NXP Semiconductors
Release Notes

Document Number: MCUXSDKKL46RN
Rev. 0, 05/2018

MCUXpresso SDK Release Notes Supporting FRDM-KL46Z

Contents

1 Overview....1
2 MCUXpresso SDK.... 1
3 Development tools.... 2
4 Supported development systems.... 2
5 Release contents.... 2
6 MCUXpresso SDK release package.... 3
7 MISRA compliance.... 6
8 Known issues.... 8

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. 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:

- IAR Embedded Workbench for Arm version 8.22.2
- MDK-Arm Microcontroller Development Kit (Keil)® 5.24a
- Makefiles support with GCC revision 7-2017-q4-major from Arm Embedded
- MCUXpresso IDE v10.2.0

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-KL46Z	MKL46Z128VLH4, MKL46Z128VLL4, MKL46Z128VMC4, MKL46Z256VLH4, **MKL46Z256VLL4**, MKL46Z256VMC4, MKL46Z256VMP4

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 MCUXSDKUSBSUG) 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 2012 rules with the following exceptions.

Table 4. MISRA exceptions

Exception Rules	Description
Directive 4.4	Sections of code should not be commented out.
Directive 4.5	Identifiers in the same name space with overlapping visibility should be typographically unambiguous.
Directive 4.6	Typedef that indicate size and signedness should be used in place of the basic numerical type.
Directive 4.8	If a pointer to a structure or union is never dereferenced within a transaction unit then the implementation of the object should hidden.
Directive 4.9	A function should be used in preference to a function like macro where they are interchangeable.
Directive 4.10	Precautions shall be taken in order to prevent the contents of a header file being included more than once.
Directive 4.11	The validity of values passed to library functions shall be checked.
Rule 2.3	A project should not contain unused type declarations.
Rule 2.4	A project should not contain unused tag declarations.
Rule 2.5	A project should not contain unused macro declarations.
Rule 2.7	There should be no unused parameters in functions.
Rule 3.1	The character sequences /* and // shall not be used within a comment.
Rule 5.1	External identifiers shall distinct.
Rule 5.3	A identifier declared in an inner scope shall not hide an identifier declared in an outer scope.
Rule 5.7	A tag name shall be a unique identifier.
Rule 5.9	Identifiers that define objects or functions with external linkage shall be unique.
Rule 8.13	A pointer should point to a const-qualified type whenever possible.

Table continues on the next page...

Table 4. MISRA exceptions (continued)

Rule 8.3	All declarations of an object or function shall use the same names and type qualifiers.
Rule 8.6	An identifier with external linage shall have exactly one external definition.
Rule 8.7	Octal constants shall not be used.
Rule 8.9	A object should be defined at block scope if its identified only appears in a single function.
Rule 10.1	Operands shall not be of an inappropriate essential type.
Rule 10.3	The value of an expression shall not be assigned to an object with a narrower essential type of a different essential type category.
Rule 10.4	Both operands of an operator in which the usual arithmetic conversions are performed shall have the same essential type category.
Rule 10.5	The value of an expression should not be cast to an inappropriate essential type.
Rule 10.6	The value of a composite expression shall not be assigned to an object with wider essential type.
Rule 10.7	If a composite expression is used as one operand of an operator in which the usual arithmetic conversions are performed then the other operand shall not have wider essential type.
Rule 10.8	The value of a composite expression shall not be cast to a different essential type category or a wider essential type.
Rule 11.1	Conversions shall not be performed between a pointer to a function and any other type.
Rule 11.3	A case shall not be performed between a pointer to object type and a pointer to a different object type.
Rule 11.4	A conversion should not be performed between a pointer to object and an integer type.
Rule 11.5	A conversion should not be performed from pointer to void into pointer to object.
Rule 11.6	A cast shall not be performed between pointer to void and an arithmetic type.
Rule 12.1	The precedence of operators within expressions should be made explicit.
Rule 12.2	The right hand operator of a shift operator shall lie in the range zero to one less than the width in bits of the essential type of the left hand operand.
Rule 13.3	A full expression containing an increment(++) or decrement(--) operator should have no other potential side effects other than that caused by the increment or decrement operator.
Rule 13.5	The right hand operand of a logical && or || operator shall not contain persistent side effects.
Rule 14.2	A for loop shall be well formed.

