MCUXpresso SDK Release Notes Supporting evkmcimx7ulp

Change Logs

NXP Semiconductors

Contents

Driver Change Log

Title Page No.

Middleware Change Log

Component Change Log

1 Driver Change Log

FUSION

Current FUSION driver version is 2.0.0

- 2.0.0
  - Initial version.

CLOCK

Current CLOCK driver version is 2.0.0

- 2.0.0
  - Initial version.

IOMUXC

The current IOMUXC driver version is 2.0.0.

- 2.0.0
  - Initial version.

RESET

The current RESET driver version is 2.0.0.

- 2.0.0
  - initial version.

UPOWER

The current UPOWER driver version is 2.0.2.

- 2.0.2
  - Remove api UPOWER_SetVoltageForFreq().
- 2.0.1
  - Add api UPOWER_SetPwrMgmtParam().
- 2.0.0
  - initial version.

SENTINEL

The current UPOWER driver version is 2.0.1.

- 2.0.1
  - Add api SENTINEL_SetPowerDown().
- 2.0.0
  - initial version.

LPI2C_CMSIS

Current LPI2C_CMSIS driver version is 2.1

- 2.1
  - Bug Fixes
    - Fixed the MISRA-2012 violations.
      - Fixed rule 8.4, 8.6, 10.1, 10.3, 10.4, 11.1, 11.8, 14.4, 16.1, 16.3, 17.7, 17.3, 17.7, 20.9.
- 2.0
  - Initial version.

LPSPI_CMSIS

Current LPSPI_CMSIS driver version is 2.4

- 2.4
  - Bug Fixes
    - Update driver to fix warnings reported by IAR v9.
- 2.3
  - Bug Fixes
    - Fixed the MISRA-2012 violations.
      - Fixed rule 8.4, 8.6, 10.1, 10.3, 10.4, 11.1, 11.8, 14.4, 16.1, 16.3, 17.7, 17.3, 17.7, 20.9.
- 2.2
  - Bug Fixes
    - Fixed the bug that, the parameter num of APIs ARM_SPI_Transfer, ARM_SPI_Send and ARM_SPI_Receive, and the return value of API ARM_SPI_GetDataCount should be the number of data item defined by datawidth, rather than the number of byte.
- 2.1
  - Bug Fixes
    - Fixed the incorrect clock polarity assignment in the driver. For ARM_SPI_CPOL0_CPHA0 and other frame format parameters, CPOL = 0 means kSPI_ClockPolarityActiveHigh not kSPI_ClockPolarityActiveLow in driver.
  - New features
    - Allowed user to set up the default transmit value by using ARM_SPI_SET_DEFAULT_TX_VALUE. Please note that this is not supported in slave interrupts, because the pin will stay tristated if tX buffer is NULL.

    - Enabled slave select mode. Note this has no effect when user sets any of them because the driver can only support the hardware control function.
    - Enabled 3-Wire mode, user can use ARM_SPI_MODE_MASTER_SIMPLEX/ARM_SPI_MODE_SLAVE_SIMPLEX to enable this feature. For ARM_SPI_MODE_MASTER_SIMPLEX mode, the SOUT pin is selected as the input/output pin, and for ARM_SPI_MODE_SLAVE_SIMPLEX, the SIN pin is selected as the input/output pin.
- 2.0
  - Initial version.

ACMP

The current ACMP driver version is 2.0.6.

- 2.0.6
  - Bug Fixes
    - Fixed the wrong comments, the DAC value should range from 0 to 255.
- 2.0.5
  - Bug Fixes
    - Fixed the out-of-bounds error of Coverity caused by missing an assert sentence to avoid the return value of ACMP_GetInstance() exceeding the array bounds.
    - Fixed the violations of MISRA C-2012 rules:
      - Rule 10.1, 14.4, 16.4, 17.7.
- 2.0.4
  - Bug Fixes
    - Avoided changing w1c bit in ACMP_SetRoundRobinPreState().
- 2.0.3
  - New Features
    - Added feature functions for usage of different power domains(1.8 V and 3 V). These functions are first enabled in ULP1. They are about:
      - ACMP_EnableLinkToDAC()
      - ACMP_SetDiscreteModeConfig()
      - ACMP_GetDefaultDiscreteModeConfig()
- 2.0.2
  - Other Changes
    - Changed coding style of peripheral base address from "s_acmpBases" to "s_acmpBase".
- 2.0.1
  - Bug Fixes
    - Fixed bug regarding the function "ACMP_SetRoundRobinConfig". It will not continue execution but returns directly after disabling round robin mode.

CACHE

The current CACHE driver version is 2.0.6.

- 2.0.6
  - Bug Fixes
    - Fixed doxygen issue.
- 2.0.5
  - Improvements
    - Updated the cache enable function, don't enable again when it is already enabled.
- 2.0.4
  - Bug Fixes
    - Updated full name for lmem driver.
    - Fixed doxygen issue.
- 2.0.3
  - Bug Fixes
    - Fixed violation of MISRA C-2012 Rule 10.4 and 14.4.
- 2.0.2
  - Improvements
    - Moved CLCR register configuration out of the while loop, it's unnecessary to repeat this operation.
- 2.0.1
  - Bug Fixes
    - Fixed the over-4KB-size maintenance issue in invalidate/clean/clean&invalidate by range APIs.
- 2.0.0
  - Initial version.

COMMON

The current COMMON driver version is 2.3.2.

- 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.

CRC

The current CRC driver version is 2.0.3.

- 2.0.3
  - Bug fix:
    - Fix MISRA issues.
- 2.0.2
  - Bug fix:
    - Fix MISRA issues.
- 2.0.1
  - Bug fix:
    - DATA and DATALL macro definition moved from header file to source file.
- 2.0.0
  - Initial version.

DAC12

The current DAC12 driver version is 2.1.0.

- 2.1.0
  - Improvements
    - Defined the macro "FSL_FEATURE_HAS_NO_ITRM_REGISTER" to distinguish

different scenes that ITRM register may not equipped one some devices.

- 2.0.1
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rules:
      - Rule 10.8, 17.7.
- 2.0.0
  - Initial version.

DMAMUX

The current DMAMUX driver version is 2.0.5.

- 2.0.5
  - Improvements
    - Added feature FSL_FEATURE_DMAMUX_CHCFG_REGISTER_WIDTH for the difference of CHCFG register width.
- 2.0.4
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.4.
- 2.0.3
  - Bug Fixes
    - Fixed the issue for MISRA-2012 check.
      - Fixed rule 10.4 and rule 10.3.
- 2.0.2
  - New Features
    - Added an always-on enable feature to a DMA channel for ULP1 DMAMUX support.
- 2.0.1
  - Bug Fixes
    - Fixed the build warning issue by changing the type of parameter source from uint8_t to uint32_t when setting DMA request source in DMAMUX_SetSourceChange.
- 2.0.0
  - Initial version.

EDMA

The current eDMA driver version is 2.4.3.

- 2.4.3
  - Improvements
    - Added FSL_FEATURE_MEMORY_HAS_ADDRESS_OFFSET to convert the address between system mapped address and dma quick access address.
  - Bug Fixes
    - Fixed the wrong tcd done count calculated in first TCD interrupt for the non scatter gather case.

