NXP Semiconductors

User's Guide

Document identifier: MCUXSDKK32RN

Rev 0, 06/2019

MCUXpresso SDK Release Notes Supporting FRDM-K32L3A6

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 FatFs, USB, mbed TLS cryptography libraries, other middleware packages, and integrated RTOS support for FreeRTOS^{TM} 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, MCUXpresso SDK is the evolution of Kinetis SDK v2.x.x, 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)® 5.27
- Makefiles support with GCC revision 8-2018-q4-major GCC8 from Arm Embedded
- MCUXpresso IDE v11.0.1

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
FRDM-K32L3A60	K32L3A60VPJ1A

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
Qualcomm WiFi	<install_dir>/middleware/wifi_qca
Demo applications	<install_dir>/boards/<board_name>/demo_apps
USB demo applications	<install_dir>/boards/<board_name>/usb_examples
Driver examples	<install_dir>/boards/<board_name>/driver_examples
Cortex Microcontroller Software Interface Standard (CMSIS) driver examples	<install_dir>/boards/<board_name>/cmsis_driver_examples
FatFS examples	<install_dir>/boards/<board_name>/fatfs_examples
RTOS examples	<install_dir>/boards/<board_name>/rtos_examples
Qualcomm WiFi stack examples	<install_dir>/boards/<board_name>/wifi_qca_examples
mbed TLS examples	<install_dir>/boards/<board_name>/mbedtls_examples
Documentation	<install_dir>/docs
USB Documentation	<install_dir>/docs/usb
Middleware	<install_dir>/middleware
mbed TLS	<install_dir>/middleware/mbedtls
FatFS stack	<install_dir>/middleware/fatfs
USB stack	<install_dir>/middleware/usb
Driver, SoC header files, extension header files and feature header files, utilities	<install_dir>/devices/<device_name>
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

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.

6.2.2 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.

6.2.3 RTOS

The MCUXpresso SDK is integrated with FreeRTOS OS.

6.2.4 CMSIS

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

6.2.5 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 comply to MISRA 2012 rules with the following exceptions.

Table 3. MISRA exceptions

Exception Rules	Description
Rule 5.1	External identifiers shall be distinct.
Rule 5.4	Macro identifiers shall be distinct.
Rule 21.1	#define and #undef shall not be used on a reserved identifieror reserved macro name.
Rule 21.2	A reserved identifier or macro name shall not be declared.
Directive 4.4	Sections of code should not be "commented out".
Directive 4.5	Identifiers in the same name space with overlapping visibility should be typographically unambiguous.
Directive 4.6	Typedefs that indicate size and signedness should be used in place of the basic numerical types.
Directive 4.8	If a pointer to a structure or union is never dereferenced within a translation unit, then the implementation of the object should be hidden.
Directive 4.9	A function should be used in preference to a function-like macro where they are interchangeable.
Directive 4.13	Functions which are designed to provide operations on a resource should be called in an appropriate sequence.
Rule 1.2	Language extensions should not be used.
Rule 2.3	A project should not contain unused type declarations.
Rule 2.4	A project should not contain unused tag declarations.
Rule 2.5	A project should not contain unused macro declarations.
Rule 2.6	A function should not contain unused label declarations.
Rule 2.7	There should be no unused parameters in functions.
Rule 4.2	Trigraphs should not be used.
Rule 5.9	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.

Table continues on the next page...

Table 3. MISRA exceptions (continued)

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.
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 21.12	The exception handling features of <fenv.h> should not be used. .

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 USB HUB power supply

The external power supply of the USB HUB must be provided before it can be used. The development board power is not enough to supply multi-level USB HUBs and connected devices. Therefore, the external USB HUB that is connected to the development board should have its own power supply.

8.3 USB PID issue

Because the PID of all USB device examples is updated, uninstall the device drivers and then reinstall when the device (with new PID) is plugged in the first time.

8.4 Some ISSDK 1.0 projects do not work with ARMGCC toolchain

Some of ISSDK 1.0 projects have a problem with inclusion of files with the wrong case alignment. This causes the builds to fail for the ARMGCC toolchain under the Linux OS.

8.5 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).

8.6 New Project Wizard compile failure

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

NOTE

"xxx"means core variants like cm0plus, cm33, cm4, cm33_nodsp.

