MCUXpresso SDK Release Notes Supporting frdmk66f

Change Logs

NXP Semiconductors

Contents

Driver Change Log

Title Page No.

1 Driver Change Log

CLOCK

Current CLOCK driver version is 2.5.2

- 2.5.2
  - Bug Fixes
    - Fixed violation of MISRA C-2012 rules.
- 2.5.1
  - Bug Fixes
    - Fixed MISRA C-2012 rule 10.1,rule 10.4,rule 10.8,rule 15.5 and so on.
    - Fixed IAR warning Pa082 for the clock driver.
- 2.5.0
  - New Features
    - Moved SDK_DelayAtLeastUs function from clock driver to common driver.
- 2.4.0
  - New Features -Added two APIs to set slow and fast internal reference clock variable.
- 2.3.0
  - Bug Fixes
    - Fixed the issue for MISRA-2012 check.
      - Fixed rule 10.4, rule 10.1, rule 10.6, rule 13.5, rule 10.8.
      - Fixed incorrect External Oscillator Configuration sequence and ensure oscillator configuration be executed before it be enabled.
  - New Features
    - Added new API CLOCK_DelayAtLeastUs() implemented by DWT to allow users set delay in unit of microsecond.
- 2.2.2
  - Bug Fixes
    - Fixed the issue that MCG could not switch to FEE/FBE/PBE modes when OSCERCLK clock not enabled.
- 2.2.1
  - New Features
    - Added APIs for USB HS PFD clock.
- 2.2.0
  - New Features
    - Updated CLOCK_SetFeiMode/CLOCK_SetFbiMode/CLOCK_BootToFeiMode() to support set MCG_C4[DMX32]=1 in FEI/FBI modes.
  - Bug Fixes
    - Updated IP_CLOCKS array, remove unused gates and add missing gates.
- 2.1.0
  - Other Changes
    - Merged fsl_mcg and fsl_osc into fsl_clock.
- 2.0.0

    - Initial version.

DSPI_CMSIS

Current dspi_cmsis driver version is 2.3

- 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 wrong clock polarity assignment in 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.
        - 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.
- 2.0
    - Initial version.

ENET

Current ENET CMSIS driver version is 2.2

- 2.2
    - New Features
        - Added code to deal with 1G enet and RGMII interface configuration in cmsis enet driver.
- 2.1
    - Bug Fixes
        - Fixed the wrong logic to control cache macro.
- 2.0
    - Initial version.

I2C

Current I2C CMSIS driver version is 2.2.0

- 2.2.0
  - 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.
- 2.0.1
  - Bug Fixes
    - In ARM_I2C_ABORT_TRANSFER operation in I2C_InterruptControl, the method to check if I2C is operating as slave is not correct, then master may have potential risk to block at the slave check code.
- 2.0.0
  - Initial version.

UART

The current UART CMSIS driver version is 2.1

- 2.1
  - Bug Fixes
    - Fixed the MISRA-2012 violations.
      - Fixed rule 8.48.610.110.310.411.111.914.415.716.116.316.416.617.720.720.9.
- 2.0
  - Initial version.

ADC16

The current ADC16 driver version is 2.2.0.

- 2.2.0
  - Improvements
    - Added hardware average mode in adc_config_t structure, then the hardware average mode can be set by invoking ADC16_Init() function.
- 2.1.0
  - New Features:
    - Supported KM series' new ADC reference voltage source, bandgap from PMC.
- 2.0.3
  - Bug Fixes
    - Fixed IAR warning Pa082: the order of volatile access should be defined.
- 2.0.2
  - Improvements
    - Used conversion control feature macro instead of that in IO map.
- 2.0.1

- Bug Fixes
  - Fixed MISRA-2012 rules.
    - Rule 16.4, 10.1, 13.2, 14.4 and 17.7.
- 2.0.0
  - Initial version

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.

CMP

The current CMP driver version is 2.0.2.

- 2.0.2
  - Bug Fixes
    - Fixed the violations of MISRA 2012 rules:
      - Rule 10.3
- 2.0.1
  - Bug Fixes

    - Fixed MISRA-2012 rules.
      - Rule 14.4, rule 10.3, rule 10.1, rule 10.4 and rule 17.7.
- 2.0.0
  - Initial version.

CMT

The current CMT driver version is 2.0.3.

- 2.0.3
  - Bug Fixes
    - Fixed MISRA C-2012 rule 15.5.
- 2.0.2
  - Bug Fixes
    - Fixed MISRA-2012 rules.
      - Rule 14.4, rule 10.4, rule 10.3, rule 10.7, rule 10.1, rule 10.6, and rule 17.7.
- 2.0.1
  - Other Changes
    - Added static to global CMT variables.
- 2.0.0
  - Initial version.

COMMON

The current COMMON driver version is 2.3.1.

- 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_DELA-Y_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_S-ECTION" 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.

DAC

The current DAC driver version is 2.0.2.

- 2.0.2
  - Bug Fixes
    - Fixed MISRA-2012 issues:
      - Rule 10.3, 10.8 and 17.7.
- 2.0.1
  - Bug Fixes
    - Moved the default DAC_Enable(..., true) from DAC_Init() to the application code so that users can enable the DAC's output.

- 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.

DSPI

The current dspi driver version is 2.2.4.

- 2.2.4
  - Bug Fixes
    - Fixed bug that instance with shared TR/RX EDMA request cannot transfer 1 datawidth of data as master in single transfer.
- 2.2.3
  - Improvements
    - Added macro of getting maximum transfer size using EDMA way.
- 2.2.2
  - Bug Fixes
    - MISRA C-2012 issue fixed:
      - Fixed rules, containing: 10.8, 10.3.
    - Fixed the build warning issue.

