

MCUXSDKMIMXRT106XRN

MCUXpresso SDK Release Notes for EVK-MIMXRT1060

Rev. 2.11.0 — 12 January 2021

Release Notes

1 Overview

The MCUXpresso SDK is a comprehensive software enablement package designed to simplify and accelerate application development with Arm® Cortex®-M-based devices from NXP, including its general purpose, crossover and Bluetooth™-enabled MCUs. MCUXpresso SW and Tools for DSC further extends the SDK support to current 32-bit Digital Signal Controllers. The MCUXpresso SDK includes production-grade software with integrated RTOS (optional), integrated enabling software technologies (stacks and middleware), reference software, and more.

In addition to working seamlessly with the MCUXpresso IDE, the MCUXpresso SDK also supports and provides example projects for IAR, KEIL, and GCC with Cmake. 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.

Underscoring our commitment to high quality, the MCUXpresso SDK is MISRA compliant and checked with Coverity® static analysis tools. For details on MCUXpresso SDK, see [MCUXpresso-SDK: Software Development Kit for MCUXpresso](#).

2 MCUXpresso SDK

As part of the MCUXpresso software and tools, MCUXpresso SDK is the evolution of Kinetis SDK, includes support for LPC, DSC, and i.MX System-on-Chip (SoC). The same drivers, APIs, and middleware are still available with support for Kinetis, LPC, DSC, 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.

In order to maintain compatibility with legacy Freescale code, the filenames and 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.

3 Development tools

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

- Keil MDK, version is 5.36
- MCUXpresso IDE, version is 11.5.0
- GCC Arm Embedded, version is 10-2021.07
- IAR Embedded Workbench for Arm, version is 9.20.2

4 Supported development system

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

Contents

1	Overview.....	1
2	MCUXpresso SDK.....	1
3	Development tools.....	1
4	Supported development system.....	1
5	MCUXpresso SDK release package	2
6	Release contents.....	5
7	MISRA compliance.....	6
8	Known Issues.....	8
9	Change Log.....	10



Table 1. Supported boards and devices

Development boards	MCU devices
EVK-MIMXRT1060	MIMXRT1062DVL6A , MIMXRT1061CVL5A, MIMXRT1061DVJ6A, MIMXRT1062CVL5A, MIMXRT1062DVL6A, MIMXRT1062DVJ6A, MIMXRT1062CVJ5A, MIMXRT1061CVJ5A, MIMXRT1061DVL6A

5 MCUXpresso SDK release package

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

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, 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 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, and middleware examples.

5.1.2 Demo application 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 Operating System

5.2.1 FreeRTOS

Real-time operating system for microcontrollers from Amazon.

5.2.2 Azure RTOS

Azure RTOS middleware (FileX, GUIX, LevelX, NetX Duo, USBX) and the ThreadX RTOS are optional components in MCUXpresso SDK. Source code for these items is included in the `rtos/azure-rtos` directory, and examples for the target board are located in `boards/evkbimxrt1060/azure_rtos_examples`.

For more information, see the *MCUXpresso SDK Azure RTOS Release Notes* and *Getting Started with MCUXpresso SDK for Azure RTOS*.

5.3 Middleware

5.3.1 AWS IoT

Amazon Web Service (AWS) IoT Core SDK.

5.3.2 Azure RTOS FileX

A file system based on Azure RTOS.

5.3.3 Azure RTOS GUIX

A GUI library based on Azure RTOS.

5.3.4 Azure RTOS LevelX

NOR/NAND Flash wear leveling component

5.3.5 Azure RTOS NetX Duo

A network protocol stack based on Azure RTOS.

5.3.6 Azure RTOS ThreadX

Azure RTOS ThreadX.

5.3.7 Azure RTOS USBX

A USB Library Based on Azure RTOS.

5.3.8 canopen

MicroCANopen Stack from Embedded Solutions Academy

5.3.9 Crank Storyboard GUI

Crank Storyboard GUI Engine from Crank Software.

5.3.10 Essential Audio Processing Library

Audio processing blocks for enhancing the tonal and spatial perception of sound in audio applications.

5.3.11 eIQ

eIQ machine learning SDK containing:

- Arm CMSIS-NN library (neural network kernels optimized for Cortex-M cores)
- Inference engines:
 - TensorFlow Lite Micro
 - DeepView™ RT
- Example code for TensorFlow Lite Micro, Glow, and DeepView RT

5.3.12 Embedded Wizard GUI

The MCUXpresso SDK is pre-integrated with the Embedded Wizard GUI framework. The integration demonstrates state-of-the-art GUI applications by using hardware accelerated graphic operations (where applicable, otherwise pure software). Embedded Wizard GUI from TARA Systems.

