NXP Semiconductors
Release Notes

Document identifier: MCUSDKRT595RN
Rev. 0, July 2019

MCUXpresso SDK Release Notes Supporting MIMXRT595-EVK

Contents

Chapter 1
Overview

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

For more details about MCUXpresso SDK, see the MCUXpresso SDK homepage MCUXpresso-SDK: Software Development Kit.

NOTE

See the attached Change Logs section at the end of this document to reference the device-specific driver logs, middleware logs, and RTOS log.

Chapter 2
MCUXpresso SDK

As part of the MCUXpresso software and tools, MCUXpresso SDK is the evolution of Kinetis SDK v2.x.x, includes support for both LPC and i.MX System-on-Chips (SoC). The same drivers, APIs, and middleware are still available with support for Kinetis, LPC, and i.MX silicon. The MCUXpresso SDK adds support for the MCUXpresso IDE, an Eclipse-based toolchain that works with all MCUXpresso SDKs. Easily import your SDK into the new toolchain to have access to all of the available components, examples, and demos for your target silicon. In addition to the MCUXpresso IDE, support for the MCUXpresso Config Tools allows for easy cloning of existing SDK examples and demos, allowing users to easily leverage the existing software examples provided by the SDK for their own projects.

NOTE

In order to maintain compatibility with legacy Freescale code, the filenames and source code in MCUXpresso SDK containing the legacy Freescale prefix 'FSL' has been left as is. The 'FSL' prefix has been redefined as the NXP Foundation Software Library.

Chapter 3
Development tools

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

- IAR Embedded Workbench for Arm version 8.32.3
- Makefiles support with GCC revision 8-2018-q4 from Arm Embedded

Chapter 4
Supported development systems

This release supports boards and devices listed in this table. Boards and devices in boldface were tested in this release:

Table 1. Supported MCU devices and development boards

Development boards	MCU devices
MIMXRT595-EVK	PIMXRT595SFFOA

Chapter 5
Release contents

This table provides an overview of the MCUXpresso SDK release package contents and locations.

Table 2. Release contents

Deliverable	**Location**
Boards	<install_dir>/boards
Demo applications	<install_dir>/boards/<board_name>/demo_apps
USB demo applications	<install_dir>/boards/<board_name>/usb_examples
VGLige examples	<install_dir>/boards/<board_name>/vglite_examples
Driver examples	<install_dir>/boards/<board_name>/driver_examples
DSP examples	<install_dir>/boards/<board_name>/dsp_examples
Cortex Microcontroller Software Interface Standard (CMSIS) driver examples	<install_dir>/boards/<board_name>/cmsis_driver_examples
Jpeg examples	<install_dir>/boards/<board_name>/jpeg_examples
LittlevGL examples	<install_dir>/boards/<board_name>/littlevgl_examples
FatFS examples	<install_dir>/boards/<board_name>/fatfs_examples
emWin examples	<install_dir>/boards/<board_name>/emwin_examples
RTOS examples	<install_dir>/boards/<board_name>/rtos_examples
mbed TLS examples	<install_dir>/boards/<board_name>/mbedtls_examples
Trustzone examples	<install_dir>/boards/<board_name>/trustzone_examples
Documentation	<install_dir>/docs
USB Documentation	<install_dir>/docs/usb
Middleware	<install_dir>/middleware
emWin library	<install_dir>/middleware/emWin
LibJpeg	<install_dir>/libjpeg
LittlevGL library	<install_dir>/middleware/littlevgl
mbed TLS	<install_dir>/middleware/mbedtls
FatFS stack	<install_dir>/middleware/fatfs
SDMMC card driver	<install_dir>/middleware/sdmmc
USB stack	<install_dir>/middleware/usb
VGLige Graphic library	<install_dir>/middleware/vglite
Driver, SoC header files, extension header files and feature header files, utilities	<install_dir>/devices/<device_name>
CMSIS Arm Cortex®-M header files, DSP library source	<install_dir>/CMSIS

Table continues on the next page...

Table 2. Release contents (continued)

Peripheral Drivers	<install_dir>/devices/<device_name>/drivers
CMSIS drivers	<install_dir>/devices/<device_name>/cmsis_drivers
Utilities such as debug console	<install_dir>/devices/<device_name>/utilities
RTOS Kernel Code	<install_dir>/rtos
Tools	<install_dir>/tools

Chapter 6
MCUXpresso SDK release package

The MCUXpresso SDK release package contents are aligned with the silicon subfamily it supports. This includes the boards, CMSIS, devices, documentation, middleware, and RTOS support.

6.1 Device support

The device folder contains all available software enablement for the specific System-on-Chip (SoC) subfamily. This folder includes clock-specific implementation, device register header file, device register feature header file, CMSIS derived device SVD, and the system configuration source files. Included with the standard SoC support are folders containing peripheral drivers, toolchain support, and a simple debug console.

The device-specific header files provide a direct access to the MCU peripheral registers. The device header file provides an overall SoC memory mapped register definition. In addition to the overall device memory mapped header file, the MCUXpresso SDK also includes the feature header file for each peripheral instantiated on the SoC.

The toolchain folder contains the startup code and linker files for each supported toolchain. The startup code is a CMSIScompliant startup that efficiently transfers the code execution to the main() function.

6.1.1 Board support

The boards folder provides the board-specific demo applications, driver examples, RTOS, and middleware examples.

6.1.2 Demo applications and other examples

The demo applications demonstrate the usage of the peripheral drivers to achieve a system level solution. Each demo application contains a readme file that describes the operation of the demo and required setup steps.

The driver examples demonstrate the capabilities of the peripheral drivers. Each example implements a common use case to help demonstrate the driver functionality.

6.2 Middleware

6.2.1 USB stack

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

6.2.1.1 Peripheral devices tested with USB Host stack

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

Table 3. Peripheral devices

Device type	Device

Table continues on the next page...

Table 3. Peripheral devices (continued)

USB HUB	BELKIN F5U233 BELKIN F5U304 BELKIN F5U307 BELKIN F4U040 UNITEK Y-2151 Z-TEK ZK032A HYUNDAI HY-HB608
USB flash drive	ADATA C008 32 GB ADATA S102 8 G ADATA S102 16 G Verbatim STORE N GO USB Device 8 G Kingston DataTraveler DT101 G2 SanDisk Cruzer Blade 8 GB Unisplendour 1 G Imation 2 GB V-mux 2 GB Sanmina-SCI 128 M Corporate Express 1 G TOSHIBA THUHYBS-008G 8 G Transcend JF700 8 G Netac U903 16 G SSK SFD205 8 GB Rex 4 GB SAMSUNG USB3.0 16GB
USB card reader/adapter	SSK TF adapter Kawau Multi Card Reader Kawau TF adapter Kawau SDHC card

Table continues on the next page...

Table 3. Peripheral devices (continued)

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

6.2.2 File system

The FatFs file system is integrated with MCUXpresso SDK and can be used to access either the SD card or the USB memory stick when the SD card driver or the USB Mass Storage Device class implementation is used.

6.2.3 RTOS

The MCUXpresso SDK is integrated with FreeRTOS OS.

6.2.4 CMSIS

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

6.2.5 SDMMC

The SDMMC software is integrated with MCUXpresso SDK to support SD/MMC/SDIO standard specification. This also includes a host adapter layer for bare-metal/RTOS applications.

6.2.6 VGLite: Graphic library

This software package contains Vivante's platform-independent pre-built VGLite Graphics libraries, and the header files for application to access the libraries.

6.2.7 emWin

The MCUXpresso SDK is pre-integrated with the SEGGER emWin GUIBuilder.

6.2.8 Other middleware

Optional middleware packages can be included in the release based on the user selection. See *<install_dir>/SW-Content-Register.txt* for a list of components and associated licenses.

Chapter 7 MISRA compliance

All MCUXpresso SDK drivers 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.
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.

Table continues on the next page...

Table 4. MISRA exceptions (continued)

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 17.1	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.

Chapter 8 Known issues

8.1 Maximum file path length in Windows 7® operating system

Windows 7 operating system imposes a 260 character maximum length for file paths. When installing the MCUXpresso SDK, place it in a directory close to the root to prevent file paths from exceeding the maximum character length specified by the Windows operating system. The recommended location is the `C:\nxp` folder.

8.2 USB issue

1. The host examples cannot detect disconnection of the high-speed device.
2. There is a board level design issue for RT500 Rev A. All USB host examples cannot currently run. If running USB device examples, make sure extra power (J39) is not used to apply power to board.