    * Fixed the bug that PCS would temporarily pull down for a while during master initialization.
    * Fixed compiling error of undefined identifier g_dspiDummyData.
    * Changed the type of kDSPI_AllStatusFlag and kDSPI_AllInterruptEnable from int to uint32_t to fix Pe068 warning of integer conversion.
    * Eliminated IAR Pa082 warning in DSPI_MasterTransferDMA.
- 2.2.1
  - Bug Fixes
    * Fixed the bug for double execution of transfer complete callback in master interrupt transfer mode. In the interrupt routine, the DSPI interrupt may drop in the situation of the interrupt pending by itself while receiving the last frame, adding check to the transfer state to execute the callback function.
    * Fixed wrong logic in DSPI_SetFifoEnable().
    * MISRA C-2012 issue fixed.
      · Fixed rules, containing: rule-12.1, rule-17.7, rule-16.4, rule-14.4, rule-10.4, rule-10.8, rule-10.3, rule-10.1, rule-10.6, rule-13.5, rule-11.3, rule-13.2, rule-8.3, and rule-8.5.
- 2.2.0
  - New Features
    * Added gasket feature for SPI EDMA driver, which reduces one channel used in the EDMA master transfer. With this feature support, only two channels are needed. For example, if the gasket feature is supported, we could use the DSPI_MasterTransfer-CreateHandleEDMA function like below: DSPI_MasterTransferCreateHandleEDMA(E-XAMPLE_DSPI_MASTER_BASEADDR, &g_dspi_edma_m_handle, DSPI_Master-UserCallback, &userData, &dspiEdmaMasterRxRegToRxDataHandle, NULL, &dspi-EdmaMasterIntermediaryToTxRegHandle);
    * Added dummy data setup API to allow users to configure the dummy data to be transferred.
    * Added new APIs for half-duplex transfer function. Users can send and receive data by one API in the polling/interrupt/EDMA way, and they can choose either to transmit first or to receive first. Additionally, the PCS pin can be configured as assert status in transmission (between transmit and receive) by setting the isPcsAssertInTransfer to true.
- 2.1.4
  - Bug Fixes
    * DSPI EDMA driver: the DSPI instance has been separated, so the DMA request source can now transfer up to 32767 Bytes data in one DSPI_MasterTransferEDMA() transfer.
- 2.1.3
  - Bug Fixes
    * DSPI EDMA driver can no longer support the case that the transfer data size is odd, but the bitsPerFrame is greater than 8.
  - Improvements
    * Added #ifndef/#endif to allow users to change the default TX value at compile time.
- 2.1.2
  - Bug Fixes
    * DSPI_MasterTransferBlocking function would hang in some corner cases (for example, some cases with bitsPerFrame is 4,6 and kDSPI_MasterPcsContinuous transfer mode).

- 2.1.1
  - Bug Fixes
    - Set the EOQ (End Of Queue) bit to TRUE for the last transfer in transactional APIs.
- 2.1.0
  - New Features
    - Added Transfer prefix in transactional APIs.
- 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.

ENET

The current ENET driver version is 2.5.3.

- 2.5.3
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 11.6.
- 2.5.2
  - Improvements
    - Updated the TXIC/RXIC register handling code according to the new header file.
- 2.5.1
  - Bug Fixes
    - Fixed document typo.
- 2.5.0
  - Bug Fixes
    - Fixed the SendFrame/SendFrameZeroCopy functions issue with scattered buffers.
    - Updated the formula of MDC calculation.
    - Used a feature macro to distinguish the old IP design from the new design, because old IP design always reads a value zero from ATCR->CAPTURE bit. For old IP, driver caculates and wait the necessary delay cycles after setting ATCR->CAPTURE then gets the timestamp value.
  - New Features
    - Added new zero copy Tx/Rx function.
    - New zero copy Tx function combines scattered and contiguous Tx buffer in one API, it also supports more Tx featrues which buffer descriptor supports but previous Tx function doesn't support.
    - New zero copy Rx function use dynamic buffer mechanism and simpler interface.
  - Improvements
    - Corrected the interrupt handler for PTP timestamp IRQ and PTP1588 event IRQ since platform difference.
    - Added missing IRQ handlers for PTP1588 events on some platforms.
    - Corrected the max Tx frame length verification, it will not depend on a fixed macro. The ENET_FRAME_MAX_FRAMELEN is only an default value for driver, application can configure it. Driver caculates the limitation with the max frame length in register which may takes extended 4 or 8 bytes VLAN tag if VLAN/SVLAN enables.
    - Deleted deprecated Clause 45 read/write legacy APIs.
- 2.4.3
  - Improvements
    - Aligned the IRQ handler name with header file.
- 2.4.2
  - Bug Fixes
    - Fixed the MISRA issue of speculative out-of-bounds access.
- 2.4.1
  - Bug Fixes
    - Fixed the PTP time capture issue.
- 2.4.0

  - Improvements
    * Exposed API ENET_ReclaimTxDescriptor for user application to relaim tx descriptors in their application.
    * Added counter to record multicast hash conflict in struct _enet_handle, improved the situation that one multicast group could be left by other conflict multicast address left operation.
    * Improved concurrent usage of relaim and send frame operation.
- 2.3.4
  - Bug Fixes
    * Fixed the issue that interrupt handler only checks the interrupt event flag but not checks interrupt mask flag.
- 2.3.3
  - Bug Fixes
    * Fixed the issue that some compilers may choose the memcpy with 4-bit aligned address limitation due to the type of address pointer is 'unsigned int *', the data address doesn't have to be 4-bit aligned.
- 2.3.2
  - New Features
    * Added the feature that ENET driver can be used in the platform which integrates both 10/100M and 1G ENET IP.
    * Deleted duplicated code about ARM errata 838869 in first/second level IRQ handler.
- 2.3.1
  - Improvements
    * Added function pointer checking in IRQ handler to make sure code can be used even it runs into the interrupt when the second level interupt handler is NULL.
- 2.3.0
  - Bug Fixes
    * Fixed the issue that clause 45 MDIO read/write API doesn't check the transmission over status between two transmissions.
    * Fixed violations of the MISRA C-2012 rules 2.2,10.3,10.4,10.7,11.6,11.8,13.5,14.4,15.-7,17.7.
  - New Features
    * Added APIs to support send/receive frame with Zero-Copy.
  - Improvements
    * Separated the clock configuration from module configuration when init and deinit.
    * Added functions to set second level interrupt handler.
    * Provided new function to get 1588 timer count without disabling interrupt.
    * Improved timestamp controlling, deleted all old timestamp management APIs and data structures.
    * Merged the single/multiple ring(s) APIs, now these APIs can handle both.
    * Used base and index to control buffer descriptor, aligned with qos and lpc enet driver.
- 2.2.6
  - Bug Fixes
    * Updated MII speed formula referring to the manual.
- 2.2.5

- Bug Fixes
  - Fixed violations of the MISRA C-2012 rules 10.1, 10.3, 10.4, 10.6, 10.7, 11.6, 11.9, 13.5, 14.4, 16.4, 17.7, 21.15, 3.1, 8.4.
  - Changed to use ARRAY_SIZE(s_enetBases) as the array size for s_ENETHandle, fixed the hardfault issue for using some ENET instance when ARRAY_SIZE(s_enetBases) is not same as FSL_FEATURE_SOC_ENET_COUNT.

