MCUXpresso SDK Release Notes Supporting evkmimxrt1024

Change Logs

NXP Semiconductors

Contents

Driver Change Log

Title Page No.

Title Page No.

Middleware Change Log

Component Change Log

1 Driver Change Log

CLOCK

The current CLOCK driver version is 2.5.1.

- 2.5.1
  - Improvements
    - Added enumeration clock_div_value_t.
- 2.5.0
  - New features
    - Added CLOCK_IsUsb1PfdEnabled and CLOCK_IsSysPfdEnabled to get the clock source status.
- 2.4.1
  - Improvements
    - Placed function internal constants into initialized data segment.
- 2.4.0
  - New Features
    - Added clock output related APIs and data structures.
    - Added one function CLOCK_GetClockRootFreq to get the frequency of each clock root.
- 2.3.2
  - Bug Fixes
    - Fixed MISRA C-2012 rule 20.7 rule 8.6 rule 10.3 and rule 10.1.
    - Fixed IAR warning Pa082 for the clock driver.
- 2.3.1
  - Improvements
    - Added new macro in case that the CCM has CAN clock affect issue.
- 2.3.0
  - New feature:
    - Moved SDK_DelayAtLeastUs function from clock driver to common driver.
- 2.2.0
  - New feature
    - Adding new API CLOCK_DelayAtLeastUs() implemented by DWT to allow users set delay in unit of microsecond.
- 2.1.6
  - Bug Fix:
    - Fix build issue with GCC compiler when include header from C++ file.
- 2.1.5
  - Bug Fix:
    - Add initialization of the fractional mode and spread spectrum mode in CLOCK_InitSysPll().
- 2.1.4
  - Optimization:
    - Add PerClk in clock_name_t and CLOCK_GetFreq.

        * Add APIs to get the frequency of AHB clock and SEMC, IPG clock and PER clock.
- 2.1.3
  - Use double instead of uint64_t to achieve better performance with double precision FPU.
- 2.1.2
  - some minor fixes.
- 2.0.0
  - initial version.

IOMUXC

The current IOMUXC driver version is 2.0.1.

- 2.0.1
  - Doxygen improvement.
- 2.0.0
  - 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.

LPI2C_CMSIS

Current LPI2C_CMSIS driver version is 2.1

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

LPSPI_CMSIS

Current LPSPI_CMSIS driver version is 2.5

- 2.5
  - Bug Fixes
    - Fixed wrong configuration of setting the bytes to be swapped during transfer when the transfer width is more than 8.
- 2.4
  - Bug Fixes
    - Update driver to fix warnings reported by IAR v9.
- 2.3
  - Bug Fixes
    - Fixed the MISRA-2012 violations.
      - Fixed rule 8.4, 8.6, 10.1, 10.3, 10.4, 11.1, 11.8, 14.4, 16.1, 16.3, 17.7, 17.3, 17.7, 20.9.
- 2.2
  - Bug Fixes
    - Fixed the bug that, the parameter num of APIs ARM_SPI_Transfer, ARM_SPI_Send and ARM_SPI_Receive, and the return value of API ARM_SPI_GetDataCount should be the number of data item defined by datawidth, rather than the number of byte.
- 2.1
  - Bug Fixes
    - Fixed the incorrect clock polarity assignment in the driver. For ARM_SPI_CPOL0_CPHA0 and other frame format parameters, CPOL = 0 means kSPI_ClockPolarityActiveHigh not kSPI_ClockPolarityActiveLow in driver.
  - New features
    - Allowed user to set up the default transmit value by using ARM_SPI_SET_DEFAULT_TX_VALUE. Please note that this is not supported in slave interrupts, because the pin will stay tristated if tX buffer is NULL.
    - Enabled slave select mode. Note this has no effect when user sets any of them because the driver can only support the hardware control function.
    - Enabled 3-Wire mode, user can use ARM_SPI_MODE_MASTER_SIMPLEX/ARM_SPI_MODE_SLAVE_SIMPLEX to enable this feature. For ARM_SPI_MODE_MASTER_SIMPLEX mode, the SOUT pin is selected as the input/output pin, and for ARM_SPI_MODE_SLAVE_SIMPLEX, the SIN pin is selected as the input/output pin.
- 2.0
  - Initial version.

ADC_ETC

The current ADC_ETC driver version is 2.2.1.

- 2.2.1
  - Improvements
    - Moditied macro "ADC_ETC_DONE2_ERR_IRQ_TRIG0_DONE2_MASK" to "ADC_-

ETC_DONE2_3_ERR_IRQ_TRIG0_DONE2_MASK" based on the updates of header file.

- 2.2.0
  - Improvements
    - Defined two macros to support some devices that do not equipped with TSC trigger.
- 2.1.1
  - Bug Fixes
    - Fixed the violation of MISRA-2012 rule.
- 2.1.0
  - New Features
    - Supported independent IRQ enable bit in ADC-ETC chain configuration registers.
    - Supported trigger n DONE3 interrupt operations.
  - Bug Fixes
    - Fixed the violation of MISRA-2012 rules:
      - Rule 10.1 10.3 10.7 15.5 16.1 16.3 16.4 17.7
- 2.0.1
  - New Features
    - Added a control macro to enable/disable the CLOCK code in current driver.
- 2.0.0
  - Initial version.

AIPSTZ

The current AIPSTZ driver version is 2.0.1.

- 2.0.1
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.3, 10.4, and 14.4.
- 2.0.0
  - Initial version.

AOI

The current AOI driver version is 2.0.1.

- 2.0.1
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.8, 2.2.
- 2.0.0
  - Initial version.

BEE

The current BEE driver version is 2.0.2.

- 2.0.2
  - Bug Fixes
    - Fix MISRA issue.
- 2.0.1
  - Bug Fixes
    - Fixed bug in key user key loading sequence. BEE must be enabled during loading of user key.
    - Fixed typos in comments.
  - New Features
    - Added configuration setting for endian swap, access permission and region security level.
  - Improvements
    - Setting of AES nonce was moved from BEE_SetRegionKey() into separate BEE_SetRegionNonce() function.
    - Changed handling of region settings. Both regions are configured simultaneously by BEE_SetConfig() function. Configuration of FAC start and end address using IOMUXC_GPRs was moved to application.
    - Default value for region address offset was changed to 0.
- 2.0.0
  - Initial version.

CACHE

The current CACHE driver version is 2.0.4.

- 2.0.4
  - Bug Fixes
    - Fixed doxygen issue.
- 2.0.3
  - Improvements
    - Deleted redundancy code about calculating cache clean/invalidate size and address aligns.
- 2.0.2
  - Bug Fixes
    - Fixed violation of MISRA C-2012 Rule 10.1, 10.3 and 10.4.
- 2.0.1
  - Bug Fixes
    - Fixed cache size issue in L2CACHE_GetDefaultConfig API.
- 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.

COMMON

The current COMMON driver version is 2.4.0.

- 2.4.0
  - New Features
    - Added EnableIRQWithPriority, IRQ_SetPriority, and IRQ_ClearPendingIRQ for ARM.
    - Added MSDK_EnableCpuCycleCounter, MSDK_GetCpuCycleCount for ARM.
- 2.3.3
  - New Features
    - Added NETC into status group.
- 2.3.2
  - Improvements
    - Make driver aarch64 compatible
- 2.3.1
  - Bug Fixes
    - Fixed MAKE_VERSION overflow on 16-bit platforms.
- 2.3.0
  - Improvements
    - Split the driver to common part and CPU architecture related part.
- 2.2.10
  - Bug Fixes
    - Fixed the ATOMIC macros build error in cpp files.
- 2.2.9
  - Bug Fixes
    - Fixed MISRA C-2012 issue, 5.6, 5.8, 8.4, 8.5, 8.6, 10.1, 10.4, 17.7, 21.3.
    - Fixed SDK_Malloc issue that not allocate memory with required size.
- 2.2.8
  - Improvements
    - Included stddef.h header file for MDK tool chain.

  - New Features:
    - Added atomic modification macros.
- 2.2.7
  - Other Change
    - Added MECC status group definition.
- 2.2.6
  - Other Change
    - Added more status group definition.
  - Bug Fixes
    - Undef __VECTOR_TABLE to avoid duplicate definition in cmsis_clang.h
- 2.2.5
  - Bug Fixes
    - Fixed MISRA C-2012 rule-15.5.
- 2.2.4
  - Bug Fixes
    - Fixed MISRA C-2012 rule-10.4.
- 2.2.3
  - New Features
    - Provided better accuracy of SDK_DelayAtLeastUs with DWT, use macro SDK_DELAY_USE_DWT to enable this feature.
    - Modified the Cortex-M7 delay count divisor based on latest tests on RT series boards, this setting lets result be closer to actual delay time.
- 2.2.2
  - New Features
    - Added include RTE_Components.h for CMSIS pack RTE.
- 2.2.1
  - Bug Fixes
    - Fixed violation of MISRA C-2012 Rule 3.1, 10.1, 10.3, 10.4, 11.6, 11.9.
- 2.2.0
  - New Features
    - Moved SDK_DelayAtLeastUs function from clock driver to common driver.
- 2.1.4
  - New Features
    - Added OTFAD into status group.
- 2.1.3
  - Bug Fixes
    - MISRA C-2012 issue fixed.
      - Fixed the rule: rule-10.3.
- 2.1.2
  - Improvements
    - Add SUPPRESS_FALL_THROUGH_WARNING() macro for the usage of suppressing fallthrough warning.
- 2.1.1
  - Bug Fixes
    - Deleted and optimized repeated macro.

- 2.1.0
  - New Features
    - Added IRQ operation for XCC toolchain.
    - Added group IDs for newly supported drivers.
- 2.0.2
  - Bug Fixes
    - MISRA C-2012 issue fixed.
      - Fixed the rule: rule-10.4.
- 2.0.1
  - Improvements
    - Removed the implementation of LPC8XX Enable/DisableDeepSleepIRQ() function.
    - Added new feature macro switch "FSL_FEATURE_HAS_NO_NONCACHEABLE_SECTION" for specific SoCs which have no noncacheable sections, that helps avoid an unnecessary complex in link file and the startup file.
    - Updated the align(x) to **attribute**(aligned(x)) to support MDK v6 armclang compiler.
- 2.0.0
  - Initial version.

DCDC

The current DCDC driver version is 2.3.0.

- 2.3.0
  - Improvements
    - REG3[MISC_DELAY_TIMING], REG2[LOOPCTRL_DC_R], and REG2[LOOPCTRL_DC_C] are reserved in the latest RM, deleted corresponding functions.
- 2.2.1
  - Improvements
    - Fixed the doxygen warning.
- 2.2.0
  - New Features
    - Added supports for i.MXRT1170 series.
  - Bug Fixes
    - Fixed the warning that the DCDC_ConvertByteArrayToWord function defined but not used.
  - Improvements
    - Updated rcscale to reduce the ripple when booting into DCM.
- 2.1.0
  - Improvements
    - Divided the DCDC_AdjustTargetVoltage() into two APIs for two different modes.
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rules:
      - Rule 10.1, 10.4, 16.4, 17.7.
- 2.0.0

  - Initial version.

DMAMUX

The current DMAMUX driver version is 2.0.5.

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

EDMA

The current eDMA driver version is 2.4.4.

- 2.4.4
  - Bug Fixes
    - Fixed comments by replacing STCD with TCD
    - Fixed the TCD overwrite issue when submit transfer request in the callback if there is a active TCD in hardware.
    - Fixed violations of MISRA C-2012 rule 10.8,5.6.
- 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 2.6.3.

- 2.6.3
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 11.6.
- 2.6.2
  - Improvements
    - Changed ENET1_MAC0_Rx_Tx_Done0_DriverIRQHandler/ENET1_MAC0_Rx_Tx_Done1_DriverIRQHandler to ENET1_MAC0_Rx_Tx_Done1_DriverIRQHandler/ENET1_MAC0_Rx_Tx_Done2_DriverIRQHandler which represent ring 1 and ring 2.
- 2.6.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.3, 10.4, 10.7, 11.6, 11.8.
- 2.6.0
  - Improvements
    - Added MDIO access wrapper APIs for ease of use.
    - Fixed the build warning introduced by 64-bit compatibility patch.
- 2.5.4
  - Improvements
    - Made the driver compatible with 64-bit platforms.

- 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.

FLEXCAN

The current FLEXCAN driver version is 2.9.2.

