MCUXpresso SDK Release Notes Supporting lpcxpresso55s69

Change Logs

NXP Semiconductors

Contents

Driver Change Log

Middleware Change Log

Title

Page No.

RTOS Change Log

Component Change Log

1 Driver Change Log

CLOCK

The current CLOCK driver version is 2.3.8.

- 2.3.8
  - Bug Fixes
    - Fixed an issue that ss_progmodfm_t, ss_progmoddp_t, and ss_modwvctrl_t use wrong shift value.
- 2.3.7
  - Improvements
    - Add errata workaround for pll lock bit in CLOCK_SetPLL0Freq() and CLOCK_SetPLL1Freq().
- 2.3.6
  - Bug Fixes
    - Correct the fail status condition in CLOCK_SetupExtClocking().
- 2.3.5
  - Improvements
    - Added lost comments for some enumerations.
- 2.3.4
  - Bug Fixes
    - Fix CLOCK_SetClkDiv function to set kCLOCK_DivFlexFrgx correctly.
    - Fixed violations of MISRA C-2012 rule 16.3, rule 10.4, rule 10.3.
- 2.3.3
  - Bug Fixes
    - Fix the pllRate in CLOCK_SetPLL1Freq().
- 2.3.2
  - Improvements
    - Optimized the CLOCK_GetPLL0OutFromSetup() function with the usage of input parameter.
- 2.3.1
  - Bug Fixes
    - Fixed MISRA C-2012 rule 10.1, rule 12.2, rule 14.4 and so on.
    - Fixed IAR warning Pa082 for the clock driver.
- 2.3.0
  - New Features
    - Moved SDK_DelayAtLeastUs function from clock driver to common driver.
- 2.2.2
  - Bug Fixes
    - Corrected the PLL.SELI setting to align with new UM.
    - Changed the PLL lock reliable condition.
- 2.2.1
  - Improvements

- Removed redundant macro definitions.

- 2.2.0
  - New Features
    - Added CLOCK_SetupPLUClkInClocking() to store the PLU CLKIN frequency.
- 2.1.1
  - Improvements
    - Updated CLOCK_SetFLASHAccessCyclesForFreq() to support up to 150MHz frequency.
- 2.1.0
  - New Features
    - Added new API CLOCK_DelayAtLeastUs() implemented by DWT to allow users to set delay in unit of microsecond.
- 2.0.4
  - Bug Fixes
    - Fixed C++ build errors in CLOCK_GetClockAttachId() and CLOCK_AttachClk().
- 2.0.3
  - Bug Fixes
    - Fixed attach incorrect attach_id.
- 2.0.2
  - New Features
    - Added get actual clock attach id API to allow users to obtain the actual clock source in target register.
  - Bug Fixes
    - The attach clock and get actual clock attach id APIs should check combination of two clock source.
  - Optimization
    - Made the judgement statements more clear.
    - Strengthened the compatibility of clock attach id.
    - Removed some non-meaningful definitions and add some useful ones to enhance readability.
- 2.0.1
  - Some minor fixes.
- 2.0.0
  - initial version.

RESET

The current RESET driver version is 2.4.0.

- 2.4.0
  - Improvements
    - Add RESET_ReleasePeripheralReset API.
- 2.0.3
  - Improvements

    - Add CASPER_RSTS,HASHCRYPT_RSTS and PUF_RSTS.
- 2.0.2
  - Bug Fixes
    - Fixed MISRA C-2012 rule 10.3.
- 2.0.1
  - Improvements
    - Updated component full_name to "Reset Driver".
- 2.0.0
  - Initial version.

ANACTRL

The current ANACTRL driver version is 2.3.1.

- 2.3.1
  - Bug Fixes
    - Added casts to prevent overflow caused by capturing large target clock.
- 2.3.0
  - Improvements
    - Added AUX_BIAS control APIs.
- 2.2.0
  - Improvements
    - Added some macros to separate the scenes that some bit fields are reserved for some devices.
    - Optimized the comments.
    - Optimized the code implementation inside some functions.
- 2.1.2
  - Bug Fixes
    - Fixed MISRA C-2012 rule 10.3 and rule 17.7.
- 2.1.1
  - Bug Fixes
    - Removed AnalogTestBus configuration to align with new header.
- 2.1.0
  - Improvements
    - Updates for LPC55xx A1.
      - Removed the control of bitfield FRO192M_CTRL_ENA_48MHZCLK, XO32M_CTRL_ACBUF_PASS_ENABLE.
      - Removed status bits in ANACTRL_STATUS: PMU_ID OSC_ID FINAL_TEST_DONE_VECT.
      - Removed API ANACTRL_EnableAdcVBATDivider() and APIs which operate the RingOSC registers.
      - Removed the configurations of 32 MHz Crystal oscillator voltage source supply control register.
      - Added API ANACTRL_ClearInterrupts().

- 2.0.0
  - Initial version.

CMP

The current CMP driver version is 2.2.1.

- 2.2.1
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.1, rule 10.4, and rule 17.7.
- 2.2.0
  - Improvements:
    - Added API to configure the sampling mode and clock divider of the CMP Filter.
    - Supported CMP filter sampling mode configuration.
- 2.1.0
  - New Features:
    - Added API to get default CMP user configuration structure.
    - Supported CMP filter clock divider settings.
    - Combined the settings of VREF source and VREF value into one API CMP_SetVREF().
    - Extracted CMP input source selection from CMP_Init() to CMP_SetInputChannels().
  - Improvements:
    - Formatted API naming, variable naming and comment style for better readability.
    - Added comments for APIs in source file.
- 2.0.1
  - Bug Fixes
    - Fixed missing ’const’ qualifier for structure variable in function parameter.
- 2.0.0
  - Initial version.

COMMON

The current COMMON driver version is 2.4.0.

- 2.4.0
  - New Features
    - Added EnableIRQWithPriority, IRQ_SetPriority, and IRQ_ClearPendingIRQ for ARM.
    - Added MSDK_EnableCpuCycleCounter, MSDK_GetCpuCycleCount for ARM.
- 2.3.3
  - New Features
    - Added NETC into status group.
- 2.3.2
  - Improvements
    - Make driver aarch64 compatible
- 2.3.1

  - Bug Fixes
    - Fixed MAKE_VERSION overflow on 16-bit platforms.
- 2.3.0
  - Improvements
    - Split the driver to common part and CPU architecture related part.
- 2.2.10
  - Bug Fixes
    - Fixed the ATOMIC macros build error in cpp files.
- 2.2.9
  - Bug Fixes
    - Fixed MISRA C-2012 issue, 5.6, 5.8, 8.4, 8.5, 8.6, 10.1, 10.4, 17.7, 21.3.
    - Fixed SDK_Malloc issue that not allocate memory with required size.
- 2.2.8
  - Improvements
    - Included stddef.h header file for MDK tool chain.
  - New Features:
    - Added atomic modification macros.
- 2.2.7
  - Other Change
    - Added MECC status group definition.
- 2.2.6
  - Other Change
    - Added more status group definition.
  - Bug Fixes
    - Undef __VECTOR_TABLE to avoid duplicate definition in cmsis_clang.h
- 2.2.5
  - Bug Fixes
    - Fixed MISRA C-2012 rule-15.5.
- 2.2.4
  - Bug Fixes
    - Fixed MISRA C-2012 rule-10.4.
- 2.2.3
  - New Features
    - Provided better accuracy of SDK_DelayAtLeastUs with DWT, use macro SDK_DELAY_USE_DWT to enable this feature.
    - Modified the Cortex-M7 delay count divisor based on latest tests on RT series boards, this setting lets result be closer to actual delay time.
- 2.2.2
  - New Features
    - Added include RTE_Components.h for CMSIS pack RTE.
- 2.2.1
  - Bug Fixes
    - Fixed violation of MISRA C-2012 Rule 3.1, 10.1, 10.3, 10.4, 11.6, 11.9.
- 2.2.0
  - New Features

    - Moved SDK_DelayAtLeastUs function from clock driver to common driver.
- 2.1.4
  - New Features
    - Added OTFAD into status group.
- 2.1.3
  - Bug Fixes
    - MISRA C-2012 issue fixed.
      - Fixed the rule: rule-10.3.
- 2.1.2
  - Improvements
    - Add SUPPRESS_FALL_THROUGH_WARNING() macro for the usage of suppressing fallthrough warning.
- 2.1.1
  - Bug Fixes
    - 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
  - Bug Fixes
    - MISRA C-2012 issue fixed.
      - Fixed the rule: rule-10.4.
- 2.0.1
  - Improvements
    - 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.

CTIMER

The current CTimer driver version is 2.3.1.

- 2.3.1
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.7 and 12.2.
- 2.3.0
  - Improvements
    - Added the CTIMER_SetPrescale(), CTIMER_GetCaptureValue(), CTIMER_EnableResetMatchChannel(), CTIMER_EnableStopMatchChannel(), CTIMER_EnableRising-

    EdgeCapture(), CTIMER_EnableFallingEdgeCapture(), CTIMER_SetShadowValue(),APIs Interface to reduce code complexity.

- 2.2.2
  - Bug Fixes
    - Fixed SetupPwm() API only can use match 3 as period channel issue.
- 2.2.1
  - Bug Fixes
    - Fixed use specified channel to setting the PWM period in SetupPwmPeriod() API.
    - Fixed Coverity Out-of-bounds issue.
- 2.2.0
  - Improvements
    - Updated three API Interface to support Users to flexibly configure the PWM period and PWM output.
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 8.4.
- 2.1.0
  - Improvements
    - Added the CTIMER_GetOutputMatchStatus() API Interface.
    - Added feature macro for FSL_FEATURE_CTIMER_HAS_NO_CCR_CAP2 and FSL_FEATURE_CTIMER_HAS_NO_IR_CR2INT.
- 2.0.3
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.3, 10.4, 10.6, 10.7 and 11.9.
- 2.0.2
  - New Features
    - Added new API "CTIMER_GetTimerCountValue" to get the current timer count value.
    - Added a control macro to enable/disable the RESET and CLOCK code in current driver.
    - Added a new feature macro to update the API of CTimer driver for lpc8n04.
- 2.0.1
  - Improvements
    - API Interface Change
      - Changed API interface by adding CTIMER_SetupPwmPeriod API and CTIMER_UpdatePwmPulsePeriod API, which both can set up the right PWM with high resolution.
- 2.0.0
  - Initial version.

FLEXCOMM

The current FLEXCOMM driver version is 2.0.2.

- 2.0.2
  - Bug Fixes
    - Fixed typos in FLEXCOMM15_DriverIRQHandler().

    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 10.8, 11.3, 11.6, 11.8, 11.9, 13.5.
  - Improvements
    - Added instance calculation in FLEXCOMM16_DriverIRQHandler() to align with Flexcomm 14 and 15.
- 2.0.1
  - Improvements
    - Added more IRQHandler code in drivers to adapt new devices.
- 2.0.0
  - Initial version.

I2C

The current I2C driver version is 2.3.3.

- 2.3.3
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.1.
    - Fixed issue that if master only sends address without data during I2C interrupt transfer, address nack cannot be detected.
- 2.3.2
  - Improvement
    - Enable or disable timeout option according to enableTimeout.
  - Bug Fixes
    - Fixed timeout value calculation error.
    - Fixed bug that the interrupt transfer cannot recover from the timeout error.
- 2.3.1
  - Improvement
    - Before master transfer with transactional APIs, enable master function while disable slave function and vise versa for slave transfer to avoid the one affecting the other.
  - Bug Fixes
    - Fixed bug in I2C_SlaveEnable that the slave enable/disable should not affect the other register bits.
- 2.3.0
  - Improvement
    - Added new return codes kStatus_I2C_EventTimeout and kStatus_I2C_SclLowTimeout, and added the check for event timeout and SCL timeout in I2C master transfer.
    - Fixed bug in slave transfer that the address match event should be invoked before not after slave transmit/receive event.
- 2.2.0
  - New Features
    - Added enumeration _i2c_status_flags to include all previous master and slave status flags, and added missing status flags.
    - Modified I2C_GetStatusFlags to get all I2C flags.

        - Added API I2C_ClearStatusFlags to clear all clearable flags not just master flags.
        - Modified master transactional APIs to enable bus event timeout interrupt during transfer, to avoid glitch on bus causing transfer hangs indefinitely.
    - Bug Fixes
        - Fixed bug that status flags and interrupt enable masks share the same enumerations by adding enumeration _i2c_interrupt_enable for all master and slave interrupt sources.
- 2.1.0
    - Bug Fixes
        - Fixed bug that during master transfer, when master is nacked during slave probing or sending subaddress, the return status should be kStatus_I2C_Addr_Nak rather than kStatus_I2C_Nak.
    - Bug Fixes
        - Fixed MISRA issues.
            - Fixed rules 10.1, 10.4, 13.5.
    - New Features
        - Added macro I2C_MASTER_TRANSMIT_IGNORE_LAST_NACK, so that user can configure whether to ignore the last byte being nacked by slave during master transfer.
- 2.0.8
    - Bug Fixes
        - Fixed I2C_MasterSetBaudRate issue that MSTSCLLOW and MSTSCLHIGH are incorrect when MSTTIME is odd.
- 2.0.7
    - Bug Fixes
        - Two dividers, CLKDIV and MSTTIME are used to configure baudrate. According to reference manual, in order to generate 400kHz baudrate, the clock frequency after CLKDIV must be less than 2mHz. Fixed the bug that, the clock frequency after CLKDIV may be larger than 2mHz using the previous calculation method.
        - Fixed MISRA 10.1 issues.
        - Fixed wrong baudrate calculation when feature FSL_FEATURE_I2C_PREPCLKFRG_8MHZ is enabled.
- 2.0.6
    - New Features
        - Added master timeout self-recovery support for feature FSL_FEATURE_I2C_TIMEOUT_RECOVERY.
    - Bug Fixes
        - Eliminated IAR Pa082 warning.
        - Fixed MISRA issues.
            - Fixed rules 10.1, 10.3, 10.4, 10.7, 10.8, 11.3, 11.6, 11.8, 11.9, 13.5.
- 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
  - Improvements
    - Unified the 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 writing the data.
- 2.0.1
  - Improvements
    - Added I2C_WATI_TIMEOUT macro to allow users to specify the timeout times for waiting flags in functional API and blocking transfer API.
- 2.0.0
  - Initial version.

I2S

The current I2S driver version is 2.3.2

- 2.3.2
  - Bug Fixes
    - Fixed warning for comparison between pointer and integer.
- 2.3.1
  - Bug Fixes
    - Updated the value of TX/RX software transfer state machine after transfer contents are submitted to avoid race condition.
- 2.3.0
  - Improvements
    - Added api I2S_InstallDMADescriptorMemory/I2S_TransferSendLoopDMA/I2S_-

    - TransferReceiveLoopDMA to support loop transfer.
    - Added api I2S_EmptyTxFifo to support blocking flush tx fifo.
    - Updated api I2S_TransferAbortDMA by removed the blocking flush tx fifo from this function.
  - Bug Fixes
    - Removed the while loop in abort transfer function to fix the dead loop issue under specific user case.
- 2.2.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 8.4.
- 2.2.1
  - Improvements
    - Added feature FSL_FEATURE_FLEXCOMM_INSTANCE_I2S_SUPPORT_SECONDARY_CHANNELn for the SOC has parts of instance support secondary channel.
  - Bug Fixes
    - Added volatile statement for the state variable of i2s_handle and enable the mainline channel pair before enable interrupt to avoid the issue of code excution reordering which may cause the interrupt generated unexpectedly.
- 2.2.0
  - Improvements
    - Added 8/16/24 bits mono data format transfer support in I2S driver.
    - Added new apis I2S_SetBitClockRate.
  - Bug Fixes
    - Fixed the PA082 build warning.
    - Fixed the sign-compare warning.
    - Fixed violations of the MISRA C-2012 rules 10.4, 10.8, 11.9, 10.1, 11.3, 13.5, 11.8, 10.3, 10.7.
    - Fixed the Operand don't affect result Coverity issue.
- 2.1.0
  - Improvements
    - Added a feature for the FLEXCOMM which supports I2S and has interconnection with DMIC.
    - Used a 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 using sizeof(uint32_t) hardcode.
    - Used the macro provided by DMA driver to define the I2S DMA descriptor.
  - Bug Fixes
    - Fixed the issue that I2S DMA driver always generated duplicate callback.
- 2.0.3
  - New Features
    - Added a feature to remove configuration for the second channel on LPC51U68.
- 2.0.2
  - New Features
    - Added ENABLE_IRQ handle after register I2S interrupt handle.
- 2.0.1

- Improvements
  - Unified the component full name to FLEXCOMM I2S (DMA) driver.
- 2.0.0
  - Initial version.

SPI

The current SPI driver version is 2.3.1

- 2.3.1
  - Improvements
    - Changed SPI_DUMMYDATA to 0x00.
- 2.3.0
  - Update version.
- 2.2.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules.
- 2.2.1
  - Bug Fixes
    - Fixed MISRA 2012 10.4 issue.
    - Added code to clear FIFOs before transfer using DMA.
- 2.2.0
  - Bug Fixes
    - Fixed bug that slave gets stuck during interrupt transfer.
- 2.1.1
  - Improvements
    - Added timeout mechanism when waiting certain states in transfer driver.
  - Bug Fixes
    - Fixed MISRA 10.1, 5.7 issues.
- 2.1.0
  - Bug Fixes
    - Fixed Coverity issue of incrementing null pointer in SPI_TransferHandleIRQInternal.
    - Eliminated IAR Pa082 warnings.
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 10.8, 11.3, 11.6, 11.8, 11.9, 13.5.
  - New Features
    - Modified the definition of SPI_SSELPOL_MASK to support the socs that have only 3 SSEL pins.
- 2.0.4
  - Bug Fixes
    - 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 kSP-

        I_Busy rather than kStatus_SPI_Busy.
  - Improvements
    - Rounded up the calculated divider value in SPI_MasterSetBaud.
- 2.0.3
  - Improvements
    - Added "SPI_FIFO_DEPTH(base)" with more definition.
- 2.0.2
  - Improvements
    - Unified the 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 not only send and receive data by one API in polling/interrupt/DMA way, but choose either to transmit first or to receive first. Besides, 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.

