NXP Semiconductors
User's Guide

Document Number: MCUXSDKMIMXRT106XCYPRESSWIFIRN
Rev 0, 04/2019

MCUXpresso SDK Release Notes Supporting EVK-MIMXRT1060

Contents

Chapter 1
Overview

The MCUXpresso Software Development Kit (SDK) is a collection of software enablement for Microcontrollers that includes peripheral drivers, high-level stacks including USB and lwIP, integration with WolfSSL and mbed TLS cryptography libraries, other middleware packages, such as multicore support and FatFs, 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.

For more details about MCUXpresso SDK, see the MCUXpresso SDK homepage MCUXpresso-SDK: Software Development Kit.

NOTE

See the attached Change Logs section at the end of this document to reference the device-specific driver logs, middleware logs, and RTOS log.

Chapter 2
MCUXpresso SDK

As part of the MCUXpresso software and tools, MCUXpressoSDK is the evolution of Kinetis SDK v2.5.1, 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 have 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 for easy cloning of existing SDK examples and demos, allowing users to easily 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 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.

Chapter 3
Development tools

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

- IAR Embedded Workbench for Arm version 8.32.3
- MDK-Arm Microcontroller Development Kit (Keil)® 5.26
- Makefiles support with GCC revision 7-2018-q2-update from Arm Embedded
- MCUXpresso IDE v10.3.1

Chapter 4
Supported development systems

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

Table 1. Supported MCU devices and development boards

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

Chapter 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
TinyCBOR	<install_dir>/rtos/amazon-freertos/lib/third_party/tinycbor
Qualcomm WiFi	<install_dir>/middleware/wifi_qca
Secure element host library	<install_dir>/middleware/se_hostlib
Demo applications	<install_dir>/boards/<board_name>/demo_apps
USB demo applications	<install_dir>/boards/<board_name>/usb_examples
lwIP demo applications	<install_dir>/boards/<board_name>/lwip_examples
Driver examples	<install_dir>/boards/<board_name>/driver_examples
CMSIS driver examples	<install_dir>/boards/<board_name>/cmsis_driver_examples
FatFS examples	<install_dir>/boards/<board_name>/fatfs_examples
emWin examples	<install_dir>/boards/<board_name>/emwin_examples
Secure element host library examples	<install_dir>/boards/<board_name>/se_hostlib_examples
RTOS examples	<install_dir>/boards/<board_name>/rtos_examples
WolfSSL examples	<install_dir>/boards/<board_name>/wolfssl_examples
Qualcomm WiFi stack examples	<install_dir>/boards/<board_name>/wifi_qca_examples
Cypress WiFi stack examples	<install_dir>/boards/<board_name>/wifi_cypress_examples
Secure element host library examples	<install_dir>/boards/<board_name>/se_hostlib_examples
mbed TLS examples	<install_dir>/boards/<board_name>/mbedtls_examples
LittleFS examples	<install_dir>/boards/<board_name>/littlefs_examples
Documentation	<install_dir>/docs
USB Documentation	<install_dir>/docs/usb
lwIP Documentation	<install_dir>/docs/lwip
Middleware	<install_dir>/middleware
mbed TLS	<install_dir>/middleware/mbedtls
LittleFS	<install_dir>/middleware/littlefs
lwIP stack	<install_dir>/middleware/lwip
Cypress Wiced WiFi SDK	<install_dir>/middleware/wifi_wiced
jsmn	<install_dir>/middleware/aws_iot/external_libs/jsmn

Table continues on the next page...

Table 2. Release contents (continued)

FatFS stack	<install_dir>/middleware/fatfs
USB stack	<install_dir>/middleware/usb
WolfSSL stack	<install_dir>/middleware/wolfssl
Driver, SoC header files, extension header files and feature header files, utilities	<install_dir>/devices/<device_name>
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
CMSIS drivers	<install_dir>/devices/<device_name>/cmsis_drivers
Utilities such as debug console	<install_dir>/devices/<device_name>/utilities
RTOS Kernel Code	<install_dir>/rtos
Tools	<install_dir>/tools
segger_systemview	<install_dir>/boards/<board>/rtos_examples/visualization/freertos_segger_sysview
percepio_snapshot	<install_dir>/boards/<board>/rtos_examples/visualization/freertos_percepio_snapshot
gradle	<install_dir>/boards/<board>/aws_examples/remote_control_android/gradle, boards/<board>/aws_examples/led_wifi_android/gradle, boards/<board>/aws_examples/device_configuration_android/gradle
FNET	<install_dir>/boards/<board>/aws_examples/device_configuration_enet/fnet_mdns

Chapter 6
MCUXpresso SDK release package

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

6.1 Device support

The device folder contains all available software enablement for the specific System-on-Chip (SoC) subfamily. This folder includes clock-specific implementation, device register header file, device register feature header file, 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 simple debug console.

The device-specific header files provide a direct access to the MCU peripheral registers. The device header file provides an overall 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 CMSIScompliant 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, RTOS, and middleware 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.

6.2 Middleware

6.2.1 USB stack

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

6.2.1.1 Peripheral devices tested with USB Host stack

This table provides a list of USB devices tested with the USB Host stack.

Table 3. Peripheral devices

Device type	Device

Table continues on the next page...

Table 3. Peripheral devices (continued)

USB HUB	BELKIN F5U233 BELKIN F5U304 BELKIN F5U307 BELKIN F4U040 UNITEK Y-2151 Z-TEK ZK032A HYUNDAI HY-HB608
USB flash drive	ADATA C008 32 GB ADATA S102 8 G ADATA S102 16 G Verbatim STORE N GO USB Device 8 G Kingston DataTraveler DT101 G2 SanDisk Cruzer Blade 8 GB Unisplendour 1 G Imation 2 GB V-mux 2 GB Sanmina-SCI 128 M Corporate Express 1 G TOSHIBA THUHYBS-008G 8 G Transcend JF700 8 G Netac U903 16 G SSK SFD205 8 GB Rex 4 GB SAMSUNG USB3.0 16GB
USB card reader/adapter	SSK TF adapter Kawau Multi Card Reader Kawau TF adapter Kawau SDHC card

Table continues on the next page...

Table 3. Peripheral devices (continued)

USB Mouse	DELL MS111-P DELL M066U0A DELL MUAVDEL8 TARGUS AMU76AP DELL MD56U0 DELL MS111-T RAPOO M110
USB Keyboard	DELL SK8135 DELL SK8115

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

6.2.3 Cypress Wiced WiFi SDK

The MCUXpresso SDK provides integration with Cypress Wiced SDK supporting the Murata Type 1DX module based on the CYW4343W processor.

The solution is based on Embedded Artists Type 1DX M.2 EVB, which provides IEEE 802.11/b/g/n connectivity. IDX M.2 EVB is connected to the RT1050 EVK Micro SD card slot using Murata uSD-M.2 Adapter.

6.2.4 File system

The FatFs file system is integrated with 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.

6.2.5 RTOS

The MCUXpresso SDK is integrated with FreeRTOS OS.

6.2.6 CMSIS

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

6.2.7 emWin

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

6.2.8 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.

Chapter 7
MISRA compliance

All MCUXpresso SDK drivers and USB stack comply to MISRA 2012 rules with the following exceptions.

Table 4. 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	Typedef that indicate size and signedness should be used in place of the basic numerical type.
Directive 4.8	If a pointer to a structure or union is never dereferenced within a transaction unit then the implementation of the object should hidden.
Directive 4.9	A function should be used in preference to a function like macro where they are interchangeable.
Directive 4.10	Precautions shall be taken in order to prevent the contents of a header file being included more than once.
Directive 4.11	The validity of values passed to library functions shall be checked.
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 2.7	There should be no unused parameters in functions.
Rule 3.1	The character sequences /* and // shall not be used within a comment.
Rule 5.1	External identifiers shall distinct.
Rule 5.3	A identifier declared in an inner scope shall not hide an identifier declared in an outer scope.
Rule 5.7	A tag name shall be a unique identifier.
Rule 5.9	Identifiers that define objects or functions with external linkage shall be unique.
Rule 8.13	A pointer should point to a const-qualified type whenever possible.
Rule 8.3	All declarations of an object or function shall use the same names and type qualifiers.
Rule 8.6	An identifier with external linage shall have exactly one external definition.
Rule 8.7	Octal constants shall not be used.

Table continues on the next page...