- 2.9.2
  - Bug Fixes
    * Fixed the issue that FLEXCAN_CheckUnhandleInterruptEvents() can't detecting the exist enhanced RX FIFO interrupt status.
    * Fixed the issue that FLEXCAN_ReadPNWakeUpMB() does not return fail even no existing valid wake-up frame.
    * Fixed the issue that FLEXCAN_ReadEnhancedRxFifo() may clear bits other than the data available bit.
    * Fixed violations of the MISRA C-2012 rules 10.4, 10.8.
  - Improvements
    * Return kStatus_FLEXCAN_RxFifoDisabled instead of kStatus_Fail when read FIFO fail during IRQ handler.
    * Remove unreachable code from timing calculates APIs.
    * Update Enhanced Rx FIFO handler to make it deal with underflow/overflow status first.
- 2.9.1
  - Bug Fixes
    * Fixed the issue that FLEXCAN_TransferReceiveEnhancedFifoBlocking() API clearing Fifo data available flag more than once.
    * Fixed the issue that entering FLEXCAN_SubHandlerForEhancedRxFifo() even if Enhanced Rx fifo interrupts are not enabled.
    * Fixed the issue that FLEXCAN_TransferReceiveEnhancedFifoEDMA() update handle even if previous Rx FIFO receive not finished.
    * Fixed the issue that FLEXCAN_SetEnhancedRxFifoConfig() not configure the ERFCR[NFE] bits to the correct value.
    * Fixed the issue that FLEXCAN_ReceiveFifoEDMACallback() can't differentiate between Rx fifo and enhanced rx fifo.
    * Fixed the issue that FLEXCAN_TransferHandleIRQ() can't report Legacy Rx FIFO warning status.
- 2.9.0
  - Improvements

    - Add public set bit rate API to make driver easier to use.
    - Update Legacy Rx FIFO transfer APIs to make it support received multiple frames during one API call.
    - Optimized FLEXCAN_SubHandlerForDataTransfered() API in interrupt handling to reduce the probability of packet loss.
- 2.8.7
  - Improvements
    - Initialized the EDMA configuration structure in the FLEXCAN EDMA driver.
- 2.8.6
  - Bug Fixes
    - Fix Coverity overrun issues in fsl_flexcan_edma driver.
- 2.8.5
  - Improvements
    - Make driver aarch64 compatible.
- 2.8.4
  - Bug Fixes
    - Fixed FlexCan_Errata_6032 to disable all interrupts.
- 2.8.3
  - Bug Fixes
    - Fixed an issue with the FLEXCAN_EnableInterrupts and FLEXCAN_DisableInterrupts interrupt enable bits in the CTRL1 register.
- 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 FLEXCAN_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 FPRESDIV 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_REMOVE 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 FLEXCAN_Init().
    - Corrected wrong helper macro values.
  - Improvements
    - Hid FLEXCAN_Reset() from user.
    - Used NDEBUG macro to wrap FLEXCAN_IsMbOccupied() function instead of DEBUG macro.
- 2.0.0
  - Initial version.

FLEXIO

The current FLEXIO driver version is 2.1.0.

- 2.1.0

  - Improvements
    - Added API FLEXIO_SetClockMode to set flexio channel counter and source clock.
- 2.0.4
  - Bug Fixes
    - Fixed MISRA 8.4 issues.
- 2.0.3
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
- 2.0.2
  - Improvements
    - Split FLEXIO component which combines all flexio/flexio_uart/flexio_i2c/flexio_i2s drivers into several components: FlexIO component, flexio_uart component, flexio_i2c_master component, and flexio_i2s component.
  - Bug Fixes
    - Fixed MISRA issues
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
- 2.0.1
  - Bug Fixes
    - Fixed the dozen mode configuration error in FLEXIO_Init API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.

FLEXIO_UART

The current FLEXIO_UART driver version is 2.4.0.

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

  - Bug Fixes
    - Fixed MISRA 10.4 issues.
- 2.1.6
  - Bug Fixes
    - Fixed IAR Pa082 warnings.
    - Fixed MISRA issues
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
- 2.1.5
  - Improvements
    - Triggered user callback after all the data in ringbuffer were received in FLEXIO_UART_TransferReceiveNonBlocking.
- 2.1.4
  - Improvements
    - Unified component full name to FLEXIO UART(DMA/EDMA) Driver.
- 2.1.3
  - Bug Fixes
    - The following modifications support FLEXIO using multiple instances:
      - Removed FLEXIO_Reset API in module Init APIs.
      - Updated module Deinit APIs to reset the shifter/timer configuration instead of disabling module and clock.
      - Updated module Enable APIs to only support enable operation.
- 2.1.2
  - Bug Fixes
    - Fixed the transfer count calculation issue in FLEXIO_UART_TransferGetReceiveCount, FLEXIO_UART_TransferGetSendCount, FLEXIO_UART_TransferGetReceiveCountDMA, FLEXIO_UART_TransferGetSendCountDMA, FLEXIO_UART_TransferGetReceiveCountEDMA and FLEXIO_UART_TransferGetSendCountEDMA.
    - Fixed the Dozen mode configuration error in FLEXIO_UART_Init API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
    - Added code to report errors if the user sets a too-low-baudrate which FLEXIO cannot reach.
    - Disabled FLEXIO_UART receive interrupt instead of all NVICs when reading data from ring buffer. If ring buffer is used, receive nonblocking will disable all NVIC interrupts to protect the ring buffer. This had negative effects on other IPs using interrupt.
- 2.1.1
  - Bug Fixes
    - Changed the API name FLEXIO_UART_StopRingBuffer to FLEXIO_UART_TransferStopRingBuffer to align with the definition in C file.
- 2.1.0
  - New Features
    - Added Transfer prefix in transactional APIs.
    - Added txSize/rxSize in handle structure to record the transfer size.

  - Bug Fixes
    * Added an error handle to handle the situation that data count is zero or data buffer is NULL.

FLEXIO_UART_EDMA

The current FLEXIO_UART_EDMA driver version is 2.3.1.

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

FLEXIO_I2C

The current FLEXIO_I2C driver version is 2.5.0.

- 2.5.0
  - Improvements
    * Split some functions, fixed CCM problem in file fsl_flexio_i2c_master.c.
- 2.4.0
  - Improvements
    * Added delay of 1 clock cycle in FLEXIO_I2C_MasterTransferRunStateMachine to ensure that bus would be idle before next transfer if master is nacked.
    * Fixed issue that the restart setup time is less than the time in I2C spec by adding delay of 1 clock cycle before restart signal.
- 2.3.0
  - Improvements
    * Used 3 timers instead of 2 to support transfer which is more than 14 bytes in single transfer.
    * Improved FLEXIO_I2C_MasterTransferGetCount so that the API can check whether the transfer is still in progress.
  - Bug Fixes
    * Fixed MISRA 10.4 issues.
- 2.2.0
  - New Features
    * Added timeout mechanism when waiting certain state in transfer API.
    * Added an API for checking bus pin status.
  - Bug Fixes
    * Fixed COVERITY issue of useless call in FLEXIO_I2C_MasterTransferRunStateMachine.
    * Fixed MISRA issues
      · Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.

    - Added codes in FLEXIO_I2C_MasterTransferCreateHandle to clear pending NVIC IRQ, disable internal IRQs before enabling NVIC IRQ.
    - Modified code so that during master's nonblocking transfer the start and slave address are sent after interrupts being enabled, in order to avoid potential issue of sending the start and slave address twice.
- 2.1.7
  - Bug Fixes
    - Fixed the issue that FLEXIO_I2C_MasterTransferBlocking did not wait for STOP bit sent.
    - Fixed COVERITY issue of useless call in FLEXIO_I2C_MasterTransferRunStateMachine.
    - Fixed the issue that I2C master did not check whether bus was busy before transfer.
- 2.1.6
  - Bug Fixes
    - Fixed the issue that I2C Master transfer APIs(blocking/non-blocking) did not support the situation of master transfer with subaddress and transfer data size being zero, which means no data followed the subaddress.
- 2.1.5
  - Improvements
    - Unified component full name to FLEXIO I2C Driver.
- 2.1.4
  - Bug Fixes
    - The following modifications support FlexIO using multiple instances:
      - Removed FLEXIO_Reset API in module Init APIs.
      - Updated module Deinit APIs to reset the shifter/timer config instead of disabling module/clock.
      - Updated module Enable APIs to only support enable operation.
- 2.1.3
  - Improvements
    - Changed the prototype of FLEXIO_I2C_MasterInit to return kStatus_Success if initialized successfully or to return kStatus_InvalidArgument if "(srcClock_Hz / masterConfig->baudRate_Bps) / 2 - 1" exceeds 0xFFU.
- 2.1.2
  - Bug Fixes
    - Fixed the FLEXIO I2C issue where the master could not receive data from I2C slave in high baudrate.
    - Fixed the FLEXIO I2C issue where the master could not receive NAK when master sent non-existent addr.
    - Fixed the FLEXIO I2C issue where the master could not get transfer count successfully.
    - Fixed the FLEXIO I2C issue where the master could not receive data successfully when sending data first.
    - Fixed the Dozen mode configuration error in FLEXIO_I2C_MasterInit API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
    - Fixed the issue that FLEXIO_I2C_MasterTransferBlocking API called FLEXIO_I2-

C_MasterTransferCreateHandle, which lead to the s_flexioHandle/s_flexioIsr/s_flexioType variable being written. Then, if calling FLEXIO_I2C_MasterTransferBlocking API multiple times, the s_flexioHandle/s_flexioIsr/s_flexioType variable would not be written any more due to it being out of range. This lead to the following situation: NonBlocking transfer APIs could not work due to the fail of register IRQ.

- 2.1.1
  - Bug Fixes
    - Implemented the FLEXIO_I2C_MasterTransferBlocking API which is defined in header file but has no implementation in the C file.
- 2.1.0
  - New Features
    - Added Transfer prefix in transactional APIs.
    - Added transferSize in handle structure to record the transfer size.

FLEXIO_SPI

The current FLEXIO_SPI driver version is 2.3.0.

- 2.3.0
  - New Features
    - Supported FLEXIO_SPI slave transfer with continuous master CS signal and CPHA=0.
    - Supported FLEXIO_SPI master transfer with continuous CS signal.
    - Support 32 bit transfer width.
  - Bug Fixes
    - Fixed wrong timer compare configuration for dma/edma transfer.
    - Fixed wrong byte order of rx data if transfer width is 16 bit, since the we use shifter buffer bit swapped/byte swapped register to read in received data, so the high byte should be read from the high bits of the register when MSB.
- 2.2.1
  - Bug Fixes
    - Fixed bug in FLEXIO_SPI_MasterTransferAbortEDMA that when aborting EDMA transfer EDMA_AbortTransfer should be used rather than EDMA_StopTransfer.
- 2.2.0
  - Improvements
    - Added timeout mechanism when waiting certain states in transfer driver.
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
    - Added codes in FLEXIO_SPI_MasterTransferCreateHandle and FLEXIO_SPI_SlaveTransferCreateHandle to clear pending NVIC IRQ before enabling NVIC IRQ, to fix issue of pending IRQ interfering the on-going process.
- 2.1.3
  - Improvements
    - Unified component full name to FLEXIO SPI(DMA/EDMA) Driver.
  - Bug Fixes

        - Fixed MISRA issues
            - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
- 2.1.2
    - Bug Fixes
        - The following modification support FlexIO using multiple instances:
            - Removed FLEXIO_Reset API in module Init APIs.
            - Updated module Deinit APIs to reset the shifter/timer config instead of disabling module/clock.
            - Updated module Enable APIs to only support enable operation.
- 2.1.1
    - Bug Fixes
        - Fixed bug where FLEXIO SPI transfer data is in 16 bit per frame mode with eDMA.
        - Fixed bug when FLEXIO SPI works in eDMA and interrupt mode with 16-bit per frame and Lsbfirst.
        - Fixed the Dozen mode configuration error in FLEXIO_SPI_MasterInit/FLEXIO_SPI_SlaveInit API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
    - Improvements
        - Added #ifndef/#endif to allow users to change the default TX value at compile time.
- 2.1.0
    - New Features
        - Added Transfer prefix in transactional APIs.
        - Added transferSize in handle structure to record the transfer size.
    - Bug Fixes
        - Fixed the error register address return for 16-bit data write in FLEXIO_SPI_GetTxDataRegisterAddress.
        - Provided independent IRQHandler/transfer APIs for Master and slave to fix the baudrate limit issue.

FLEXIO_I2S

The current FLEXIO_I2S driver version is 2.2.0.

