Document Number: MCUXSDKIMX8MQRN
Rev. B, 12/2017

# MCUXpresso SDK Release Notes for i.MX 8M Quad

**Contents**



## 1 Overview

The MCUXpresso Software Development Kit (MCUXpresso SDK) is a collection of software enablement for Microcontrollers (MCU) that includes peripheral drivers, other middleware packages, such as multicore support, and integrated RTOS support for FreeRTOS OS. In addition to the base enablement, the MCUXpresso SDK is augmented with demo applications and driver example projects, and API documentation to help the customers quickly leverage the support of the MCUXpresso SDK.

## 2 MCUXpresso SDK

As part of the MCUXpresso software and tools, MCUXpressoSDK is the evolution of Kinetis SDK v2.0.0 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.

**NOTE**

In order to maintain compatibility with legacy FSL code, the filenames and source code in MCUXpresso SDK containing the legacy

Freescale prefix 'FSL' have been left as is. The 'FSL' prefix has been redefined as the NXP Foundation Software Library.

# 3 Development Tools

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

- IAR Embedded Workbench for ARM version 8.20.1
- Makefiles support with GCC revision v6-2017-q2-update from ARM Embedded

# 4 Supported Development Systems

This release supports boards and devices listed in this table. The boards and devices in boldface were tested in this release.

**Table 1. Supported MCU devices and development boards**

| Development boards | MCU devices |
|---|---|
| **MIMX8MQ6-EVK** | MIMX8MQ5CVAHZ, MIMX8MQ5CZKHZ, MIMX8MQ5DVAJZ, MIMX8MQ5DZKJZ, MIMX8MQ6CVAHZ, MIMX8MQ6CZKHZ, **MIMX8MQ6DVAJZ**, MIMX8MQ6DZKJZ, MIMX8MQ7CVAHZ, MIMX8MQ7CZKHZ, MIMX8MQ7DVAJZ, MIMX8MQ7DZKJZ |

# 5 Release Contents

This table provides an overview of the MCUXpresso SDK release package contents and locations.

**Table 2. Release contents**

| Deliverable | Location |
|---|---|
| Boards | <install_dir>/boards |
| Demo applications | <install_dir>/boards/<board_name>/demo_apps |
| Driver examples | <install_dir>/boards/<board_name>/driver_examples |
| RTOS examples | <install_dir>/boards/<board_name>/rtos_examples |
| Multicore examples | <install_dir>/boards/<board_name>/multicore_examples |
| Documentation | <install_dir>/docs |
| Driver, SoC header files, extension header files and feature header files, utilities | <install_dir>/devices/<device_name> |
| Multicore stack | <install_dir>/middleware/multicore |
| Cortex Microcontroller Software Interface Standard (CMSIS) ARM Cortex®-M header files, DSP library source | <install_dir>/CMSIS |
| Peripheral Drivers | <install_dir>/devices/<device_name>/drivers |
| Utilities such as debug console | <install_dir>/devices/<device_name>/utilities |

*Table continues on the next page...*

**Table 2. Release contents (continued)**

| | |
|---|---|
| RTOS Kernel Code | <install_dir>/rtos |
| Tools | <install_dir>/tools |

# 6 MCUXpresso SDK Release Package

The MCUXpresso SDK release package contents are with the SoC it supports. This includes the boards, devices, documentation, and RTOS support.

## 6.1 Device support

The device folder contains available software enablement for the specific SoC subfamily. This folder includes clock-specific implementation, device register header file, device register feature header file, and the system configuration source files. Included with the standard SoC support are folders containing peripheral drivers, toolchain support, and a simple debug console.

The device-specific header files provide a direct access to the MCU peripheral registers. The device header file provides an overall System-on-Chip (SoC) memory mapped register definition. In addition to the overall device memory mapped header file, the MCUXpresso SDK also includes the feature header file for each peripheral instantiated on the SoC.

