

MCUXSDKMIMXRT1021RN

MCUXpresso SDK Release Notes for EVK-MIMXRT1020

Rev. 0 — 18 January 2021

Release Notes

1 Overview

The MCUXpresso Software Development Kit (SDK) is a collection of software enablement for microcontrollers that includes peripheral drivers, high-level stacks including other middleware packages. 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 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 v2.7.0, 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 5.33
- MCUXpresso IDE, version is 11.3.0
- GCC ARM Embedded, version is 9-2020-q2-update
- IAR Embedded Workbench for ARM, version is 8.50.9

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.

Table 1. Supported boards and devices

Development boards	MCU devices
EVK-MIMXRT1020	MIMXRT1021DAG5A , MIMXRT1021DAF5A, MIMXRT1021DAG5A, MIMXRT1021CAG4A, MIMXRT1021CAF4A

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.....	4
7	MISRA compliance.....	5
8	Known Issues.....	7
9	Change Log.....	8



5 MCUXpresso SDK release package

The MCUXpresso DSC 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 CMSIS DSP Lib

5.2.1 CMSIS DSP Library

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

5.3 Operating System

5.3.1 FreeRTOS

The MCUXpresso SDK is integrated with FreeRTOS OS.

5.3.2 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/evkbimxrt1020/azure_rtos_examples*.

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

5.4 Middleware

5.4.1 AWS IoT

AWS IoT

5.4.2 canopen

canopen library

5.4.3 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.

5.4.4 FreeMASTER

FreeMASTER communication driver

5.4.5 ISSDK

IoT Sensing SDK (ISSDK) provides sensor drivers and reference examples for NXP's IoT sensors.

5.4.6 littlevgl

littlevgl graphics library

5.4.7 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).

5.4.8 mbedtls

mbedtls SSL/TLS library

5.4.9 mcu-boot

mcu-boot

5.4.10 Safety

NXP IEC60730B Safety Library

5.4.11 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.

5.4.12 Touch

NXP Touch Library

5.4.13 USB Authentication

USB Type-C PD authentication

5.4.14 USB stack

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

5.4.14.1 Peripheral devices tested with USB Host stack

[Table 2](#) provides a list of USB devices tested with the USB Host stack.

Table 2. Peripheral devices

Device type	Device
USB HUB	BELKIN F5U233BELKIN F5U304BELKIN F5U307BELKIN F4U040UNITEK Y-2151Z-TEK ZK032AHYUNDAI HY-HB608
USB flash drive	ADATA C008 32 GBADATA S102 8 GADATA S102 16 GVerbatim STORE N GO USB Device 8 GKingston DataTraveler DT101 G2SanDisk Cruzer Blade 8 GBUnisplendour 1 Glmation 2 GBV-mux 2 GBSanmina-SCI 128 MCorporate Express 1 GTOSHIBA THUHYBS-008G 8 GTranscend JF700 8 GNetac U903 16 GSSK SFD205 8 GBReX 4 GBSAMSUNG USB3.0 16GB
USB card reader/adaptor	SSK TF adapterKawau Multi Card ReaderKawau TF adapterKawau SDHC cardUSB Mouse DELL MS111-PDELL M066U0ADELL MUAVDEL8TARGUS AMU76APDELL MD56U0DELL MS111-TRAPOO M110
USB Keyboard	DELL SK8135DELL SK8115

5.4.15 USB Type-C PD Stack

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

NOTE

The USB TYPE-C PD stack supports IAR only.

5.4.16 wifi_wiced

The MCUXpresso SDK provides integration with Cypress Wiced SDK supporting the Murata Type 1DX and Azurewave AWNM372SM modules based on the CYW4343W and CYW43438 processors.

6 Release contents

[Table 3](#) provides an overview of the MCUXpresso SDK release package contents and locations.

Table 3. 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>/utilities
Driver, SoC header files, extension header files and feature header files, utilities	INSTALL_DIR/devices/<device_name>
CMSIS drivers	INSTALL_DIR/devices/<device_name>/cmsis_drivers

Table continues on the next page...

Table 3. MCUXpresso SDK release package contents and locations (continued)

Deliverable	Location
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
Device Project Template for MCUXpresso IDE NPW	INSTALL_DIR/devices/<device_name>/utilities
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 4](#).

Table 4. 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.

Table continues on the next page...

Table 4. MISRA exception rules (continued)

Exception rules	Description
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.
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.

Table continues on the next page...

Table 4. MISRA exception rules (continued)

Exception rules	Description
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.
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 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.

8.7 safety_iec60730b cloned project fails to build

When you use the MCUXpresso Config Tool to clone the "safety_iec60730b" project in MCUXpresso SDK package, the created project fails to build.

The build fails because the post-build setup for CRC is incorrect. Therefore, It is recommended to use the "safety_iec60730b" project in MCUXpresso SDK package.

9 Change Log

Change log of software components included in the package, see MCUXpresso SDK ChangeLog_MIMXRT1021.pdf.

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.

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, µVision, 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.

© NXP B.V. 2021.

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: 18 January 2021

Document identifier: MCUXSDKMIMXRT1021RN