Components: Assert, assert_cm0plus, assert_xxx, assert_lite, baremetal, button, codec_i2c, codec_i2c_xxx, debug_console, debug_console_xxx, debug_console_lite, dialog7212, led, misc_utilities, panic, serial_manager, serial_manager_xxx, serial_manager_swo, serial_manager_swo_xxx, serial_manager_uart, serial_manager_uart_xxx, serial_manager_usb_cdc, serial_manager_usb_cdc_xxx, sgtl_adapter, sgtl5000, shell, shell_xxx, timer_manager, wm8904, wm8904_xxx, wm8904_adapter, wm8904_adapter_xxx, wm8960, wm8960_adapter, xip_device.

MCUXpresso SDK Release Notes Supporting FRDM-K32L3A6

Change Logs

Contents

Contents

Title | Page Number

Contents

Title | Page Number

1 Driver Change Log

CACHE

The current CACHE driver version is 2.1.0.

- 2.1.0
  - Delete L1CACHE_EnableCodeCacheWriteBuffer function because of no enable bit in register CPCR2.
- 2.0.0
  - Initial version.

COMMON

The current COMMON driver version is 2.1.3.

- 2.1.3
  - MISRA C-2012 issue fixed.
    - Fix the rule: rule-10.3.
- 2.1.2
  - Add SUPPRESS_FALL_THROUGH_WARNING() macro for the usage of suppressing fallthrough warning.
- 2.1.1
  - fix bug
    - Deleted and optimized repeated macro.
- 2.1.0
  - New Feature
    - Add IRQ operation for XCC toolchain
    - Add group IDs for newly supported drivers.
- 2.0.2
  - MISRA C-2012 issue fixed.
    - Fix the rule: rule-10.4.
- 2.0.1
  - Remove the implementation of LPC8XX Enable/DisableDeepSleepIRQ() function.
  - Add new feature macro switch "FSL_FEATURE_HAS_NO_NONCACHEABLE_SECTIO-N" 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.

CRC

The current CRC driver version is 2.0.1.

- 2.0.1
  - Bug fix:
    - DATA and DATALL macro definition moved from header file to source file.
- 2.0.0
  - Initial version.

DAC

The current DAC 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.

DMAMUX

The current DMAMUX driver version is 2.0.3.

- 2.0.3
  - Fix the issue for MISRA-2012 check.
    - Fixed rule 10.4, rule 10.3.
- 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 2.1.9.

- 2.1.9 -Bug fix:
  - Fixed MISRA issue, Rule 10.7, 10.8 in function EDMA_DisableChannelInterrupts and EDM-A_SubmitTransfer.
  - Fixed MISRA issue, Rule 10.7 in function EDMA_EnableAsyncRequest.
- 2.1.8

  - Bug fix:
    * Fixed wrong channel preemption base address used in EDMA_SetChannelPreemption-Config api that will cause channel preemption register cannot configure correctly.
- 2.1.7
  - Bug fix:
    * Fixed wrong transfer size setting
      · Add 8 bytes transfer configuration and feature for RT series
      · Add feature to support 16 bytes transfer for Kinetis
    * Fixed the issue that EDMA_HandleIRQ will go to incorrect branch When TCD is not used and callback function is not registered.
- 2.1.6
  - Bug fix:
    * Fixed KW3X MISRA Issue.
      · Rule 14.4, 10.8, 10.4, 10.7, 10.1, 10.3, 13.5, 13.2.
  - Improvements:
    * Clear IRQ handler that not available for specific platform with macro FSL_FEATURE_-EDMA_MODULE_CHANNEL_IRQ_ENTRY_SHARED_OFFSET.
- 2.1.5
  - Improvements:
    * Improve EDMA IRQ handler to support half interrupt feature.
- 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.

EWM

The current EWM driver version is 2.0.1.

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

FLASH

Current FLASH driver version is 3.0.1

- 3.0.0
  - New Features:
    - Add support FlexNVM alias for (kw37/38/39)
- 3.0.0
  - Improvement
    - Reorganize FTFx flash driver source file
    - Extract flash cache driver from FTFx driver
    - Extract flexnvm flash driver from FTFx driver
- 2.3.1
  - Bug Fix:
    - Unified Flash IFR design from K3

