Devices SDK Version	Manifest Version	Location
			Location Common>\eaimxrt1052_sdk_2.8

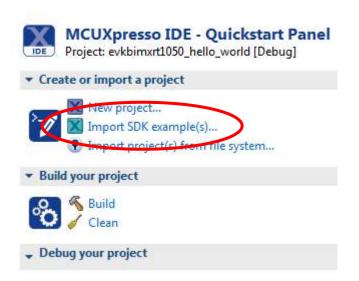
During the installation MCUXpresso will make a copy of the zip file that you drag-n-dropped so it is ok to delete it afterwards if you don't need it for one of the other IDEs.

### 6.2 Build an Example Application

The following steps will guide you through opening the hello\_world application from the SDK.

- 1. Install the SDK as described in section 6.1 if you have not done so already
- Click the "Import SDK example(s)..." link in the Quickstart Panel

U Quickstar... (\*)= Global V... (\*)= Variables 💁 Breakpoi.



3. Select the MIMXRT1050 and eaimxrt1052. Click Next to go to the project selector.

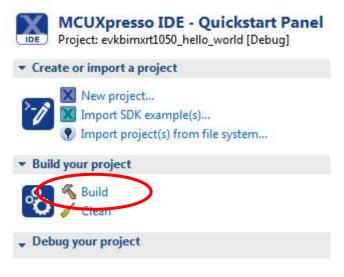
SDK Import Wizard							×
😣 Please select a board					NK		G
Board and/or Device se	election page						
- SDK MCUs	Available boards					↓ªz ↑ª	
MCUs from installed SDKs. Please click	Please select an available board for your project.						
above or visit <u>mcuxpresso.nxp.com</u> to obtain additional SDKs.	Supported backes for device MIMXRT1052xxxxB						^
NXP MIMXRT1052xxxxB > LPC55xx > LPC50 MIMXRT1050 MIMXRT1052xxxxB > MIMXRT1052xxxA > MIMXRT1064 > MIMXRT1064	SDK simxrt1052						
							~
Selected Device:		SDKs for selected MCU					
Target Core: cm7	00MHz, 512KB SRAM Microcontrollers (MCUs) based on ARM	Name	SDK Version	Manifest Ve	Location		
Description: i.MX MIMXRT1052 6 Cortex-M7 Core	UUMHZ, 312KB SKAM MICFOCONTROLIEFS (MICUS) based on Akivi	BDK_2.x_MIMXRT1052xxxxI	2.8.6 (411 2020	3.6.0	Common>\eaim	xrt1052_	sdk_2.8
?			< <u>B</u> ack	Next>	Einish	Can	cel

4. Select the hello\_world example and make sure to switch from Semihost to UART for the SDK Debug Console.

Import projects		
roject name prefix: evkbimxrt1050_	2 Project name suffix	
Use default location		
ocation: C:\temp\2018-05-15_tmp\ws\evkbimxrt1050_		Brows
Project Type	Project Options	
O Project C++ Project C Static Library C++ Static Library     C++ Static Library	SDK Debug Console 💿 Seminost 💿 UART 📝 Copy sources 🕼 Import other files	
xamples		ک 🗹 🔬 🗠 🔛
type to filter		
>     aws_examples       >     cmsis_driver_examples       >     demo_apps       >     mbedtls       >     mbedtls       >     wolfssl       >     bubble       >     compass       >     flexio_pwm       Y     hello_world       >     power_mode_switch_ca       >     sai       >     sdipeg       >     driver_examples       >     ertos_examples		

- 5. Click finish to have MCUXpresso import and set up the selected project.
- 6. Click Build in the Quickstart Panel

🜙 Quickstar... 🕫 Global V... 🔅 Variables 🏻 🎭 Breakpoi.



7. The program builds without errors

### 6.3 Run an Example Application

To download and run the application, perform these steps:

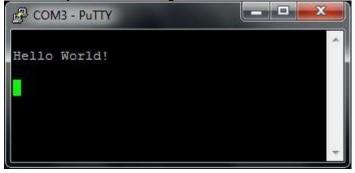
- Connect the debug probe to the development platform to your PC via USB cable. See chapter 7 for details how to connect the debug probe.
- Open the terminal application on the PC, such as TeraTerm or PuTTY, and connect to the debug serial port number. Configure the terminal with 115200 baud, 8N1.
   You can alter the baud rate be searching for the reference BOARD\_DEBUG\_UART\_BAUDRATE variable in file: board.h
- 3. Click the "Debug" button in the Quickstart Panel to download the application to the target.

Creat	e or impor	t a project			
_		raproject			
		oject SDK example(s). project(s) from			
	your proje				
°°	Suild Clean				
, Debu	g your proj	ect	• 🔀	🔛 🕶 🔝	•

- 4. The application is then downloaded to the target and automatically runs to the main() function.
- Run the code by clicking the "Resume" button to start the application.
   Help



6. The hello\_world application is now running and a banner is displayed on the terminal. If this is not true, check your terminal settings and connections.