- 2.2.0
    - New Features
        - Added timeout mechanism when waiting certain state in transfer API.
    - Bug Fixes
        - Fixed IAR Pa082 warnings.
        - Fixed violations of the MISRA C-2012 rules 10.4, 14.4, 11.8, 11.9, 10.1, 17.7, 11.6, 10.3, 10.7.
- 2.1.6
    - Bug Fixes
        - Added reset flexio before flexio i2s init to make sure flexio status is normal.
- 2.1.5

  - Bug Fixes
    - Fixed the issue that I2S driver used hard code for bitwidth setting.
- 2.1.4
  - Improvements
    - Unified component's full name to FLEXIO I2S (DMA/EDMA) driver.
- 2.1.3
  - Bug Fixes
    - The following modifications support FLEXIO using multiple instances:
      - Removed FLEXIO_Reset API in module Init APIs.
      - Updated module Deinit APIs to reset the shifter/timer config instead of disabling module/clock.
      - Updated module Enable APIs to only support enable operation.
- 2.1.2
  - New Features
    - Added configure items for all pin polarity and data valid polarity.
    - Added default configure for pin polarity and data valid polarity.
- 2.1.1
  - Bug Fixes
    - Fixed FlexIO I2S RX data read error and eDMA address error.
    - Fixed FlexIO I2S slave timer compare setting error.
- 2.1.0
  - New Features
    - Added Transfer prefix in transactional APIs.
    - Added transferSize in handle structure to record the transfer size.

FLEXIO_MCU_LCD

The current FLEXIO_MCU_LCD driver version is 2.0.7.

- 2.0.7
  - Bug Fixes
    - Fixed bug that FLEXIO_MCULCD_Init return kStatus_Success even with invalid parameter.
- 2.0.6
  - Bug Fixes
    - Fixed MISRA 10.4 issues when FLEXIO_MCULCD_DATA_BUS_WIDTH defined as signed value.
- 2.0.5
  - Improvements
    - Changed FLEXIO_MCULCD_WriteDataArrayBlocking's data parameter to const type.
- 2.0.4
  - Bug Fixes
    - Fixed MISRA 10.4 issues.

- 2.0.3
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.1, 10.3, 10.4, 10.6, 14.4, 17.7.
- 2.0.2
  - Improvements
    - Unified component full name to FLEXIO_MCU_LCD (EDMA) driver.
- 2.0.1
  - Bug Fixes
    - The following modification to support FlexIO using multiple instances:
      - Removed FLEXIO_Reset API in module Init APIs.
      - Updated module Deinit APIs to reset the shifter/timer configuration instead of disabling module and clock.
      - Updated module Enable APIs to only support enable operation.
- 2.0.0
  - Initial version.

FLEXIO_CAMERA

The current FLEXIO_CAMERA driver version is 2.1.3.

- 2.1.3
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
- 2.1.2
  - Improvements
    - Unified component full name to FLEXIO CAMERA (EDMA) driver.
  - Bug Fixes
    - Fixed MISRA issues
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
- 2.1.1
  - Bug Fixes
    - The following modifications support FlexIO using multiple instances:
      - Removed FLEXIO_Reset API in module Init APIs.
      - Updated module Deinit APIs to reset the shifter/timer configuration instead of disabling module and clock.
      - Updated module Enable APIs to only support enable operation.
- 2.1.0
  - New Features
    - Added Transfer prefix in transactional APIs.

FLEXRAM

The current FLEXRAM driver version is 2.2.0.

- 2.2.0
  - New Features
    - Supported flexram ECC error injection function.
- 2.1.0
  - New Features
    - Supported flexram ECC function.
- 2.0.7
  - Bug Fixes
    - Fixed doxygen issue.
- 2.0.6
  - New Features
    - Updated bank configuration and TCM size with GPR16/GPR17/GPR18 into SOC level for different SOC.
- 2.0.5
  - New Features
    - Added the magic address feature for OCRAM, DTCM and ITCM.
- 2.0.4
  - Bug Fixes
    - Fixed FlexRAM driver's missing extern C around functions in header file.
    - Removed magic address feature from driver.
- 2.0.3
  - Bug Fixes
    - Fixed the issue that TCM size configuration was wrong when TCM bank number was not a value power of 2.
- 2.0.2
  - Bug Fixes
    - Updated driver due to Reference Manual update.
- 2.0.1
  - Bug Fixes
    - Fixed MISRA issue.
- 2.0.0
  - Initial version.

FLEXSPI

The current FLEXSPI driver version is 2.5.0.

- 2.5.0
  - Improvements
    - Supported word un-aligned access for write/read blocking/non-blocking API functions.
    - Fixed dead loop issue in DLL update function when using FRO clock source.
    - Fixed violations of the MISRA C-2012 Rule 10.3.
- 2.4.0
  - Improvements

    - Isolated IP command parallel mode and AHB command parallel mode using feature MACRO.
    - Supported new column address shift feature for external memory.
- 2.3.5
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 14.2.
- 2.3.4
  - Bug Fixes
    - Updated flexspi_config_t structure and FlexSPI_Init to support new feature FSL_FEATURE_FLEXSPI_HAS_NO_MCR0_CONBINATION.
- 2.3.3
  - Bug Fixes
    - Removed feature FSL_FEATURE_FLEXSPI_DQS_DELAY_PS for DLL delay setting. Changed to use feature FSL_FEATURE_FLEXSPI_DQS_DELAY_MIN to set slave delay target as 0 for DLL enable and clock frequency higher than 100MHz.
- 2.3.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 8.4, 8.5, 10.1, 10.3, 10.4, 11.6 and 14.4.
- 2.3.1
  - Bug Fixes
    - Wait for bus to be idle before using it as access to external flash with new setting in FLEXSPI_SetFlashConfig() API.
    - Fixed the potential buffer overread and Tx FIFO overwrite issue in FLEXSPI_WriteBlocking.
- 2.3.0
  - New Features
    - Added new API FLEXSPI_UpdateDllValue for users to update DLL value after updating flexspi root clock.
    - Corrected grammatical issues for comments.
    - Added support for new feature FSL_FEATURE_FLEXSPI_DQS_DELAY_PS in DLL configuration.
- 2.2.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 10.1, 10.3 and 10.4.
    - Updated _flexspi_command from named enumerator into anonymous enumerator.
- 2.2.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 10.1, 10.3, 10.4, 10.8, 11.9, 14.4, 15.7, 16.4, 17.7, 7.3.
    - Fixed IAR build warning Pe167.
    - Fixed the potential buffer overwrite and Rx FIFO overread issue in FLEXSPI_ReadBlocking.
- 2.2.0
  - Bug Fixes
    - Fixed flag name typos: kFLEXSPI_IpTxFifoWatermarkEmpltyFlag to kFLEXSPI_Ip-

TxFifoWatermarkEmptyFlag; kFLEXSPI_IpCommandExcutionDoneFlag to kFLEXSPI_IpCommandExecutionDoneFlag.
    - Fixed comments typos such as sequencen->sequence, levle->level.
    - Fixed FLSHCR2[ARDSEQID] field clean issue.
    - Updated flexspi_config_t structure and FlexSPI_Init to support new feature FSL_FEATURE_FLEXSPI_HAS_NO_MCR0_ATDFEN and FSL_FEATURE_FLEXSPI_HAS_NO_MCR0_ARDFEN.
    - Updated flexspi_flags_t structure to support new feature FSL_FEATURE_FLEXSPI_HAS_INTEN_AHBBUSERROREN.
- 2.1.1
  - Improvements
    - Defaulted enable prefetch for AHB RX buffer configuration in FLEXSPI_GetDefaultConfig, which is align with the reset value in AHBRXBUFxCR0.
    - Added software workaround for ERR011377 in FLEXSPI_SetFlashConfig; added some delay after DLL lock status set to ensure correct data read/write.
- 2.1.0
  - New Features
    - Added new API FLEXSPI_UpdateRxSampleClock for users to update read sample clock source after initialization.
    - Added reset peripheral operation in FLEXSPI_Init if required.
- 2.0.5
  - Bug Fixes
    - Fixed FLEXSPI_UpdateLUT cannot do partial update issue.
- 2.0.4
  - Bug Fixes
    - Reset flash size to zero for all ports in FLEXSPI_Init; fixed the possible out-of-range flash access with no error reported.
- 2.0.3
  - Bug Fixes
    - Fixed AHB receive buffer size configuration issue. The FLEXSPI_AHBRXBUFCR0_BUFSZ field should configure 64 bits size, and currently the AHB receive buffer size is in bytes which means 8-bit, so the correct configuration should be config->ahbConfig.buffer[i].bufferSize / 8.
- 2.0.2
  - New Features
    - Supported DQS write mask enable/disable feature during set FLEXSPI configuration.
    - Provided new API FLEXSPI_TransferUpdateSizeEDMA for users to update eDMA transfer size(SSIZE/DSIZE) per DMA transfer.
  - Bug Fixes
    - Fixed invalid operation of FLEXSPI_Init to enable AHB bus Read Access to IP RX FIFO.
    - Fixed incorrect operation of FLEXSPI_Init to configure IP TX FIFO watermark.
- 2.0.1
  - Bug Fixes
    - Fixed the flag clear issue and AHB read Command index configuration issue in FLEX-

SPI_SetFlashConfig.
    - Updated FLEXSPI_UpdateLUT function to update LUT table from any index instead of previous command index.
    - Added bus idle wait in FLEXSPI_SetFlashConfig and FLEXSPI_UpdateLUT to ensure bus is idle before any change to FlexSPI controller.
    - Updated interrupt API FLEXSPI_TransferNonBlocking and interrupt handle flow FLEXSPI_TransferHandleIRQ.
    - Updated eDMA API FLEXSPI_TransferEDMA.
- 2.0.0
  - Initial version.

GPC

The current GPC driver version is 2.1.1.

- 2.1.1
  - Bug Fixes
    - Moved the assert sentence that irq register number has to be greater than 0 to platforms which irq 0-31 is not available.
    - Fixed the violations of MISRA C-2012 rules:
      - Rule 10.7 12.2.
- 2.1.0
  - Improvements
    - Updated driver for IMXRT.
- 2.0.0
  - Initial version.

GPT

The current GPT driver version is 2.0.4.

- 2.0.4
  - Bug Fixes
    - Fixed compiler warning when built with FSL_SDK_DISABLE_DRIVER_CLOCK_CONTROL flag enabled.
- 2.0.3
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 5.3 by customizing function parameter.
- 2.0.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 17.7.
- 2.0.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.1, 10.3, 10.4, 10.6, 10.8, 17.7.

- 2.0.0
  - Initial version.

GPIO

The current GPIO driver version is 2.0.6.

- 2.0.6
  - Bug Fixes
    - Fixed compile warning: 'GPIO_GetInstance' defined but not used when macro FSL_SDK_DISABLE_DRIVER_CLOCK_CONTROL is defined.
- 2.0.5
  - Bug Fixes
    - Fixed MISRA C-2012 issue: rule-17.7.
- 2.0.4
  - Improvements
    - Updated the GPIO_PinWrite to use atomic operation if possible.
  - Bug Fixes
    - Fixed GPIO_PortToggle bug with platforms don't have register DR_TOGGLE.
- 2.0.3
  - Bug Fixes
    - MISRA C-2012 issue fixed.
      - Fixed rules, containing: rule-10.3, rule-14.4, and rule-15.5.
- 2.0.2
  - Bug Fixes
    - Fixed the bug of enabling wrong GPIO clock gate in initial API. Since some GPIO instances may not have a clock gate enabled, it checks the clock gate number and makes sure the clock gate is valid.
- 2.0.1
  - Improvements
    - API interface changes:
      - Refined naming of the API while keeping all original APIs, marking them as deprecated. Original APIs will be removed in next release. The main change is to update the API with prefix of _PinXXX() and _PortXXX().
- 2.0.0
  - Initial version.

KPP

The current KPP driver version is 2.0.1.

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

· Rule 10.3 10.4 10.6 14.4 17.7

- 2.0.0
  - Initial version.

LPI2C

The current LPI2C driver version is 2.4.1.

- 2.4.1
  - Improvements
    - Before master transfer with transactional APIs, enable master function while disable slave function and vise versa for slave transfer to avoid the one affecting the other.
- 2.4.0
  - Improvements
    - Split some functions, fixed CCM problem in file fsl_lpi2c.c.
  - Bug Fixes
    - Fixed bug in LPI2C_MasterInit that the MCFGR2's value set in LPI2C_MasterSetBaudRate may be overwritten by mistake.
- 2.3.2
  - Improvements
    - Initialized the EDMA configuration structure in the LPI2C EDMA driver.
- 2.3.1
  - Improvements
    - Updated LPI2C_GetCyclesForWidth to add the parameter of minimum cycle, because for master SDA/SCL filter, master bus idle/pin low timeout and slave SDA/SCL filter configuration, 0 means disabling the feature and cannot be used.
  - Bug Fixes
    - Fixed bug in LPI2C_SlaveTransferHandleIRQ that when restart detect event happens the transfer structure should not be cleared.
    - Fixed bug in LPI2C_RunTransferStateMachine, that when only slave address is transferred or there is still data remaining in tx FIFO the last byte's nack cannot be ignored.
    - Fixed bug in slave filter doze enable, that when FILTDZ is set it means disable rather than enable.
    - Fixed bug in the usage of LPI2C_GetCyclesForWidth. First its return value cannot be used directly to configure the slave FILTSDA, FILTSCL, DATAVD or CLKHOLD, because the real cycle width for them should be FILTSDA+3, FILTSCL+3, FILTSCL+DATAVD+3 and CLKHOLD+3. Second when cycle period is not affected by the prescaler value, prescaler value should be passed as 0 rather than 1.
    - Fixed wrong default setting for LPI2C slave. If enabling the slave tx SCL stall, then the default clock hold time should be set to 250ns according to I2C spec for 100kHz standard mode baudrate.
    - Fixed bug that before pushing command to the tx FIFO the FIFO occupation should be checked first in case FIFO overflow.

