NXP Semiconductors
User's Guide

Document Number: MCUXSDKK80FRN
Rev. 0, 06/2019

MCUXpresso SDK Release Notes Supporting TWR-K80F150M, TWR-K81F150M, and FRDM-K82F

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 v11.0.0

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
TWR-K80F150M, TWR-K81F150M, FRDM-K82F	MK81FN256CAx15, **MK81FN256VDC15**, MK81FN256VLL15, MK81FN256VLQ15, MK80FN256CAx15, **MK80FN256VDC15**, MK80FN256VLL15, MK80FN256VLQ15, MK82FN256CAx15, MK82FN256VDC15, **MK82FN256VLL15**, MK82FN256VLQ15

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
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
mmCAU examples	<install_dir>/middleware/mmcau_examples
EMV examples	<install_dir>/boards/<board_name>/emv_examples
RTOS examples	<install_dir>/boards/<board_name>/rtos_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
EMV stack	<install_dir>/middleware/emv
FatFS stack	<install_dir>/middleware/fatfs
mmCAU	<install_dir>/middleware/mmcau
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

Table continues on the next page...

Table 2. Release contents (continued)

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.

6.2.1.1 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 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.

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
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 4. 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 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.

8.4 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.

8.5 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 TWR-K80F150M, TWRK81F150M, and FRDM-K82F

Change Logs

Contents

Contents

Title	Page Number

Contents

Page
Title Number

1 Driver Change Log

ADC16

The current ADC16 driver version is 2.0.2.

- 2.0.2
  - Used conversion control feature macro instead of that in IO map.
- 2.0.1
  - Bug fix:
    - Fixed MISRA-2012 rules.
      - Rule 16.4, 10.1, 13.2, 14.4, 17.7.
- 2.0.0
  - Initial version

CMP

The current CMP driver version is 2.0.1.

- 2.0.1
  - Bug Fix:
    - Fix MISRA-2012 rules.
      - Rule 14.4, rule 10.3, rule 10.1, rule 10.4, rule 17.7.
- 2.0.0
  - Initial version.

CMT

The current CMT driver version is 2.0.2.

- 2.0.2
  - Bug Fix:
    - Fix MISRA-2012 rules.
      - Rule 14.4, rule 10.4, rule 10.3, rule 10.7, rule 10.1, rule 10.6, rule 17.7.
- 2.0.1
  - Miscellaneous changes:
    - Added static to global CMT variables.
- 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
  - Bug fix:
    - Moved the default DAC_Enable(..., true) from DAC_Init() to the application code so users can enable the DAC's output.

2.0.0

- Initial version.

DMAMUX

The current DMAMUX driver version is 2.0.3.

- 2.0.3
  - Fixed 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_SetSourceChange issue by changing the type of the parameter source from uint8_t to uint32_t.
- 2.0.0
  - Initial version.

DSPI

The current dspi driver version is 2.2.1.

- 2.2.1
  - Bug Fix:

        - Fix the bug for double excution of transfer complete callback in master interrupt transfer mode. In the interrupt routine, the DSPI interrupt may meet the situation of the interrupt pending by itself while receiveing the last frame, adding check the transfer state to excute the callback fucntion.
        - Fix the wrong logic in DSPI_SetFifoEnable().
    - MISRA C-2012 issue fixed.
        - Fixed rule contain: rule-12.1, rule-17.7, rule-16.4, rule-14.4, rule-10.4, rule-10.8, rule-10.3, rule-10.1, rule-10.6, rule-13.5, rule-11.3, rule-13.2, rule-8.3, rule-8.5.
- 2.2.0
    - New features:
        - Added gasket feature for SPI EDMA driver, which reduces one channel used in the EDMA master transfer. With this feature support, only two channels are needed. For example, if the gasket feature is supported, we could use the DSPI_MasterTransferCreateHandleEDMA function like below: DSPI_MasterTransferCreateHandleEDMA(EXAMPLE_DSPI_MASTER_BASEADDR, &g_dspi_edma_m_handle, DSPI_MasterUserCallback, &userData, &dspiEdmaMasterRxRegToRxDataHandle, NULL, &dspiEdmaMasterIntermediaryToTxRegHandle);
        - Added dummy data setup API to allow users to configure the dummy data to be transferred.
        - Added new APIs for half-duplex transfer function. Users can send and receive data by one API in the polling/interrupt/EDMA way, and users can choose to either transmit first or receive first. Additionally, the PCS pin can be configured as assert status in transmission (between transmit and receive) by setting the isPcsAssertInTransfer to true.
- 2.1.4
    - Bug fix:
        - DSPI EDMA driver: The DSPI instance that has separated so the DMA request source can now transfer up to 32767 Bytes data in one DSPI_MasterTransferEDMA() transfer.
- 2.1.3
    - Bug fix:
        - DSPI EDMA driver can no longer support the case that the transfer data size is odd, but the bitsPerFrame is greater than 8.
    - Optimization:
        - Added #ifndef/#endif to allow users to change the default TX value at compile time.
- 2.1.2
    - Bug fix:
        - DSPI_MasterTransferBlocking function would hang in some corner cases (for example, some cases with bitsPerFrame is 4,6 and kDSPI_MasterPcsContinuous transfer mode).
- 2.1.1
    - Bug fix:
        - Set the EOQ (End Of Queue) bit to TRUE for the last transfer in transactional APIs.
- 2.1.0
    - New features:
        - Added Transfer prefix in transactional APIs.