SPI_DMA

The current SPI_DMA driver version is 2.2.1.

- 2.2.1
  - Bug Fixes
    - Fixed MISRA 2012 11.6 issue..
- 2.2.0
  - Improvements
    - Supported dataSize larger than 1024 data transmit.

USART

The current USART driver version is 2.8.3.

- 2.8.3
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.3, 11.8.
- 2.8.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 14.2.
- 2.8.1
  - Bug Fixes
    - Fixed the Baud Rate Generator(BRG) configuration in 32kHz mode.

- 2.8.0
  - New Features
    - Added the rx timeout interrupts and status flags of bus status.
    - Added new rx timeout configuration item in usart_config_t.
    - Added API USART_SetRxTimeoutConfig for rx timeout configuration.
  - Improvements
    - When the calculated baudrate cannot meet user's configuration, lower OSR value is allewed to use.
- 2.7.0
  - New Features
    - Added the missing interrupts and status flags of bus status.
    - Added the check of tx error, noise error framing error and parity error in interrupt handler.
- 2.6.0
  - Improvements
    - Used separate data for TX and RX in usart_transfer_t.
  - Bug Fixes
    - Fixed bug that when ring buffer is used, if some data is received in ring buffer first before calling USART_TransferReceiveNonBlocking, the received data count returned by USART_TransferGetReceiveCount is wrong.
  - New Features
    - Added missing API USART_TransferGetSendCountDMA get send count using DMA.
- 2.5.0
  - New Features
    - Added APIs USART_GetRxFifoCount/USART_GetTxFifoCount to get rx/tx FIFO data count.
    - Added APIs USART_SetRxFifoWatermark/USART_SetTxFifoWatermark to set rx/tx FIFO water mark.
  - Bug Fixes
    - Fixed DMA transfer blocking issue by enabling tx idle interrupt after DMA transmission finishes.
- 2.4.0
  - New Features
    - Modified usart_config_t, USART_Init and USART_GetDefaultConfig APIs so that the hardware flow control can be enabled during module initialization.
  - Bug Fixes
    - Fixed MISRA 10.4 violation.
- 2.3.1
  - Bug Fixes
    - Fixed bug that operation on INTENSET, INTENCLR, FIFOINTENSET and FIFOINTENCLR should use bitwise operation not 'or' operation.
    - Fixed bug that if rx interrupt occurrs before TX interrupt is enabled and after txDataSize is configured, the data will be sent early by mistake, thus TX interrupt will be enabled after data is sent out.
  - Improvements
    - Added check for baud rate's accuracy that returns kStatus_USART_BaudrateNotSupport

when the best achieved baud rate is not within 3% error of configured baud rate.

- 2.3.0
  - New Features
    - Added APIs to configure 9-bit data mode, set slave address and send address.
    - Modified USART_TransferReceiveNonBlocking and USART_TransferHandleIRQ to use 9-bit mode in multi-slave system.
- 2.2.0
  - New Features
    - Added the feature of supporting USART working at 32 kHz clocking mode.
  - Improvements
    - Modified USART_TransferHandleIRQ so that txState will be set to idle only when all data has been sent out to bus.
    - Modified USART_TransferGetSendCount so that this API returns the real byte count that USART has sent out rather than the software buffer status.
    - Added timeout mechanism when waiting for certain states in transfer driver.
  - Bug Fixes
    - Fixed MISRA 10.1 issues.
    - Fixed bug that operation on INTENSET, INTENCLR, FIFOINTENSET and FIFOINTENCLR should use bitwise operation not 'or' operation.
    - Fixed bug that if rx interrupt occurrs before TX interrupt is enabled and after txDataSize is configured, the data will be sent early by mistake, thus TX interrupt will be enabled after data is sent out.
- 2.1.1
  - Improvements
    - Added check for transmitter idle in USART_TransferHandleIRQ and USART_TransferSendDMACallback to ensure all the data would be sent out to bus.
    - Modified USART_ReadBlocking so that if more than one receiver errors occur, all status flags will be cleared and the most severe error status will be returned.
  - Bug Fixes
    - Eliminated IAR Pa082 warnings.
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 10.8, 11.3, 11.6, 11.8, 11.9, 13.5.
- 2.1.0
  - New Features
    - Added features to allow users to configure the USART to synchronous transfer(master and slave) mode.
  - Bug Fixes
    - Modified USART_SetBaudRate to get more acurate configuration.
- 2.0.3
  - New Features
    - Added new APIs to allow users to enable the CTS which determines whether CTS is used for flow control.
- 2.0.2
  - Bug Fixes
    - Fixed the bug where transfer abort APIs could not disable the interrupts. The FIFOINT-

    - ENSET register should not be used to disable the interrupts, so use the FIFOINTENCLR register instead.
- 2.0.1
  - Improvements
    - Unified the component full name to FLEXCOMM USART (DMA/FREERTOS) driver.
- 2.0.0
  - Initial version.

USART_DMA

The current USART_DMA driver version is 2.6.0.

- 2.6.0
  - Refer USART driver change log 2.0.1 to 2.6.0

USART_FREERTOS

The current USART_FREERTOS driver version is 2.6.0.

- 2.6.0
  - Refer USART driver change log 2.0.1 to 2.6.0

GINT

The current GINT driver version is 2.1.0.

- 2.1.0
  - Improvements
    - Updated for platforms which only has one port.
- 2.0.3
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.8.
- 2.0.2
  - Bug Fixes
    - Fixed issue for MISRA-2012 check.
      - Fixed rule 17.7.
- 2.0.1
  - Added control macro to enable/disable the RESET and CLOCK code in current driver.
- 2.0.0
  - Initial version.

IAP

The current IAP driver version is 2.1.5.

- 2.1.5
  - Improvements
    - Update Flash_Program src parameter to const.
    - Check CPU frequency <= 100MHZ for Flash Erase and Program.
    - Add BOOTLOADER_UserEntry API.
- 2.1.4
  - Bug Fixes
    - Fixed misra issue.
- 2.1.3
  - Bug Fixes
    - Fix the CFPA version wasn't transferred into SDK driver.
- 2.1.2
  - Bug Fixes
    - Fix IAP driver status definitions don't match ROM_API.pdf from User Manual.
- 2.1.1
  - Bug Fixes
    - The last 17 pages are reserved for chips with 640KB flash.
- 2.1.0
  - New Features
    - Added new API FLASH_Read for users to read flash.
    - Added new API skboot_authenticate for image authentication api.
    - Added new AP kb_init, kb_deinit, kb_execute for users to operate BOOT ROM.
- 2.0.3
  - Bug Fixes
    - Resolve incompatibility issue.
- 2.0.2
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 11.1.
  - Improvements
    - Improved the format of IAP driver version, using versionMajor to obtain the major version of bootloader.
- 2.0.1
  - Improvements
    - Removed the enumeration item kSysToFlashFreq_100MHz which cannot be supported.
    - Removed the invalid FFR commands.
    - Improved the format of IAP driver version, using S_VersionMajor to obtain the major version of bootloader.
- 2.0.0
  - Initial version.

INPUTMUX

The current INPUTMUX driver version is 2.0.7.

- 2.0.7
  - Improvements
    - Release peripheral from reset if necessary in init function.
- 2.0.6
  - Bug Fixes
    - Fixed the documentation wrong in API INPUTMUX_AttachSignal.
- 2.0.5
  - Bug Fixes
    - Fixed build error because some devices has no sct.
- 2.0.4
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rule 10.4, 12.2 in INPUTMUX_EnableSignal() function.
- 2.0.3
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.4, 10.7, 12.2.
- 2.0.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.4, 12.2.
- 2.0.1
  - Support channel mux setting in INPUTMUX_EnableSignal().
- 2.0.0
  - Initial version.

LPADC

The current LPADC driver version is 2.8.4.

- 2.8.4
  - Bug Fixes
    - Remove function ’LPADC_SetOffsetValue’ assert statement, this statement may cause runtime errors in existing code.
- 2.8.3
  - Bug Fixes
    - Fixed SDK lpadc driver examples compile issue, move condition ’commandId < ADC_-CV_COUNT’ to a more appropriate location.
- 2.8.2
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rule 18.1, 10.3, 10.1 and 10.4.
- 2.8.1
  - Bug Fixes

    - Fixed LPADC sample mode enum name mistake.
- 2.8.0
  - Improvements
    - Release peripheral from reset if necessary in init function.
  - Bug Fixes
    - Fixed function LPADC_GetConvResult() issue.
    - Fixed function LPADC_SetConvCommandConfig() bugs.
- 2.7.2
  - Improvements
    - Use feature macros instead of header file macros.
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rule 10.1, 10.3, 10.4 and 14.3.
- 2.7.1
  - Improvements
    - Corrected descriptions of several functions.
    - Improved function LPADC_GetOffsetValue and LPADC_SetOffsetValue.
    - Revert changes of feature macros for lpadc.
    - Use feature macros instead of header file macros.
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rule 10.8.
    - Fixed the violations of MISRA C-2012 rule 10.1, 10.3, 10.4 and 14.3.
- 2.7.0
  - Improvements
    - Added supports of CFG2 register.
    - Removed some useless macros.
- 2.6.2
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rules.
    - Fixed LPADC driver code compile error issue.
- 2.6.1
  - Improvements
    - Updated the use of macros in the driver code.
- 2.6.0
  - Improvements
    - Added the API LPADC_SetOffset12BitValue() to configure 12bit ADC conversion offset trim value manually.
    - Added the API LPADC_SetOffset16BitValue() to configure 16bit ADC conversion offset trim value manually.
    - Added API to set offset calibration mode.
    - Added configuration of alternate channel.
    - Updated auto calibration API and added calibration value conversion API.
  - New feature
    - Added API LPADC_EnableHardwareTriggerCommandSelection() to enable trigger commands controlled by ADC_ETC.
    - Updated LPADC_DoAutoCalibration() to allow doing something else before the ADC

    - inititialization to be totally complete. Enhance initialization duration time of the ADC.
    - Added two new APIs to get/set calibration value.
- 2.5.2
  - Improvements
    - Added while loop, LPADC_GetConvResult() will return only when the FIFO will not be empty.
- 2.5.1
  - Bug Fixes
    - Fixed some typos in Lpadc driver comments.
- 2.5.0
  - Improvements
    - Added missing items to enable trigger interrupts.
- 2.4.0
  - New features
    - Added APIs to get/clear trigger status flags.
- 2.3.0
  - Improvements
    - Removed LPADC_MeasureTemperature() function for the LPADC supports different temperature sensor calculation equations.
- 2.2.1
  - Improvements
    - Optimized LPADC_MeasureTemperature() function to support the specific series with flash solidified calibration value.
    - Clean doxygen warnings.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.3, rule 10.8 and rule 17.7.
- 2.2.0
  - New Feature
    - Added API LPADC_MeasureTemperature() to get correct temperature from the internal sensor.
  - Improvements
    - Separated lpadc_conversion_resolution_mode_t with related feature macro.
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rules:
      - Rule 10.3, 10.4, 10.6, 10.7 and 17.7.
- 2.1.1
  - Improvements
    - 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 removed invalid macros.

- 2.0.2
  - Improvements
    - Added support for platforms with 2 FIFOs and different calibration measures.
- 2.0.1
  - Bug Fixes
    - Ensured the API LPADC_SetConvCommandConfig configure related registers correctly.
- 2.0.0
  - Initial version.

CRC

The current CRC driver version is 2.1.1.

- 2.1.1
  - Fix MISRA issue.
- 2.1.0
  - Add CRC_WriteSeed function.
- 2.0.2
  - Fix MISRA issue.
- 2.0.1
  - Fixed KPSDK-13362. MDK compiler issue when writing to WR_DATA with -O3 optimize for time.
- 2.0.0
  - Initial version.

DMA

The current DMA driver version is 2.5.1.

- 2.5.1
  - Bug Fixes
    - Fixed violation of the MISRA C-2012 rule 11.6.
- 2.5.0
  - Improvements
    - Added a new api DMA_SetChannelXferConfig to set DMA xfer config.
- 2.4.4
  - Bug Fixes
    - Fixed the issue that DMA_IRQHandle might generate redundant callbacks.
    - Fixed the issue that DMA driver cannot support channel bigger then 32.
    - Fixed violation of the MISRA C-2012 rule 13.5.
- 2.4.3
  - Improvements
    - Added features FSL_FEATURE_DMA_DESCRIPTOR_ALIGN_SIZEn/FSL_FEATURE_DMA0_DESCRIPTOR_ALIGN_SIZE/FSL_FEATURE_DMA1_DESCRIPTOR_-

ALIGN_SIZE to support the descriptor align size not constant in the two instances.

- 2.4.2
  - Bug Fixes
    - Fixed violation of the MISRA C-2012 rule 8.4.
- 2.4.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 5.7, 8.3.
- 2.4.0
  - Improvements
    - Added new APIs DMA_LoadChannelDescriptor/DMA_ChannelIsBusy to support polling transfer case.
  - Bug Fixes
    - Added address alignment check for descriptor source and destination address.
    - Added DMA_ALLOCATE_DATA_TRANSFER_BUFFER for application buffer allocation.
    - Fixed the sign-compare warning.
    - Fixed violations of the MISRA C-2012 rules 18.1, 10.4, 11.6, 10.7, 14.4, 16.3, 20.7, 10.8, 16.1, 17.7, 10.3, 3.1, 18.1.
- 2.3.0
  - Bug Fixes
    - 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 configuring wrap descriptor.
    - Added wrap support in function DMA_SubmitChannelTransfer.
- 2.2.4
  - Bug Fixes
    - Fixed the issue that macro DMA_CHANNEL_CFER used wrong parameter to calculate DSTINC.
- 2.2.3
  - Bug Fixes
    - Improved DMA driver Deinit function for correct logic order.
  - Improvements
    - Added API DMA_SubmitChannelTransferParameter to support creating 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 Fixes
    - Do not use software trigger when hardware trigger is enabled.
- 2.2.1
  - Bug Fixes
    - Fixed Coverity issue.

- 2.2.0
  - Improvements
    - Changed API DMA_SetupDMADescriptor to non-static.
    - Marked APIs below as deprecated.
      - DMA_PrepareTransfer.
      - DMA_Submit transfer.
    - Added new APIs as below:
      - DMA_SetChannelConfig.
      - DMA_PrepareChannelTransfer.
      - DMA_InstallDescriptorMemory.
      - DMA_SubmitChannelTransfer.
      - DMA_SetChannelConfigValid.
      - DMA_DoChannelSoftwareTrigger.
      - DMA_LoadChannelTransferConfig.
- 2.0.1
  - Improvements
    - Added volatile for DMA descriptor member xfercfg to avoid optimization.
- 2.0.0
  - Initial version.

GPIO

The current GPIO driver version is 2.1.7.

- 2.1.7
  - Improvements
    - Enhanced GPIO_PinInit to enable clock internally.
- 2.1.6
  - Bug Fixes
    - Clear bit before set it within GPIO_SetPinInterruptConfig() API.
- 2.1.5
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 3.1, 10.6, 10.7, 17.7.
- 2.1.4
  - Improvements
    - Added API GPIO_PortGetInterruptStatus to retrieve interrupt status for whole port.
    - Corrected typos in header file.
- 2.1.3
  - Improvements
    - Updated "GPIO_PinInit" API. If it has DIRCLR and DIRSET registers, use them at set 1 or clean 0.
- 2.1.2
  - Improvements
    - Removed deprecated APIs.

- 2.1.1
  - Improvements
    - API interface changes:
      - Refined naming of APIs while keeping all original APIs, marking them as deprecated. Original APIs will be removed in next release. The mainin change is updating APIs with prefix of _PinXXX() and _PorortXXX
- 2.1.0
  - New Features
    - Added GPIO initialize API.
- 2.0.0
  - Initial version.

IOCON

The current IOCON driver version is 2.2.0.

- 2.2.0
  - Improvements
    - Removed duplicate macro defintions.
    - Renamed 'IOCON_I2C_SLEW' macro to 'IOCON_I2C_MODE' to match its companion 'IOCON_GPIO_MODE'. The original is kept as a deprecated symbol.
- 2.1.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.3.
- 2.1.1
  - Updated left shift format with mask value instead of a constant value to automatically adapt to all platforms.
- 2.1.0
  - Added a new IOCON_PinMuxSet() function with a feature IOCON_ONE_DIMENSION for LPC845MAX board.
- 2.0.0
  - Initial version.

RTC

The current RTC driver version is 2.2.0.

- 2.2.0
  - New Features
    - Created new APIs for the RTC driver.
      - RTC_EnableSubsecCounter
      - RTC_GetSubsecValue
- 2.1.3
  - Bug Fixes

- Fixed issue that RTC_GetWakeupCount may return wrong value.

- 2.1.2
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.1, 10.4 and 10.7.
- 2.1.1
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.3 and 11.9.
- 2.1.0
  - Bug Fixes
    - Created new APIs for the RTC driver.
      - RTC_EnableTimer
      - RTC_EnableWakeUpTimerInterruptFromDPD
      - RTC_EnableAlarmTimerInterruptFromDPD
      - RTC_EnableWakeupTimer
      - RTC_GetEnabledWakeupTimer
      - RTC_SetSecondsTimerMatch
      - RTC_GetSecondsTimerMatch
      - RTC_SetSecondsTimerCount
      - RTC_GetSecondsTimerCount
    - deprecated legacy APIs for the RTC driver.
      - RTC_StartTimer
      - RTC_StopTimer
      - RTC_EnableInterrupts
      - RTC_DisableInterrupts
      - RTC_GetEnabledInterrupts
- 2.0.0
  - Initial version.

MAILBOX

The current MAILBOX driver version is 2.3.1.