- 2.3.0
  - New Features
    - Supported reading more than 256 bytes of data in one transfer as master.
    - Added API LPI2C_GetInstance.
  - Bug Fixes
    - Fixed bug in LPI2C_MasterTransferAbortEDMA, LPI2C_MasterTransferAbort and LPI2C_MasterTransferHandleIRQ that before sending stop signal whether master is active and whether stop signal has been sent should be checked, to make sure no FIFO error or bus error will be caused.
    - Fixed bug in LPI2C master EDMA transactional layer that the bus error cannot be caught and returned by user callback, by monitoring bus error events in interrupt handler.
    - Fixed bug in LPI2C_GetCyclesForWidth that the parameter used to calculate clock cycle should be 2^prescaler rather than prescaler.
    - Fixed bug in LPI2C_MasterInit that timeout value should be configured after baudrate, since the timeout calculation needs prescaler as parameter which is changed during baudrate configuration.
    - Fixed bug in LPI2C_MasterTransferHandleIRQ and LPI2C_RunTransferStateMachine that when master writes with no stop signal, need to first make sure no data remains in the tx FIFO before finishes the transfer.
- 2.2.0
  - Bug Fixes
    - Fixed issue that the SCL high time, start hold time and stop setup time do not meet I2C specification, by changing the configuration of data valid delay, setup hold delay, clock high and low parameters.
    - MISRA C-2012 issue fixed.
      - Fixed rule 8.4, 13.5, 17.7, 20.8.
- 2.1.12
  - Bug Fixes
    - Fixed MISRA advisory 15.5 issues.
- 2.1.11
  - Bug Fixes
    - Fixed the bug that, during master non-blocking transfer, after the last byte is sent/received, the kLPI2C_MasterNackDetectFlag is expected, so master should not check and clear kLPI2C_MasterNackDetectFlag when remainingBytes is zero, in case FIFO is emptied when stop command has not been sent yet.
    - Fixed the bug that, during non-blocking transfer slave may nack master while master is busy filling tx FIFO, and NDF may not be handled properly.
- 2.1.10
  - Bug Fixes
    - MISRA C-2012 issue fixed.
      - Fixed rule 10.3, 14.4, 15.5.
    - Fixed unaligned access issue in LPI2C_RunTransferStateMachine.
    - Fixed uninitialized variable issue in LPI2C_MasterTransferHandleIRQ.
    - Used linked TCD to disable tx and enable rx in read operation to fix the issue that for

platform sharing the same DMA request with tx and rx, during LPI2C read operation if interrupt with higher priority happened exactly after command was sent and before tx disabled, potentially both tx and rx could trigger dma and cause trouble.
    - Fixed MISRA issues.
        - Fixed rules 10.1, 10.3, 10.4, 11.6, 11.9, 14.4, 17.7.
    - Fixed the waitTimes variable not re-assignment issue for each byte read.
  - New Features
    - Added the IRQHandler for LPI2C5 and LPI2C6 instances.
  - Improvements
    - Updated the LPI2C_WAIT_TIMEOUT macro to unified name I2C_RETRY_TIMES.
- 2.1.9
  - Bug Fixes
    - Fixed Coverity issue of unchecked return value in I2C_RTOS_Transfer.
    - Fixed Coverity issue of operands did not affect the result in LPI2C_SlaveReceive and LPI2C_SlaveSend.
    - Removed STOP signal wait when NAK detected.
    - Cleared slave repeat start flag before transmission started in LPI2C_SlaveSend/LPI2C_SlaveReceive. The issue was that LPI2C_SlaveSend/LPI2C_SlaveReceive did not handle with the reserved repeat start flag. This caused the next slave to send a break, and the master was always in the receive data status, but could not receive data.
- 2.1.8
  - Bug Fixes
    - Fixed the transfer issue with LPI2C_MasterTransferNonBlocking, kLPI2C_TransferNoStopFlag, with the wait transfer done through callback in a way of not doing a blocking transfer.
    - Fixed the issue that STOP signal did not appear in the bus when NAK event occurred.
- 2.1.7
  - Bug Fixes
    - Cleared the stopflag before transmission started in LPI2C_SlaveSend/LPI2C_SlaveReceive. The issue was that LPI2C_SlaveSend/LPI2C_SlaveReceive did not handle with the reserved stop flag and caused the next slave to send a break, and the master always stayed in the receive data status but could not receive data.
- 2.1.6
  - Bug Fixes
    - Fixed driver MISRA build error and C++ build error in LPI2C_MasterSend and LPI2C_SlaveSend.
    - Reset FIFO in LPI2C Master Transfer functions to avoid any byte still remaining in FIFO during last transfer.
    - Fixed the issue that LPI2C_MasterStop did not return the correct NAK status in the bus for second transfer to the non-existing slave address.
- 2.1.5
  - Bug Fixes
    - Extended the Driver IRQ handler to support LPI2C4.
    - Changed to use ARRAY_SIZE(kLpi2cBases) instead of FEATURE COUNT to decide the array size for handle pointer array.

- 2.1.4
  - Bug Fixes
    - Fixed the LPI2C_MasterTransferEDMA receive issue when LPI2C shared same request source with TX/RX DMA request. Previously, the API used scatter-gather method, which handled the command transfer first, then the linked TCD which was pre-set with the receive data transfer. The issue was that the TX DMA request and the RX DMA request were both enabled, so when the DMA finished the first command TCD transfer and handled the receive data TCD, the TX DMA request still happened due to empty TX FIFO. The result was that the RX DMA transfer would start without waiting on the expected RX DMA request.
    - Fixed the issue by enabling IntMajor interrupt for the command TCD and checking if there was a linked TCD to disable the TX DMA request in LPI2C_MasterEDMA-Callback API.
- 2.1.3
  - Improvements
    - Added LPI2C_WATI_TIMEOUT macro to allow the user to specify the timeout times for waiting flags in functional API and blocking transfer API.
    - Added LPI2C_MasterTransferBlocking API.
- 2.1.2
  - Bug Fixes
    - In LPI2C_SlaveTransferHandleIRQ, reset the slave status to idle when stop flag was detected.
- 2.1.1
  - Bug Fixes
    - Disabled the auto-stop feature in eDMA driver. Previously, the auto-stop feature was enabled at transfer when transferring with stop flag. Since transfer was without stop flag and the auto-stop feature was enabled, when starting a new transfer with stop flag, the stop flag would be sent before the new transfer started, causing unsuccesful sending of the start flag, so the transfer could not start.
    - Changed default slave configuration with address stall false.
- 2.1.0
  - Improvements
    - API name changed:
      - LPI2C_MasterTransferCreateHandle -> LPI2C_MasterCreateHandle.
      - LPI2C_MasterTransferGetCount -> LPI2C_MasterGetTransferCount.
      - LPI2C_MasterTransferAbort -> LPI2C_MasterAbortTransfer.
      - LPI2C_MasterTransferHandleIRQ -> LPI2C_MasterHandleInterrupt.
      - LPI2C_SlaveTransferCreateHandle -> LPI2C_SlaveCreateHandle.
      - LPI2C_SlaveTransferGetCount -> LPI2C_SlaveGetTransferCount.
      - LPI2C_SlaveTransferAbort -> LPI2C_SlaveAbortTransfer.
      - LPI2C_SlaveTransferHandleIRQ -> LPI2C_SlaveHandleInterrupt.
- 2.0.0
  - Initial version.

LPSPI

The current LPSPI driver version is 2.4.0.

- 2.4.0
  - Improvements
    - Split some functions, fixed CCM problem in file fsl_lpspi.c.
- 2.3.1
  - Improvements
    - Initialized the EDMA configuration structure in the LPSPI EDMA driver.
  - Bug Fixes
    - Fixed bug that function LPSPI_MasterTransferBlocking should return after the transfer complete flag is set to make sure the PCS is re-asserted.
- 2.3.0
  - New Features
    - Supported the master configuration of sampling the input data using a delayed clock to improve slave setup time.
- 2.2.1
  - Bug Fixes
    - Fixed bug in LPSPI_SetPCSContinous when disabling PCS continous mode.
- 2.2.0
  - Bug Fixes
    - Fixed bug in 3-wire polling and interrupt transfer that the received data is not correct and the PCS continous mode is not working.
- 2.1.0
  - Improvements
    - Improved LPSPI_SlaveTransferHandleIRQ to fill up TX FIFO instead of write one data to TX register which improves the slave transmit performance.
    - Added new functional APIs LPSPI_SelectTransferPCS and LPSPI_SetPCSContinous to support changing PCS selection and PCS continous mode.
  - Bug Fixes
    - Fixed bug in non-blocking and EDMA transfer APIs that kStatus_InvalidArgument is returned if user configures 3-wire mode and full-duplex transfer at the same time, but transfer state is already set to kLPSPI_Busy by mistake causing following transfer can not start.
    - Fixed bug when LPSPI slave using EDMA way to transfer, tx should be masked when tx data is null, otherwise in 3-wire mode which tx/rx use the same pin, the received data will be interfered.
- 2.0.5
  - Improvements
    - Added timeout mechanism when waiting certain states in transfer driver.
  - Bug Fixes
    - Fixed the bug that LPSPI can not transfer large data using EDMA.
    - Fixed MISRA 17.7 issues.
    - Fixed variable overflow issue introduced by MISRA fix.
    - Fixed issue that rxFifoMaxBytes should be calculated according to transfer width rather

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

LPUART

The current LPUART driver version is 2.7.0.

- 2.7.0
  - Improvements
    - Split some functions, fixed CCM problem in file fsl_lpuart.c.
- 2.6.0
  - Bug Fixes
    - Fixed bug that when there are multiple lpuart instance, unable to support different ISR.
- 2.5.3
  - Bug Fixes
    - Fixed comments by replacing unused status flags kLPUART_NoiseErrorInRxDataRegFlag and kLPUART_ParityErrorInRxDataRegFlag with kLPUART_NoiseErrorFlag and kLPUART_ParityErrorFlag.

- 2.5.2
  - Bug Fixes
    - Fixed bug that when setting watermark for TX or RX FIFO, the value may exceed the maximum limit.
  - Improvements
    - Added check in LPUART_TransferDMAHandleIRQ and LPUART_TransferEdmaHandleIRQ to ensure if user enables any interrupts other than transfer complete interrupt, the dma transfer is not terminated by mistake.
- 2.5.1
  - Improvements
    - Use separate data for TX and RX in lpuart_transfer_t.
  - Bug Fixes
    - Fixed bug that when ring buffer is used, if some data is received in ring buffer first before calling LPUART_TransferReceiveNonBlocking, the received data count returned by LPUART_TransferGetReceiveCount is wrong.
- 2.5.0
  - Bug Fixes
    - Added missing interrupt enable masks kLPUART_Match1InterruptEnable and kLPUART_Match2InterruptEnable.
    - Fixed bug in LPUART_EnableInterrupts, LPUART_DisableInterrupts and LPUART_GetEnabledInterrupts that the BAUD[LBKDIE] bit field should be soc specific.
    - Fixed bug in LPUART_TransferHandleIRQ that idle line interrupt should be disabled when rx data size is zero.
    - Deleted unused status flags kLPUART_NoiseErrorInRxDataRegFlag and kLPUART_ParityErrorInRxDataRegFlag, since firstly their function are the same as kLPUART_NoiseErrorFlag and kLPUART_ParityErrorFlag, secondly to obtain them one data word must be read out thus interfering with the receiving process.
    - Fixed bug in LPUART_GetStatusFlags that the STAT[LBKDIF], STAT[MA1F] and STAT[MA2F] should be soc specific.
    - Fixed bug in LPUART_ClearStatusFlags that tx/rx FIFO is reset by mistake when clearing flags.
    - Fixed bug in LPUART_TransferHandleIRQ that while clearing idle line flag the other bits should be masked in case other status bits be cleared by accident.
    - Fixed bug of race condition during LPUART transfer using transactional APIs, by disabling and re-enabling the global interrupt before and after critical operations on interrupt enable register.
    - Fixed DMA/eDMA transfer blocking issue by enabling tx idle interrupt after DMA/eDMA transmission finishes.
  - New Features
    - Added APIs LPUART_GetRxFifoCount/LPUART_GetTxFifoCount to get rx/tx FIFO data count.
    - Added APIs LPUART_SetRxFifoWatermark/LPUART_SetTxFifoWatermark to set rx/tx FIFO water mark.
- 2.4.1
  - Bug Fixes