- 2.2.4
  - Improvements
    - Added call to Data Synchronization Barrier instruction before activating Tx/Rx buffer descriptor to ensure previous data update is completed.
    - Improved ENET_TransmitIRQHandler to store timestamps for multiple transmit buffer descriptors.
    - Bug Fixes
    - Fixed the issue that ENET_Ptp1588GetTimer did not handle the timer wrap situation.
- 2.2.3
  - Improvements
    - Improved data buffer cache maintenance in the ENET driver.
- 2.2.2
  - New Features
    - Added APIs for extended multi-ring support.
    - Added the AVB configure API for extended AVB feature support.
- 2.2.1
  - Improvements
    - Changed the input data pointer attribute to const in ENET_SendFrame().
- 2.1.1
  - New Features
    - Added the extended MDIO IEEE802.3 Clause 45 MDIO format SMI command APIs.
    - Added the extended interrupt coalescing feature.
  - Improvements
    - Combined all storage operations in the ENET_Init to ENET_SetHandler API.
- 2.0.1
  - Bug Fixes
    - Used direct transmit busy check when doing data transmit.
  - Miscellaneous Changes
    - Updated IRQ handler work flow.
    - Changed the TX/RX interrupt macro from kENET_RxByteInterrupt to kENET_RxBuffer-Interrupt, from kENET_TxByteInterrupt to kENET_TxBufferInterrupt.
    - Deleted unnecessary parameters in ENET handler.
- 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.

FLASH

Current FLASH driver version is 3.1.2

- 3.1.2
  - Bug Fixes — Remove redundant comments.
- 3.1.1
  - Bug Fixes — MISRA C-2012 issue fixed: rule 10.3
- 3.1.0
  - New Feature
    - Support erase flash asynchronously.
- 3.0.2
  - Bug Fixes — MISRA C-2012 issue fixed: rule 8.4, 17.7, 10.4, 16.1, 21.15, 11.3, 10.7 — building warning -Wnull-dereference on arm compiler v6
- 3.0.1
  - New Features
    - Added support FlexNVM alias for (kw37/38/39).
- 3.0.0
  - Improvements
    - Reorganized FTFx flash driver source file.
    - Extracted flash cache driver from FTFx driver.
    - Extracted flexnvm flash driver from FTFx driver.
- 2.3.1
  - Bug Fixes
    - Unified Flash IFR design from K3.
    - New encoding rule for K3 flash size.
- 2.3.0
  - New Features
    - Added support for device with LP flash (K3S/G).
    - Added flash prefetch speculation APIs.
  - Improvements
    - Refined flash_cache_clear function.

    * Reorganized the member of flash_config_t struct.
- 2.2.0
  - New Features
    * Supported FTFL device in FLASH_Swap API.
    * Supported various pflash start addresses.
    * Added support for KV58 in cache clear function.
    * Added support for device with secondary flash (KW40).
  - Bug Fixes
    * Compiled execute-in-ram functions as PIC binary code for driver use.
    * Added missed flexram properties.
    * Fixed unaligned variable issue for execute-in-ram function code array.
- 2.1.0
  - Improvements
    * Updated coding style to align with KSDK 2.0.
    * Different-alignment-size support for pflash and flexnvm.
    * Improved the implementation of execute-in-ram functions.
- 2.0.0
  - Initial version

FLEXCAN

The current FLEXCAN driver version is 2.8.2.

- 2.8.2
  - Bug Fixes
    * Fixed errors in timing calculations and simplify the calculation process.
    * Fixed issue of CBT and FDCBT register may write failure.
- 2.8.1
  - Bug Fixes
    * Fixed the issue of CAN FD three sampling points.
    * Added macro to support the devices that no MCR[SUPV] bit.
    * Remove unnecessary clear WMB operations.
- 2.8.0
  - Improvements
    * Update config configuration.
      · Added enableSupervisorMode member to support enable/disable Supervisor mode.
    * Simplified the algorithm in CAN FD improved timing APIs.
- 2.7.1
  - Bug Fixes
    * Fixed violations of the MISRA C-2012 rules 10.3, 10.7.
- 2.7.0
  - Improvements
    * Update config configuration.
      · Added enablePretendedeNetworking member to support enable/disable Pretended

        Networking feature.
      - Added enableTransceiverDelayMeasure member to support enable/disable Transceiver Delay MeasurementPretended feature.
      - Added bitRate/bitRateFD member to work as baudRate/baudRateFD member union.
    - Rename all "baud" in code or comments to "bit" to align with the CAN spec.
    - Added Pretended Networking mode related APIs.
      - FLEXCAN_SetPNConfig
      - FLEXCAN_GetPNMatchCount
      - FLEXCAN_ReadPNWakeUpMB
    - Added support for Enhanced Rx FIFO.
    - Removed independent memory error interrupt/status APIs and put all interrupt/status control operation into FLEXCAN_EnableInterrupts/FLEXCAN_DisableInterrupts and F-LEXCAN_GetStatusFlags/FLEXCAN_ClearStatusFlags APIs.
    - Update improved timing APIs to make it calculate improved timing according to CiA doc recommended.
      - FLEXCAN_CalculateImprovedTimingValues.
      - FLEXCAN_FDCalculateImprovedTimingValues.
    - Update FLEXCAN_SetBitRate/FLEXCAN_SetFDBitRate to added the use of enhanced timing registers.
- 2.6.2
  - Improvements
    - Add CANFD frame data length enumeration.
- 2.6.1
  - Bug Fixes
    - Fixed the issue of not fully initializing memory in FLEXCAN_Reset() API.
- 2.6.0
  - Improvements
    - Enable CANFD ISO mode in FLEXCAN_FDInit API.
    - Enable the transceiver delay compensation feature when enable FD operation and set bitrate switch.
    - Implementation memory error control in FLEXCAN_Init API.
    - Improve FLEXCAN_FDCalculateImprovedTimingValues API to get same value for FP-RESDIV and PRESDIV.
    - Added memory error configuration for user.
      - enableMemoryErrorControl
      - enableNonCorrectableErrorEnterFreeze
    - Added memory error related APIs.
      - FLEXCAN_GetMemoryErrorReportStatus
      - FLEXCAN_GetMemoryErrorStatusFlags
      - FLEXCAN_ClearMemoryErrorStatusFlags
      - FLEXCAN_EnableMemoryErrorInterrupts
      - FLEXCAN_DisableMemoryErrorInterrupts
  - Bug Fixes
    - Fixed the issue of sent duff CAN frame after call FLEXCAN_FDInit() API.
- 2.5.2