- 2.3.1
  - Improvements
    - Added support for the LPC55S66 series.
- 2.3.0
  - Improvements
    - Added support for the MCXNx4x series with new value for kMAILBOX_CM33_Core0 or kMAILBOX_CM33_Core1.
- 2.2.0
  - Improvements
    - Fixed missing conditional defines for the LPC5411x series.
- 2.1.0
  - Improvements

    - Added support for the LPC55S69 series. cpu_id parameter can be newly assigned to kMAILBOX_CM33_Core0 or kMAILBOX_CM33_Core1.
- 2.0.0
  - Initial version.

MRT

The current MRT driver version is 2.0.4.

- 2.0.4
  - Improvements
    - Don’t reset MRT when there is not system level MRT reset functions.
- 2.0.3
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.1 and 10.4.
    - Fixed the wrong count value assertion in MRT_StartTimer API.
- 2.0.2
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.4.
- 2.0.1
  - Added control macro to enable/disable the RESET and CLOCK code in current driver.
- 2.0.0
  - Initial version.

OSTIMER

The current OSTIMER driver version is 2.2.1.

- 2.2.1
  - Improvements
    - Release peripheral from reset if necessary in init function.
- 2.2.0
  - Improvements
    - Move the PMC operation out of the OSTIMER driver to board specific files.
    - Added low level APIs to control OSTIMER MATCH and interrupt.
- 2.1.2
  - Bug Fixes
    - Fixed MISRA-2012 rule 10.8.
- 2.1.1
  - Bug Fixes
    - removes the suffix ’n’ for some register names and bit fields’ names
  - Improvements
    - Added HW CODE GRAY feature supported by CODE GRAY in SYSCTRL register group.

- 2.1.0
  - Bug Fixes
    - Added a workaround to fix the issue that no interrupt was reported when user set smaller period.
    - Fixed violation of MISRA C-2012 rule 10.3 and 11.9.
  - Improvements
    - Added return value for the two APIs to set match value.
      - OSTIMER_SetMatchRawValue
      - OSTIMER_SetMatchValue
- 2.0.3
  - Bug Fixes
    - Fixed violation of MISRA C-2012 rule 10.3, 14.4, 17.7.
- 2.0.2
  - Improvements
    - Added support for OSTIMER0
- 2.0.1
  - Improvements
    - 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.

PINT

The current PINT driver version is 2.1.12.

- 2.1.12
  - Bug Fixes
    - Fixed coverity issue.
- 2.1.11
  - Bug Fixes
    - Fixed MISRA C-2012 rule 10.7 violation.
- 2.1.10
  - New Features
    - Added the driver support for MCXN10 platform with combined interrupt handler.
- 2.1.9
  - Bug Fixes
    - Fixed MISRA-2012 rule 8.4.
- 2.1.8
  - Bug Fixes
    - Fixed MISRA-2012 rule 10.1 rule 10.4 rule 10.8 rule 18.1 rule 20.9.
- 2.1.7
  - Improvements

    - Added fully support for the SECPINT, making it can be used just like PINT.
- 2.1.6
  - Bug Fixes
    - Fixed the bug of not enabling common pint clock when enabling security pint clock.
- 2.1.5
  - Bug Fixes
    - Fixed issue for MISRA-2012 check.
      - Fixed rule 10.1 rule 10.3 rule 10.4 rule 10.8 rule 14.4.
    - Changed interrupt init order to make pin interrupt configuration more reasonable.
- 2.1.4
  - Improvements
    - Added feature to control distinguish PINT/SECPINT relevant interrupt/clock configurations for PINT_Init and PINT_Deinit API.
    - Swapped the order of clearing PIN interrupt status flag and clearing pending NVIC interrupt in PINT_EnableCallback and PINT_EnableCallbackByIndex function.
    - Bug Fixes
      - Fixed 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.

PLU

The current PLU driver version is 2.2.1.

- 2.2.1
  - Bug Fixes
    - Fixed MISRA C-2012 rule 10.3 and rule 17.7.
- 2.2.0
  - Bug Fixes
    - Fixed wrong parameter of the PLU_EnableWakeIntRequest function.
- 2.1.0
  - New Features
    - Added 4 new APIs to support Niobe4's wake-up/interrupt control feature, including PLU-_GetDefaultWakeIntConfig() PLU_EnableWakeIntRequest(), PLU_LatchInterrupt() and PLU_ClearLatchedInterrupt().
  - Other Changes
    - Changed the register name LUT_INP to LUT_INP_MUX due to register map update.
- 2.0.1
  - New Features
    - Added control macro to enable/disable the RESET and CLOCK code in current driver.
- 2.0.0
  - Initial version.

POWERQUAD

The current POWERQUAD driver version is 2.2.0.

- 2.2.0
  - New Features
    - Added new API PQ_Arctan2Fixed.
- 2.1.1
  - Bug Fixes
    - Remove PQ_WaitDone from PQ_ArctanFixed and PQ_ArctanhFixed because it is unnecessary.
- 2.1.0
  - Improvements
    - Fixed typo issue for biquad related function name.
    - Changed operator from "%" into "&" to reduce heavy cycle for biquad functions.
- 2.0.5
  - Improvements
    - Added a note in driver for FIR that powerquad has a hardware limitation, when using it for FIR increment calculation, the address of pSrc needs to be a continuous address.
- 2.0.4
  - Improvements
    - Supported the platforms which don't have PowerQuad clock and reset control.

- 2.0.3
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 8.4, 10.1, 10.3, 10.4, 10.6, and so on.
- 2.0.2
  - Bug Fixes
    - Fixed array size issue in fsl_powerquad_data.h file.
    - Fixed vector function pipeline issue.
- 2.0.1
  - Bug Fixes
    - Fixed build error in C++ mode.
- 2.0.0
  - Initial version.

PRINCE

The current PRINCE driver version is 2.5.1.

- Version 2.6.0
  - Renamed CSS to ELS.
- Version 2.5.1
  - Fix build error due to renamed symbols.
- Version 2.3.2
  - Fix documentation of enumeration.
  - Extend PRINCE example.
- Version 2.3.1
  - Fix MISRA-2012 issues.
  - Add support for LPC55S0x series
- Version 2.3.0
  - Add support for LPC55S1x and LPC55S2x series
- Version 2.2.0
  - Add runtime checking of the A0 and A1 rev. of LPC55Sxx serie to support both silicone revisions.
- Version 2.1.0
  - Update for the A1 rev. of LPC55Sxx serie.
- 2.0.0
  - Initial version.

PUF

The current PUF driver version is 2.1.6.

- 2.1.6
  - Changed wait time in PUF_Init(), when initialization fails it will try PUF_Powercycle() with shorter time. If this shorter time will also fail, initialization will be tried with worst case time

as before.

- 2.1.5
  - Use common SDK delay in puf_wait_usec().
- 2.1.4
  - Replace register uint32_t ticksCount with volatile uint32_t ticksCount in puf_wait_usec() to prevent optimization out delay loop.
- 2.1.3
  - Fix MISRA C-2012 issue.
- 2.1.2
  - Update: Add automatic big to little endian swap for user (pre-shared) keys destinated to secret hardware bus (PUF key index 0).
- 2.1.1
  - Fix ARMGCC build warning .
- 2.1.0
  - Align driver with PUF SRAM controller registers on LPCXpresso55s16.
  - Update initizalition logic .
- 2.0.3
  - Fix MISRA C-2012 issue.
- 2.0.2
  - New feature:
    * Add PUF configuration structure and support for PUF SRAM controller.
  - Improvements:
    * Remove magic constants.
- 2.0.1
  - Bug Fixes:
    * Fixed puf_wait_usec function optimization issue.
- 2.0.0
  - Initial version.

SCTIMER

The current SCTimer driver version is 2.4.9.

- 2.4.9
  - Improvements
    * Supported platforms which don't have system level SCTIMER reset.
- 2.4.8
  - Bug Fixes
    * Fixed the issue that the SCTIMER_UpdatePwmDutycycle() can't writes MATCH_H bit and RELOADn_H.
- 2.4.7
  - Bug Fixes
    * Fixed the issue that the SCTIMER_UpdatePwmDutycycle() can't configure 100% duty cycle PWM.

- 2.4.6
  - Bug Fixes
    - Fixed the issue where the H register was not written as a word along with the L register.
    - Fixed the issue that the SCTIMER_SetCOUNTValue() is not configured with high 16 bits in unify mode.
- 2.4.5
  - Bug Fixes
    - Fix SCT_EV_STATE_STATEMSKn macro build error.
- 2.4.4
  - Bug Fixes
    - Fix MISRA C-2012 issue 10.8.
- 2.4.3
  - Bug Fixes
    - Fixed the wrong way of writing CAPCTRL and REGMODE registers in SCTIMER_SetupCaptureAction.
- 2.4.2
  - Bug Fixes
    - Fixed SCTIMER_SetupPwm 100% duty cycle issue.
- 2.4.1
  - Bug Fixes
    - Fixed the issue that MATCHn_H bit and RELOADn_H bit could not be written.
- 2.4.0
- 2.3.0
  - Bug Fixes
    - Fixed the potential overflow issue of pulseperiod variable in SCTIMER_SetupPwm/SCTIMER_UpdatePwmDutycycle API.
    - Fixed the issue of SCTIMER_CreateAndScheduleEvent API does not correctly work with 32 bit unified counter.
    - Fixed the issue of position of clear counter operation in SCTIMER_Init API.
  - Improvements
    - Update SCTIMER_SetupPwm/SCTIMER_UpdatePwmDutycycle to support generate 0% and 100% PWM signal.
    - Add SCTIMER_SetupEventActiveDirection API to configure event activity direction.
    - Update SCTIMER_StartTimer/SCTIMER_StopTimer API to support start/stop low counter and high counter at the same time.
    - Add SCTIMER_SetCounterState/SCTIMER_GetCounterState API to write/read counter current state value.
    - Update APIs to make it meaningful.
      - SCTIMER_SetEventInState
      - SCTIMER_ClearEventInState
      - SCTIMER_GetEventInState
- 2.2.0
  - Improvements
    - Updated for 16-bit register access.
- 2.1.3

  - Bug Fixes
    - Fixed the issue of uninitialized variables in SCTIMER_SetupPwm.
    - Fixed the issue that the Low 16-bit and high 16-bit work independently in SCTIMER driver.
  - Improvements
    - Added an enumerable macro of unify counter for user.
      - kSCTIMER_Counter_U
    - Created new APIs for the RTC driver.
      - SCTIMER_SetupStateLdMethodAction
      - SCTIMER_SetupNextStateActionwithLdMethod
      - SCTIMER_SetCOUNTValue
      - SCTIMER_GetCOUNTValue
      - SCTIMER_SetEventInState
      - SCTIMER_ClearEventInState
      - SCTIMER_GetEventInState
    - Deprecated legacy APIs for the RTC driver.
      - SCTIMER_SetupNextStateAction
- 2.1.2
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.3, 10.4, 10.6, 10.7, 11.9, 14.2 and 15.5.
- 2.1.1
  - Improvements
    - Updated the register and macro names 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
  - New Features
    - Added control macro to enable/disable the RESET and CLOCK code in current driver.
- 2.0.0
  - Initial version.

RNG

The current RNG driver version is 2.0.3.

- 2.0.3
  - Modified RNG_Init and RNG_GetRandomData functions, added rng_accumulateEntropy and rng_readEntropy functions. These changes are reflecting recommended usage of RNG according to device UM

- 2.0.2
  - Add RESET_PeripheralReset function inside RNG_Init and RNG_Deinit functions.
- 2.0.1
  - Fix MISRA C-2012 issue.
- 2.0.0
  - Initial version.

SDIF

The current SDIF driver version is 2.1.0

- 2.1.0
  - Improvements
    - Removed reduntant member endianMode in sdif_config_t.
    - Added error status check in function SDIF_WaitCommandDone.
    - Fixed the read fifo data incomplete issue in interrupt non-dma mode.
- 2.0.15
  - Bug Fixes
    - Cleared the interrupt status before enable the interrupt to avoid interrupt generate unexpectedly.
    - Fixed the SDIF_ReadDataPortBlocking blocking at wrong condition issue.
  - Improvements
    - Enabled the functionality of timeout parameter in SDIF_SendCommand.
    - Added the error recovery while sending sync clock command timeout.
- 2.0.14
  - Improvements
    - Used different status code for command and data interrupt callback.
  - Bug Fixes
    - Fixed the DMA descriptor attribute field unreset when configuing the current transfer D-MA descriptor issue which may cause the transfer terminate unexpected.
- 2.0.13
  - Improvements
    - Disabled redundant interrupt per different transfer request.
    - Disabled interrupt and reset command/data pointer in handle when transfer completes.
  - Bug Fixes
    - Fixed the PA082 build warning.
    - Fixed violations of the MISRA C-2012 rules 14.4, 17.7, 10.4, 10.3, 10.8, 14.3, 10.1, 16.4, 15.7, 12.2, 11.3, 11.9.
- 2.0.12
  - Bug Fixes
    - Fixed the issue that SDIF_ConfigClockDelay didn't reset the delay field before write.
    - Removed useless fifo reset code in transfer function.
    - Fixed the divider overflow issue in function SDIF_SetCardClock.
- 2.0.11

- Improvements
  - Added API SDIF_GetEnabledInterruptStatus/SDIF_GetEnabledDMAInterruptStatus and used in SDIF_TransferHandleIRQ.
  - Removed useless members interruptFlags/dmaInterruptFlags in the sdif_handle_t.
  - Improved SDIF_SendCommand with return success directly when timeout is 0.
  - Added timeout error check when sending update clock command in SDIF_SetCardClock.
  - Removed START_CMD status polling for normal command sending in SDIF_TransferBloking/SDIF_TransferNonBlocking.
  - Disabled timeout parameter in function SDIF_SendCommand.
- Bug Fixes
  - Added delay cycle for the default speed mode(400 K and 25 M) to fix the timing issue when different AHB clocks are configured.

- 2.0.10
  - Bug Fixes
    - Fixed the issue that API SDIF_EnableCardClock could not clear the clock enable bit.
- 2.0.9
  - Bug Fixes
    - Fixed MDK 66-D warning.
- 2.0.8
  - New Features
    - Added control macro to enable/disable the RESET and CLOCK code in current driver.
    - Disabled useless interrupt while DMA is used.
    - Updated SDIF driver for one instance support two cards.
- 2.0.7
  - Bug Fixes
    - Enlarged the timeout value to avoid a command conflict issue.
- 2.0.6
  - Bug Fixes
    - Removed assert(srcClock_Hz <= FSL_FEATURE_SDIF_MAX_SOURCE_CLOCK).
    - Used hardware reset instead of software reset during initialization.
- 2.0.5
  - New Features
    - Added non-word aligned data address and DMA descriptor address transfer support. Once one of the above addresses is not aligned, switch to host transfer mode.
  - Bug Fixes
    - Fixed the issue that DMA suspended during initialization.
    - Removed useless memset function call.
- 2.0.4
  - Improvements
    - Added cardInserted/cardRemoved callback function.
    - Added host base address/user data parameter for all call back functions.
- 2.0.3
  - Improvements
    - Improved Clock Delay macro to allow the user to redefine and remove useless delay for clock below 25 MHz.

- 2.0.2
  - Bug Fixes
    - Fixed the issue that the status flag could not be cleared entirely after transfer complete.
- 2.0.1
  - New Features
    - Improved interrupt transfer callback.
  - Bug Fixes
    - Added assert to limit the SDIF source clock below 52 MHz.
- 2.0.0
  - Initial version.

SYSCTL

The current SYSCTL driver version is 2.0.5.

- 2.0.5
  - Bug Fixes:
    - Fixed violations of MISRA C-2012 rule 8.3, 10.1, 10.4, 10.7.
- 2.0.4
  - Improvements:
    - Update macro name to align with the header of devices.
- 2.0.3
  - Improvements:
    - Update the register and macro name to align with the header of devices.
- 2.0.2
  - Removed kSYSCTL_Flexcomm3DataOut enumeration definition.
- 2.0.1
  - Fixed some typo error comments and improved driver integral ability.
- 2.0.0
  - Initial version.

UTICK

The current UTICK driver version is 2.0.5.

- 2.0.5
  - Improvements
    - Improved for SOC RW610.
- 2.0.4
  - Bug Fixes
    - Fixed compile fail issue of no-supporting PD configuration in utick driver.
- 2.0.3
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rules: 8.4, 14.4, 17.7

- 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.

WWDT

The current WWDT driver version is 2.1.9.

- 2.1.9
  - Bug Fixes
    - Fixed violation of the MISRA C-2012 rule 10.4.
- 2.1.8
  - Improvements
    - Updated the "WWDT_Init" API to add wait operation. Which can avoid the TV value read by CPU still be 0xFF (reset value) after WWDT_Init function returns.
- 2.1.7
  - Bug Fixes
    - Fixed the issue that the watchdog reset event affected the system from PMC.
    - Fixed the issue of setting watchdog WDPROTECT field without considering the backwards compatibility.
    - Fixed the issue of clearing bit fields by mistake in the function of WWDT_ClearStatusFlags.
- 2.1.5
  - Bug Fixes
    - deprecated a unusable API in WWWDT driver.
      - WWDT_Disable
- 2.1.4
  - Bug Fixes
    - Fixed violation of the MISRA C-2012 rules Rule 10.1, 10.3, 10.4 and 11.9.
    - Fixed the issue of the inseparable process interrupted by other interrupt source.
      - WWDT_Init
- 2.1.3
  - Bug Fixes
    - Fixed legacy issue when initializing the MOD register.
- 2.1.2
  - Improvements
    - Updated the "WWDT_ClearStatusFlags" API and "WWDT_GetStatusFlags" API to match QN9090. WDTOF is not set in case of WD reset. Get info from PMC instead.
- 2.1.1
  - New Features

    - Added new feature definition macro for devices which have no LCOK control bit in MOD register.
    - Implemented delay/retry in WWDT driver.
- 2.1.0
  - Improvements
    - Added new parameter in configuration when initializing WWDT module. This parameter, which must be set, allows the user to deliver the WWDT clock frequency.
- 2.0.0
  - Initial version.

