Document Number: MCUXSDKK80FRN
Rev. 0, 11/2017

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

Contents

1 Overview

The MCUXpresso Software Development Kit (SDK) is a collection of software enablement for Microcontrollers that includes peripheral drivers, high-level stacks including USB and lwIP, integration with WolfSSL and mbed TLS cryptography libraries, other middleware packages, such as multicore support and FatFs, and integrated RTOS support for FreeRTOS™ OS. In addition to the base enablement, the MCUXpresso SDK is augmented with demo applications and driver example projects, and API documentation to help the customers quickly leverage the support of the MCUXpresso SDK.

For the latest version of this and other MCUXpresso SDK documents, see the MCUXpresso SDK homepage MCUXpresso-SDK: Software Development Kit.

2 MCUXpresso SDK

As part of the MCUXpresso software and tools, MCUXpressoSDK is the evolution of Kinetis SDK v2.3.0, 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, a new 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 FSL code, the filenames and source code in MCUXpresso SDK containing the legacy Freescale prefix 'FSL' has been left as is. The 'FSL' prefix has been redefined as the NXP Foundation Software Library.

3 Development Tools

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

- Kinetis Design Studio IDE v3.2
- IAR Embedded Workbench for Arm version 8.11.3
- MDK-Arm Microcontroller Development Kit (Keil)® 5.23
- Makefiles support with GCC revision v6-2017-q2 from Arm Embedded
- MCUXpresso IDE v10.1.0

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

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
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

Table continues on the next page...

Table 2. Release contents (continued)

RTOS examples	<install_dir>/boards/<board_name>/rtos_examples
Multicore examples	<install_dir>/boards/<board_name>/multiprocessor_examples
Documentation	<install_dir>/docs
USB Documentation	<install_dir>/docs/usb
lwIP Documentation	<install_dir>/docs/lwip
Middleware	<install_dir>/middleware
lwIP stack	<install_dir>/middleware/lwip
DMA manager	<install_dir>/middleware/dma_manager
EMV stack	<install_dir>/middleware/emv
FatFS stack	<install_dir>/middleware/fatfs
mmCAU	<install_dir>/middleware/mmcau
Multicore stack	<install_dir>/middleware/multicore
SDMMC card driver	<install_dir>/middleware/sdmmc
USB stack	<install_dir>/middleware/usb
WolfSSL stack	<install_dir>/middleware/wolfssl
Driver, SoC header files, extension header files and feature header files, utilities	<install_dir>/devices/<device_name>
Cortex Microcontroller Software Interface Standard (CMSIS) ARM Cortex®-M header files, DSP library source	<install_dir>/CMSIS
Peripheral Drivers	<install_dir>/devices/<device_name>/drivers
Utilities such as debug console	<install_dir>/devices/<device_name>/utilities
RTOS Kernel Code	<install_dir>/rtos
Tools	<install_dir>/tools

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 CMSIS-compliant 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.

The RTOS and middleware folders each contain examples demonstrating the use of the included source.

6.2 Middleware

6.2.1 USB stack

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

6.2.1.1 Peripheral devices tested with the USB Host stack

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

Table 3. Peripheral devices

Device type	Device
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

Table continues on the next page...

Table 3. Peripheral devices (continued)

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

6.2.2 TCP/IP stack

The lwIP TCP/IP stack is pre-integrated with MCUXpresso SDK and runs on top of the MCUXpresso SDK Ethernet driver with Ethernet-capable devices/boards. For details, see the *lwIP TCPIP Stack and MCUXpresso SDK Integration User's Guide* (document MCUXSDKLWIPUG).

6.2.3 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.

6.2.4 RTOS

The MCUXpresso SDK is integrated with FreeRTOS OS.

6.2.5 CMSIS

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

7 MISRA Compliance

All MCUXpresso SDK drivers and USB stack comply to MISRA 2004 rules with the following exceptions.

Exception Rules	Description
1.1	All code shall conform to ISO 9899:1990 Programming languages - C, amended and corrected by ISO/IEC 9899/COR1:1995, ISO/IEC 9899/AMD1:1995, and ISO/IEC
2.4	Sections of code should not be commented out.
5.1	Identifiers (internal and external) shall not rely on the significance of more than 31 characters.
6.3	typedefs that indicate size and signedness should be used in place of the basic types.
6.4	Bitfields shall only be defined to be of type unsigned int or signed int.
8.1	Functions shall have prototype declarations and the prototype shall be visible at both the function definition and call.
8.5	There shall be no definitions of objects or functions in a header file.
8.1	All declarations and definitions of objects or functions at file scope shall have internal linkage unless external linkage is required.
8.12	When an array is declared with external linkage, its size shall be stated explicitly or defined implicitly by initialization.
10.1	The value of an expression of integer type shall not be implicitly converted to a different underlying type if: a. it is not a conversion to a wider integer type of the same signedness, or b. the expression is complex, or c. the expression is not constant and is a function argument, or d. the expression is not constant and is a return expression.
10.3	The value of a complex expression of integer type shall only be cast to a type that is not wider and of the same signedness as the underlying type of the expression.
11.3	A cast should not be performed between a pointer type and an integral type.
11.4	A cast should not be performed between a pointer to object type and a different pointer to object type.
11.5	A cast shall not be performed that removes any const or volatile qualification from the type addressed by a pointer.
12.2	The value of an expression shall be the same under any order of evaluation that the standard permits.
12.4	The right-hand operand of a logical && or | | operator shall not contain side effects.
12.6	The operands of logical operators (&&, | |, and !) should be effectively boolean. Expressions that are effectively boolean should not be used as operands to operators other than (&&, | |, !, =, ==, !=, and ?:).
12.13	The increment (++) and decrement (--) operators should not be mixed with other operators in an expression.
14.3	Before preprocessing, a null statement shall only occur on a line by itself; it may be followed by a comment, provided that the first character following the null statement is a whitespace character.
14.5	The continue statement shall not be used.
14.7	A function shall have a single point of exit at the end of the function.
16.1	Functions shall not be defined with a variable number of arguments.
17.4	Array indexing shall be the only allowed form of pointer arithmetic.
18.4	Unions shall not be used.
19.1	#include statements in a file should only be preceded by other preprocessor directives or comments.
19.1	In the definition of a function-like macro, each instance of a parameter shall be enclosed in parentheses unless it is used as the operand of # or ##.
20.4	Dynamic heap memory allocation shall not be used.
20.9	The input/output library <stdio.h> shall not be used in production code.