MCUXpresso SDK Release Notes Supporting i.MX RT595S

NXP Semiconductors

Document Number: MCUXSDKAPIRM
Rev. 0
Jul 2019

Contents

Chapter **Driver Change Log**

Contents

Section Number | Title | Page Number

Contents

Section Number | Title | Page Number

Chapter Middleware Change Log

Chapter RTOS Change Log

Chapter 1
Driver Change Log

1.1 CLOCK

The current CLOCK driver version is 2.2.0.

- 2.2.0
  - New feature
    - Adding Deinit PLL&PFD API.
  - API change
    - Add delay_us parameter in CLOCK_EnableSysOscClk()
- 2.1.0
  - New feature
    - Adding new API CLOCK_DelayAtLeastUs() implemented by DWT to allow users set delay in unit of microsecond.
- 2.0.1
  - Update clock_attach_id_t elements, removing the FRG(Fractional Generator) clock source selection from CLOCK_AttachClk.
  - Users need call CLOCK_SetFRGClock to set FRG clock source.
- 2.0.0
  - initial version.

1.2 POWER

The current POWER driver version is 2.0.1.

- 2.0.1
  - Add POWER_UpdateOscSettlingTime() API to set on-board system osc settling time.
- 2.0.0
  - initial version.

1.3 RESET

The current RESET driver version is 2.0.0.

- 2.0.0
  - initial version.

1.4 ACMP

The current ACMP driver version is 2.0.4.

- 2.0.4
  - Bug fix:
    - Avoided changing w1c bit in ACMP_SetRoundRobinPreState().

- 2.0.3
  - Added feature functions for different power domain's usage (1.8 V and 3 V). These functions are first enabled in ULP1. They are about:
    - ACMP_EnableLinkToDAC()
    - ACMP_SetDiscreteModeConfig()
    - ACMP_GetDefaultDiscreteModeConfig()
- 2.0.2
  - Coding style changes:
    - Changed coding style of peripheral base address from "s_acmpBases" to "s_acmpBase".
- 2.0.1
  - Bug fix:
    - Fixed bug regarding the function "ACMP_SetRoundRobinConfig". It will not continue execution but returns directly after disabling round robin mode.

1.5 CACHE

The current CACHE driver version is 2.0.0.

- 2.0.0
  - Initial version.

1.6 COMMON

The current COMMON driver version is 2.1.3.

- 2.1.3
  - MISRA C-2012 issue fixed.
    - Fix the rule: rule-10.3.
- 2.1.2
  - Add SUPPRESS_FALL_THROUGH_WARNING() macro for the usage of suppressing fallthrough warning.
- 2.1.1
  - fix bug
    - Deleted and optimized repeated macro.
- 2.1.0
  - New features:
    - Added IRQ operation for XCC toolchain.
    - Added group IDs for newly supported drivers.
- 2.0.2
  - MISRA C-2012 issue fixed.
    - Fix the rule: rule-10.4.
- 2.0.1
  - Removed the implementation of LPC8XX Enable/DisableDeepSleepIRQ() function.
  - Added new feature macro switch "FSL_FEATURE_HAS_NO_NONCACHEABLE_SECTION" for specific SoCs which have no noncacheable sections, that helps avoid an unnecessary complex in link file and the startup file.
  - Updated the align(x) to **attribute**(aligned(x)) to support MDK v6 armclang compiler.

- 2.0.0
  - Initial version.

1.7 CTIMER

The current CTimer driver version is 2.0.2.

- 2.0.2
  - Added new API "CTIMER_GetTimerCountValue" to get the current timer count value.
  - Added control macro to enable/disable the RESET and CLOCK code in current driver.
  - Added new feature macro to update the API of CTimer driver for lpc8n04.
- 2.0.1
  - API Interface Change Added CTIMER_SetupPwmPeriod and CTIMER_UpdatePwmPulsePeriod API. These two APIs can set up the right PWM with high resolution.
- 2.0.0
  - Initial version.

1.8 CRC

The current CRC driver version is 2.0.1.

- 2.0.1
  - Fixed KPSDK-13362. MDK compiler issue when writing to WR_DATA with -O3 optimize for time.
- 2.0.0
  - Initial version.

1.9 DMA

The current DMA driver version is 2.4.0.

- 2.4.0
  - Improvements:
    - Added new apis DMA_LoadChannelDescriptor/DMA_ChannelIsBusy to support polling transfer case.
  - Bug fix:
    - Add address alignment check for descriptor source and destination address.
    - Add DMA_ALLOCATE_DATA_TRANSFER_BUFFER for application buffer allocation.
- 2.3.0
  - Bug fix:
    - Removed DMA_HandleIRQ prototype definition from header file.
    - Added DMA_IRQHandle prototype definition in header file.
- 2.2.5
  - Improvements:
    - Added new API DMA_SetupChannelDescriptor to support configure wrap descriptor.
    - Added wrap support in function DMA_SubmitChannelTransfer.
- 2.2.4

  - Bug fix:
    - Fixed the macro DMA_CHANNEL_CFER use wrong parameter to calculate DSTINC issue.
- 2.2.3
  - Bug fix:
    - Improved DMA driver Deinit function for correct logic order.
  - Improvement:
    - Added API DMA_SubmitChannelTransferParameter to support create head descriptor directly.
    - Added API DMA_SubmitChannelDescriptor to support ping pong transfer.
    - Added macro DMA_ALLOCATE_HEAD_DESCRIPTOR/DMA_ALLOCATE_LINK_DESCRIPTOR to simplify DMA descriptor allocation.
- 2.2.2
  - Bug fix:
    - Do not use software trigger when hardware trigger is enabled.
- 2.2.1
  - Bug fix:
    - Fixed coverity issue.
- 2.2.0
  - Improvements:
    - Changed API DMA_SetupDMADescriptor to non-static.
    - Marked below API as deprecated. DMA_PrepareTransfer. DMA_Submit transfer.
    - Added below new API: DMA_SetChannelConfig. DMA_PrepareChannelTransfer. DMA_InstallDescriptorMemory. DMA_SubmitChannelTransfer. DMA_SetChannelConfigValid. DMA_DoChannelSoftwareTrigger. DMA_LoadChannelTransferConfig.
- 2.0.1
  - Improvement:
    - Added volatile for DMA descriptor member xfercfg to avoid optimization.
- 2.0.0
  - Initial version.

1.10 DMIC

The current DMIC driver version is 2.2.0.

- 2.2.0
  - Bug fix:
    - Correct the usage of feature FSL_FEATURE_DMIC_IO_HAS_NO_BYPASS.
- 2.1.1
  - Improvements:
    - Add feature FSL_FEATURE_DMIC_HAS_NO_IOCFG for IOCFG register.
- 2.1.0
  - New feature
    - Added API DMIC_EnbleChannelInterrupt/DMIC_EnbleChannelDma to replace API DMIC_SetOperationMode.
    - Added API DMIC_SetIOCFG and mark DMIC_ConfigIO as deprecated.

    * Added API DMIC_EnableChannelSignExtend to support sign extend feature.
- 2.0.5
  - Improvements:
    * Changed some parameters value of DMIC_FifoChannel API, such as enable, resetn, and trig_level. This is not possible for the current code logic, so it improves the DMIC_FifoChannel logic and fixes incorrect math logic.
- 2.0.4
  - Bug fix:
    * Fixed DMIC DMA driver (ver2.0.3) that does not support call DMIC_TransferReceiveDMA in DMA callback, which is supported before 2.0.3, but calling DMIC_TransferReceiveDMA in callback is not recommended.
- 2.0.3
  - New features:
    * Supported linked transfer in DMIC DMA driver.
    * Added new API DMIC_EnableChannelFifo/DMIC_DoFifoReset/DMIC_InstallDMADescriptor.
- 2.0.2
  - New feature:
    * Supports more channels in driver.
- 2.0.1
  - Added control macro to enable/disable the RESET and CLOCK code in current driver.
- 2.0.0
  - Initial version.

1.11 FLEXCOMM

The current FLEXCOMM driver version is 2.0.1.

- 2.0.2
  - Fixed typo in FLEXCOMM15_DriverIRQHandler().
  - Added instance calculation in FLEXCOMM16_DriverIRQHandler() to align with Flexcomm 14 and 15.
- 2.0.1
  - Added more IRQHandler code in drivers to adapt new devices.
- 2.0.0
  - Initial version.

1.12 I2C

The current I2C driver version is 2.0.5.

- 2.0.5
  - Bug fixes:
    * Fixed wrong assignment for datasize in I2C_InitTransferStateMachineDMA.
    * Fixed wrong working flow in I2C_RunTransferStateMachineDMA to ensure master can work in no start flag and no stop flag mode.

