MCUXSDKMIMXRT117XRN

MCUXpresso SDK Release Notes for MIMXRT1170-EVK

Rev. 0 — 14 May, 2021 Release Notes

Contents

1 Overview

The MCUXpresso Software Development Kit (SDK) is a collection of software enablement for microcontrollers that includes peripheral drivers, high-level stacks including FatFs, lwIP, mbed TLS cryptography libraries, other middleware packages, multicore support and integrated RTOS support for FreeRTOS™ OS and Azure RTOS. In addition to the base enablement, the MCUXpresso SDK is augmented with demo applications, driver example projects, and API documentation to help the customers quickly leverage the support of the MCUXpresso SDK.

For more details about MCUXpresso SDK, see MCUXpresso Software Development Kit (SDK).

2 MCUXpresso SDK

As part of the MCUXpresso software and tools, MCUXpresso SDK is the evolution of Kinetis SDK, includes support for both LPC and i.MX System-on-Chips (SoC). The same drivers, APIs, and middleware are still available with support for Kinetis, LPC, and i.MX silicon. The MCUXpresso SDK adds support for the MCUXpresso IDE, an Eclipse-based toolchain that works with all MCUXpresso SDKs. Easily import your SDK into the new toolchain to access to all of the available components, examples, and demos for your target silicon. In addition to the MCUXpresso IDE, support for the MCUXpresso Config Tools allows easy cloning of existing SDK examples and demos, allowing users to leverage the existing software examples provided by the SDK for their own projects.

NOTE

In order to maintain compatibility with legacy Freescale code, the filenames and the source code in MCUXpresso SDK containing the legacy Freescale prefix **FSL** has been left as is. The **FSL** prefix has been redefined as the NXP Foundation Software Library. It is suggested to keep the downloaded SDK archive in the root directory of your drive to avoid any unexpected build issues caused by deep path of files.

3 Development tools

The MCUXpresso SDK was compiled and tested with these development tools:

- IAR Embedded Workbench for Arm® version 8.50.9
- MDK-Arm Microcontroller Development Kit (Keil)® 5.33
- Makefiles support with GCC revision 9-2020-q2-update GCC9 from Arm Embedded
- MCUXpresso IDE version 11.3.1
- Segger J-Link version 7.00

NOTE

There are some issues when using Segger J-Link Version 7.00 to debug CM4/CM7 flash related targets. NXP will work with Segger to get this issue fixed in the future. Before that, if you want to use JLink probe to debug flash related targets, please consult NXP for a temporary patch.

4 Supported development systems

This release supports boards and devices listed in Table 1. The boards and devices in bold were tested in this release.

Table 1. Supported MCU devices and development boards

Development boards	MCU devices
MIMXRT1170-EVK (SCH-32171 REV C), MIMXRT1170 EVK Board Hardware User's Guide (document MIMXRT1170EVKHUG)	MIMXRT1171AVM8A, MIMXRT1171CVM8A, MIMXRT1171DVMAA, MIMXRT1172AVM8A, MIMXRT1172CVM8A, MIMXRT1172DVMAA, MIMXRT1173CVM8A, MIMXRT1175AVM8A, MIMXRT1175CVM8A, MIMXRT1175DVMAA, MIMXRT1176AVM8A, MIMXRT1176CVM8A, **MIMXRT1176DVMAA**

5 MCUXpresso SDK release package

The MCUXpresso SDK release package content is aligned with the silicon subfamily it supports. This includes the boards, CMSIS, devices, documentation, middleware, and RTOS support.

5.1 Device support

The device folder contains the whole software enablement available for the specific System-on-Chip (SoC) subfamily. This folder includes clock-specific implementation, device register header files, device register feature header files, CMSIS derived device SVD, and the system configuration source files. Included with the standard SoC support are folders containing peripheral drivers, toolchain support, and a standard debug console.

The device-specific header files provide a direct access to the microcontroller peripheral registers. The device header file provides an overall SoC memory mapped register definition. The folder also includes the feature header file for each peripheral on the microcontroller.

The toolchain folder contains the startup code and linker files for each supported toolchain. The startup code is a CMSIS compliant startup code that efficiently transfers the code execution to the `main()` function.

5.1.1 Board support

The boards folder provides the board-specific demo applications, driver examples, RTOS, and middleware examples.

5.1.2 Demo applications and other examples

The demo applications demonstrate the usage of the peripheral drivers to achieve a system level solution. Each demo application contains a readme file that describes the operation of the demo and required setup steps.

The driver examples demonstrate the capabilities of the peripheral drivers. Each example implements a common use case to help demonstrate the driver functionality.

5.2 Middleware

5.2.1 USB stack

See *MCUXpresso SDK USB Stack User's Guide* (document MCUXSDKUSBSUG) for more information.

5.2.2 TCP/IP stack

The lwIP TCP/IP stack is pre-integrated with MCUXpresso SDK and runs on top of the MCUXpresso SDK Ethernet driver with *Ethernet-capable devices/boards*.

5.2.3 File system

The FatFs file system is integrated with the MCUXpresso SDK and can be used to access either the SD card or the USB memory stick when the SD card driver or the USB Mass Storage Device class implementation is used.