- 2.4.2
  - Bug Fixes
    - Fixed the wrong tcd done count calculated in first TCD interrupt by correct the initial value of the header.
    - Fixed violations of MISRA C-2012 rule 10.3, 10.4.
- 2.4.1
  - Bug Fixes
    - Added clear CITER and BITER registers in EDMA_AbortTransfer to make sure the TCD registers in a correct state for next calling of EDMA_SubmitTransfer.
    - Removed the clear DONE status for ESG not enabled case to aovid DONE bit cleared unexpectedly.
- 2.4.0
  - Improvements
    - Added api EDMA_EnableContinuousChannelLinkMode to support continuous link mode.
    - Added apis EDMA_SetMajorOffsetConfig/EDMA_TcdSetMajorOffsetConfig to support major loop address offset feature.
    - Added api EDMA_EnableChannelMinorLoopMapping for minor loop offset feature.
    - Removed the reduntant IRQ Handler in edma driver.
- 2.3.2
  - Improvements
    - Fixed HIS ccm issue in function EDMA_PrepareTransferConfig.
    - Fixed violations of MISRA C-2012 rule 11.6, 10.7, 10.3, 18.1.
  - Bug Fixes
    - Added ACTIVE & BITER & CITER bitfields to determine the channel status to fixed the issue of the transfer request cannot submit by function EDMA_SubmitTransfer when channel is idle.
- 2.3.1
  - Improvements
    - Added source/destination address alignment check.
    - Added driver IRQ handler support for multi DMA instance in one SOC.
- 2.3.0
  - Improvements
    - Added new api EDMA_PrepareTransferConfig to allow different configurations of width and offset.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.4, 10.1.
    - Fixed the Coverity issue regarding out-of-bounds write.
- 2.2.0
  - Improvements
    - Added peripheral-to-peripheral support in EDMA driver.
- 2.1.9
  - Bug Fixes
    - Fixed MISRA issue: Rule 10.7 and 10.8 in function EDMA_DisableChannelInterrupts and EDMA_SubmitTransfer.

    - Fixed MISRA issue: Rule 10.7 in function EDMA_EnableAsyncRequest.
- 2.1.8
  - Bug Fixes
    - Fixed incorrect channel preemption base address used in EDMA_SetChannelPreemption-Config API which causes incorrect configuration of the channel preemption register.
- 2.1.7
  - Bug Fixes
    - Fixed incorrect transfer size setting.
      - Added 8 bytes transfer configuration and feature for RT series;
      - Added feature to support 16 bytes transfer for Kinetis.
    - Fixed the issue that EDMA_HandleIRQ would go to incorrect branch when TCD was not used and callback function not registered.
- 2.1.6
  - Bug Fixes
    - Fixed KW3X MISRA Issue.
      - Rule 14.4, 10.8, 10.4, 10.7, 10.1, 10.3, 13.5, and 13.2.
  - Improvements
    - Cleared the IRQ handler unavailable for specific platform with macro FSL_FEATURE_-EDMA_MODULE_CHANNEL_IRQ_ENTRY_SHARED_OFFSET.
- 2.1.5
  - Improvements
    - Improved EDMA IRQ handler to support half interrupt feature.
- 2.1.4
  - Bug Fixes
    - Cleared enabled request, status during EDMA_Init for the case that EDMA is halted before reinitialization.
- 2.1.3
  - Bug Fixes
    - Added clear DONE bit in IRQ handler to avoid overwrite TCD issue.
    - Optimized above solution for the case that transfer request occurs in callback.
- 2.1.2
  - Improvements
    - Added interface to get next TCD address.
    - Added interface to get the unused TCD number.
- 2.1.1
  - Improvements
    - Added documentation for eDMA data flow when scatter/gather is implemented for the EDMA_HandleIRQ API.
    - Updated and corrected some related comments in the EDMA_HandleIRQ API and edma-_handle_t struct.
- 2.1.0
  - Improvements
    - Changed the EDMA_GetRemainingBytes API into EDMA_GetRemainingMajorLoop-Count due to eDMA IP limitation (see API comments/note for further details).
- 2.0.5

  - Improvements
    - Added pubweak DriverIRQHandler for K32H844P (16 channels shared).
- 2.0.4
  - Improvements
    - Added support for SoCs with multiple eDMA instances.
    - Added pubweak DriverIRQHandler for KL28T DMA1 and MCIMX7U5_M4.
- 2.0.3
  - Bug Fixes
    - Fixed the incorrect pubweak IRQHandler name issue, which caused re-definition build errors when client set his/her own IRQHandler, by changing the 32-channel IRQHandler name to DriverIRQHandler.
- 2.0.2
  - Bug Fixes
    - Fixed incorrect minorLoopBytes type definition in _edma_transfer_config struct, and defined minorLoopBytes as uint32_t instead of uint16_t.
- 2.0.1
  - Bug Fixes
    - Fixed the eDMA callback issue (which did not check valid status) in EDMA_HandleIRQ API.
- 2.0.0
  - Initial version.

EWM

The current EWM driver version is 2.0.3.

- 2.0.3
  - Bug Fixes
    - Fixed violation of MISRA C-2012 rules: 10.1, 10.3.
- 2.0.2
  - Bug Fixes
    - Fixed violation of MISRA C-2012 rules: 10.3, 10.4.
- 2.0.1
  - Bug Fixes
    - Fixed the hard fault in EWM_Deinit.
- 2.0.0
  - Initial version.

FLEXIO

The current FLEXIO driver version is 2.0.4.

- 2.0.4
  - Bug Fixes

        - Fixed MISRA 8.4 issues.
- 2.0.3
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
- 2.0.2
  - Improvements
    - Split FLEXIO component which combines all flexio/flexio_uart/flexio_i2c/flexio_i2s drivers into several components: FlexIO component, flexio_uart component, flexio_i2c_master component, and flexio_i2s component.
  - Bug Fixes
    - Fixed MISRA issues
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
- 2.0.1
  - Bug Fixes
    - Fixed the dozen mode configuration error in FLEXIO_Init API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.

FLEXIO_UART

The current FLEXIO_UART driver version is 2.4.0.

- 2.4.0
  - Improvements
    - Use separate data for TX and RX in flexio_uart_transfer_t.
  - Bug Fixes
    - Fixed bug that when ring buffer is used, if some data is received in ring buffer first before calling FLEXIO_UART_TransferReceiveNonBlocking, the received data count returned by FLEXIO_UART_TransferGetReceiveCount is wrong.
- 2.3.0
  - Improvements
    - Added check for baud rate's accuracy that returns kStatus_FLEXIO_UART_BaudrateNotSupport when the best achieved baud rate is not within 3% error of configured baud rate.
  - Bug Fixes
    - Added codes in FLEXIO_UART_TransferCreateHandle to clear pending NVIC IRQ before enabling NVIC IRQ, to fix issue of pending IRQ interfering the on-going process.
- 2.2.0
  - Improvements
    - Added timeout mechanism when waiting for certain states in transfer driver.
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
- 2.1.6
  - Bug Fixes
    - Fixed IAR Pa082 warnings.

        - Fixed MISRA issues
            - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
- 2.1.5
    - Improvements
        - Triggered user callback after all the data in ringbuffer were received in FLEXIO_UART_TransferReceiveNonBlocking.
- 2.1.4
    - Improvements
        - Unified component full name to FLEXIO UART(DMA/EDMA) Driver.
- 2.1.3
    - Bug Fixes
        - The following modifications support FLEXIO using multiple instances:
            - Removed FLEXIO_Reset API in module Init APIs.
            - Updated module Deinit APIs to reset the shifter/timer configuration instead of disabling module and clock.
            - Updated module Enable APIs to only support enable operation.
- 2.1.2
    - Bug Fixes
        - Fixed the transfer count calculation issue in FLEXIO_UART_TransferGetReceiveCount, FLEXIO_UART_TransferGetSendCount, FLEXIO_UART_TransferGetReceiveCountDMA, FLEXIO_UART_TransferGetSendCountDMA, FLEXIO_UART_TransferGetReceiveCountEDMA and FLEXIO_UART_TransferGetSendCountEDMA.
        - Fixed the Dozen mode configuration error in FLEXIO_UART_Init API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
        - Added code to report errors if the user sets a too-low-baudrate which FLEXIO cannot reach.
        - Disabled FLEXIO_UART receive interrupt instead of all NVICs when reading data from ring buffer. If ring buffer is used, receive nonblocking will disable all NVIC interrupts to protect the ring buffer. This had negative effects on other IPs using interrupt.
- 2.1.1
    - Bug Fixes
        - Changed the API name FLEXIO_UART_StopRingBuffer to FLEXIO_UART_TransferStopRingBuffer to align with the definition in C file.
- 2.1.0
    - New Features
        - Added Transfer prefix in transactional APIs.
        - Added txSize/rxSize in handle structure to record the transfer size.
    - Bug Fixes
        - Added an error handle to handle the situation that data count is zero or data buffer is NULL.