EDMA

The current eDMA driver version is 2.1.8.

- 2.1.8
  - Bug fix:
    - Fixed wrong channel preemption base address used in EDMA_SetChannelPreemptionConfig 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 hard fault issue.
- 2.0.0
  - Initial version.

FLASH

The current FLASH driver version is 3.0.0.

- 3.0.0
  - Improvements:
    - Reorganized FTFx flash driver source file.
    - Extracted flash cache driver from FTFx driver.
    - Extracted 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:
        - Added support for device with LP flash (K3S/G).
        - Added flash prefetch speculation APIs.
    - Improvements:
        - Refined flash_cache_clear function.
        - Reorganized the member of flash_config_t structure.
- 2.2.0
    - New features:
        - Supported FTFL device in FLASH_Swap API.
        - Supported various PFLASH start addresses.
        - Added support for KV58 in cache clear function.
        - Added 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
    - Improvements:
        - Updated coding style to align with KSDK 2.0.
        - Different alignment size support for PFLASH and FLEXNVM.
        - Improved 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:
        - Fixed 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
  - Triggered user callback when all data is in ringbuffer are received when calling 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 configuration 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_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 2.1.8.

- 2.1.8

  - Fixed coverity issue of useless call in FLEXIO_I2C_MasterTransferRunStateMachine.
- 2.1.7
  - Bug fix:
    - Fixed the issue that FLEXIO_I2C_MasterTransferBlocking does not wait for STOP bit sent.
- 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
  - 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 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 any more 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:
    - Added reset FLEXIO before FLEXIO I2S init to make sure FLEXIO status is normal.
- 2.1.5
  - Bug fix:
    - Fixed I2S driver use hard code for bitwidth setting.

- 2.1.4
  - Unified component's 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.
    - 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 configuration 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
  - 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 configuration 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.

FTM

The current FTM driver version is 2.1.1.

- 2.1.1 <<<<<<< Updated upstream

- Fix Coverity interger handing issue that the right operand of a left bit shift statement

- Fixed coverity integer handing issue where the right operand of a left bit shift statement

Stashed changes

should not be a negative value. This appears in FTM_SetReloadPoints().

2.1.0

- New feature:
  - Add a new API FTM_SetupPwmMode() to allow the user to set the channel match value in units of timer ticks. New configure structure called ftm_chnl_pwm_config_param_t was added to configure the channel's PWM parameters. This API is similar with FTM_SetupPwm() API, but the new API will not set the timer period(MOD value), it will be useful for users to set the PWM parameters without changing the timer period.
- Bug fixes:
  - Add feature macro to enable/disable the external trigger source configuration.

2.0.4

- Features:
  - Added to enable DMA transfer with new API:
    * FTM_EnableDmaTransfer()

2.0.3

- Bug fixes:
  - Updated the FTM driver to enable fault input after configuring polarity.

2.0.2

- Features:
  - Added support to Quad Decoder feature with new APIs:
    * FTM_GetQuadDecoderFlags()

        - FTM_SetQuadDecoderModuloValue()
        - FTM_GetQuadDecoderCounterValue()
        - FTM_ClearQuadDecoderCounterValue()

2.0.1

- Bug fixes:
    - Updated the FTM driver to fix write to ELSA and ELSB bits.
    - FTM combine mode: set the COMBINE bit before writing to CnV register.