Table 4. MISRA exceptions (continued)

Rule 8.9	A object should be defined at block scope if its identified only appears in a single function.
Rule 10.1	Operands shall not be of an inappropriate essential type.
Rule 10.3	The value of an expression shall not be assigned to an object with a narrower essential type of a different essential type category.
Rule 10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category.
Rule 10.5	The value of an expression should not be cast to an inappropriate essential type.
Rule 10.6	The value of a composite expression shall not be assigned to an object with wider essential type.
Rule 10.7	If a composite expression is used as one operand of an operator in which the usual arithmetic conversions are performed then the other operand shall not have wider essential type.
Rule 10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type.
Rule 11.1	Conversions shall not be performed between a pointer to a function and any other type.
Rule 11.3	A case shall not be performed between a pointer to object type and a pointer to a different object 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 11.6	A cast shall not be performed between pointer to void and an arithmetic type.
Rule 12.1	The precedence of operators within expressions should be made explicit.
Rule 12.2	The right hand operator of a shift operator shall lie in the range zero to one less than the width in bits of the essential type of the left hand operand.
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 13.5	The right hand operand of a logical && or || operator shall not contain persistent side effects.
Rule 14.2	A for loop shall be well formed.

Table continues on the next page...

Table 4. MISRA exceptions (continued)

Rule 14.4	The controlling expressions of an statement and the controlling expression of an iteration-statement shall have essentially Boolean type.
Rule 15.5	A function should have a single point of exit at the end.
Rule 16.1	All switch statements shall be well-formed.
Rule 17.1	The feature of <stdarg.h> shall not be used.
Rule 18.4	The +,-,+=and -=operators should not be applied to an expression of pointer type.
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.

Chapter 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.

8.2 USBFS controller issue

Because of the USBFS controller design issues, the USB host suspend/resume demos (usb_suspend_resume_host_hid_mouse) of the full speed controller do not support the low speed device directly.

8.3 Create new project without board template

The following components should be selected at the same time when creating a new project without using a board template, including serial_manager, serial_manager_uart, debug_console, and one UART adapter (lpuart_adapter for LPUART IP, uart_adapter for UART IP, lpsci_adapter for LPSCI IP, etc).

MCUXpresso SDK Release Notes Supporting EVK-MIMXRT1060

Change Logs

Contents

Contents

Contents

1 Driver Change Log

CLOCK

The current CLOCK driver version is 2.1.6.

- 2.1.6
  - Bug Fix:
    - Add initialization of the fractional mode and spread spectrum mode in CLOCK_InitSysPll().
- 2.1.5
  - New feature:
    - Add support for ENET2.
- 2.1.4
  - Optimization:
    - Add PerClk in clock_name_t and CLOCK_GetFreq.
    - Add APIs to get the frequency of AHB clock and SEMC, IPG clock and PER clock.
- 2.1.3
  - Use double instead of uint64_t to achieve better performance with double precision FPU.
- 2.1.2
  - some minor fixes.
- 2.0.0
  - initial version.

IOMUXC

The current IOMUXC driver version is 2.0.1.

- 2.0.1
  - Add missing enum value kIOMUXC_GPR_ENET2RefClkMode & kIOMUXC_GPR_ENET2TxClkOutputDir in the iomuxc_gpr_mode_t.
- 2.0.0
  - initial version.

LPI2C

Current LPI2C CMSIS driver version is 2.0

- 2.0
  - Initial version.

LPSPI_CMSIS

Current lpspi_cmsis driver version is 2.1

- 2.1
  - Bug Fix:
    - Fix the incorrect clock polarity assignment in the driver. For ARM_SPI_CPOL0_CPHA0 and other frame format parameters, CPOL = 0 means kSPI_ClockPolarityActiveHigh not kSPI_ClockPolarityActiveLow in driver.
  - New feature:
    - Allow user to set up the default Transmit value by using ARM_SPI_SET_DEFAULT_TX_VALUE. Please note that set default value is not supported in slave interrupts, because the pin will stay tristated if TX buffer is NULL.
    - Enable slave select mode in the new driver, but this has no effect when users set any of them because the driver can only support the hardware control function.
    - Enable 3-Wire mode, users can use ARM_SPI_MODE_MASTER_SIMPLEX/ARM_SPI_MODE_SLAVE_SIMPLEX to enable this feature. For ARM_SPI_MODE_MASTER_SIMPLEX mode, select SOUT pin as the input/output pin, and for ARM_SPI_MODE_SLAVE_SIMPLEX, the SIN pin is selected as the input/output pin.
- 2.0
  - - Initial version.

LPUART

Current LPUART CMSIS driver version is 2.0

- 2.0
  - Initial version.

ADC

The current ADC driver version is 2.0.1.

- 2.0.1
  - Add control macro to enable/disable the CLOCK code in current driver.
- 2.0.0
  - Initial version.

ADC_ETC

The current ADC_ETC driver version is 2.0.1.

- 2.0.1
  - Add control macro to enable/disable the CLOCK code in current driver.
- 2.0.0

  - Initial version.

AIPSTZ

The current AIPSTZ driver version is 2.0.0.

- 2.0.0
  - Initial version.

AOI

The current AOI driver version is 2.0.0.

- 2.0.0
  - Initial version.

BEE

The current BEE driver version is 2.0.1.

- 2.0.1
  - Bug fix:
    * Fixed bug in key user key loading sequence. BEE must be enabled during loading of user key.
    * Fixed typos in comments.
  - Feature:
    * Setting of AES nonce was moved from BEE_SetRegionKey() into separate BEE_SetRegionNonce() function.
    * Changed handling of region settings. Both regions are configured simultaneously by BEE_SetConfig() function. Configuration of FAC start and end address using IOMUXC_GPRs was moved to application.
    * Added configuration setting for endian swap, access permission and region security level.
    * Default value for region address offset was changed to 0.
- 2.0.0
  - Initial version.

CACHE

The current CACHE driver version is 2.0.1.

- 2.0.1
  - Fixed cache size issue in L2CACHE_GetDefaultConfig API.
- 2.0.0

  - Initial version.

CMP

The current CMP driver version is 2.0.0.

- 2.0.0
  - Initial version.

COMMON

The current COMMON driver version is 2.0.1.

- 2.0.1
  - Remove the implementation of LPC8XX Enable/DisableDeepSleepIRQ() function.
  - Add new feature macro switch "FSL_FEATURE_HAS_NO_NONCACHEABLE_SECTION" for specific SoC which has no noncacheable sections, this will help avoid unnecessary complex in link file and startup file.
  - Update the align(x) to **attribute**(aligned(x)) to support MDK v6 armclang compiler.
- 2.0.0
  - Initial version.

CSI

The current CSI driver version is 2.0.2.

- 2.0.2
  - Add fragment mode support.
- 2.0.1
  - Switch DMA output buffer at the first data after each VSYNC, originally it happened when the DMA transfer was done.
- 2.0.0
  - Initial version.

DCDC

The current DCDC driver version is 2.0.0.

- 2.0.0
  - Initial version.

DMAMUX

The current DMAMUX driver version is 2.0.2.

- 2.0.2
  - New feature:
    - Added an always-on enable feature to a DMA channel for ULP1 DMAMUX support.
- 2.0.1
  - Bug fix:
    - Fixed build warning while setting the DMA request source in DMAMUX_SetSource-Change issue by changing the type of the parameter source from uint8_t to uint32_t.
- 2.0.0
  - Initial version.

EDMA

The current eDMA driver version is 2.1.4.

- 2.1.4
  - Bug fix:
    - Clear enabled request, status during EDMA_Init for the case that EDMA is halted before reinitialization.
- 2.1.3
  - Bug fix:
    - Add clear DONE bit in IRQ handler to avoid overwrite TCD issue.
    - Optimize above solution for the case that transfer request occurs in callback.
- 2.1.2
  - Improvements:
    - Added interface to get next TCD address.
    - Added interface to get the unused TCD number.
- 2.1.1
  - Improvements:
    - Added documentation for eDMA data flow when scatter/gather is implemented for the EDMA_HandleIRQ API.
    - Updated and corrected some related comments in the EDMA_HandleIRQ API and edma-_handle_t struct.
- 2.1.0
  - Improvements:
    - Changed the EDMA_GetRemainingBytes API into EDMA_GetRemainingMajorLoop-Count due to eDMA IP limitation (see API comments/note for further details).
- 2.0.5
  - Improvements:
    - Added pubweak DriverIRQHandler for K32H844P (16 channels shared).
- 2.0.4
  - Improvements:

        * Added support for SoCs with multiple eDMA instances.
        * Added pubweak DriverIRQHandler for KL28T DMA1 and MCIMX7U5_M4.
- 2.0.3
  - Bug fix:
    * Fixed the incorrect pubweak IRQHandler name issue, which causes re-definition build errors when client sets his/her own IRQHandler, by changing the 32-channel IRQHandler name to DriverIRQHandler.
- 2.0.2
  - Bug fix:
    * Fixed incorrect minorLoopBytes type definition in _edma_transfer_config struct, and defined minorLoopBytes as uint32_t instead of uint16_t.
- 2.0.1
  - Bug fix:
    * Fixed the eDMA callback issue (which did not check valid status) in EDMA_HandleIRQ API.
- 2.0.0
  - Initial version.