    - Fixed MISRA advisory 17.7 issues.
- 2.4.0
  - New Features
    - Added APIs to configure 9-bit data mode, set slave address and send address.
- 2.3.1
  - Bug Fixes
    - Fixed MISRA advisory 15.5 issues.
- 2.3.0
  - Improvements
    - Modified LPUART_TransferHandleIRQ so that txState will be set to idle only when all data has been sent out to bus.
    - Modified LPUART_TransferGetSendCount so that this API returns the real byte count that LPUART has sent out rather than the software buffer status.
    - Added timeout mechanism when waiting for certain states in transfer driver.
- 2.2.8
  - Bug Fixes
    - Fixed issue for MISRA-2012 check.
      - Fixed rule-10.3, rule-14.4, rule-15.5.
    - Eliminated Pa082 warnings by assigning volatile variables to local variables and using local variables instead.
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 10.8, 14.4, 11.6, 17.7.
  - Improvements
    - Added check for kLPUART_TransmissionCompleteFlag in LPUART_WriteBlocking, LPUART_TransferHandleIRQ, LPUART_TransferSendDMACallback and LPUART_SendEDMACallback to ensure all the data would be sent out to bus.
    - Rounded up the calculated sbr value in LPUART_SetBaudRate and LPUART_Init to achieve more acurate baudrate setting. Changed osr from uint32_t to uint8_t since osr's bigest value is 31.
    - Modified LPUART_ReadBlocking so that if more than one receiver errors occur, all status flags will be cleared and the most severe error status will be returned.
- 2.2.7
  - Bug Fixes
    - Fixed issue for MISRA-2012 check.
      - Fixed rule-12.1, rule-17.7, rule-14.4, rule-13.3, rule-14.4, rule-10.4, rule-10.8, rule-10.3, rule-10.7, rule-10.1, rule-11.6, rule-13.5, rule-11.3, rule-13.2, rule-8.3.
- 2.2.6
  - Bug Fixes
    - Fixed the issue of register's being in repeated reading status while dealing with the IRQ routine.
- 2.2.5
  - Bug Fixes
    - Do not set or clear the TIE/RIE bits when using LPUART_EnableTxDMA and LPUART_EnableRxDMA.
- 2.2.4

  - Improvements
    - Added hardware flow control function support.
    - Added idle-line-detecting feature in LPUART_TransferNonBlocking function. If an idle line is detected, a callback is triggered with status kStatus_LPUART_IdleLine-Detected 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 F-IFO 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

OCOTP

The current OCOTP driver version is 2.1.3.

- 2.1.3
  - Bug fixes
    - Fixed MISRA 2012 issue: 8.4, 10.3, 10.4, 14.3.
    - Fixed doxygen warning.
- 2.1.2
  - Improvements
    - Updated for new MIMXRT117X header file.
- 2.1.1
  - Improvements
    - Updated OCOTP_ReloadShadowRegister to return error status.
    - Added functions OCOTP_ReadFuseShadowRegisterExt and OCOTP_WriteFuseShadowRegisterWithLock.
  - Bug fixes
    - Fixed MISRA 2012 rule 10.3 issue.
- 2.0.1
  - Bug Fixes
    - Fixed doxygen issues.
- 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.

PMU

The current PMU driver version is 2.1.1.

- 2.1.1
    - Bug Fixes
        - Fixed the violations of MISRA 2012 rules: Rule 10.1 10.4
- 2.1.0
    - Improvements
        - Added feature macros for low power control APIs to support conditional compile.
        - Renamed "PMU_2P1EnablePullDown" to "PMU_2P5EnablePullDown".
- 2.0.0
    - Initial version.

PWM

The current PWM driver version is 2.5.1.

- 2.5.1
    - Bug Fixes
        - Fixed MISRA C-2012 rules: 10.1, 10.3, 10.4 , 10.6 and 10.8.
        - Fixed the issue that PWM_UpdatePwmDutycycle() can't update duty cycle status value correct.
- 2.5.0
    - Improvements
        - Added API PWM_SetOouputToIdle to set pwm channel output to idle.
        - Added API PWM_GetPwmChannelState to get the pwm channel output duty cycle value.
        - Added API PWM_SetPwmForceOutputToZero to set the pwm channel output to zero logic.
        - Added API PWM_SetChannelOutput to set the pwm channel output state.
        - Added API PWM_SetClockMode to set the value of the clock prescaler.
        - Added API PWM_SetupPwmPhaseShift to set PWM which a special phase shift and

50% duty cycle.
        - Added API PWM_SetVALxValue/PWM_GetVALxValue to set/get PWM VALs registers values directly.
- 2.4.0
  - Improvements
    - Supported the PWM which can't work in wait mode.
- 2.3.0
  - Improvements
    - Add PWM output enable&disbale API for SDK.
  - Bug Fixes
    - Fixed changing channel B configuration when parameter is kPWM_PWMX and PWMX configuration is not supported yet.
- 2.2.1
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rules: 10.3, 10.4.
  - Bug Fixes
    - Fixed the issue that PWM drivers computed VAL1 improperly.
  - Improvements
    - Updated calculation accuracy of reloadValue in dutyCycleToReloadValue function.
- 2.2.0
  - Improvements
    - Added new enumeration and two APIs to support enabling and disabling one or more PWM output triggers.
    - Added a new function to make the most of 16-bit resolution PWM.
    - Added one API to support updating fault status of PWM output.
    - Added one API to support PWM DMA write request.
    - Added three APIs to support PWM DMA capture read request.
    - Added one API to support get default fault config of PWM.
    - Added one API to support setting PWM fault disable mapping.
- 2.1.0
  - Improvements
    - Moved the configuration of fault input filter into a new API to avoid be initialized multiple times.
  - Bug Fixes
    - MISRA C-2012 issue fixed.
      - Fix rules, containing: rule-10.2, rule-10.3, rule-10.4, rule-10.7, rule-10.8, rule-14.4, rule-16.4.
- 2.0.1
  - Bug Fixes
    - Fixed the issue that PWM submodule may be initialized twice in function PWM_SetupPwm().
- 2.0.0
  - Initial version.

QTMR

The current QTMR driver version is 2.2.1.

- 2.2.1
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rules: 10.1, 10.8.
- 2.2.0
  - Improvements
    - Added API QTMR_SetPwmOutputToIdle to set the generated pwm signal to the configured idle value.
    - Added API QTMR_GetPwmOutputStatus to return the output status of the generated pwm signal.
    - Added API QTMR_GetPwmChannelStatus to return the channel dutycycle value.
    - Added API QTMR_SetPwmClockMode to set clock mode change peripheral clock frequency.
  - Bug Fixes
    - Fixed the issue that pwm duty cycle could not be 0 and 100.
- 2.1.0
  - Bug Fixes
    - Fixed the issue QTMR_SetTimerPeriod needs to decrement down count by 1, and added new APIs to configure the LOAD register, COMP register.
- 2.0.2
  - Bug Fixes
    - Fixed the issue introduced by previous code correction for improving the output signal accuracy.
- 2.0.1
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rules: 10.1, 10.3, 11.5, 11.9.
  - Improvements
    - Improved the output signal accuracy.
- 2.0.0
  - Initial version.

RTWDOG

The current RTWDOG 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-10.8, rule-11.9, rule-14.4, rule-15.5.
- 2.1.0

  - Improvements
    - Added an API to enable or disable the window mode.
    - Added an API to convert a raw count value to millisecond.
    - Used AT_QUICKACCESS_SECTION_CODE macro to decorate RTWDOG_Init, and copied this function from flash to QUICKACCESS section.
- 2.0.1
  - Bug Fixes
    - Fixed bug in the RTWDOG_Init; added check for register's unlock status when configuring the RTWDOG in RTWDOG_init.
- 2.0.0
  - Initial version.

SAI

The current SAI driver version is 2.3.8

- 2.3.8
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.4.
- 2.3.7
  - Improvements
    - Change feature "FSL_FEATURE_SAI_FIFO_COUNT" to "FSL_FEATURE_SAI_HAS_FIFO".
    - Added feature "FSL_FEATURE_SAI_FIFO_COUNTn(x)" to align SAI fifo count function with IP in function

2.3.6

- Bug Fixes
  - Fixed violations of MISRA C-2012 rule 5.6.
- 2.3.5
  - Improvements
    - Make driver to be aarch64 compatible.
- 2.3.4
  - Bug Fixes
    - Corrected the fifo combine feature macro used in driver.
- 2.3.3
  - Bug Fixes
    - Added bit clock polarity configuration when sai act as slave.
    - Fixed out of bound access coverity issue.
    - Fixed violations of MISRA C-2012 rule 10.3, 10.4.
- 2.3.2
  - Bug Fixes
    - Corrected the frame sync configuration when sai act as slave.
- 2.3.1

  - Bug Fixes
    - Corrected the peripheral name in function SAI0_DriverIRQHandler.
    - Fixed violations of MISRA C-2012 rule 17.7.
- 2.3.0
  - Bug Fixes
    - Fixed the build error caused by the SOC has no fifo feature.
- 2.2.3
  - Bug Fixes
    - Corrected the peripheral name in function SAI0_DriverIRQHandler.
- 2.2.2
  - Bug Fixes
    - Fixed the issue of MISRA 2004 rule 9.3.
    - Fixed sign-compare warning.
    - Fixed the PA082 build warning.
    - Fixed sign-compare warning.
    - Fixed violations of MISRA C-2012 rule 10.3,17.7,10.4,8.4,10.7,10.8,14.4,17.7,11.-6,10.1,10.6,8.4,14.3,16.4,18.4.
    - Allow to reset Rx or Tx FIFO pointers only when Rx or Tx is disabled.
  - Improvements
    - Added 24bit raw audio data width support in sai sdma driver.
    - Disabled the interrupt/DMA request in the SAI_Init to avoid generates unexpected sai FIFO requests.
- 2.2.1
  - Improvements
    - Added mclk post divider support in function SAI_SetMasterClockDivider.
    - Removed useless configuration code in SAI_RxSetSerialDataConfig.
  - Bug Fixes
    - Fixed the SAI SDMA driver build issue caused by the wrong structure member name used in the function SAI_TransferRxSetConfigSDMA/SAI_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_CHANNEL_COUNTn(base) for the difference between the different SAI instances.
  - Added new APIs:
    - SAI_TxSetBitClockDirection
    - SAI_RxSetBitClockDirection
    - SAI_RxSetFrameSyncDirection
    - SAI_TxSetFrameSyncDirection
- 2.1.8
  - Improvements
    - Added feature macro test for the sync mode2 and mode 3.
    - Added feature macro test for masterClockHz in sai_transfer_format_t.
- 2.1.7
  - Improvements
    - Added feature macro test for the mclkSource member in sai_config_t.
    - Changed "FSL_FEATURE_SAI5_SAI6_SHARE_IRQ" to "FSL_FEATURE_SAI_SAI5_SAI6_SHARE_IRQ".
    - Added #ifndef #endif check for SAI_XFER_QUEUE_SIZE to allow redefinition.
  - Bug Fixes
    - Fixed build error caused by feature macro test for mclkSource.
- 2.1.6
  - Improvements
    - Added feature macro test for mclkSourceClockHz check.
    - Added bit clock source name for general devices.
  - Bug Fixes
    - Fixed incorrect channel numbers setting while calling RX/TX set format together.
- 2.1.5
  - Bug Fixes
    - Corrected SAI3 driver IRQ handler name.
    - Added I2S4/5/6 IRQ handler.