### 6.4 Target Memory

Almost all the examples for MCUXpresso are setup to run from flash. If the flash is used, or not, can be seen when compiling:

👩 Instal 🔲 Prop	🖹 Probl	Cons 🛛	🍠 Termi	🗟 Imag
		•	ኑ 🔂 🔁	
CDT Build Console [eaim	xrt1062_hello_w	orld]		
Building target: e Invoking: MCU Link arm-none-eabi-gcc	er –	-		062 hello
Memory region				
BOARD_FLASH:	27752 E	3 4	MB Ø	.66%
BOARD_SDRAM:	8432 8	3 30	MB Ø	.03%
NCACHE_REGION:	28 E	3 2	MB Ø	.00%
SRAM_DTC:	0 GE	3 128	КВ 0	.00%
SRAM_ITC:	0 GE	3 128	КВ 0	.00%
SRAM_OC:	0 GE	3 768	КВ 0	.00%
Finished building	target: eaim	rt1062_hell	lo_world.a	xf

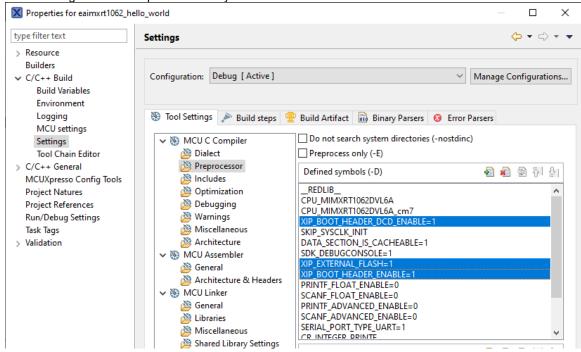
To run the program in RAM/SDRAM (if it is small enough to fit):

1. Open the Project -> Properties menu and navigate to MCU Settings

/pe filter text	MCU settings						🤤 🔻 o	-> -
Resource Builders C/C++ Build Build Variables	Avai	lable parts						
Environment	SDK MCUs			Preinstall	ed MCUs			
MCU settings Stttings	MCUs from in:	stalled SDKs. Please c presso.nxp.com 's.			n preinstalled	LPC and ger	neric Cortex-	М
Tool Chain Editor	E			Target				^
C/C++ General	NXP MIMXF	T1062xxxxA		> LPC1	102			
MCUXpresso Config Tools	> LPC55xx			> LPC1	12x			
Project Natures	> LPC802			> LPC1	1Axx			
Project References	> MIMXRT			> LPC1	1E6x			
Run/Debug Settings	✓ MIMXRT1	and a second		> LPC1	1Exx			
Task Tags		RT1062xxxxA		> LPC1	1U6x			
Validation	> MIMXRT			> LPC1	1Uxx			
	> MIMXRT	170		> LPC1	1xx			0.2
				> LPC1	1xxLV			~
	Target architect	ure:		cortex-m7				
	Preserve me	ure: emory configuration oject configuration		cortex-m7				
	Preserve me	emory configuration		cortex-m7				
	Preserve me	emory configuration		cortex-m7			Browse	
	Preserve me	emory configuration oject configuration ils (MIMXRT1062xx		cortex-m7	Size	Driver	- hipotecondone	
٢	Preserve me Preserve pro Memory deta Default Link	emory configuration oject configuration ils (MIMXRT1062xx Server Flash Driver	xxA)*		Size 0x400000	Driver	10	
c	Preserve me Preserve pro Memory deta Default Link: Type	emory configuration oject configuration ils (MIMXRT1062xx Server Flash Driver Name	xxA)* Alias	Location				
c	Preserve me Preserve pro Memory deta Default Link: Type Flash	emory configuration oject configuration ils (MIMXRT1062xx Server Flash Driver Name BOARD_FLASH	Alias Flash	Location 0x6000000	0x400000		10	
c	Preserve me Preserve pro Memory deta Default Link: Type Flash RAM	emory configuration oject configuration ils (MIMXRT1062xx Server Flash Driver Name BOARD_FLASH BOARD_FLASH	Alias Flash KAM	Location 0x6000000 0x8000000	0x400000 0x1e00000		10	
c	Preserve me Preserve pro Memory deta Default Link: Type Flash RAM RAM	emory configuration oject configuration ils (MIMXRT1062xx Server Flash Driver Name BOARD_FLASH BOARD_FLASH BOARD_SDRAM NCACHE_REGION	Alias Flash KAM RAM2	Location 0x6000000 0x8000000 0x81e00000	0x400000 0x1e00000 0x200000		10	
c	Preserve me Preserve pro Memory deta Default Link: Type Flash RAM RAM RAM	emory configuration oject configuration ils (MIMXRT1062xx Server Flash Driver Name BOARD_FLASH BOARD_FLASH NCACHE_REGION SRAM_DTC	Alias Flash KAM RAM2 RAM3	Location 0x6000000 0x8000000 0x81e00000 0x2000000	0x400000 0x1e00000 0x200000 0x20000		10	

2. Select the BOARD\_FLASH row in the table and then click the Delete button to remove the flash.

3. Go to Settings and the Preprocessor entry



- There are three symbols that must be changed. The symbols are XIP\_BOOT\_HEADER\_DCD\_ENABLE, XIP\_EXTERNAL\_FLASH and XIP\_BOOT\_HEADER\_ENABLE. Double-click each one and change the value from 1 to 0 to disable the feature.
- 5. Build the project and look at the output. The BOARD\_FLASH is no longer present and instead the SDRAM is used:

🎁 Instal 🔲 Prop	🖹 Probl 📃	Cons 🛛 🐰	🛙 Termi 📷 Imag
		÷	수 🔄 🖬 🔐 😑
CDT Build Console [eaim	xrt1062_hello_wor	ld]	
Finished building:	/board/pin_	mux.c	
Building target: e	aimynt1062 bel	le world avt	F
Invoking: MCU Link	_	10_wor10.axi	
arm-none-eabi-gcc		nker -Man="e	aimyrt1062 bello
Memory region			
	28260 B		0.09%
NCACHE REGION:			3 0.00%
	0 GB		
	0 GB		
_	0 GB		
Finished building			
maka na maint di	_		

make -- no-nrint-directory nost-build

### 7 Debug Interface

It is strongly recommended to use a debug/JTAG probe during program development. The low-cost MCU-Link or LPC-Link2 are excellent choices. Keil ULINK2 and ULINKplus, as well as Segger JLINK, are also excellent debug probes.

There are two debug interface connectors available on the *iMX OEM Carrier board*:

- J10 this is a Cortex Debug connector. It is a 2x5 pos, 50 mil pitch connector without a shroud. Be careful when inserting the debug probe cable. Position 1 is specifically marked on the PCB silkscreen. It is located in the lower right corner, see Figure 4 below. The connector supports both the SWD and JTAG interfaces.
- J11 this is an ARM Debug connector. It is a 2x10 pos, 100 mil pitch connector with shroud.

Both connectors are defined and support both the SWD and JTAG type of debug interfaces.

Note that to enable the JTAG/SWD interface on the i.MX RT, JP5 shall not be shorted/inserted.

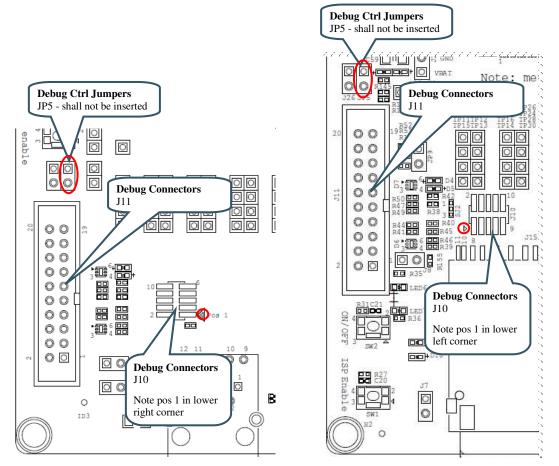


Figure 4 - Debug Interfaces on rev A and rev B1 iMX OEM Carrier board, respectively

Note that due to the powering sequencing requirements on the i.MX RT family, the debug probe I/O voltage **MUST** follow the i.MX RT I/O voltage.

The debug adapter must not drive any output higher than the Vcc/Vref voltage (and if that voltage is zero, then the debug adapter must not drive any output signal). Vcc/Vref is pin 1 on both J10 and J11.

Make sure the debug probe does not have a fixed output voltage, but rather follow Vcc/Vref. If using LPC-Link2 as debug interface, make sure there is **NO** jumper inserted in JP2 on the LPC-Link2.

### 7.1 J-LINK/J-TRACE Support

This section describes the steps necessary to get the Segger J-TRACE to work with NXP MCUXpresso and Keil uVision. The same instructions are likely to work for Segger J-LINK as well, but it has not been verified.

### 7.1.1 Install J-LINK Software

Use version v6.90a or later to get the best support for J-TRACE/J-LINK. The latest version can be found here: https://www.segger.com/downloads/jlink/.

### 7.1.2 MCUXpresso

Build and then launch the debugger. MCUXpresso will detect the J-LINK / J-TRACE and configure itself correctly.

### 7.1.3 Keil uVision

All projects have been configured to use CMSIS-DAP as debug hardware. When switching to J-LINK/J-TRACE it is important that the flash algorithm is changed to one of:

• For iMX RT1062 OEM boards with revision >= C1 MIMXRT106x 8mB QuadSPI NOR Flash and that the flash size is changed to 0x01000000.

#### • All other iMX RT1052/1062 boards

MIMXRT105x EcoXiP Flash and that the flash size is changed to 0x00400000. Note that the same flash driver shall be used for both iMX RT1052 and RT1062.