ELCDIF

The current ELCDIF driver version is 2.0.0.

- 2.0.0
  - Initial version.

ENET

The current ENET driver version is 2.2.3.

- 2.2.3
  - Improved data buffer cache maintainence in the ENET driver.
- 2.2.2
  - Added the APIs for extended multi-ring support.
  - Added the AVB configure API for extended AVB feature support.
- 2.2.1
  - Changed the input data pointer attribute to const in ENET_SendFrame().
- 2.1.1
  - Added the extended MDIO IEEE802.3 Clause 45 MDIO format SMI command APIs.
  - Added the extended interrupt coalescing feature.
  - Combined all storage operations in the ENET_Init to ENET_SetHandler API.
- 2.0.1
  - Bug fix:
    * Used direct transmit busy check when doing data transmit.
  - Miscellaneous changes:

        - Updated IRQ handler work flow.
        - Changed the TX/RX interrupt macro from kENET_RxByteInterrupt to kENET_RxBuffer-Interrupt, from kENET_TxByteInterrupt to kENET_TxBufferInterrupt.
        - Deleted unnecessary parameters in ENET handler.
- 2.0.0
    - Initial version.

EWM

The current EWM driver version is 2.0.1.

- 2.0.1
    - Fixed EWM_Deinit hardfault issue.
- 2.0.0
    - Initial version.

FLEXCAN

The current FLEXCAN driver version is 2.3.2.

- 2.3.2
    - Improvements:
        - Implementation for ERR005959.
        - Implementation for ERR005829.
        - Implementation for ERR006032.
- 2.3.1
    - Bug fixes:
        - Adding correct handle when kStatus_FLEXCAN_TxSwitchToRx comming.
- 2.3.0
    - Improvements:
        - Added self wake support from STOP mode in the interrupt handling.
- 2.2.3
    - Bug fix:
        - Fixed CANFD data phase's bit rate not set as expected.
- 2.2.2
    - Improvements:
        - Added time stamp feature and enabled in interrupt_transfer example.
- 2.2.1
    - Improvements:
        - Separated CANFD initialization API.
        - In the interrupt handling, fixed issue that user cannot use normal CAN API when FD is present.
- 2.2.0
    - Improvements:

        * Added FSL_FEATURE_FLEXCAN_HAS_SUPPORT_ENGINE_CLK_SEL_REMOVE feature to support SoCs without CAN Engine Clock selection in FlexCAN module.
        * Added FlexCAN Serial Clock Operation to support i.MX SoCs.
- 2.1.0
  - Bug fixes:
    * Fixed wrong function name spelling: FLEXCAN_XXX() -> FLEXCAN_XXX();
    * Moved Freeze Enable/Disable setting from FLEXCAN_Enter/ExitFreezeMode() to FLEXCAN_Init();
    * Fixed wrong helper macro values.
  - Other changes:
    * Hided FLEXCAN_Reset() to user.
    * Used NDEBUG macro to wrap FLEXCAN_IsMbOccupied() function instead of DEBUG macro.
- 2.0.0
  - Initial version.

FLEXIO

The current FLEXIO driver version is 2.0.2.

- 2.0.2:
  - Improvements:
    * Split FlexIO component which combines all flexio/flexio_uart/flexio_i2c/flexio_i2s drivers into several components. FlexIO component, flexio_uart component, flexio_i2c_master component, and flexio_i2s component.
- 2.0.1
  - Bug fix:
    * Fix the Dozen mode configuration error in FLEXIO_Init API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.

FLEXIO_UART

The current FLEXIO_UART driver version is 2.1.5.

- 2.1.5
  - Trigger user callback when all data are in ringbuffer are received when calling FLEXIO_UART_TransferReceiveNonBlocking.
- 2.1.4
  - Unify component full name to FLEXIO UART(DMA/EDMA) Driver
- 2.1.3
  - Bug fixes: The following modifications support FlexIO using multiple instances.
    * Removed FLEXIO_Reset API in module Init APIs.
    * Updated module Deinit APIs to reset the shifter/timer config instead of disable module and disable clock.

        * Updated module Enable APIs to only support enable operation.
- 2.1.2
  - Bug fixes:
    * Fixed the transfer count calculation issue in FLEXIO_UART_TransferGetReceiveCount, FLEXIO_UART_TransferGetSendCount, FLEXIO_UART_TransferGetReceiveCount-DMA, FLEXIO_UART_TransferGetSendCountDMA, FLEXIO_UART_TransferGet-ReceiveCountEDMA and FLEXIO_UART_TransferGetSendCountEDMA
    * Fixed the Dozen mode configuration error in FLEXIO_UART_Init API. For enableIn-Doze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
    * Reported error when set baudrate too low and FLEXIO cannot reach that baudrate.
    * Disabled FLEXIO_UART receive interrupt instead of disable all NVIC when read data from ring buffer. Because ring buffer is used, receive nonblocking disables all NVIC interrupts to protect the ring buffer. This has negative effects on other IPS which are using interrupt.
- 2.1.1
  - Bug fixes:
    * Changed the API name FLEXIO_UART_StopRingBuffer to FLEXIO_UART_Transfer-StopRingBuffer to align with the definition in C file.
- 2.1.0
  - New features:
    * Added Transfer prefix in transactional APIs.
    * Added txSize/rxSize in handle structure to record the transfer size.
  - Bug fixes:
    * Added error handle to handle the data count is zero or data buffer is NULL situation.

FLEXIO_I2C

The current FLEXIO_I2C driver version is 2.1.6.

- 2.1.6
  - Bug fix:
    * Fixed the issue that I2C Master transfer APIs(blocking/non-blocking) does not support the situation that master transfer with subaddress and transfer data size zero, which means no data follows by the subaddress.
- 2.1.5
  - Unify component full name to FLEXIO I2C Driver
- 2.1.4
  - Bug fixes: The following modifications support FlexIO using multiple instances.
    * Removed FLEXIO_Reset API in module Init APIs.
    * Updated module Deinit APIs to reset the shifter/timer config instead of disable module and disable clock.
    * Updated module Enable APIs to only support enable operation.
- 2.1.3

  - Changed the prototype of FLEXIO_I2C_MasterInit to return kStatus_Success if initialization successfully and return kStatus_InvalidArgument if "(srcClock_Hz / masterConfig->baudRate_Bps) / 2 - 1" exceeds 0xFFU.
- 2.1.2
  - Fixed the FLEXIO I2C issue where the master cannot receive data from I2C slave in high baudrate.
  - Fixed the FLEXIO I2C issue where the master cannot receive NAK when master sends non-existent addr.
  - Fixed the FLEXIO I2C issue where the master cannot get transfer count successfully.
  - Fixed the FLEXIO I2C issue where the master cannot receive data successfully when sending data first.
  - Fixed the Dozen mode configuration error in FLEXIO_I2C_MasterInit API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
  - Fixed the FLEXIO_I2C_MasterTransferBlocking API calls FLEXIO_I2C_MasterTransferCreateHandle issue. This leads the s_flexioHandle/s_flexioIsr/s_flexioType variable written. Then, if calling FLEXIO_I2C_MasterTransferBlocking API multiple times, the s_flexioHandle/s_flexioIsr/s_flexioType variable cannot be written anymore due to it being out of range. This leads to the following: NonBlocking transfer APIs cannot work due to register IRQ failed.
- 2.1.1
  - Bug fixes:
    * Implemented the FLEXIO_I2C_MasterTransferBlocking API which defined in header file but has no implementation in the C file.
- 2.1.0
  - New features:
    * Added Transfer prefix in transactional APIs.
    * Added transferSize in handle structure to record the transfer size.

FLEXIO_SPI

The current FLEXIO_SPI driver version is 2.1.3.

- 2.1.3
  - Unify component full name to FLEXIO SPI(DMA/EDMA) Driver
- 2.1.2
  - Bug fixes: The following modification support FlexIO using multiple instances.
    * Removed FLEXIO_Reset API in module Init APIs.
    * Updated module Deinit APIs to reset the shifter/timer config instead of disable module and disable clock.
    * Updated module Enable APIs to only support enable operation.
- 2.1.1
  - Bug fixes:
    * Fixed bug where FLEXIO SPI transfer data is in 16 bit per frame mode with eDMA.
    * Fixed bug where FLEXIO SPI transfer data is in 16 bit per frame and direction is Lsbfirst

mode with eDMA and interrupt.
        - Fixed the Dozen mode configuration error in FLEXIO_SPI_MasterInit/FLEXIO_SPI_SlaveInit API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
    - Optimization:
        - Added #ifndef/#endif to allow user to change the default tx value at compile time.
- 2.1.0
    - New features:
        - Added Transfer prefix in transactional APIs.
        - Added transferSize in handle structure to record the transfer size.
    - Bug fixes:
        - Fixed the error register address return for 16-bit data write in FLEXIO_SPI_GetTxDataRegisterAddress.
        - Provided independent IRQHandler/transfer APIs for Master and slave to fix the baudrate limit issue.