        * Added base in handler structure to support different instances sharing one IRQ number.
    - New Features
        * Updated SAI driver for MCR bit MICS.
        * Added 192 KHZ/384 KHZ in the sample rate enumeration.
        * Added multi FIFO interrupt/SDMA transfer support for TX/RX.
        * Added an API to read/write multi FIFO data in a blocking method.
        * Added bclk bypass support when bclk is same with mclk.
- 2.1.4
    - New Features
        * Added an API to enable/disable auto FIFO error recovery in platforms that support this feature.
        * Added an API to set data packing feature in platforms which support this feature.
- 2.1.3
    - New Features
        * Added feature to make I2S frame sync length configurable according to bitWidth.
- 2.1.2
    - Bug Fixes
        * Added 24-bit support for SAI eDMA transfer. All data shall be 32 bits for send/receive, as eDMA cannot directly handle 3-Byte transfer.
- 2.1.1
    - Improvements
        * Reduced code size while not using transactional API.
- 2.1.0
    - Improvements
        * API name changes:
            · SAI_GetSendRemainingBytes -> SAI_GetSentCount.
            · SAI_GetReceiveRemainingBytes -> SAI_GetReceivedCount.
            · All names of transactional APIs were added with "Transfer" prefix.
            · All transactional APIs use base and handle as input parameter.
            · Unified the parameter names.
    - Bug Fixes
        * Fixed WLC bug while reading TCSR/RCSR registers.
        * Fixed MOE enable flow issue. Moved MOE enable after MICS settings in SAI_TxInit/-SAI_RxInit.
- 2.0.0
    - Initial version.

SEMC

The current SEMC driver version is 2.4.3.

- 2.4.3
    - Bug Fixes
        * Fixed violations of the MISRA C-2012 Rule 5.6.

- 2.4.2
  - Improvements
    - Deleted meaningless parameter in memory size conversion function.
- 2.4.1
  - Bug Fixes
    - Fixed PSRAM A8 configuration issue, which should be 0x06U for PSRAM while pix mux bit width is 0x04U, based on different pix mux bit width.
- 2.4.0
  - Improvements
    - Improved nor and sram timing configuration on sync mode.
- 2.3.1
  - Bug Fixes
    - Updated refresh timer period(RT) timing setting, which updated into (RT+1)∗(Prescaler period) for SDRAM.
    - Supported new DBI control register 2 to configure CSX interval time(CEITV).
    - Fixed violations of the MISRA C-2012 Rule 10.8.
    - Fixed doxygen warning.
- 2.3.0
  - New Features
    - Limited burst length as 1 according to ERR050577, Auto-refresh command may possibly fail to be triggered during long time back-to-back write (or read) when SDRAM controller's burst length is greater than 1.
    - Supported 8 bits column address for SDRAM.
- 2.2.1
  - New Features
    - Added queue weight control, which can control queue a/b is working or not.
    - Updated NAND FLASH configuration API which disables and enables SEMC between configure control registers.
    - Added ONFI parameter Integrity CRC check for SEMC flash component.
- 2.2.0
  - New Features
    - Supported up to 4 PSRAM CS.
    - Added programmable delay line for DQS.
    - Added ready/wait feature for SRAM in asynchronous mode.
- 2.1.0
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.3, 10.4, and 14.4.
    - Updated parameter type from uint16_t into uint32_t for send IP command API.
- 2.0.4
  - Bug Fixes
    - Fixed the SEMC queueA and queueB weight configuration issue.
    - Fixed the wrong configuration of DBICR1 register in SEMC_ConfigureDBI.
- 2.0.3
  - Bug Fixes
    - Added feature macro to control WDS&WDH bit setting for NOR synchronous transfer.

- 2.0.2
  - Bug Fixes
    - Changed SEMC NAND configuration structure and verify SEMC NAND related APIs.
    - Added extended SEMC clock enable.
- 2.0.1
  - Bug Fixes
    - Fixed data size mask configure in SEMC_ConfigureIPCommand API.
    - Updated the command mode in IP command type.
- 2.0.0
  - Initial version.

SPDIF

The current SPDIF driver version is 2.0.6.

- 2.0.6
  - Bug Fixes
    - Fixed the Q/U channel interrupt enabled unexpectly while Q/U transfer pointer is NULL.
- 2.0.5
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.3.
- 2.0.4
  - Bug Fixes
    - Added udata/qdata buffer address validation in driver IRQ handler to ensure that NULL pointer dereferences do not occur.
- 2.0.3
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.3, 10.4, and 14.4.
- 2.0.2
  - Bug Fixes
    - Corrected operator used for size value assertion in SPDIF_ReadBlocking/SPDIF_WriteBlocking.
- 2.0.1
  - Bug Fixes
    - Corrected the feature macro name used to define s_edmaPrivateHandle.
- 2.0.0
  - Initial version.

SRC

The current SRC driver version is 2.0.1.

- 2.0.1

  - Improvements
    - Updated SRC driver for adding SRC_SRSR_JTAG_SW_RST enumeration.
- 2.0.0
  - Initial version.

TEMPMON

The current TEMPMON driver version is 2.1.1.

- 2.1.1
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rules:
      - Rule 10.3 10.4.
- 2.1.0
  - Bug Fixes
    - Supported minus value for alarm temperature setting.
    - Fixed wrong temperature calculation equation.
- 2.0.3
  - Improvements
    - Added temperature threshold check for high/low/panic to avoid temperature overflow.
- 2.0.2
  - Bug Fixes
    - Fixed wrong alarm value setting API, it need to clear it firstly and set a new value into it.
- 2.0.1
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rules:
      - Rule 10.1 10.3 10.4 10.8 17.7.
- 2.0.0
  - Initial version.

USDHC

The current USDHC driver version is 2.8.2.

- 2.8.2
  - Improvements
    - Added feature macro FSL_FEATURE_USDHC_HAS_NO_VOLTAGE_SELECT.
- 2.8.1
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9.
- 2.8.0
  - Improvements
    - Fixed the mmc boot transfer failed issue which is caused by the Dma complete interrupt

not enabled.
    - Marked api USDHC_AdjustDelayForManualTuning as deprecated and added new api USDHC_SetTuingDelay/USDHC_GetTuningDelayStatus.
    - Improved the manual tuning flow accroding to specification.
    - Added memory address conversion to support buffers which could only be accessed using alias address by non-core masters.
    - Fixed violations of MISRA C-2012 rule 10.4.
- 2.7.0
  - Improvements
    - Added api USDHC_TransferScatterGatherADMANonBlocking to support scatter gather transfer.
    - Added feature FSL_FEATURE_USDHC_REGISTER_HOST_CTRL_CAP_HAS_NO_RETUNING_TIME_COUNTER for re-tuning time counter field in HOST_CTRL_CAP register.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9, 10.1, 10.3, 10.4, 8.4.
- 2.6.0
  - Improvements
    - Added api USDHC_SetStandardTuningCounter to support adjust tuning counter of Standard tuning.
- 2.5.1
  - Improvements
    - Used different status code for command and data interrupt callback.
    - Added cache line invalidate for receive buffer in driver IRQ handler to fix CM7 speculative access issue.
- 2.5.0
  - Improvements
    - Added new api USDHC_SetStrobeDllOverride for HS400 strobe dll override mode delay taps configurations.
    - Corrected the STROBE DLL configurations sequence.
- 2.4.0
  - Improvements
    - Added feature macro for read/write burst length.
      - Disabled redundant interrupt per different transfer request.
      - Disabled interrupt and reset command/data pointer in handle when transfer completes.
  - 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.
    - Fixed PA082 build warning.
    - Fixed logically dead code Coverity issue.
- 2.3.0
  - Improvements
    - Added USDHC_SetDataConfig API to support manual tuning.
    - Removed the limitaion that source clock must be bigger than the target in function US-

DHC_SetSdClock by using source clock frequency as target directly.
    - Added peripheral reset in USDHC_Init function.
    - Added tuning reset support in function USDHC_Reset function.
- 2.2.8
  - Bug Fixes
    - Fixed out-of bounds write in function USDHC_ReceiveCommandResponse.
- 2.2.7
  - Improvements
    - Added API USDHC_GetEnabledInterruptStatusFlags and used in USDHC_Transfer-HandleIRQ.
    - Removed useless member interruptFlags in usdhc_handle_t.
- 2.2.6
  - Improvements
    - Added address align check for ADMA descriptor table address.
    - Changed USDHC_ADMA1_DESCRIPTOR_MAX_LENGTH_PER_ENTRY to (65536-4096) to make sure the data address is 4KB align for a transfer which need more than one ADMA1 descriptor.
- 2.2.5
  - Bug Fixes
    - Fixed MDK 66-D warning.
- 2.2.4
  - Bug Fixes
    - Fixed issue that real clock frequency wss mismatched with target clock frequency, which was caused by an incorrect prescaler calculation.
  - New Features
    - Added control macro to enable/disable the CLOCK code in current driver.
- 2.2.3
  - Bug Fixes
    - Fixed issue where AMDA did not disable with DMAEN clear.
  - Improvements
    - Improved set clock function to check the output frequency range.
    - Dynamic set SDCLKFS during DDR enable or disable.
- 2.2.2
  - Improvements
    - Improved read transfer cache maintain operation, combined clean, and invalidated them into one function.
- 2.2.1
  - Bug Fixes
    - Disabled the invalidate cache operation for tuning.
- 2.2.0
  - Improvements
    - Improved USDHC to support MMC boot feature.
- 2.1.3
  - Bug Fixes
    - Fixed MISRA issue.

- 2.1.2
  - Bug Fixes
    - Fixed Coverity issue.
    - Added base address and userData parameter for all callback functions.
- 2.1.1
  - Improvements
    - Added cache maintain operation.
    - Added timeout status check for the DATA transfer which ignore error.
    - Added feature macro for SDR50/SDR104 mode.
    - Removed useless IRQ handler from different platforms.
- 2.1.0
  - Improvements
    - Integrated tuning into transfer function.
    - Added strobe DLL feature.
    - Added enableAutoCommand23 in data structure.
    - Removed enable card clock function because the controller would handle the clock on/off.
- 2.0.0
  - Initial version.

WDOG

The current WDOG driver version is 2.1.1.

- 2.1.1
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.1, 10.3, 10.4, 10.6, 10.7 and 11.9.
    - Fixed the issue of the inseparable process interrupted by other interrupt source.
      - WDOG_Init
      - WDOG_Refresh
- 2.1.0
  - New Features
    - Added new API "WDOG_TriggerSystemSoftwareReset()" to allow users to reset the system by software.
    - Added new API "WDOG_TriggerSoftwareSignal()" to allow users to trigger a WDOG_B signal by software.
    - Removed the parameter "softwareAssertion" and "softwareResetSignal" out of the wdog_config_t structure.
    - Added new parameter "enableTimeOutAssert" to the wdog_config_t structure. With this parameter enabled, when the WDOG timeout occurs, a WDOG_B signal will be asserted. This signal can be routed to external pin of the chip. Note that WDOG_B signal remains asserted until a power-on reset (POR) occurs.
- 2.0.1
  - New Features

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

XBARA

The current XBARA driver version is 2.0.6.

- 2.0.6
    - Bug Fixes
        - Fixed typo in kXBARA_RequestInterruptEnalbe item.
- 2.0.5
    - Bug Fixes
        - Fixed IAR build warning Pa082.
        - Fixed violations of the MISRA C-2012 rules 10.1, 10.3, 10.4, 10.6, 10.7, 10.8, 12.1, 18.1, 20.7.
- 2.0.4
    - Improvements
        - Optimized XBARA_SetOutputSignalConfig.
- 2.0.3
    - Bug Fixes
        - Corrected configuration for function XBAR_SetOutputSignalConfig.
- 2.0.2
    - Other Changes
        - Changed array clock name.
- 2.0.1
    - Bug Fixes
        - Fixed w1c bits for XBARA_SetOutputSignalConfig function.
- 2.0.0
    - Initial version.

XBARB

The current XBARB driver version is 2.0.2.

- 2.0.2
    - Bug Fixes
        - Fixed violations of the MISRA C-2012 rules 12.2, 10.7
- 2.0.1
    - Bug Fixes
        - Corrected XBARB_SetSignalsConnection function.
    - Other Changes
        - Changed array clock name.
- 2.0.0

- Initial version.

2 Middleware Change Log

FatFs for MCUXpresso SDK

Current version is FatFs R0.14b_rev0.

- R0.14b_rev1
  - Applied patches from http://elm-chan.org/fsw/ff/patches.html
- R0.14b_rev0
  - Upgraded to version 0.14b
- R0.14a_rev0
  - Upgraded to version 0.14a
  - Applied patch ff14a_p1.diff and ff14a_p2.diff
- R0.14_rev0
  - Upgraded to version 0.14
  - Applied patch ff14_p1.diff and ff14_p2.diff
- R0.13c_rev0
  - Upgraded to version 0.13c
  - Applied patches ff_13c_p1.diff,ff_13c_p2.diff, ff_13c_p3.diff and ff_13c_p4.diff.
- R0.13b_rev0
  - Upgraded to version 0.13b
- R0.13a_rev0
  - Upgraded to version 0.13a. Added patch ff_13a_p1.diff.
- R0.12c_rev1
  - Add NAND disk support.
- R0.12c_rev0
  - Upgraded to version 0.12c and applied patches ff_12c_p1.diff and ff_12c_p2.diff.
- R0.12b_rev0
  - Upgraded to version 0.12b.
- R0.11a
  - Added glue functions for low-level drivers (SDHC, SDSPI, RAM, MMC). Modified diskio.c.
  - Added RTOS wrappers to make FatFs thread safe. Modified syscall.c.
  - Renamed ffconf.h to ffconf_template.h. Each application should contain its own ffconf.h.
  - Included ffconf.h into diskio.c to enable the selection of physical disk from ffconf.h by macro definition.
  - Conditional compilation of physical disk interfaces in diskio.c.