Figure 1. MISRA exceptions

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 New project in MCUXpresso sometimes cannot be built

In KDS 3.2, a new project is created by selecting 'File' -> 'New' -> 'Kinetis SDK 2.x Project'. If you select `All drivers` but do not select RTOS, the project will contain the `freertos/portable/heap_4.c file`, which cannot be built. As a workaround, remove the FreeRTOS folder with all content from your project, because these files are only needed for FreeRTOS.

9 Change Log - Peripheral drivers

ADC16

The current ADC16 driver version is 2.0.0

- 2.0.0
  - Initial version

CMP

The current CMP driver version is 2.0.0

- 2.0.0
  - Initial version

CMT

The current CMT driver version is 2.0.1

- 2.0.0
  - Initial version
- 2.0.1

- Changes
  - Added static to global CMT variables.

CRC

The current CRC driver version is 2.0.1

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

DAC

The current DAC driver version is 2.0.1

- 2.0.0
  - Initial version
- 2.0.1
  - Bug fix:
    - Moved the default DAC_Enable(..., true) from the DAC_Init() to the application code to enable the DAC output

DMAMUX

The current DMAMUX driver version is 2.0.2

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

DSPI

The current DSPI driver version is 2.2.0

- 2.1.0
  - New features:
    - Added transfer prefix to transactional APIs
- 2.1.1
  - Bug fix:
    - Set the EOQ (End Of Queue) bit to TRUE for the last transfer in transactional APIs
- 2.1.2
  - Bug fix:
    - The DSPI_MasterTransferBlocking function hangs in corner cases, for example, when bitsPerFrame is 4 and 6 and in the kDSPI_MasterPcsContinuous transfer mode
- 2.1.3
  - Bug Fix:
    - DSPI eDMA driver doesn't support the odd transfer data size and the bitsPerFrame greater than 8.
  - Optimization:
    - Added the #ifndef/#endif to allow users to change the default tx value at compile time.
- 2.1.4
  - Bug fix:

        - DSPI EDMA driver: The DSPI instance that has separated DMA request source can transfer up to 32767 byte data in one DSPI_MasterTransferEDMA() transfer now

- 2.2.0
    - New features:
        - Add gasket feature for the SPI EDMA driver, which reduces one channel used in the EDMA master transfer. With this feature, only two channels are needed. For example, if the gasket feature is supported, one 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);
        - Add dummy data setup API to allow users to configure the dummy data to be transferred.
        - Add new APIs for half-duplex transfer function. Users can send and receive data by one API in polling/interrupt/EDMA way, and users can choose either transmit first or receive first. The PCS pin can be configured as assert status in transimission (between transmit and receive) by setting the isPcsAssertInTransfer to true.

eDMA

The current eDMA driver version is 2.1.2

- 2.0.0
    - Initial version
- 2.0.1
    - Bug fix:
        - Fixed the issue where an eDMA callback does not check a valid status in the EDMA_HandleIRQ API
- 2.0.2
    - Bug fix:
        - Fixed the incorrect minorLoopBytes type definition in the _edma_transfer_config structure. Defined the minorLoopBytes as uint32_t instead of uint16_t
- 2.0.3
    - Bug fix:
        - Fixed the incorrect pubweak IRQHandler name issue, which causes re-definition build errors when a client sets his/her own IRQHandler, by changing the 32-channel IRQHandler name to DriverIRQHandler
- 2.0.4
    - Improvement:
        - Added support for SoCs with multiple eDMA instances.
        - Added the pubweak DriverIRQHandler for the KL28T DMA1 and MCIMX7U5_M4.
- 2.0.5
    - Improvement:
        - Added the pubweak DriverIRQHandler for the K32H844P (16 channels shared).
- 2.1.0
    - Improvement:
        - Changed the EDMA_GetRemainingBytes API to EDMA_GetRemainingMajorLoopCount because of the eDMA IP limitation (see API comments/note for details).
- 2.1.1
    - Improvement:
        - Added documentation of the eDMA data flow when scatter/gather is implemented for the EDMA_HandleIRQ API.
        - Updated and corrected comments in the EDMA_HandleIRQ API and edma_handle_t struct.
- 2.1.2
    - Improvement:
        - Add interface to get next TCD address.
        - Add interface to get the unused TCD number.