FLEXIO_I2S

The current FLEXIO_I2S driver version is 2.1.6.

- 2.1.6
    - Bug fix:
        - Add reset flexio before flexio i2s init to make sure flexio status is normal.
- 2.1.5
    - Bug fix:
        - Fix i2s driver use hard code for bitwidth setting.
- 2.1.4
    - Unify component full name to FLEXIO I2S(DMA/EDMA) Driver
- 2.1.3
    - Bug fixes: The following modifications support FlexIO using multiple instances.
        - Removed FLEXIO_Reset API in module Init APIs.
        - Updated module Deinit APIs to reset the shifter/timer config instead of disable module and disable clock.
        - Updated module Enable APIs to only support enable operation.
- 2.1.2
    - New features:
        - Added configure items for all pin polarity and data valid polarity.
        - Added default configure for pin polarity and data valid polarity.
- 2.1.1
    - Bug fixes:
        - Fixed FlexIO I2S RX data read error and eDMA address error.
        - Fix FlexIO I2S slave timer compare setting error.
- 2.1.0
    - New features:

    * Added Transfer prefix in transactional APIs.
    * Added transferSize in handle structure to record the transfer size.

FLEXIO_MCU_LCD

The current FLEXIO_MCU_LCD driver version is 2.0.2.

- 2.0.2
  - Unify component full name to FLEXIO_MCU_LCD(EDMA) Driver
- 2.0.1
  - Bug fixes: The following modification to support FlexIO using multiple instances.
    * Removed FLEXIO_Reset API in module Init APIs.
    * Updated module Deinit APIs to reset the shifter/timer config instead of disable module and disable clock.
    * Updated module Enable APIs to only support enable operation.
- 2.0.0
  - Initial version.

FLEXIO_CAMERA

The current FLEXIO_CAMERA driver version is 2.1.2.

- 2.1.2
  - Unify component full name to FLEXIO CAMERA(EDMA) Driver
- 2.1.1
  - Bug fixes: The following modifications support FlexIO using multiple instances.
    * Removed FLEXIO_Reset API in module Init APIs.
    * Updated module Deinit APIs to reset the shifter/timer config instead of disable module and disable clock.
    * Updated module Enable APIs to only support enable operation.
- 2.1.0
  - New features:
    * Added Transfer prefix in transactional APIs.

FLEXRAM

The current FLEXRAM driver version is 2.0.4.

- 2.0.4
  - Fix FlexRAM driver missing extern C around functions in header file.
  - Remove magic address feature from driver.
- 2.0.3
  - Fix TCM size configuration is wrong when TCM bank number is not a value power of 2.
- 2.0.2

  - Updated driver due to Reference Manual update.
- 2.0.1
  - Fixed MISRA issue.
- 2.0.0
  - Initial version.

FLEXSPI

The current FLEXSPI driver version is 2.0.5.

- 2.0.5
  - Bug fixes:
    - Fix FLEXSPI_UpdateLUT cannot do partial update issue .
- 2.0.4
  - Bug fixes:
    - Reset flash size for all ports to zero in FLEXSPI_Init, fixed the possible out of range flash access with no error reported.
- 2.0.3
  - Bug fixes:
    - Fixed AHB receive buffer size configuration issue, the FLEXSPI_AHBRXBUFCR0_BUFSZ field should configure 64 bits size, and currently the the AHB receive buffer size is in byte which means 8-bit, so the correct configuration should be config->ahbConfig.buffer[i].bufferSize / 8.
- 2.0.2
  - New features:
    - Supports DQS write mask enable/disable feature during set FLEXSPI configuration.
    - Provides new API FLEXSPI_TransferUpdateSizeEDMA for user to update eDMA transfer size(SSIZE/DSIZE) per DMA transfer.
  - Bug fixes:
    - Fixed FLEXSPI_Init invalid operation to enable AHB bus Read Access to IP RX FIFO issue.
    - Fixed FLEXSPI_Init incorrect operation to configure IP TX FIFO watermark issue.
- 2.0.1
  - Bug fixes:
    - Fixed the flag clear issue and AHB read Command index configuration issue in FLEXSPI_SetFlashConfig.
    - Updated FLEXSPI_UpdateLUT function to update LUT table from any index instead of previous command index.
    - Added bus idle wait in FLEXSPI_SetFlashConfig and FLEXSPI_UpdateLUT to ensure bus is idle before any change to FlexSPI controller.
    - Updated interrupt API FLEXSPI_TransferNonBlocking and interrupt handle flow FLEXSPI_TransferHandleIRQ.
    - Updated eDMA API FLEXSPI_TransferEDMA.
- 2.0.0

  - Initial version.

GPC

The current GPC driver version is 2.1.0.

- 2.1.0
  - Updated driver for IMX6RT.
- 2.0.0
  - Initial version.

GPT

The current GPT driver version is 2.0.0.

- 2.0.0
  - Initial version.

GPIO

The current GPIO driver version is 2.0.1.

- 2.0.1:
  - API interface changes:
    * Refined naming of API while keeping all original APIs, marking them as deprecated. Original API will be removed in next release. The main change is update API with prefix of _PinXXX() and _PortXXX. main change is update API with prefix of _PinXXX() and _PortXXX().
- 2.0.0
  - Initial version.

KPP

The current KPP driver version is 2.0.0.

- 2.0.0
  - Initial version.

LPI2C

The current LPI2C driver version is 2.1.6.

- 2.1.6

  - Bug fix:
    - Fix driver MISRA build error and c++ build error in LPI2C_MasterSend and LPI2C_-SlaveSend.
- 2.1.5
  - Bug fix:
    - Extended the Driver IRQ handler to support LPI2C4 and change to use ARRAY_SIZE(k-Lpi2cBases) instead of FEATURE COUNT to decide the array size for handle pointer array.
    - 2.1.4
  - Bug fix:
    - Fixed the LPI2C_MasterTransferEDMA receive issue when LPI2C share same request source for TX/RX DMA request. In the previous way the API uses scatter gather method, handle command transfer first, then handles the linked TCD which preset with the receive data transfer. The issue is that TX DMA request and RX DMA request are both enabled, when DMA finished the first command TCD transfer and handled the receive data TCD, the TX DMA request still happens due to TX FIFO empty. This results the RX DM-A transfer starts, without waiting on the expected RX DMA request. Fix the issue by enabling IntMajor interrupt for the command TCD and checking if there is a linked TCD to disable the TX DMA request in LPI2C_MasterEDMACallback API.
- 2.1.3
  - Improvement:
    - Added LPI2C_WATI_TIMEOUT macro to allow the user to specify the timeout times for waiting flags in functional API and blocking transfer API.
    - Added LPI2C_MasterTransferBlocking API.
- 2.1.2
  - Bug fix:
    - In LPI2C_SlaveTransferHandleIRQ, reset the slave status to idle when stop flag is detected.
- 2.1.1
  - Bug fix:
    - Disabled auto stop feature in eDMA driver. Previously, the autostop feature was enabled at transfer when transferring with stop flag. If the previous transfer was without stop flag, because the auto stop feature is enabled, then when starting a new transfer with stop flag, the stop flag sends before starting the new transfer, and the start flag cannot successfully send, so the transfer can not start.
    - Changed default slave configuration with address stall false.
- 2.1.0
  - API name change:
    - LPI2C_MasterTransferCreateHandle -> LPI2C_MasterCreateHandle.
    - LPI2C_MasterTransferGetCount -> LPI2C_MasterGetTransferCount.
    - LPI2C_MasterTransferAbort -> LPI2C_MasterAbortTransfer.
    - LPI2C_MasterTransferHandleIRQ -> LPI2C_MasterHandleInterrupt.
    - LPI2C_SlaveTransferCreateHandle -> LPI2C_SlaveCreateHandle.
    - LPI2C_SlaveTransferGetCount -> LPI2C_SlaveGetTransferCount.
    - LPI2C_SlaveTransferAbort -> LPI2C_SlaveAbortTransfer.

        * LPI2C_SlaveTransferHandleIRQ -> LPI2C_SlaveHandleInterrupt.
- 2.0.0
  - Initial version.

LPSPI

The current LPSPI driver version is 2.0.3.