2.0.0

- Initial version.

GPIO

The current driver version is 2.3.2.

- 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".

I2C

The current I2C driver version is 2.0.7.

- 2.0.7

  - Fix the issue for MISRA-2012 check.
    - Fixed rule 11.9 ,15.7 ,14.4 ,10.4 ,10.8 ,10.3, 10.1, 10.6, 13.5, 11.3, 13.2, 17.7, 5.7, 8.3, 8.5, 11.1, 16.1.
  - Improvements:
    - Added I2C_MASTER_FACK_CONTROL macro to enable FACK control for master transfer receive flow when IP support double buffer, then master could hold the SCL by manually set TX AK/NAK during data transfer.
  - Fix coverity issue of unchecked return value in I2C_RTOS_Transfer.
  - Bug fix:
    - Fix variable redefine issue by moving i2cBases from fsl_i2c.h to fsl_i2c.c.
- 2.0.6
  - 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.
- 2.0.5
  - Improvements:
    - Added I2C_WATI_TIMEOUT macro to allow the user to specify the timeout times for waiting flags in functional API and blocking transfer API.
- 2.0.4
  - Bug fixes:
    - Added proper handle for transfer config flag kI2C_TransferNoStartFlag to support transmit with kI2C_TransferNoStartFlag flag. Only supports write only or write+read with no start flag, does not support read only with no start flag.
- 2.0.3
  - Bug fixes:
    - Removed enableHighDrive member in the master/slave configuration structure because the operation to HDRS bit is useless, user needs to use DSE bit in port register to configure the high drive capability.
    - Added reset registers operation in I2C_MasterInit and I2C_SlaveInit APIs. Fixed issue where I2C could not switch between master and slave mode.
    - Improved slave IRQ handler to handle the corner case that stop flag and address match flag come synchronously.
- 2.0.2
  - Bug fixes:
    - Fixed issue in master receive and slave transmit mode with no stop flag. The master could not succeed to start next transfer because the master could not send out re-start signal.
    - Fixed data transfer out of order issue due to memory barrier
    - Added hold time configuration for slave. By leaving the SCL divider and MULT reset values when configure to slave mode, the setup and hold time of the slave is then reduced outside of spec for lower baudrates. This can cause intermittent arbitration loss on the master side.
  - New features:
    - Added address nak event for master.

        - Added general call event for slave.
- 2.0.1
    - New features:
        - Added double buffer enable configuration for Socs which have the DFEN bit in S2 register.
        - Added flexible transmit/receive buffer size support in I2C_SlaveHandleIRQ.
        - Added start flag clear, address match, and release bus operation in I2C_SlaveWrite/-ReadBlocking API.
    - Bug fix:
        - Changed the kI2C_SlaveRepeatedStartEvent to kI2C_SlaveStartEvent.

LLWU

The current LLWU driver version is 2.0.2.

- 2.0.2
    - Optimization:
        - Corrected driver function LLWU_SetResetPinMode parameter name.
    - Bug fix:
        - Fixed 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.

LMEM

The current LMEM driver version is 2.1.0.

- 2.1.0
    - Removed the write buffer enable from the cache enable API.
    - Added Enable write buffer APIs.
- 2.0.0
    - Initial version.

LPTMR

The current LPTMR driver version is 2.0.2.

- 2.0.2
    - Bug fix:
        - Fixed 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
  - Fixed 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
  - Fixed repeated 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.

PDB

The current PDB driver version is 2.0.2.

- 2.0.2
    - Improvement:
        - Used macros in feature file instead of that in IO map.
- 2.0.1
    - Changed PDB register base array to const.
- 2.0.0
    - Initial version.

PIT

The current PIT driver version is 2.0.2.

- 2.0.2
    - Bug fix:
        - Fixed MISRA-2012 issues.
            - Rule 10.1.
- 2.0.1
    - Bug fix:
        - Cleared timer enable bit for all channels in function PIT_Init() to make sure all channels stay in disable status before setting other configurations.
        - Fixed MISRA-2012 rules.
            - Rule 14.4, rule 10.4.
- 2.0.0
    - Initial version.

PMC

The current PMC driver version is 2.0.1.

- 2.0.1
    - Fixed MISRA issue.
        - Rule 10.8, rule 10.3.
- 2.0.0

    - Initial version.

PORT

The current PORT driver version is 2.1.0.

- 2.1.0
    - New feature:
        - Updated 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.