- Bug Fixes
  - Fixed the code error issue and simplified the algorithm in improved timing APIs.
    - The bit field in CTRL1 register couldn't calculate higher ideal SP, we set it as the lowest one(75%)
    - FLEXCAN_CalculateImprovedTimingValues
    - FLEXCAN_FDCalculateImprovedTimingValues
  - Fixed MISRA-C 2012 Rule 17.7 and 14.4.
- Improvements
  - Pass EsrStatus to callback function when kStatus_FLEXCAN_ErrorStatus is comming.

- 2.5.1
  - Bug Fixes
    - Fixed the non-divisible case in improved timing APIs.
      - FLEXCAN_CalculateImprovedTimingValues
      - FLEXCAN_FDCalculateImprovedTimingValues
- 2.5.0
  - Bug Fixes
    - MISRA C-2012 issue check.
      - Fixed rules, containing: rule-10.1, rule-10.3, rule-10.4, rule-10.7, rule-10.8, rule-11.8, rule-12.2, rule-13.4, rule-14.4, rule-15.5, rule-15.6, rule-15.7, rule-16.4, rule-17.3, rule-5.8, rule-8.3, rule-8.5.
    - Fixed the issue that API FLEXCAN_SetFDRxMbConfig lacks inactive message buff.
    - Fixed the issue of Pa082 warning.
    - Fixed the issue of dead lock in the function of interruption handler.
    - Fixed the issue of Legacy Rx Fifo EDMA transfer data fail in evkmimxrt1060 and evkmimxrt1064.
    - Fixed the issue of setting CANFD Bit Rate Switch.
    - Fixed the issue of operating unknown pointer risk.
      - when used the pointer "handle->mbFrameBuf[mbIdx]" to update the timestamp in a short-live TX frame, the frame pointer became as unknown, the action of operating it would result in program stack destroyed.
    - Added assert to check current CAN clock source affected by other clock gates in current device.
      - In some chips, CAN clock sources could be selected by CCM. But for some clock sources affected by other clock gates, if user insisted on using that clock source, they had to open these gates at the same time. However, they should take into consideration the power consumption issue at system level. In RT10xx chips, CAN clock source 2 was affected by the clock gate of lpuart1. ERRATA ID: (ERR050235 in CCM).
  - Improvements
    - Implementation for new FLEXCAN with ECC feature able to exit Freeze mode.
    - Optimized the function of interruption handler.
    - Added two APIs for FLEXCAN EDMA driver.
      - FLEXCAN_PrepareTransfConfiguration
      - FLEXCAN_StartTransferDatafromRxFIFO
    - Added new API for FLEXCAN driver.

          · FLEXCAN_GetTimeStamp
          · For TX non-blocking API, we wrote the frame into mailbox only, so no need to register TX frame address to the pointer, and the timestamp could be updated into the new global variable handle->timestamp[mbIdx], the FLEXCAN driver provided a new API for user to get it by handle and index number after TX DONE Success.
          · FLEXCAN_EnterFreezeMode
          · FLEXCAN_ExitFreezeMode
      * Added new configuration for user.
          · disableSelfReception
          · enableListenOnlyMode
      * Renamed the two clock source enum macros based on CLKSRC bit field value directly.
          · The CLKSRC bit value had no property about Oscillator or Peripheral type in lots of devices, it acted as two different clock input source only, but the legacy enum macros name contained such property, that misled user to select incorrect CAN clock source.
      * Created two new enum macros for the FLEXCAN driver.
          · kFLEXCAN_ClkSrc0
          · kFLEXCAN_ClkSrc1
      * Deprecated two legacy enum macros for the FLEXCAN driver.
          · kFLEXCAN_ClkSrcOsc
          · kFLEXCAN_ClkSrcPeri
      * Changed the process flow for Remote request frame response..
          · Created a new enum macro for the FLEXCAN driver.
          · kStatus_FLEXCAN_RxRemote
      * Changed the process flow for kFLEXCAN_StateRxRemote state in the interrupt handler.
          · Should the TX frame not register to the pointer of frame handle, interrupt handler would not be able to read the remote response frame from the mail box to ram, so user should read the frame by manual from mail box after a complete remote frame transfer.

- 2.4.0
  - Bug Fixes
    * MISRA C-2012 issue check.
      · Fixed rules, containing: rule-12.1, rule-17.7, rule-16.4, rule-11.9, rule-8.4, rule-14.4, rule-10.8, rule-10.4, rule-10.3, rule-10.7, rule-10.1, rule-11.6, rule-13.5, rule-11.3, rule-8.3, rule-12.2 and rule-16.1.
    * Fixed the issue that CANFD transfer data fail when bus baudrate is 30Khz.
    * Fixed the issue that ERR009595 does not folllow the ERRATA document.
    * Fixed code error for ERR006032 work around solution.
    * Fixed the Coverity issue of BAD_SHIFT in FLEXCAN.
    * Fixed the Repo build warning issue for variable without initial.
  - Improvements
    * Fixed the run fail issue of FlexCAN RemoteRequest UT Case.
    * Implementation all TX and RX transfering Timestamp used in FlexCAN demos.
    * Fixed the issue of UT Test Fail for CANFD payload size changed from 64BperMB to