- 2.0.3
  - Bug Fix:
    * Remove the LPSPI_Reset() from LPSPI_MasterInit() and LPSPI_SlaveInit(), because this API may glitch the slave select line, if needed, please call this function manually.
- 2.0.2
  - New feature:
    * Added dummy data setup API to allow users to configure the dummy data to be transferred.
    * Enabled the 3-wire mode, SIN and SOUT pins can be configured as input/output pin.
- 2.0.1
  - Bug fix:
    * The clock source should divided by PRESCALE setting in LPSPI_MasterSetDelayTimes function.
    * Fixed the bug that LPSPI_MasterTransferBlocking function would hang in some corner cases.
  - Optimization:
    * Added #ifndef/#endif to allow user to change the default TX value at compile time.
- 2.0.0
  - Initial version.

LPUART

The current LPUART driver version is 2.2.6.

- 2.2.6
  - Fix the repeatly reading status register issue while dealing with the IRQ routine.
- 2.2.5
  - Do not set or clear the TIE/RIE bits when using LPUART_EnableTxDMA() and LPUART_EnableRxDMA().
- 2.2.4
  - Added hardware flow control function support.
  - Added idle line detected feature in LPUART_TransferNonBlocking function. If an idle line was detected, a callback is triggered with status kStatus_LPUART_IdleLineDetected returned. This feature may be useful when the received Bytes is less than the expected receive data size. Before triggering the callback, data in the FIFO (if has FIFO) is read out, and all interrupts will not be disabled, except if the receive data size reaches 0.

  - Enabled the RX FIFO watermark function. With the idle line detected feature enabled, you can set the watermark value to whatever you want (should be less than the RX FIFO size). Data is received and a callback is triggered when data receive is end.
- 2.2.3
  - Changed parameter type in LPUART_RTOS_Init() struct rtos_lpuart_config –> lpuart_rtos_config_t.
  - Bug fix:
    * Disabled LPUART receive interrupt instead of disabling all NVIC when read data from ring buffer. Because the ring buffer is used, receive nonblocking disables all NVIC interrupts to protect the ring buffer. This has a negative effect to other IPS which are using the interrupt.
- 2.2.2
  - Added software reset feature support.
  - Added software reset API to LPUART_Init().
- 2.2.1
  - Added separate RX,TX IRQ number support.
- 2.2.0
  - Added 7 data bits and MSB support.
- 2.1.1
  - Removed needless check of event flags and assert in LPUART_RTOS_Receive.
  - Always wait for RX event flag in LPUART_RTOS_Receive.
- 2.1.0
  - Update transactional APIs.

PIT

The current PIT driver version is 2.0.1.

- 2.0.1
  - Bug Fix:
    * Clear timer enable bit for all channels in function PIT_Init() to make sure all channels stay in disable status before setting other configurations.
- 2.0.0
  - Initial version.

PMU

The current PMU driver version is 2.1.0.

- 2.1.0
  - Added feature macros for low power control APIs to support to conditional compile.
  - Renamed "PMU_2P1EnablePullDown" to "PMU_2P5EnablePullDown"
- 2.0.0
  - Initial version.

PWM

The current PWM driver version is 2.0.0.

- 2.0.0
  - Initial version.

PXP

The current PXP driver version is 2.0.1.

- 2.0.1
  - Fix the rotate function issue for iMX6ULL.
- 2.0.0
  - Initial version.

QTMR

The current QTMR driver version is 2.0.0.

- 2.0.0
  - Initial version.

RTWDOG

The current RTWDOG driver version is 2.0.1.

- 2.0.1
  - Fix the bug in the RTWDOG_Init, add the check of register's unlock status when configure the RTWDOG in RTWDOG_init.
- 2.0.0
  - Initial version.

SAI

The current SAI driver version is 2.1.7.

- 2.1.7
  - Improvement:
    - Add feature macro test for the mclkSource member in sai_config_t.
  - Bug fix:
    - Fix the build error caused by feature macro test for mclkSource.

-2.1.6

- Improvement:

  - Add feature macro test for mclkSourceClockHz check.
  - Add bit clock source name for general devices.
- Bug fix:
  - Fix incorrect channel numbers setting while call RX/TX set format together.

-2.1.5

- Bug fix:
  - Correct SAI3 driver IRQ handler name.
  - Add I2S4/5/6 IRQ handler.
  - Add base in handler structure to support different instances share one IRQ number.
- New feature:
  - Update sai driver for MCR bit MICS.
  - Added 192KHZ/384KHZ in the sample rate enumeration.
  - Added multi fifo interrupt/sdma transfer support for TX/RX.
  - Added API to read/write multi fifo data in a blocking method.
  - Added bclk bypass support when bclk is same with mclk.

2.1.4

- New feature:
  - Added API to enable/disable auto FIFO error recovery in platforms that support this feature.
  - Added API to set data packing feature in platform which support this feature.

2.1.3

- New feature:
  - Added feature to make I2S frame sync length configurable according to bitWidth.

2.1.2

- Bug fix:
  - Added 24-bit support for SAI eDMA transfer. All data shall be 32 bits for send/receive, as eDMA cannot directly handle 3 Byte transfer.

2.1.1

- Optimization:
  - Reduceed code size while not using transactional API.

2.1.0

- API name change:
  - SAI_GetSendRemainingBytes -> SAI_GetSentCount.
  - SAI_GetReceiveRemainingBytes -> SAI_GetReceivedCount.
  - All transcational API name add "Transfer" prefix.
  - All transactional API use base and handle as input parameter.
  - Unify the parameter names.
- Bug fix:
  - Fixed WLC bug while reading TCSR/RCSR registers.
  - Fixd MOE enable flow issue, move MOE enable after MICS settings in SAI_TxInit/SAI_RxInit.

2.0.0

- Initial version.

SEMC

The current SEMC driver version is 2.0.4.

- 2.0.4
  - Fix the SEMC queueA and queueB weight configuration issue.
  - Fix the wrong configuration of DBICR1 register in SEMC_ConfigureDBI.
- 2.0.3
  - Add feature macro to control WDS&WDH bit setting for NOR synchronous transfer.
- 2.0.2
  - Changed SEMC NAND configuration structure and verify SEMC NAND related APIs.
  - Added extended SEMC clock enable
- 2.0.1
  - Fixed data size mask configure in SEMC_ConfigureIPCommand API.
  - Updated the command mode in IP command type.
- 2.0.0
  - Initial version.

SPDIF

The current SPDIF driver version is 2.0.1.

- 2.0.1
  - Correct the feature macro name used to define s_edmaPrivateHandle.
- 2.0.0
  - Initial version.

SRC

The current SRC driver version is 2.0.0.

- 2.0.0
  - Initial version.

TEMPMON

The current TEMPMON driver version is 2.0.0.

- 2.0.0
  - Initial version.

TSC

The current TSC driver version is 2.0.1.

- 2.0.1
  - Add control macro to enable/disable the CLOCK code in current driver.
- 2.0.0
  - Initial version.
  - This module was first developed on I.MX 6ULL.

USDHC

The current USDHC driver version is 2.2.7.

- 2.2.7
  - Add api USDHC_GetEnabledInterruptStatusFlags and use it in USDHC_TransferHandleIRQ.
  - Remove useless member interruptFlags in usdhc_handle_t.
- 2.2.6
  - Add address align check for ADMA descriptor table address.
  - Change USDHC_ADMA1_DESCRIPTOR_MAX_LENGTH_PER_ENTRY to (65536-4096) to make sure the data address is 4KB align for a transfer need more than one ADMA1 descriptor.
- 2.2.5
  - Fix MDK 66-D warning.
- 2.2.4
  - Fix issue that real clock frequency is mismatch with target clock frequency, which is caused by an incorrect prescaler caculation.
  - Add control macro to enable/disable the CLOCK code in current driver.
- 2.2.3
  - Fixed issue where AMDA did not disable with DMAEN clear.
  - Improved set clock function to check the output frequency range.
  - Dynamic set SDCLKFS during DDR enable or disable.
- 2.2.2
  - Improved read transfer cache maintain operation, combined clean and invalidated into one function.
- 2.2.1
  - Disabled the invalidate cache operation for tuning.
- 2.2.0
  - Improved USDHC to support mmc boot feature.
- 2.1.3
  - Fixed MISRA issue.
- 2.1.2
  - Fixed Coverity issue.
  - Added base address and userData parameter for all callback functions.
- 2.1.1

  - Added cache maintain operation.
  - Added timeout status check for the DATA transfer which ignore error.
  - Added feature macro for SDR50/SDR104 mode.
  - Removed useless IRQ handler for different platform.
- 2.1.0
  - Integrated tuning into transfer function.
  - Added strobe DLL feature.
  - Added enableAutoCommand23 in data structure.
  - Removed enable card clock function because the controller will handle the clock on/off.
- 2.0.0
  - Initial version.

WDOG

The current WDOG driver version is 2.1.0.