    - New encoding rule for K3 flash size
- 2.3.0
  - New Features:
    - Add support for device with LP flash (K3S/G)
    - Add flash prefetch speculation APIs
  - Improvement
    - Refine flash_cache_clear function
    - Reorganize the member of flash_config_t struct
- 2.2.0
  - New Features:
    - Support FTFL device in FLASH_Swap API
    - Support various pflash start addresses
    - Add support for KV58 in cache clear function
    - Add support for device with secondary flash (KW40)
  - Bug Fix:
    - Compiled execute-in-ram functions as PIC binary code for driver use
    - Added missed flexram properties
    - Fixed unaligned variable issue for execute-in-ram function code array
- 2.1.0
  - Improvement
    - Update coding style to align with KSDK 2.0
    - Different alignment size support for pflash and flexnvm
    - Improve the implementation of execute-in-ram functions
- 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 after all the data in ringbuffer are received in FLEXIO_UART_TransferReceiveNonBlocking.
- 2.1.4
  - Unified 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 disabling module/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_TransferGetReceiveCountDMA, FLEXIO_UART_TransferGetSendCountDMA, FLEXIO_UART_TransferGetReceiveCountEDMA and FLEXIO_UART_TransferGetSendCountEDMA
    - Fixed the Dozen mode configuration error in FLEXIO_UART_Init API. For enableInDoze = 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_TransferStopRingBuffer 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 2.1.7.

- 2.1.7
  - New feature:
    - Added API of checking bus pin status.
  - Bug fixes:

    * Fixed the issue that FLEXIO_I2C_MasterTransferBlocking does not wait for STOP bit sent.
    * Fixed COVERITY issue of useless call in FLEXIO_I2C_MasterTransferRunStateMachine.
    * Fixed the issue that I2C master does not check whether bus is busy before transfer.
- 2.1.6
  - Bug fix:
    * Fixed the issue that I2C Master transfer APIs(blocking/non-blocking) do not support the situation of master transfer with subaddress and transfer data size zero, which means no data follows the subaddress.
- 2.1.5
  - Unified 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 disabling module/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 which leads to s_flexioHandle/s_flexioIsr/s_flexioType variable being written. 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
  - Unified 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 disabling module/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_GetTxData-RegisterAddress.
    * 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:
    * Added 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
  - Unified 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 disabling module/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.
    - Fixed 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
  - Unified 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 disabling module/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
  - Unified 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 disabling module/clock.
    * Updated module Enable APIs to only support enable operation.
- 2.1.0
  - New features:
    * Added Transfer prefix in transactional APIs.

GPIO

The current driver version is 2.4.0.

- 2.4.0:
  - API interface added:
    * New APIs were added to configure the GPIO interrupt clear settings.
- 2.3.2
  - Fix the issue for MISRA-2012 check.
    * Fixed rule 3.1, 10.1, 8.6, 10.6, 10.3.
- 2.3.1:
  - Remove deprecated APIs.
- 2.3.0:
  - New feature:
    * Update the driver code to adapt the case of interrupt configurations in GPIO module. New APIs were added to configure the GPIO interrupt settings if the module has this feature on it.
- 2.2.1:
  - API interface changes:
    * Refined naming of API while keep all original APIs by marking them as deprecated. Original API will be removed in next release. The main change is update API with prefix of _PinXXX() and _PortXXX.
- 2.1.1:
  - API interface changes:
    * Added API for the check attribute bytes.
- 2.1.0:
  - API interface changes:
    * Added "pins" or "pin" to some APIs' names.
    * Renamed "_PinConfigure" to "GPIO_PinInit".

INTMUX

The current INTMUX driver version is 2.0.1.

- 2.0.1
  - Improvements:

    - Added weak function implementations of INTMUX1_x_DriverIRQHandler, x ranges from 0 to 7 for supporting 8 channels.
- 2.0.0
  - Initial version.

LLWU

The current LLWU driver version is 2.0.3.

- 2.0.3
  - Bug Fix: -Fix MISRA-2012 rules.
    - Rule 16.4.
- 2.0.2
  - Optimization
    - correct driver function LLWU_SetResetPinMode parameter name.
  - Bug Fix:
    - Fix MISRA-2012 rules.
      - Rule 14.4, 10.8, 10.4, 10.3.
- 2.0.1
  - Miscellaneous changes:
    - Updates for KL8x.
- 2.0.0
  - Initial version.

LPADC

The current LPADC driver version is 2.1.1.

- 2.1.1
  - Update the gain calibration formula.
  - Use feature to segregate the new item kLPADC_TriggerPriorityPreemptSubsequently.
- 2.1.0
  - New Features:
    - Add the API LPADC_SetOffsetValue() to support configure offset trim value manually.
    - Add the API LPADC_DoOffsetCalibration() to do offset calibration independently.
  - Improvements:
    - Improve the usage of macros and remove invalid macros.
- 2.0.2
  - Add supports for platforms with 2 FIFOs and different calibration measures.
- 2.0.1
  - Ensure the API LPADC_SetConvCommandConfig configure related registers correctly.
- 2.0.0
  - Initial version.

LPCMP

The current LPCMP driver version is 2.0.2.