    - Fixed wrong working flow in I2C_RunTransferStateMachine and added kReceiveDataBeginState in _i2c_transfer_states to ensure master can work in no start flag and no stop flag mode.
    - Fixed wrong handle state in I2C_MasterTransferDMAHandleIRQ. After all the data has been transfered or nak is returned, handle state should be changed to idle.
  - Improvements:
    - Rounded up the calculated divider value in I2C_MasterSetBaudRate.
- 2.0.4
  - Improvements:
    - Updated the I2C_WATI_TIMEOUT macro to unified name I2C_RETRY_TIMES
    - Updated the "I2C_MasterSetBaudRate" API to support baudrate configuration for feature QN9090.
  - Bug fixes:
    - Fixed build warnning caused by uninitialized variable.
    - Fixed COVERITY issue of unchecked return value in I2C_RTOS_Transfer.
- 2.0.3
  - Unified component full name to FLEXCOMM I2C(DMA/FREERTOS) driver.
- 2.0.2
  - Improvements: In slave IRQ:
    1. Changed slave receive process to first set the I2C_SLVCTL_SLVCONTINUE_MASK to acknowledge the received data, then do data receive.
    2. Improved slave transmit process to set the I2C_SLVCTL_SLVCONTINUE_MASK immediately after write the data.
- 2.0.1
  - Improvements:
    - Added I2C_WATI_TIMEOUT macro to allow user to specify the timeout times for waiting flags in functional API and blocking transfer API.
- 2.0.0
  - Initial version.

1.13 I2S

The current I2S driver version is 2.1.0.

- 2.1.0
  - Improvements:
    - Added feature for the FLEXCOMM which supports I2S and has interconnection with DMIC.
    - Used feature to control PDMDATA instead of I2S_CFG1_PDMDATA.
    - Added member bytesPerFrame in i2s_dma_handle_t, used for DMA transfer width configure instead of use sizeof(uint32_t) hardcode.
    - Used the macro provide by DMA driver to define the I2S DMA descriptor.
  - Bug fix:
    - Fixed I2S DMA driver to always generate duplicate callback.
- 2.0.3
  - Added feature to remove configuration for second channel on LPC51U68.

- 2.0.2
  - Added ENABLE_IRQ handle after register I2S interrupt handle.
- 2.0.1
  - Unified component full name to FLEXCOMM I2S (DMA) driver.
- 2.0.0
  - Initial version.

1.14 SPI

The current SPI driver version is 2.0.4.

- 2.0.4
  - Bug fix:
    - Fixed the bug of using read only mode in DMA transfer. In DMA transfer mode, if transfer->txData is NULL, code attempts to read data from the address of 0x0 for configuring the last frame.
    - Fixed wrong assignment of handle->state. During transfer handle->state should be kSPI_Busy not kStatus_SPI_Busy.
  - Improvements:
    - Rounded up the calculated divider value in SPI_MasterSetBaud.
- 2.0.3
  - Added "SPI_FIFO_DEPTH(base)" more definition.
- 2.0.2
  - Unified component full name to FLEXCOMM SPI(DMA/FREERTOS) driver.
- 2.0.1
  - Changed the data buffer from uint32_t to uint8_t which matches the real applications for SPI DMA driver.
  - Added dummy data setup API to allow users to configure the dummy data to be transferred.
  - Added new APIs for half-duplex transfer function, users can send and receive data by one API in polling/interrupt/DMA way, and users can choose either transmit first or receive first. Also, the PCS pin can be configured as assert status in transmission (between transmit and receive) by setting the isPcsAssertInTransfer to true.
- 2.0.0
  - Initial version.

1.15 USART

- The current USART driver version is 2.1.0.
- 2.1.0
  - New feature:
    - Added features to allow users configure the USART to synchronous transfer(master and slave) mode.
  - Bug fix:
    - Modified USART_SetBaudRate to get more acurate configuration.
- 2.0.3
  - New feature:

    - ∗ Added new APIs to allow users to enable the CTS which determines whether CTS is used for flow control.
- 2.0.2
  - Bug fix:
    - ∗ Fixed the bug where transfer abort APIs cannot disable the interrupts. The FIFOINTENSET register should not be used to disable the interrupts, instead using the FIFOINTENCLR register.
- 2.0.1
  - Unified component full name to FLEXCOMM USART (DMA/FREERTOS) driver.
- 2.0.0
  - Initial version.

1.16 FLEXIO

The current FLEXIO driver version is 2.0.2.

- 2.0.2:
  - Improvements:
    - ∗ Split FLEXIO component which combines all flexio/flexio_uart/flexio_i2c/flexio_i2s drivers into several components. FlexIO component, flexio_uart component, flexio_i2c_master component, and flexio_i2s component.
- 2.0.1
  - Bug fix:
    - ∗ Fixed the dozen mode configuration error in FLEXIO_Init API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.

1.17 FLEXIO_UART

The current FLEXIO_UART driver version is 2.1.5.

- 2.1.5
  - Trigger user callback after all the data in ringbuffer are received in FLEXIO_UART_TransferReceiveNonBlocking.
- 2.1.4
  - Unified component full name to FLEXIO UART(DMA/EDMA) Driver.
- 2.1.3
  - Bug fixes: The following modifications support FLEXIO using multiple instances.
    - ∗ Removed FLEXIO_Reset API in module Init APIs.
    - ∗ Updated module Deinit APIs to reset the shifter/timer configuration instead of disable module and disable clock.
    - ∗ Updated module Enable APIs to only support enable operation.
- 2.1.2
  - Bug fixes:
    - ∗ Fixed the transfer count calculation issue in FLEXIO_UART_TransferGetReceiveCount, FLEXIO_UART_TransferGetSendCount, FLEXIO_UART_TransferGetReceiveCountDMA, FLEXIO_UART_TransferGetSendCountDMA, FLEXIO_UART_TransferGetReceiveCountEDMA and FLEXIO_UART_TransferGetSendCountEDMA

        - Fixed the Dozen mode configuration error in FLEXIO_UART_Init API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
        - Reported error when set baudrate too low and FLEXIO cannot reach that baudrate.
        - Disabled FLEXIO_UART receive interrupt instead of disable all NVIC when read data from ring buffer. Because ring buffer is used, receive nonblocking disables all NVIC interrupts to protect the ring buffer. This has negative effects on other IPS which are using interrupt.
- 2.1.1
    - Bug fixes:
        - Changed the API name FLEXIO_UART_StopRingBuffer to FLEXIO_UART_TransferStopRingBuffer to align with the definition in C file.
- 2.1.0
    - New features:
        - Added Transfer prefix in transactional APIs.
        - Added txSize/rxSize in handle structure to record the transfer size.
    - Bug fixes:
        - Added error handle to handle the data count is zero or data buffer is NULL situation.

1.18 FLEXIO_I2C

The current FLEXIO_I2C driver version is 2.1.8.

- 2.1.8
    - Fixed coverity issue of useless call in FLEXIO_I2C_MasterTransferRunStateMachine.
- 2.1.7
    - New feature:
        - Added API of checking bus pin status.
    - Bug fixes:
        - Fixed the issue that FLEXIO_I2C_MasterTransferBlocking does not wait for STOP bit sent.
        - Fixed COVERITY issue of useless call in FLEXIO_I2C_MasterTransferRunStateMachine.
        - Fixed the issue that I2C master does not check whether bus is busy before transfer.
- 2.1.6
    - Bug fix:
        - Fixed the issue that I2C Master transfer APIs(blocking/non-blocking) do not support the situation of master transfer with subaddress and transfer data size zero, which means no data follows the subaddress.
- 2.1.5
    - Unified component full name to FLEXIO I2C Driver
- 2.1.4
    - Bug fixes: The following modifications support FlexIO using multiple instances.
        - Removed FLEXIO_Reset API in module Init APIs.
        - Updated module Deinit APIs to reset the shifter/timer config instead of disabling module/clock.