FLEXIO_UART_EDMA

The current FLEXIO_UART_EDMA driver version is 2.3.1.

- 2.3.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules.
- 2.3.0
  - Refer FLEXIO_UART driver change log to 2.3.0

FLEXIO_I2C

The current FLEXIO_I2C driver version is 2.4.0.

- 2.4.0
  - Improvements
    - Added delay of 1 clock cycle in FLEXIO_I2C_MasterTransferRunStateMachine to ensure that bus would be idle before next transfer if master is nacked.
    - Fixed issue that the restart setup time is less than the time in I2C spec by adding delay of 1 clock cycle before restart signal.
- 2.3.0
  - Improvements
    - Used 3 timers instead of 2 to support transfer which is more than 14 bytes in single transfer.
    - Improved FLEXIO_I2C_MasterTransferGetCount so that the API can check whether the transfer is still in progress.
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
- 2.2.0
  - New Features
    - Added timeout mechanism when waiting certain state in transfer API.
    - Added an API for checking bus pin status.
  - Bug Fixes
    - Fixed COVERITY issue of useless call in FLEXIO_I2C_MasterTransferRunStateMachine.
    - Fixed MISRA issues
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
    - Added codes in FLEXIO_I2C_MasterTransferCreateHandle to clear pending NVIC IRQ, disable internal IRQs before enabling NVIC IRQ.
    - Modified code so that during master's nonblocking transfer the start and slave address are sent after interrupts being enabled, in order to avoid potential issue of sending the start and slave address twice.
- 2.1.7
  - Bug Fixes
    - Fixed the issue that FLEXIO_I2C_MasterTransferBlocking did not wait for STOP bit sent.

    - Fixed COVERITY issue of useless call in FLEXIO_I2C_MasterTransferRunStateMachine.
    - Fixed the issue that I2C master did not check whether bus was busy before transfer.
- 2.1.6
  - Bug Fixes
    - Fixed the issue that I2C Master transfer APIs(blocking/non-blocking) did not support the situation of master transfer with subaddress and transfer data size being zero, which means no data followed the subaddress.
- 2.1.5
  - Improvements
    - Unified component full name to FLEXIO I2C Driver.
- 2.1.4
  - Bug Fixes
    - The following modifications support FlexIO using multiple instances:
      - Removed FLEXIO_Reset API in module Init APIs.
      - Updated module Deinit APIs to reset the shifter/timer config instead of disabling module/clock.
      - Updated module Enable APIs to only support enable operation.
- 2.1.3
  - Improvements
    - Changed the prototype of FLEXIO_I2C_MasterInit to return kStatus_Success if initialized successfully or to return kStatus_InvalidArgument if "(srcClock_Hz / masterConfig->baudRate_Bps) / 2 - 1" exceeds 0xFFU.
- 2.1.2
  - Bug Fixes
    - Fixed the FLEXIO I2C issue where the master could not receive data from I2C slave in high baudrate.
    - Fixed the FLEXIO I2C issue where the master could not receive NAK when master sent non-existent addr.
    - Fixed the FLEXIO I2C issue where the master could not get transfer count successfully.
    - Fixed the FLEXIO I2C issue where the master could not receive data successfully when sending data first.
    - Fixed the Dozen mode configuration error in FLEXIO_I2C_MasterInit API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
    - Fixed the issue that FLEXIO_I2C_MasterTransferBlocking API called FLEXIO_I2C_MasterTransferCreateHandle, which lead to the s_flexioHandle/s_flexioIsr/s_flexioType variable being written. Then, if calling FLEXIO_I2C_MasterTransferBlocking API multiple times, the s_flexioHandle/s_flexioIsr/s_flexioType variable would not be written any more due to it being out of range. This lead to the following situation: NonBlocking transfer APIs could not work due to the fail of register IRQ.
- 2.1.1
  - Bug Fixes
    - Implemented the FLEXIO_I2C_MasterTransferBlocking API which is defined in header file but has no implementation in the C file.

- 2.1.0
  - New Features
    - Added Transfer prefix in transactional APIs.
    - Added transferSize in handle structure to record the transfer size.

FLEXIO_SPI

The current FLEXIO_SPI driver version is 2.2.1.

- 2.2.1
  - Bug Fixes
    - Fixed bug in FLEXIO_SPI_MasterTransferAbortEDMA that when aborting EDMA transfer EDMA_AbortTransfer should be used rather than EDMA_StopTransfer.
- 2.2.0
  - Improvements
    - Added timeout mechanism when waiting certain states in transfer driver.
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
    - Added codes in FLEXIO_SPI_MasterTransferCreateHandle and FLEXIO_SPI_SlaveTransferCreateHandle to clear pending NVIC IRQ before enabling NVIC IRQ, to fix issue of pending IRQ interfering the on-going process.
- 2.1.3
  - Improvements
    - Unified component full name to FLEXIO SPI(DMA/EDMA) Driver.
  - Bug Fixes
    - Fixed MISRA issues
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
- 2.1.2
  - Bug Fixes
    - The following modification support FlexIO using multiple instances:
      - Removed FLEXIO_Reset API in module Init APIs.
      - Updated module Deinit APIs to reset the shifter/timer config instead of disabling module/clock.
      - Updated module Enable APIs to only support enable operation.
- 2.1.1
  - Bug Fixes
    - Fixed bug where FLEXIO SPI transfer data is in 16 bit per frame mode with eDMA.
    - Fixed bug when FLEXIO SPI works in eDMA and interrupt mode with 16-bit per frame and Lsbfirst.
    - Fixed the Dozen mode configuration error in FLEXIO_SPI_MasterInit/FLEXIO_SPI_SlaveInit API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
  - Improvements
    - Added #ifndef/#endif to allow users to change the default TX value at compile time.

- 2.1.0
  - New Features
    - Added Transfer prefix in transactional APIs.
    - Added transferSize in handle structure to record the transfer size.
  - Bug Fixes
    - Fixed the error register address return for 16-bit data write in FLEXIO_SPI_GetTxDataRegisterAddress.
    - Provided independent IRQHandler/transfer APIs for Master and slave to fix the baudrate limit issue.

FLEXIO_I2S

The current FLEXIO_I2S driver version is 2.2.0.

- 2.2.0
  - New Features
    - Added timeout mechanism when waiting certain state in transfer API.
  - Bug Fixes
    - Fixed IAR Pa082 warnings.
    - Fixed violations of the MISRA C-2012 rules 10.4, 14.4, 11.8, 11.9, 10.1, 17.7, 11.6, 10.3, 10.7.
- 2.1.6
  - Bug Fixes
    - Added reset flexio before flexio i2s init to make sure flexio status is normal.
- 2.1.5
  - Bug Fixes
    - Fixed the issue that I2S driver used hard code for bitwidth setting.
- 2.1.4
  - Improvements
    - Unified component's full name to FLEXIO I2S (DMA/EDMA) driver.
- 2.1.3
  - Bug Fixes
    - The following modifications support FLEXIO using multiple instances:
      - Removed FLEXIO_Reset API in module Init APIs.
      - Updated module Deinit APIs to reset the shifter/timer config instead of disabling module/clock.
      - Updated module Enable APIs to only support enable operation.
- 2.1.2
  - New Features
    - Added configure items for all pin polarity and data valid polarity.
    - Added default configure for pin polarity and data valid polarity.
- 2.1.1
  - Bug Fixes
    - Fixed FlexIO I2S RX data read error and eDMA address error.