5.2.4 RTOS

FreeRTOS and Azure RTOS are pre-integrated with MCUXpresso SDK.

5.2.5 CMSIS

The MCUXpresso SDK is shipped with the standard CMSIS development pack, including the prebuilt libraries.

5.2.6 emWin

The MCUXpresso SDK is pre-integrated with the SEGGER emWin.

5.2.7 Azure RTOS Middleware

Azure RTOS middleware components may optionally be included in the MCUXpresso SDK: File-X, GUI-X, NetX Duo, USB-X. Examples are provided for each.

5.2.8 Other middleware

Optional middleware packages can be included in the release based on the user selection. See *<install_dir>/SW-Content-Register.txt* for a list of components and associated licenses.

6 Release contents

Table 2 provides an overview of the MCUXpresso SDK release package contents and locations.

Table 2. Release contents

Deliverable	Location
AWS examples	`<install_dir>/boards/<board_name>/aws_examples`
Azure RTOS example	`<install_dir>/boards/<board_name>/azure_rtos_examples`
Boards	`<install_dir>/boards`
CMSISArm Cortex®-M header files, DSP library source	`<install_dir>/CMSIS`
CMSIS drivers	`<install_dir>/devices/<device_name>/cmsis_drivers`
Component examples	`<install_dir>/boards/<board_name>/component_examples`
Cortex Microcontroller Software Interface Standard (CMSIS) driver examples	`<install_dir>/boards/<board_name>/cmsis_driver_examples`
Demo applications	`<install_dir>/boards/<board_name>/demo_apps`
Documentation	`<install_dir>/docs`

Table continues on the next page...

Table 2. Release contents (continued)

Deliverable	Location
Driver examples	<install_dir>/boards/<board_name>/driver_examples
Driver, SoC header files, extension header files and feature header files, utilities	<install_dir>/devices/<device_name>
eIQ examples	<install_dir>/boards/<board_name>/eiq_examples
emWin examples	<install_dir>/boards/<board_name>/emwin_examples
FatFS examples	<install_dir>/boards/<board_name>/fatfs_examples
FatFS stack	<install_dir>/middleware/fatfs
FreeRTOS examples	<install_dir>/boards/<board_name>/rtos_examples
Jpeg examples	<install_dir>/boards/<board_name>/jpeg_examples
LittleFS examples	<install_dir>/boards/<board_name>/littlefs_examples
LittleFS	<install_dir>/middleware/littlefs
LittlevGL examples	<install_dir>/boards/<board_name>/littlevgl_examples
lwIP demo applications	<install_dir>/boards/<board_name>/lwip_examples
lwIP stack	<install_dir>/middleware/lwip
mbed TLS	<install_dir>/middleware/mbedtls
mbed TLS examples	<install_dir>/boards/<board_name>/mbedtls_examples
mmCAU	<install_dir>/middleware/mmcau
mmCAU examples	<install_dir>/middleware/mmcau_examples
Multicore examples	<install_dir>/boards/<board_name>/multicore_examples
Multicore stack	<install_dir>/middleware/multicore
Openvg examples	<install_dir>/boards/<board_name>/openvg_examples
Openvg library	<install_dir>/middleware/openvg
Peripheral Drivers	<install_dir>/devices/<device_name>/drivers
RTOS Kernel Code	<install_dir>/rtos
Tools	<install_dir>/tools
USB demo applications	<install_dir>/boards/<board_name>/usb_examples
USB Documentation	<install_dir>/docs/usb
USB stack	<install_dir>/middleware/usb
Utilities such as debug console	<install_dir>/devices/<device_name>/utilities

Table continues on the next page...

Table 2. Release contents (continued)

Deliverable	Location
VGLite examples	<install_dir>/boards/<board_name>/vglite_examples
VGLite Graphic library	<install_dir>/middleware/vglite
Wifi_cypress examples	<install_dir>/boards/<board_name>/wifi_cypress_examples

7 MISRA compliance

All MCUXpresso SDK drivers comply to MISRA 2012 rules with exceptions in Table 3.

Table 3. MISRA exceptions

Exception rules	Description
Directive 4.4	Sections of code should not be commented out.
Directive 4.5	Identifiers in the same name space with overlapping visibility should be typographically unambiguous..
Directive 4.6	Typedefs that indicate size and signedness should be used in place of the basic numerical types.
Directive 4.8	If a pointer to a structure or union is never dereferenced within a translation unit, then the implementation of the object should be hidden.
Directive 4.9	A function should be used in preference to a function-like macro where they are interchangeable.
Directive 4.13	Functions which are designed to provide operations on a resource should be called in an appropriate sequence.
Rule 1.2	Language extensions should not be used.
Rule 2.3	A project should not contain unused type declarations.
Rule 2.4	A project should not contain unused tag declarations.
Rule 2.5	A project should not contain unused macro declarations.
Rule 2.6	A function should not contain unused label declarations.
Rule 2.7	There should be no unused parameters in functions.
Rule 4.2	Trigraphs should not be used.
Rule 5.1	External identifiers shall be distinct.
Rule 5.4	Macro identifiers shall be distinct.
Rule 5.9	Identifiers that define objects or functions with internal linkage should be unique.
Rule 8.7	Functions and objects should not be defined with external linkage if they are referenced in only one translation unit.
Rule 8.9	An object should be defined at block scope if its identifier only appears in a single function.
Rule 8.11	When an array with external linkage is declared, its size should be explicitly specified.
Rule 8.13	A pointer should point to a const-qualified type whenever possible.