    - Updated module Enable APIs to only support enable operation.
- 2.1.3
  - Changed the prototype of FLEXIO_I2C_MasterInit to return kStatus_Success if initialization successfully and return kStatus_InvalidArgument if "(srcClock_Hz / masterConfig->baudRate_Bps) / 2 - 1" exceeds 0xFFU.
- 2.1.2
  - Fixed the FLEXIO I2C issue where the master cannot receive data from I2C slave in high baudrate.
  - Fixed the FLEXIO I2C issue where the master cannot receive NAK when master sends non-existent addr.
  - Fixed the FLEXIO I2C issue where the master cannot get transfer count successfully.
  - Fixed the FLEXIO I2C issue where the master cannot receive data successfully when sending data first.
  - Fixed the Dozen mode configuration error in FLEXIO_I2C_MasterInit API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
  - Fixed the FLEXIO_I2C_MasterTransferBlocking API calls FLEXIO_I2C_MasterTransferCreateHandle issue. This leads the s_flexioHandle/s_flexioIsr/s_flexioType variable written. Then, if calling FLEXIO_I2C_MasterTransferBlocking API multiple times, the s_flexioHandle/s_flexioIsr/s_flexioType variable cannot be written any more due to it being out of range. This leads to the following: NonBlocking transfer APIs cannot work due to register IRQ failed.
- 2.1.1
  - Bug fixes:
    - Implemented the FLEXIO_I2C_MasterTransferBlocking API which defined in header file but has no implementation in the C file.
- 2.1.0
  - New features:
    - Added Transfer prefix in transactional APIs.
    - Added transferSize in handle structure to record the transfer size.

1.19 FLEXIO_SPI

The current FLEXIO_SPI driver version is 2.1.3.

- 2.1.3
  - Unified component full name to FLEXIO SPI(DMA/EDMA) Driver.
- 2.1.2
  - Bug fixes: The following modification support FlexIO using multiple instances.
    - Removed FLEXIO_Reset API in module Init APIs.
    - Updated module Deinit APIs to reset the shifter/timer config instead of disabling module/clock.
    - Updated module Enable APIs to only support enable operation.
- 2.1.1
  - Bug fixes:
    - Fixed bug where FLEXIO SPI transfer data is in 16 bit per frame mode with eDMA.
    - Fixed bug where FLEXIO SPI transfer data is in 16 bit per frame and direction is Lsbfirst

    mode with eDMA and interrupt.
    * Fixed the Dozen mode configuration error in FLEXIO_SPI_MasterInit/FLEXIO_SPI_SlaveInit API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
  - Optimization:
    * Added #ifndef/#endif to allow user to change the default TX value at compile time.
- 2.1.0
  - New features:
    * Added Transfer prefix in transactional APIs.
    * Added transferSize in handle structure to record the transfer size.
  - Bug fixes:
    * Fixed the error register address return for 16-bit data write in FLEXIO_SPI_GetTxDataRegisterAddress.
    * Provided independent IRQHandler/transfer APIs for Master and slave to fix the baudrate limit issue.

1.20 FLEXIO_I2S

The current FLEXIO_I2S driver version is 2.1.6.

- 2.1.6
  - Bug fix:
    * Added reset flexio before flexio i2s init to make sure flexio status is normal.
- 2.1.5
  - Bug fix:
    * Fixed I2S driver use hard code for bitwidth setting.
- 2.1.4
  - Unified component's full name to FLEXIO I2S (DMA/EDMA) driver.
- 2.1.3
  - Bug fixes: The following modifications support FLEXIO using multiple instances.
    * Removed FLEXIO_Reset API in module Init APIs.
    * Updated module Deinit APIs to reset the shifter/timer config instead of disabling module/clock.
    * Updated module Enable APIs to only support enable operation.
- 2.1.2
  - New features:
    * Added configure items for all pin polarity and data valid polarity.
    * Added default configure for pin polarity and data valid polarity.
- 2.1.1
  - Bug fixes:
    * Fixed FlexIO I2S RX data read error and eDMA address error.
    * Fixed FlexIO I2S slave timer compare setting error.
- 2.1.0
  - New features:
    * Added Transfer prefix in transactional APIs.
    * Added transferSize in handle structure to record the transfer size.

1.21 FLEXIO_MCU_LCD

The current FLEXIO_MCU_LCD driver version is 2.0.2.

- 2.0.2
  - Unified component full name to FLEXIO_MCU_LCD (EDMA) driver.
- 2.0.1
  - Bug fixes: The following modification to support FlexIO using multiple instances.
    - Removed FLEXIO_Reset API in module Init APIs.
    - Updated module Deinit APIs to reset the shifter/timer configuration instead of disable module and disable clock.
    - Updated module Enable APIs to only support enable operation.
- 2.0.0
  - Initial version.

1.22 FLEXIO_CAMERA

The current FLEXIO_CAMERA driver version is 2.1.2.

- 2.1.2
  - Unified component full name to FLEXIO CAMERA (EDMA) driver.
- 2.1.1
  - Bug fixes: The following modifications support FlexIO using multiple instances.
    - Removed FLEXIO_Reset API in module Init APIs.
    - Updated module Deinit APIs to reset the shifter/timer configuration instead of disable module and disable clock.
    - Updated module Enable APIs to only support enable operation.
- 2.1.0
  - New features:
    - Added Transfer prefix in transactional APIs.

1.23 FLEXSPI

The current FLEXSPI driver version is 2.1.2.

- 2.1.2
  - Bug fixes:
    - Fixed flag name typos: kFLEXSPI_IpTxFifoWatermarkEmpltyFlag to kFLEXSPI_IpTxFifoWatermarkEmptyFlag, kFLEXSPI_IpCommandExcutionDoneFlag to kFLEXSPI_IpCommandExecutionDoneFlag.
    - Fixed comments typos such as sequencen->sequence, levle->level.
    - Fixed FLSHCR2[ARDSEQID] field clean issue.
    - Updated flexspi_config_t structure and FlexSPI_Init to support new feature FSL_FEATURE_FLEXSPI_HAS_NO_MCR0_ATDFEN and FSL_FEATURE_FLEXSPI_HAS_NO_MCR0_ARDFEN.
    - Updated flexspi_flags_t structure to support new feature FSL_FEATURE_FLEXSPI_HAS_INTEN_AHBBUSERROREN.
- 2.1.1

  - Improvements:
    - Default enable prefetch for AHB RX buffer configuration in FLEXSPI_GetDefaultConfig, which is align with the reset value in AHBRXBUFxCR0.
    - Added software workaround for ERR011377 in FLEXSPI_SetFlashConfig, add some delay after DLL lock status set to ensure correct data read/write.
- 2.1.0
  - New features:
    - Added new API FLEXSPI_UpdateRxSampleClock for user to update read sample clock source after initialization.
    - Added reset peripheral operation in FLEXSPI_Init if required.
- 2.0.5
  - Bug fixes:
    - Fixed FLEXSPI_UpdateLUT cannot do partial update issue.
- 2.0.4
  - Bug fixes:
    - Reset flash size for all ports to zero in FLEXSPI_Init, fixed the possible out of range flash access with no error reported.
- 2.0.3
  - Bug fixes:
    - Fixed AHB receive buffer size configuration issue. The FLEXSPI_AHBRXBUFCR0_BUFSZ field should configure 64 bits size, and currently the AHB receive buffer size is in bytes which means 8-bit, so the correct configuration should be config->ahbConfig.buffer[i].bufferSize / 8.
- 2.0.2
  - New features:
    - Supports DQS write mask enable/disable feature during set FLEXSPI configuration.
    - Provides new API FLEXSPI_TransferUpdateSizeEDMA for user to update eDMA transfer size(SSIZE/DSIZE) per DMA transfer.
  - Bug fixes:
    - Fixed FLEXSPI_Init invalid operation to enable AHB bus Read Access to IP RX FIFO issue.
    - Fixed FLEXSPI_Init incorrect operation to configure IP TX FIFO watermark issue.
- 2.0.1
  - Bug fixes:
    - Fixed the flag clear issue and AHB read Command index configuration issue in FLEXSPI_SetFlashConfig.
    - Updated FLEXSPI_UpdateLUT function to update LUT table from any index instead of previous command index.
    - Added bus idle wait in FLEXSPI_SetFlashConfig and FLEXSPI_UpdateLUT to ensure bus is idle before any change to FlexSPI controller.
    - Updated interrupt API FLEXSPI_TransferNonBlocking and interrupt handle flow FLEXSPI_TransferHandleIRQ.
    - Updated eDMA API FLEXSPI_TransferEDMA.
- 2.0.0
  - Initial version.

1.24 FMEAS

The current FMEAS driver version is 2.1.0.

- 2.1.0
  - Updated "FMEAS_GetFrequency","FMEAS_StartMeasure","FMEAS_IsMeasureComplete" API and add definition to match ASYNC_SYSCON.
- 2.0.0
  - Initial version ported from LPCOpen.

1.25 GPIO

The current GPIO driver version is 2.1.3.