EWM

The current EWM driver version is 2.0.1

- 2.0.0
  - Initial version
- 2.0.1
  - Fix EWM_Deinit hardfault issue

Flash

The current Flash driver version is 2.3.1

- 2.0.0
  - Initial version
- 2.1.0
  - New features:
    - Support for the FTFL device in FLASH_Swap API
    - Support for various pflash start addresses
    - Added support for KV58 in the cache clear function
  - Bug fix
    - Compiled execute-in-RAM functions as a PIC binary code for driver use
    - Added missed FlexRAM properties
    - Fixed an unaligned variable issue for the execute-in-RAM function code array
- 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
- 2.3.0
  - 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
- 2.3.1
  - Bug fix
    - Unified Flash IFR design from K3
    - New encoding rule for K3 flash size

FlexIO

The current FlexIO driver version is 2.0.2

- 2.0.0
  - Initial version
- 2.0.1
  - Bug fix:
    - Fix the Dozen mode configuration error in FLEXIO_Init API. For enableInDoze = true, the configuration shoud be 0; for enableInDoze = false, the configuration should be 1
- 2.0.2
  - Improvement:
    - Split the FlexIO component which combines all FlexIO/FlexIO UART/FlexIO I2C/FlexIO I2S drivers into several components: the flexio component, flexio_uart component, flexio_i2c_master component and flexio_i2s component.

FlexIO UART

The current FlexIO UART driver version is 2.1.3

- 2.1.0
  - New features:
    - Added a transfer prefix to the transactional APIs
    - Added the txSize/rxSize parameters to the handle structure to record the transfer size
  - Bug fix
    - Added an error handle to handle the data count zero or data buffer NULL situation
- 2.1.1
  - Bug fix
    - Renamed FLEXIO_UART_StopRingBuffer to FLEXIO_UART_TransferStopRingBuffer to align with the definition in the C file
- 2.1.2
  - Bug fix
    - 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
    - Fix the Dozen mode configuration error in FLEXIO_UART_Init API. For enableInDoze = true, the configuration shoud be 0; for enableInDoze = false, the configuration should be 1.
    - Report error when set baudrate too low and FLEXIO cannot reach that baudrate.
    - Disable FLEXIO_UART receive interrupt instead of disable all NVIC when read data from ring buffer. Because with ring buffer used, receive nonblocking will disable all NVIC interrupts to protect the ring buffer, this will have negative effect to other IPS which are using interrupt.
- 2.1.3
  - Bug fix: the following modiciations were made to support FlexIO using multiple instances.
    - Removed FLEXIO_Reset API in module Init APIs.
    - Updated module Deinit APIs to reset the shifter/timer config instead of disable mode and disable clock.
    - Updated module Enable APIs to only support the enable operation.

FlexIO I2C

The current FlexIO I2C driver version is 2.1.4

- 2.1.0
  - New features:
    - Added a transfer prefix to the transactional APIs
    - Added the txSize/rxSize parameters to the handle structure to record the transfer size
- 2.1.1
  - Bug fix
    - Implemented the FLEXIO_I2C_MasterTransferBlocking API which is defined in the header file but has no implementation in the C file
- 2.1.2
  - Fixed the FLEXIO I2C master can not receive data from I2C slave in high baudrate issue.
  - Fixed the FLEXIO I2C master can not receive NAK when master send non exist addr issue.
  - Fixed the FLEXIO I2C master can not get transfer count successfully issue.
  - Fixed the FLEXIO I2C master can not receive data successfully when send data first issue.
  - Fixed the Dozen mode configuration error in FLEXIO_I2C_MasterInit API. For enableInDoze = true, the configuration shoud 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, and if it calls the FLEXIO_I2C_MasterTransferBlocking API multiple times, the s_flexioHandle/s_flexioIsr/s_flexioType variable cannot be written anymore due to out of range. The following NonBlocking transfer APIs cannot work due to the failed register IRQ.
- 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.

- 2.1.4
  - Bug fix: the following modications were made to support FlexIO using multiple instances.
    - Removed FLEXIO_Reset API in module Init APIs.
    - Updated module Deinit APIs to reset the shifter/timer config instead of disable module and disable clock.
    - Updated module Enable APIs to only support enable operation.

FlexIO SPI

The current FlexIO SPI driver version is 2.1.2

- 2.1.0
  - New features:
    - Added a transfer prefix to the transactional APIs
    - Added the txSize/rxSize parameters to the handle structure to record the transfer size
  - Bug fix
    - Fixed the error register address return for 16-bit data write in the FLEXIO_SPI_GetTxDataRegisterAddress
- 2.1.1
  - Bug fix:
    - Fixed bug when FlexIO SPI transfer data in 16-bit per frame mode with eDMA.
    - Fixed bug when FlexIO SPI transfer data 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 shoud be 0; for enableInDoze = false, the configuration should be 1.
  - Optimization:
    - Add #ifndef/#endif to allow user to change the default tx value at compile time.
- 2.1.2
  - Bug fix: The following modifications to support FlexIO using multiple instances.
    - Removed FLEXIO_Reset API in module Init APIs.
    - Updated module Deinit APIs to reset the shifter/timer config instead of disable module and disable clock.
    - Updated module Enable APIs to only support enable operation.

FlexIO I2S

The current FlexIO I2S driver version is 2.1.3

- 2.1.0
  - New features:
    - Added a transfer prefix to the transactional APIs
    - Added the txSize/rxSize parameters to the handle structure to record the transfer size
- 2.1.1
  - Bug fix
    - Fixed the FlexIO I2S RX data read error and eDMA address error
    - Fixed the FlexIO I2S slave timer comparison setting error
- 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.
- 2.1.3
  - Bug fix: 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.