Table continues on the next page...

Table 4. MISRA exceptions (continued)

Rule 14.4	The controlling expressions of an statement and the controlling expression of an iteration-statement shall have essentially Boolean type.
Rule 15.5	A function should have a single point of exit at the end.
Rule 16.1	All switch statements shall be well-formed.
Rule 17.7	The feature of <stdarg.h> shall not be used.
Rule 18.4	The +,-,+=and -=operators should not be applied to an expression of pointer type.
Rule 19.2	The union keyword should not be used.
Rule 20.1	#include directives should only be preceded by preprocessor directives or comments.
Rule 20.10	The #and ## preprocessor operators should not be used.
Rule 21.1	#define and #undef shall not be used on a reserved identifier or reserved macro name.

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.

MCUXpresso SDK Release Notes Supporting FRDM-KL46Z

Change Logs

Contents

Driver Change Log 1

ADC16 . . . 1
CMP . . . 1
COP . . . 1
DAC . . . 1
DMA . . . 1
DMAMUX . . . 2
FLASH . . . 2
GPIO . . . 3
I2C . . . 3
LLWU . . . 4
LPSCI . . . 4
LPTMR . . . 5
PIT . . . 5
PMC . . . 5
PORT . . . 5
RCM . . . 6
RTC . . . 6
SAI . . . 6
SIM . . . 7
SLCD . . . 7

Contents

Title	Page Number
SMC	7
SPI	8
TPM	9
TSI_V2	9
TSI_V4	9
TSI_V5	10
UART	10
CLOCK	11
Middleware Change Log	**12**
DMA_MANAGER	12
FatFs for MCUXpresso SDK	12
SDMMC	12
USB stack for MCUXpresso SDK	14
RTOS Change Log	**19**
FreeRTOS for MCUXpresso SDK	19

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.

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.

DMA

The current DMA driver version is 2.0.1.

- 2.0.1
  - Bug fix:
    - Fixed the DMA driver build fail due to MISRA C 2004 rule 12.5 by adding parenthesis.
- 2.0.0

  - Initial version.

DMAMUX

The current DMAMUX driver version is 2.0.2.

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

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 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 2.0.5.

- 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.

LPSCI

The current LPSCI driver version is 2.0.4.

- 2.0.4
  - Added idle line detected feature in LPSCI_TransferNonBlocking function. If an idle line was detected, a callback is triggered with status kStatus_LPSCI_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 it has FIFO) is read out, and all interrupts will not be disabled, except if the receive data size reaches 0.
- 2.0.3
  - Changed parameter type in LPSCI_RTOS_Init() struct rtos_lpsci_config –> lpsci_rtos_config-_t.

  - Bug fix:
    - Disabled LPSCI receives 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 on other IPS which are using interrupt.
- 2.0.2
  - Added RX framing error and parity error status check when using interrupt transfer.
- 2.0.1
  - Updated baudrate code. Before setting baudrate, check whether the value is valid or not, if not valid, do not change baudrate and return values.
  - Removed needless check of event flags and assert in LPSCI_RTOS_Receive.
  - Always wait for RX event flag in LPSCI_RTOS_Receive.
- 2.0.0
  - Initial version.

LPTMR

The current LPTMR driver version is 2.0.1.

- 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.

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.

- 2.0.2

  - 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.

SAI

The current SAI driver version is 2.1.4.

-2.1.4

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

2.1.3

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

2.1.2

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

2.1.1

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

2.1.0

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

2.0.0

- Initial version.

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.

SLCD

The current SLCD driver version is 2.0.1.

