Document Number: MCUXSDKKL03RN
Rev. 0, 03/2018

MCUXpresso SDK Release Notes Supporting FRDM-KL03Z

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.

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.

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. Easily import your SDK into the new MCUXpresso IDE 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 8.22.1
- 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
FRDM-KL03Z	MKL03Z16VFG4, MKL03Z16VFK4, MKL03Z32CAF4, MKL03Z32VFG4, **MKL03Z32VFK4**, MKL03Z8VFG4, MKL03Z8VFK4

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
Motor Control libraries	<install_dir>/middleware/motor_control
Multicore stack	<install_dir>/middleware/multicore
RTCESL libraries	<install_dir>/middleware/rtcesl
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

Table continues on the next page...

Table 3. Peripheral devices (continued)

	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
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 USBFS controller issue

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

8.3 USB PID issue

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

8.4 IAR build warning for 8.22.1

The IAR 8.22.1 has an integrated CMSIS 5.3.0 core inside while the current SDK package is using CMSIS 5.1.0, so the following build warning pops ups while compiling IAR projects in the package.

Warning: The header file 'cmsis_iar.h' is obsolete and should not be used.

A suitable version is automatically included from the CMSIS-core package. This file will be removed in a future release.

MCUXpresso SDK Release Notes Supporting FRDM-KL03Z

Change Logs

Contents

Contents

1 Driver Change Log

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

COP

The current COP driver version is 2.0.0.

- 2.0.0
  - Initial version

FLASH

The current FLASH driver version is 2.3.1.

- 2.3.1
  - Bug fixes:
    - 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 struct.
- 2.2.0
  - New features:
    - Supports FTFL device in FLASH_Swap API.
    - Supports various pflash start addresses.
    - Added support for KV58 in cache clear function.

        * Added support for device with secondary flash (KW40).
    - Bug fixes:
        * 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.

GPIO

The current driver version is 2.2.1.

- 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 mainin change is update API with prefix of _PinXXX() and _PorortXXX.
- 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 2.0.5.

- 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 tranfer 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/Read-Blocking API.
  - Bug fix:
    - Changed the kI2C_SlaveRepeatedStartEvent to kI2C_SlaveStartEvent.

LLWU

The current LLWU driver version is 2.0.1.

- 2.0.1
  - Miscellaneous changes:
    - Updates for KL8x.
- 2.0.0
  - Initial version.

LPTMR

The current LPTMR driver version is 2.0.1.

- 2.0.1

  - Driver updates:
    * Updated LPTMR driver due to the register LPTMRx_CMR/CNR in some devices becomes 32 bit, so updated LPTMR driver to support the 32 bit CNR and CMR register.
- 2.0.0
  - Initial version.

LPUART

The current LPUART driver version is 2.2.5.

- 2.2.5
  - Not set or clear the TIE/RIE bits when using LPUART_EnableTxDMA() and LPUART_EnableRxDMA().
- 2.2.4
  - Addd hardware flow control function support.
  - Added idle line detected feature in LPUART_TransferNonBlocking function, if an idle line was detected, a callback will be 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 will be read out (if it has FIFO), and all interrupt will not be disabled except the receive data size reach 0.
  - Enable 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), data will be received and a callback will be 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 fixes:
    * Disabled LPUART receive interrupt instead of disable all NVIC when read data from ring buffer. Since ring buffer is used, receive nonblocking disables all NVIC interrupts to protect the ring buffer. This will have negative effect to other IPS which are using interrupt.
- 2.2.2
  - Added software reset feature support.
  - Added software reset API to LPUART_Init().
- 2.2.1
  - Added seprarte RX,TX irq number support.
- 2.2.0
  - Added seven 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
  - Updated transactional APIs.

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.

- 2.0.2:
  - Miscellaneous changes:
    - Added feature guard macros in the driver.
- 2.0.1:
  - Miscellaneous changes:
    - Added "const" in function parameter.
    - Updated some enumeration variables' names.

RCM

The current RCM driver version is 2.0.1.

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

RTC

The current RTC driver version is 2.0.0.

- 2.0.0
  - Initial version.

SIM

The current SIM driver version is 2.1.0.

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

SMC

The current SMC driver version is 2.0.3.

- 2.0.3
  - Added APIs SMC_PreEnterStopModes, SMC_PreEnterWaitModes, SMC_PostExitWaitModes, and SMC_PostExitStopModes.
- 2.0.2
  - Bug fixes:
    - Added DSB before WFI, add ISB after WFI.
  - Miscellaneous changes:
    - Updated SMC_SetPowerModeVlpw implementation.
- 2.0.1
  - Misc Changes:
    - Update for KL8x.
- 2.0.0
  - Initial version.