2 Middleware Change Log

emWin library

The currently supported version is 6.34c

- v6.34c
  - upgraded to v6.34c
- v6.28_rev1
  - add cm33_nodsp_fpu libraries for Cortec M33 without DSP extension with SP FPU
- v6.28
  - upgraded to v6.28
- v6.24_rev2
  - add cm33_nodsp libraries for Cortex M33 without DSP extension
- v6.24_rev1
  - recompiled cm33 library with fpu single precision
  - added cm7_sp library for Cortex M7 with sp fpu for IAR
- v6.24
  - upgraded to v6.24
- v6.16c
  - upgraded to v6.16c
  - updated temperature_control demo generated by AppWizard
- v6.14d
  - upgraded to v6.14d
- v6.10f
  - upgraded to v6.10f

Embedded Wizard GUI library

The currently supported version is 10.00.

10.00_rev0

- update to version 10.00
- add support for RT1010, RT1170, LPC55S69, RT1040

9.30_rev0

- Remove conditional include of RTE_Components.h in ewdef.h

9.20_rev0

- Fix build warnings in tlsf.c, DeviceDriver.h, remove conditional include in ewextgfx.h

FatFs for MCUXpresso SDK

Current version is FatFs R0.15_rev0.

- R0.15_rev0
  - Upgraded to version 0.15
  - Applied patches from http://elm-chan.org/fsw/ff/patches.html
- R0.14b_rev1
  - Applied patches from http://elm-chan.org/fsw/ff/patches.html
- R0.14b_rev0
  - Upgraded to version 0.14b
- R0.14a_rev0
  - Upgraded to version 0.14a
  - Applied patch ff14a_p1.diff and ff14a_p2.diff
- R0.14_rev0
  - Upgraded to version 0.14
  - Applied patch ff14_p1.diff and ff14_p2.diff
- R0.13c_rev0
  - Upgraded to version 0.13c
  - Applied 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.

FreeMASTER Communication Driver

Current version is 3.0.6. Visit https://www.nxp.com/freemaster for more information. Reach out for a support at https://community.nxp.com/community/freemaster.

- 3.0.0
  - Initial version of FreeMASTER driver reworked from a standalone package to MCUXpresso

  SDK middleware.
  - This driver version supports new version V4 of FreeMASTER serial communication protocol.
  - Supports UART, LPUART, USART, MINIUSART, FlexCAN, USB-CDC and JTAG/BDM communication.
  - Initial version was tested with the following boards: evkmimxrt1060, frdmk64f, frdmke15z, frdmkl28z, lpcxpresso54628 lpcxpresso55s69, lpcxpresso845max and twrk64f120m.
  - Use with FreeMASTER PC Host tool version 2.5 or later.
- 3.0.1
  - FreeMASTER driver extended to support wide range of Kinetis, LPC and i.MX-RT platforms.
  - Low-level communication drivers also available for few non-SDK NXP platforms like S12Z, S32x and more.
  - Use with FreeMASTER PC Host tool version 3.0 or later.
- 3.0.2
  - FreeMASTER driver support of DSC56F800EX and S12 platforms extended.
  - Removed dependency on C99 compiler features.
  - Use with FreeMASTER PC Host tool version 3.0.2 or later.
- 3.0.3
  - General update for SDK 2.9.0
  - fmstr_any demo added to selected platforms - use with MCUXpresso SDK and FreeMASTER peripheral configuration tool.
  - New example.pmp project file embedded into application flash storage.
  - USB-CDC implementation fixed, new JTAG EOnCE communication interface added to DSC 56F800E family.
  - Use with FreeMASTER PC Host tool version 3.0.3 or later. Version 3.1.x is recommended.
- 3.0.4
  - Fixed component dependency logic of FreeMASTER driver.
  - Use with FreeMASTER PC Host tool version 3.1.x
- 3.0.5
  - General update for SDK 2.11 and 2.12
  - New TCP and UDP support with lwIP stack
  - New communication over Segger RTT interface
  - Add fmstr_net and fmstr_wifi examples for selected i.MX-RT platforms
  - Add fmstr_rtt example for selected platforms
  - Fixed negative recorder threshold trigger processing
- 3.0.6
  - General update for SDK 2.13
  - Use of new Ethernet MDIO driver concept.
  - Support of ENET and NETC Ethernet modules in the fmstr_net example application.
- 3.0.7
  - General update for SDK 2.14

LVGL for KSDK

- 8.3.10_rev1

  - Integrate LVGL 8.3.10 to SDK.
- 8.3.9_rev1
  - Integrate LVGL 8.3.9 to SDK.
- 8.3.5_rev1
  - Integrate LVGL 8.3.5 to SDK.
- 8.3.2_rev1
  - Integrate LVGL 8.3.2 to SDK.
- 8.3.0_rev1
  - Integrate LVGL 8.3.0 to SDK.
- 8.2.0_rev1
  - Integrate LVGL 8.2.0 to SDK.
- 8.0.2_rev1
  - Integrate LVGL 8.0.2 to SDK.
- 7.10.1_rev1
  - Integrate LVGL 7.10.1 to SDK.
  - Added PXP, VGLite hardware acceleration.
- 7.4.0_rev1
  - Integrate LVGL 7.4.0 to SDK.
- 7.0.0_rev1
  - Integrate LVGL 7.0.0 to SDK.
  - Added PXP hardware acceleration initial version.
- 6.1.1_rev1
  - Integrate LVGL 6.1.1 to SDK.
- 5.3_rev1
  - Integrate LVGL 5.3 to SDK.

lwIP for MCUXpresso SDK

Lightweight IP (lwIP) is a small independent implementation of the TCP/IP protocol suite. Source code included in this SDK is based on version 2.2.0 taken from 3rd party lwIP GIT repository. The webpage https://git.savannah.nongnu.org/cgit/lwip.git allows to browse the repository and also contains URLs for its cloning. The development versions (X.Y.Z.dev) do not refer to a single source code snapshots. To avoid ambiguity, change log below contains SHA-1 hashes of GIT commits used when importing the code into the SDK.

- 2.2.0_rev10
  - New features:
    - Ported lwIP 2.2.0 (2023-09-25, branch: master, SHA-1: 0a0452b2c39bdd91e252aef045c115f88f6ca7 tag: STABLE-2_2_0_RELEASE) to MCUXpresso SDK.
    - Enabled hardware-accelerated CRC computation and verification (MAC, IPv4, TCP, UDP, ICMPv4, ICMPv6) for ENET Kinetis, ENET QoS and ENET LPC.
    - Enabled link state detection based on PHY interrupts. The ETH_LINK_POLLING_INTERVAL_MS macro controls this - setting it to 0 and specifying ethernetif_config_t->phyIntGpio enables it, setting it to a value greater than zero enables polling instead. Supported

only under an RTOS (NO_SYS == 0). By default, the link state is polled.

    - ND6: Implemented RFC 4191 type C host, which means default router list (learned from Router Advertisement messages) has been replaced with routing table, which contains default route records for each router and also routes learned from received Route Information Options. Changes partially based on https://savannah.nongnu.org/patch/?10114. The option LWIP_ND6_NUM_ROUTERS has been removed, and the new option LWIP_ND6_NUM_ROUTES has been added to configure the size of the routing table.
    - IPv6: Implemented a new hook - LWIP_HOOK_IP6_CANFORWARD. This hook can be used, for example, for multicast forwarding between netifs. Defining this hook enables multicast traffic forwarding, thus the hook is also invoked for multicast traffic.
    - MLD6: Multicast Listener Discovery v1 replaced by v2 (RFC 3810) but without support of source specific multicast.
    - port/enet_ethernetif_kinetis.c: Added check to generate/validate ICMPv6 checksum in SW as the Kinetis ENET peripheral does not do it.
    - Added disabling of Rx interrupt when the port is out of Rx buffers. See port/README.md for more details.
  - Bug fixes:
    - src/apps/lwiperf: Fixed access to invalid data when UDP report is to be sent from a timer but abort has been called before.
    - src/apps/lwiperf: Fixed deallocation of TCP server started by client (in reverse or dual modes) which failed to connect.
    - port/netc_ethernetif.c: Fixed cache control enablement macro (FSL_SDK_ENABLE_DRIVER_CACHE_CONTROL > FSL_ETH_ENABLE_CACHE_CONTROL).
    - port/sys_arch.c: The function sys_assert does not call portENTER_CRITICAL when called from an interrupt.
    - src/core/ipv4/ip4.c: Fixed checksum reset condition.
    - ND6:
      - Lladdr length is now taken from netif->hwaddr_len so ND6 works properly regardless of NETIF_MAX_HWADDR_LEN.
      - Added check of sufficient length of lladdr options from incoming messages.
    - src/apps/httpsrv/httpsrv.c: Fixed hangup in HTTPSRV_release if caller's task has higher priority than server task.
  - port/arch/cc.h: LWIP_PLATFORM_DIAG is defined (and can be overridden) independently of the LWIP_DEBUG setting. Removed printing extra newline symbols from LWIP_PLATFORM_DIAG.
  - src/apps/lwiperf: The "end of test" UDP datagram is resent more often. This increases the probability of the server to receive it and end the test when datagrams are getting lost.
  - Added port/README.md describing possible settings and helper functions in the port layer.
- 2.2.0_rev9
  - New features:
    - Ported lwIP 2.2.0.dev (2023-01-03, branch: master, SHA-1: 3fe8d2fc43a9b69f7ed28c63d44a7744f9c to MCUXpresso SDK.
    - Applied patch to allow sending IPv6 router advertisement. Improved to allow selection of interface and router life time and to allow sending route information options.

    - src/apps/lwiperf: Support for reverse test (client receives, server sends). Requires iperf version 2.1.0 or newer.
  - Bug fixes:
    - src/apps/httpsrv: Fixed operation with LWIP_IPV6 enabled. Server can be also accessed using both IPv4 and IPv6 at the same time if compiled with both LWIP_IPV4=1 and LWIP_IPV6=1. Note the type of the field struct httpsrv_param_struct.address has changed from struct sockaddr to struct sockaddr_storage.
- 2.2.0_rev8
  - New features:
    - src/apps/lwiperf: Added new parameter "buffer_len" to functions lwiperf_start_tcp_client() and lwiperf_start_udp_client() to configure TCP/UDP packet size.
    - src/apps/lwiperf: Added new parameter "tos" to functions lwiperf_start_tcp_client() to configure TCP packet priority.
    - NETC adaptation layer: Not forcing the RX/TX buffers placement in non-cacheable memory. Requires the symbol FSL_ETH_ENABLE_CACHE_CONTROL to be defined on project level if the memory region, where the buffers are placed by a linker, has cache enabled.
  - Bug fixes:
    - src/apps/httpsrv: Added missing includes.
    - src/apps/lwiperf: Fixed TCP client to send settings at the beginning of each 128 KB block like the PC iperf 2.0.x application does.
    - src/apps/lwiperf: Fixed validation of TCP received data (with LWIPERF_CHECK_RX_DATA enabled, works with iperf 2.0.x).
    - src/apps/lwiperf: Fixed lwiperf_list_remove() to clear references to the removed item.
  - src/apps/lwiperf: Program does not assert when buffer cannot be cloned in UDP test, only "can't clone buffer" message is printed.
- 2.2.0_rev7
  - New features:
    - Ported lwIP 2.2.0.dev (2022-05-09, branch: master, SHA-1: 239918ccc173cb2c2a62f41a40fd893f57f to MCUXpresso SDK.
    - Added function ethernetif_probe_link() which reads actual link, speed and duplex settings from phy and passes them to driver. Stack could be set to call this function periodically by setting ETH_LINK_POLLING_INTERVAL_MS to value higher than zero.
    - Added helper functions ethernetif_wait_linkup() and ethernetif_wait_ipv4_valid() to allow blocking of RTOS task or bare metal application until link is up or IPv4 address becomes valid.
    - Added NETC adaptation layer.
    - Processing of rx packets under RTOS moved from ISR to a separate task to improve system reaction times. Switch back to old behavior can be done by setting ETH_DO_RX_IN_SEPARATE_TASK macro to 0.
  - Bug fixes:
    - port: Fixed copying of pbuf contents. Previous code was using an incorrect end condition and could result in the overrun of the destination buffer if more packets were on the queue.
    - port: Delegating pbuf_free calls to tcpip_thread via pbuf_free_callback where possible (RTOS), ensured pbuf_free is not called from interrupt context when LWIP_ALLOW_-

MEM_FREE_FROM_OTHER_CONTEXT is not set (bare metal).
    - port/enet_ethernetif_qos.c - Fixed ENET_RXBD_NUM which was used instead of ENET_TXBD_NUM.
    - port/enet_ethernetif_qos.c - Fixed buffer alignment to be at least 64.
    - src/apps/lwiperf: Fixed IPv6 TCP TX throughput lower than IPv4 by modifying maximum segment size to avoid sending two segments instead of one.
    - src/apps/lwiperf: Out-of-order datagrams in UDP RX server mode are counted to the throughput.
    - src/apps/httpsrv: Implemented receive timeouts on sockets.
    - src/apps/httpsrv: Don't assert on HTTP session task creation failure.
    - src/apps/httpsrv: Fixed build with IPv6 enabled.
    - src/apps/httpsrv: Updated endianess macros required for websocket SHA generation.
    - src/apps/httpsrv: Added missing includes.
- 2.2.0_rev6
  - New features:
    - Ported lwIP 2.2.0.dev (2022-03-25, branch: master, SHA-1: 124dc0a64ef5d7c14a27e3115e5888df65 to MCUXpresso SDK.
    - Implemented leaving of multicast groups on ENET and ENET QOS.
- 2.2.0_rev5
  - New features:
    - Ported lwIP 2.2.0.dev (2021-05-11, branch: master, SHA-1: 7ec4e9be304e7f8953740f10b2c810a292 to MCUXpresso SDK.
    - LPC ENET adaptation layer allocates more buffers for frame reception now. Previously the number of receive buffers was determined by ENET_RXBD_NUM, which defaults to 5. It is determined by ENET_RXBUFF_NUM now, which is 2 ∗ ENET_RXBD_NUM by default. Increase was needed because the actual version of LPC ENET driver always hold ENET_RXBD_NUM number of buffers and few additional buffers are needed for passing zero-copy frame data to lwIP. If this takes too much memory in your application, you can counteract by decreasing PBUF_POOL_SIZE, since PBUF_POOL is used only for transmission when LPC ENET, Kinetis ENET or ENET QOS is used.
- 2.2.0_rev4
  - New features:
    - Ported lwIP 2.2.0.dev (2021-03-05, branch: master, SHA-1: 0056522cc974d2be2005c324f37187b5b3 to KSDK 2.0.0.
    - LWIP_DHCP_DOES_ACD_CHECK option default changed to 0 (disabled):
      - Although the ACD check makes getting IP address from DHCP more robust, it added several seconds delay at startup of all applications which use DHCP.
      - This feature was not present in earlier versions of lwIP.
    - ENET QOS adaptation layer - implemented zero-copy on receive.
    - Kinetis ENET and ENET QOS adaptation layers allocate more buffers for frame reception now. Previously the number of receive buffers was determined by ENET_RXBD_NUM, which defaults to 5. It is determined by ENET_RXBUFF_NUM now, which is 2 ∗ ENET_RXBD_NUM by default. Increase was needed because the actual version of Kinetis ENET and ENET QOS drivers always hold ENET_RXBD_NUM number of buffers and few additional buffers are needed for passing zero-copy frame data to lwIP. If this takes

too much memory in your application, you can counteract by decreasing PBUF_POOL_SIZE, since PBUF_POOL is used only for transmission when Kinetis ENET or ENET QOS is used.
    - Removed ethernetif_config_t.non_dma_memory field which was required to configure memory ranges unusable by ENET DMA on LPC devices. The setting has been replaced by BOARD_ENET_NON_DMA_MEMORY_ARRAY macro.

- 2.2.0_rev3
  - New features:
    - Ported lwIP 2.2.0.dev (2020-07-07, branch: master, SHA-1: c385f31076b27efb8ee37f00cb5568783a
      to KSDK 2.0.0.
- 2.2.0_rev2
  - New features:
    - Kinetis ENET adaptation layer - implemented zero-copy on receive.
    - lwiperf - counter of transferred bytes extended from 32 to 64 bit
  - Bug fixes:
    - Fixed restarting Auto IP from DHCP.
- 2.2.0_rev1
  - New features:
    - Ported lwIP 2.2.0.dev (2019-12-12, branch: master, SHA-1: 555812dcec38c9a2ef1ef9b318162915549
      to KSDK 2.0.0.
    - Implemented LWIP_ASSERT_CORE_LOCKED related functions in sys_arch.c. It can be enabled in lwipopts.h:
      - `#define LWIP_ASSERT_CORE_LOCKED() sys_check_core_locking()`
      - `#define LWIP_MARK_TCPIP_THREAD() sys_mark_tcpip_thread() // if NO_SYS == 0`
      - `#define LOCK_TCPIP_CORE() sys_lock_tcpip_core() // if NO_SYS == 0 and LWIP_TCPIP_CORE_LOCKING == 1`
      - `#define UNLOCK_TCPIP_CORE() sys_unlock_tcpip_core() // if NO_SYS == 0 and LWIP_TCPIP_CORE_LOCKING == 1`
- 2.1.2_rev5
  - New features:
    - Implemented TCP_USER_TIMEOUT socket option.
    - Implemented SIOCOUTQ ioctl.
- 2.1.2_rev4
  - New features:
    - Ported lwIP 2.1.3.dev (2019-02-27, branch: STABLE-2_1_x, SHA-1: 1bb6e7f52de1cd86be0eed31e3
      to KSDK 2.0.0.
    - Updated sys_thread_new implementation and comment.
    - Kinetis ENET adaptation layer - reading frames into a pbuf chain is conditionally compiled only when a single pbuf from pool cannot hold maximum frame size (PBUF_POOL_BUFSIZE >= maximum frame size). Avoiding this code also reduces stack size requirements by about 1.5 kilobytes.
  - Bug fixes:
    - Fixes in ethernetif_linkoutput() in enet_ethernetif_lpc.c:
      - Removed access to possibly freed pbuf.