        - Fixed FlexIO I2S slave timer compare setting error.
- 2.1.0
  - New Features
    - Added Transfer prefix in transactional APIs.
    - Added transferSize in handle structure to record the transfer size.

GPIO

The current driver version is 2.6.0.

- 2.6.0
  - New Features
    - Added API to get GPIO version information.
    - Added API to control a pin for general purpose input.
    - Added some APIs to control pin in secure and previliege status.
- 2.5.3
  - Bug Fixes
    - Correct the feature macro typo: FSL_FEATURE_GPIO_HAS_NO_INDEP_OUTPUT_-CONTORL.
- 2.5.2
  - Improvements
    - Improved GPIO_PortSet/GPIO_PortClear/GPIO_PortToggle functions to support devices without Set/Clear/Toggle registers.
- 2.5.1
  - Bug Fixes
    - Fixed wrong macro definition.
    - Fixed MISRA C-2012 rule issues in the FGPIO_CheckAttributeBytes() function.
    - Defined the new macro to separate the scene when the width of registers is different.
    - Removed some redundant macros.
  - New Features
    - Added some APIs to get/clear the interrupt status flag when the port doesn't control pins' interrupt.
- 2.4.1
  - Improvements
    - Improved GPIO_CheckAttributeBytes() function to support 8 bits width GACR register.
- 2.4.0
  - Improvements
    - API interface added:
      - New APIs were added to configure the GPIO interrupt clear settings.
- 2.3.2
  - Bug Fixes
    - Fixed the issue for MISRA-2012 check.
      - Fixed rule 3.1, 10.1, 8.6, 10.6, and 10.3.
- 2.3.1

  - Improvements
    - Removed deprecated APIs.
- 2.3.0
  - New Features
    - Updated the driver code to adapt the case of interrupt configurations in GPIO module. New APIs were added to configure the GPIO interrupt settings if the module has this feature on it.
- 2.2.1
  - Improvements
    - API interface changes:
      - Refined naming of APIs while keeping all original APIs by marking them as deprecated. The original APIs will be removed in next release. The main change is updating APIs with prefix of _PinXXX() and _PortXXX.
- 2.1.1
  - Improvements
    - API interface changes:
      - Added an API for the check attribute bytes.
- 2.1.0
  - Improvements
    - API interface changes:
      - Added "pins" or "pin" to some APIs' names.
      - Renamed "_PinConfigure" to "GPIO_PinInit".

LLWU

The current LLWU driver version is 2.0.5.

- 2.0.5
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.3.
    - Fixed the issue that function LLWU_SetExternalWakeupPinMode() does not work on 32-bit width platforms.
- 2.0.4
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.3, 10.4, 10.6, 10.7, 11.3.
    - Fixed issue that LLWU_ClearExternalWakeupPinFlag may clear other filter flags by mistake on platforms with 32-bit LLWU registers.
- 2.0.3
  - Bug Fixes
    - Fixed MISRA-2012 rules.
      - Rule 16.4.
- 2.0.2
  - Improvements
    - Corrected driver function LLWU_SetResetPinMode parameter name.

  - Bug Fixes
    - Fixed MISRA-2012 rules.
      - Rule 14.4, 10.8, 10.4, 10.3.
- 2.0.1
  - Other Changes
    - Updates for KL8x.
- 2.0.0
  - Initial version.

LPADC

The current LPADC driver version is 2.5.0.

- 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.

LPI2C

The current LPI2C driver version is 2.3.1.

- 2.3.1
  - Improvements
    - Updated LPI2C_GetCyclesForWidth to add the parameter of minimum cycle, because for master SDA/SCL filter, master bus idle/pin low timeout and slave SDA/SCL filter configuration, 0 means disabling the feature and cannot be used.
  - Bug Fixes
    - Fixed bug in LPI2C_SlaveTransferHandleIRQ that when restart detect event happens the transfer structure should not be cleared.
    - Fixed bug in LPI2C_RunTransferStateMachine, that when only slave address is transferred or there is still data remaining in tx FIFO the last byte's nack cannot be ignored.
    - Fixed bug in slave filter doze enable, that when FILTDZ is set it means disable rather than enable.
    - Fixed bug in the usage of LPI2C_GetCyclesForWidth. First its return value cannot be used directly to configure the slave FILTSDA, FILTSCL, DATAVD or CLKHOLD, because the real cycle width for them should be FILTSDA+3, FILTSCL+3, FILTSCL+DATAVD+3 and CLKHOLD+3. Second when cycle period is not affected by the prescaler value, prescaler value should be passed as 0 rather than 1.
    - Fixed wrong default setting for LPI2C slave. If enabling the slave tx SCL stall, then the default clock hold time should be set to 250ns according to I2C spec for 100kHz standard mode baudrate.
- 2.3.0
  - New Features
    - Supported reading more than 256 bytes of data in one transfer as master.
    - Added API LPI2C_GetInstance.
  - Bug Fixes

    - Fixed bug in LPI2C_MasterTransferAbortEDMA, LPI2C_MasterTransferAbort and LPI2C_MasterTransferHandleIRQ that before sending stop signal whether master is active and whether stop signal has been sent should be checked, to make sure no FIFO error or bus error will be caused.
    - Fixed bug in LPI2C master EDMA transactional layer that the bus error cannot be caught and returned by user callback, by monitoring bus error events in interrupt handler.
    - Fixed bug in LPI2C_GetCyclesForWidth that the parameter used to calculate clock cycle should be 2^prescaler rather than prescaler.
    - Fixed bug in LPI2C_MasterInit that timeout value should be configured after baudrate, since the timeout calculation needs prescaler as parameter which is changed during baudrate configuration.
    - Fixed bug in LPI2C_MasterTransferHandleIRQ and LPI2C_RunTransferStateMachine that when master writes with no stop signal, need to first make sure no data remains in the tx FIFO before finishes the transfer.
- 2.2.0
  - Bug Fixes
    - Fixed issue that the SCL high time, start hold time and stop setup time do not meet I2C specification, by changing the configuration of data valid delay, setup hold delay, clock high and low parameters.
    - MISRA C-2012 issue fixed.
      - Fixed rule 8.4, 13.5, 17.7, 20.8.
- 2.1.12
  - Bug Fixes
    - Fixed MISRA advisory 15.5 issues.
- 2.1.11
  - Bug Fixes
    - Fixed the bug that, during master non-blocking transfer, after the last byte is sent/received, the kLPI2C_MasterNackDetectFlag is expected, so master should not check and clear kLPI2C_MasterNackDetectFlag when remainingBytes is zero, in case FIFO is emptied when stop command has not been sent yet.
    - Fixed the bug that, during non-blocking transfer slave may nack master while master is busy filling tx FIFO, and NDF may not be handled properly.
- 2.1.10
  - Bug Fixes
    - MISRA C-2012 issue fixed.
      - Fixed rule 10.3, 14.4, 15.5.
    - Fixed unaligned access issue in LPI2C_RunTransferStateMachine.
    - Fixed uninitialized variable issue in LPI2C_MasterTransferHandleIRQ.
    - Used linked TCD to disable tx and enable rx in read operation to fix the issue that for platform sharing the same DMA request with tx and rx, during LPI2C read operation if interrupt with higher priority happened exactly after command was sent and before tx disabled, potentially both tx and rx could trigger dma and cause trouble.
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 11.6, 11.9, 14.4, 17.7.
    - Fixed the waitTimes variable not re-assignment issue for each byte read.