- 2.1.4
  - Added API GPIO_PortGetInterruptStatus to retrieve interrupt status for whole port.
  - Corrected typo in header file.
- 2.1.3
  - Updated "GPIO_PinInit" API. If it has DIRCLR and DIRSET registers, use them at set 1 or clean 0.
- 2.1.2:
  - Removed deprecated APIs.
- 2.1.1:
  - API interface changes:
    - Refined naming of API while keep all original APIs, marking them as deprecated. Original API will be removed in next release. The mainin change is update API with prefix of _PinXXX() and _PorortXXX
- 2.1.0
  - Added GPIO initialize API.
- 2.0.0
  - Initial version.

1.26 I3C

- 2.0.0
  - Initial version.

1.27 IAP

The current IAP driver version is 2.0.0.

- 2.0.0
  - Initial version.

1.28 INPUTMUX

The current INPUTMUX driver version is 2.0.1.

- 2.0.1

  - Support channel mux setting in INPUTMUX_EnableSignal().
- 2.0.0
  - Initial version.

1.29 IOPCTL

The current IOPCTL driver version is 2.0.0.

- 2.0.0
  - Initial version.

1.30 LCDIF

The current LCDIF driver version is 2.0.0.

- 2.0.0
  - Initial version.

1.31 LPADC

The current LPADC driver version is 2.1.1.

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

1.32 MIPI_DSI

The current MIPI_DSI driver version is 2.0.2.

- 2.0.2
  - New feature: Support send separate DSI command from TX data array.
  - Disable all interrupts in DSI_Init.
- 2.0.1
  - Updated to support the DPHY which does not have internal DPHY PLL.
- 2.0.0
  - Initial version.

1.33 MRT

The current MRT driver version is 2.0.1.

- 2.0.1
  - Added control macro to enable/disable the RESET and CLOCK code in current driver.
- 2.0.0
  - Initial version.

1.34 MU

The Current MU driver version is 2.0.4.

- 2.0.4
  - Improvements:
    - Improved for the platforms which don't support reset assert interrupt and get the other core power mode.
- 2.0.3 -Bug fix
  - MISRA C-2012 issue fixed.
    - Fix rules, containing: rule-10.3, rule-14.4, rule-15.5.
- 2.0.2
  - Added support for MIMX8MQx.
- 2.0.1
  - Added support for MCIMX7Ux_M4.
- 2.0.0
  - Initial version.

1.35 OSTIMER

The current OSTIMER driver version is 2.0.1.

- 2.0.1
  - Removed the software reset function out of the initialization API.
  - Enabled interrupt directly instead of enabling deep sleep interrupt. Users need to enable the deep sleep interrupt in application code if needed.
- 2.0.0
  - Initial version.

1.36 PINT

The current PINT driver version is 2.1.4.

- 2.1.4
  - Improvement
    - Add feature to control distinguish PINT/SECPINT relevant interrupt/clock configurations for PINT_Init and PINT_Deinit API.
  - Bug fix
    - Fix build issue caused by incorrect macro definitions.
- 2.1.3

  - Bug fix:
    - Updated PINT_PinInterruptClrStatus to clear PINT interrupt status when the bit is asserted and check whether was triggered by edge-sensitive mode.
    - Write 1 to IST corresponding bit will clear interrupt status only in edge-sensitive mode and will switch the active level for this pin in level-sensitive mode.
    - Fixed MISRA c-2012 rule 10.1, rule 10.6, rule 10.7.
    - Added FSL_FEATURE_SECPINT_NUMBER_OF_CONNECTED_OUTPUTS to distinguish IRQ relevant array definitions for SECPINT/PINT on lpc55s69 board.
    - Fixed PINT driver c++ build error and remove index offset operation.
- 2.1.2
  - Improvement:
    - Improved way of initialization for SECPINT/PINT in PINT_Init API.
- 2.1.1
  - Improvement:
    - Enabled secure pint interrupt and add secure interrupt handle.
- 2.1.0
  - Added PINT_EnableCallbackByIndex/PINT_DisableCallbackByIndex APIs to enable/disable callback by index.
- 2.0.2
  - Added control macro to enable/disable the RESET and CLOCK code in current driver.
- 2.0.1
  - Bug fix:
    - Updated PINT driver to clear interrupt only in Edge sensitive.
- 2.0.0
  - Initial version.

1.37 POWERQUAD

- 2.0.2 -Bug fix
  - Fixed array size issue in fsl_powerquad_data.h file.
  - Fixed vector function pipeline issue.
- 2.0.1
  - Fixed build error in C++ mode.
- 2.0.0
  - Initial version.

1.38 PUF

The current PUF driver version is 2.0.1.

- 2.0.1
  - Fixed puf_wait_usec function optimization issue.
- 2.0.0
  - Initial version.

1.39 RTC

The current RTC driver version is 2.0.0.

- 2.0.0
  - Initial version.

1.40 SCTIMER

The current SCTimer driver version is 2.1.1.

- 2.1.1
  - Improvements:
    - Update the register and macro name to align with the header of devices.
- 2.1.0
  - Bug fixes:
    - Fixed issue where SCT application level Interrupt handler function is occupied by SCT driver.
    - Fixed issue where wrong value for INSYNC field inside SCTIMER_Init function.
    - Fixed issue to change Default value for INSYNC field inside SCTIMER_GetDefaultConfig.
- 2.0.1
  - Added control macro to enable/disable the RESET and CLOCK code in current driver.
- 2.0.0
  - Initial version.

1.41 SEMA42

The current SEMA42 driver version is 2.0.0.

- 2.0.0
  - Initial version.

1.42 SMARTDMA

The current SMARTDMA driver version is 2.0.0.

- 2.0.0
  - Initial version.

1.43 USDHC

The current USDHC driver version is 2.4.0.

- 2.4.0
  - Improvements:
    - Add feature macro for read/write burst length.
- 2.3.0
  - Improvements:
    - Add USDHC_SetDataConfig api to support manual tuning.

        - ∗ Remove the limitaion that source clock must be bigger than the target in function USDHC_SetSdClock by use source clock frequency as target directly.
        - ∗ Add peripheral reset in USDHC_Init function.
        - ∗ Add tuning reset support in function USDHC_Reset function.
- 2.2.8
    - Fixed out-of bounds write in function USDHC_ReceiveCommandResponse.
- 2.2.7
    - Added API USDHC_GetEnabledInterruptStatusFlags and used in USDHC_TransferHandleIRQ.
    - Removed useless member interruptFlags in usdhc_handle_t.
- 2.2.6
    - Added address align check for ADMA descriptor table address.
    - Changed USDHC_ADMA1_DESCRIPTOR_MAX_LENGTH_PER_ENTRY to (65536-4096) to make sure the data address is 4KB align for a transfer need more than one ADMA1 descriptor.
- 2.2.5
    - Fixed MDK 66-D warning.
- 2.2.4
    - Fixed issue where real clock frequency is mismatched with target clock frequency, which is caused by an incorrect prescaler calculation.
    - Added control macro to enable/disable the CLOCK code in current driver.
- 2.2.3
    - Fixed issue where AMDA did not disable with DMAEN clear.
    - Improved set clock function to check the output frequency range.
    - Dynamic set SDCLKFS during DDR enable or disable.
- 2.2.2
    - Improved read transfer cache maintain operation, combined clean and invalidated into one function.
- 2.2.1
    - Disabled the invalidate cache operation for tuning.
- 2.2.0
    - Improved USDHC to support MMC boot feature.
- 2.1.3
    - Fixed MISRA issue.
- 2.1.2
    - Fixed coverity issue.
    - Added base address and userData parameter for all callback functions.
- 2.1.1
    - Added cache maintain operation.
    - Added timeout status check for the DATA transfer which ignore error.
    - Added feature macro for SDR50/SDR104 mode.
    - Removed useless IRQ handler for different platform.
- 2.1.0
    - Integrated tuning into transfer function.
    - Added strobe DLL feature.

  - Added enableAutoCommand23 in data structure.
  - Removed enable card clock function because the controller will handle the clock on/off.
- 2.0.0
  - Initial version.

1.44 UTICK

The current UTICK driver version is 2.0.2.

- 2.0.2
  - Added new feature definition macro to enable/disable power control in drivers for some devices have no power control function.
- 2.0.1
  - Added control macro to enable/disable the CLOCK code in current driver.
- 2.0.0
  - Initial version.

1.45 WWDT

The current WWDT driver version is 2.1.3.

- 2.1.3
  - fix legacy issue when initial the MOD register.
- 2.1.2
  - Updated "WWDT_ClearStatusFlags" and "WWDT_GetStatusFlags" API to match QN9090. WDTOF is not set in case of WD reset. Get info from PMC instead.
- 2.1.1
  - Added new feature definition macro for devices have no LCOK control bit in MOD register.
  - Implemented delay/retry in WWDT driver
- 2.1.0
  - Added new parameter in configuration when initializing WWDT module, this parameter allows the user to deliver the WWDT clock frequency, and this parameter must be set.
- 2.0.0
  - Initial version.