- 2.0.2
  - Bug Fix:
    - Current API LPCMP_ClearStatusFlags has to check w1c bits.
- 2.0.1
  - Add control macro to enable/disable the CLOCK code in current driver.
- 2.0.0
  - Initial version.

LPI2C

The current LPI2C driver version is 2.1.10.

- 2.1.10
  - Fix unaligned access issue in LPI2C_RunTransferStateMachine.
  - Fix uninitialized variable issue in LPI2C_MasterTransferHandleIRQ.
- 2.1.9
  - Fix coverity issue of unchecked return value in I2C_RTOS_Transfer, fix coverity issue of operands don't affect result in LPI2C_SlaveReceive and LPI2C_SlaveSend.
  - Remove STOP signal wait when NAK detected.
  - Clear slave repeat start flag before transmission started in LPI2C_SlaveSend/LPI2C_Slave-Receive. The issue is that LPI2C_SlaveSend/LPI2C_SlaveReceive don't handle with the re-served repeat start flag and this will cause the next slave send break and the master will always in receive data status but can't receive data.
- 2.1.8
  - Bug fix:
    - Fixed the issue that transfer with LPI2C_MasterTransferNonBlocking, kLPI2C_Transfer-NoStopFlag and wait tranfer done through callback way is not doing blocking transfer.
    - Fixed the issue that STOP signal does not appear in the bus when NAK event happens.
- 2.1.7
  - Bug fix:
    - Clear the stopflag before transmission started in LPI2C_SlaveSend/LPI2C_SlaveReceive. The issue is that LPI2C_SlaveSend/LPI2C_SlaveReceive don't handle with the reserved stop flag and this will cause the next slave send break and the master will always in receive data status but can't receive data.
- 2.1.6
  - Bug fix:
    - Fixed driver MISRA build error and C++ build error in LPI2C_MasterSend and LPI2C_-SlaveSend.
    - Reset fifos in LPI2C Master Transfer functions to aviod there are still remaining bytes in fifo during last transfer.
    - Fixed the issue that LPI2C_MasterStop does not return the correct nak status in the bus

for second transfer to the non-existing slave address.

- 2.1.5
  - Bug fix:
    - Extended the Driver IRQ handler to support LPI2C4 and change to use ARRAY_SIZE(kLpi2cBases) 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. Previously, the API uses scatter-gather method, which handles the command transfer first, then handles the linked TCD which is preset with the receive data transfer. The issue is that the TX DMA request and the RX DMA request are both enabled, so when the DMA finished the first command TCD transfer and handled the receive data TCD, the TX DMA request still happens due to TX FIFO empty. The result is the RX DMA transfer starts, without waiting on the expected RX DMA request.
    - Fixed 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 auto-stop 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 cannot 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.

LPIT

The current LPIT driver version is 2.0.0.

- 2.0.0
  - Initial version.

LPSPI

The current LPSPI driver version is 2.0.4.

- 2.0.4
  - Fixed in LPSPI_MasterTransferBlocking that master rxfifo may overflow in stall condition.
- 2.0.3
  - Bug Fix:
    * Removed the LPSPI_Reset from LPSPI_MasterInit and LPSPI_SlaveInit because this API may glitch the slave select line. Please call this function manually if needed.
- 2.0.2
  - New features:
    * Added dummy data setup API to allow users to configure the dummy data to be transferred.
    * Enabled the 3-wire mode in which SIN and SOUT pins can be configured as input/output pin.
- 2.0.1
  - Bug fixes:
    * The clock source should be divided by the 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.

LPTMR

The current LPTMR driver version is 2.1.0.

- 2.1.0
  - Implement:
    * Implement for some special devices no supporting for all clock sources.

  - Bug Fix:
    - Fix issue when access CMR register.
- 2.0.2
  - Bug Fix:
    - Fix MISRA-2012 issues.
      - Rule 10.1.
- 2.0.1
  - Driver update:
    - Updated the LPTMR driver to support 32-bit CNR and CMR registers in some devices.
- 2.0.0
  - Initial version.

LPUART

The current LPUART driver version is 2.2.7.

- 2.2.7
  - Fix the issue for MISRA-2012 check.
    - Fixed rule-12.1, rule-17.7, rule-14.4, rule-13.3, rule-14.4, rule-10.4, rule-10.8, rule-10.3, rule-10.7, rule-10.1, rule-11.6, rule-13.5, rule-11.3, rule-13.2, rule-8.3.
- 2.2.6
  - Fix the repeatedly reading status register issue while dealing with the IRQ routine.
- 2.2.5
  - Bug fix:
    - 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 detecting feature in LPUART_TransferNonBlocking function. If an idle line is 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 ends.
- 2.2.3
  - Changed parameter type in LPUART_RTOS_Init struct from rtos_lpuart_config to lpuart_rtos_config_t.
  - Bug fix:
    - Disabled LPUART receive interrupt instead of all NVIC when reading data from ring buffer. Otherwise when the ring buffer is used, receive nonblocking method will disable all NVICs to protect the ring buffer. This may has a negative effect on other IPs that 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 unnecessary 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.