Change the Debugger from the default CMSIS-DAP to J-LINK / J-TRACE Cortex, as shown in picture below:

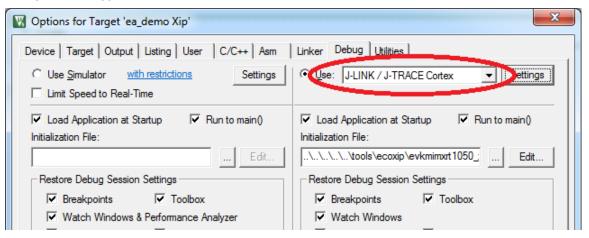


Figure 5 – Setting Debug Interface in Keil uVision

Open the settings dialog and change to the following settings:

		ce	10	
SN: 203200309	-	IDCODE	Device Name	Move
Device: J-Trace Cortex-M Re	v.3 SWD	⊙ 0x0BD11477	ARM CoreSight SW-DP	Up
HW : V3.10 dll : V6.	32a			Down
FW : J-Trace Cortex-M Rev.3		L		DOWN
		matic Detection	ID CODE:	
SW - 5 MHz	n	ual Configuration		
		enner av ten des ut		
Auto	o Clk Add	Delete Upo	date IR len:	
	11:			
Connect & Reset Options		Cash	e Options Download Op	tiona
	set: Core		ache Code	
		J• G	ache <u>M</u> emory	i to <u>r</u> iash
R set after Connect				
9				
			M	lisc
Interface TCP/IP	etwork Settings		Autodataut	
Interface TCP/IP	etwork Settings P-Address	Port (Au	to: 0) Autodetect	lisc JLink Info
Interface TCP/IP	etwork Settings		to: 0) Autodetect	JLink Info
Interface TCP/IP	etwork Settings P-Address		to: 0) Autodetect	

Figure 6 – Configuring J-TRACE/J-LINK Interface in Keil uVision

Cortex JLink/JTrace Target Driver Se	etup			x
Debug Trace Flash Download Download Function C Erase Full Chip C Erase Sectors C Do not Erase			Algorithm Dx20000000 Size: 0x8000	
Programming Algorithm     Description     MIMXPT105: 5	Device Size	Device Type	Address Range	
MIMXRT105x EcoXiP Flash	4M	Ext. Flash SPI	60000000H - 603FFFFFH	
		Start:	Size:	

Figure 7 – Configuring Flash Programming for J-TRACE/J-LINK Interface in Keil uVision

### 7.2 Debug Troubleshooting

In some cases, the IDE complains about not being able to connect to the target. This is most likely because the program already running on the target is interfering. The solution is to put the hardware in ISP mode before starting the flash/debug operation in the IDE. To do this:

- 1. Push and hold down the ISP enable button
- 2. Press the Reset button
- 3. Release the Reset button
- 4. Wait 1 seconds
- 5. Release the ISP enable button

If the LPC-Link2 debugger is used, then there are some additional things to note:

- Make sure that the J2 jumper on the LPC-Link2 is not inserted. If the jumper is inserted/closed then the target will be powered by the LPC-Link2 which might be too much power for the USB port that the LPC-Link2 is connected to.
- 2. If the LPC-Link2 is not found by the IDE and you are working on a laptop, then try using a powered USB hub instead.
- 3. The troubleshooting section in this forum post has a couple of additional things to try: https://community.nxp.com/thread/388964
- 4. There is a Using and troubleshooting LPC-Link2 in the Appendix Additional Hints and Tips of the User Guide for MCUXpresso IDE. The location of the document is c:\nxp\MCUXpressoIDE\_11.0.0\_2516\MCUXpresso\_IDE\_User\_Guide.pdf if the IDE was installed with the default settings (correct path for your specific version of MCUXpresso).

Note that MCUXpresso does not support the "Restart" functionality when debugging. This issue has been reported to NXP. "Restart" works on Keil/ARM uVision/MDK and IAR Embedded Workbench.

### 8 Standalone Program Download

This chapter describes how to download an application to the *iMX RT OEM board* without using the IDE. Note that this section does not describe how to create the application code (create the application, compile and link it). It is assumed that a binary file exists that represent the application program.

As a reminder, there are two basic methods for program download:

### ISP over USB Program Download

ISP is short for In-System Programming. The i.MX RT MCU contains a bootloader in ROM that can be enabled by pressing the ISP Enable push-button.

An application (*MCUXpresso Secure Provisioning Tools*) provided by NXP is needed on the PC for downloading and flashing the application code. It is this method that will be described in this chapter.

- o This method of programming is useful during production
- The MCUXpresso Secure Provisioning Tools application is needed to generate an authenticated or Encrypted image of the application.
- Technically it is possible to program/flash the OctalSPI/QuadSPI flash without a JTAG probe (via NXP's *MCUXpresso Secure Provisioning Tools* application), but it is strongly recommended to use the proper tool for debugging - i.e., use a JTAG probe!

#### SWD/JTAG Debug Interface

Using this method, the application can be downloaded to internal SRAM, to external SDRAM or external OctalSPI/QuadSPI flash.

This method is tightly integrated with the Integrated Development Environment (IDE) used. Specific scripts (and sometimes flash programming algorithms) must exist for the used IDE. Currently such scripts and drivers exist for Keil uVision/MDK, NXP MCUXpresso and IAR Embedded Workbench. For other IDEs, check supported functions.

 There are many different SWD/JTAG interfaces on the market. NXP has created the lowcost MCU-Link and LPC-LINK2, Keil has ULINK2/ULINKpro, Segger has J-LINK, etc.

### 8.1 Install and Patch the Required Software

First, download *MCUXpresso Secure Provisioning Tools* from NXP's website. It can be found under the *Tools & Software* tab for each MCU or directly here. The tool is available for Windows, MacOS and Linux but this document only covers the Windows use case. Patching and using the MacOS and Linux versions should be very similar with only paths and file names varying.