Chapter 2
Middleware Change Log

2.1 emWin library

The currently supported version is 5.48r.

2.2 FatFs for MCUXpresso SDK

Current version is FatFs R0.13c_rev0.

- R0.13c_rev0
  - Upgraded to version 0.13c
  - Apply patches ff_13c_p1.diff,ff_13c_p2.diff, ff_13c_p3.diff and ff_13c_p4.diff.
- R0.13b_rev0
  - Upgraded to version 0.13b
- R0.13a_rev0
  - Upgraded to version 0.13a. Added patch ff_13a_p1.diff.
- R0.12c_rev1
  - Add NAND disk support.
- R0.12c_rev0
  - Upgraded to version 0.12c and applied patches ff_12c_p1.diff and ff_12c_p2.diff.
- R0.12b_rev0
  - Upgraded to version 0.12b.
- R0.11a
  - Added glue functions for low-level drivers (SDHC, SDSPI, RAM, MMC). Modified diskio.c.
  - Added RTOS wrappers to make FatFs thread safe. Modified syscall.c.
  - Renamed ffconf.h to ffconf_template.h. Each application should contain its own ffconf.h.
  - Included ffconf.h into diskio.c to enable the selection of physical disk from ffconf.h by macro definition.
  - Conditional compilation of physical disk interfaces in diskio.c.

2.3 LigJpeg for KSDK

Current version is LigJpeg 9b.

- 9b_rev1
  - New Feature:
    - The configuration file libjpeg/inc/jmorecfg.h could include user defined header file to override pixel format configuration.
- 9b_rev0
  - Initial version. Changes when integrate with SDK:
    - In libjpeg/inc/jinclude.h line 88-96, map JFREAD and JFWRITE to FATFS f_read and f_write

    - In libjpeg/inc/jmorecfg.h line 397-406, change RGB color offset.
    - In libjpeg/src/jerror.c line 79-81, don't call function exit.

2.4 LittlevGL for KSDK

- 5.3_rev1
  - Integrate LittlevGL 5.3 to SDK.

2.5 mbedTLS for MCUXpresso SDK

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

- 2.13.1_rev5
  - Bug fixes:
    - ecp_alt_ksdk.c fix CASPER port for ECJPAKE shortcut when points equal 1. This case is point addition and this shortcut follows original mbedtls_ecp_muladd() implementation which is required for ecjpake_ecp_add3().
- 2.13.1_rev4
  - New features:
    - Added support for NIST P-384 elliptic curve with CASPER driver.
- 2.13.1_rev3
  - Bug fixes:
    - Force align AES_CCM and AES_GCM self-test keys to fix unaligned key issue when using HW acceleration.
- 2.13.1_rev2
  - Bug fixes:
    - Disable default HW acceleration of SHA in parallel with AES.
- 2.13.1_rev1
  - Bug fixes:
    - Fixed incorrect macro check when skipping AES-192 or AES-256
- 2.13.1
  - New features:
    - Ported mbedTLS 2.13.1 to KSDK.
- 2.12.0_rev1
  - New features:
    - Added support for NIST P-256 elliptic curve with CASPER driver.
- 2.12.0
  - New features:
    - Ported mbedTLS 2.12.0 to KSDK.
- 2.9.0_rev2
  - New features:
    - Added support for Hashcrypt driver.
- 2.9.0_rev1
  - New features:
    - Added support for CASPER driver.
- 2.9.0

  - New features:
    - Ported mbedTLS 2.9.0 to KSDK.
- 2.6.0_rev2
  - Bug fixes:
    - ssl_cookie.c now uses SHA256 for COOKIE_MD (instead of original SHA224). Some hw crypto acceleration (such as CAU3) don't support SHA224 but all support SHA256.
- 2.6.0_rev1
  - Bug fixes:
    - ksdk_mbedtls.c bignum functions now read sign of input mbedtls_mpi at beginning of functions to properly support in place computations (when output bignum is the same as one of input bignums). Affected functions: mbedtls_mpi_mul_mpi(), mbedtls_mpi_mod_mpi(), ecp_mul_comb().
- 2.6.0
  - New features:
    - Ported mbedTLS 2.6.0 to KSDK.
    - Added MBEDTLS_FREESCALE_FREERTOS_CALLOC_ALT to allow alternate implementation of pvPortCalloc() when using .c.
- 2.5.1_rev1
  - New features:
    - Added support for DCP driver.
- 2.5.1
  - New features:
    - Ported mbedTLS 2.5.1 to KSDK.
- 2.4.2_rev2
  - New features:
    - Added Curve25519 support for CAU3.
    - Added MBEDTLS_ECP_MUL_MXZ_ALT configuration parameter enabling overloading of ecp_mul_mxz().
- 2.4.2_rev1
  - New features:
    - Added support for CAU3 driver.
    - Added new files:
    - .c - contains regular software implementation of DES algorithm with added MBEDTLS_DES3_SETKEY_DEC_ALT and MBEDTLS_DES3_SETKEY_ENC_ALT config parameters.
    - .h - contains modified mbedtls_des_context and mbedtls_des3_context structures.
    - Added MBEDTLS_DES3_SETKEY_DEC_ALT configuration parameter enabling reloading of mbedtls_des3_set2key_dec() and mbedtls_des3_set3key_dec().
    - Added MBEDTLS_DES3_SETKEY_ENC_ALT configuration parameter enabling reloading of mbedtls_des3_set2key_enc() and mbedtls_des3_set3key_enc().
- 2.4.2
  - New features:
    - Ported mbedTLS 2.4.2 to KSDK 2.0.0.
    - Added CRYPTO_InitHardware() function.
    - Added new file:

        - .h - contains declaration of CRYPTO_InitHardware() function and should be included in applications.
- 2.3.0_rev1
    - New features:
        - Added support for CAAM driver.
        - In LTC-specific wrapper, allocate temporary integers from heap in one large block.
- 2.3.0
    - New features:
        - Ported mbedTLS 2.3.0 to KSDK 2.0.0.

2.2.1

- New features:
    - Ported mbedTLS 2.2.1 to KSDK 2.0.0.
    - Added support of MMCAU cryptographic acceleration module. Accelerated MD5, SHA, AES, and DES.
    - Added support of LTC cryptographic acceleration module. Accelerated AES, DES, and PKHA.
    - Added new files:
    - .c - alternative implementation of cryptographic algorithm functions using LTC and MMCAU module drivers.
    - .h - configuration settings used by mbedTLS KSDK bare metal examples.
    - Added mbedTLS KSDK bare-metal examples:
        - <board name> - KSDK mbedTLS benchmark application.
        - <board name> - KSDK mbedTLS self-test application.
    - Added MBEDTLS_GCM_CRYPT_ALT configuration parameter enabling reloading of mbedtls_gcm_crypt_and_tag().
    - Added MBEDTLS_ECP_MUL_COMB_ALT to enable alternate implementation of ecp_mul_comb().
    - Added MBEDTLS_ECP_ADD_ALT configuration parameter enabling reloading of ecp_add().
    - Added MBEDTLS_DES_SETKEY_DEC_ALT configuration parameter enabling reloading of mbedtls_des_setkey_dec(), mbedtls_des3_set2key_dec() and mbedtls_des3_set3key_dec().
    - Added MBEDTLS_DES_SETKEY_ENC_ALT configuration parameter enabling reloading of mbedtls_des_setkey_enc(), mbedtls_des3_set2key_enc() and mbedtls_des3_set3key_enc().
    - Added MBEDTLS_DES_CRYPT_CBC_ALT configuration parameter enabling reloading of mbedtls_des_crypt_cbc().
    - Added MBEDTLS_DES3_CRYPT_CBC_ALT configuration parameter enabling reloading of mbedtls_des3_crypt_cbc().
    - Added MBEDTLS_AES_CRYPT_CBC_ALT configuration parameter enabling reloading of mbedtls_aes_crypt_cbc().
    - Added MBEDTLS_AES_CRYPT_CTR_ALT configuration parameter enabling reloading of mbedtls_aes_crypt_ctr().
    - Added MBEDTLS_CCM_CRYPT_ALT configuration parameter enabling reloading of mbedtls_ccm_encrypt_and_tag() and mbedtls_ccm_auth_decrypt().
    - Added MBEDTLS_MPI_ADD_ABS_ALT configuration parameter enabling reloading of

    mbedtls_mpi_add_abs().
  - Added MBEDTLS_MPI_SUB_ABS_ALT configuration parameter enabling reloading of mbedtls_mpi_sub_abs().
  - Added MBEDTLS_MPI_EXP_MOD_ALT configuration parameter enabling reloading of mbedtls_mpi_exp_mod().
  - Added MBEDTLS_MPI_MUL_MPI_ALT configuration parameter enabling reloading of mbedtls_mpi_mul_mpi().
  - Added MBEDTLS_MPI_MOD_MPI_ALT configuration parameter enabling reloading of mbedtls_mpi_mod_mpi().
  - Added MBEDTLS_MPI_GCD_ALT configuration parameter enabling reloading of mbedtls_mpi_gcd().
  - Added MBEDTLS_MPI_INV_MOD_ALT configuration parameter enabling reloading of mbedtls_mpi_inv_mod().
  - Added MBEDTLS_MPI_IS_PRIME_ALT configuration parameter enabling reloading of mbedtls_mpi_is_prime().
  - Added encrypt/decrypt mode to mbedtls_des_context and mbedtls_des3_context structure.
  - Added carriage return ” for mbedtls_printf() in self test functions.