- 2.1.0
  - New Feature:
    * Add new API "WDOG_TriggerSystemSoftwareReset()" to allow user reset the system by software.
    * Add new API "WDOG_TriggerSoftwareSignal()" to allow user trigger a WDOG_B signal by software.
    * Remove the parameter "softwareAssertion" and "softwareResetSignal" out of the wdog_config_t structure.
    * Add new parameter "enableTimeOutAssert" to the wdog_config_t structure, with this parameter enabled, while the WDOG timeout occurred, a WDOG_B signal will be asserted, this signal can be routed to external pin of the chip. Please note that, WDOG_B signal remains asserted until a power-on reset (POR) occurs.
- 2.0.1
  - Add control macro to enable/disable the CLOCK code in current driver.
- 2.0.0
  - Initial version.

XBARA

The current XBARA driver version is 2.0.3.

- 2.0.3
  - Bug fixes:
    * Corrected configuration for function XBAR_SetOutputSignalConfig.
- 2.0.2
  - Other changes:
    * Changed array clock name.
- 2.0.1

  - Bug fixes:
    - Fixed w1c bits for XBARA_SetOutputSignalConfig function.
- 2.0.0
  - Initial version.

XBARB

The current XBARB driver version is 2.0.1.

- 2.0.1
  - Bug fixes:
    - Corrected XBARB_SetSignalsConnection function.
  - Other changes:
    - Changed array clock name.
- 2.0.0
  - Initial version.

2 Middleware Change Log

emWin library

The currently supported version is 5.48.

FatFs for MCUXpresso SDK

Current version is FatFs R0.13b_rev0.

- R0.13b_rev0
  - Upgraded to version 0.13b
- R0.13a_rev0
  - Upgraded to version 0.13a. Added patch ff_13a_p1.diff.
- R0.12c_rev1
  - Add NAND disk support.
- R0.12c_rev0
  - Upgraded to version 0.12c and applied patches ff_12c_p1.diff and ff_12c_p2.diff.
- R0.12b_rev0
  - Upgraded to version 0.12b.
- R0.11a
  - Added glue functions for low-level drivers (SDHC, SDSPI, RAM, MMC). Modified diskio.c.
  - Added RTOS wrappers to make FatFs thread safe. Modified syscall.c.
  - Renamed ffconf.h to ffconf_template.h. Each application should contain its own ffconf.h.
  - Included ffconf.h into diskio.c to enable the selection of physical disk from ffconf.h by macro definition.
  - Conditional compilation of physical disk interfaces in diskio.c.

lwIP for MCUXpresso SDK

The current version of lwIP is based on lwIP 2.1.2 and lwIP-contrib 2.1.0.

- 2.1.2_rev1
  - New features:
    - Ported lwIP 2.1.2 (2018-11-22, SHA-1: 159e31b689577dbf69cf0683bbaffbd71fa5ee10) to KSDK 2.0.0.
    - Ported lwIP-contrib 2.1.0 (2018-09-24, SHA-1: 35b011d4cf4c4b480f8859c456587a884ec9d287) to KSDK 2.0.0.
- 2.0.3_rev1
  - New features:
    - Ported lwIP 2.0.3 (2017-09-15, SHA-1: 92f23d6ca0971a32f2085b9480e738d34174417b) to KSDK 2.0.0.
- 2.0.2_rev1

  - New features:
    - Ported lwIP 2.0.2 (2017-03-13, SHA-1: c0862d60746e2d1ceae69af4c6f24e469570ecef) to KSDK 2.0.0.
- 2.0.0_rev3
  - New features:
    - Ported lwIP 2.0.0 (2016-11-10, SHA-1: 216bf89491815029aa15463a18744afa04df58fe) to KSDK 2.0.0.
- 2.0.0_rev2
  - New features:
    - Ported lwIP 2.0.0 RC2 (2016-08-08, SHA-1: b1dfd00f9233d124514a36a8c8606990016f2ad4) to KSDK 2.0.0.
- 2.0.0_rev1
  - New features:
    - Ported lwIP 2.0.0 RC0 (2016-05-26) to KSDK 2.0.0.
    - Changed lwIP bare-metal examples to use poll-driven approach instead of interrupt-driven one.
- 1.4.1_rev2
  - New features:
    - Enabled critical sections in lwIP.
  - Bug fixes:
    - Fixed default lwIP packet-buffer size to be able to accept a maximum size frame from the ENET driver.
    - Fixed possible drop of multi-frame packets during transmission.
- 1.4.1_rev1
  - New features:
    - Ported lwIP 1.4.1 to KSDK 2.0.0.

mbedTLS for MCUXpresso SDK

The current version of mbedTLS is based on mbedTLS 2.13.1 released 2018-09-06

- 2.13.1_rev2
  - Bug fixes:
    - Disable default HW acceleration of SHA in parallel with AES.
- 2.13.1_rev1
  - Bug fixes:
    - Fixed incorrect macro check when skipping AES-192 or AES-256
- 2.13.1
  - New features:
    - Ported mbedTLS 2.13.1 to KSDK.
- 2.12.0_rev1
  - New features:
    - Added support for NIST P-256 elliptic curve with CASPER driver.
- 2.12.0

  - New features:
    - Ported mbedTLS 2.12.0 to KSDK.
- 2.9.0_rev2
  - New features:
    - Added support for Hashcrypt driver.
- 2.9.0_rev1
  - New features:
    - Added support for CASPER driver.
- 2.9.0
  - New features:
    - Ported mbedTLS 2.9.0 to KSDK.
- 2.6.0_rev2
  - Bug fixes:
    - ssl_cookie.c now uses SHA256 for COOKIE_MD (instead of original SHA224). Some hw crypto acceleration (such as CAU3) don't support SHA224 but all support SHA256.
- 2.6.0_rev1
  - Bug fixes:
    - ksdk_mbedtls.c bignum functions now read sign of input mbedtls_mpi at beginning of functions to properly support in place computations (when output bignum is the same as one of input bignums). Affected functions: mbedtls_mpi_mul_mpi(), mbedtls_mpi_mod_mpi(), ecp_mul_comb().
- 2.6.0
  - New features:
    - Ported mbedTLS 2.6.0 to KSDK.
    - Added MBEDTLS_FREESCALE_FREERTOS_CALLOC_ALT to allow alternate implementation of pvPortCalloc() when using .c.
- 2.5.1_rev1
  - New features:
    - Added support for DCP driver.
- 2.5.1
  - New features:
    - Ported mbedTLS 2.5.1 to KSDK.
- 2.4.2_rev2
  - New features:
    - Added Curve25519 support for CAU3.
    - Added MBEDTLS_ECP_MUL_MXZ_ALT configuration parameter enabling overloading of ecp_mul_mxz().
- 2.4.2_rev1
  - New features:
    - Added support for CAU3 driver.
    - Added new files:
    - .c - contains regular software implementation of DES algorithm with added MBEDTLS_DES3_SETKEY_DEC_ALT and MBEDTLS_DES3_SETKEY_ENC_ALT config parameters.
    - .h - contains modified mbedtls_des_context and mbedtls_des3_context structures.

        - Added MBEDTLS_DES3_SETKEY_DEC_ALT configuration parameter enabling reloading of mbedtls_des3_set2key_dec() and mbedtls_des3_set3key_dec().
        - Added MBEDTLS_DES3_SETKEY_ENC_ALT configuration parameter enabling reloading of mbedtls_des3_set2key_enc() and mbedtls_des3_set3key_enc().
- 2.4.2
  - New features:
    - Ported mbedTLS 2.4.2 to KSDK 2.0.0.
    - Added CRYPTO_InitHardware() function.
    - Added new file:
      - .h - contains declaration of CRYPTO_InitHardware() function and should be included in applications.
- 2.3.0_rev1
  - New features:
    - Added support for CAAM driver.
    - In LTC-specific wrapper, allocate temporary integers from heap in one large block.
- 2.3.0
  - New features:
    - Ported mbedTLS 2.3.0 to KSDK 2.0.0.

2.2.1

- New features:
  - Ported mbedTLS 2.2.1 to KSDK 2.0.0.
  - Added support of MMCAU cryptographic acceleration module. Accelerated MD5, SHA, AES, and DES.
  - Added support of LTC cryptographic acceleration module. Accelerated AES, DES, and PKHA.
  - Added new files:
  - .c - alternative implementation of cryptographic algorithm functions using LTC and MMCAU module drivers.
  - .h - configuration settings used by mbedTLS KSDK bare metal examples.
  - Added mbedTLS KSDK bare-metal examples:
    - <board name> - KSDK mbedTLS benchmark application.
    - <board name> - KSDK mbedTLS self-test application.
  - Added MBEDTLS_GCM_CRYPT_ALT configuration parameter enabling reloading of mbedtls_gcm_crypt_and_tag().
  - Added MBEDTLS_ECP_MUL_COMB_ALT to enable alternate implementation of ecp_mul_comb().
  - Added MBEDTLS_ECP_ADD_ALT configuration parameter enabling reloading of ecp_add().
  - Added MBEDTLS_DES_SETKEY_DEC_ALT configuration parameter enabling reloading of mbedtls_des_setkey_dec(), mbedtls_des3_set2key_dec() and mbedtls_des3_set3key_dec().
  - Added MBEDTLS_DES_SETKEY_ENC_ALT configuration parameter enabling reloading of mbedtls_des_setkey_enc(), mbedtls_des3_set2key_enc() and mbedtls_des3_set3key_enc().
  - Added MBEDTLS_DES_CRYPT_CBC_ALT configuration parameter enabling reloading of mbedtls_des_crypt_cbc().