            - Call pbuf_free() when transmit buffers not available.
            - When copying pbuf chain, updating the number of necessary transmit buffers to wait for, which can be often smaller in the copy.
        - When CGI script is reading POST data by chunks, the loop in httpsrv_read() may cause blocking in receive function waiting for more data at the end of the stream
            - HTTPSRV_cgi_read() - added limiting of the last chunk length according to content length to avoid undesired blocking
        - Applied AUTOIP patch `https://savannah.nongnu.org/patch/?9847` - with modification to support multiple network interfaces.
        - Fixed buffer overflow in httpsrv when application provided CGI script does not handle the whole content of POST request
    - Removed LwipMibCompiler contrib application as it contained LGPL licensed files in SharpSnmpLib.
- 2.1.2_rev3
    - New features:
        - lwiperf updated with UDP client/server support from the patch 9751 (`https://savannah.nongnu.org/patch/?9751`)
- 2.1.2_rev2
    - Bug fixes:
        - Fixed lwiperf_abort() in lwiperf.c to correctly close connections and free resources
- 2.1.2_rev1
    - New features:
        - Ported lwIP 2.1.2 (2018-11-22, SHA-1: 159e31b689577dbf69cf0683bbaffbd71fa5ee10) to KSDK 2.0.0.
        - Ported lwIP-contrib 2.1.0 (2018-09-24, SHA-1: 35b011d4cf4c4b480f8859c456587a884ec9d287) to KSDK 2.0.0.
- 2.0.3_rev1
    - New features:
        - Ported lwIP 2.0.3 (2017-09-15, SHA-1: 92f23d6ca0971a32f2085b9480e738d34174417b) to KSDK 2.0.0.
- 2.0.2_rev1
    - New features:
        - Ported lwIP 2.0.2 (2017-03-13, SHA-1: c0862d60746e2d1ceae69af4c6f24e469570ecef) to KSDK 2.0.0.
- 2.0.0_rev3
    - New features:
        - Ported lwIP 2.0.0 (2016-11-10, SHA-1: 216bf89491815029aa15463a18744afa04df58fe) to KSDK 2.0.0.
- 2.0.0_rev2
    - New features:
        - Ported lwIP 2.0.0 RC2 (2016-08-08, SHA-1: b1dfd00f9233d124514a36a8c8606990016f2ad4) to KSDK 2.0.0.
- 2.0.0_rev1
    - New features:
        - Ported lwIP 2.0.0 RC0 (2016-05-26) to KSDK 2.0.0.

        - Changed lwIP bare-metal examples to use poll-driven approach instead of interrupt-driven one.
- 1.4.1_rev2
    - New features:
        - Enabled critical sections in lwIP.
    - Bug fixes:
        - Fixed default lwIP packet-buffer size to be able to accept a maximum size frame from the ENET driver.
        - Fixed possible drop of multi-frame packets during transmission.
- 1.4.1_rev1
    - New features:
        - Ported lwIP 1.4.1 to KSDK 2.0.0.

mbedTLS for MCUXpresso SDK

The current version of mbedTLS is based on mbed TLS 2.28.5 branch released 2023-10-05

- 2.28.5
    - New features:
        - Ported mbedTLS 2.28.5 to SDK.
- 2.28.4
    - New features:
        - Ported mbedTLS 2.28.4 to SDK.
- 2.28.3
    - New features:
        - Ported mbedTLS 2.28.3 to SDK.
- 2.28.1
    - New features:
        - Ported mbedTLS 2.28.1 to SDK.
- 2.28.0
    - New features:
        - Ported mbedTLS 2.28.0 to SDK.
- 2.27.0
    - New features:
        - Ported mbedTLS 2.27.0 to SDK.
- 2.26.0
    - New features:
        - Ported mbedTLS 2.26.0 to SDK.
- 2.16.6_rev7
    - Bug fixes:
        - Corrected definition of global variable g_isCryptoHWInitialized to be only internal static variable in sssapi_mbedtls.c file.
- 2.16.6_rev6
    - Bug fixes:

    - Adding #ifdef in ecdsa.c to remove warning: "function "derive_mpi" was declared but never referenced", when alternative implementation of ECDSA sign and verify is used and not used Deterministic ECDSA, then was derive_mpi function never used.
- 2.16.6_rev5
  - New features:
    - Changed return type of CRYPTO_InitHardware() from void to status_t. Added check of this return value in selftest.c and benchmark.c files.
- 2.16.6_rev4
  - New features:
    - Added mutex for HW modules HASHCRYPT and CASPER. Enabled by MBEDTLS_THREADING_C
- 2.16.6_rev3
  - New features:
    - Added support for KW45 device with latest Sentinel200. Port of SSS API mbedtls implementation to KW45.
- 2.16.6_rev2
  - New features:
    - Added support for SW computing AES-192/256 while using DCP driver.
- 2.16.6_rev1
  - New features:
    - Added support for NIST P-521 elliptic curve with CASPER driver.
    - Added support for using multiple elliptic curves at once with CASPER driver.
- 2.16.6
  - New features:
    - Ported mbedTLS 2.16.6 to SDK.
- 2.16.2_rev2
  - Bug fixes:
    - Add support for HASHCRYPT context switch check, Hashcrypt without context switch is not able to calculate SHA in parallel with AES. HW acceleration of SHA is disabled by default in MbedTLS integration, enabled on chip with context switch.
- 2.16.2_rev1
  - Bug fixes:
    - Add support for CTR_DRBG using AES-128 for crypto engines without AES-256 capability.
- 2.16.2
  - New features:
    - Ported mbedTLS 2.16.2 to SDK.
- 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 /middleware/mbedtls/port/ksdk/ksdk_mbedtls.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:
    - /middleware/mbedtls/port/ksdk/des_alt.c - contains regular software implementation of DES algorithm with added MBEDTLS_DES3_SETKEY_DEC_ALT and MBEDTLS_DES3_SETKEY_ENC_ALT config parameters.
    - /middleware/mbedtls/port/ksdk/des_alt.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:
      - /middleware/mbedtls/port/ksdk/ksdk_mbedtls.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:

- /middleware/mbedtls/port/ksdk/ksdk_mbedtls.c - alternative implementation of cryptographic algorithm functions using LTC and MMCAU module drivers.
- /middleware/mbedtls/port/ksdk/ksdk_mbedtls_config.h - configuration settings used by mbedTLS KSDK bare metal examples.
- Added mbedTLS KSDK bare-metal examples:
  - /boards/<board name>/demo_apps/mbedtls/mbedtls_benchmark - KSDK mbedTLS benchmark application.
  - /boards/<board name>/demo_apps/mbedtls/mbedtls_selftest - 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 '\r' for mbedtls_printf() in self test functions.

Multicore SDK

The current version of Multicore SDK is 2.15.0

- 2.15.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.12.0
    - eRPC generator (erpcgen) v.1.12.0
    - Multicore Manager (MCMgr) v4.1.5
    - RPMsg-Lite v5.1.1
  - New features:
    - eRPC: Add dynamic/static option for transport init, GitHub PR #361.
    - eRPC: Fix receive error value for spidev, GitHub PR #363.
    - eRPC: UartTransport::init adaptation to changed driver.
    - eRPC: Fix typo in assert, GitHub PR #371.
    - eRPC,erpcgen: Move enums to enum classes, GitHub PR #379.
    - eRPC: Fixed rpmsg tty transport to work with serial transport, GitHub PR #373.
    - eRPC,erpcgen: Winsock2 support, GitHub PR #365.
    - eRPC,erpcgen: Feature/support multiple clients, GitHub PR #271.
    - eRPC,erpcgen: Feature/buffer head - Framed transport header data stored in MessageBuffer, GitHub PR #378.
    - eRPC,erpcgen: Add experimental Java support.
    - MCMgr: Added notification into MCMGR_EarlyInit and mcmgr_early_init_internal functions to avoid using uninitialized data in their implementations.
    - RPMsg-Lite: Minor changes in platform and env. layers, minor test code updates.
- 2.14.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.11.0
    - eRPC generator (erpcgen) v.1.11.0
    - Multicore Manager (MCMgr) v4.1.4
    - RPMsg-Lite v5.1.0
  - New features:
    - eRPC: Makefiles update, GitHub PR #301.
    - eRPC: Resolving warnings in Python, GitHub PR #325.
    - eRPC: Python3.8 is not ready for usage of typing.Any type, GitHub PR #325.
    - eRPC: Improved codec function to use reference instead of address, GitHub PR #324.
    - eRPC: Fix NULL check for pending client creation, GitHub PR #341.
    - eRPC: Replace sprintf with snprintf, GitHub PR #343.

        - eRPC: Use MU_SendMsg blocking call in MU transport.
        - eRPC: New LPSPI and LPI2C transport layers.
        - eRPC: Freeing static objects, GitHub PR #353.
        - eRPC: Fixed casting in deinit functions, GitHub PR #354.
        - eRPC: Align LIBUSBSIO.GetNumPorts API use with libusbsio python module v. 2.1.11.
        - erpcgen: Renamed temp variable to more generic one, GitHub PR #321.
        - erpcgen: Add check that string read is not more than max length, GitHub PR #328.
        - erpcgen: Move to g++ in pytest, GitHub PR #335.
        - erpcgen: Use build=release for make, GitHub PR #334.
        - erpcgen: Removed boost dependency, GitHub PR #346.
        - erpcgen: Mingw support, GitHub PR #344.
        - erpcgen: VS build update, GitHub PR #347.
        - erpcgen: Modified name for common types macro scope, GitHub PR #337.
        - erpcgen: Fixed memcpy for template, GitHub PR #352.
        - eRPC,erpcgen: Change default build target to release + adding artefacts, GitHub PR #334.
        - eRPC,erpcgen: Remove redundant includes, GitHub PR #338.
        - eRPC,erpcgen: Many minor code improvements, GitHub PR #323.
        - MCMgr: Avoid calling tx isr callbacks when respective Messaging Unit Transmit Interrupt Enable flag is not set in the CR/TCR register.
        - MCMgr: Messaging Unit RX and status registers are cleared after the initialization.
        - RPMsg-Lite: Resolved issues in ThreadX env. layer implementation.
        - RPMsg-Lite: Added aarch64 support.
        - RPMsg-Lite: Increased the queue size to (2 * RL_BUFFER_COUNT) to cover zero copy cases.

- 2.13.0_imxrt1180a0
    - Multicore SDK component versions:
        - embedded Remote Procedure Call (eRPC) v1.10.0
        - eRPC generator (erpcgen) v.1.10.0
        - Multicore Manager (MCMgr) v4.1.3
        - RPMsg-Lite v5.0.0
    - New features:
        - MCMgr, RPMsg-Lite: Added porting layers for imxrt1180.
        - MCMgr: mu_isr() updated to avoid calling tx isr callbacks when respective Transmit Interrupt Enable flag is not set in the CR/TCR register.
        - RPMsg-Lite, eRPC: RPMsg_Lite queue size adjusted.
        - eRPC: MU transport layer switched to blocking MU_SendMsg() API use.
- 2.13.0
    - Multicore SDK component versions:
        - embedded Remote Procedure Call (eRPC) v1.10.0
        - eRPC generator (erpcgen) v.1.10.0
        - Multicore Manager (MCMgr) v4.1.3
        - RPMsg-Lite v5.0.0
    - New features:
        - eRPC: MUTransport adaptation to new supported SoCs.
        - eRPC: Simplifying CI with installing dependencies using shell script, GitHub PR #267.

    * eRPC: Using event for waiting for sock connection in TCP python server, formatting python code, C specific includes, GitHub PR #269.
    * eRPC: Endianness agnostic update, GitHub PR #276.
    * eRPC: Assertion added for functions which are returning status on freeing memory, GitHub PR #277.
    * eRPC: Fixed closing arbitrator server in unit tests, GitHub PR #293.
    * eRPC: Makefile updated to reflect the correct header names, GitHub PR #295.
    * eRPC: Compare value length to used length() in reading data from message buffer, GitHub PR #297.
    * eRPC: Add TCP_NODELAY option to python, GitHub PR #298.
    * eRPC: Replace EXPECT_TRUE with EXPECT_EQ in unit tests, GitHub PR #318.
    * eRPC: Adapt rpmsg_lite based transports to changed rpmsg_lite_wait_for_link_up() API parameters.
    * eRPC, erpcgen: Better distuingish which file can and cannot by linked by C linker, GitHub PR #266.
    * eRPC, erpcgen: Stop checking if pointer is NULL before sending it to the erpc_free function, GitHub PR #275.
    * eRPC, erpcgen: Changed api to count with more interfaces, GitHub PR #304.
    * erpcgen: Check before reading from heap the buffer boundaries, GitHub PR #287.
    * erpcgen: Several fixes for tests and CI, GitHub PR #289.
    * erpcgen: Refactoring erpcgen code, GitHub PR #302.
    * erpcgen: Fixed assigning const value to enum, GitHub PR #309.
    * erpcgen: Enable runTesttest_enumErrorCode_allDirection, serialize enums as int32 instead of uint32.
    * MCMgr: mcmgr_mu_internal.c code adaptation to new supported SoCs.
    * RPMsg-Lite: Improveed debug check buffers implementation - instead of checking the pointer fits into shared memory check the presence in the VirtIO ring descriptors list.
    * RPMsg-Lite: Timeout parameter added to rpmsg_lite_wait_for_link_up API function.
    * RPMsg-Lite: VRING_SIZE is set based on number of used buffers now (as calculated in vring_init) - updated for all platforms that are not communicating to Linux rpmsg counterpart.
    * RPMsg-Lite: Fixed wrong RL_VRING_OVERHEAD macro comment in platform.h files.
    * RPMsg-Lite: Misra corrections.

- 2.12.0_imx93
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.9.1
    * eRPC generator (erpcgen) v.1.9.1
    * Multicore Manager (MCMgr) v4.1.2
    * RPMsg-Lite v4.0.1
  - New features:
    * RPMsg-Lite: Added porting layers for i.mx93 device.
- 2.12.0
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.9.1
    * eRPC generator (erpcgen) v.1.9.1

        - Multicore Manager (MCMgr) v4.1.2
        - RPMsg-Lite v4.0.0
    - New features:
        - eRPC: Construct the USB CDC transport, rather than a client, GitHub PR #220.
        - eRPC: Fix premature import of package, causing failure when attempting installation of Python library in a clean environment, GitHub PR #38, #226.
        - eRPC: Improve python detection in make, GitHub PR #225.
        - eRPC: Fix several warnings with deprecated call in pytest, GitHub PR #227.
        - eRPC: Fix freeing union members when only default need be freed, GitHub PR #228.
        - eRPC: Fix making test under Linux, GitHub PR #229.
        - eRPC: Assert costumizing, GitHub PR #148.
        - eRPC: Fix corrupt clientList bug in TransportArbitrator, GitHub PR #199.
        - eRPC: Fix build issue when invoking g++ with -Wno-error=free-nonheap-object, GitHub PR #233.
        - eRPC: Fix inout cases, GitHub PR #237.
        - eRPC: Remove ERPC_PRE_POST_ACTION dependency on return type, GitHub PR #238.
        - eRPC: Adding NULL to ptr when codec function failed, fixing memcpy when fail is present during deserialization, GitHub PR #253.
        - eRPC: MessageBuffer usage improvement, GitHub PR #258.
        - eRPC: Get rid for serial and enum34 dependency (enum34 is in python3 since 3.4 (from 2014)), GitHub PR #247.
        - eRPC: Several MISRA violations addressed.
        - eRPC: Fix timeout for Freertos semaphore, GitHub PR #251.
        - eRPC: Use of rpmsg_lite_wait_for_link_up() in rpmsg_lite based transports, GitHub PR #223.
        - eRPC: Fix codec nullptr dereferencing, GitHub PR #264.
        - erpcgen: Fix two syntax errors in erpcgen Python output related to non-encapsulated unions, improved test for union, GitHub PR #206, #224.
        - erpcgen: Fix serialization of list/binary types, GitHub PR #240.
        - erpcgen: Fix empty list parsing, GitHub PR #72.
        - erpcgen: Fix templates for malloc errors, GitHub PR #110.
        - erpcgen: Get rid of encapsulated union declarations in global scale, improve enum usage in unions, GitHub PR #249, #250.
        - erpcgen: Fix compile error:UniqueIdChecker.cpp:156:104:'sort' was not declared, GitHub PR #265.
        - MCMgr: Update mcmgr_stop_core_internal() implementations to set core state to kMCMGR_ResetCoreState.
        - RPMsg-Lite: Introduce new rpmsg_lite_wait_for_link_up() API function - this allows to avoid using busy loops in rtos environments, GitHub PR #21.
        - RPMsg-Lite: Adjust rpmsg_lite_is_link_up() to return RL_TRUE/RL_FALSE.
- 2.11.1
    - Multicore SDK component versions:
        - embedded Remote Procedure Call (eRPC) v1.9.0
        - eRPC generator (erpcgen) v.1.9.0

    - Multicore Manager (MCMgr) v4.1.1
    - RPMsg-Lite v3.2.1
  - New features:
    - RPMsg-Lite: Add support for custom shared memory arangement per the RPMsg_Lite instance.
- 2.11.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.9.0
    - eRPC generator (erpcgen) v.1.9.0
    - Multicore Manager (MCMgr) v4.1.1
    - RPMsg-Lite v3.2.0
  - New features:
    - eRPC: Improving template usage, GitHub PR #153.
    - eRPC: run_clang_format.py cleanup, GitHub PR #177.
    - eRPC: Build TCP transport setup code into liberpc, GitHub PR #179.
    - eRPC: Fix multiple definitions of g_client error, GitHub PR #180.
    - eRPC: Fix memset past end of buffer in erpc_setup_mbf_static.cpp, GitHub PR #184.
    - eRPC: Fix deprecated error with newer pytest version, GitHub PR #203.
    - eRPC: Allow used LIBUSBSIO device index being specified from the Python command line argument.
    - eRPC, erpcgen: Static allocation support and usage of rpmsg static FreeRTOSs related APi, GitHub PR #168, #169.
    - erpcgen: Remove redundant module imports in erpcgen, GitHub PR #196.
    - RPMsg-Lite: Improve static allocations - allow OS-specific objects being allocated statically, GitHub PR #14.
    - RPMsg-Lite: Minor Misra and typo corrections, GitHub PR #19, #20.
- 2.10.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.8.1
    - eRPC generator (erpcgen) v.1.8.1
    - Multicore Manager (MCMgr) v4.1.1
    - RPMsg-Lite v3.1.2
  - New features:
    - eRPC: Fix misra erpc c, GitHub PR #158.
    - eRPC: Allow conditional compilation of message_loggers and pre_post_action.
    - eRPC: New i2c_slave_transport trasnport introduced.
    - eRPC: (D)SPI slave transports updated to avoid busy loops in rtos environments.
    - erpcgen: Re-implement EnumMember::hasValue(), GitHub PR #159.
    - erpcgen: Fixing several misra issues in shim code, erpcgen and unit tests updated, GitHub PR #156.
    - erpcgen: Fix bison file, GitHub PR #156.
    - RPMsg-Lite: Fixed incorrect description of the rpmsg_lite_get_endpoint_from_addr function.
    - RPMsg-Lite: Updated RL_BUFFER_COUNT documentation.
    - RPMsg-Lite: env_print macro adjusted to address MISRA 21.6 rule in MCUXpressoSDK

projects.