2.6 SDMMC

The current driver version is 2.2.12.

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

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

- Increase the sd io driver strength for SD2.0 card.

BugFix:

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

2.2.11

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

2.2.10

- BugFix:
  - Add NULL pointer check for USDHC FreeRTOS adaptor transfer complete callback.

  - Add event value check for all the FreeRTOS events to fix program hangs when a card event occurs before create.

2.2.9

- Improvement:
  - Add NULL pointer check for sdmmchostcard_usr_param_t member cd in card detect callback to avoid memory corruption.
  - Add card voltage switch function in sdmmhostcard_usr_param_t to allow application register card signal line voltage switch function.
- Bug fix
  - Fix host FreeRTOS adaptor and polling adaptor can't detect card insert bug for usdhc.
  - Fix sdhc host layer build issue and typo issue.

2.2.8

- Improvement:
  - Update SDMMC to support SDIO interrupt.

2.2.7

- BugFix:
  - Fix MDK 66-D warning.

2.2.6

- Improvement:
  - Remove some soc specific header files from porting layer.
  - Save MMC OCR registers while sending CMD1 with argument 0.
- Bugfix:
  - Add MMC_PowerOn function in which there is delay function after powerup sdcard. Otherwise, the card initialization by fail.

2.2.5

- New features:
  - Add SD_ReadStatus api to get 512bit SD status.
  - Add error log support in sdcard functions.
  - Add SDMMC_ENABLE_SOFTWARE_TUNING to enable/disable software tuning and it is disabled by default.
  - Add error procedure in the transfer function to improve stability.
  - Remove deprecated gpio api in host layer.

2.2.4

- Bug fix:
  - Fixed DDR mode data sequence miss issue, which is caused by NIBBLE_POS.
- New features:
  - Increased g_sdmmc 512byte to improve the performance when application use a non-word align data buffer address.
  - Used OCR access mode bits to determine the mmccard high capacity flag.
  - Enabled auto cmd12 for SD read/write.

  - Disabled DDR mode frequency multiply by 2.

2.2.3

- Bug fix:
  - Added response check for send operation condition command. If not checked, the card may occasionally init fail.

2.2.2

- Moved set card detect priority operation before enable IRQ.

2.2.1

- New features:
  - Improved MMC Boot feature.
  - Keep SD_Init/SDIO_Init function for forward compatibility.

2.2.0

- New features:
  - Separated the SD/MMC/SDIO init API to xxx_CardInit/xxx_HostInit.
  - Allowed user register card detect callback, select card detect type, and determine the card detect timeout value.
  - Allowed user register the power on/off function, and determine the power on/off delay time.
  - SD_Init/SDIO_Init will be deprecated in the next version.
  - Added write complete wait operation for MMC_Write to fix command timeout issue.

2.1.6

- Enhanced SD IO default driver strength.

2.1.5

- Fixed Coverity issue.
- Fixed SD v1.x card write fail issue. It was caused by the block length set error.

2.1.4

- Miscellaneous:
  - Added Host reset function for card re-initialization.
  - Added Host_ErrorRecovery function for host error recovery procedure.
  - Added cache maintain operation
  - Added HOST_CARD_INSERT_CD_LEVEL to improve compatibility.
- Bug fix:
  - Fixed card cannot detect dynamically.

2.1.3

- Bug fix:
  - Non high-speed sdcard init fail at switch to high speed.
- Miscellaneous:
  - Optimized tuning/mmc switch voltage/mmc select power class/mmc select timing function.
  - Added strobe dll for mmc HS400 mode.

  - Added Delay for SDCard power up.

2.1.2

- New features:
  - Added fsl_host.h to provide prototype to adapt different controller IPs(SDHC/SDIF).
  - Added adaptor code in SDMMC/Port folder to adapt different host controller IPs with different transfer modes(interrupt/polling/FreeRTOS). Application includes a different adaptor code to make application more simple.
  - Adaptor code provides HOST_Init/HOST_Deinit/CardInsertDetect. APIs to do host controller initialize and transfer function configuration. SDMMC card stack uses adaptor code inside stack to wait card insert and configure host when calling card init APIs (SD_Init/MMC_Init/-SDIO_Init).
  - This change requires the user to include host adaptor code into the application. If not changed, link errors saying it cannot find the definition of HOST_Init/HOST_Deinit/CardInsertDetect appear.
- New features: Improved SDMMC to support SD v3.0 and eMMC v5.0.
- Bug fix:
  - Fixed incorrect comparison between count and length in MMC_ReadBlocks/MMC_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.

2.7 SDIO

The current driver version is 2.2.12.

- 2.2.12
  - Improvement:
    - Add manual tuning function for looking for the tuning window automatically.
    - Fix the build warning by changing the old style function declaration static status_t inline to static inline status_t(found by adding -Wold-style-declaration in armgcc build flag).

    - Fix the fall through build warning by adding SUPPRESS_FALL_THROUGH_WARNING() in sdio driver.
- 2.2.11
  - Bug fix:
    - Add check card async interrupt capability in function SDIO_GetCardCapability.
    - Fix OUT OF BOUNDS access in function SDIO_IO_Transfer.
- 2.2.10
  - Bug fix:
    - Fix SDIO card driver get an incorrect io number when the card io number is bigger than 2.
  - New feature:
    - Add SDIO 3.0 support.
    - Add API SDIO_IO_RW_Direct for direct read/write card register access.
- 2.2.9
  - Improvement:
    - Add api SDIO_SetIOIRQHandler/SDIO_HandlePendingIOInterrupt to handle multi io pending IRQ.
- 2.2.8
  - Improvement:
    - Update sdmmc to support SDIO interrupt.
    - Add api SDIO_GetPendingInterrupt to get the pending io interrupt.
- 2.2.7
  - Bug fix:
    - Fix MDK 66-D warning.
- 2.2.6
  - New features:
    - Add a unify transfer interface for SDIO.
  - Bug fix:
    - Wrong pointer address used by SDMMCHOST_Init.
- 2.1.5
  - Bug fix:
    - Improved SDIO card init sequence and add retry option for SDIO_SwitchToHighSpeed function.
- 2.1.4
  - Miscellaneous:
    - Added Go_Idle function for SDIO card.
- 2.0.0
  - Initial version.

2.8 SDSPI

The current driver version is 2.1.4.

- 2.1.4
  - Bug fix:
    - Fix MDK 66-D warning.

- 2.1.3
  - Improve sdspi code size and performance.
- 2.0.0
  - Initial version.

2.9 HOST CONTROLLER ADAPTER

The current driver version is 2.2.13.

- 2.2.13
  - Improvements:
    - Add feature macro FSL_FEATURE_USDHC_HAS_NO_RW_BURST_LEN for USDH-C host controller adapter.

2.10 USB stack for MCUXpresso SDK

-improvement for release after 2.4.0 -separate composite audio examples' channel,sample rate,format parameters from commom macro to in dedicated macro and out dedicated macro. -replace USB_PrepareData with USB_AudioRecorderGetBuffer. The current version of USB stack is 2.4.0.