The default installation location is c:\nxp\MCUX\_Provi\_v5\ which will be referred to as <install directory> from now on.

There is one thing that must be patched in the tool itself before it can be used. Download the zip-file from http://imx.embeddedartists.com. It will have a filename like

#### imxrt\_secure\_provisioning\_<date>.zip

Unpack the downloaded file and move EA\_iMXRT10x2\_DevKit\_4MB\_OctalSPI\_ATXP032 to

<install directory>\bin\data\boot\_devices\devices\flex-spi-nor\

The MCUXpresso Secure Provisioning Tool is now ready to be used.

#### 8.2 Prepare the Program to Flash

A program can be setup to run directly in external nor flash, in internal SRAM or in (external) SDRAM and the project must be modified accordingly. This is described in detail in section 7.2.1 of the *User Manual - MCUXpresso Secure Provisioning Tools* that comes with the installation of the tool (filename: MCUXpresso Secure Provisioning Tools.pdf) or it can be viewed here: https://www.nxp.com/docs/en/user-guide/MCUXSPTUG.pdf

### 8.3 Booting an Unsigned Image

Unsigned images are typically used for development. It's recommended to start with this boot type before working with secured images to verify that the executable image works properly.

The first step is to convert the prepared application into a bootable image:

- 1. Select the correct processor
- 2. Make sure that Boot Type is Unsigned
- 3. Set Boot Device:

OEM Board	Boot Device
iMX RT1052 OEM, any revision	flex-spi-nor/EA iMX RT10x2 DevKit 4MB OctalSPI ATXP032
iMX RT1062 OEM, revision <= rev B2	flex-spi-nor/EA iMX RT10x2 DevKit 4MB OctalSPI ATXP032
iMX RT1062 OEM, revision >= rev C1	flex-spi-nor/ IS25LPxxxA_IS25WPxxxA

- 4. Switch to the Build Image tab
- 5. Select the Source executable image that was prepared in the previous section
- If (and only if) the application uses SDRAM, select the Use custom DCD option and point to the dcd\_sdram.bin file from the <install directory>\bin\data\targets\mimxrt\*\ folder
- 7. Click Build Image

MCUXpresso Secure Provisioning - C:	\temp\sec1062_hello765		- 🗆 X
File Target Tools Help MIMXRT1060 via USB Boot Unsign	ned 🗸 f	rom FlexSPI NOR/EA iMX RT10x2 DevKit 4MB OctalSPI ATXP032 💛 Edit	Open, HAB disabled
🗸 Build image 💙 Write image 🥥	PKI management		
Source executable image:	C:\sdk_bld\eaimxrt1062_16mb_sdk_2_12_1	\boards\evkbmimxrt1060\demo_apps\hello_world\mdk\flexspi_nor_release\he	ello_world.hex ~ Browse
Start address:	0x60002000		Application image; XIP: yes
Use custom bootable image path:	bootable_images\hello_world.bin		V Browse
DCD (binary): None 🗸	C:\temp\sec1062_hello765		V Browse
Use the following keys:			
HAB encryption algorithm:			
XIP encryption (BEE user keys) X	IP encryption (BEE OTPMK) OTP conf	iguration	Show build script Build image
Log			
elftosb succeeded			^
<pre>### RESULT of the script `Build Status of the operation: SUCCES:</pre>	ing image`: SUCCESS (return code S : Building image	= [0]SUCCESS)	~
Status of the last operation: SUCCESS : Bui	ilding image		

To write the image:

- 1. Switch to the Write Image tab
- 2. Make sure that the Use built image option is selected
- Connect the hardware to the PC using micro-B to A USB cables in both J12 and J22 connectors
- 4. Put the hardware in ISP mode:
  - a. Push and hold down the ISP enable button
  - b. Press the Reset button
  - c. Release the Reset button
  - d. Wait 1 seconds
  - e. Release the ISP enable button
- 5. Press the Write Image button
- When finished, press the Reset button on the hardware to run the program

MCUXpresso Securi File Target Tools H		ning - C:\temp\sec1062_he	llo765				<u> </u>		\$
MIMXRT1060 via US	SB Boo	Unsigned	✓ from	FlexSPI NOR/IS25LPxxxA_IS25WPxxxA	~	Edit	Open, HAB disabled		
🐓 Build image 🗹 🗸	Write ima	ge 🖉 PKI management							
Bootable image to be	e written								
🗹 Use built image									
Image path:	bootabl	e_images\hello_world_nopa	adding.bin				~	Browse	
Additional input files									
Title	1	Path	[	Description					
Unsigned flashloade	and a second second second second	oootable_images\unsigned		Flashloader (bootable RAM image) used if process					
write parameters (.j:	son) (	:onfigs\write_parameters.js	on i	Parameters needed in write and fuses to be burnt b	y write script	(or shad	ow registers)	Import	
Advanced Start flashloader								Show write scri Write image	
Log									
blhost succeeded	script	'Writing image': SUC	CESS (return code = [0]	SUCCESS)					^
	100	SUCCESS : Writing im	Service and a service service service						~
Status of the last operati	ion: SUCC	ESS : Writing image							