5.3.13 emWin

The MCUXpresso SDK is pre-integrated with the SEGGER emWin GUI middleware. The AppWizard provides developers and designers with a flexible tool to create stunning user interface applications, without writing any code.

5.3.14 Fatfs

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.

5.3.15 FreeMASTER

FreeMASTER communication driver for 32-bit platforms.

5.3.16 IEC60730B Safety Library

NXP IEC60730B Safety Library

5.3.17 IoT Sensing Software Development Kit (ISSDK)

The IoT Sensing Software Development Kit (ISSDK) is the embedded software framework enabling NXP's digital and analog sensors for IoT applications. ISSDK combines a set of robust sensor drivers and algorithms along with example applications to allow users to get started with using NXP IoT motion & pressure sensors. ISSDK is being offered as a middleware component in MCUXpresso SDK.

5.3.18 JPEG library

JPEG library

5.3.19 littlefs

LittleFS filesystem stack

5.3.20 LVGL

LVGL Open Source Graphics Library

5.3.21 lwIP

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.

For details, see the *lwIP TCP/IP Stack and MCUXpresso SDK Integration User's Guide* (document MCUXSDKLWIPUG).

5.3.22 Maestro Audio Framework for MCU

Maestro Audio Framework library for MCU

5.3.23 mbedTLS

mbedtls SSL/TLS library

5.3.24 MCU Boot

MCU Bootloader source code.

5.3.25 NXP Touch Library

NXP Touch Library

5.3.26 NXP Wi-Fi

The MCUXpresso SDK provides driver for NXP Wi-Fi external modules. The Wi-Fi driver is integrated with LWIP TCPIP stack and demonstrated with several network applications (iperf and AWS IoT).

For more information, see Getting Started with NXP based Wireless Modules and i.MX RT Platform Running on RTOS (document: UM11441).

5.3.27 sdmmc stack

The SDMMC software is integrated with MCUXpresso SDK to support SD/MMC/SDIO standard specification. This also includes a host adapter layer for bare-metal/RTOS applications.

5.3.28 Secure Element Host Library

Secure Element Host Library

5.3.29 USB Host, Device, OTG Stack

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

5.3.30 USB Type-C PD Stack

See the *MCUXpresso SDK USB Type-C PD Stack User's Guide* (document MCUXSDKUSBPDUG) for more information.

NOTE

The USB TYPE-C PD stack supports IAR only.

5.3.31 Wireless EdgeFast Bluetooth PAL

For more information, see the MCUXpresso SDK EdgeFast Bluetooth Protocol Abstraction Layer User's Guide.

6 Release contents

[Table 2](#) provides an overview of the MCUXpresso SDK release package contents and locations.

Table 2. MCUXpresso SDK release package contents and locations

Deliverable	Location
Boards	INSTALL_DIR/boards
Demo Applications	INSTALL_DIR/boards/<board_name>/demo_apps
Driver Examples	INSTALL_DIR/boards/<board_name>/driver_examples
Board Project Template for MCUXpresso IDE NPW	INSTALL_DIR/boards/<board_name>/project_template
Driver, SoC header files, extension header files and feature header files, utilities	INSTALL_DIR/devices/<device_name>
Peripheral drivers	INSTALL_DIR/devices/<device_name>/drivers
Toolchain linker files and startup code	INSTALL_DIR/devices/<device_name>/<toolchain_name>
Utilities such as debug console	INSTALL_DIR/devices/<device_name>/utilities

Table continues on the next page...

Table 2. MCUXpresso SDK release package contents and locations (continued)

Deliverable	Location
Device Project Template for MCUXpresso IDE NPW	INSTALL_DIR/devices/<device_name>/project_template
CMSIS Arm Cortex-M header files, DSP library source	INSTALL_DIR/CMSIS
Components and board device drivers	INSTALL_DIR/components
Documents	INSTALL_DIR/docs
RTOS	INSTALL_DIR/rtos
Release Notes, Getting Started Document and other documents	INSTALL_DIR/docs
Tools such as shared cmake files	INSTALL_DIR/tools

7 MISRA compliance

All MCUXpresso SDK drivers comply to MISRA 2012 rules with exceptions in [Table 3](#).

Table 3. MISRA exception rules

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.

Table continues on the next page...

Table 3. MISRA exception rules (continued)

Exception rules	Description
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.
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.

Table continues on the next page...

Table 3. MISRA exception rules (continued)

Exception rules	Description
Rule 17.8	A function parameter should not be modified.
Rule 19.2	The union keyword should not be used.
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 identifier or 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 7 operating system

The 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:\<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.