- 2.9.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.8.0
    - eRPC generator (erpcgen) v.1.8.0
    - Multicore Manager (MCMgr) v4.1.1
    - RPMsg-Lite v3.1.1
  - New features:
    - eRPC: Support win32 thread, GitHub PR #108.
    - eRPC: Add mbed support for malloc() and free(), GitHub PR #92.
    - eRPC: Update makefile.
    - eRPC: Fixed warnings and error with using MessageLoggers, GitHub PR #127.
    - eRPC: Extend error msg for python server service handle function, GitHub PR #132.
    - eRPC: Update CMSIS UART transport layer to avoid busy loops in rtos environments, introduce semaphores.
    - eRPC: Introduced pre and post callbacks for eRPC call, GitHub PR #131.
    - eRPC: Introduced new USB CDC transport.
    - eRPC: Introduced new Linux spidev-based transport.
    - eRPC: SPI transport update to allow usage without handshaking GPIO.
    - eRPC: Native *WIN32 erpc serial transport and threading.*
    - *eRPC: Arbitrator deadlock fix, TCP transport updated, TCP setup functions introduced, GitHub PR #121.*
    - *eRPC: Update of matrix_multiply.py example: Add –serial and –baud argument, GitHub PR #137.*
    - *eRPC: Added formatting extension for VSC, GitHub PR #134.*
    - *eRPC: Update of .clang-format, GitHub PR #140.*
    - *eRPC: Update of erpc_framed_transport.cpp: return error if received message has zero length, GitHub PR #141.*
    - *eRPC, erpcgen: Fixed error messages produced by -Wall -Wextra -Wshadow -pedantic-errors compiler flags, GitHub PR #136, #139.*
    - *eRPC, erpcgen: Core re-formatted using Clang version 10.*
    - *erpcgen: Enable deallocation in server shim code when callback/function pointer used as out parameter in IDL.*
    - *erpcgen: Removed '$' character from generated symbol name in* '$union' suffix, GitHub PR #103.
    - erpcgen: Resolved mismatch between C++ and Python for callback index type, GitHub PR #111.
    - erpcgen: Python generator improvements, GitHub PR #100, #118.
    - erpcgen: Fixed error messages produced by -Wall -Wextra -Wshadow -pedantic-errors compiler flags, GitHub PR #136.
    - erpcgen: Introduce ustring type for unsigned char and force cast to char∗, GitHub PR #125.
    - RPMsg-Lite: Introduced RL_ALLOW_CONSUMED_BUFFERS_NOTIFICATION config option to allow opposite side notification sending each time received buffers are consumed and put into the queue of available buffers.

    - RPMsg-Lite: Added environment layers for Threadx.
- 2.8.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.7.4
    - eRPC generator (erpcgen) v.1.7.4
    - Multicore Manager (MCMgr) v4.1.0
    - RPMsg-Lite v3.1.0
  - New features:
    - eRPC: Unit test code updated to handle service add and remove operations.
    - eRPC: Several MISRA issues in rpmsg-based transports addressed.
    - eRPC: Support MU transport unit testing.
    - eRPC: Adding mbed os support.
    - eRPC: Fixed Linux/TCP acceptance tests in release target.
    - eRPC: Minor documentation updates, code formatting.
    - erpcgen: Whitespace removed from C common header template.
    - RPMsg-Lite: MISRA C-2012 violations fixed (7.4).
    - RPMsg-Lite: Fix missing lock in rpmsg_lite_rx_callback() for QNX env.
    - RPMsg-Lite: Correction of rpmsg_lite_instance structure members description.
    - RPMsg-Lite: Address -Waddress-of-packed-member warnings in GCC9.
    - RPMsg-Lite: Clang update to v10.0.0, code re-formatted.
- 2.7.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.7.3
    - eRPC generator (erpcgen) v.1.7.3
    - Multicore Manager (MCMgr) v4.1.0
    - RPMsg-Lite v3.0.0
  - New features:
    - eRPC: Improved the test_callbacks logic to be more understandable and to allow requested callback execution on the server side.
    - eRPC: TransportArbitrator::prepareClientReceive modified to avoid incorrect return value type.
    - eRPC: The ClientManager and the ArbitratedClientManager updated to avoid performing client requests when the previous serialization phase fails.
    - erpcgen: Generate the shim code for destroy of statically allocated services.
    - MCMgr: Code adjustments to address MISRA C-2012 Rules
    - RPMsg-Lite: MISRA C-2012 violations fixed, incl. data types consolidation.
    - RPMsg-Lite: Code formatted
- 2.6.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.7.2
    - eRPC generator (erpcgen) v.1.7.2
    - Multicore Manager (MCMgr) v4.0.3
    - RPMsg-Lite v2.2.0
  - New features:
    - eRPC: Improved support of const types.

    - eRPC: Fixed Mac build.
    - eRPC: Fixed serializing python list.
    - eRPC: Documentation update.
    - eRPC: Add missing doxygen comments for transports.
    - RPMsg-Lite: Added configuration macro RL_DEBUG_CHECK_BUFFERS.
    - RPMsg-Lite: Several MISRA violations fixed.
    - RPMsg-Lite: Added environment layers for QNX and Zephyr.
    - RPMsg-Lite: Allow environment context required for some environments (controlled by the RL_USE_ENVIRONMENT_CONTEXT configuration macro).
    - RPMsg-Lite: Data types consolidation.
    - MCMgr: Documentation updated to describe handshaking in a graphic form.
    - MCMgr: Minor code adjustments based on static analysis tool findings
- 2.5.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.7.1
    - eRPC generator (erpcgen) v.1.7.1
    - Multicore Manager (MCMgr) v4.0.2
    - RPMsg-Lite v2.0.2
  - New features:
    - RPMsg-Lite, MCMgr: Align porting layers to the updated MCUXpressoSDK feature files.
    - eRPC: Fixed semaphore in static message buffer factory.
    - erpcgen: Fixed MU received error flag.
    - erpcgen: Fixed tcp transport.
- 2.4.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.7.0
    - eRPC generator (erpcgen) v.1.7.0
    - Multicore Manager (MCMgr) v4.0.1
    - RPMsg-Lite v2.0.1
  - New features:
    - eRPC: Improved code size of generated code.
    - eRPC: Generating crc value is optional.
    - eRPC: Fixed CMSIS Uart driver. Removed dependency on KSDK.
    - eRPC: List names are based on their types. Names are more deterministic.
    - eRPC: Service objects are as a default created as global static objects.
    - eRPC: Added missing doxygen comments.
    - eRPC: Forbid users use reserved words.
    - eRPC: Removed outByref for function parameters.
    - eRPC: Added support for 64bit numbers.
    - eRPC: Added support of program language specific annotations.
    - eRPC: Optimized code style of callback functions.
    - RPMsg-Lite: New API rpmsg_queue_get_current_size()
    - RPMsg-Lite: Fixed bug in interrupt handling for lpc5411x, lpc5410x
    - RPMsg-Lite: Code adjustments based on static analysis tool findings
- 2.3.1

  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.6.0
    - eRPC generator (erpcgen) v.1.6.0
    - Multicore Manager (MCMgr) v4.0.0
    - RPMsg-Lite v1.2.0
  - New features:
    - eRPC: Improved code size of generated code.
    - eRPC: Improved eRPC nested calls.
    - eRPC: Improved eRPC list length variable serialization.
    - eRPC: Added @nullable support for scalar types.
    - MCMgr: Added new MCMGR_TriggerEventForce() API.
- 2.3.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.5.0
    - eRPC generator (erpcgen) v.1.5.0
    - Multicore Manager (MCMgr) v3.0.0
    - RPMsg-Lite v1.2.0
  - New features:
    - eRPC: Added support for unions type non-wrapped by structure.
    - eRPC: Added callbacks support.
    - eRPC: Added support @external annotation for functions.
    - eRPC: Added support @name annotation.
    - eRPC: Added Messaging Unit transport layer.
    - eRPC: Added RPMSG Lite RTOS TTY transport layer.
    - eRPC: Added version verification and IDL version verification between eRPC code and eRPC generated shim code.
    - eRPC: Added support of shared memory pointer.
    - eRPC: Added annotation to forbid generating const keyword for function parameters.
    - eRPC: Added python matrix multiply example.
    - eRPC: Added nested call support.
    - eRPC: Added struct member "byref" option support.
    - eRPC: Added support of forward declarations of structures
    - eRPC: Added Python RPMsg Multiendpoint kernel module support
    - eRPC: Added eRPC sniffer tool
    - MCMgr: Unused API removed
    - MCMgr: Added the ability for remote core monitoring and event handling
    - RPMsg-Lite: Several source files renamed to avoid conflicts with other middleware sw components
    - RPMsg-Lite: Added the ability to use Multicore Manager (MCMGR) as the IPC interrupts router
- 2.2.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.4.0
    - eRPC generator (erpcgen) v.1.4.0
    - Multicore Manager (MCMgr) v2.0.1

        - RPMsg-Lite v1.1.0
    - New features:
        - eRPC: win_flex_bison.zip for windows updated.
        - eRPC: Use one codec (instead of inCodec outCodec).
        - eRPC: New RPMsg-Lite Zero Copy (RPMsgZC) transport layer.
        - MCMgr: code updated to be Misra compliant.
        - RPMsg-Lite: Added macros for packed structures (compiler.h).
        - RPMsg-Lite: Improved interrupt handling in platform layer.
        - RPMsg-Lite: Changed RL_BUFFER_SIZE definition.
        - RPMsg-Lite: Fix of double initialization of vring shared data structure.
        - RPMsg-Lite: Support for the multi-instance.
- 2.1.0
    - Multicore SDK component versions:
        - embedded Remote Procedure Call (eRPC) v1.3.0
        - eRPC generator (erpcgen) v.1.3.0
    - New features:
        - eRPC: New annotation types introduced (@length, @max_length, ...).
        - eRPC: Support for running both erpc client and erpc server on one side.
        - eRPC: New transport layers for (LP)UART, (D)SPI.
        - eRPC: Error handling support.
- 2.0.0
    - Multicore SDK component versions:
        - embedded Remote Procedure Call (eRPC) v1.2.0
        - eRPC generator (erpcgen) v.1.2.0
        - Multicore Manager (MCMgr) v2.0.0
        - RPMsg-Lite v1.0.0
    - New features:
        - Multicore SDK support for lpcxpresso54114 board added.
        - RPMsg component of the Open-AMP framework re-implemented and the RPMsg-Lite version introduced.
        - eRPC source directory organization changed.
        - Many eRPC improvements.
- 1.1.0
    - Multicore SDK component versions:
        - embedded Remote Procedure Call (eRPC) v1.1.0
        - Multicore Manager (MCMgr) v1.1.0
        - Open-AMP / RPMsg based on SHA1 ID 44b5f3c0a6458f3cf80 rev01
    - New features:
        - Multicore SDK 1.1.0 ported to KSDK 2.0.0.
        - Python support added into eRPC.
- 1.0.0
    - Multicore SDK component versions:
        - embedded Remote Procedure Call (eRPC) v1.0.0
        - Multicore Manager (MCMgr) v1.0.0
        - Open-AMP / RPMsg based on SHA1 ID 44b5f3c0a6458f3cf80 rev00

NTAG I2C plus library

The current version is 1.0.0.

- 1.0.0
  - initial release.

RTCESL for KSDK

Current version is 4.3

- 4.3
  - Initial version.

SAFETY_IEC60730B for KSDK

Current version is 1.1.0

- 1.1.0
  - Initial version.

Host SDIF driver for MCUXpresso SDK

The current driver version is 2.4.1.

- 2.4.1
  - Improvements
    - Added macro SDMMCHOST_SUPPORT_VOLTAGE_CONTROL.
- 2.4.0
  - Improvements
    - Removed deprecated api in SDIF host driver.
    - Added SDMMCHOST_ConvertDataToLittleEndian api.
    - Added capability/maxBlockCount/maxBlockSize in host decriptior.
    - Added mutual exclusive access for function init/deinit/reset/transfer function.
    - Fixed violations of MISRA C-2012 rule 10.1.
- 2.3.1
  - Improvements
    - Added host instance capability macro.
    - Added clear card inserted/removed event when card removed/inserted interrupt generated.
    - Increased the reset timeout value to fix the data machine still busy after sdif reset issue.
    - Enabled the error recovery function by adding host reset operations.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9, 15.7, 4.7, 16.4, 10.1, 10.3, 10.4, 11.3, 14.4, 10.6, 17.7, 16.1, 16.3.
- 2.3.0

- Improvements
  - Merged the host controller driver from polling/freertos/interrupt to non_blocking/blocking.
  - Added SDMMC OSA layer to support muxtex access/event/delay.
- 2.2.14
  - Bug Fixes
    - Fixed uninitialized value Coverity issue.
- 2.2.13
  - Improvements:
    - Added host reset after the card being powered on for host controller SDIF to fix the DATA_BUSY issue.
    - Removed the SDIF_Reset from SDMMCHOST_Reset.
- 2.0.0
  - Initial version

MMC Card driver for MCUXpresso SDK

The current driver version is 2.5.0.

- 2.5.0
  - Improvements
    - Added api MMC_SetSleepAwake to support enter/exit sleep state.
    - Added new api MMC_PollingCardStatusBusy for application polling card status.
    - Removed deprecated api in mmc driver and mark MMC_HostReset as deprecated.
    - Improved the read/write/erase function flow.
    - Added mutual exclusive access for init/deinit/read/write/erase function.
    - Fixed violations of MISRA C-2012 rule 4.7, 17.7, 10.7, 10.4, 13.5, 14.4, 10.6.
- 2.4.1
  - Improvements
    - Improved the voltage window argument of CMD1 according to host capabilty instead of use card ocr directly.
    - Added host HS200/HS400/8bit bus width capability validation during card initialization.
    - Used cache line size align buffer for MMC relate api.
    - Increased the CMD13 timeout count to avoid polling CMD13 time out issue.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9, 15.7, 4.7, 16.4, 10.1, 10.3, 10.4, 11.3, 14.4, 10.6, 17.7, 16.1, 16.3.
- 2.4.0
  - Improvements
    - Added new apis MMC_EnableCacheControl/MMC_FlushCache to support cache feature.
- 2.3.1
  - Improvements
    - Removed the dead loop while polling DAT0 and CMD13 instead of using timeout mechanism.
    - Added card state check before switching to HS400 to improve the emmc initialization

stability.
    - Removed the redundant operation of memset internal buffer in MMC_WrtiteBlocks function.
  - Bug Fixes
    - Fixed the sandisk emmc always busy while sending CMD1 without supported voltage provide in argument.
- 2.3.0
  - Improvements
    - Deprecated api MMC_PowerOnCard/MMC_PowerOffCard by api MMC_SetCardPower.
    - Added internalBuffer in mmc_card_t and removed rawCid/rawCsd/rawExtendedCsd.
    - Added retuning support during data transfer under HS200 mode.
    - Increased the read/write blocks failed retry times for stability.
    - Added delay while retry the CMD1 for stability.
    - Added legacy card support, the card not support CMD6, CMD8.
- 2.2.13
  - Improvements
    - Used the boot mode value instead of boot mode mask value as the parameter of MMC_-SetBootConfig to improve user experience.
    - Removed dynamic voltage switch feature for mmc, according to JEDEC standard, the voltage should be fixed after power up.
- 2.2.12
  - Improvement
    - Increased the CMD1 retry times in the MMC card driver to improve driver compatibility.
  - Bug Fixes
    - Fixed 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).
    - Fixed the fall through build warning by adding SUPPRESS_FALL_THROUGH_WARNING() in mmc driver.
- 2.2.7
  - Bug Fixes
    - Fixed MDK 66-D warning.
- 2.2.6
  - Improvements
    - Saved MMC OCR registers while sending CMD1 with argument 0.
  - Bug Fixes
    - Added MMC_PowerOn function in which there is delay function after powerup sdcard. Otherwise, the card initialization by fail.
- 2.2.5
  - Improvements
    - Added SDMMC_ENABLE_SOFTWARE_TUNING to enable/disable software tuning and it is disabled by default.
- 2.2.4
  - Bug Fixes
    - Fixed DDR mode data sequence miss issue, which is caused by NIBBLE_POS.
  - Improvements

    - 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.
- 2.2.3
  - Bug Fixes
    - Added response check for send operation condition command. If not checked, the card may occasionally init fail.
- 2.2.1
  - Improvements
    - Improved MMC Boot feature.
- 2.2.0
  - Improvements
    - Optimized tuning/mmc switch voltage/mmc select power class/mmc select timing function.
    - Added strobe dll for mmc HS400 mode.
    - Added write complete wait operation for MMC_Write to fix command timeout issue.
- 2.1.2
  - Improvements
    - Improved SDMMC to support eMMC v5.0.
  - Bug Fixes
    - Fixed incorrect 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.
- 2.1.0
  - Improvements
    - Optimized the function of setting maximum data bus width for MMC card.
- 2.0.0
  - Initial version

SD Card driver for MCUXpresso SDK

The current driver version is 2.4.2.