### 8.4 Booting an Authenticated or Encrypted Image

The *MCUXpresso Secure Provisioning Tools* support authenticated (signed) and encrypted (signed+encrypted) images. This is described in detail in the User Manual for the tool. A couple of **very important notes**:

- Encrypted images cannot be used for applications that execute directly in the flash (XiP).
- Burning the fuses in the processor is an irreversible operation. If the fuses are burned and the key is lost, then there is no way to burn anything again on that hardware so make sure to backup the (generated) keys <u>BEFORE</u> burning them to the hardware.
- If the instructions mention setting the SW7 DIP to the board to 0001 it means putting the board into ISP mode. This is done on the Developers Kit by:
  - a. Push and hold down the ISP enable button
  - b. Press the Reset button
  - c. Release the Reset button
  - d. Wait 1 seconds
  - e. Release the ISP enable button

### 9 Terminal Application Setup

This chapter contains information about the terminal connection that exists on the *iMX OEM Carrier Board*, and how to setup a terminal application on the PC. The terminal connection connects UART1 of the i.MX RT to a virtual COM port over the USB interface available on J22. The terminal is commonly used during program development.

### 9.1 UART-to-USB Bridge

The UART-to-USB bridge chip (FT230XS-R from FTDI) on the *iMX OEM Carrier Board* connects to UART channel #1 on the i.MX RT. It exists to simplify connection to a PC because serial ports are not so common anymore, especially not on laptops. The USB port also offers the possibility to power the board.

There are two LEDs, transmit from the board (LED11) and receive to the board (LED10), that signal communication activity.

See picture below for locating relevant components.

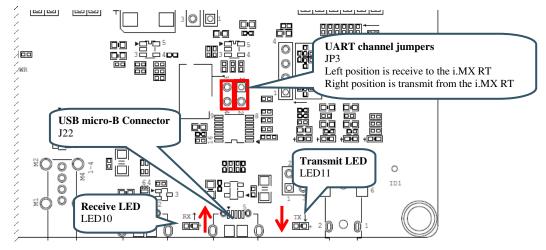


Figure 8 – UART-to-USB Bridge

### 9.2 Terminal Application on the PC

Begin by connecting the micro-B USB connector to J22, see picture above. Connect the other end of the USB cable to the PC. The PC will typically immediately begin installing drivers automatically for the UART-to-USB bridge that creates a Virtual COM port, if they are not already installed. If you have problems the drivers can be downloaded from the links below:

http://www.ftdichip.com/Drivers/VCP.htm http://www.ftdichip.com/Support/Documents/InstallGuides.htm

When the driver has been installed, a new COM port will list under "Ports" in the Device Manager as shown in Figure 9. Please note that the actual port number will most likely be different on your computer.

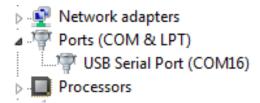


Figure 9 – Virtual COM port shown in device manager

The next step is to open a terminal application and attach it to the Virtual COM port that has just been created. The baud rate should be **115200**.

Some development environments/IDEs have a built-in terminal application that can be used. Sometimes it is better to have a terminal application with more features. For increased flexibility, we recommend using any of the two alternative terminal applications presented in the following sub-sections.

### 9.2.1 Tera Term Terminal Emulation Application

We recommend that you use **Tera Term** which can be downloaded and installed from either of the links below.

```
https://ttssh2.osdn.jp/index.html.en
http://sourceforge.jp/projects/ttssh2/releases/
```

Launch *Tera Term*. The first time it launches, it will show you the following dialog. Select the serial option. Assuming the USB cable is connected to the *iMX OEM Carrier Board*, there should be a COM port automatically populated in the list.

) TCP <u>/I</u> P	Hos <u>t</u> :	inyhost.exa	mple.com		Ŧ
		☑ Hist <u>o</u> ry © Te <u>I</u> net	TCP po	. E.	
		SSH	SSH version	SSH2	
		Other	Proto <u>c</u> ol:	UNSPEC	4
S <u>e</u> rial	Po <u>r</u> t:	COM15: US	B Serial Port (C	OM15)	•

Figure 10 – Tera Term New Connection Window

Configure the serial port settings (using the COM port number identified earlier) to 115200 baud rate, 8 data bits, no parity and 1 stop bit. To do this, go to Setup  $\rightarrow$  Serial Port and change the settings.

<u>P</u> ort:	COM +	ок
<u>B</u> aud rate:	115200 🗸	
<u>D</u> ata:	8 bit 🔻	Cancel
P <u>a</u> rity:	none 🔻	
<u>S</u> top:	1 bit 🔹	Help
Elow control:	none 🔻	
Transmit dela	iy	
0 mse	:c <u>/c</u> har 0	msec/ <u>l</u> ine

Figure 11 – Tera Term Serial Port Setup

Verify that the connection is open. If connected, Tera Term will show something like below in its title bar.

ile	Edit	Setun	Control	Window	Help

Figure 12 – Tera Term Menu

### 9.2.2 PuTTY terminal emulation application

Alternatively, you can use **PuTTY**. It is another commonly used terminal emulation application. PuTTY can be downloaded and installed from the link below.