FlexIO MCU LCD

The current FlexIO MCU LCD driver version is 2.0.1

- 2.0.0
  - Initial version
- 2.0.1

- Bug fix: the following modifications support FlexIO using multiple instances.
  - Remove FLEXIO_Reset API in module Init APIs.
  - Update module Deinit APIs to reset the shifter/timer config instead of disable module and disable clock.
  - Update module Enable APIs to only support enable operation.

FlexIO Camera

The current FlexIO Camera driver version is 2.1.1

- 2.0.0
  - Initial version
- 2.1.0
  - New features
    - Added Transfer prefix in transactional APIs
- 2.1.1
  - Bug fix: the following modifications support FlexIO using multiple instances.
    - Remove FLEXIO_Reset API in module Init APIs.
    - Update module Deinit APIs to reset the shifter/timer config instead of disable module and disable clock.
    - Update module Enable APIs to only support enable operation.

FTM

The current FTM driver version is 2.0.2

- 2.0.0
  - Initial version
- 2.0.1
  - Bug fix:
    - Updated the FTM driver to fix write to ELSA and ELSB bits
    - Set the COMBINE bit before writing to the CnV register
- 2.0.2
  - Feature:
    - Add support to Quad Decoder feature with new APIs:
      - FTM_GetQuadDecoderFlags()
      - FTM_SetQuadDecoderModuloValue()
      - FTM_GetQuadDecoderCounterValue()
      - FTM_ClearQuadDecoderCounterValue()

GPIO

The current GPIO driver version is 2.2.1

- 2.1.0
  - API Interface Change
    - Added "pins" or "pin" to some API names
    - Renamed the "GPIO_PinConfigure" to "GPIO_PinInit"
- 2.1.1
  - API Interface Change
    - Added API to check attribute bytes
- 2.2.1
  - API Interface Change
    - Refined naming of API while keeping all original APIs, marking them as deprecated. The original API will be removed in the next release. The main change is the updated API with prefix of _PinXXX() and _PorortXXX.

I2C

The current I2C driver version is 2.0.5

- 2.0.1
  - New features
    - Added a double buffer enable configuration for SoCs which have the DFEN bit in S2 register
    - Added the flexible transmit/receive buffer size support in I2C_SlaveHandleIRQ
    - Added the start flag clear address match and release bus operation in I2C_SlaveWrite/ReadBlocking API
  - Bug fix:
    - Updated the kI2C_SlaveRepeatedStartEvent to kI2C_SlaveStartEvent
- 2.0.2
  - Bug Fix:
    - Fixed the issue that occurs in master receive and slave transmit mode with no stop flag and master can't start a next transfer because it can't send out restart signal
    - Fixed a data transfer out of order issue which occurs because of a memory barrier
  - New Features:
    - Added an address nak event for the master.
    - Added a general call event for the slave.
- 2.0.3
  - Bug fix
    - Removed enableHighDrive member in the master/slave configuration structure because the user needs to use the DSE bit in the port register to configure the high drive strength capability.
    - Added reset registers operation in I2C_MasterInit and I2C_SlaveInit APIs, and fix the issue that I2C could not switch between master and slave mode.
    - Improved the slave IRQ handler to handle the corner case that stop flag and address match flag come synchronously.
- 2.0.4
  - Bug fix
    - Added proper handle for tranfer config flag kI2C_TransferNoStartFlag to support transmit with kI2C_TransferNoStartFlag flag. Only supports write only or write+read with no start flag. It does not support read only with no start flag.
- 2.0.5
  - Improvement
    - Added the I2C_WATI_TIMEOUT macro to allow users to specify the timeout times for waiting flags in functional API and blocking transfer API.

LLWU

The current LLWU driver version is 2.0.1

- 2.0.0
  - Initial version
- 2.0.1
  - Changes:
    - Updated for KL8x

LMEM

The current LMEM driver version is 2.1.0

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

LPTMR

The current LPTMR driver version is 2.0.1

- 2.0.0
  - Initial version
- 2.0.1
  - Driver update
    - Update the LPTMR driver due to the register LPTMRx_CMR/CNR in some devices that have become 32-bit, so that the updated LPTMR driver supports the 32 bit CNR and CMR register.

LPUART

The current LPUART driver version is 2.2.4

- 2.1.0
  - Updated transactional APIs
- 2.1.1
  - Removed the needless check of event flags and assert in LPUART_RTOS_Receive
  - Wait always for RX event flag in LPUART_RTOS_Receive
- 2.2.0
  - Added seven data bits and MSB support
- 2.2.1
  - Added a separate RX and TX IRQ number support
- 2.2.2
  - Added software reset feature support.
  - Added software reset API to LPUART_Init().
- 2.2.3
  - Changed the parameter type in the LPUART_RTOS_Init() struct rtos_lpuart_config --> lpuart_rtos_config_t.
  - Bug fixed:
    - Disable LPUART receive interrupt instead of disable all NVIC when read data from ring buffer. Because with ring buffer used, receive nonblocking will disable all NVIC interrupts to protect the ring buffer, this will have negative effect to other IPS which are using interrupt.
- 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 reads out (if has FIFO), and all interrupts will not be disabled except 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 less than the RX FIFO size), and data is received and a callback is triggered when data receive has ended.