FreeMASTER Communication Driver

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

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

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

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_rev7
  - New features:
    - Ported lwIP 2.2.0.dev (2022-05-09, branch: master, SHA-1: 239918ccc173cb2c2a62f41a40fd893f5 to MCUXpresso SDK.
    - Added function ethernetif_probe_link() which reads actual link, speed and duplex settings from phy and passes them to driver. Stack could be set to call this function periodically by setting ETH_LINK_POLLING_INTERVAL_MS to value higher than zero.
    - Added helper functions ethernetif_wait_linkup() and ethernetif_wait_ipv4_valid() to allow blocking of RTOS task or bare metal application until link is up or IPv4 address becomes valid.
    - Added NETC adaptation layer.
    - Processing of rx packets under RTOS moved from ISR to a separate task to improve system reaction times. Switch back to old behavior can be done by setting ETH_DO_-RX_IN_SEPARATE_TASK macro to 0.
  - Bug fixes:
    - port: Fixed copying of pbuf contents. Previous code was using an incorrect end condition and could result in the overrun of the destination buffer if more packets were on the queue.
    - port: Delegating pbuf_free calls to tcpip_thread via pbuf_free_callback where possible (RTOS), ensured pbuf_free is not called from interrupt context when LWIP_ALLOW-_MEM_FREE_FROM_OTHER_CONTEXT is not set (bare metal).
    - port/enet_ethernetif_qos.c - Fixed ENET_RXBD_NUM which was used instead of E-NET_TXBD_NUM.
    - port/enet_ethernetif_qos.c - Fixed buffer alignment to be at least 64.
    - src/apps/lwiperf: Fixed IPv6 TCP TX throughput lower than IPv4 by modifying maximum segment size to avoid sending two segments instead of one.
    - src/apps/lwiperf: Out-of-order datagrams in UDP RX server mode are counted to the throughput.
    - src/apps/httpsrv: Implemented receive timeouts on sockets.
    - src/apps/httpsrv: Don't assert on HTTP session task creation failure.
    - src/apps/httpsrv: Fixed build with IPv6 enabled.
    - src/apps/httpsrv: Updated endianess macros required for websocket SHA generation.
    - src/apps/httpsrv: Added missing includes.
- 2.2.0_rev6
  - New features:
    - Ported lwIP 2.2.0.dev (2022-03-25, branch: master, SHA-1: 124dc0a64ef5d7c14a27e3115e5888df to MCUXpresso SDK.
    - Implemented leaving of multicast groups on ENET and ENET QOS.
- 2.2.0_rev5
  - New features:

    - Ported lwIP 2.2.0.dev (2021-05-11, branch: master, SHA-1: 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_NUM by default. Increase was needed because the actual version of LPC ENET driver always hold ENET_RXBD_NUM number of buffers and few additional buffers are needed for passing zero-copy frame data to lwIP. If this takes too much memory in your application, you can counteract by decreasing PBUF_POOL_SIZE, since PBUF_POOL is used only for transmission when LPC ENET, Kinetis ENET or ENET QOS is used.
- 2.2.0_rev4
  - New features:
    - Ported lwIP 2.2.0.dev (2021-03-05, branch: master, SHA-1: 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: 555812dcec38c9a2ef1ef9b318162915

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.

MOTOR_CONTROL for KSDK

Current version is 1.1.0

- 1.1.0
  - Initial version.

RTCESL for KSDK

Current version is 4.3

- 4.3
  - Initial version.

SAFETY_IEC60730B for KSDK

Current version is 1.1.0

- 1.1.0
  - Initial version.

Host USDHC driver for MCUXpresso SDK

The current driver version is 2.6.3.

- 2.6.3
  - Improvements
    - Added macro SDMMCHOST_SUPPORT_VOLTAGE_CONTROL.
- 2.6.2
  - Bug Fixes
    - Added clock force on during standard tuning to fix the card access not stable after initialization.
- 2.6.1
  - Improvements
    - Increased the delay after enable DAT3 detect card feature to fix the misdetect issue.
- 2.6.0
  - Improvements
    - Removed deprecated api in SDHC host driver.
    - Added SDMMCHOST_ConvertDataToLittleEndian api.
    - Added capability/maxBlockCount/maxBlockSize in host decriptior.
    - Improved the manual tuning flow according to specification.
    - Added mutual exclusive access for function init/deinit/reset/transfer function.
    - Fixed violations of MISRA C-2012 rule 10.1, 10.4, 16.3, 4.7.
- 2.5.3
  - Bug Fixes
    - Corrected the DAT3 detect card flow by PULL down the DAT3 pin firstly and then enable the host DAT3 function.
- 2.5.2

  - Improvements
    - Improved DAT3 card detect mechanism to avoid card false detection.
- 2.5.1
  - Improvements
    - Enabled DAT3 card detect interrupt in function SDMMCHOST_PollingCardDetectStatus to support DAT3 re-detect card.
- 2.5.0
  - Improvements
    - Added cache line size alignment maintain for the read transfer.
    - Added FSL_FEATURE_HAS_L1CACHE to enable cache maintain operation for the soc has LMEM cache.
  - 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 cache maintain functionality in the host driver.
    - Enabled DAT3 card detect feature.
    - Increase the default STD tuning counter to 60 to cover range of the tuning window.
    - Added host instance capability macro.
    - Added clear card inserted/removed event when card removed/inserted interrupt generated.
- 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_SetCardPower.
    - Added internalBuffer in mmc_card_t and removed rawCid/rawCsd/rawExtendedCsd.
    - Added retuning support during data transfer under HS200 mode.
    - Increased the read/write blocks failed retry times for stability.
    - Added delay while retry the CMD1 for stability.
    - Added legacy card support, the card not support CMD6, CMD8.
- 2.2.13
  - Improvements
    - Used the boot mode value instead of boot mode mask value as the parameter of MMC_SetBootConfig to improve user experience.
    - Removed dynamic voltage switch feature for mmc, according to JEDEC standard, the voltage should be fixed after power up.
- 2.2.12
  - Improvement
    - Increased the CMD1 retry times in the MMC card driver to improve driver compatibility.
  - Bug Fixes

    - Fixed the build warning by changing the old style function declaration static status_t inline to static inline status_t(found by adding -Wold-style-declaration in armgcc build flag).
    - Fixed the fall through build warning by adding SUPPRESS_FALL_THROUGH_WARNING() in mmc driver.
- 2.2.7
  - Bug Fixes
    - Fixed MDK 66-D warning.
- 2.2.6
  - Improvements
    - Saved MMC OCR registers while sending CMD1 with argument 0.
  - Bug Fixes
    - Added MMC_PowerOn function in which there is delay function after powerup sdcard. Otherwise, the card initialization by fail.
- 2.2.5
  - Improvements
    - Added SDMMC_ENABLE_SOFTWARE_TUNING to enable/disable software tuning and it is disabled by default.
- 2.2.4
  - Bug Fixes
    - Fixed DDR mode data sequence miss issue, which is caused by NIBBLE_POS.
  - Improvements
    - Increased g_sdmmc 512byte to improve the performance when application use a non-word align data buffer address.
    - Used OCR access mode bits to determine the mmccard high capacity flag.
- 2.2.3
  - Bug Fixes
    - Added response check for send operation condition command. If not checked, the card may occasionally init fail.
- 2.2.1
  - Improvements
    - Improved MMC Boot feature.
- 2.2.0
  - Improvements
    - Optimized tuning/mmc switch voltage/mmc select power class/mmc select timing function.
    - Added strobe dll for mmc HS400 mode.
    - Added write complete wait operation for MMC_Write to fix command timeout issue.
- 2.1.2
  - Improvements
    - Improved SDMMC to support eMMC v5.0.
  - Bug Fixes
    - Fixed incorrect comparison between count and length in MMC_ReadBlocks/MMC_WriteBlocks.
- 2.1.1

  - Bug Fixes
    - Fixed the block range boundary error when transferring data to MMC card.
- 2.1.0
  - Improvements
    - Optimized the function of setting maximum data bus width for MMC card.
- 2.0.0
  - Initial version

SD Card driver for MCUXpresso SDK

The current driver version is 2.4.1.

- 2.4.1
  - Improvements
    - Added macro SDMMCHOST_SUPPORT_VOLTAGE_CONTROL for the host which not support voltage control.
- 2.4.0
  - Improvements
    - Removed deprecated api in sd driver.
    - Added new api SD_PollingCardStatusBusy for application polling card status.
    - Improved the read/write/erase function flow.
    - Improved the signal line voltage switch flow.
    - Added powerOnDelayMS/powerOffDelayMS in sd_usr_param_t to allow redefine the default power on/off delay.
    - Added mutual exclusive access for init/deinit/read/write/erase function.
    - Fixed the driver strength configurations missed when timing mode switch to non SDR50/SDR104 mode.
    - Fixed violations of MISRA C-2012 rule 4.7, 17.7, 10.7, 10.4, 13.5, 14.4.
- 2.3.3
  - Improvements
    - Added host SDR timing mode capability validation during card initialization.
    - Added plling card ready for data status when transfer data failed.
    - Used cache line size align buffer for SD initialization api.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9, 15.7, 4.7, 16.4, 10.1, 10.3, 10.4, 11.3, 14.4, 10.6, 17.7, 16.1, 16.3.
- 2.3.2
  - Improvements
    - Moved power off function after card detect in SD_Init for DAT3 detect card feature.
- 2.3.1
  - Improvements
    - Removed the dead loop while polling DAT0 and CMD13 instead of using timeout mechanism.
- 2.3.0

  - Improvements
    - Marked api SD_HostReset/SD_PowerOnCard/SD_PowerOffCard/SD_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 2.4.1.

- 2.4.1
  - Improvements
    - Added macro SDMMCHOST_SUPPORT_VOLTAGE_CONTROL for the host which not support voltage control.
- 2.4.0
  - Improvements
    - Removed deprecated api in sdio driver.
    - Improved the signal line voltage switch flow.

    - Added powerOnDelayMS/powerOffDelayMS in sdio_usr_param_t to allow redefine the default power on/off delay.
    - Added mutual exclusive access for init/deinit/direct/extend function.
    - Fixed violations of MISRA C-2012 rule 4.7, 17.7, 10.1, 12.2.
- 2.3.3
  - Bug Fixes
    - Fixed logical dead code coverity issue.
  - Improvements
    - Removed deprecated api in sdio driver.
- 2.3.2
  - Improvements
    - Added host SDR timing mode capability validation during card initialization.
    - Used cache line size align buffer for SDIO initialization api.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9, 15.7, 4.7, 16.4, 10.1, 10.3, 10.4, 11.3, 14.4, 10.6, 17.7, 16.1, 16.3.
- 2.3.1
  - Improvements
    - Moved power off function after card detect in SD_Init for DAT3 detect card feature.
- 2.3.0
  - Improvements
    - Marked api SDIO_HostReset/SDIO_PowerOnCard/SDIO_PowerOffCard/SDIO_WaitCardDetectStatus as deprecated.
    - Added new api SDIO_SetCardPower/SDIO_PollingCardDetectStatus/SDIO_HostDoReset.
    - Added internalBuffer in sdio_card_t for card register content extract and improve the data access efficiency.
    - Added retry function after switch to target timing failed in SDIO_SelectBusTiming.
    - Changed defalut bus clock from 400KHZ to 25MHZ.
- 2.2.13
  - Improvements
    - Removed the sdio card interrupt from sdio host initialization, since the card interrupt enablement should be determined by application.
  - Bug Fixes
    - Fixed Out-of-bounds write Coverity issue.
- 2.2.12
  - Improvements
    - Added manual tuning function for looking for the tuning window automatically.
    - Fixed the build warning by changing the old style function declaration static status_t inline to static inline status_t(found by adding -Wold-style-declaration in armgcc build flag).
    - Fixed the fall through build warning by adding SUPPRESS_FALL_THROUGH_WARNING() in sdio driver.
- 2.2.11
  - Bug Fixes

    - Added check card async interrupt capability in function SDIO_GetCardCapability.
    - Fixed OUT OF BOUNDS access in function SDIO_IO_Transfer.
- 2.2.10
  - Bug Fixes
    - Fixed SDIO card driver get an incorrect io number when the card io number is bigger than 2.
  - Improvements
    - Added SDIO 3.0 support.
    - Added API SDIO_IO_RW_Direct for direct read/write card register access.
- 2.2.9
  - Improvements
    - Added API SDIO_SetIOIRQHandler/SDIO_HandlePendingIOInterrupt to handle multi io pending IRQ.
- 2.2.8
  - Improvements
    - Updated sdmmc to support SDIO interrupt.
    - Added API SDIO_GetPendingInterrupt to get the pending io interrupt.
- 2.2.7
  - Bug Fixes
    - Fixed MDK 66-D warning.
- 2.2.6
  - Improvements
    - Added an unify transfer interface for SDIO.
  - Bug Fixes
    - Fixed Wrong pointer address used by SDMMCHOST_Init.
- 2.1.5
  - Improvements
    - Improved SDIO card init sequence and add retry option for SDIO_SwitchToHighSpeed function.
- 2.1.4
  - Improvements
    - Added Go_Idle function for SDIO card.
- 2.0.0
  - Initial version.