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

### 6.1.1 Board support

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

### 6.1.2 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.

The RTOS folder contains examples demonstrating the use of the included source.

**NOTE**

Some demo applications and driver examples are intended for a single ARM Cortex-M4 application reference. They cannot support running with the Linux BSP, which requires additional service protocol implementation. See the readme file for the specific application to know whether it supports running with the Linux BSP.

## 6.2 Middleware

### 6.2.1 RTOS

The MCUXpresso SDK is integrated with FreeRTOS OS.

### 6.2.2 CMSIS

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

# 7 MISRA Compliance

All MCUXpresso SDK drivers comply to MISRA 2004 rules with the following exceptions.

| Exception Rules | Description |
|---|---|
| 1.1 | All code shall conform to ISO 9899:1990 Programming languages - C, amended and corrected by ISO/IEC 9899/COR1:1995, ISO/IEC 9899/AMD1:1995, and ISO/IEC |
| 2.4 | Sections of code should not be commented out. |
| 5.1 | Identifiers (internal and external) shall not rely on the significance of more than 31 characters. |
| 6.3 | typedefs that indicate size and signedness should be used in place of the basic types. |
| 6.4 | Bitfields shall only be defined to be of type unsigned int or signed int. |
| 8.1 | Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call. |
| 8.5 | There shall be no definitions of objects or functions in a header file. |
| 8.1 | All declarations and definitions of objects or functions at file scope shall have internal linkage unless external linkage is required. |
| 8.12 | When an array is declared with external linkage, its size shall be stated explicitly or defined implicitly by initialization. |
| 10.1 | The value of an expression of integer type shall not be implicitly converted to a different underlying type if:<br>a. it is not a conversion to a wider integer type of the same signedness, or<br>b. the expression is complex, or<br>c. the expression is not constant and is a function argument, or<br>d. the expression is not constant and is a return expression. |
| 10.3 | The value of a complex expression of integer type shall only be cast to a type that is not wider and of the same signedness as the underlying type of the expression. |
| 11.3 | A cast should not be performed between a pointer type and an integral type. |
| 11.4 | A cast should not be performed between a pointer to object type and a different pointer to object type. |
| 11.5 | A cast shall not be performed that removes any const or volatile qualification from the type addressed by a pointer. |
| 12.2 | The value of an expression shall be the same under any order of evaluation that the standard permits. |
| 12.4 | The right-hand operand of a logical && or \| \| operator shall not contain side effects. |
| 12.6 | The operands of logical operators (&&, \| \|, and !) should be effectively boolean. Expressions that are effectively boolean should not be used as operands to operators other than (&&, \| \|, !, =, ==, !=, and ?:). |
| 12.13 | The increment (++) and decrement (--) operators should not be mixed with other operators in an expression. |
| 14.3 | Before preprocessing, a null statement shall only occur on a line by itself; it may be followed by a comment, provided that the first character following the null statement is a whitespace character. |
| 14.5 | The continue statement shall not be used. |
| 14.7 | A function shall have a single point of exit at the end of the function. |
| 16.1 | Functions shall not be defined with a variable number of arguments. |
| 17.4 | Array indexing shall be the only allowed form of pointer arithmetic. |
| 18.4 | Unions shall not be used. |
| 19.1 | #include statements in a file should only be preceded by other preprocessor directives or comments. |
| 19.1 | In the definition of a function-like macro, each instance of a parameter shall be enclosed in parentheses unless it is used as the operand of # or ##. |
| 20.4 | Dynamic heap memory allocation shall not be used. |
| 20.9 | The input/output library <stdio.h> shall not be used in production code. |

**Figure 1. MISRA exceptions**

# 8 Known Issues

## 8.1 Maximum file path length in Windows® 7 Operating System

Windows 7 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.

***How to Reach Us:***

**Home Page:**
nxp.com

**Web Support:**
nxp.com/support



Document Number MCUXSDKIMX8MQRN
Revision B, 12/2017