PDB

The current PDB driver version is 2.0.1

- 2.0.0
  - Initial version
- 2.0.1
  - Changed the PDB register base array to a constant

PIT

The current PIT driver version is 2.0.0

- 2.0.0
  - Initial version

PMC

The current PMC driver version is 2.0.0

- 2.0.0
  - Initial version

PORT

The current PORT driver version is 2.0.2

- 2.0.1
  - Changes:
    - Added "const" in function parameters
    - Updated enumeration variable names
- 2.0.2
  - Changes:
    - Added feature guard macros in the driver

QSPI

The current QSPI driver version is 2.0.2

- 2.0.0
  - Initial version
- 2.0.1
  - New API:
    - QSPI_SetReadArea to set the read area.
  - Bug fix:
    - Fixed the QSPI_UpdateLUT function by updating the first LUT issue.
    - Fixed an issue caused by some functions that have the QSPI0 hardcoded as the base.
- 2.0.2
  - New macro function:
    - Add QSPI_LUT_SEQ() function for users to set LUT table easily.
    - Add LUT command macros for users to easy use.
  - Comment update:
    - Add the comments for the limitation of QSPI_ReadBlocking and QSPI_TransferReceiveBlocking.

RCM

The current RCM driver version is 2.0.1

- 2.0.0
  - Initial version
- 2.0.1
  - [KPSDK-10249] Fixed the kRCM_SourceSw bit shift issue.

RTC

The current RTC driver version is 2.0.0

- 2.0.0
  - Initial version

SAI

The current SAI driver version is 2.1.3

- 2.0.0
  - Initial version
- 2.1.0
  - API name change:
    - SAI_GetSendRemainingBytes -> SAI_GetSentCount

        - SAI_GetReceiveRemainingBytes -> SAI_GetReceivedCount
        - Added "Transfer" prefix to all transactional API names
        - All transactional APIs use a base and a handle as input parameters
        - Unify the parameter names.
    - Bug fix:
        - Fixed the W1C bug while reading TCSR/RCSR registers
        - Fixed the MOE enable flow issue by moving the MOE enable after MICS settings in SAI_TxInit/ SAI_RxInit
- 2.1.1
    - Optimization;
        - Reduced code size while not using transactional APIs
- 2.1.2
    - Bug fix:
        - Add 24-bit support for SAI EDMA transfer. All data shall be 32 bits for send/receive, as EDMA cannot directly handle 3 byte transfer.
- 2.1.3
    - New feature:
        - Add feature to make I2S frame sync length configurable according to bitWidth.
- 2.1.4
    - New feature:
        - Add API to enable/disable auto FIFO error recovery in platforms which support this feature.
        - Add API to set data packing feature in platform which support this feature.

SDHC

The current SDHC driver version is 2.1.6

- 2.1.0
    - New Features:
        - Added a host descriptor to contain the SDHC-related attributes.
    - Bug Fix:
        - Removed the clock auto-gated function because it causes a hardware issue.
    - Changes:
        - Added more SDIO card-related command types.
        - Changed the callback mechanism in the non-blocking transactional APIs.
        - Merged the two ADMA configuration functions into one.
        - Changed the transactional API names.
- 2.1.1
    - Bug Fix:
        - Fixed the compile error when the ADMA1 is enabled.
- 2.1.2
    - Bug fix:
        - Use the function pointer for interrupt handler to reduce code size
        - Bad status bit check behavior when wait for initialization of SD card
        - Add support NON-WORD aligned data size transfer mode for SDIO card
- 2.1.3
    - Modify a definition for compatible with middleware adapter
- 2.1.4
    - New features:
        - Add response error flag to check the response once read from the card
    - Bug fix:
        - Fix clock divider calculate incorrect issue
- 2.1.5
    - New feature:

        - Add non-word aligned data address transfer support in DMA mode
- 2.1.6
    - New feature:
        - Added SDHC_CardDetectByData3 API to support card detect through DATA3.
        - Added host base address/user data parameter for all call back functions.

SDRAMC

The current SDRAMC driver version is 2.1.0

- 2.0.0
    - Initial version
- 2.0.1
    - Changes:
        - Added static to the global SDRAMC variables.
- 2.1.0
    - API change:
        - Changed the status_t SDRAMC_SendCommand() function to void the SDRAMC_SendCommand();

SIM

The current SIM driver version is 2.1.0

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

Smart Card

The current Smart Card driver version is 2.2.0

- 2.1.0
    - Initial version
- 2.1.1
    - New features:
        - Added a default PHY interface selection to Smart Card RTOS drivers (KPSDK-9063)
        - Replaced the smartcard_phy_ncn8025 driver with the smartcard_phy_tda8035
    - Bug fix
        - Fixed the protocol timers activation sequences in the smartcard_emvsim and smartcard_phy_tda8035 drivers during the EMVL1 pre-certification tests (KPSDK-9170, KPSDK-9556)
- 2.1.2
    - Needs to provide 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)
- 2.2.0
    - New features:
        - Updated to use RX/TX FIFO

SMC

The current SMC driver version is 2.0.3

- 2.0.0

        - Initial version
- 2.0.1
    - Changes:
        - Updated for KL8x
- 2.0.2
    - Bug fix:
        - Added DSB before WFI and ISB after WFI
    - Changes:
        - Updated the SMC_SetPowerModeVlpw implementation
- 2.0.3
    - Add APIs SMC_PreEnterStopModes, SMC_PreEnterWaitModes, SMC_PostExitWaitModes, and SMC_PostExitStopModes

SYSMPU

The current SYSMPU driver version is 2.2.0

- 2.0.0
    - Initial version
- 2.1.0
    - API changes:
        - Change the mpu_region_num_t and mpu_master_t to uint32_t
        - Change 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
        - Change the MPU_SetRegionLowMasterAccessRights(), MPU_SetRegionHighMasterAccessRights() to MPU_SetRegionRwxMasterAccessRights(), MPU_SetRegionRwMasterAccessRights()
- 2.1.1
    - Add the feature file macro definition limitation for the MPU_SetRegionRwMasterAccessRights()
- 2.2.0
    - Rename MPU to SYSMPU
    - Changed the macro definition for slave number and fix the get error status calculation