QSPI

The current QSPI driver version is 2.2.0.

- 2.2.0
    - New feature:
        - Added new API QSPI_ClearCache to clear cache for new IP feature FSL_FEATURE_QSPI_SOCCR_HAS_CLR_LPCAC.
    - Bug fix:
        - Fixed the QSPI_WriteBlocking API programming issue for low watermark, caused by previous improvement changing to use TX watermark signal to fill the TX FIFO. Revert change to previous implementation to use TX FIFO full flag for filling the FIFO, improve previous API by accessing TX data register directly.
        - Fixed the issue that QSPI_SetIPCommandSize incorrectly triggers a transaction.
        - Fixed clock divider accurate issue when use internal QSPI internal divider.
        - fixed build fail issue for some devices not support API QSPI_SetDqsConfig for DQS configuration.
- 2.1.0
    - New feature:
        - Added new API QSPI_SetDqsConfig for DQS configuration.
    - Improvements:
        - Updated the QSPI_WriteBlocking API to fill the TX FIFO once there are bytes of TX watermark room in the FIFO. This improves the performance of filling TX FIFO when watermark is high.
- 2.0.2
    - New Macro function:

        - Added QSPI_LUT_SEQ() function for users to set LUT table easily.
        - Added LUT command macros for users to easy use.
    - Comment update:
        - Added the comments for the limitation of QSPI_ReadBlocking and QSPI_TransferReceiveBlocking.
- 2.0.1
    - New API:
        - QSPI_SetReadArea to set the read area.
    - Bug fix:
        - Fixed QSPI_UpdateLUT function only update first LUT issue.
        - Fixed issue that some function that hardcode QSPI0 as base.
- 2.0.0
    - Initial version.

RCM

The current RCM driver version is 2.0.2.

- 2.0.2
    - Fixed MISRA issue.
        - Rule 10.8, rule 10.1, rule 13.2, rule 3.1.
- 2.0.1
    - [KPSDK-10249] Fixed kRCM_SourceSw bit shift issue.
- 2.0.0
    - Initial version.

RTC

The current RTC driver version is 2.2.0.

- 2.2.0
    - Fixed 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.
    - Fixed central repository code formatting issue.
    - Added API for enabling wakeup pin.
- 2.1.0
    - Added feature macro check for many features.
- 2.0.0
    - Initial version.

SAI

The current SAI driver version is 2.2.1.

- 2.2.1
  - Improvements:
    - Added mclk post divider support in function SAI_SetMasterClockDivider.
    - Removed useless configuration code in SAI_RxSetSerialDataConfig.
  - Bug fix:
    - Fixed the SAI SDMA driver build issue caused by the wrong structure member name used in the function SAI_TransferRxSetConfigSDMA/SAI_TransferTxSetConfigSDMA.
    - Fixed BAD BIT SHIFT OPERATION issue caused by the FSL_FEATURE_SAI_CHANNEL_COUNTn.
    - Apply ERR05144: not set FCONT = 1 when TMR > 0, otherwise the TX may not work.
- 2.2.0
  - Improvements:
    - Add new APIs for parameters collection and user interfaces simplify: SAI_Init SAI_SetMasterClockConfig SAI_TxSetBitClockRate SAI_TxSetSerialDataConfig SAI_TxSetFrameSyncConfig SAI_TxSetFifoConfig SAI_TxSetBitclockConfig SAI_TxSetConfig SAI_TxSetTransferConfig SAI_RxSetBitClockRate SAI_RxSetSerialDataConfig SAI_RxSetFrameSyncConfig SAI_RxSetFifoConfig SAI_RxSetBitclockConfig SAI_RXSetConfig SAI_RxSetTransferConfig SAI_GetClassicI2SConfig SAI_GetLeftJustifiedConfig SAI_GetRightJustifiedConfig SAI_GetTDMConfig
- 2.1.9
  - Improvements:
    - Improved SAI driver comment for clock polarity.
    - Added enumeration for SAI for sample inputs on different edge.
    - Changed FSL_FEATURE_SAI_CHANNEL_COUNT to FSL_FEATURE_SAI_CHANNEL_COUNTn(base) for the difference between the different SAI instance.
    - Added new API: SAI_TxSetBitClockDirection SAI_RxSetBitClockDirection SAI_RxSetFrameSyncDirection SAI_TxSetFrameSyncDirection
- 2.1.8
  - Improvements:
    - Added feature macro test for the sync mode2 and mode 3.
    - Added feature macro test for masterClockHz in sai_transfer_format_t.
- 2.1.7
  - Improvements:
    - Added feature macro test for the mclkSource member in sai_config_t.
    - Changed "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:
    - Fixed build error caused by feature macro test for mclkSource.