MMDVSQ

The current MMDVSQ driver version is 2.0.2.

- 2.0.2
  - Bug fix:
    - Fixed MMDVSQ_GetExecutionStatus function get execution status wrong.
- 2.0.1
  - Other changes:
    - Changed name of MMDVSQ_GetDivideRemainder and MMDVSQ_GetDivideQuotient functions.
- 2.0.0
  - Initial version.

MSMC

The current MSMC driver version is 2.1.0.

- 2.1.0
  - Added new APIs with FEATURE macros support: SMC_GetStopEntryStatus() SMC_ClearStopEntryStatus() SMC_SetForceBootOptionConfig() SMC_SRAMEnableLowPowerMode() SMC_SRAMEnableDeepSleepMode()
  - Updated APIs with FEATURE macros support: SMC_SetPowerModeStop() SMC_SetPowerModeVlpr() SMC_SetPowerModeLls() SMC_SetPowerModeVlls() SMC_ConfigureResetPinFilter()
- 2.0.0
  - Initial version.

MU

The Current MU driver version is 2.0.2.

- 2.0.2
  - Added support for MIMX8MQx.
- 2.0.1
  - Added support for MCIMX7Ux_M4.
- 2.0.0
  - Initial version.

PORT

The current PORT driver version is 2.1.0.

- 2.1.0
  - New feature:
    - Update the driver code to adapt the case of the interrupt configurations in GPIO module. will move the pin configuration APIs to GPIO module.
- 2.0.2
  - Miscellaneous changes:
    - Added feature guard macros in the driver.
- 2.0.1
  - Miscellaneous changes:
    - Added "const" in function parameter.
    - Updated some enumeration variables' names.

RTC

The current RTC driver version is 2.2.0.

- 2.2.0
  - Fix MISRA C-2012 issue. -Fixed rule contain: rule-17.7, rule-14.4, rule-10.4, rule-10.7, rule-10.1, rule-10.3.
  - Fix Central repo code formatting issue.
  - Add api for enabling wakeup pin.
- 2.1.0
  - Add feature macro check for many features.
- 2.0.0
  - Initial version.

SAI

The current SAI driver version is 2.2.1.

- 2.2.1
  - Improvement:
    - Add mclk post divider support in function SAI_SetMasterClockDivider.
    - Remove useless configuration code in SAI_RxSetSerialDataConfig.
  - Bugfix:
    - Fix the sai sdma drvier build issue caused by the wrong structure member name used in function SAI_TransferRxSetConfigSDMA/SAI_TransferTxSetConfigSDMA.
    - Fix BAD BIT SHIFT OPERATION issue that caused by the FSL_FEATURE_SAI_CH-ANNEL_COUNTn.
    - Apply ERR05144: not set FCONT = 1 when TMR > 0 , otherwise the TX may not work.
- 2.2.0
  - Improvement:
    - Add new APIs for parameters collection and user interfaces simplify: SAI_Init SAI_SetMasterClockConfig
      SAI_TxSetBitClockRate SAI_TxSetSerialDataConfig SAI_TxSetFrameSyncConfig S-AI_TxSetFifoConfig SAI_TxSetBitclockConfig SAI_TxSetConfig SAI_TxSetTransfer-Config
      SAI_RxSetBitClockRate SAI_RxSetSerialDataConfig SAI_RxSetFrameSyncConfig S-AI_RxSetFifoConfig SAI_RxSetBitclockConfig SAI_RXSetConfig SAI_RxSetTransfer-Config
      SAI_GetClassicI2SConfig SAI_GetLeftJustifiedConfig SAI_GetRightJustifiedConfig S-AI_GetTDMConfig
- 2.1.9
  - Improvement:
    - Improve SAI driver comment for clock polarity.
    - Add enum for SAI for sample inputs on different edge.
    - Change FSL_FEATURE_SAI_CHANNEL_COUNT to FSL_FEATURE_SAI_CHANN-EL_COUNTn(base) for the difference between the different SAI instance.
    - Add new api SAI_TxSetBitClockDirection SAI_RxSetBitClockDirection SAI_RxSet-FrameSyncDirection SAI_TxSetFrameSyncDirection
- 2.1.8
  - Improvement:
    - Add feature macro test for the sync mode2 and mode 3.
    - Add feature macro test for masterClockHz in sai_transfer_format_t.
- 2.1.7
  - Improvement:
    - Add feature macro test for the mclkSource member in sai_config_t.
    - Change "FSL_FEATURE_SAI5_SAI6_SHARE_IRQ" to "FSL_FEATURE_SAI_SAI5-_SAI6_SHARE_IRQ".
    - Add #ifndef #endif check for SAI_XFER_QUEUE_SIZE to allow redefinition.
  - 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:
    - Reduced code size while not using transactional API.