TPM

The current TPM driver version is 2.0.2

- 2.0.0
    - Initial version
- 2.0.1
    - Bug fix:
        - Fixed the TPM_UpdateChnIEdgeLevelSelect ACK wait issue.
        - Fixed the TPM_SetupdualEdgeCapture failure to set the FILTER register.
        - Fixed the TPM_UpdateChnEdgeLevelSelect ACK wait issue.
- 2.0.2
    - Bug fix:
        - Fixed issues in TPM_SetupPwm/TPM_UpdateChnlEdgeLevelSelect /TPM_SetupInputCapture/ TPM_SetupOutputCompare/TPM_SetupDualEdgeCapture functions wait acknowledgement when the channel is disabled.

VREF

The current VREF driver version is 2.1.0

- 2.1.0
    - Added new functions:

- Added VREF_SetTrim2V1Val() and VREF_GetTrim2V1Val() functions to supply 2V1 output mode.

WDOG

The current WDOG driver version is 2.0.0

- 2.0.0
  - Initial version

CLOCK

The current CLOCK driver version is 2.2.2

- 2.0.0
  - Initial version
- 2.1.0
  - Changes:
    - Merged the fsl_mcg and fsl_osc into fsl_clock
- 2.2.0
  - New features:
    - [KPSDK-9157] Updated the CLOCK_SetFeiMode/CLOCK_SetFbiMode/CLOCK_BootToFeiMode() to support set MCG_C4[DMX32]=1 in FEI/FBI modes
  - Bug fix:
    - Updated the IP_CLOCKS array, removed unused gates, and added missing gates
- 2.2.1
  - Added APIs for USB HS PFD clock.
- 2.2.2
  - Bug fix:
    - Fix the issue that MCG could not switch to FEE/FBE/PBE modes when the OSCERCLK clock is not enabled

10 Change Log - Middleware

DMA Manager

The current DMA Manager driver version is 2.1.0

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

EMVL1

The current EMVL1 driver version is 2.1.0

- 2.0.0
  - Initial version
- 2.0.1
  - Bug fix:

        - Fixed low level driver protocol timer failures during EMVL1 pre-certification tests (KPSDK-9556)
        - Fixed incorrect T0 command response that causes long command responses (KPSDK-8707). Command case2, case3, and case4 are affected
- 2.0.2
    - Re-implemented a function for sending commands in T=0
    - Bug Fix:
        - Fixed the incorrect size of response in T=0 (KPSDK-11248)
        - Fixed a problem with command cases 3 in T=1; expected incorrect length of response (KPSDK-11335)
        - Fixed an incorrect length of response in T=1 (KPSDK-11868)
        - Fixed the usage application buffer for the data payload and overhead associated with T=1 protocol (KPSDK-11336)

- 2.1.0
    - Added abort transfer functionality

FatFs

The current FatFs driver version is R0.12c
- R0.11a
    - Added glue functions for low level drivers (SDHC, SDSPI, RAM, and MMC) and modified the diskio.c file
    - Added RTOS wrappers to make FatFs thread-safe. Modified the syscall.c file
    - Renamed ffconf.h file to ffconf_template.h file. Each application should contain its own ffconf.h file
    - Include ffconf.h into diskio.c to enable selection of physical disk from ffconf.h by macro definition
    - Conditional compilation of physical disk interfaces in diskio.c
- R0.12b_rev0
    - Upgrade to version 0.12b
- R0.12c_rev0
    - Upgrade to version 0.12c and apply patches ff_12c_p1.diff and ff_12c_p2.diff

mbedTLS

The current mbedTLS driver version is based on the mbedTLS 2.6.0 released 2017-Aug-10
- 2.2.1
    - New features:
        - Ported mbedTLS 2.2.1 to KSDK 2.0.0
        - Added support for the mmCAU cryptographic acceleration module. Accelerated MD5, SHA, AES, and DES
        - Added support for the 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 the MBEDTLS_GCM_CRYPT_ALT configuration parameter to enable reloading the mbedtls_gcm_crypt_and_tag() function
        - Added the MBEDTLS_ECP_MUL_COMB_ALT to enable an alternate implementation of the ecp_mul_comb() function
        - Added the MBEDTLS_ECP_ADD_ALT configuration parameter to enable reloading the ecp_add() function
        - Added the MBEDTLS_DES_SETKEY_DEC_ALT configuration parameter to enable reloading mbedtls_des_setkey_dec(), mbedtls_des3_set2key_dec(), and mbedtls_des3_set3key_dec() functions
        - Added the MBEDTLS_DES_SETKEY_ENC_ALT configuration parameter to enable reloading mbedtls_des_setkey_enc(), mbedtls_des3_set2key_enc(), and mbedtls_des3_set3key_enc() functions