- 2.4.2
  - Improvements
    - Improved the erase timeout calculation logical in function SD_EraseBlocks according to SD specifications.
    - Added polling erase done status after each erase operations.
- 2.4.1
  - Improvements
    - Added macro SDMMCHOST_SUPPORT_VOLTAGE_CONTROL for the host which not support voltage control.

- 2.4.0
  - Improvements
    - Removed deprecated api in sd driver.
    - Added new api SD_PollingCardStatusBusy for application polling card status.
    - Improved the read/write/erase function flow.
    - Improved the signal line voltage switch flow.
    - Added powerOnDelayMS/powerOffDelayMS in sd_usr_param_t to allow redefine the default power on/off delay.
    - Added mutual exclusive access for init/deinit/read/write/erase function.
    - Fixed the driver strength configurations missed when timing mode switch to non SDR50/-SDR104 mode.
    - Fixed violations of MISRA C-2012 rule 4.7, 17.7, 10.7, 10.4, 13.5, 14.4.
- 2.3.3
  - Improvements
    - Added host SDR timing mode capability validation during card initialization.
    - Added plling card ready for data status when transfer data failed.
    - Used cache line size align buffer for SD initialization api.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9, 15.7, 4.7, 16.4, 10.1, 10.3, 10.4, 11.3, 14.4, 10.6, 17.7, 16.1, 16.3.
- 2.3.2
  - Improvements
    - Moved power off function after card detect in SD_Init for DAT3 detect card feature.
- 2.3.1
  - Improvements
    - Removed the dead loop while polling DAT0 and CMD13 instead of using timeout mechanism.
- 2.3.0
  - Improvements
    - Marked api SD_HostReset/SD_PowerOnCard/SD_PowerOffCard/SD_WaitCardDetect-Status as deprecated.
    - Added new api SD_SetCardPower/SD_PollingCardDetectStatus/SD_HostDoReset.
    - Added internalBuffer in sd_card_t and removed rawCid/rawCsd/rawScr.
    - Added retuning support during data transfer under SDR50/SDR104 mode.
    - Increased the read/write blocks failed retry times for stability.
    - Added delay while retry the ACMD41 for stability.
- 2.2.12
  - Improvements
    - Increased the sd io driver strength for SD2.0 card.
  - Bug Fixes
    - Fixed 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).
- 2.2.10
  - Bug Fixes
    - Added event value check for all the FreeRTOS events to fix program hangs when a card

event occurs before create.

- 2.2.7
  - Bug Fixes
    - Fixed MDK 66-D warning.
- 2.2.5
  - Improvements
    - Added SD_ReadStatus api to get 512bit SD status.
    - Added error log support in sdcard functions.
    - Added SDMMC_ENABLE_SOFTWARE_TUNING to enable/disable software tuning and it is disabled by default.
- 2.2.4
  - Bug Fixes
    - Fixed DDR mode data sequence miss issue, which is caused by NIBBLE_POS.
  - Improvements
    - Increased g_sdmmc 512byte to improve the performance when application use a non-word align data buffer address.
    - Enabled auto cmd12 for SD read/write.
- 2.2.3
  - Bug Fixes
    - Added response check for send operation condition command. If not checked, the card may occasionally init fail.
- 2.2.1
  - Improvements
    - Kept SD_Init function for forward compatibility.
- 2.2.0
  - Improvements
    - Separated the SD/MMC/SDIO init API to xxx_CardInit/xxx_HostInit.
    - SD_Init/SDIO_Init will be deprecated in the next version.
- 2.1.6
  - Improvements
    - Enhanced SD IO default driver strength.
- 2.1.5
  - Bug Fixes
    - Fixed Coverity issue.
    - Fixed SD v1.x card write fail issue. It was caused by the block length set error.
    - Fixed card cannot detect dynamically.
- 2.1.3
  - Bug Fixes
    - Fixed Non high-speed sdcard init fail at switch to high speed.
  - Improvements
    - Added Delay for SDCard power up.
- 2.1.2
  - Improvements
    - Improved SDMMC to support SD v3.0.
- 2.1.1

- Bug Fixes
  - Fixed the bit mask error in the SD card switch to high speed function.
- Improvements
  - 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.
  - Improvements
    - Changed the name of some error codes returned by internal function.
    - Merged all host related attributes to one structure.
- 2.0.0
  - Initial version.

SDIO Card driver for MCUXpresso SDK

The current driver version is 2.4.1.

- 2.4.1
  - Improvements
    - Added macro SDMMCHOST_SUPPORT_VOLTAGE_CONTROL for the host which not support voltage control.
- 2.4.0
  - Improvements
    - Removed deprecated api in sdio driver.
    - Improved the signal line voltage switch flow.
    - Added powerOnDelayMS/powerOffDelayMS in sdio_usr_param_t to allow redefine the default power on/off delay.
    - Added mutual exclusive access for init/deinit/direct/extend function.
    - Fixed violations of MISRA C-2012 rule 4.7, 17.7, 10.1, 12.2.
- 2.3.3
  - Bug Fixes
    - Fixed logical dead code coverity issue.
  - Improvements
    - Removed deprecated api in sdio driver.
- 2.3.2
  - Improvements
    - Added host SDR timing mode capability validation during card initialization.
    - Used cache line size align buffer for SDIO initialization api.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9, 15.7, 4.7, 16.4, 10.1, 10.3, 10.4, 11.3, 14.4, 10.6, 17.7, 16.1, 16.3.
- 2.3.1
  - Improvements

    - Moved power off function after card detect in SD_Init for DAT3 detect card feature.
- 2.3.0
  - Improvements
    - Marked api SDIO_HostReset/SDIO_PowerOnCard/SDIO_PowerOffCard/SDIO_WaitCardDetectStatus as deprecated.
    - Added new api SDIO_SetCardPower/SDIO_PollingCardDetectStatus/SDIO_HostDoReset.
    - Added internalBuffer in sdio_card_t for card register content extract and improve the data access efficiency.
    - Added retry function after switch to target timing failed in SDIO_SelectBusTiming.
    - Changed defalut bus clock from 400KHZ to 25MHZ.
- 2.2.13
  - Improvements
    - Removed the sdio card interrupt from sdio host initialization, since the card interrupt enablement should be determined by application.
  - Bug Fixes
    - Fixed Out-of-bounds write Coverity issue.
- 2.2.12
  - Improvements
    - Added manual tuning function for looking for the tuning window automatically.
    - Fixed 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).
    - Fixed the fall through build warning by adding SUPPRESS_FALL_THROUGH_WARNING() in sdio driver.
- 2.2.11
  - Bug Fixes
    - Added check card async interrupt capability in function SDIO_GetCardCapability.
    - Fixed OUT OF BOUNDS access in function SDIO_IO_Transfer.
- 2.2.10
  - Bug Fixes
    - Fixed SDIO card driver get an incorrect io number when the card io number is bigger than 2.
  - Improvements
    - Added SDIO 3.0 support.
    - Added API SDIO_IO_RW_Direct for direct read/write card register access.
- 2.2.9
  - Improvements
    - Added API SDIO_SetIOIRQHandler/SDIO_HandlePendingIOInterrupt to handle multi io pending IRQ.
- 2.2.8
  - Improvements
    - Updated sdmmc to support SDIO interrupt.
    - Added API SDIO_GetPendingInterrupt to get the pending io interrupt.
- 2.2.7
  - Bug Fixes

    - Fixed MDK 66-D warning.
- 2.2.6
  - Improvements
    - Added an unify transfer interface for SDIO.
  - Bug Fixes
    - Fixed Wrong pointer address used by SDMMCHOST_Init.
- 2.1.5
  - Improvements
    - Improved SDIO card init sequence and add retry option for SDIO_SwitchToHighSpeed function.
- 2.1.4
  - Improvements
    - Added Go_Idle function for SDIO card.
- 2.0.0
  - Initial version.

Secure Element hostlib for KSDK

- For Detailed change log please download the Plug & Trust MW package from `https://www.nxp.com/products/:SE050?tab=Design_Tools_Tab` and refer to the Changes Present in the User guide.
- 3.01.00
  - Cleanup for heap management macros. Added support to redirect macros to FreeRTOS APIs.
- 3.00.03
  - smCom_Init: return type is now *U16* instead of *void*. Return value indicates success/failure to create mutex/semophore.
- 3.00.02
  - Fixed: potential null pointer dereference
- 2.16.00
  - MIMXRT1050 support replaced with MIMXRT1060
  - Examples renamed
- 2.14.00
  - Extensively revamped `fsl_sss_ftr.h` file for finer control of build configuration selection.
  - smCom Layer is refactored so that Application send down the connection handles/parameters to lower layer.
  - Define `T1oI2C_UM1225_SE050` is no longer applicable, use `T1oI2C_UM11225_SE05X` instead.
- 2.11.03
  - hostlib file/folder moved
- 2.11.0
  - Added Support for SE050 and SSS API's
- 1.6.0
  - Host Library version goes from 01.40 to 01.41 (A71CH Host API has not been extended)
  - Conditional translation of i2c_Failed into i2c_NoAddrAck removed (this translation is no

longer required as a NACK is no longer lumped into i2c_Failed by the I2C driver wrapper)
  - Additional update of SCI2C implementation: Ensure multiple consecutive NACK's on address will trigger a return from function 'sci2c_SlaveToMasterDataTx'.
  - sci2c_Init update: initial loop fetching sci2c status modified so a SoftReset is issued for all exception types.
  - Solved potential buffer overflow in the implementation of smApduAppendCmdData
  - Added additional explicit checks that pointer arguments passed at the AX_API/A71CH_API level are not NULL pointers. In case this check was done through an 'assert' statement, the 'assert' statement has been removed.
- 1.5.0
  - Added i.MX RT1050 EVKB and LPC54018 IoT module projects
  - Added Watson IoT demo
  - GP Storage Lookup table is no longer restricted to 5 objects, up to 254 objects can be stored (use HLSE_MAX_OBJECTS_IN_TABLE (>=8) to limit support and save memory)
  - GP Storage Update Counter can be disabled by defining HLSE_DISABLE_UPDATE_COUNTER (not defined by default).
  - It's possible to fetch the absolute storage offset of an object
  - Specific error code when attempting to update a locked Lookup table
  - Extended doxygen documentation
- 1.4.2
  - Upgraded rtos\amazon-freertos\lib\pkcs11\portable\nxp\se_hostlib\pkcs11_se.c
  - Added GCP (Google Cloud Platform Demo)
  - Readme updates for AWS JITR Demo
- 1.4.0.1
  - Examples: Fix time stamp printing on embedded platform. (Time stams are supported only for Linux/Windows)
- 1.0.0
  - New integration

Trusted Firmware M (TF-M) for MCUXpresso SDK

The current version is based on TF-M v1.8.0, released 2023-07-31 (42f052298bd0babd3484fad3befd2e77f395654d) on https://git.trustedfirmware.org/TF-M/trusted-firmware-m.git

- 1.8.0 Rev1
  - Ported TF-M v1.8.0 to MCUXpresso SDK. Based on the 2023-07-31 snapshot(42f052298bd0babd3484fad
- 1.8.0
  - Ported TF-M v1.8.0 to MCUXpresso SDK. Based on the 2023-04-27 snapshot(35ac80c8581d483da7d18cb
- 1.7.0
  - Ported TF-M v1.7.0 to MCUXpresso SDK. Based on the 2023-04-04 snapshot(60c4936a7bb3f56bdb50b0
- 1.6.0
  - Ported TF-M v1.6.0 to MCUXpresso SDK. Based on the 2022-08-29 snapshot(a5048fb687e4fbf28f9aac69
  - Added lpcxpresso55s36 platform.
- 1.5.0

  - Ported TF-M v1.5.0 to MCUXpresso SDK. Based on the 2022-03-11 snapshot(47c26ecd0dec177fe2ddf82
- 1.4.0
  - Ported TF-M v1.4.0 to MCUXpresso SDK. Based on the 2021-08-30 snapshot(00c1106624d733aade4486
- 1.3.0
  - Ported TF-M v1.3.0 to MCUXpresso SDK. Based on the 2021-04-13 snapshot(cad01ab98c34bb3a13ab49
  - Added evkmimxrt685 platform.
  - Added evkmimxrt595 platform.
- 1.1
  - Ported TF-M v1.1 to MCUXpresso SDK. Based on the 2020-08-19 snapshot(54507b1645087e92b5a1159
- 1.0 Rev3
  - Ported TF-M v1.0 to MCUXpresso SDK. Based on the 2020-04-20 snapshot(1e089705899ff68c617cc0c01
  - Added HUK derivation function.
- 1.0 Rev2
  - Ported TF-M v1.0 to MCUXpresso SDK. Based on the 2020-04-06 snapshot(d4ac5d14e74d1aa2a78526aa
  - Added the LPC55S Flash module support for ITS and PS.
  - Added lpcxpresso55s16 platform.
- 1.0 Rev1
  - Ported TF-M v1.0-RC1 to MCUXpresso SDK. Based on the 2019-09-23 snapshot(011c0ad0e76d62bd6f2
  - Added MCUx and GCC demo applications.
  - Added PSA test suite application.
  - Used mbedCrypto instead of mbedTLS
- 1.0
  - Ported TF-M v1.0-beta to MCUXpresso SDK. Based on the 2019-03-13 snapshot(a5a2a5bc32bc50f4def8
  - Added lpcxpresso55s69 platform.
  - Added MDK demo applications.
  - Bug Fixes:
    * Fixed compilation warnings and errors.
    * Changes in the TF-M original source code are marked by the "NXP" comment.

Trusted Firmware-M(TF-M) Tests for MCUXpresso SDK

The current version is based on TF-M Tests v1.8.0, released 2023-07-24 (4625c232de8c1e9a28033856c5e1981413b82
on https://git.trustedfirmware.org/TF-M/tf-m-tests.git

- 1.8.0 Rev1
  - Ported TF-M Tests v1.8.0 to MCUXpresso SDK. Based on the 2023-07-24 snapshot(4625c232de8c1e9a28
- 1.8.0
  - Ported TF-M Tests v1.8.0 to MCUXpresso SDK. Based on the 2023-04-19 snapshot(4f181fdce62cc894633
- 1.7.0
  - Ported TF-M Tests v1.7.0 to MCUXpresso SDK. Based on the 2023-03-28 snapshot(7cf20ef136068907953
- 1.6.0
  - Ported TF-M Tests v1.6.0 to MCUXpresso SDK. Based on the 2022-08-29 snapshot(f0634da7100eeb4921
- 1.5.0
  - Ported TF-M Tests v1.5.0 to MCUXpresso SDK. Based on the 2022-03-01 snapshot(9349aff01fa0e4f689c

- 1.4.0
  - Ported TF-M Tests v1.4.0 to MCUXpresso SDK. Based on the 2021-08-21 snapshot(d4b2ba3683356b6f16
- 1.3.0
  - Ported TF-M Tests v1.3.0 to MCUXpresso SDK. Based on the 2021-03-19 snapshot(e7430d13b388a106c9

Arm Platform Security Architecture Test Suite for MCUXpresso SDK

The current version is based on psa-arch-tests v1.4, released 2023-07-07 (16ec358fc12b0ae87e9d0133c9c53abee03e4
on https://github.com/ARM-software/psa-arch-tests

- 1.4 Rev2
  - Ported psa-arch-tests v1.4 to MCUXpresso SDK. Based on the 2023-07-07 snapshot(16ec358fc12b0ae87e
- 1.4 Rev1
  - Ported psa-arch-tests v1.4 to MCUXpresso SDK. Based on the 2022-10-28 snapshot(cf8bd7191219df9bde
- 1.4
  - Ported psa-arch-tests v1.4 to MCUXpresso SDK. Based on the 2022-04-12 snapshot(ed006125c73ed3008
- 1.3
  - Ported psa-arch-tests v1.3 to MCUXpresso SDK. Based on the 2021-10-18 snapshot(01ee3ffef2a0a7d72ba
- 1.2
  - Ported psa-arch-tests v1.2 to MCUXpresso SDK. Based on the 2021-04-08 snapshot(e005834ff0d55233f4
- 1.1
  - Ported psa-arch-tests v1.1 to MCUXpresso SDK. Based on the 2021-24-03 snapshot(b0635d9af8ba031e5d
- 1.0 Rev2
  - Ported psa-arch-tests v1.0 to MCUXpresso SDK. Based on the 2020-10-06 snapshot(a6b3351d36875a75d
- 1.0 Rev1
  - Ported psa-arch-tests v1.0 to MCUXpresso SDK. Based on the 2020-08-10 snapshot(656685b76e33f16ae2
- 1.0
  - Ported psa-arch-tests v1.0 to MCUXpresso SDK. Based on the 2020-04-03 snapshot(3e2358bf417f771682
  - Added lpcxpresso55s16 platform.
- 0.9
  - New features:
    - Ported psa-arch-tests v0.9 to MCUXpresso SDK. Based on the 2019-07-25 snapshot (c80681ed7c7f3e2cbf02ded1ef2464ba2ca7ccd5).
    - Added lpcxpresso55s69 platform.
  - Bug Fixes:
    - Fixed compilation warnings and errors. Changes in the psa-arch-tests original source code are marked by the "NXP" comment.
    - Renamed test_entry.c files to test_rntry_<test_id>.c to avoid the Keil compilation issue when a lot files have the same name.