USB stack for MCUXpresso SDK

The current version of USB stack is 2.8.4.

- 2.8.4
  - Improvement:
    - Add the new netc adatper for the new netc driver.
    - Fix issues for USB device dfu and usb device msc when enable the macro USB_DEVICE_CONFIG_RETURN_VALUE_CHECK.

    - Change the header file including order for usb.h header.
    - Update the USB host audio class driver to fix the wrong output log.
    - Add the workaround on dev_hid_mouse_bm case for the errata TN00071.
    - Enable ROOT2 macro in USB device stack.
    - Use an unified definiton for the base address of RTxxxx platforms.
- 2.8.3
  - Improvement:
    - Update the EHCI controller driver to support the address convert for TCM.
    - Update the USB host EHCI controller driver to make sure the mutual exclusion access under multiple tasks' environment.
- 2.8.2
  - Improvement:
    - Fix noise issue of UAC 3.1, UAC 5.1, UAC 7.1 on usb audio speaker demo.
    - Fix the issue that incorrect PC behavior when ejecting USB MSC devices.
    - Update the EHCI controller driver to support RW610 that does not reply on PHY driver, especially for low power feature.
    - Update the USB_HostHelperParseAlternateSetting to fix the wrong interface parse.
    - Update dev_composite_hid_audio_unified_bm demo to suppport independent mute/unmute and volume control.
- 2.8.1
  - Improvement:
    - update USB audio demos to use audio component (components).
    - Add the checking of function call return value.
    - Add audio multiple channels demo (usb_device_composite_audio_multi_ch_unified) on RT600 audio board.
    - Fix audio noise on sync mode and improve overflow/underflow checking method.
    - Support UAC 3.1, 5.1 and 7.1 on audio speaker demo.
    - Set USB device CDC demo not to depend on DTR setting from host.
    - Support MCUX toolchain on some RTxxxx platforms.
- 2.8.0
  - Improvement:
    - Fix the USB device stack vulnerability issues.
    - Update the audio PLL and FRO adjustment codes for audio examples in RTxxx, LPC54xxx and LPC55xxx.
    - Improve the USB PD AMS collision avoidance.
    - Improve IP3511 controller driver's dedicated ram allocation.
    - Change the USB_DATA_ALIGN_SIZE to 4 because the controller driver uses the dedicated RAM to do memcpy.
  - New features:
    - Enable USB host audio recorder demo for mutilple boards.
- 2.7.0
  - Improvement:
    - Use new feeback solution and low latency playback for usb device speaker demo and unified demos. Add underflow and overflow protection.
    - Optimize hard code for usb audio demos.

    - Update Unconstrained Power field in the Sink Capabilities Message according to the external power state.
    - Fix CVE-2021-38258 and CVE-2021-38260
  - New features:
    - Enable USB host video demo for mutilple boards.
    - Enable USB device MTP demo for mutilple boards.
    - Add PPS message to usb pd stack.
- 2.6.1
  - Improvement:
    - rename sdcard as disk for all of sdcard demos. For ramdisk demos, they are not changed.
    - add wrapper for all of disk demos to support emmc.
- 2.6.0
  - Improvement:
    - Added more ufi event to support dynamic sdcard capacity.
    - Passed MISRA-2012 mandatory and required rules.
      - Except rule 17.2 in host hub and otg stack.
      - Except rule 5.1, rule 5.4, rule 21.1 and rule 21.2.
    - Re-implemented USB components and supported NPW.
    - Improved IP3511 controller driver's cancelling transfer function.
    - Enabled the audio2.0 defaultly for device audio demos.
    - Enabled the host audio2.0 function in host audio class driver and host audio speaker demo.
  - New features:
    - enable two USB controllers in one USB host mouse demo which named as host_hid_mouse_dual.
    - enable UAC 5.1 for usb device audio speaker demo.
- 2.5.0
  - Improvement:
    - Integrated sdk components (OSA, Timer, GPIO and serial_manager) to USB stack and demos.
    - Improved the ip3511 driver throughput.
    - Improved audio initialization codes after SDK audio drivers update.
    - Improved auido to support the audio2.0 in win10.
    - Add one "enumeration fail" callback event to host stack.
- 2.4.2
  - Improvement:
    - Put the USB controller data and transfer buffer to noncache section, removed the setting that sets the whole ocram and sdram as noncached.
    - Separated composite audio examples' channel,sample rate,format parameters from commom macro to in dedicated macro and out dedicated macro.
    - replaced USB_PrepareData with USB_AudioRecorderGetBuffer.
- 2.4.1
  - New features:
    - Added enumeration fail callback to host stack when the attached device's enumeration failed.

- 2.4.0
  - Improvement:
    - Device Charger Detection (DCD) software architecture was refactored.
  - New features:
    - Enabled Device Charger Detection (DCD) on RT1060.
    - Enabled Device Charger Detection on RT600.
    - Enabled host battery charger function on RT600.
- 2.3.0
  - New features:
    - Added host video camera support. example: usb_host_video_camera
    - Added a new device example. example: usb_device_composite_cdc_hid_audio_unified
- 2.2.0
  - New features:
    - Added device DFU support.
    - Supported OM13790DOCK on LPCXpresso54018.
    - Added multiple logical unit support in msc class driver, updated usb_device_lba_information_struct_t to support this.
    - Supported multiple transfers for host ISO on IP3516HS.
  - Bug fixes:
    - Fixed device ip3511 prime data length than maxpacket size issue.
    - Initialized interval attribute in usb_device_endpoint_struct_t/usb_device_endpoint_init_struct_t.
    - Removed unnecessary header file in device CDC class driver, removed unnecessary usb_echo, and added DEBUG macro for necessary usb_echo in device CDC class driver.
    - Fixed device IP3511HS unfinished interrupt transfer missing issue.
- 2.1.0
  - New features:
    - Added host RNDIS support. example: lwip_dhcp_usb
    - Enabled USB 3.0 support on device stack.
    - Power Delivery feature: Added OM13790HOST support; Added auto policy feature; Printed e-marked cable information;
- 2.0.1
  - Bug fixes:
    - Fixed some USB issues: Fixed MSC CV test failed in MSC examples.
    - Changed audio codec interfaces.
- 2.0.0
  - New features:
    - PTN5110N support.
  - Bug fix:
    - Added some comments, fixed some minor USB issues.
- 1.9.0
  - New features:
    - Examples:
      - usb_pd_alt_mode_dp_host
- 1.8.2

  - Updated license.
- 1.8.1
  - Bug fix:
    - Verified some hardware issues, support aruba_flashless.
- 1.8.0
  - New features:
    - Examples:
      - usb_device_composite_cdc_vcom_cdc_vcom
      - usb_device_composite_hid_audio_unified
      - usb_pd_sink_battery
      - Changed usb_pd_battery to usb_pd_charger_battery.
  - Bug fix:
    - Code clean up, removed some irrelevant code.
- 1.7.0
  - New features:
    - USB PD stack support.
  - Examples:
    - usb_pd
    - usb_pd_battery
    - usb_pd_source_charger
- 1.6.3
  - Bug fix: -IP3511_HS driver control transfer sequence issue, enabled 3511 ip cv test.
- 1.6.2
  - New features:
    - Multi instance support.
- 1.6.1
  - New features:
  - Changed the struct variable address method for device_video_virtual_camera and host_phdc-_manager.
- 1.6.0
  - New features:
    - Supported Device Charger Detect feature on usb_device_hid_mouse.
- 1.5.0
  - New features:
    - Supported controllers
      - OHCI (Full Speed, Host mode)
      - IP3516 (High Speed, Host mode)
      - IP3511 (High Speed, Device mode)
    - Examples:
      - usb_lpm_device_hid_mouse
      - usb_lpm_device_hid_mouse_lite
      - usb_lpm_host_hid_mouse
- 1.4.0
  - New features:
    - Examples:

        - usb_device_hid_mouse/freertos_static
        - usb_suspend_resume_device_hid_mouse_lite
- 1.3.0
  - New features:
    - Supported roles
      - OTG
    - Supported classes
      - CDC RNDIS
    - Examples
      - usb_otg_hid_mouse
      - usb_device_cdc_vnic
      - usb_suspend_resume_device_hid_mouse
      - usb_suspend_resume_host_hid_mouse
- 1.2.0
  - New features:
    - Supported controllers
      - LPC IP3511 (Full Speed, Device mode)
- 1.1.0
  - Bug fix:
    - Fixed some issues in USB certification.
    - Changed VID and Manufacturer string to NXP.
  - New features:
    - Supported classes
      - Pinter
    - Examples:
      - usb_device_composite_cdc_msc_sdcard
      - usb_device_printer_virtual_plain_text
      - usb_host_printer_plain_text
- 1.0.1
  - Bug fix:
    - Improved the efficiency of device audio speaker by changing the transfer mode from interrupt to DMA, thus providing the ability to eliminate the periodic noise.
- 1.0.0
  - New features:
    - Supported roles
      - Device
      - Host
    - Supported controllers:
      - KHCI (Full Speed)
      - EHCI (High Speed)
    - Supported classes:
      - AUDIO
      - CCID
      - CDC
      - HID

  - MSC
  - PHDC
  - VIDEO
- Examples:
  - usb_device_audio_generator
  - usb_device_audio_speaker
  - usb_device_ccid_smart_card
  - usb_device_cdc_vcom
  - usb_device_cdc_vnic
  - usb_device_composite_cdc_msc
  - usb_device_composite_hid_audio
  - usb_device_composite_hid_mouse_hid_keyboard
  - usb_device_hid_generic
  - usb_device_hid_mouse
  - usb_device_msc_ramdisk
  - usb_device_msc_sdcard
  - usb_device_phdc_weighscale
  - usb_device_video_flexio_ov7670
  - usb_device_video_virtual_camera
  - usb_host_audio_speaker
  - usb_host_cdc
  - usb_host_hid_generic
  - usb_host_hid_mouse
  - usb_host_hid_mouse_keyboard
  - usb_host_msd_command
  - usb_host_msd_fatfs
  - usb_host_phdc_manager
  - usb_keyboard2mouse
  - usb_pin_detect_hid_mouse

3 Component Change Log

CODEC

The current codec common driver version is 2.3.1.

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

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

WM8904

The current wm8904 driver version is 2.5.1.

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

- 2.4.0
  - New features
    - Added fll support in wm8904 driver.
- 2.3.0
  - Improvements
    - Added new API WM8904_SetMasterClock to support BCLK/LRCLK output mode.
- 2.1.0
  - new APIs
    - WM8904_ReadRegister
    - WM8904_WriteRegister
    - WM8904_ModifyRegister
    - WM8904_SetRecord
    - WM8904_SetPlay
    - WM8904_SetRecordChannel
    - WM8904_SetModulePower
    - WM8904_SetChannelVolume
    - WM8904_SetChannelMute

New features

- Removed dependency on codec common driver.
- Added dependency on codec i2c.

Bug Fixes

- Fixed unchecked return value in WM8904_Deinit.
- Fixed the alignment fault issue by adding __NOP between continuous memory access.

2.0.3

- Bug Fixes
  - Fixed issue that wm8904 register access function truncated return value.

2.0.2

- Bug Fixes
  - Fixed using uninitialized value format.fsRatio when calling WM8904_UpdateFormate.

2.0.1

- Added WM8904_CheckAudioFormat API.
- Changed the second parameter's name of WM8904_SetAudioFormat to sysclk.

2.0.0

- Initial version.

.1 WM8960

The current wm8960 driver version is 2.2.1.

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

SGTL5000

The current sgtl5000 driver version is 2.1.1.

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

DA7212

The current da7212 driver version is 2.2.2.

- 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

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support