- Added the MBEDTLS_DES_CRYPT_CBC_ALT configuration parameter to enable reloading the mbedtls_des_crypt_cbc() function
- Added the MBEDTLS_DES3_CRYPT_CBC_ALT configuration parameter to enable reloading the mbedtls_des3_crypt_cbc() function
- Added the MBEDTLS_AES_CRYPT_CBC_ALT configuration parameter to enable reloading the mbedtls_aes_crypt_cbc() function
- Added the MBEDTLS_AES_CRYPT_CTR_ALT configuration parameter to enable reloading the mbedtls_aes_crypt_ctr() function
- Added the MBEDTLS_CCM_CRYPT_ALT configuration parameter to enable reloading mbedtls_ccm_encrypt_and_tag() and mbedtls_ccm_auth_decrypt() functions
- Added the MBEDTLS_MPI_ADD_ABS_ALT configuration parameter to enable reloading the mbedtls_mpi_add_abs() function
- Added the MBEDTLS_MPI_SUB_ABS_ALT configuration parameter to enable reloading the mbedtls_mpi_sub_abs() function
- Added the MBEDTLS_MPI_EXP_MOD_ALT configuration parameter to enable reloading the mbedtls_mpi_exp_mod() function
- Added the MBEDTLS_MPI_MUL_MPI_ALT configuration parameter to enable reloading the mbedtls_mpi_mul_mpi() function
- Added the MBEDTLS_MPI_MOD_MPI_ALT configuration parameter to enable reloading the mbedtls_mpi_mod_mpi() function
- Added the MBEDTLS_MPI_GCD_ALT configuration parameter to enable reloading the mbedtls_mpi_gcd() function
- Added the MBEDTLS_MPI_INV_MOD_ALT configuration parameter to enable reloading the mbedtls_mpi_inv_mod() function
- Added the MBEDTLS_MPI_IS_PRIME_ALT configuration parameter to enable reloading the mbedtls_mpi_is_prime() function
- Added encrypt/decrypt modes to the mbedtls_des_context and the mbedtls_des3_context structure
- Added a carriage return to the mbedtls_printf() in self test functions

- 2.3.0
  - New features:
    - Ported mbedTLS 2.3.0 to KSDK 2.0.0
- 2.3.0_rev1
  - New features:
    - Added support for CAAM driver
    - In LTC-specific wrapper, allocate temporary integers from heap in one large block
- 2.4.2
  - 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
- 2.4.2_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_rev2

    - New features:
        - Added Curve25519 support for CAU3
        - Added MBEDTLS_ECP_MUL_MXZ_ALT configuration parameter enabling overloading of ecp_mul_mxz()
- 2.5.1
    - New features:
        - Ported mbedTLS 2.5.1 to MCUXpresso SDK
- 2.5.1_rev1
    - New features:
        - Added support for DCP driver
- 2.6.0
    - New features:
        - Ported mbedTLS 2.6.0 to MCUXpresso SDK

mmCAU library

The current mmCAU driver version is 2.0.1

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

SDMMC

The current SDMMC driver version is 2.2.1

- 2.1.0
    - Bug fix:
        - Changed the callback mechanism when sending a command
        - Fixed the performance low issue when transferring data
    - Changes:
        - Changed the name of error codes returned by an internal function
        - Merged all host-related attributes into one structure
        - Optimized the function to set a maximum data bus width for the MMC card
- 2.1.1
    - Bug fix:
        - Fixed the block range boundary error when transferring data to the MMC card
        - Fixed the bit mask error in the SD card when switching to a high-speed function
    - Changes:
        - Added an error code to indicate that SDHC ADMA1 transfer type is not supported
        - Optimized the SD card initialization function
- 2.1.2
    - New feature
        - Add fsl_host.h to provide prototype to adapt different controller IPs(SDHC/SDIF)
        - Add adaptor code in sdmmc/port folder to adapt different host controller IPs with different transfer modes(int/polling/freertos). Application inlude different adpator code to make application simplier
        - 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)
        - So this change requires user to include host adaptor code into application. If not, link errors for cannot find the definition of HOST_Init/HOST_Deinit/CardInsertDetect will appear
    - New feature

        - Improve SDMMC to support SD v3.0 and EMMC v5.0
    - Bug fix:
        - Fix wrong comparison between count and length in MMC_ReadBlocks/MMC_WriteBlocks
- 2.1.3
    - Bug fix:
        - Non high-speed SD Card init fail at switch to high speed
    - Miscellaneous changes:
        - Optimized tuning/mmc switch voltage/mmc select power class/mmc select timing function
        - Added strobe dll for mmc HS400 mode
        - Added Delay for SD Card power up
- 2.1.4
    - Miscellaneous changes:
        - Added Host reset function for card re-initialization
        - Added Go_Idle function for SDIO card
        - 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 issue where card cannot detect dynamically
- 2.1.5
    - Fix coverity issue.
    - Fix SD v1.x card write fail issue, it was caused by the block length set error.
    - Improve SDIO card init sequence and add retry option for SDIO_SwitchToHighSpeed function.
- 2.1.6
    - Enhance SD IO default driver strength
- 2.2.0
    - New features
        - Separate the SD/MMC/SDIO init API to xxx_CardInit/xxx_HostInit.
        - Allow user register card detect callback, select card detect type, determine the card detect timeout value.
        - Allow user register the power on/off function, determine the power on/off delay time.
        - SD_Init/SDIO_Init will be deprecated in next version.
        - Add write complete wait operation for MMC_Write to fix command timeout issue.
- 2.2.1
    - Improve mmc boot feature.
    - Keep SD_Init/SDIO_Init function for forward compatibility.