8PerMB.
    * Implementation for improved timing API by baud rate.
- 2.3.2
  - Improvements
    * Implementation for ERR005959.
    * Implementation for ERR005829.
    * Implementation for ERR006032.
- 2.3.1
  - Bug Fixes
    * Added correct handle when kStatus_FLEXCAN_TxSwitchToRx is comming.
- 2.3.0
  - Improvements
    * Added self-wakeup support for STOP mode in the interrupt handling.
- 2.2.3
  - Bug Fixes
    * Fixed the issue of CANFD data phase's bit rate not set as expected.
- 2.2.2
  - Improvements
    * Added a time stamp feature and enable it in the interrupt_transfer example.
- 2.2.1
  - Improvements
    * Separated CANFD initialization API.
    * In the interrupt handling, fix the issue that the user cannot use the normal CAN API when with an FD.
- 2.2.0
  - Improvements
    * Added FSL_FEATURE_FLEXCAN_HAS_SUPPORT_ENGINE_CLK_SEL_REMO-VE feature to support SoCs without CAN Engine Clock selection in FlexCAN module.
    * Added FlexCAN Serial Clock Operation to support i.MX SoCs.
- 2.1.0
  - Bug Fixes
    * Corrected the spelling error in the function name FLEXCAN_XXX().
    * Moved Freeze Enable/Disable setting from FLEXCAN_Enter/ExitFreezeMode() to F-LEXCAN_Init().
    * Corrected wrong helper macro values.
  - Improvements
    * Hid FLEXCAN_Reset() from user.
    * Used NDEBUG macro to wrap FLEXCAN_IsMbOccupied() function instead of DEB-UG macro.
- 2.0.0
  - Initial version.

FTM

The current FTM driver version is 2.5.0.

- 2.5.0
  - Improvements
    - Added FTM_CalculateCounterClkDiv to help calculates the counter clock prescaler.
    - Modify FTM_UpdatePwmDutycycle API to make it return pwm duty cycles status.
  - Bug Fixes
    - Fixed TPM_SetupPwm can't configure 100% center align combined PWM issues.
- 2.4.1
  - Bug Fixes
    - Added function macro to determine if FTM instance has only basic features, to prevent access to protected register bits.
- 2.4.0
  - Improvements
    - Added CNTIN register initialization in FTM_SetTimerPeriod API.
    - Added a new API to read the captured value of a FTM channel configured in capture mode:
      - FTM_GetInputCaptureValue()
- 2.3.0
  - Improvements
    - Added support of EdgeAligned/CenterAligned/Asymmetrical combine PWM mode in FTM_SetupPWM() and FTM_SetupPwmMode() APIs.
    - Remove kFTM_ComplementaryPwm from support PWM mode, and add new parameter "enableComplementary" in structure ftm_chnl_pwm_signal_param_t.
    - Rename FTM_SetupFault() API to FTM_SetupFaultInput() to avoid ambiguity.
- 2.2.3
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 14.4 and 17.7.
- 2.2.2
  - Bug Fixes
    - Fixed the issue that when FTM instance has only TPM features cannot be initialized by FTM_Init() function. By added function macro to assert FTM is TPM only instance.
- 2.2.1
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.1, 10.3, 10.4, 10.6, 10.7 and 11.9.
- 2.2.0
  - Bug Fixes
    - Fixed the issue of comparison between signed and unsigned integer expressions.
  - Improvements
    - Added support of complementary mode in FTM_SetupPWM() and FTM_SetupPwm-Mode() APIs.
    - Added new parameter "enableDeadtime" in structure ftm_chnl_pwm_signal_param_t.
- 2.1.1
  - Bug Fixes

    - Fixed COVERITY integer handing issue where the right operand of a left bit shift statement should not be a negative value. This appears in FTM_SetReloadPoints().
- 2.1.0
  - Improvements
    - Added a new API FTM_SetupPwmMode() to allow the user to set the channel match value in units of timer ticks. New configure structure called ftm_chnl_pwm_config-_param_t was added to configure the channel's PWM parameters. This API is similar with FTM_SetupPwm() API, but the new API will not set the timer period(MOD value), it will be useful for users to set the PWM parameters without changing the timer period.
  - Bug Fixes
    - Added feature macro to enable/disable the external trigger source configuration.
- 2.0.4
  - Improvements
    - Added a new API to enable DMA transfer:
      - FTM_EnableDmaTransfer()
- 2.0.3
  - Bug Fixes
    - Updated the FTM driver to enable fault input after configuring polarity.
- 2.0.2
  - Improvements
    - Added support to Quad Decoder feature with new APIs:
      - FTM_GetQuadDecoderFlags()
      - FTM_SetQuadDecoderModuloValue()
      - FTM_GetQuadDecoderCounterValue()
      - FTM_ClearQuadDecoderCounterValue()
- 2.0.1
  - Bug Fixes
    - Updated the FTM driver to fix write to ELSA and ELSB bits.
    - FTM combine mode: set the COMBINE bit before writing to CnV register.
- 2.0.0
  - Initial version.

GPIO

The current driver version is 2.5.3.

- 2.5.3
  - Bug Fixes
    - Correct the feature macro typo: FSL_FEATURE_GPIO_HAS_NO_INDEP_OUTPU-T_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".

I2C

The current I2C driver version is 2.0.9.

- 2.0.9
  - Bug Fixes
    - Fixed the MISRA-2012 violations.
      - Fixed rule 8.4, 10.1, 10.4, 13.5, 20.8.
- 2.0.8
  - Bug Fixes
    - Fixed the bug that DFEN bit of I2C Status register 2 could not be set in I2C_MasterInit.
    - MISRA C-2012 issue fixed: rule 14.2, 15.7, and 16.4.
    - Eliminated IAR Pa082 warnings from I2C_MasterTransferDMA and I2C_MasterTransferCallbackDMA by assigning volatile variables to local variables and using local variables instead.
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 11.9, 14.4, 15.7, 17.7.
  - Improvements
    - Improved timeout mechanism when waiting certain state in transfer API.
    - Updated the I2C_WAIT_TIMEOUT macro to unified name I2C_RETRY_TIMES.
    - Moved the master manually acknowledge byte operation into static function I2C_MasterAckByte.
    - Fixed control/status clean flow issue inside I2C_MasterReadBlocking to avoid potential issue that pending status is cleaned before it's proceeded.
- 2.0.7
  - Bug Fixes
    - Fixed the issue for MISRA-2012 check.
      - Fixed rule 11.9 ,15.7 ,14.4 ,10.4 ,10.8 ,10.3, 10.1, 10.6, 13.5, 11.3, 13.2, 17.7, 5.7, 8.3, 8.5, 11.1, 16.1.
    - Fixed Coverity issue of unchecked return value in I2C_RTOS_Transfer.
    - Fixed variable redefine issue by moving i2cBases from fsl_i2c.h to fsl_i2c.c.
  - Improvements
    - Added I2C_MASTER_FACK_CONTROL macro to enable FACK control for master transfer receive flow with IP supporting double buffer, then master could hold the SCL by manually setting TX AK/NAK during data transfer.
- 2.0.6
  - 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 by the subaddress.
- 2.0.5
  - Improvements
    - Added I2C_WATI_TIMEOUT macro to allow the user to specify the timeout times for waiting flags in functional API and blocking transfer API.
- 2.0.4
  - Bug Fixes