Mbed Crypto library for MCUXpresso SDK

The current version of the Mbed-Crypto library is based on mbedTLS v.3.4.0, released 2023-28-03, snapshot(1873d3bfc2da771672bd8e7e8f41f57e0af77f33) on https://github.com/Mbed-TLS/mbedtls

- 3.4.0
  - Ported Mbed-Crypto library, based on mbedTLS v.3.4.0, to MCUXpresso SDK. Released 2023-28-03, snapshot(1873d3bfc2da771672bd8e7e8f41f57e0af77f33).
- 3.3.0
  - Ported Mbed-Crypto library, based on mbedTLS v.3.3.0, to MCUXpresso SDK. Released 2022-12-14, snapshot(8c89224991adff88d53cd380f42a2baa36f91454).
- 3.2.1
  - Ported Mbed-Crypto library, based on mbedTLS v.3.2.1, to MCUXpresso SDK. Released 2022-07-12, snapshot (869298bffeea13b205343361b7a7daf2b210e33d).
- 3.1.0
  - Ported Mbed-Crypto library, based on mbedTLS v.3.1.0, to MCUXpresso SDK. Released 2021-12-15, snapshot (d65aeb37349ad1a50e0f6c9b694d4b5290d60e49).
- 3.0.0
  - Ported Mbed-Crypto library, based on mbedTLS v.3.0.0, to MCUXpresso SDK. Released 2021-07-06, snapshot(8df2f8e7b9c7bb9390ac74bb7bace27edca81a2b).
- 2.26.0
  - Ported Mbed-Crypto library, based on mbedTLS v.2.26.0, to MCUXpresso SDK. Released 2021-03-12, snapshot(e483a77c85e1f9c1dd2eb1c5a8f552d2617fe400).
- 2.25.0
  - Ported Mbed-Crypto library, based on mbedTLS v.2.25.0, to MCUXpresso SDK. Released 2020-12-10, snapshot(1c54b5410fd48d6bcada97e30cac417c5c7eea67).
- 2.23.0
  - Ported Mbed-Crypto library, based on mbedTLS v.2.23.0, to MCUXpresso SDK. Released 2020-08-27, snapshot(848a4e06b375e067552f1a21d4bc69322c673217).
- 2.20.0
  - Ported Mbed-Crypto v3.0.1 to MCUXpresso SDK. Based on the 2020-01-27 snapshot(1146b4e06011b69a
- 1.1.0_rev2
  - Added the Casper module acceleration.
- 1.1.0_rev1
  - Added the HashCrypt module acceleration.
- 1.1.0
  - Ported Mbed-Crypto v1.1.0 to MCUXpresso SDK. Based on the 2019-06-07 snapshot(47f2de132936905d
  - Added lpcxpresso55s69 platform.

USB stack for MCUXpresso SDK

The current version of USB stack is 2.9.1.

- 2.9.1

  - Improvement:
    - Update EHCI controller driver for basic support of eUSB.
    - Replace the hard code in audio cases with macro.
    - Uniform the Chapter9 for device lite cases.
- 2.9.0
  - Improvement:
    - Change ROOT2 as enabled by default in device stack.
    - Implement independent frequency adjustment for speaker and recorder of composite audio unified demos.
    - Fix vulnerability for host stack. CVE number: CVE-2023-38749
    - Delete deprecated enet driver function for enet adapter.
- 2.8.4
  - Improvement:
    - Add the new netc adatper for the new netc driver.
    - Fix issues for USB device dfu and usb device msc when enable the macro USB_DEVICE_CONFIG_RETURN_VALUE_CHECK.
    - Change the header file including order for usb.h header.
    - Update the USB host audio class driver to fix the wrong output log.
    - Add the workaround on dev_hid_mouse_bm case for the errata TN00071.
    - Enable ROOT2 macro in USB device stack.
    - Use an unified definiton for the base address of RTxxxx platforms.
- 2.8.3
  - Improvement:
    - Update the EHCI controller driver to support the address convert for TCM.
    - Update the USB host EHCI controller driver to make sure the mutual exclusion access under multiple tasks' environment.
- 2.8.2
  - Improvement:
    - Fix noise issue of UAC 3.1, UAC 5.1, UAC 7.1 on usb audio speaker demo.
    - Fix the issue that incorrect PC behavior when ejecting USB MSC devices.
    - Update the EHCI controller driver to support RW610 that does not reply on PHY driver, especially for low power feature.
    - Update the USB_HostHelperParseAlternateSetting to fix the wrong interface parse.
    - Update dev_composite_hid_audio_unified_bm demo to suppport independent mute/unmute and volume control.
- 2.8.1
  - Improvement:
    - update USB audio demos to use audio component (components/audio).
    - Add the checking of function call return value.
    - Add audio multiple channels demo (usb_device_composite_audio_multi_ch_unified) on RT600 audio board.
    - Fix audio noise on sync mode and improve overflow/underflow checking method.
    - Support UAC 3.1, 5.1 and 7.1 on audio speaker demo.
    - Set USB device CDC demo not to depend on DTR setting from host.
    - Support MCUX toolchain on some RTxxxx platforms.

- 2.8.0
  - Improvement:
    - Fix the USB device stack vulnerability issues.
    - Update the audio PLL and FRO adjustment codes for audio examples in RTxxx, LPC54xxx and LPC55xxx.
    - Improve the USB PD AMS collision avoidance.
    - Improve IP3511 controller driver's dedicated ram allocation.
    - Change the USB_DATA_ALIGN_SIZE to 4 because the controller driver uses the dedicated RAM to do memcpy.
  - New features:
    - Enable USB host audio recorder demo for mutilple boards.
- 2.7.0
  - Improvement:
    - Use new feeback solution and low latency playback for usb device speaker demo and unified demos. Add underflow and overflow protection.
    - Optimize hard code for usb audio demos.
    - Update Unconstrained Power field in the Sink Capabilities Message according to the external power state.
    - Fix CVE-2021-38258 and CVE-2021-38260
  - New features:
    - Enable USB host video demo for mutilple boards.
    - Enable USB device MTP demo for mutilple boards.
    - Add PPS message to usb pd stack.
- 2.6.1
  - Improvement:
    - rename sdcard as disk for all of sdcard demos. For ramdisk demos, they are not changed.
    - add wrapper for all of disk demos to support emmc.
- 2.6.0
  - Improvement:
    - Added more ufi event to support dynamic sdcard capacity.
    - Passed MISRA-2012 mandatory and required rules.
      - Except rule 17.2 in host hub and otg stack.
      - Except rule 5.1, rule 5.4, rule 21.1 and rule 21.2.
    - Re-implemented USB components and supported NPW.
    - Improved IP3511 controller driver's cancelling transfer function.
    - Enabled the audio2.0 defaultly for device audio demos.
    - Enabled the host audio2.0 function in host audio class driver and host audio speaker demo.
  - New features:
    - enable two USB controllers in one USB host mouse demo which named as host_hid_mouse_dual.
    - enable UAC 5.1 for usb device audio speaker demo.
- 2.5.0
  - Improvement:
    - Integrated sdk components (OSA, Timer, GPIO and serial_manager) to USB stack and demos.

    - Improved the ip3511 driver throughput.
    - Improved audio initialization codes after SDK audio drivers update.
    - Improved auido to support the audio2.0 in win10.
    - Add one "enumeration fail" callback event to host stack.
- 2.4.2
  - Improvement:
    - Put the USB controller data and transfer buffer to noncache section, removed the setting that sets the whole ocram and sdram as noncached.
    - Separated composite audio examples' channel,sample rate,format parameters from commom macro to in dedicated macro and out dedicated macro.
    - replaced USB_PrepareData with USB_AudioRecorderGetBuffer.
- 2.4.1
  - New features:
    - Added enumeration fail callback to host stack when the attached device's enumeration failed.
- 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

3 RTOS Change Log

kernel for MCUXpresso SDK.

The current version is FreeRTOS kernel 10.4.3-LTS-Patch-2 Original package is available at github.com/FreeRTOS/FreeRTOS-Kernel.

- 10.5.1_rev0
  - update amazon freertos version
- 10.4.3_rev1
  - Apply CM33 security fix from 10.4.3-LTS-Patch-2. See rtos\freertos\freertos_kernel\History.txt
  - Apply CM33 security fix from 10.4.3-LTS-Patch-1. See rtos\freertos\freertos_kernel\History.txt
- 10.4.3_rev0
  - update amazon freertos version.
- 10.4.3_rev0
  - update amazon freertos version.
- 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.

4 Component Change Log

CODEC

The current codec common driver version is 2.3.1.

- 2.3.1
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 16.1,16.3.
- 2.3.0
  - Improvements
    - Added enum _codec_volume_capability for CODEC_SetVolume/CODEC_SetMute to cover more volume configurations.
- 2.2.2
  - Bug Fixes
    - Fixed the typo in codec common driver.
- 2.2.1
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.3, 8.3, 10.7, 17.7.
- 2.2.0
  - Improvements
    - Used HAL_CODEC_HANDLER_SIZE which is determined by low level driver instead of use CODEC_HANDLE_SIZE for the codec device handle definition.
- 2.1.1
  - Improvements
    - Supported all of the codec in the codec adapter.
    - Modified the codec handle definition to improve user experience.
    - Modified the capability member type from entity to pointer in codec handle.
  - Bug Fixes
    - Fixed the Coverity issue regrading array compared agaist 0.
- 2.1.0
  - Deprecated APIs
    - CODEC_GetMappedFormatBits
    - CODEC_I2C_WriteReg
    - CODEC_I2C_ReadReg
    - CODEC_I2C_ModifyReg
    - CODEC_SetEncoding
  - new APIs
    - CODEC_SetPower
    - CODEC_SetVolume
    - CODEC_SetMute
    - CODEC_SetPlay
    - CODEC_SetRecord
    - CODEC_SetRecordChannel

        - CODEC_ModuleControl
    - new features
        - Removed duplicate members in codec_handle_t and codec_config_t.
        - Added codec_config_t pointer in codec_handle_t.
        - Added codec capability flag in codec_handle_t.
        - Used codec adapter instead of function opinter in codec common driver.
- 2.0.1
    - Added delayMs function pointer in codec handle.
- 2.0.0
    - Initial version.

WM8904

The current wm8904 driver version is 2.5.1.

- 2.5.1
    - Bug Fixes
        - Fixed invalid clock divider issue generated form WM8904_SetMasterClock api
        - Replace '__REV16' with general implementation to swap bytes in a short variable.
- 2.5.0
    - Improvements
        - Added master clock configuration support in function WM8904_SetAudioFormat.
        - Align the sysclk paramter definition for the WM8904_SetAudioFormat/WM8904_SetMasterClock.
        - Added api WM8904_SetDACVolume to support adjust DAC volume.
        - Fixed the MISRA-2012 violation of 12.2, 10.3.
- 2.4.4
    - Bug Fixes
        - Added the 11.025kHz/22.05kHz/44.1kHz samplerate support on codec WM8904.
        - Fixed the MISRA-2012 violation of 4.7.
- 2.4.3
    - Bug Fixes
        - Fixed the MISRA-2012 violations.
            - Fixed rule 8.6, 9.3, 10.1, 10.3, 10.4, 10.7, 10.8, 11.8, 11.9, 14.4, 16.1, 16.3, 16.4, 17.7, 20.9.
- 2.4.2
    - Bug Fixes
        - Corrected the volume setting function behavior in wm8904 driver, support range align with its specification range.
        - Corrected the volume setting function behavior in wm8904 adapter, support range 0 - 100, 0 for mute, 100 for maximum volume.
- 2.4.1
    - Bug Fixes
        - Fixed the bit width reigster field overwritten issue.

- 2.4.0
  - New features
    - Added fll support in wm8904 driver.
- 2.3.0
  - Improvements
    - Added new API WM8904_SetMasterClock to support BCLK/LRCLK output mode.
- 2.1.0
  - new APIs
    - WM8904_ReadRegister
    - WM8904_WriteRegister
    - WM8904_ModifyRegister
    - WM8904_SetRecord
    - WM8904_SetPlay
    - WM8904_SetRecordChannel
    - WM8904_SetModulePower
    - WM8904_SetChannelVolume
    - WM8904_SetChannelMute
  - New features
    - Removed dependency on codec common driver.
    - Added dependency on codec i2c.
  - Bug Fixes
    - Fixed unchecked return value in WM8904_Deinit.
    - Fixed the alignment fault issue by adding __NOP between continuous memory access.
- 2.0.3
  - Bug Fixes
    - Fixed issue that wm8904 register access function truncated return value.
- 2.0.2
  - Bug Fixes
    - Fixed using uninitialized value format.fsRatio when calling WM8904_UpdateFormate.
- 2.0.1
  - Added WM8904_CheckAudioFormat API.
  - Changed the second parameter's name of WM8904_SetAudioFormat to sysclk.
- 2.0.0
  - Initial version.

.1 WM8960

The current wm8960 driver version is 2.2.4.

- 2.2.4
  - Improvements
    - Remove CODEC_I2C_Deinit in WM8960_Deinit.
- 2.2.3
  - Improvements

    - Reinitialise I2C in Deinit function.
- 2.2.2
  - Bug fixes
    - Fixed violations of MISRA C-2012 rule 10.3.
- 2.2.1
  - Bug fixes
    - Improved the internal PLL fatctor calculation formula.
- 2.2.0
  - Improvements
    - Added masterClock member in wm8960_config_t to support wm8960 master mode.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 4.7, 5.8, 10.3, 10.4, 12.2, 14.4.
    - Added the bit clock divider configuration when wm8960 act as master.
- 2.1.3
  - Bug Fixes
    - Fixed the issue that WM8960 had no ack when performing write register by updating the byte count to be written.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.3, 8.3, 10.7, 17.7.
- 2.1.2
  - Improvements
    - Enabled the class D output in WM8960_Init.
  - Bug Fixes
    - Corrected the volume setting function behavior in wm8960 driver, support range aligned with its specification range.
    - Corrected the volume setting function behavior in wm8960 adapter, support range 0 - 100, 0 for mute, 100 for maximum volume.
- 2.1.1
  - Improvements
    - Removed useless bit clock divider configuration in function WM8960_ConfigDataFormat.
- 2.1.0
  - Improvements
    - Added new API WM8960_SetPlay.
    - Fixed error status overwrite issue in WM8960_ConfigDataFormat function.
    - Removed dependency on codec common driver.
    - Added dependency on codec i2c.
  - Bug Fixes
    - Fixed the alignment fault issue by adding __NOP between continuous memory access.
- 2.0.2
  - Removed bit width hard code setting in function WM8960_SetProtocol.
- 2.0.1
  - Corrected the bclk divider calculation.
- 2.0.0
  - Initial version.

SGTL5000

The current sgtl5000 driver version is 2.1.1.

- 2.1.1
  - Improvements
    - Corrected the volume setting function behavior in SGTL5000 driver, support range align with its specification range.
    - Corrected the volume setting function behavior in SGTL5000 adapter, support range 0 - 100, 0 for mute, 100 for maximum volume.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.3, 8.3, 10.7, 17.7.
- 2.1.0
  - Improvements
    - Added API SGTL_SetPlay/SGTL_SetRecord.
    - Removed dependency on codec common driver.
    - Added dependency on codec i2c.
    - Fixed divison or modulo by zero issue in SGTL_ConfigDataFormat function.
  - Bug Fixes
    - Fixed the alignment fault issue by adding __NOP between continuous memory access.
- 2.0.0
  - Initial version.

DA7212

The current da7212 driver version is 2.3.0.

- 2.3.0
  - Improvements
    - Add input sourece select in init function.
- 2.2.3
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.4, 4.7.
- 2.2.2
  - Bug Fixes
    - Fixed the MISRA-2012 violations.
      - Fixed rule 8.6, 9.3, 10.1, 10.3, 10.4, 10.7, 10.9, 11.1, 11.8, 14.4, 16.1, 16.3, 17.7, 17.3, 17.7, 20.9.
- 2.2.1
  - Improvements
    - Corrected the volume setting function behavior in DA7212 driver, support range align with its specification range.
    - Corrected the volume setting function behavior in DA7212 adapter, support range 0 - 100, 0 for mute, 100 for maximum volume.
- 2.2.0

  - Improvements
    - Added bclk invert parameter in the format structure.
    - Added API DA7212_SetMasterModeBits/DA7212_SetPLLConfig.
    - Added pll/sysClkSource parameters in the da7212 configuration structure.
    - Disbaled PLL by default.
- 2.1.0
  - Improvements
    - Removed dependency on codec common driver.
    - Added dependency on codec i2c.
  - Bug Fixes
    - Fixed the alignment fault issue by adding __NOP between continuous memory access.
- 2.0.0
  - Initial version.

CS42888

The current cs42888 driver version is 2.1.3

- 2.1.3
  - Improvements
    - Removed the assertion for codec reset function pointer.
- 2.1.2
  - Improvements
    - Corrected the volume setting function behavior in CS42888 adapter, support range 0 - 100, 0 for mute, 100 for maximum volume.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 4.7, 10.3, 8.3, 10.7, 17.7.
    - Corrected the channel index during setting AIN volume in CS42888_Init.
- 2.1.1
  - Improvements
    - Used software delay with delayMs pointer not provided by application.
    - Fixed error status overwrite issue in CS42888_Init function.
    - Removed dependency on codec common driver.
    - Added API CS42888_SelectFunctionalMode/CS42888_SetChannelMute.
    - Added dependency on codec i2c.
- 2.1.0
  - Improvements
    - Unified CS42888 codec driver interface.
  - Bug Fixes
    - Corrected the ADC/DAC functional mode macro definitaion.
    - Added TDM and OLM mode support in the function CS42888_SetProtocol.
- 2.0.0
  - Initial version.

SERIAL_MANAGER

The current Serial_Manager component version is 1.0.2.

- 1.0.2
  - Add SerialManager_WriteTimeDelay()/SerialManager_ReadTimeDelay() for serial manager's read/write non-blocking mode.
- 1.0.1
  - Add prefixing fsl_component_xxx/fsl_adapter_xxx.
- 1.0.0
  - Initial version

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 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.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, Freescale, the Freescale logo, Kinetis, Processor Expert, and Tower are trademarks of NXP B.V. All other product or service names are the property of their respective owners. Arm, Cortex, Keil, Mbed, Mbed Enabled, and Vision are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.