  - New Features
    - Added the IRQHandler for LPI2C5 and LPI2C6 instances.
  - Improvements
    - Updated the LPI2C_WAIT_TIMEOUT macro to unified name I2C_RETRY_TIMES.
- 2.1.9
  - Bug Fixes
    - Fixed Coverity issue of unchecked return value in I2C_RTOS_Transfer.
    - Fixed Coverity issue of operands did not affect the result in LPI2C_SlaveReceive and LPI2C_SlaveSend.
    - Removed STOP signal wait when NAK detected.
    - Cleared slave repeat start flag before transmission started in LPI2C_SlaveSend/LPI2C_-SlaveReceive. The issue was that LPI2C_SlaveSend/LPI2C_SlaveReceive did not handle with the reserved repeat start flag. This caused the next slave to send a break, and the master was always in the receive data status, but could not receive data.
- 2.1.8
  - Bug Fixes
    - Fixed the transfer issue with LPI2C_MasterTransferNonBlocking, kLPI2C_TransferNo-StopFlag, with the wait transfer done through callback in a way of not doing a blocking transfer.
    - Fixed the issue that STOP signal did not appear in the bus when NAK event occurred.
- 2.1.7
  - Bug Fixes
    - Cleared the stopflag before transmission started in LPI2C_SlaveSend/LPI2C_Slave-Receive. The issue was that LPI2C_SlaveSend/LPI2C_SlaveReceive did not handle with the reserved stop flag and caused the next slave to send a break, and the master always stayed in the receive data status but could not receive data.
- 2.1.6
  - Bug Fixes
    - Fixed driver MISRA build error and C++ build error in LPI2C_MasterSend and LPI2C_-SlaveSend.
    - Reset FIFO in LPI2C Master Transfer functions to avoid any byte still remaining in FIFO during last transfer.
    - Fixed the issue that LPI2C_MasterStop did not return the correct NAK status in the bus for second transfer to the non-existing slave address.
- 2.1.5
  - Bug Fixes
    - Extended the Driver IRQ handler to support LPI2C4.
    - Changed to use ARRAY_SIZE(kLpi2cBases) instead of FEATURE COUNT to decide the array size for handle pointer array.
- 2.1.4
  - Bug Fixes
    - Fixed the LPI2C_MasterTransferEDMA receive issue when LPI2C shared same request source with TX/RX DMA request. Previously, the API used scatter-gather method, which handled the command transfer first, then the linked TCD which was pre-set with the receive data transfer. The issue was that the TX DMA request and the RX DMA request

were both enabled, so when the DMA finished the first command TCD transfer and handled the receive data TCD, the TX DMA request still happened due to empty TX FIFO. The result was that the RX DMA transfer would start without waiting on the expected RX DMA request.
    - Fixed the issue by enabling IntMajor interrupt for the command TCD and checking if there was a linked TCD to disable the TX DMA request in LPI2C_MasterEDMACallback API.
- 2.1.3
  - Improvements
    - Added LPI2C_WATI_TIMEOUT macro to allow the user to specify the timeout times for waiting flags in functional API and blocking transfer API.
    - Added LPI2C_MasterTransferBlocking API.
- 2.1.2
  - Bug Fixes
    - In LPI2C_SlaveTransferHandleIRQ, reset the slave status to idle when stop flag was detected.
- 2.1.1
  - Bug Fixes
    - Disabled the auto-stop feature in eDMA driver. Previously, the auto-stop feature was enabled at transfer when transferring with stop flag. Since transfer was without stop flag and the auto-stop feature was enabled, when starting a new transfer with stop flag, the stop flag would be sent before the new transfer started, causing unsuccesful sending of the start flag, so the transfer could not start.
    - Changed default slave configuration with address stall false.
- 2.1.0
  - Improvements
    - API name changed:
      - LPI2C_MasterTransferCreateHandle -> LPI2C_MasterCreateHandle.
      - LPI2C_MasterTransferGetCount -> LPI2C_MasterGetTransferCount.
      - LPI2C_MasterTransferAbort -> LPI2C_MasterAbortTransfer.
      - LPI2C_MasterTransferHandleIRQ -> LPI2C_MasterHandleInterrupt.
      - LPI2C_SlaveTransferCreateHandle -> LPI2C_SlaveCreateHandle.
      - LPI2C_SlaveTransferGetCount -> LPI2C_SlaveGetTransferCount.
      - LPI2C_SlaveTransferAbort -> LPI2C_SlaveAbortTransfer.
      - LPI2C_SlaveTransferHandleIRQ -> LPI2C_SlaveHandleInterrupt.
- 2.0.0
  - Initial version.

LPIT

The current LPIT driver version is 2.0.2.

- 2.0.2
  - Improvements

    - Improved LPIT_SetTimerPeriod implementation, configure timeout value with LPIT ticks minus 1 generate more correct interval.
    - Added timeout value configuration check for LPIT_SetTimerPeriod, at least input 3 ticks for calling LPIT_SetTimerPeriod.
  - Bug Fixes
    - Fixed MISRA C-2012 rule 17.7 violations.
- 2.0.1
  - Bug Fixes
    - MISRA C-2012 issue fixed.
      - Fixed rules, containing: rule-10.3, rule-14.4, rule-15.5.
- 2.0.0
  - Initial version.

LPSPI

The current LPSPI driver version is 2.2.1.

- 2.2.1
  - Bug Fixes
    - Fixed bug in LPSPI_SetPCSContinous when disabling PCS continous mode.
- 2.2.0
  - Bug Fixes
    - Fixed bug in 3-wire polling and interrupt transfer that the received data is not correct and the PCS continous mode is not working.
- 2.1.0
  - Improvements
    - Improved LPSPI_SlaveTransferHandleIRQ to fill up TX FIFO instead of write one data to TX register which improves the slave transmit performance.
    - Added new functional APIs LPSPI_SelectTransferPCS and LPSPI_SetPCSContinous to support changing PCS selection and PCS continous mode.
  - Bug Fixes
    - Fixed bug in non-blocking and EDMA transfer APIs that kStatus_InvalidArgument is returned if user configures 3-wire mode and full-duplex transfer at the same time, but transfer state is already set to kLPSPI_Busy by mistake causing following transfer can not start.
    - Fixed bug when LPSPI slave using EDMA way to transfer, tx should be masked when tx data is null, otherwise in 3-wire mode which tx/rx use the same pin, the received data will be interfered.
- 2.0.5
  - Improvements
    - Added timeout mechanism when waiting certain states in transfer driver.
  - Bug Fixes
    - Fixed the bug that LPSPI can not transfer large data using EDMA.
    - Fixed MISRA 17.7 issues.

    - Fixed variable overflow issue introduced by MISRA fix.
    - Fixed issue that rxFifoMaxBytes should be calculated according to transfer width rather than FIFO width.
    - Fixed issue that completion flag was not cleared after transfer completed.
- 2.0.4
  - Bug Fixes
    - Fixed in LPSPI_MasterTransferBlocking that master rxfifo may overflow in stall condition.
    - Eliminated IAR Pa082 warnings.
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 10.6, 11.9, 14.2, 14.4, 15.7, 17.7.
- 2.0.3
  - Bug Fixes
    - Removed LPSPI_Reset from LPSPI_MasterInit and LPSPI_SlaveInit, because this API may glitch the slave select line. If needed, call this function manually.
- 2.0.2
  - New Features
    - Added dummy data set up API to allow users to configure the dummy data to be transferred.
    - Enabled the 3-wire mode, SIN and SOUT pins can be configured as input/output pin.
- 2.0.1
  - Bug Fixes
    - Fixed the bug that the clock source should be divided by the PRESCALE setting in LPSPI_MasterSetDelayTimes function.
    - Fixed the bug that LPSPI_MasterTransferBlocking function would hang in some corner cases.
  - Optimization
    - Added #ifndef/#endif to allow user to change the default TX value at compile time.
- 2.0.0
  - Initial version.