    - Added a proper handle for transfer config flag kI2C_TransferNoStartFlag to support transmit with kI2C_TransferNoStartFlag flag. Support write only or write+read with no start flag; does not support read only with no start flag.
- 2.0.3
  - Bug Fixes
    - Removed enableHighDrive member in the master/slave configuration structure because the operation to HDRS bit is useless, the user need to use DSE bit in port register to configure the high drive capability.
    - Added register reset operation in I2C_MasterInit and I2C_SlaveInit APIs. Fixed issue where I2C could not switch between master and slave mode.
    - Improved slave IRQ handler to handle the corner case that stop flag and address match flag come synchronously.
- 2.0.2
  - Bug Fixes
    - Fixed issue in master receive and slave transmit mode with no stop flag. The master could not succeed to start next transfer because the master could not send out re-start signal.
    - Fixed the out-of-order issue of data transfer due to memory barrier.
    - Added hold time configuration for slave. By leaving the SCL divider and MULT reset values when configured to slave mode, the setup and hold time of the slave is then reduced outside of spec for lower baudrates. This can cause intermittent arbitration loss on the master side.
  - New Features
    - Added address nak event for master.
    - Added general call event for slave.
- 2.0.1
  - New Features
    - Added double buffer enable configuration for SoCs which have the DFEN bit in S2 register.
    - Added flexible transmit/receive buffer size support in I2C_SlaveHandleIRQ.
    - Added start flag clear, address match, and release bus operation in I2C_SlaveWrite/-ReadBlocking API.
  - Bug Fixes
    - Changed the kI2C_SlaveRepeatedStartEvent to kI2C_SlaveStartEvent.
- 2.0.0
  - Initial version.

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.

LMEM

The current LMEM driver version is 2.1.2.

- 2.1.2 -Bug Fixes
  - Fixed doxygen issue.
- 2.1.1
  - Bug Fixes
    * MISRA C-2012 issue fixed.
      · Fixed rules, containing: rule-10.3, rule-14.4, rule-15.5.
- 2.1.0
  - Improvements
    * Removed the write buffer enable from the cache enable API.
    * Added Enable write buffer APIs.
- 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 kLPUAR-T_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/e-DMA 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

PDB

The current PDB driver version is 2.0.4.

- 2.0.4
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.1 and 10.4.
- 2.0.3
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 17.7.
- 2.0.2
  - Improvement:
    - Used macros in feature file instead of that in IO map.

- 2.0.1
  - Changed PDB register base array to const.
- 2.0.0
  - Initial version.

PIT

The current PIT driver version is 2.0.4.

- 2.0.4
  - Bug Fixes
    - Fixed PIT_SetTimerPeriod implementation, the load value trigger should be PIT clock cycles minus 1.
- 2.0.3
  - Bug Fixes
    - Clear all status bits for all channels to make sure the status of all TCTRL registers is clean.
- 2.0.2
  - Bug Fixes
    - Fixed MISRA-2012 issues.
      - Rule 10.1.
- 2.0.1
  - Bug Fixes
    - Cleared timer enable bit for all channels in function PIT_Init() to make sure all channels stay in disable status before setting other configurations.
    - Fixed MISRA-2012 rules.
      - Rule 14.4, rule 10.4.
- 2.0.0
  - Initial version.

PMC

The current PMC driver version is 2.0.3.

- 2.0.3
  - Bug Fixes
    - Fixed the violation of MISRA C-2012 rule 11.3.
- 2.0.2
  - Bug Fixes
    - Fixed the violations of MISRA 2012 rules:
      - Rule 10.3.
- 2.0.1
  - Bug Fixes
    - Fixed MISRA issues.

      - Rule 10.8, Rule 10.3.
- 2.0.0
  - Initial version.

PORT

The current PORT driver version is 2.2.0.

- 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.

RCM

The current RCM driver version is 2.0.4.

- 2.0.4
  - Bug Fixes
    - Fixed violation of MISRA C-2012 rule 10.3
- 2.0.3
  - Bug Fixes
    - Fixed violation of MISRA C-2012 rules.
- 2.0.2
  - Bug Fixes
    - Fixed MISRA issue.
      - Rule 10.8, rule 10.1, rule 13.2, rule 3.1.
- 2.0.1
  - Bug Fixes
    - Fixed kRCM_SourceSw bit shift issue.
- 2.0.0

    - Initial version.

RTC

The current RTC driver version is 2.2.1.

- 2.2.1
    - Bug Fixes
        * Fixed the issue of Pa082 warning.
        * Fixed the issue of bit field mask checking.
        * Fixed the issue of hard code in RTC_Init.
- 2.2.0
    - Bug Fixes
        * Fixed MISRA C-2012 issue.
            · Fixed rule contain: rule-17.7, rule-14.4, rule-10.4, rule-10.7, rule-10.1, rule-10.3.
        * Fixed central repository code formatting issue.
    - Improvements
        * Added an API for enabling wakeup pin.
- 2.1.0
    - Improvements
        * Added feature macro check for many features.
- 2.0.0
    - Initial version.

SAI

The current SAI driver version is 2.3.4

- 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_TransferTxSetConfigSDM-A.
    - Fixed BAD BIT SHIFT OPERATION issue caused by the FSL_FEATURE_SAI_CH-ANNEL_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_CHA-NNEL_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_S-AI5_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.

SDHC

The current SDHC driver version is 2.1.13.

- 2.1.13
    - Improvements
        - Cleared SDHC status before transfer start in SDHC_TransferBlocking function.
- 2.1.12
    - Bug Fixes
        - Fixed the ADMA descriptor configuration issue when transfer size bigger than the max length of one descriptor.