#### http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

Launch PuTTY by either double clicking on the \*.exe file you downloaded or from the Start menu, depending on the type of download you selected.

In the window that launches, select the Serial radio button and enter the COM port number that you determined earlier. Also enter the baud rate, in this case 115200.

	Basic options for y	your PuTTY session
	Basic options for your PuTTY session Specify the destination you want to connect to Serial line Speed	
	M12	115200
COI	M12 nection type:	Rlogin © SSF

Figure 13 – PuTTY New Session Configuration

Click Open to open the serial connection. Assuming the FTDI cable is connected, and you entered the correct COM port, the terminal window will open. If the configuration is not correct, PuTTY will alert you.

### **10 Booting from External Memory**

The i.MX RT does not have any internal flash memory for storing the application. It must be stored in an external memory. On the *iMX RT OEM boards*, with revision up to, and including rev B2 this external memory is a 4 MByte OctalSPI, called EcoXiP ATXP032 from Adesto Technologies. On rev C1 boards, and later, the external memory is a 16 Mbyte QuadSPI flash, called IS25WP128. The i.MX RT MCU always boots (i.e., starts executing) from this OctalSPI/QuadSPI memory.

First, let's investigate the three use-cases when executing an application. The picture below illustrates the first main use-case when executing an application.

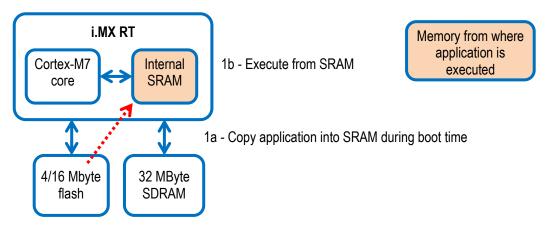
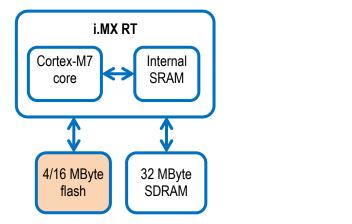


Figure 14 – i.MX RT and External Memories - Executing from SRAM

- The application is stored in the OctalSPI/QuadSPI flash and the bootloader (inside the i.MX RT) copies it into internal SRAM and then run from there.
  - a. The execution performance will be the highest in this setup.
  - b. CoreMark score (about) 2600 / 3000 when running at 528 / 600 MHz
  - c. During program development the application is just downloaded to internal SRAM by the debugger. There is no need to first download the application to the OctalSPI/QuadSPI flash memory. The address (in SRAM) where the application is downloaded is the same that it will be copied to by the on-chip bootloader in a final deployed system.

The second use-case is illustrated below. It is the default option supported when compiling and building the XiP targets.



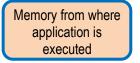


Figure 15 – i.MX RT and External Memories - Executing from External Flash

- The application is stored in the OctalSPI/QuadSPI flash and also executed from there. In this case, the internal SRAM is probably too small for the application or is simply used for other things like data storage.
  - a. When executing from an OctalSPI flash, the execution performance will be about 2/3 of the performance when executing from internal SRAM (assuming 5% cache miss, which is typical). CoreMark score is about 1736 / 1912 when running at 528 / 600 MHz
  - b. When executing from a QuadSPI flash, the performance is about half that of the OctalSPI flash.
  - c. During program development the application must be downloaded/flashed to the OctalSPI/QuadSPI memory before debugging actually starts. This is normally handled automatically by the IDE (Integrated Development Environment).

The third use-case is just a mixture of the two main ones. Two, or more memories, are used for storing executable code.

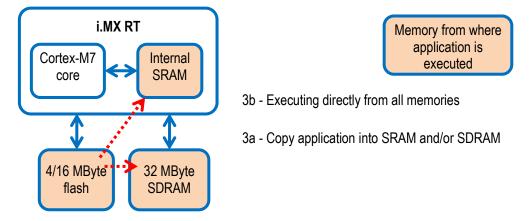


Figure 16 - i.MX RT and External Memories - Executing from all memories

- 3. The third setup is a mixture of the two above. Part of the application is copied into SRAM and/or SDRAM and part is executed directly from the OctalSPI/QuadSPI flash. A reason for placing part of an application in SRAM can be a need for highest performance for a data processing algorithm or a time critical interrupt service routine.
  - a. Note that this is a more complicated system architecture. The application must implement a dynamic loader that can copy code from the OctalSPI/QuadSPI flash to SRAM, either on-demand or in a pre-scheduled way. The linker script can be much more complicated because of this. There is no general solution for this system solution. Every system must be individually investigated to select and implement the best solution.

### 11 Things to Note

### 11.1 ESD Precaution

Please note that the *iMX RT OEM Board* and *iMX OEM Carrier Board* come without any case/box and all components are exposed for finger touches – and therefore extra attention must be paid to ESD (electrostatic discharge) precautions.

Make it a habit always to first touch the metal surface of one of the USB, SD or Ethernet connectors for a few seconds with both hands before touching any other parts of the boards. That way, you will have the same potential as the board and therefore minimize the risk for ESD.