LPTMR

The current LPTMR driver version is 2.1.1.

- 2.1.1
  - Improvements
    - Updated the characters from "PTMR" to "LPTMR" in "FSL_FEATURE_PTMR_HAS_NO_PRESCALER_CLOCK_SOURCE_1_SUPPORT" feature definition.
- 2.1.0
  - Improvements
    - Implement for some special devices' not supporting for all clock sources.
  - Bug Fixes
    - Fixed issue when accessing CMR register.

- 2.0.2
  - Bug Fixes
    - Fixed MISRA-2012 issues.
      - Rule 10.1.
- 2.0.1
  - Improvements
    - Updated the LPTMR driver to support 32-bit CNR and CMR registers in some devices.
- 2.0.0
  - Initial version.

LPUART

The current LPUART driver version is 2.5.2.

- 2.5.2
  - Bug Fixes
    - Fixed bug that when setting watermark for TX or RX FIFO, the value may exceed the maximum limit.
  - Improvements
    - Added check in LPUART_TransferDMAHandleIRQ and LPUART_TransferEdmaHandleIRQ to ensure if user enables any interrupts other than transfer complete interrupt, the dma transfer is not terminated by mistake.
- 2.5.1
  - Improvements
    - Use separate data for TX and RX in lpuart_transfer_t.
  - Bug Fixes
    - Fixed bug that when ring buffer is used, if some data is received in ring buffer first before calling LPUART_TransferReceiveNonBlocking, the received data count returned by LPUART_TransferGetReceiveCount is wrong.
- 2.5.0
  - Bug Fixes
    - Added missing interrupt enable masks kLPUART_Match1InterruptEnable and kLPUART_Match2InterruptEnable.
    - Fixed bug in LPUART_EnableInterrupts, LPUART_DisableInterrupts and LPUART_GetEnabledInterrupts that the BAUD[LBKDIE] bit field should be soc specific.
    - Fixed bug in LPUART_TransferHandleIRQ that idle line interrupt should be disabled when rx data size is zero.
    - Deleted unused status flags kLPUART_NoiseErrorInRxDataRegFlag and kLPUART_ParityErrorInRxDataRegFlag, since firstly their function are the same as kLPUART_NoiseErrorFlag and kLPUART_ParityErrorFlag, secondly to obtain them one data word must be read out thus interfering with the receiving process.
    - Fixed bug in LPUART_GetStatusFlags that the STAT[LBKDIF], STAT[MA1F] and STAT[MA2F] should be soc specific.
    - Fixed bug in LPUART_ClearStatusFlags that tx/rx FIFO is reset by mistake when clearing

flags.
        - Fixed bug in LPUART_TransferHandleIRQ that while clearing idle line flag the other bits should be masked in case other status bits be cleared by accident.
        - Fixed bug of race condition during LPUART transfer using transactional APIs, by disabling and re-enabling the global interrupt before and after critical operations on interrupt enable register.
        - Fixed DMA/eDMA transfer blocking issue by enabling tx idle interrupt after DMA/eDMA transmission finishes.
    - New Features
        - Added APIs LPUART_GetRxFifoCount/LPUART_GetTxFifoCount to get rx/tx FIFO data count.
        - Added APIs LPUART_SetRxFifoWatermark/LPUART_SetTxFifoWatermark to set rx/tx FIFO water mark.
- 2.4.1
    - Bug Fixes
        - Fixed MISRA advisory 17.7 issues.
- 2.4.0
    - New Features
        - Added APIs to configure 9-bit data mode, set slave address and send address.
- 2.3.1
    - Bug Fixes
        - Fixed MISRA advisory 15.5 issues.
- 2.3.0
    - Improvements
        - Modified LPUART_TransferHandleIRQ so that txState will be set to idle only when all data has been sent out to bus.
        - Modified LPUART_TransferGetSendCount so that this API returns the real byte count that LPUART has sent out rather than the software buffer status.
        - Added timeout mechanism when waiting for certain states in transfer driver.
- 2.2.8
    - Bug Fixes
        - Fixed issue for MISRA-2012 check.
            - Fixed rule-10.3, rule-14.4, rule-15.5.
        - Eliminated Pa082 warnings by assigning volatile variables to local variables and using local variables instead.
        - Fixed MISRA issues.
            - Fixed rules 10.1, 10.3, 10.4, 10.8, 14.4, 11.6, 17.7.
    - Improvements
        - Added check for kLPUART_TransmissionCompleteFlag in LPUART_WriteBlocking, LPUART_TransferHandleIRQ, LPUART_TransferSendDMACallback and LPUART_SendEDMACallback to ensure all the data would be sent out to bus.
        - Rounded up the calculated sbr value in LPUART_SetBaudRate and LPUART_Init to achieve more acurate baudrate setting. Changed osr from uint32_t to uint8_t since osr's bigest value is 31.
        - Modified LPUART_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.

- 2.2.7
  - Bug Fixes
    - Fixed issue for MISRA-2012 check.
      - Fixed rule-12.1, rule-17.7, rule-14.4, rule-13.3, rule-14.4, rule-10.4, rule-10.8, rule-10.3, rule-10.7, rule-10.1, rule-11.6, rule-13.5, rule-11.3, rule-13.2, rule-8.3.
- 2.2.6
  - Bug Fixes
    - Fixed the issue of register's being in repeated reading status while dealing with the IRQ routine.
- 2.2.5
  - Bug Fixes
    - Do not set or clear the TIE/RIE bits when using LPUART_EnableTxDMA and LPUART_EnableRxDMA.
- 2.2.4
  - Improvements
    - Added hardware flow control function support.
    - Added idle-line-detecting feature in LPUART_TransferNonBlocking function. If an idle line is detected, a callback is triggered with status kStatus_LPUART_IdleLineDetected returned. This feature may be useful when the received Bytes is less than the expected received data size. Before triggering the callback, data in the FIFO (if has FIFO) is read out, and no interrupt will be disabled, except for that the receive data size reaches 0.
    - Enabled the RX FIFO watermark function. With the idle-line-detecting feature enabled, users can set the watermark value to whatever you want (should be less than the RX FIFO size). Data is received and a callback will be triggered when data receive ends.
- 2.2.3
  - Improvements
    - Changed parameter type in LPUART_RTOS_Init struct from rtos_lpuart_config to lpuart_rtos_config_t.
  - Bug Fixes
    - Disabled LPUART receive interrupt instead of all NVICs when reading data from ring buffer. Otherwise when the ring buffer is used, receive nonblocking method will disable all NVICs to protect the ring buffer. This may has a negative effect on other IPs that are using the interrupt.
- 2.2.2
  - Improvements
    - Added software reset feature support.
    - Added software reset API in LPUART_Init.
- 2.2.1
  - Improvements
    - Added separate RX/TX IRQ number support.
- 2.2.0
  - Improvements
    - Added support of 7 data bits and MSB.
- 2.1.1

  - Improvements
    - Removed unnecessary check of event flags and assert in LPUART_RTOS_Receive.
    - Added code to always wait for RX event flag in LPUART_RTOS_Receive.
- 2.1.0
  - Improvements
    - Update transactional APIs.

LPUART_EDMA

The current LPUART_EDMA driver version is 2.4.0.

- 2.4.0
  - Refer LPUART driver change log 2.1.0 to 2.4.0

LPUART_FREERTOS

The current LPUART_FREERTOS driver version is 2.4.0.

- 2.4.0
  - Refer LPUART driver change log 2.1.0 to 2.4.0

MSMC

The current MSMC driver version is 2.1.2.