- 2.4.0
  - Improvement:
    - Device Charger Detection (DCD) software architecture was refactored.
  - New features:
    - Enabled Device Charger Detection (DCD) on RT1060.
    - Enabled Device Charger Detection on RT600.
    - Enabled host battery charger function on RT600.
- 2.3.0
  - New features:
    - Added host video camera support. example: usb_host_video_camera
    - Added a new device example. example: usb_device_composite_cdc_hid_audio_unified
- 2.2.0
  - New features:
    - Added device DFU support.
    - Supported OM13790DOCK on LPCXpresso54018.
    - Added multiple logical unit support in msc class driver, updated usb_device_lba_information_struct_t to support this.
    - Supported multiple transfers for host ISO on IP3516HS.
  - Bug fixes:
    - Fixed device ip3511 prime data length than maxpacket size issue.
    - Initialized interval attribute in usb_device_endpoint_struct_t/usb_device_endpoint_init_struct_t.
    - Removed unnecessary header file in device CDC class driver, removed unnecessary usb_echo, and added DEBUG macro for necessary usb_echo in device CDC class driver.
    - Fixed device IP3511HS unfinished interrupt transfer missing issue.
- 2.1.0
  - New features:

        - Added host RNDIS support. example: lwip_dhcp_usb
        - Enabled USB 3.0 support on device stack.
        - Power Delivery feature: Added OM13790HOST support; Added auto policy feature; Printed e-marked cable information;
- 2.0.1
    - Bug fixes:
        - Fixed some USB issues: Fixed MSC CV test failed in MSC examples.
        - Changed audio codec interfaces.
- 2.0.0
    - New features:
        - PTN5110N support.
    - Bug fix:
        - Added some comments, fixed some minor USB issues.
- 1.9.0
    - New features:
        - Examples:
            - usb_pd_alt_mode_dp_host
- 1.8.2
    - Updated license.
- 1.8.1
    - Bug fix:
        - Verified some hardware issues, support aruba_flashless.
- 1.8.0
    - New features:
        - Examples:
            - usb_device_composite_cdc_vcom_cdc_vcom
            - usb_device_composite_hid_audio_unified
            - usb_pd_sink_battery
            - Changed usb_pd_battery to usb_pd_charger_battery.
    - Bug fix:
        - Code clean up, removed some irrelevant code.
- 1.7.0
    - New features:
        - USB PD stack support.
    - Examples:
        - usb_pd
        - usb_pd_battery
        - usb_pd_source_charger
- 1.6.3
    - Bug fix: -IP3511_HS driver control transfer sequence issue, enabled 3511 ip cv test.
- 1.6.2
    - New features:
        - Multi instance support.
- 1.6.1
    - New features:

  - Changed the struct variable address method for device_video_virtual_camera and host_phdc_manager.
- 1.6.0
  - New features:
    - Supported Device Charger Detect feature on usb_device_hid_mouse.
- 1.5.0
  - New features:
    - Supported controllers
      - OHCI (Full Speed, Host mode)
      - IP3516 (High Speed, Host mode)
      - IP3511 (High Speed, Device mode)
    - Examples:
      - usb_lpm_device_hid_mouse
      - usb_lpm_device_hid_mouse_lite
      - usb_lpm_host_hid_mouse
- 1.4.0
  - New features:
    - Examples:
      - usb_device_hid_mouse/freertos_static
      - usb_suspend_resume_device_hid_mouse_lite
- 1.3.0
  - New features:
    - Supported roles
      - OTG
    - Supported classes
      - CDC RNDIS
    - Examples
      - usb_otg_hid_mouse
      - usb_device_cdc_vnic
      - usb_suspend_resume_device_hid_mouse
      - usb_suspend_resume_host_hid_mouse
- 1.2.0
  - New features:
    - Supported controllers
      - LPC IP3511 (Full Speed, Device mode)
- 1.1.0
  - Bug fix:
    - Fixed some issues in USB certification.
    - Changed VID and Manufacturer string to NXP.
  - New features:
    - Supported classes
      - Pinter
    - Examples:
      - usb_device_composite_cdc_msc_sdcard
      - usb_device_printer_virtual_plain_text

      - usb_host_printer_plain_text
- 1.0.1
  - Bug fix:
    - Improved the efficiency of device audio speaker by changing the transfer mode from interrupt to DMA, thus providing the ability to eliminate the periodic noise.
- 1.0.0
  - New features:
    - Supported roles
      - Device
      - Host
    - Supported controllers:
      - KHCI (Full Speed)
      - EHCI (High Speed)
    - Supported classes:
      - AUDIO
      - CCID
      - CDC
      - HID
      - MSC
      - PHDC
      - VIDEO
    - Examples:
      - usb_device_audio_generator
      - usb_device_audio_speaker
      - usb_device_ccid_smart_card
      - usb_device_cdc_vcom
      - usb_device_cdc_vnic
      - usb_device_composite_cdc_msc
      - usb_device_composite_hid_audio
      - usb_device_composite_hid_mouse_hid_keyboard
      - usb_device_hid_generic
      - usb_device_hid_mouse
      - usb_device_msc_ramdisk
      - usb_device_msc_sdcard
      - usb_device_phdc_weighscale
      - usb_device_video_flexio_ov7670
      - usb_device_video_virtual_camera
      - usb_host_audio_speaker
      - usb_host_cdc
      - usb_host_hid_generic
      - usb_host_hid_mouse
      - usb_host_hid_mouse_keyboard
      - usb_host_msd_command
      - usb_host_msd_fatfs
      - usb_host_phdc_manager

- usb_keyboard2mouse
- usb_pin_detect_hid_mouse

Chapter 3
RTOS Change Log

3.1 FreeRTOS for MCUXpresso SDK.

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

- 1.4.7_rev0
  - New features:
    - Add optional allocation scheme heap_useNewlib.c by D. Nadler.
    - Enable task aware debugging for cm33 platforms
    - Move tickless implementation to application layer
  - Other changes:
    - Fix other build warnings, errors
- 1.4.6_rev0
  - New features:
    - Update support of CM33 port with Trustzone, MPU, FPU support
    - Add support for AWS test for Cypress WiFi
    - Use lwip netif api to avoid lwIP raw API calls outside of tcpip thread in aws_wifi.c
  - Other changes:
    - Fix issues with mflash driver
    - Fix other build warnings, errors
- 1.4.0_rev1
  - New features:
    - Add implementation of vTaskEndScheduler for CM0 GCC port.
    - Support for CM33, CM33F architectures based on CM3, CM4F ports
- 1.4.0_rev0
  - New features:
    - Support for pkcs11 for several platforms, secure element host library under pkcs11/portable/nxp folder
    - Lwip, wifi_qca support for secure_sockets in secure_sockets/portable/nxp folder
    - Flash driver support for several platforms in third_party/mcu_vendor/nxp folder
    - Generic support for aws_wifi under wifi/portable/nxp/common folder
  - Other changes:
    - Fix several build warnings, errors

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

- 9.0.0_rev3
  - New features:
    - Tickless idle mode support for Cortex-A7. Add fsl_tickless_epit.c and fsl_tickless_generic.h in portable/IAR/ARM_CA9 folder.

    - Enabled float context saving in IAR for Cortex-A7. Added configUSE_TASK_FPU_SUPPORT macros. Modified port.c and portmacro.h in portable/IAR/ARM_CA9 folder.
  - Other changes:
    - Transformed ARM_CM core specific tickless low power support into generic form under freertos/Source/portable/low_power_tickless/.
- 9.0.0_rev2
  - New features:
    - Enabled MCUXpresso thread aware debugging. Add freertos_tasks_c_additions.h and configINCLUDE_FREERTOS_TASK_C_ADDITIONS_H and configFRTOS_MEMORY_SCHEME macros.
- 9.0.0_rev1
  - New features:
    - Enabled -flto optimization in GCC by adding **attribute**((used)) for vTaskSwitchContext.
    - Enabled KDS Task Aware Debugger. Apply FreeRTOS patch to enable configRECORD-_STACK_HIGH_ADDRESS macro. Modified files are task.c and FreeRTOS.h.
- 9.0.0_rev0
  - New features:
    - Example freertos_sem_static.
    - Static allocation support RTOS driver wrappers.
  - Other changes:
    - Tickless idle rework. Support for different timers is in separated files (fsl_tickless_-systick.c, fsl_tickless_lptmr.c).
    - Removed configuration option configSYSTICK_USE_LOW_POWER_TIMER. Low power timer is now selected by linking of apropriate file fsl_tickless_lptmr.c.
    - Removed configOVERRIDE_DEFAULT_TICK_CONFIGURATION in RVDS port. Use of **attribute**((weak)) is the preferred solution. Not same as _weak!
- 8.2.3
  - New features:
    - Tickless idle mode support.
    - Added template application for Kinetis Expert (KEx) tool (template_application).
  - Other changes:
    - Folder structure reduction. Keep only Kinetis related parts.

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer’s applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: July 2019

Document identifier: MCUSDKRT595RN