Components:issdk_mag3110, issdk_host, systick, gpio_kinetis, gpio_lpc, issdk_mpl3115, sensor_fusion_agm01, sensor_fusion_agm01_lpc, issdk_mma845x, issdk_mma8491q, issdk_mma865x, issdk_mma9553, and CMSIS_RTOS2.

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.

NOTE

Most of middleware components have complex dependencies and are not fully supported in new project wizard. Adding a middleware component may result in compile failure.

8.3 RAM targets build issue in CMSIS bsp pack

CMSIS pack does not support different macro definitions for different targets, all RAM targets for projects inside CMSIS BSP PACKs for RT10XX boards will get the same macro definitions with Flash targets, resulting in build failure. To pass build for RAM targets, manually update the XIP_EXTERNAL_FLASH and XIP_BOOT_HEADER_ENABLE value to 0 in RTE_Components.h.

8.4 Non XIP target debug issue on toolchain MDK

When debugging non XIP targets in flash boot mode, if application changes any settings which have impacts on flexspi, the build output window might show "Debug access failed" when start debugging next time. It is recommended to keep the board in serial downloader mode when debugging non XIP targets.

8.5 CMSIS PACK new project compile failure

The generated configuration cannot be applied globally. The components, serial_manager_usb_cdc_virtual and serial_manager_usb_cdc_virtual_xxx (xxx means core variants like cm0plus, cm33, cm4, and cm33_nodsp) are unsupported for new project wizard of CMSIS pack and will lead to compile failure if selected while creating new project(s).

8.6 Corrupted data in freertos_lpspi_b2b (slave) example

Corrupted data in freertos_lpspi_b2b(slave) example.

Tool: Keil MDK

Target: freertos_lpspi_b2b_slave_flexspi_nor_debug.

Changing the optimization level from -O1 to -O0 can avoid the issue. However, the optimization level -O1 is not the root cause

8.7 Cannot add SDK components into FreeRTOS projects

It is not possible to add any SDK components into FreeRTOS project using the MCUXpresso IDE New Project wizard.

8.8 Maestro recording to a file has missing audio samples

When recording audio from the on-board mic and saving the output to a file on SD card, there are missing samples in the output pcm file. The maestro_demo design causes the issue. Writing data to SD card should be buffered and handled in a separate thread. The fix for the issue is planned for the next release.

8.9 Log output may be mixed in shell/hfp example

When multiple tasks print the log, the serial port terminal output has the probability to appear mixed.

8.10 LE encryption failure causes connection to fail

There can be a corner case when LE link encryption can fail. This occurs when device under test (DUT); RT Bluetooth controller here, while waiting for the response to LL_SLAVE_FEATURES_REQ, instead receives the LL_ENC_REQ response from a remote device.

This causes deadlock scenario where DUT and remote devices are stuck waiting for response from peer.

8.11 Connection disconnects with 7.5 ms connection interval

When wireless example works as a peripheral, the central role with 7.5 ms connection interval connects to the wireless example. However, when using the 6th/7th/8th central to connect the wireless example, all the previous connections with the previous centrals are disconnected.

When wireless example works as central and the connection interval is 7.5 ms, the 4th peripheral is not scanned.

8.12 a2dp sink demo: Noise may occur when phone plays music with other operations

Noise may occur when phone plays music with other operations.

For example:

- Switches the WIFI network of the phone when playing music
- Switches music when playing the online music

8.13 Wireless EdgeFast_Bluetooth stack is not supported by any kits

Wireless EdgeFast_Bluetooth stack can be downloaded by EVK-MIMXRT1060-AGM01, EVK-MIMXRT1060-OM13790HOST and EVK-MIMXRT1060-OM13588 in <https://mcuxpresso.nxp.com/en/welcome>. However, it is not supported in these kits.

8.14 Debugger cannot connect to target board when connecting TP11 of RT1060 to J11 (pin 2) of AW-AM510 uSD wireless module

To download hands-free or handsfree_ag example to RT1060, J11 (pin 2) should only be connected to TP11 after downloading executable file to target board.

9 Change Log

Change log of software components included in the package, see MCUXpresso SDK ChangeLog_MIMXRT1062.pdf.

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Limited warranty and liability — 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.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, ICODE, JCOP, LIFE, VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, Altivec, CodeWarrior, ColdFire, ColdFire+, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, Tower, TurboLink, EdgeScale, EdgeLock, eIQ, and Immersive3D are trademarks of NXP B.V. All other product or service names are the property of their respective owners. AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, uVision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org. M, M Mobileye and other Mobileye trademarks or logos appearing herein are trademarks of Mobileye Vision Technologies Ltd. in the United States, the EU and/or other jurisdictions.

© NXP B.V. 2022.

All rights reserved.

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

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

Date of release: 12 January 2021

Document identifier: MCUXSDKMIMXRT106XRN