-2.1.6

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

-2.1.5

- Bug fix:
  - Corrected SAI3 driver IRQ handler name.
  - Added I2S4/5/6 IRQ handler.
  - Added base in handler structure to support different instances share one IRQ number.
- New features:
  - Updated SAI driver for MCR bit MICS.
  - Added 192 KHZ/384 KHZ 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_-RxInit.

2.0.0

- Initial version.

SDHC

The current SDHC driver version is 2.1.9.

- 2.1.9
  - Improvements:
    - Added API SDHC_GetEnabledInterruptStatusFlags and used it in SDHC_TransferHandleIRQ.
    - Removed useless member interruptFlag in sdhc_handle_t.
    - Fixed OUT-OF BOUNDS write in function SDHC_ReceiveCommandResponse.
- 2.1.8 Bug fix:
  - Disabled useless interrupt while DMA is used.
  - Fixed MDK 66-D warning.
- 2.1.7
  - Bug fix:
    - Fixed ADMA1 descriptor configuration error.
    - Improved set clock function to check the output frequency range.
- 2.1.6 -New features:
  - Added SDHC_CardDetectByData3 API to support detect card through DATA3.
  - Added host base address/user data parameter for all call back function.
- 2.1.5 -New features:
  - Added NON-WORD align data addr transfer support in DMA mode.
- 2.1.4
  - New features:
    - Added response error flag to check response once read from the card.
  - Bug fixes:
    - Fixed clock divider calumniate incorrect issue.
- 2.1.3
  - Improvement:
    - Modified some definition to be compatible with middleware adapter.
- 2.1.2
  - Bug fix:
    - Used function pointer for interrupt handler to reduce code size.
    - Bad status bit check behaviour when wait for initialization of SD card.
    - Added support NON-WORD aligned data size transfer mode for SDIO card.
- 2.1.1
  - Bug fix:

    - ∗ Fixed the compile error when ADMA1 is enabled.
- 2.1.0
  - New features:
    - ∗ Added a host descriptor to contain SDHC related attributes.
  - Bug fix:
    - ∗ Removed clock auto gated function because of that it is a hardware issue.
  - Other changes:
    - ∗ Added more SDIO card related command type.
    - ∗ Changed the callback mechanism in the non-blocking transaction API.
    - ∗ Merged the two ADMA configuration function to be one.
    - ∗ Changed the transaction API's name.

SDRAMC

The current SDRAMC driver version is 2.1.0.

- 2.1.0
  - API change:
    - ∗ Changed status_t SDRAMC_SendCommand() to void SDRAMC_SendCommand().
- 2.0.1
  - Miscellaneous changes:
    - ∗ Added static to the global SDRAMC variables.
- 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.

SMC

The current SMC driver version is 2.0.5.

- 2.0.5
  - Fixed issue for MISRA-2012 check.
    - * Fixed rule 15.7, rule 14.4, rule 10.3, rule 10.1, rule 10.4.
- 2.0.4
  - When entering stop modes, use RAM function for the flash synchronize issue. Application should make sure that, the RW data of fsl_smc.c is located in memory region which is not powered off in stop modes.
- 2.0.3
  - Added APIs SMC_PreEnterStopModes, SMC_PreEnterWaitModes, SMC_PostExitWait-Modes, and SMC_PostExitStopModes.
- 2.0.2
  - Bug fix:
    - * Added DSB before WFI, add ISB after WFI.
  - Miscellaneous changes:
    - * Updated SMC_SetPowerModeVlpw implementation.
- 2.0.1
  - Miscellaneous changes:
    - * Updated for KL8x.
- 2.0.0
  - Initial version.

SYSMPU

The current SYSMPU driver version is 2.2.1.