Never touch directly on the *iMX RT OEM Board* and in general as little as possible on the *iMX OEM Carrier Board*. The pushbuttons on the *iMX OEM Carrier Board* have grounded shields to minimize the effect of ESD.

Note that Embedded Artists does not replace boards that have been damaged by ESD.

### **11.2 General Handling Care**

Handle the *iMX RT OEM Board* and *iMX OEM Carrier Board* with care. The boards are not mounted in a protective case/box and are not designed for rough physical handling. Connectors can wear out after excessive use. The *iMX OEM Carrier Board* is designed for prototyping use, and not for integration into an end-product.

For boards with LCD, do not exercise excessive pressure on the LCD glass area. That will damage the display. Also, do not apply pressure on the flex cables connecting the LCD/touch screen. These are relatively sensitive and can be damaged if too much pressure is applied to them.

Note that Embedded Artists does not replace boards where the LCD has been improperly handled.

### 11.3 OTP Fuse Programming

The i.MX RT MCU has on-chip OTP fuses that can be programmed, see NXP documents *IMXRT1050CEC*, *.MX RT1050 Crossover Processors for Consumer Products - Data Sheet* and *IMXRT1050RM*, *i.MX RT1050 Processor Reference Manual* and *IMXRT1060CEC*, *.MX RT1060 Crossover Processors for Consumer Products - Data Sheet* and *IMXRT1060RM*, *i.MX RT1060 Processor Reference Manual* for details. Once programmed, there is no possibility to reprogram them.

*iMX RT OEM Boards* are delivered without any OTP fuse programming. It is completely up to the OEM board user to decide if OTP fuses shall be programmed and, in that case, which ones.

Note that Embedded Artists does not replace iMX RT OEM Boards because of wrong OTP programming. It's the user's responsibility to be absolutely certain before OTP programming and not to program the fuses by accident.

### **11.4 Further Information**

The following NXP documents are important reference documents and should be consulted for functional details:

- IMXRT1050CEC, .MX RT1050 Crossover Processors for Consumer Products Data Sheet, latest revision
- IMXRT1050IEC, .MX RT1050 Crossover Processors for Industrial Products Data Sheet, latest revision
- IMXRT1050RM, i.MX RT1050 Processor Reference Manual, latest revision
- IMXRT1050CE, Chip Errata for the i.MX RT1050, latest revision
   Note: It is the user's responsibility to make sure all errata published by the manufacturer are taken note of. The manufacturer's advice should be followed.
- AN12094, Power consumption and measurement of i.MXRT1050, latest revision
- IMXRT1060CEC, i.MX RT1060 Crossover Processors for Consumer Products Data Sheet, latest revision
- IMXRT1060IEC, i.MX RT1060 Crossover Processors for Industrial Products Data Sheet, latest revision
- IMXRT1060RM, i.MX RT1060 Processor Reference Manual, latest revision
- IMXRT1060CE, Chip Errata for the i.MX RT1060, latest revision
   Note: It is the user's responsibility to make sure all errata published by the manufacturer are taken note of. The manufacturer's advice should be followed.
- AN12245, Power consumption and measurement of i.MXRT1060, latest revision
- AN12253, i.MXRT1060 Product Lifetime Usage Estimates, latest revision

### **12 Disclaimers**

Embedded Artists reserves the right to make changes to information published in this document, including, without limitation, specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Customer is responsible for the design and operation of their applications and products using Embedded Artists' products, and Embedded Artists accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the Embedded Artists' product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products. Customer is required to have expertise in electrical engineering and computer engineering for the installation and use of Embedded Artists' products.

Embedded Artists does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using Embedded Artists' products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). Embedded Artists does not accept any liability in this respect.

Embedded Artists does not accept any liability for errata on individual components. Customer is responsible to make sure all errata published by the manufacturer of each component are taken note of. The manufacturer's advice should be followed.

Embedded Artists does not accept any liability and no warranty is given for any unexpected software behavior due to deficient components.

Customer is required to take note of manufacturer's specification of used components, for example MCU, SDRAM and FLASH. Such specifications, if applicable, contains additional information that must be taken note of for the safe and reliable operation. These documents are stored on Embedded Artists' product support page.

All Embedded Artists' products are sold pursuant to Embedded Artists' terms and conditions of sale: http://www.embeddedartists.com/sites/default/files/docs/General\_Terms\_and\_Conditions.pdf

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by Embedded Artists for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN EMBEDDED ARTISTS' TERMS AND CONDITIONS OF SALE EMBEDDED ARTISTS DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF EMBEDDED ARTISTS PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY THE CEO OF EMBEDDED ARTISTS, PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, NUCLEAR, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE.

Resale of Embedded Artists' products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by Embedded Artists

for the Embedded Artists' product or service described herein and shall not create or extend in any manner whatsoever, any liability of Embedded Artists.

This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from national authorities.

### 12.1 Definition of Document Status

**Preliminary** – The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. Embedded Artists does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information. The document is in this state until the product has passed Embedded Artists product qualification tests.

**Approved** – The information and data provided define the specification of the product as agreed between Embedded Artists and its customer, unless Embedded Artists and customer have explicitly agreed otherwise in writing.