2.1.0

- API name change:
    - SAI_GetSendRemainingBytes -> SAI_GetSentCount.
    - SAI_GetReceiveRemainingBytes -> SAI_GetReceivedCount.
    - All transactional 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.
    - Fixed MOE enable flow issue, move MOE enable after MICS settings in SAI_TxInit/SAI_Rx-Init.

2.0.0

- Initial version.

SEMA42

The current SEMA42 driver version is 2.0.0.

- 2.0.0
  - Initial version.

SIM

The current SIM driver version is 2.1.0.

- 2.1.0
  - Added new APIs of SIM_GetRfAddr() and SIM_EnableSystickClock().
- 2.0.0
  - Initial version.

SMARTCARD

The current SMARTCARD driver version is 2.2.0.

- 2.2.0
  - New features:
    - Updated to use RX/TX FIFO.
- 2.1.2
  - Provided time delay function which works in microseconds.
  - Bug fix:
    - Changed event to semaphore in RTOS driver (KPSDK-11634).
    - Added check if de-initialized variables are not null in SMARTCARD_RTOS_Deinit() (KPSDK-8788).
    - Changed deactivation sequence in SMARTCARD_PHY_TDA8035_Deactivate() to properly stop the clock (POSCR-35).
    - Fixed timing issue with VSEL0/1 signals in smartcard TDA8035 driver (KPSDK-10160).
- 2.1.1
  - New features:
    - Added default phy interface selection into smartcard RTOS drivers (KPSDK-9063).
    - Replaced smartcard_phy_ncn8025 driver by smartcard_phy_tda8035.
  - Bug fix:
    - Fixed protocol timers activation sequences in smartcard_emvsim and smartcard_phy_-tda8035 drivers during emvl1 pre-certification tests (KPSDK-9170, KPSDK-9556).
- 2.1.0

  - Initial version.

SPM

The current SPM driver version is 2.2.0.

- 2.2.0
  - Bug Fix
    * Fix MDK 66-D waring (enumeration value out of int range) on kSPM_DcdcStableOK-Flag.
  - Other Changes
    * Remove the support for K3S.
    * Change register and mask names to support K32-L3.
  - Optimization
    * Convert SPM_EnableVddxStepLock() and SPM_SetLowPowerReqOutPinConfig() to in-line functions.
- 2.1.0
  - Bug Fix
    * Correct spelling mistake of function name.
- 2.0.0
  - Initial version.

TPM

The current TPM driver version is 2.0.5.

- 2.0.5 -Bug Fix:
  - Fix MISRA-2012 rules.
    * Rule 10.6, 10.7
- 2.0.4
  - Bug fixes.
    * Fixed ERR050050 in functions TPM_SetupPwm/TPM_UpdatePwmDutycycle. When T-PM is configured in EPWM mode as PS = 0, the compare event is missed on the first reload/overflow after writing 1 to the CnV register.
- 2.0.3
  - MISRA-2012 issue fixed.
    * Fixed rule contain: rule-12.1, rule-17.7, rule-16.3, rule-14.4, rule-1.3, rule-10.4, rule-10.3, rule-10.7, rule-10.1, rule-10.6, rule-18.1.
- 2.0.2
  - Bug fixes:
    * Fixed issues in functions TPM_SetupPwm/TPM_UpdateChnlEdgeLevelSelect /TPM-_SetupInputCapture/TPM_SetupOutputCompare/TPM_SetupDualEdgeCapture, wait ac-knowledgement when channel disabled.
- 2.0.1

  - Bug fixes:
    * Fix TPM_UpdateChnlEdgeLevelSelect ACK wait issue.
    * Fix TPM_SetupdualEdgeCapture cannot set FILTER register issue.
    * Fix TPM_UpdateChnEdgeLevelSelect ACK wait issue.
- 2.0.0
  - Initial version.

TRGMUX

The current TRGMUX driver version is 2.0.0.

- 2.0.0
  - Initial version.