- 2.2.1
  - Fixed MISRA issue.
- 2.2.0
  - Renamed MPU to SYSMPU.
  - Changed macro definition for slave number and fix the get error status calculation.
- 2.1.1
  - Added the feature file macro definition limitation for the MPU_SetRegionRwMasterAccessRights().
- 2.1.0
  - API changes:
    - Changed the mpu_region_num_t and mpu_master_t to uint32_t.
    - Changed the mpu_low_masters_access_rights_t, mpu_high_masters_access_rights_t to mpu_rwxrights_master_access_control_t, mpu_rwrights_master_access_control_t.
    - Changed the MPU_SetRegionLowMasterAccessRights(), MPU_SetRegionHighMasterAccessRights() to MPU_SetRegionRwxMasterAccessRights(), MPU_SetRegionRwMasterAccessRights().
- 2.0.0
  - Initial version.

TPM

The current TPM driver version is 2.0.4.

- 2.0.4
  - Bug fixes:
    - Fixed ERR050050 in functions TPM_SetupPwm/TPM_UpdatePwmDutycycle. When TPM 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 acknowledgement when channel disabled.
- 2.0.1
  - Bug fixes:
    - Fixed TPM_UpdateChnIEdgeLevelSelect ACK wait issue.
    - Fixed TPM_SetupdualEdgeCapture cannot set FILTER register issue.

    - Fixed TPM_UpdateChnEdgeLevelSelect ACK wait issue.
- 2.0.0
  - Initial version.

TSI_V2

The current TSI_V2 driver version is 2.1.2.

- 2.1.2
  - Bug fixes:
    - Fixed w1c issues in status handling API.
- 2.1.1
  - New features:
    - Changed void TSI_DeInit(TSI_Type *base) to void TSI_Deinit(TSI_Type *base).
- 2.0.1
  - Other changes:
    - Changed default configuration structure member order.

TSI_V4

The current TSI_V4 driver version is 2.1.2.

- 2.1.2
  - Bug fixes:
    - Fixed w1c issues in status handling API.
    - Fixed register naming error in API "static inline void TSI_EnableEndOfScanDmaTransferOnly(TSI_Type *base, bool enable)".
    - Removed redundant status flags clear APIs when enable interrupts.
- 2.1.1
  - New features:
    - Changed void TSI_DeInit(TSI_Type *base) to void TSI_Deinit(TSI_Type *base).
- 2.0.1
  - Other changes:
    - Changed default configuration structure member order.

TSI_V5

The current TSI_V5 driver version is 2.1.0.

- 2.1.0
  - Bug fixes:
    - Fix incorrect TSI SSC clock calculation.
- 2.0.1
  - Added functions for M_TX_USED bitfield for ke16z only (Unused TX mutual pins can work

as GPIO).
- 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.

WDOG

The current WDOG driver version is 2.0.0.

- 2.0.0
  - Initial version.

CLOCK

Current CLOCK driver version is 2.3.0

- 2.3.0
  - Fix the issue for MISRA-2012 check.
    - Fixed rule 10.4, rule 10.1, rule 10.6, rule 13.5, rule 10.8.
  - Bug Fix:
    - Fix incorrect External Oscillator Configuration sequence and ensure oscillator configuration be executed before it be enabled.
  - New feature
    - Adding new API CLOCK_DelayAtLeastUs() implemented by DWT to allow users set delay in unit of microsecond.
- 2.2.1
  - Bug Fix:
    - Fix the issue that MCG could not switch to FEE/FBE/PBE modes when OSCERCLK clock not enabled.
- 2.2.0
  - New Features:

        - [KPSDK-9157] Update CLOCK_SetFeiMode/CLOCK_SetFbiMode/CLOCK_BootToFeiMode() to support set MCG_C4[DMX32]=1 in FEI/FBI modes.
    - Bug Fix:
        - Update IP_CLOCKS array, remove unused gates and add missing gates.
- 2.1.0
    - Other Changes:
        - Merge fsl_mcg and fsl_osc into fsl_clock.
- 2.0.0
    - Initial version.

2 Middleware Change Log

DMA_MANAGER

The current DMA_MANAGER driver version is 2.1.0.

- 2.1.0
    - Updated DMA manager interface to support dynamic configuration of the managed area. This is used for a platform with multiple cores.
- 2.0.0
    - Initial version.

EMVL1 for MCUXpresso SDK

The current driver version is 2.1.0.