- 2.1.11
  - Improvements
    - Used different status code for command and data interrupt callback.
- 2.1.10
  - Improvements
    - Disabled redundant interrupt per different transfer request.
    - Disabled interrupt and reset command/data pointer in handle when transfer completes.
  - Bug Fixes
    - Fixed PA082 build warning.
    - Fixed violations of MISRA C-2012 rule 10.1, 10.3, 10.4, 4.7, 16.4, 15.7, 14.4, 10.6, 10.8, 11.3, 17.7, 13.5, 11.9.
- 2.1.9
  - Improvements
    - Added API SDHC_GetEnabledInterruptStatusFlags and used it in SDHC_TransferHandleIRQ.
    - Removed useless member interruptFlag in sdhc_handle_t.
    - Fixed OUT-OF BOUNDS write in function SDHC_ReceiveCommandResponse.
- 2.1.8
  - Bug Fixes
    - Disabled useless interrupt while DMA is used.
    - Fixed MDK 66-D warning.
- 2.1.7
  - Bug Fixes
    - Fixed ADMA1 descriptor configuration error.
    - Improved set clock function to check the output frequency range.
- 2.1.6
  - New Features
    - Added SDHC_CardDetectByData3 API to support detect card through DATA3.
    - Added host base address/user data parameter for all call back function.
- 2.1.5
  - New Features
    - Added NON-WORD align data addr transfer support in DMA mode.
- 2.1.4
  - New Features
    - Added response error flag to check response once read from the card.
  - Bug Fixes
    - Fixed the issue of incorrect calculation of the clock divider.
- 2.1.3
  - Improvements
    - Modified some definitions to be compatible with middleware adapter.
- 2.1.2
  - Bug Fixes
    - Used function pointer for interrupt handler to reduce code size.
    - Bad status bit check behavior when waiting for initialization of an SD card.
    - Added support NON-WORD aligned data size transfer mode for SDIO card.

- 2.1.1
  - Bug Fixes
    - Fixed the compile error when ADMA1 is enabled.
- 2.1.0
  - New Features
    - Added a host descriptor to contain SDHC related attributes.
  - Bug Fixes
    - Removed clock auto gated function because of that it is a hardware issue.
  - Other Changes
    - Added more SDIO card related command type.
    - Changed the callback mechanism in the non-blocking transaction API.
    - Merged the two ADMA configuration functions to be one.
    - Changed the transaction API's name.

SDRAMC

The current SDRAMC driver version is 2.1.1.

- 2.1.1
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.3, 10.4, and 16.4.
- 2.1.0
  - API change:
    - Changed status_t SDRAMC_SendCommand() to void SDRAMC_SendCommand().
- 2.0.1
  - Miscellaneous changes:
    - Added static to the global SDRAMC variables.
- 2.0.0
  - Initial version.

SIM

The current SIM driver version is 2.1.3.

- 2.1.3
  - Improvements
    - Updated function SIM_GetUniqueId to support different register names.
- 2.1.2
  - Bug Fixes
    - Fixed SIM_GetUniqueId bug that could not get UIDH.
- 2.1.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.1, 10.4
- 2.1.0

  - Improvements
    - Added new APIs: SIM_GetRfAddr() and SIM_EnableSystickClock().
- 2.0.0
  - Initial version.

SMC

The current SMC driver version is 2.0.7.

- 2.0.7
  - Bug Fixes
    - Fixed MISRA-2012 issue 10.3.
- 2.0.6
  - Bug Fixes
    - Fixed issue for MISRA-2012 check.
      - Fixed rule 10.3, rule 11.3.
- 2.0.5
  - Bug Fixes
    - Fixed issue for MISRA-2012 check.
      - Fixed rule 15.7, rule 14.4, rule 10.3, rule 10.1, rule 10.4.
- 2.0.4
  - Bug Fixes
    - When entering stop modes, used RAM function for the flash synchronization issue. Application should make sure that, the RW data of fsl_smc.c is located in memory region which is not powered off in stop modes.
- 2.0.3
  - Improvements
    - Added APIs SMC_PreEnterStopModes, SMC_PreEnterWaitModes, SMC_PostExitWaitModes, and SMC_PostExitStopModes.
- 2.0.2
  - Bug Fixes
    - Added DSB before WFI while ISB after WFI.
  - Other Changes
    - Updated SMC_SetPowerModeVlpw implementation.
- 2.0.1
  - Other Changes
    - Updated for KL8x.
- 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.

TSI_V4

The current TSI_V4 driver version is 2.1.3.

- 2.1.3
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rules:
      - Rule 10.1, 10.3, 10.4, 10.8, 12.2, 14.4, 17.7.
- 2.1.2
  - Bug Fixes
    - Fixed w1c issues in status handling API.
    - Fixed register naming error in API "static inline void TSI_EnableEndOfScanDmaTransferOnly(TSI_Type *base, bool enable)".
    - Removed redundant status flags clear APIs when enable interrupts.
- 2.1.1
  - New Features
    - Changed void TSI_DeInit(TSI_Type *base) to void TSI_Deinit(TSI_Type *base).
- 2.0.1
  - Other Changes
    - Changed default configuration structure member order.

UART

The current UART driver version is 2.5.1.

- 2.5.1
  - Improvements
    - Use separate data for TX and RX in uart_transfer_t.
  - Bug Fixes
    - Fixed bug that when ring buffer is used, if some data is received in ring buffer first before calling UART_TransferReceiveNonBlocking, the received data count returned by UART_TransferGetReceiveCount is wrong.
- 2.5.0
  - New Features
    - Added APIs UART_GetRxFifoCount/UART_GetTxFifoCount to get rx/tx FIFO data count.
    - Added APIs UART_SetRxFifoWatermark/UART_SetTxFifoWatermark to set rx/tx FIFO water mark.
  - Bug Fixes
    - Fixed bug of race condition during UART transfer using transactional APIs, by disabling and re-enabling the global interrupt before and after critical operations on interrupt enable registers.
    - Fixed DMA/eDMA transfer blocking issue by enabling tx idle interrupt after DMA/eDMA transmission finishes.
- 2.4.0
  - New Features
    - Added APIs to configure 9-bit data mode, set slave address and send address.
- 2.3.0
  - Bug Fixes
    - Fixed the bug that, when framing/parity/noise/overflow flag or idle line detect flag is set, receive FIFO should be flushed to avoid FIFO pointer being in unknown state, since FIFO has no valid data.
  - Improvements
    - Modified UART_TransferHandleIRQ so that txState will be set to idle only when all data has been sent out to bus.
    - Modified UART_TransferGetSendCount so that this API returns the real byte count that UART has sent out rather than the software buffer status.
    - Added timeout mechanism when waiting for certain states in transfer driver.
- 2.2.0
  - New Features
    - Added UART hardware FIFO enable/disable API.
  - Improvements
    - Added check for kUART_TransmissionCompleteFlag in UART_TransferHandleIRQ, UART_SendEDMACallback and UART_TransferSendDMACallback to ensure all the data would be sent out to bus.
  - Bug Fixes
    - Eliminated IAR Pa082 warnings from UART_TransferGetRxRingBufferLength, UAR-