- 2.0.1
  - Bug fix:
    - Changed the Blink mode start setting flow.
  - Miscellaneous changes:
    - Added static to SLCD global variables.
- 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 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.

SPI

The current SPI driver version is 2.0.4.

- 2.0.4
  - New feature:
    - Supports 3-wire mode for SPI driver. Added new API SPI_SetPinMode() to control the transfer direction of the single wire. For master instance, MOSI is selected as I/O pin. For slave instance, 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 feature:
    - Allows user to set the transfer size for SPI_TransferNoBlocking non-integer times of watermark.
    - Allows user to define the dummy data, users only need to define the macro SPI_DUMMYDATA in applications.
- 2.0.1
  - Bug fix:
    - 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.

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 2.0.0.

- 2.0.0
  - Initial version.

UART

The current UART driver version is 2.1.5.

- 2.1.5
  - Added hardware flow control function support.
  - Added idle line detected feature in UART_TransferNonBlocking function. If an idle line is detected, a callback is triggered with status kStatus_UART_IdleLineDetected returned. This feature may be useful when the number of received bytes is less than the expected receive data size. Before triggering the callback, data in the FIFO is read out (if it has FIFO), and all interrupts are not 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 not be bigger than the RX FIFO size). Data is then received and a callback is triggered when data receive ends.
- 2.1.4
  - Changed parameter type in UART_RTOS_Init() struct rtos_uart_config –> uart_rtos_config_t.
  - Bug fixed:
    * Disabled UART receive interrupt instead of disable all NVIC when read data from ring buffer. Because with 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 interrupt.
- 2.1.3
  - Added RX framing error and parity error status check when use interrupt transfer.
- 2.1.2
  - Fixed baud rate fine adjust bug to make the computed baud rate more accurately.
- 2.1.1
  - Removed needless check of event flags and assert in UART_RTOS_Receive.
  - Waited always for RX event flag in UART_RTOS_Receive.
- 2.1.0
  - Added transactional API.
- 2.0.0
  - Initial version.

CLOCK

The current CLOCK driver version is 2.2.1.

- 2.2.1
  - Bug fixes:
    * Fixed issue where MCG could not switch to FEE/FBE/PBE modes when OSCERCLK clock not enabled.
- 2.2.0
  - New features:
    * [KPSDK-9157] Updated CLOCK_SetFeiMode/CLOCK_SetFbiMode/ CLOCK_BootTo-FeiMode() to support set MCG_C4[DMX32]=1 in FEI/FBI modes.
  - Bug fixes:
    * Updated IP_CLOCKS array, removed 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.

FatFs for MCUXpresso SDK

The current version is FatFs R0.13a_rev0.

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

SDMMC

The current driver version is 2.2.4.

- 2.2.4
  - Bug fix:
    - Fixed DDR mode data sequence miss issue, which is caused by NIBBLE_POS.
  - New features:
    - Increased g_sdmmc 512byte to improve the performance when application use a non-word align data buffer address.
    - Used OCR access mode bits to determine the mmccard high capacity flag.

        - Enabled auto cmd12 for SD read/write.
        - Disabled DDR mode frequency multiply by 2.
- 2.2.3
  - Bug fixes:
    - 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 fixes:
    - 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 fixes:
    * Fixed wrong comparison between count and length in MMC_ReadBlocks/MMC_WriteBlocks.
- 2.1.1
  - Bug fixes:
    * 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 fixes:
    * Changed the callback mechanism when sending a command.
    * Fixed 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.

USB stack for MCUXpresso SDK

The current version of USB stack is 2.0.1.

- 2.0.1
  - Bug fixes:
    * Fixed some USB issues.
    * Changed the audio codec interfaces.
- 2.0.0
  - New features:
    * PTN5110N support.

    - Bug fixes:
        * 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 fixes:
        * 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 fixes:

- Code cleaned 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 fixes: -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 fixes:
  - 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 fixes:
  - 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 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 MCUXSDKKL46RN
Revision 0, 05/2018