- 2.1.0
    - Added abort transfer functionality.
- 2.0.2
    - Re-implemented function for sending commands in T=0.
    - Bug fix:
        * Fixed wrong size of response in T=0 (KPSDK-11248).
        * Fixed problem with command cases 3 in T=1, expected wrong length of response (KPSDK-11335).
        * Fixed wrong length of response in T=1 (KPSDK-11868).
        * Fixed usage application buffer for data payload and overhead associated with T=1 protocol (KPSDK-11336).
- 2.0.1
    - Bug fix:
        * Fixed low level driver protocol timers failures during emvl1 pre-certification tests (KPSDK-9556).
        * Fixed improper T0 commands response receiving (commands case2, case3 & case4 affected) what causes long commands responses (KPSDK-8707).
- 2.0.0
    - Initial version.

FatFs for MCUXpresso SDK

The 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
  - Added 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:
    - Forced to align AES_CCM and AES_GCM self-test keys to fix unaligned key issue when using hardware acceleration.
- 2.13.1_rev2
  - Bug fixes:
    - Disabled default hardware 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 SDK.
- 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 SDK.
- 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 SDK.
- 2.6.0_rev2
  - Bug fixes:
    - ssl_cookie.c now uses SHA256 for COOKIE_MD (instead of original SHA224). Some hardware crypto acceleration (such as CAU3) does not 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 SDK.
- 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 PK-HA.
  - 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.

MMCAU library

The current version is 2.0.1.

- 2.0.1
  - Bug fixes:
    - KPSDK-17133 fix bug in fsl_mmcau.c when AES key schedule array is not aligned.
- 2.0.0
  - New features:
    - Q4/2013 release of the CAU library.
    - Added fsl_mmcau.h/fsl_mmcau.c optional layer between application and legacy CAU library (cau_api.h). This API has no alignment requirements.

SDMMC

The current driver version is 2.2.11.

- 2.2.11
  - Bug fixes:
    - Fixed NULL pointer dereference issue when calling function SDMMCHOST_Card-DetectInit in host adaptor layer.
    - Fixed logical dead code issue in SDMMC_SwitchToVoltage function.

- 2.2.10
  - Bug fixes:
    - Added NUll pointer check for USDHC FreeRTOS adaptor transfer complete. callback.
    - Added event value check for all the FreeRTOS event to fix program hang when a card event occurs before created.
- 2.2.9
  - Improvements:
    - Added NULL pointer check for sdmmchostcard_usr_param_t member CD in card detect callback to avoid memory corruption.
    - Added card voltage switch function in sdmmhostcard_usr_param_t to allow application register card signal line voltage switch function.
  - Bug fixes:
    - Fixed host FreeRTOS adaptor and polling adaptor, where they cannot detect card insert bug for USDHC.
    - Fixed SDHC host layer build issue and typo issue.
- 2.2.8
  - Improvement:
    - Updated SDMMC to support SDIO interrupt.
- 2.2.7
  - Bug fixes:
    - Fixed MDK 66-D warning.
- 2.2.6
  - Improvements:
    - Removed some SoC-specific header files from porting layer.
    - Saved MMC OCR registers while sending CMD1 with argument 0.
  - Bug fixes:
    - Added MMC_PowerOn function in which there is delay function after powerup SD Card. Otherwise, the card may init fail.
- 2.2.5
  - New features:
    - Added SD_ReadStatus API to get 512-bit SD status.
    - Added error log support in SD Card functions.
    - Added SDMMC_ENABLE_SOFTWARE_TUNING to enable/disable software tuning. It is disabled by default.
    - Added error procedure in the transfer function to improve stability.
    - Removed 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 512 Bytes to improve performance when application uses a non-word align data buffer address.
    - Used OCR access mode bits to determine the MMC Card 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.
    - Kept 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 is 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:
        * Changed callback mechanism when sending a command.
        * Fixed low performance 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
    - Bug fix:
        * Added check card async interrupt capability in function SDIO_GetCardCapability.
        * Fixed OUT OF BOUNDS access in function SDIO_IO_Transfer.
- 2.2.10
    - Bug fix:
        * Fixed SDIO card driver get wrong IO number when the card IO number is bigger than 2.
    - New feature:
        * Added SDIO 3.0 support.
        * Added API SDIO_IO_RW_Direct for direct read/write card register access.
- 2.2.9

    - Improvement:
        - Added API SDIO_SetIOIRQHandler/SDIO_HandlePendingIOInterrupt to handle multi IO pending IRQ.
- 2.2.8
    - Improvement:
        - Updates SDMMC to support SDIO interrupt.
        - Added API SDIO_GetPendingInterrupt to get the pending IO interrupt.
- 2.2.7
    - Bug fix:
        - Fixed MDK 66-D warning.
- 2.2.6
    - New features:
        - Added a unified transfer interface for SDIO.
    - Bug fix:
        - Wrong pointer address used by SDMMCHOST_Init.
- 2.1.5
    - Bug fix:
        - Improved SDIO card init sequence and added 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 SDSPI driver version is 2.1.4.