USB stack

The current USB stack version is 1.8.0

- 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
        - vusb_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
- 1.0.1
    - Bug fix
        - Improved the device audio speaker efficiency by changing the transfer mode from interrupt to DMA to eliminate the periodic noise
- 1.1.0
    - Bug fix:
        - Fixed issues in the USB certification
        - Updated the VID and Manufacturer string to NXP Semiconductors
    - New features:
        - Supported classes
            - Printer
        - Examples
            - usb_device_composite_cdc_msc_sdcard
            - usb_device_printer_virtual_plain_text
            - usb_host_printer_plain_text
    - Changes:
        - Renamed example usb_device_composite_cdc_msc to usb_device_composite_cdc_msc_ramdisk
- 1.2.0
    - New features:
        - Supported controllers
            - LPC IP3511 (Full speed, device mode)
- 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
- 1.4.0
    - New features:
        - Examples
            - usb_device_hid_mouse/freertos_static
            - usb_suspend_resume_device_hid_mouse_lite
- 1.5.0
    - New features:
        - Supported controllers
            - OHCI (Full Speed, Host mode)
            - IP3516 (High Speed, Host mode)
            - IP3511 (High Speed, Device mode)
- 1.6.0
    - New features:
        - Supported Device Charger Detect feature on usb_device_hid_mouse
- 1.6.1
    - New features:
        - Change the struct variable address method for device_video_virtual_camera and host_phdc_manager
- 1.6.2
    - New features:
        - Multi instance support
- 1.6.3
    - Bug fix:
        - -IP3511_HS driver control transfer sequence issue, enbale 3511 ip cv test

- 1.7.0
    - New features:
        - USB Type-C PD stack support.
    - Examples:
        - usb_pd
        - usb_pd_battery
        - usb_pd_source_charger

- 1.8.0
    - New features:
        - Examples:
            - usb_device_composite_cdc_vcom_cdc_vcom
            - usb_device_audio_speaker_lpc
            - usb_device_composite_hid_audio_unified
            - usb_device_composite_hid_audio_unified_lpc
            - usb_pd_sink_battery
            - Change usb_pd_battery to usb_pd_charger_battery
    - Bug fix:
        - Code clean up, removed some irrelevant code.

wolfSSL

The current version of wolfSSL is 3.9.8_rev3, based on the release 3.9.8 of wolfSSL

- 3.8.0
    - New features:

        - Added support for the Kinetis LTC hardware acceleration module, which accelerates AES, 3DES, TFM module (modular integer arithmetic), and ECC wolfSSL modules
        - Added support for the Kinetis random number generator modules TRNG and RNGA.
    - Other changes:
        - The Kinetis mmCAU acceleration now uses the "fsl_mmcau.h" file instead of the "cau_api.h" file
        - In DSA, wc_dsaSign() function is updated to repeat the wc_RNG_GenerateBlock() until k is less than q
        - The wolfssl/wolfcrypt/settings.h file is changed to remove the unused macros and add support for the KSDK 2.0.0
        - In the wolfcrypt/src/asn.c file, the ksdk_time(time_t) is changed to external, to be defined by an application
- 3.9.0
    - New Features:
        - Added more LTC public key acceleration (curve25519, ed25519, and RSA4096)
        - FREESCALE_LTC_TFM_RSA_4096_ENABLE macro added to enable RSA4096 on K8x/KL8x LTC
        - LTC_MAX_ECC_BITS increased to 384 to enable ECC-384 curve acceleration on LTC
        - FREESCALE_LTC_SHA added for KL8x SHA-1 and SHA-256 hardware acceleration
    - Other changes:
        - The file wolfssl/wolfcrypt/settings.h is changed to remove unused macros and add support for KSDK 2.0.0
        - LTC public key acceleration is implemented in a separate source file ksdk_port.h and ksdk_port.c

- 3.9.8
    - New features:
        - Added support for AES and SHA acceleration modules of LPC devices. Accelerates AES and SHA wolfSSL modules
    - Bug fixes:
        - Fixed K8x/KL8x LTC RSA sign when FREESCALE_LTC_TFM_RSA_4096_ENABLE macro is enabled
- 3.9.8_rev1
    - New features:
        - Added support for CAAM driver.
        - Added FREESCALE_ALT macros.
- 3.9.8_rev2
    - New features:
        - Added support for CAU3 driver.
- 3.9.8_rev3
    - New features:
        - Added support for DCP driver.

11 Change Log - RTOS

FreeRTOS OS

The current version is FreeRTOS OS 9.0.0. The original package is available at freertos.org.

- 8.2.3
    - New features:
        - Added tickless idle mode support
        - Added a template application for Kinetis Expert (KEx) tool (template_application)
    - Changes:
        - Reduced the folder structure to keep only Kinetis-related information
- 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)
        - Remove configuration option configSYSTICK_USE_LOW_POWER_TIMER. Low power timer is now selected by linking of appropriate file fsl_tickless_lptmr.c
        - Remove configOVERRIDE_DEFAULT_TICK_CONFIGURATION in RVDS port. Use of **attribute**((weak)) is preferred solution. Not same as _weak
- 9.0.0_rev1
    - New features:
        - Enable -flto optimization in GCC by adding **attribute**((used)) for vTaskSwitchContext
        - Enable 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_rev2
    - New features:
        - Enabled MCUXpresso thread aware debugging. Added freertos_tasks_c_additions.h and configINCLUDE_FREERTOS_TASK_C_ADDITIONS_H and configFRTOS_MEMORY_SCHEME macros
- 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.
        - Enable float context saving in IAR for Cortex-A7. Add configUSE_TASK_FPU_SUPPORT macros. Modify port.c and portmacro.h in portable/IAR/ARM_CA9 folder.
    - Other changes:
        - Transform ARM_CM core specific tickless low power support into generic form under FreeRTOS.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number MCUXSDKK80FRN
Revision 0, 11/2017