  - Added MBEDTLS_DES3_CRYPT_CBC_ALT configuration parameter enabling reloading of mbedtls_des3_crypt_cbc().
  - Added MBEDTLS_AES_CRYPT_CBC_ALT configuration parameter enabling reloading of mbedtls_aes_crypt_cbc().
  - Added MBEDTLS_AES_CRYPT_CTR_ALT configuration parameter enabling reloading of mbedtls_aes_crypt_ctr().
  - Added MBEDTLS_CCM_CRYPT_ALT configuration parameter enabling reloading of mbedtls_ccm_encrypt_and_tag() and mbedtls_ccm_auth_decrypt().
  - Added MBEDTLS_MPI_ADD_ABS_ALT configuration parameter enabling reloading of mbedtls_mpi_add_abs().
  - Added MBEDTLS_MPI_SUB_ABS_ALT configuration parameter enabling reloading of mbedtls_mpi_sub_abs().
  - Added MBEDTLS_MPI_EXP_MOD_ALT configuration parameter enabling reloading of mbedtls_mpi_exp_mod().
  - Added MBEDTLS_MPI_MUL_MPI_ALT configuration parameter enabling reloading of mbedtls_mpi_mul_mpi().
  - Added MBEDTLS_MPI_MOD_MPI_ALT configuration parameter enabling reloading of mbedtls_mpi_mod_mpi().
  - Added MBEDTLS_MPI_GCD_ALT configuration parameter enabling reloading of mbedtls-_mpi_gcd().
  - Added MBEDTLS_MPI_INV_MOD_ALT configuration parameter enabling reloading of mbedtls_mpi_inv_mod().
  - Added MBEDTLS_MPI_IS_PRIME_ALT configuration parameter enabling reloading of mbedtls_mpi_is_prime().
  - Added encrypt/decrypt mode to mbedtls_des_context and mbedtls_des3_context structure.
  - Added carriage return ” for mbedtls_printf() in self test functions.

SDMMC

The current driver version is 2.2.11.

- 2.2.11
  - BugFix
    - Fix NULL pointer dereference issue when calling function SDMMCHOST_CardDetect-Init in host adaptor layer.
- 2.2.10
  - BugFix:
    - Add NUll pointer check for USDHC freertos adaptor transfer complete callback.
    - Add event value check for all the freertos event to fix program hang when a card event occur before create.
- 2.2.9
  - Improvement:
    - Add NULL pointer check for sdmmchostcard_usr_param_t member cd in card detect callback to avoid memory corruption.

        * Add card voltage switch function in sdmmhostcard_usr_param_t to allow application register card signal line voltage switch function.
    - Bug fix
        * Fix host freertos adaptor and polling adaptor can't detect card insert bug for usdhc.
        * Fix sdhc host layer build issue and typo issue.
- 2.2.8
    - Improvement:
        * Update sdmmc to support sdio interrupt.
- 2.2.7
    - BugFix:
        * Fix MDK 66-D warning.
- 2.2.6
    - Improvement:
        * Remove some soc specific header files from porting layer.
        * Save MMC OCR registers while sending CMD1 with argument 0.
    - Bugfix:
        * Add MMC_PowerOn function in which there is delay function after powerup sdcard.-otherwise,card may init failed.
- 2.2.5
    - New features:
        * Add SD_ReadStatus api to get 512bit SD status.
        * Add error log support in sdcard functions.
        * Add SDMMC_ENABLE_SOFTWARE_TUNING to enable/disable software tuning and it is disabled by default.
        * Add error procedure in the transfer function to improve stability.
        * Remove deprecated gpio api in host layer.
- 2.2.4
    - Bug fix:
        * Fixed DDR mode data sequence miss issue, which is caused by NIBBLE_POS.
    - New features:
        * Increased g_sdmmc 512byte to improve the performance when application use a non-word align data buffer address.
        * Used OCR access mode bits to determine the mmccard high capacity flag.
        * Enabled auto cmd12 for SD read/write.
        * Disabled DDR mode frequency multiply by 2.
- 2.2.3
    - Bug fix:
        * Added response check for send operation condition command. If not checked, the card may occasionally init fail.
- 2.2.2
    - Moved set card detect priority operation before enable IRQ.
- 2.2.1
    - New features:
        * Improved MMC Boot feature.
        * Keep SD_Init/SDIO_Init function for forward compatibility.

- 2.2.0
  - New features:
    - Separated the SD/MMC/SDIO init API to xxx_CardInit/xxx_HostInit.
    - Allowed user register card detect callback, select card detect type, and determine the card detect timeout value.
    - Allowed user register the power on/off function, and determine the power on/off delay time.
    - SD_Init/SDIO_Init will be deprecated in the next version.
    - Added write complete wait operation for MMC_Write to fix command timeout issue.
- 2.1.6
  - Enhanced SD IO default driver strength.
- 2.1.5
  - Fixed coverity issue.
  - Fixed SD v1.x card write fail issue. It was caused by the block length set error.
- 2.1.4
  - Miscellaneous:
    - Added Host reset function for card re-initialization.
    - Added Host_ErrorRecovery function for host error recovery procedure.
    - Added cache maintain operation
    - Added HOST_CARD_INSERT_CD_LEVEL to improve compatibility.
  - Bug fix:
    - Fixed card cannot detect dynamically.
- 2.1.3
  - Bug fix:
    - Non high-speed sdcard init fail at switch to high speed.
  - Miscellaneous:
    - Optimized tuning/mmc switch voltage/mmc select power class/mmc select timing function.
    - Added strobe dll for mmc HS400 mode.
    - Added Delay for SDCard power up.
- 2.1.2
  - New features:
    - Added fsl_host.h to provide prototype to adapt different controller IPs(SDHC/SDIF).
    - Added adaptor code in SDMMC/Port folder to adapt different host controller IPs with different. transfer modes(interrupt/polling/freertos). Application includes a different adaptor code to make application more simple.
    - Adaptor code provides HOST_Init/HOST_Deinit/CardInsertDetect. APIs to do host controller initialize and transfer function configuration. SDMMC card stack uses adaptor code inside stack to wait card insert and configure host when calling card init APIs (SD_Init/MMC_Init/SDIO_Init).
    - This change requires the user to include host adaptor code into the application. If not changed, link errors saying it cannot find the definition of HOST_Init/HOST_Deinit/CardInsertDetect appear.
  - New features: Improved SDMMC to support SD v3.0 and eMMC v5.0.
  - Bug fix:

        * Fixed incorrect comparison between count and length in MMC_ReadBlocks/MMC_-WriteBlocks.
- 2.1.1
  - Bug fix:
    * Fixed the block range boundary error when transferring data to MMC card.
    * Fixed the bit mask error in the SD card switch to high speed function.
  - Other changes:
    * Added error code to indicate that SDHC ADMA1 transfer type is not supported yet.
    * Optimized the SD card initialization function.
- 2.1.0
  - Bug fix:
    * Change the callback mechanism when sending a command.
    * Fix the performance low issue when transferring data.
  - Other changes:
    * Changed the name of some error codes returned by internal function.
    * Merged all host related attributes to one structure.
    * Optimize the function of setting maximum data bus width for MMC card.

SDIO

The current driver version is 2.2.11.

- 2.2.11
  - Buf fix:
    * Add check card async interrupt capability in function SDIO_GetCardCapability.
- 2.2.10
  - Bug fix:
    * Fix sdio card driver get wrong io number when the card io number is bigger than 2.
  - New feature:
    * Add sdio 3.0 support.
    * Add api SDIO_IO_RW_Direct for direct read/write card register access.
- 2.2.9
  - Improvement:
    * Add api SDIO_SetIOIRQHandler/SDIO_HandlePendingIOInterrupt to handle multi io pending IRQ.
- 2.2.8
  - Improvement:
    * Update sdmmc to support sdio interrupt.
    * Add api SDIO_GetPendingInterrupt to get the pending io interrupt.
- 2.2.7
  - Bug fix:
    * Fix MDK 66-D warning.
- 2.2.6
  - New features:

        * Add a unify transfer interface for SDIO.
    - Bug fix:
        * Wrong pointer address used by SDMMCHOST_Init.
- 2.1.5
    - Bug fix:
        * Improved SDIO card init sequence and add retry option for SDIO_SwitchToHighSpeed function.
- 2.1.4
    - Miscellaneous:
        * Added Go_Idle function for SDIO card.
- 2.0.0
    - Initial version.