TSTMR

The current TSTMR driver version is 2.0.0.

- 2.0.0
  - Initial version.

USDHC

The current USDHC driver version is 2.3.0.

- 2.3.0
  - Improvements:
    * Add USDHC_SetDataConfig api to support manual tuning.
    * Remove the limitaion that source clock must be bigger than the target in function USDH-C_SetSdClock by use source clock frequency as target directly.
    * Add peripheral reset in USDHC_Init function.
    * Add tuning reset support in function USDHC_Reset function.
- 2.2.8
  - Fix out-of bounds write in function USDHC_ReceiveCommandResponse.
- 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.

VREF

The current VREF driver version is 2.1.1.

- 2.1.1
  - MISRA-2012 issue fixed.
    - Fixed rule contain: rule-10.4, rule-10.3, rule-10.1.
- 2.1.0
  - Added new functions:
    - Supported L5K board: add VREF_SetTrim2V1Val() and VREF_GetTrim2V1Val() functions to supply 2V1 output mode.

- 2.0.0
  - Initial version.

WDOG32

The current WDOG32 driver version is 2.0.1.

- 2.0.1
  - Bug fixes:
    - WDOG must be configured within its configuration time period
      - Added WDOG32_Init API to quick access section.
      - Defined register variable in WDOG32_Init API.
- 2.0.0
  - Initial version.

XRDC

The current XRDC driver version is 2.0.3.

- 2.0.3
  - Updates:
    - Added necessary driver supports for K32H844P.
    - Added new APIs concerning new features of Exclusive Access Lock and domain programmable access flags configurations.
- 2.0.2
  - Bug fixes:
    - Fixed wrong assert of assignIndex input check in the xRDC driver.
  - Improvements:
    - Added master input CPU/non-CPU check in XRDC_SetNonProcessorDomainAssignment and XRDC_SetProcessorDomainAssignment API.
    - Added necessary assert checks for several config inputs.
- 2.0.1
  - Improvements:
    - Changed reserved bit fields in the structs into unnamed-identifier bit fields.
- 2.0.0
  - Initial version.

2 Middleware Change Log

FatFs for MCUXpresso SDK

Current version is FatFs R0.13c_rev0.

- R0.13c_rev0
  - Upgraded to version 0.13c
  - Apply patches ff_13c_p1.diff,ff_13c_p2.diff, ff_13c_p3.diff and ff_13c_p4.diff.
- 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.

mbedTLS for MCUXpresso SDK

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

- 2.13.1_rev3
  - Bug fixes:
    - Force align AES_CCM and AES_GCM self-test keys to fix unaligned key issue when using HW acceleration.
- 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.

Multicore SDK

The current version of Multicore SDK is 2.6.0.

- 2.6.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.7.2
    - eRPC generator (erpcgen) v.1.7.2
    - Multicore Manager (MCMgr) v4.0.3
    - RPMsg-Lite v2.2.0
  - New features:
    - eRPC: Improved support of const types.
    - eRPC: Fixed Mac build.

    * eRPC: Fixed serializing python list.
    * eRPC: Documentation update.
    * eRPC: Add missing doxygen comments for transports.
    * RPMsg-Lite: Added configuration macro RL_DEBUG_CHECK_BUFFERS.
    * RPMsg-Lite: Several MISRA violations fixed.
    * RPMsg-Lite: Added environment layers for QNX and Zephyr.
    * RPMsg-Lite: Allow environment context required for some environments (controlled by the RL_USE_ENVIRONMENT_CONTEXT configuration macro).
    * RPMsg-Lite: Data types consolidation.
    * MCMgr: Documentation updated to describe handshaking in a graphic form.
    * MCMgr: Minor code adjustments based on static analysis tool findings
- 2.5.0
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.7.1
    * eRPC generator (erpcgen) v.1.7.1
    * Multicore Manager (MCMgr) v4.0.2
    * RPMsg-Lite v2.0.2
  - New features:
    * RPMsg-Lite, MCMgr: Align porting layers to the updated MCUXpressoSDK feature files.
    * eRPC: Fixed semaphore in static message buffer factory.
    * erpcgen: Fixed MU received error flag.
    * erpcgen: Fixed tcp transport.
- 2.4.0
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.7.0
    * eRPC generator (erpcgen) v.1.7.0
    * Multicore Manager (MCMgr) v4.0.1
    * RPMsg-Lite v2.0.1
  - New features:
    * eRPC: Improved code size of generated code.
    * eRPC: Generating crc value is optional.
    * eRPC: Fixed CMSIS Uart driver. Removed dependency on KSDK.
    * eRPC: List names are based on their types. Names are more deterministic.
    * eRPC: Service objects are as a default created as global static objects.
    * eRPC: Added missing doxygen comments.
    * eRPC: Forbid users use reserved words.
    * eRPC: Removed outByref for function parameters.
    * eRPC: Added support for 64bit numbers.
    * eRPC: Added support of program language specific annotations.
    * eRPC: Optimized code style of callback functions.
    * RPMsg-Lite: New API rpmsg_queue_get_current_size()
    * RPMsg-Lite: Fixed bug in interrupt handling for lpc5411x, lpc5410x
    * RPMsg-Lite: Code adjustments based on static analysis tool findings
- 2.3.1
  - Multicore SDK component versions:

        * embedded Remote Procedure Call (eRPC) v1.6.0
        * eRPC generator (erpcgen) v.1.6.0
        * Multicore Manager (MCMgr) v4.0.0
        * RPMsg-Lite v1.2.0
    - New features:
        * eRPC: Improved code size of generated code.
        * eRPC: Improved eRPC nested calls.
        * eRPC: Improved eRPC list length variable serialization.
        * eRPC: Added support for scalar types.
        * MCMgr: Added new MCMGR_TriggerEventForce() API.
- 2.3.0
    - Multicore SDK component versions:
        * embedded Remote Procedure Call (eRPC) v1.5.0
        * eRPC generator (erpcgen) v.1.5.0
        * Multicore Manager (MCMgr) v3.0.0
        * RPMsg-Lite v1.2.0
    - New features:
        * eRPC: Added support for unions type non-wrapped by structure.
        * eRPC: Added callbacks support.
        * eRPC: Added support annotation for functions.
        * eRPC: Added support

SDMMC

The current driver version is 2.2.12.

- 2.2.12
    - Improvement:
        * Add manual tuning function for looking for the tuning window automatically.
            · Add reset all in the manual tuning to make sure host controller state machine is correct for next tuning.
            · Add delay between each tuning block to make sure card status is ready for next tuning.

Host contoller layer update to support USDHC(not support SD 3.0).

- BugFix:
    - Fix the build warning by changing the old style function declaration static status_t inline to static inline status_t(found by adding -Wold-style-declaration in armgcc build flag).
    - Fix the fall through build warning by adding SUPPRESS_FALL_THROUGH_WARNING() in sdmmc driver.

2.2.11

- BugFix
    - Fix NULL pointer dereference issue when calling function SDMMCHOST_CardDetectInit in host adaptor layer.
    - Fix logical dead code issue in SDMMC_SwitchToVoltage function.

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 reigster 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_Write-Blocks.

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 2.2.12.

- 2.2.12
  - Improvement:
    - Add manual tuning function for looking for the tuning window automatically.
    - Fix the build warning by changing the old style function declaration static status_t inline to static inline status_t(found by adding -Wold-style-declaration in armgcc build flag).
    - Fix the fall through build warning by adding SUPPRESS_FALL_THROUGH_WARNING() in sdio driver.
- 2.2.11
  - Buf fix:
    - Add check card async interrupt capability in function SDIO_GetCardCapability.
    - Fix OUT OF BOUNDS access in function SDIO_IO_Transfer.
- 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 2.3.0.

- 2.3.0
  - New features:
    - add host video camera support. example: usb_host_video_camera
    - add a new device example. example: usb_device_composite_cdc_hid_audio_unified
- 2.2.0
  - New features:
    - add device DFU support.
    - Support OM13790DOCK on LPCXpresso54018.
    - add multiple logical unit support in msc class driver, update usb_device_lba_information-_struct_t to support this.
    - support multiple transfers for host ISO on IP3516HS.
  - Bug fix:
    - fix device ip3511 prime data length than maxpacket size issue.
    - initialize interval attribute in usb_device_endpoint_struct_t/usb_device_endpoint_init_-struct_t.
    - remove unnecessary headfile in device CDC class driver, remove unnecessary usb_echo and add DEBUG macro for necessary usb_echo in device CDC class driver.
    - fix device IP3511HS unfinished interrupt transfer missing issue.
- 2.1.0
  - New features:
    - add host RNDIS support. example: lwip_dhcp_usb
    - enable USB 3.0 support on device stack.
    - Power Delivery 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

QCA WiFi

The current version is 2.0.0.

- 2.0.0
  - Initial version.
    - Added QCA WiFi, ported from SDK 1.3, synchronized with latest MQX Qualcomm v3.-3.5.
  - Known issues:
    - Low power mode may not work, require further investigation.
    - DHCP request requires some timeout to retrieve valid data.