- 2.1.4
    - Bug fix:
        - Fixed MDK 66-D warning.
- 2.1.3
    - Improved SDSPI code size and performance.
- 2.0.0
    - Initial version.

USB stack for MCUXpresso SDK

The current version of USB stack is 2.2.0.

- 2.2.0
    - New features:
        - Added device DFU support.

        - Supports OM13790DOCK on LPCXpresso54018.
        - Added multiple logical unit support in MSC class driver, updated usb_device_lba_information_struct_t to support this.
        - Supports multiple transfers for host ISO on IP3516HS.
    - Bug fixes:
        - Fixed device ip3511 prime data length than maxpacket size issue.
        - Initialized interval attribute in usb_device_endpoint_struct_t/usb_device_endpoint_init_struct_t.
        - Removed unnecessary header file in device CDC class driver, removed unnecessary usb_echo, and added DEBUG macro for necessary usb_echo in device CDC class driver.
        - Fixed device IP3511HS unfinished interrupt transfer missing issue.
- 2.1.0
    - New features:
        - Added host RNDIS support. example: lwip_dhcp_usb
        - Enabled USB 3.0 support on device stack.
        - Power Delivery feature: Added OM13790HOST support. Added auto policy feature. Printed e-marked cable information.
- 2.0.1
    - Bug fixes:
        - Fixed some USB issues: Fixed MSC CV test failed in MSC examples.
        - Changed 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

3 RTOS Change Log

FreeRTOS for MCUXpresso SDK.

The current version is Amazon-FreeRTOS 1.4.0 Original package is available at github.com/aws/amazon-freertos.

- 1.4.7_rev0
  - New features:
    - Add optional allocation scheme heap_useNewlib.c by D. Nadler.
    - Enable task aware debugging for cm33 platforms
    - Move tickless implementation to application layer
  - Other changes:
    - Fix other build warnings, errors
- 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

Updates applied to FreeRTOS kernel up to version 10.0.0 (up to Amazon - FreeRTOS merge). New kernel related changes will be described in section above as part of AWS package.

- 9.0.0_rev3
  - New features:
    - Tickless idle mode support for Cortex-A7. Add fsl_tickless_epit.c and fsl_tickless_generic.h in portable/IAR/ARM_CA9 folder.
    - Enabled float context saving in IAR for Cortex-A7. Added configUSE_TASK_FPU_SUPPORT macros. Modified port.c and portmacro.h in portable/IAR/ARM_CA9 folder.

  - Other changes:
    - Transformed ARM_CM core specific tickless low power support into generic form under freertos/Source/portable/low_power_tickless/.
- 9.0.0_rev2
  - New features:
    - Enabled MCUXpresso thread aware debugging. Add freertos_tasks_c_additions.h and configINCLUDE_FREERTOS_TASK_C_ADDITIONS_H and configFRTOS_MEMORY_SCHEME macros.
- 9.0.0_rev1
  - New features:
    - Enabled -flto optimization in GCC by adding **attribute**((used)) for vTaskSwitchContext.
    - Enabled KDS Task Aware Debugger. Apply FreeRTOS patch to enable configRECORD_STACK_HIGH_ADDRESS macro. Modified files are task.c and FreeRTOS.h.
- 9.0.0_rev0
  - New features:
    - Example freertos_sem_static.
    - Static allocation support RTOS driver wrappers.
  - Other changes:
    - Tickless idle rework. Support for different timers is in separated files (fsl_tickless_systick.c, fsl_tickless_lptmr.c).
    - Removed configuration option configSYSTICK_USE_LOW_POWER_TIMER. Low power timer is now selected by linking of apropriate file fsl_tickless_lptmr.c.
    - Removed configOVERRIDE_DEFAULT_TICK_CONFIGURATION in RVDS port. Use of **attribute**((weak)) is the preferred solution. Not same as _weak!
- 8.2.3
  - New features:
    - Tickless idle mode support.
    - Added template application for Kinetis Expert (KEx) tool (template_application).
  - Other changes:
    - Folder structure reduction. Keep only Kinetis related parts.

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: 06/2019
Document identifier: MCUXSDKK80FRN