SDSPI

The current driver version is 2.1.4.

- 2.1.4
    - Bug fix:
        * Fix MDK 66-D warning.
- 2.1.3
    - Improve sdspi code size and performance.
- 2.0.0
    - Initial version.

USB stack for MCUXpresso SDK

The current version of USB stack is 2.1.0.

- 2.1.0
    - New features:
        * add host rndis support. example: lwip_dhcp_usb
        * enable usb3.0 support on device stack.
        * pd feature Add OM13790HOST support; Add auto policy feature; Print e-marked cable information;
- 2.0.1
    - Bug fix:
        * fixed some USB issues: fix msc cv test failed in msc examples.
        * Change the audio codec interfaces.
- 2.0.0
    - New features:
        * PTN5110N support.
    - Bug fix:
        * Added some comments, fixed some minor USB issues.

- 1.9.0
  - New features:
    - Examples:
      - usb_pd_alt_mode_dp_host
- 1.8.2
  - Updated license.
- 1.8.1
  - Bug fix:
    - Verified some hardware issues, support aruba_flashless.
- 1.8.0
  - New features:
    - Examples:
      - usb_device_composite_cdc_vcom_cdc_vcom
      - usb_device_composite_hid_audio_unified
      - usb_pd_sink_battery
      - Changed usb_pd_battery to usb_pd_charger_battery.

```
- Bug fix:
  - Code clean up, removed some irrelevant code.
```

- 1.7.0
  - New features:
    - USB PD stack support.
  - Examples
    - usb_pd
    - usb_pd_battery
    - usb_pd_source_charger
- 1.6.3
  - Bug fix: -IP3511_HS driver control transfer sequence issue, enabled 3511 ip cv test.
- 1.6.2
  - New features:
    - Multi instance support.
- 1.6.1
  - New features:
  - Changed the struct variable address method for device_video_virtual_camera and host_phdc-_manager.
- 1.6.0
  - New features:
    - Supported Device Charger Detect feature on usb_device_hid_mouse.
- 1.5.0
  - New features:
    - Supported controllers
      - OHCI (Full Speed, Host mode)
      - IP3516 (High Speed, Host mode)
      - IP3511 (High Speed, Device mode)

        - Examples:
            - usb_lpm_device_hid_mouse
            - usb_lpm_device_hid_mouse_lite
            - usb_lpm_host_hid_mouse
- 1.4.0
  - New features:
    - Examples:
      - usb_device_hid_mouse/freertos_static
      - usb_suspend_resume_device_hid_mouse_lite
- 1.3.0
  - New features:
    - Supported roles
      - OTG
    - Supported classes
      - CDC RNDIS
    - Examples
      - usb_otg_hid_mouse
      - usb_device_cdc_vnic
      - usb_suspend_resume_device_hid_mouse
      - usb_suspend_resume_host_hid_mouse
- 1.2.0
  - New features:
    - Supported controllers
      - LPC IP3511 (Full Speed, Device mode)
- 1.1.0
  - Bug fix:
    - Fixed some issues in USB certification.
    - Changed VID and Manufacturer string to NXP.
  - New features:
    - Supported classes
      - Pinter
    - Examples:
      - usb_device_composite_cdc_msc_sdcard
      - usb_device_printer_virtual_plain_text
      - usb_host_printer_plain_text
- 1.0.1
  - Bug fix:
    - Improved the efficiency of device audio speaker by changing the transfer mode from interrupt to DMA, thus providing the ability to eliminate the periodic noise.
- 1.0.0
  - New features:
    - Supported roles
      - Device
      - Host
    - Supported controllers:

        - KHCI (Full Speed)
        - EHCI (High Speed)
    - Supported classes:
        - AUDIO
        - CCID
        - CDC
        - HID
        - MSC
        - PHDC
        - VIDEO
    - Examples:
        - usb_device_audio_generator
        - usb_device_audio_speaker
        - usb_device_ccid_smart_card
        - usb_device_cdc_vcom
        - usb_device_cdc_vnic
        - usb_device_composite_cdc_msc
        - usb_device_composite_hid_audio
        - usb_device_composite_hid_mouse_hid_keyboard
        - usb_device_hid_generic
        - usb_device_hid_mouse
        - usb_device_msc_ramdisk
        - usb_device_msc_sdcard
        - usb_device_phdc_weighscale
        - usb_device_video_flexio_ov7670
        - usb_device_video_virtual_camera
        - usb_host_audio_speaker
        - usb_host_cdc
        - usb_host_hid_generic
        - usb_host_hid_mouse
        - usb_host_hid_mouse_keyboard
        - usb_host_msd_command
        - usb_host_msd_fatfs
        - usb_host_phdc_manager
        - usb_keyboard2mouse
        - usb_pin_detect_hid_mouse

Cypress WICED framework for MCUXpresso SDK

The current version of Cypress WICED framework is 1.0.0.

- 1.0.0
  - Initial version of Cypress WICED framework for MCUXpresso SDK

wolfSSL

The current version is 3.9.8_rev3, based on Release 3.9.8 of wolfSSL.

- 3.9.8_rev3
  - New features:
    - Added support for DCP driver.
- 3.9.8_rev2
  - New features:
    - Added support for CAU3 driver.
- 3.9.8_rev1
  - New features:
    - Added support for CAAM driver.
    - Added FREESCALE_ALT macros.
- 3.9.8
  - New features:
    - Added support for AES and SHA acceleration modules of LPC devices. Accelerates AES and SHA wolfSSL modules.
    - LTC acceleration for AES CBC now updates IV.
  - Bug fixes:
    - Fixed K8x/KL8x LTC RSA sign when FREESCALE_LTC_TFM_RSA_4096_ENABLE macro is enabled.
- 3.9.0
  - New features:
    - Added more LTC public key acceleration (curve25519, ed25519 and RSA4096).
    - FREESCALE_LTC_TFM_RSA_4096_ENABLE macro added to enable RSA4096 on K8x/KL8x LTC.
    - LTC_MAX_ECC_BITS increased to 384 to enable ECC-384 curve acceleration on LTC.
    - FREESCALE_LTC_SHA added for KL8x SHA-1 and SHA-256 hardware acceleration.
  - Other changes:
    - wolfSSL/wolfcrypt/settings.h is changed to remove unused macros and add support for KSDK 2.0.
    - LTC public key acceleration is implemented in separate source file ksdk_port.h and ksdk_port.c
- 3.8.0
  - New features:
    - Added support for LTC hardware acceleration module. Accelerates AES, 3DES, TFM module (modular integer arithmetic) and ECC wolfSSL modules.
    - Added support for random number generator modules TRNG and RNGA.
  - Other changes:
    - The MMCAU acceleration now uses "fsl_mmcau.h" instead of "cau_api.h".
    - In DSA, wc_dsaSign() changed to repeate wc_RNG_GenerateBlock() until k is less than q.
    - wolfSSL/wolfcrypt/settings.h is changed to remove unused macros and add support for KSDK 2.0.
    - In wolfcrypt/src/asn.c, ksdk_time(time_t) changed to extern, to be defined by application.

3 RTOS Change Log

FreeRTOS for MCUXpresso SDK.

The current version is Amazon-FreeRTOS 1.4.6 Original package is available at github.-com/aws/amazon-freertos.

- 1.4.6_rev0
  - New features:
    - Update support of CM33 port with Trustzone, MPU, FPU support
    - Add support for AWS test for Cypress WiFi
    - Use lwip netif api to avoid lwIP raw API calls outside of tcpip thread in aws_wifi.c
  - Other changes:
    - Fix issues with mflash driver
    - Fix other build warnings, errors
- 1.4.0_rev1
  - New features:
    - Add implementation of vTaskEndScheduler for CM0 GCC port.
    - Support for CM33, CM33F architectures based on CM3, CM4F ports
- 1.4.0_rev0
  - New features:
    - Support for pkcs11 for several platforms, secure element host library under pkcs11/portable/nxp folder
    - Lwip, wifi_qca support for secure_sockets in secure_sockets/portable/nxp folder
    - Flash driver support for several platforms in third_party/mcu_vendor/nxp folder
    - Generic support for aws_wifi under wifi/portable/nxp/common folder
  - Other changes:
    - Fix several build warnings, errors

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.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer’s applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and 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:
Document identifier: MCUXSDKMIMXRT106XCYPRESSWIFIRN

arm