T_GetEnabledInterrupts, UART_GetStatusFlags and UART_TransferHandleIRQ.
    * Added code in UART_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.
    * Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 14.4, 11.6, 17.7.
- 2.1.6
  - Bug Fixes
    * Fixed the issue of register's being in repeatedly reading status while performing the IRQ routine.
- 2.1.5
  - Improvements
    * Added hardware flow control function support.
    * Added idle-line-detecting feature in UART_TransferNonBlocking function. If an idle line is detected, a callback will be triggered with status kStatus_UART_IdleLine-Detected returned. This feature may be useful when the number of received bytes is less than the expected receive data size. Before triggering the callback, data in the FIFO is read out (if it has FIFO), and no interrupt will be disabled except for the case that the receive data size reaches 0.
    * Enabled the RX FIFO watermark function. With the idle-line-detecting feature enabled, you can set the watermark value to whatever you want (should not be bigger than the RX FIFO size). Data is then received and a callback will be triggered when data receive ends.
- 2.1.4
  - Improvements
    * Changed parameter type in UART_RTOS_Init() struct rtos_uart_config –> uart_rtos_-config_t.
  - Bug Fixes
    * Disabled UART receive interrupt instead of global interrupt when reading data from ring buffer. With ring buffer used, receive nonblocking will disable global interrupt to protect the ring buffer. This has a negative effect on other IPs using interrupt.
- 2.1.3
  - New Features
    * Added RX framing error and parity error status check when using interrupt transfer.
- 2.1.2
  - Bug Fixes
    * Fixed baud rate fine adjust bug to make the computed baud rate more accurate.
- 2.1.1
  - Bug Fixes
    * Removed needless check of event flags and assert in UART_RTOS_Receive.
    * Always waited for RX event flag in UART_RTOS_Receive.
- 2.1.0
  - Improvements
    * Added transactional API.
- 2.0.0
  - Initial version.

UART_EDMA

The current UART_EDMA driver version is 2.5.2.

- 2.5.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules.
- 2.5.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules.
- 2.5.0
  - Refer UART driver change log 2.1.0 to 2.5.0

UART_FREERTOS

The current UART_FREERTOS driver version is 2.5.0.

- 2.5.0
  - Refer UART driver change log 2.1.0 to 2.5.0

VREF

The current VREF driver version is 2.1.2.

- 2.1.2
  - Bug Fixes
    - Fixed the violation of MISRA-2012 rule 10.3.
    - Fixed MISRA C-2012 rule 10.3, rule 10.4 violation.
- 2.1.1
  - Bug Fixes
    - MISRA-2012 issue fixed.
      - Fixed rules containing: rule-10.4, rule-10.3, rule-10.1.
- 2.1.0
  - Improvements
    - Added new functions to support L5K board: added VREF_SetTrim2V1Val() and VREF_GetTrim2V1Val() functions to supply 2V1 output mode.
- 2.0.0
  - Initial version.

WDOG

The current WDOG driver version is 2.0.1.

- 2.0.1
  - Bug Fixes

        * MISRA C-2012 issue fixed: rule 10.3, 10.4, 10.6, 10.7 11.9 and 17.7.
- 2.0.0
    - Initial version.

2 Middleware Change Log

emWin library

The currently supported version is 6.16c

- 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

FatFs for MCUXpresso SDK

Current version is FatFs R0.14b_rev0.

- 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 3.0.4. 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

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 development version 2.2.0.dev 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_rev5
  - New features:
    - Ported lwIP 2.2.0.dev (2021-05-11, branch: master, SHA-1: 7ec4e9be304e7f8953740f10b2c810a2
      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_N-UM 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: 0056522cc974d2be2005c324f37187b5
      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: c385f31076b27efb8ee37f00cb556878

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: 555812dcec38c9a2ef1ef9b31816291
      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: 1bb6e7f52de1cd86be0eed31
      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.

Multicore SDK

The current version of Multicore SDK is 2.11.0.

- 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

Host SDHC driver for MCUXpresso SDK

The current driver version is 2.4.0.

- 2.4.0
  - Improvements
    - Removed deprecated api in SDHC host driver.
    - Added SDMMCHOST_ConvertDataToLittleEndian api.
    - Added capability/maxBlockCount/maxBlockSize in host decriptior.
    - Added mutual exclusive access for function init/deinit/transfer function.
- 2.3.2
  - Improvements
    - Enabled DAT3 card detect interrupt in function SDMMCHOST_PollingCardDetectStatus to support DAT3 re-detect card.
- 2.3.1
  - Improvements
    - Enabled DAT3 card detect feature.
    - Added host instance capability macro.
    - Added clear card inserted/removed event when card removed/inserted interrupt generated.
  - 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.
- 2.0.0
  - 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_SetCard-Power.
    - 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 2.4.0.

- 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 SD-R50/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_WaitCardDetectStatus 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 2.4.0.

- 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_Wait-CardDetectStatus as deprecated.
    - Added new api SDIO_SetCardPower/SDIO_PollingCardDetectStatus/SDIO_HostDo-Reset.
    - 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_WA-RNING() 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.

SD(SPI) Card driver for MCUXpresso SDK

The current driver version is 2.2.1.

- 2.2.1
  - 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.
- 2.2.0

  - Improvements
    - Added init/deinit/csActivePolarity function pointer in sdspi_host_t.
    - Added retry function in SDSPI_SetBlockSize to fix the issue that set block size command failed found on low capability card.
    - Added internalBuffer in sdspi_card_t instead if use rawcid/rawcsd/rawscr to reduce the memoery cost of card descriptor.
- 2.1.4
  - Bug Fixes
    - Fixed MDK 66-D warning.
- 2.1.3
  - Improvements
    - Improved sdspi code size and performance.
- 2.0.0
  - Initial version.

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, LP-C54xxx 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

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.

WM8904

The current wm8904 driver version is 2.5.0.

- 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 2.2.0.

- 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.

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