Table continues on the next page...

Table 3. MISRA exceptions (continued)

Exception rules	Description
Rule 10.5	The value of an expression should not be cast to an inappropriate essential type.
Rule 11.4	A conversion should not be performed between a pointer to object and an integer type.
Rule 11.5	A conversion should not be performed from pointer to void into pointer to object.
Rule 12.1	The precedence of operators within expressions should be made explicit.
Rule 12.3	The comma operator should not be used.
Rule 12.4	Evaluation of constant expressions should not lead to unsigned integer wrap-around.
Rule 13.3	A full expression containing an increment (`++`) or decrement (`--`) operator should have no other potential side effects other than that caused by the increment or decrement operator.
Rule 15.4	There should be no more than one break or go to statement used to terminate any iteration statement.
Rule 17.5	The function argument corresponding to a parameter declared to have an array type shall have an appropriate number of elements.
Rule 17.8	A function parameter should not be modified.
Rule 19.2	The union keyword should not be used.
Exception rules	Description
Rule 20.1	`#include` directives should only be preceded by preprocessor directives or comments.
Rule 20.10	The `#` and `##` preprocessor operators should not be used.
Rule 21.1	`#define` and `#undef` shall not be used on a reserved identifieror reserved macro name.
Rule 21.2	A reserved identifier or macro name shall not be declared.
Rule 21.12	The exception handling features of `<fenv.h>` should not be used.

8 Known issues

8.1 Maximum file path length in Windows operating system

The Windows operating system imposes a 260-character maximum length for file paths. When installing the MCUXpresso SDK, place it in a directory close to the root to prevent file paths from exceeding the maximum character length specified by the Windows operating system. The recommended location is the *C:\nxp* folder.

8.2 New project wizard compile failure

The following components request the user to manually select other components that they depend upon in order to compile. These components depend on several other components and the New Project Wizard (NPW) is not able to decide which one is needed by the user.

NOTE

`xxx` means core variants, such as, `cm0plus`, `cm33`, `cm4`, `cm33_nodsp`.

Some peripheral drives and middleware components have complex dependencies and are not fully supported in new project wizard. For example, `enet_qos`, which includes cache driver is unsupported for new project wizard and will lead to compile failure if selected while creating new project(s).

Also for low-level adapter components, currently the different types of the same adapter cannot be selected at the same time. For example, if there are two types of timer adapters, `gpt_adapter` and `pit_adapter`, only one can be selected as timer adapter in one project at a time. Duplicate implementation of the function results in an error.

8.3 CMSIS-PACK svd issue

CMSIS-PACK DFP installation will take quite a while. When installing cmsis-pack DFP, Keil MDK will process the MCU SVD file: the large size of SVD file takes considerable time to finish this conversion. During the interval the installation progress looks like hanging, but it will eventually be finished after approximate 20 minutes.

8.4 CMSIS-PACK new project compile failure

The generated configuration cannot be applied globally. Some components such as `enet_qos_xxx` (`xxx` means core variants like cm4, cm7) which include cache drivers are unsupported for new project wizard of CMSIS pack and will lead to compile failure if selected while creating new project(s).

8.5 MCUXPresso IDE limitation

1. **Can't debug cm4 sdram related demos with CMSIS-DAP**

   MCUXpresso IDE won't support initialization of sdram when debugging.

2. **Can't debug cm4 demos with JLINK**

   Text section in MCUXpresso IDE projects are linked to RAM. Currently, SEGGER doesn't support debugging under cache enabled area.

8.6 IAR debug limitation

1. **Debugging cm4 flexspi_nor projects will hang on LPI2C master operations.**

   IAR CM4 debugging will suspend LPI2C master operations. To debug demos related to LPI2C master, please set **DBGEN** bit for Master **Control Register (MCR)**.

2. **CM4 flash target demos can't be debugged on IAR with JLINK.**

8.7 An extra option required when using CMSIS-DAP to debug

When using CMSIS-DAP to debug CM4 sdram related target (flexspi_nor_sdram, sdram_txt, etc) in IAR. An extra option need to be specified in debugger settings.

Please refer to **Note** in Section 4.2 in *Getting Started with MCUXpresso SDK for MIMXRT1170-EVK* (document MCUXSDKMIMXRT117XGSUG).

8.8 Issue about debugging flash release target on MDK with JLINK

Please modify size value from **0x0800** to **0x8000** on RAM for Algorithm window under Flash Download option.

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

Right to make changes - NXP Semiconductors 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.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer’s applications and products. Customer’s responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer’s applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 14 May, 2021

Document identifier: MCUXSDKMIMXRT117XRN

arm