- 2.1.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.3.
- 2.1.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 3.1, 10.1, 10.4, 10.8, 14.4.
- 2.1.0
  - Added new APIs with FEATURE macros support: SMC_GetStopEntryStatus() SMC_ClearStopEntryStatus() SMC_SetForceBootOptionConfig() SMC_SRAMEnableLowPowerMode() SMC_SRAMEnableDeepSleepMode()
  - Updated APIs with FEATURE macros support: SMC_SetPowerModeStop() SMC_SetPowerModeVlpr() SMC_SetPowerModeLls() SMC_SetPowerModeVlls() SMC_ConfigureResetPinFilter()
- 2.0.0
  - Initial version.

MU

The Current MU driver version is 2.1.1.

- 2.1.1
  - Bug Fixes
    - Fixed MISRA C-2012 issues.
- 2.1.0
  - Improvements
    - Added new enum mu_msg_reg_index_t.
- 2.0.0
  - Initial version.

PMC0

The current PMC0 driver version is 2.1.0.

- 2.1.0
  - New features
    - Added temperature sensor control APIs.
- 2.0.0
  - Initial version.

PORT

The current PORT driver version is 2.3.0.

- 2.3.0
  - New Features
    - Added new APIs for Electrical Fast Transient(EFT) detect.
    - Added new API to configure port voltage range.
- 2.2.0
  - New Features
    - Added new api PORT_EnablePinDoubleDriveStrength.
- 2.1.1
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rules: 10.1, 10.411.311.8, 14.4.
- 2.1.0
  - New Features
    - Updated the driver code to adapt the case of the interrupt configurations in GPIO module. Will move the pin configuration APIs to GPIO module.
- 2.0.2
  - Other Changes
    - Added feature guard macros in the driver.
- 2.0.1

  - Other Changes
    * Added "const" in function parameter.
    * Updated some enumeration variables' names.

QSPI

The current QSPI driver version is 2.2.4.

- 2.2.4
  - Bug Fixes
    * Fixed violations of MISRA C-2012 rule 10.3.
- 2.2.3
  - Bug Fixes
    * Cleared buffer generic configuration when do software reset.
- 2.2.2
  - Bug Fixes
    * MISRA C-2012 issue fixed: rule 10.1 and 11.9.
- 2.2.1
  - Bug Fixes
    * Fixed violations of MISRA C-2012 rule 10.1, 10.3, 10.4, 10.6, 10.8, 11.3, 11.6, 11.8, 11.9, 14.4, 16.1, 16.4, 17.7.
- 2.2.0
  - New Features
    * Added new API QSPI_ClearCache to clear cache for new IP feature FSL_FEATURE_QSPI_SOCCR_HAS_CLR_LPCAC.
  - Bug Fixes
    * Fixed the QSPI_WriteBlocking API programming issue for low watermark, caused by previous improvement change of using TX watermark signal to fill the TX FIFO. Reverted change to previous implementation to use TX FIFO full flag for filling the FIFO. Improved previous API by accessing TX data register directly.
    * Fixed the issue that QSPI_SetIPCommandSize incorrectly triggered a transaction.
    * Fixed clock divider accurate issue when using internal QSPI internal divider.
    * Fixed build fail issue for some devices' not supporting API QSPI_SetDqsConfig for DQS configuration.
- 2.1.0
  - New Features
    * Added new API QSPI_SetDqsConfig for DQS configuration.
  - Improvements
    * Updated the QSPI_WriteBlocking API to fill the TX FIFO once there are bytes of TX watermark room in the FIFO. This will improve the performance of filling TX FIFO when watermark is high.
- 2.0.2
  - Improvements
    * New Macro function:

        - Added QSPI_LUT_SEQ() function for users to set LUT table easily.
        - Added LUT command macros for users to easy use.
      - Comment update:
        - Added the comments for the limitation of QSPI_ReadBlocking and QSPI_TransferReceiveBlocking.
- 2.0.1
  - Improvements
    - New API:
      - QSPI_SetReadArea to set the read area.
  - Bug Fixes
    - Fixed the issue that QSPI_UpdateLUT function only updated first LUT.
    - Fixed issue that some function that hardcode QSPI0 as base.
- 2.0.0
  - Initial version.

SAI

The current SAI driver version is 2.3.5

- 2.3.5
  - Improvements
    - Make driver to be aarch64 compatible.
- 2.3.4
  - Bug Fixes
    - Corrected the fifo combine feature macro used in driver.
- 2.3.3
  - Bug Fixes
    - Added bit clock polarity configuration when sai act as slave.
    - Fixed out of bound access coverity issue.
    - Fixed violations of MISRA C-2012 rule 10.3, 10.4.
- 2.3.2
  - Bug Fixes
    - Corrected the frame sync configuration when sai act as slave.
- 2.3.1
  - Bug Fixes
    - Corrected the peripheral name in function SAI0_DriverIRQHandler.
    - Fixed violations of MISRA C-2012 rule 17.7.
- 2.3.0
  - Bug Fixes
    - Fixed the build error caused by the SOC has no fifo feature.
- 2.2.3
  - Bug Fixes
    - Corrected the peripheral name in function SAI0_DriverIRQHandler.
- 2.2.2

  - Bug Fixes
    - Fixed the issue of MISRA 2004 rule 9.3.
    - Fixed sign-compare warning.
    - Fixed the PA082 build warning.
    - Fixed sign-compare warning.
    - Fixed violations of MISRA C-2012 rule 10.3,17.7,10.4,8.4,10.7,10.8,14.4,17.7,11.6,10.-1,10.6,8.4,14.3,16.4,18.4.
    - Allow to reset Rx or Tx FIFO pointers only when Rx or Tx is disabled.
  - Improvements
    - Added 24bit raw audio data width support in sai sdma driver.
    - Disabled the interrupt/DMA request in the SAI_Init to avoid generates unexpected sai FIFO requests.
- 2.2.1
  - Improvements
    - Added mclk post divider support in function SAI_SetMasterClockDivider.
    - Removed useless configuration code in SAI_RxSetSerialDataConfig.
  - Bug Fixes
    - Fixed the SAI SDMA driver build issue caused by the wrong structure member name used in the function SAI_TransferRxSetConfigSDMA/SAI_TransferTxSetConfigSDMA.
    - Fixed BAD BIT SHIFT OPERATION issue caused by the FSL_FEATURE_SAI_CHANNEL_COUNTn.
    - Applied ERR05144: not set FCONT = 1 when TMR $>$ 0, otherwise the TX may not work.
- 2.2.0
  - Improvements
    - Added new APIs for parameters collection and simplified user interfaces:
      - SAI_Init
      - SAI_SetMasterClockConfig
      - SAI_TxSetBitClockRate
      - SAI_TxSetSerialDataConfig
      - SAI_TxSetFrameSyncConfig
      - SAI_TxSetFifoConfig
      - SAI_TxSetBitclockConfig
      - SAI_TxSetConfig
      - SAI_TxSetTransferConfig
      - SAI_RxSetBitClockRate
      - SAI_RxSetSerialDataConfig
      - SAI_RxSetFrameSyncConfig
      - SAI_RxSetFifoConfig
      - SAI_RxSetBitclockConfig
      - SAI_RXSetConfig
      - SAI_RxSetTransferConfig
      - SAI_GetClassicI2SConfig
      - SAI_GetLeftJustifiedConfig
      - SAI_GetRightJustifiedConfig
      - SAI_GetTDMConfig