SPI

The current SPI driver version is 2.0.4.

- 2.0.4
  - New features:
    - Supports 3-wire mode for SPI driver. Added new API SPI_SetPinMode() to control the transfer direction of the single wire. For master instances, MOSI is selected as I/O pin. For slave instances, MISO is selected as I/O pin.
    - Added dummy data setup API to allow users to configure the dummy data to be transferred.
- 2.0.3
  - Bug fix:
    - Fixed the potential interrupt race condition in high baudrate when call API SPI_MasterTransferNonBlocking.
- 2.0.2
  - New features:
    - Allowed user to set the transfer size for SPI_TransferNoBlocking non-integer times of watermark.
    - Allowed user to define the dummy data, users only need to define the macro SPI_DUMMYDATA in applications.
- 2.0.1
  - Bug fixes:
    - Fixed SPI_Enable function parameter error.
    - Set the s_dummy variable as static variable in fsl_spi_dma.c
  - Optimazation:
    - Optimized the code size while not use transactional API.

        * Improved performance in polling method.
        * Added #ifndef/#endif to allow user to change the default tx value at compile time.
- 2.0.0
  - Initial version.

TPM

The current TPM driver version is 2.0.2.

- 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:
    * Fix TPM_UpdateChnIEdgeLevelSelect ACK wait issue.
    * Fix TPM_SetupdualEdgeCapture can not set FILTER register issue.
    * Fix TPM_UpdateChnEdgeLevelSelect ACK wait issue.
- 2.0.0
  - Initial version.

VREF

The current VREF driver version is 2.1.0.

- 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.

CLOCK

The current CLOCK driver version is 2.1.1

- 2.0.0
  - Initial version.
- 2.1.0
  - Other changes:
    * Mergee fsl_mcglite and fsl_osc into fsl_clock.
- 2.1.1
  - Bug fixes:

* Updated IP_CLOCKS array, remove unused gates and add missing gates.

2 Middleware Change Log

FatFs

The current version is FatFs R0.12c.

- R0.12c_rev0
  - Upgraded to version 0.12c and apply 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 selection of physical disk from ffconf.h by macro definition.
  - Conditional compilation of physical disk interfaces in diskio.c.

SDMMC

The current driver version is 2.2.4.

- 2.2.4
  - Bug fix:
    - Fixed DDR mode data sequence mess issue which caused by NIBBLE_POS.
  - New features:
    - Increased g_sdmmc 512byte to improve the performance when application use a non-word align data buffer address.
    - Used OCR access mode bits to determine the mmccard high capacity flag.
    - Enabled auto cmd12 for SD read/write.
    - Disabled DDR mode frequency multiply by 2.
- 2.2.3
  - Bug fix:
    - Added reponse check for send operation condition command. If not checked, the card may occasionally init fail.
- 2.2.2
  - Moved set card detect priority operation before enable IRQ.
- 2.2.1
  - New features:
    - Improved MMC Boot feature.
    - Keep SD_Init/SDIO_Init function for forward compatibility.
- 2.2.0
  - New features:
    - Separated the SD/MMC/SDIO init API to xxx_CardInit/xxx_HostInit.

        * Allowed user register card detect callback, select card detect type, and determine the card detect timeout value.
        * Allowed user register the power on/off function, and determine the power on/off delay time.
        * SD_Init/SDIO_Init will be deprecated in the next version.
        * Added write complete wait operation for MMC_Write to fix command timeout issue.
- 2.1.6
  - Enhanced SD IO default driver strength.
- 2.1.5
  - Fixed coverity issue.
  - Fixed SD v1.x card write fail issue. It was caused by the block length set error.
  - Improved SDIO card init sequence and add retry option for SDIO_SwitchToHighSpeed function.
- 2.1.4
  - Miscellaneous:
    * 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 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 wrong comparison between count and length in MMC_ReadBlocks/MMC_WriteBlocks.
- 2.1.1
  - Bug fix:
    - Fixed the block range boundary error when transferring data to MMC card.
    - Fixed the bit mask error in the SD card switch to high speed function.
  - Other changes:
    - Added error code to indicate that SDHC ADMA1 transfer type is not supported yet.
    - Optimized the SD card initialization function.
- 2.1.0
  - Bug fix:
    - Change the callback mechanism when sending a command.
    - Fix the performance low issue when transferring data.
  - Other changes:
    - Changed the name of some error codes returned by internal function.
    - Merged all host related attributes to one structure.
    - Optimize the function of setting maximum data bus width for MMC card.

3 RTOS Change Log

FreeRTOS

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

- 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.
- 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 preffered solution. Not same as _week!
- 8.2.3
  - New features:
    - Tickles 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

Document Number MCUXSDKKL03RN
Revision 0, 03/2018