- 2.1.9
  - Improvements
    - Improved SAI driver comment for clock polarity.
    - Added enumeration for SAI for sample inputs on different edges.
    - Changed FSL_FEATURE_SAI_CHANNEL_COUNT to FSL_FEATURE_SAI_CHANNEL_COUNTn(base) for the difference between the different SAI instances.
  - Added new APIs:
    - SAI_TxSetBitClockDirection
    - SAI_RxSetBitClockDirection
    - SAI_RxSetFrameSyncDirection
    - SAI_TxSetFrameSyncDirection
- 2.1.8
  - Improvements
    - Added feature macro test for the sync mode2 and mode 3.
    - Added feature macro test for masterClockHz in sai_transfer_format_t.
- 2.1.7
  - Improvements
    - Added feature macro test for the mclkSource member in sai_config_t.
    - Changed "FSL_FEATURE_SAI5_SAI6_SHARE_IRQ" to "FSL_FEATURE_SAI_SAI5_SAI6_SHARE_IRQ".
    - Added #ifndef #endif check for SAI_XFER_QUEUE_SIZE to allow redefinition.
  - Bug Fixes
    - Fixed build error caused by feature macro test for mclkSource.
- 2.1.6
  - Improvements
    - Added feature macro test for mclkSourceClockHz check.
    - Added bit clock source name for general devices.
  - Bug Fixes
    - Fixed incorrect channel numbers setting while calling RX/TX set format together.
- 2.1.5
  - Bug Fixes
    - Corrected SAI3 driver IRQ handler name.
    - Added I2S4/5/6 IRQ handler.
    - Added base in handler structure to support different instances sharing one IRQ number.
  - New Features
    - Updated SAI driver for MCR bit MICS.
    - Added 192 KHZ/384 KHZ in the sample rate enumeration.
    - Added multi FIFO interrupt/SDMA transfer support for TX/RX.
    - Added an API to read/write multi FIFO data in a blocking method.
    - Added bclk bypass support when bclk is same with mclk.
- 2.1.4
  - New Features
    - Added an API to enable/disable auto FIFO error recovery in platforms that support this feature.
    - Added an API to set data packing feature in platforms which support this feature.

- 2.1.3
  - New Features
    - Added feature to make I2S frame sync length configurable according to bitWidth.
- 2.1.2
  - Bug Fixes
    - Added 24-bit support for SAI eDMA transfer. All data shall be 32 bits for send/receive, as eDMA cannot directly handle 3-Byte transfer.
- 2.1.1
  - Improvements
    - Reduced code size while not using transactional API.
- 2.1.0
  - Improvements
    - API name changes:
      - SAI_GetSendRemainingBytes -> SAI_GetSentCount.
      - SAI_GetReceiveRemainingBytes -> SAI_GetReceivedCount.
      - All names of transactional APIs were added with "Transfer" prefix.
      - All transactional APIs use base and handle as input parameter.
      - Unified the parameter names.
  - Bug Fixes
    - Fixed WLC bug while reading TCSR/RCSR registers.
    - Fixed MOE enable flow issue. Moved MOE enable after MICS settings in SAI_TxInit/SAI_RxInit.
- 2.0.0
  - Initial version.

SEMA42

The current SEMA42 driver version is 2.0.2.

- 2.0.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 17.7.
- 2.0.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.3, 10.4, 14.4, 18.1.
- 2.0.0
  - Initial version.

TPM

The current TPM driver version is 2.2.0.

- 2.2.0
  - Improvements

    - Added TPM_SetChannelPolarity to support select channel input/output polarity.
    - Added TPM_EnableChannelExtTrigger to support enable external trigger input to be used by channel.
    - Added TPM_CalculateCounterClkDiv to help calculates the counter clock prescaler.
    - Added TPM_GetChannelValue to support get TPM channel value.
    - Added new TPM configuration.
      - syncGlobalTimeBase
      - extTriggerPolarity
      - chnlPolarity
    - Added new PWM signal configuration.
      - secPauseLevel
  - Bug Fixes
    - Fixed TPM_SetupPwm can't configure 0% combined PWM issues.
- 2.1.1
  - Improvements
    - Add feature macro for PWM pause level select feature.
- 2.1.0
  - Improvements
    - Added TPM_EnableChannel and TPM_DisableChannel APIs.
    - Added new PWM signal configuration.
      - pauseLevel
      - Support select output level when counter first enabled or paused.
      - enableComplementary
      - Support enable/disable generate complementary PWM signal.
      - deadTimeValue
      - Support deadtime insertion for each pair of channels in combined PWM mode.
  - Bug Fixes
    - Fixed issues about channel MSnB:MSnA and ELSnB:ELSnA bit fields and CnV register change request acknowledgement. Writes to these bits are ignored when the interval between successive writes is less than the TPM clock period.
- 2.0.8
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.1, 10.4 ,10.7 and 14.4.
- 2.0.7
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.4 and 17.7.
- 2.0.6
  - Bug Fixes
    - Fixed Out-of-bounds issue.
- 2.0.5
  - Bug Fixes
    - Fixed MISRA-2012 rules.
      - Rule 10.6, 10.7
- 2.0.4
  - Bug Fixes

        - Fixed ERR050050 in functions TPM_SetupPwm/TPM_UpdatePwmDutycycle. When TPM was configured in EPWM mode as PS = 0, the compare event was missed on the first reload/overflow after writing 1 to the CnV register.
- 2.0.3
  - Bug Fixes
    - MISRA-2012 issue fixed.
      - Fixed rules: rule-12.1, rule-17.7, rule-16.3, rule-14.4, rule-1.3, rule-10.4, rule-10.3, rule-10.7, rule-10.1, rule-10.6, and rule-18.1.
- 2.0.2
  - Bug Fixes
    - Fixed issues in functions TPM_SetupPwm/TPM_UpdateChnlEdgeLevelSelect /TPM_SetupInputCapture/TPM_SetupOutputCompare/TPM_SetupDualEdgeCapture, wait acknowledgement when the channel is disabled.
- 2.0.1
  - Bug Fixes
    - Fixed TPM_UpdateChnIEdgeLevelSelect ACK wait issue.
    - Fixed the issue that TPM_SetupdualEdgeCapture could not set FILTER register.
    - Fixed TPM_UpdateChnEdgeLevelSelect ACK wait issue.
- 2.0.0
  - Initial version.

TRGMUX

The current TRGMUX driver version is 2.0.1.

- 2.0.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.1, 10.3, 10.8.
- 2.0.0
  - Initial version.

TSTMR

The current TSTMR driver version is 2.0.0.

- 2.0.0
  - Initial version.

WDOG32

The current WDOG32 driver version is 2.0.4.

- 2.0.4
  - Improvements

    - To ensure that the reconfiguration is inside 128 bus clocks unlock window, put all reconfiguration APIs in quick access code section.
- 2.0.3
  - Bug Fixes
    - Fixed the noncompliance issue of the reference document.
      - Waited until for new configuration to take effect by checking the RCS bit field.
      - Waited until for registers to be unlocked by checking the ULK bit field.
  - Improvements
    - Added 128 bus clocks delay ensures a smooth transition before restarting the counter with the new configuration when there is no RCS status bit.
- 2.0.2
  - Bug Fixes
    - MISRA C-2012 issue fixed.
      - Fixed rules, containing: rule-10.3, rule-14.4, rule-15.5.
    - Fixed the issue of the inseparable process interrupted by other interrupt source.
      - WDOG32_Refresh
- 2.0.1
  - Bug Fixes
    - WDOG must be configured within its configuration time period.
      - Added WDOG32_Init API to quick access section.
      - Defined register variable in WDOG32_Init API.
- 2.0.0
  - Initial version.

2 Middleware Change Log

Multicore SDK

The current version of Multicore SDK is 2.11.1.

- 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, Git-

Hub 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

USB stack for MCUXpresso SDK

The current version of USB stack is 2.8.1.

- 2.8.1
  - Improvement:
    - update USB audio demos to use audio component (components).
    - 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 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 Component Change Log

CODEC

The current codec common driver version is 2.3.0.

- 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.

.1 WM8960

The current wm8960 driver version is 2.2.0.

- 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 2.2.2.

- 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.

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