MCUXpresso SDK Release Notes Supporting frdmk64f

Change Logs

Contents

Contents

Contents

Driver Change Log

CLOCK

Current CLOCK driver version is 2.5.1

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

DSPI_CMSIS

Current dspi_cmsis driver version is 2.2

- 2.2

  - Bug Fixes
    - Fixed the bug that, the parameter num of APIs ARM_SPI_Transfer, ARM_SPI_Send and ARM_SPI_Receive, and the return value of API ARM_SPI_GetDataCount should be the number of data item defined by datawidth, rather than the number of byte.
- 2.1
  - Bug Fixes
    - Fixed the wrong clock polarity assignment in driver. For ARM_SPI_CPOL0_CPHA0 and other frame format parameters, CPOL = 0 means kSPI_ClockPolarityActiveHigh not kSPI_ClockPolarityActiveLow in driver.
  - New Features
    - Allowed user to set up the default transmit value by using ARM_SPI_SET_DEFAULT_-TX_VALUE.
    - Enabled slave select mode. Note this has no effect when user sets any of them because the driver can only support the hardware control function.
- 2.0
  - Initial version.

ENET

Current ENET CMSIS driver version is 2.1

- 2.1
  - Bug Fixes
    - Fixed the wrong logic to control cache macro.
- 2.0
  - Initial version.

I2C

Current I2C CMSIS driver version is 2.0.1

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

UART

The current UART CMSIS driver version is 2.0.0.

- 2.0

  - Initial version.

ADC16

The current ADC16 driver version is 2.1.0.

- 2.1.0
  - New Features:
    - Supported KM series' new ADC reference voltage source, bandgap from PMC.
- 2.0.3
  - Bug Fixes
    - Fixed IAR warning Pa082: the order of volatile access should be defined.
- 2.0.2
  - Improvements
    - Used conversion control feature macro instead of that in IO map.
- 2.0.1
  - Bug Fixes
    - Fixed MISRA-2012 rules.
      - Rule 16.4, 10.1, 13.2, 14.4 and 17.7.
- 2.0.0
  - Initial version

CMP

The current CMP driver version is 2.0.2.

- 2.0.2
  - Bug Fixes
    - Fixed the violations of MISRA 2012 rules:
      - Rule 10.3
- 2.0.1
  - Bug Fixes
    - Fixed MISRA-2012 rules.
      - Rule 14.4, rule 10.3, rule 10.1, rule 10.4 and rule 17.7.
- 2.0.0
  - Initial version.

CMT

The current CMT driver version is 2.0.3.

- 2.0.3
  - Bug Fixes
    - Fixed MISRA C-2012 rule 15.5.

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

COMMON

The current COMMON driver version is 2.2.4.

- 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 let result be more close to actual delay time.
- 2.2.2
  - New Features
    - Added include RTE_Components.h for CMSIS pack RTE.
- 2.2.1
  - Bug Fixes
    - Fixed violation of MISRA C-2012 Rule 3.1, 10.1, 10.3, 10.4, 11.6, 11.9.
- 2.2.0
  - New Features
    - Moved SDK_DelayAtLeastUs function from clock driver to common driver.
- 2.1.4
  - New Features
    - Added OTFAD into status group.
- 2.1.3
  - Bug Fixes
    - MISRA C-2012 issue fixed.
      - Fixed the rule: rule-10.3.
- 2.1.2
  - Improvements
    - Add SUPPRESS_FALL_THROUGH_WARNING() macro for the usage of suppressing fallthrough warning.
- 2.1.1

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

CRC

The current CRC driver version is 2.0.2.

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

DAC

The current DAC driver version is 2.0.2.

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

users can enable the DAC's output.
- 2.0.0
  - Initial version.

DMAMUX

The current DMAMUX driver version is 2.0.4.

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

DSPI

The current dspi driver version is 2.2.4.

- 2.2.4
  - Bug Fixes
    - Fixed bug that instance with shared TR/RX EDMA request cannot transfer 1 datawidth of data as master in single transfer.
- 2.2.3
  - Improvements
    - Added macro of getting maximum transfer size using EDMA way.
- 2.2.2
  - Bug Fixes
    - MISRA C-2012 issue fixed:
      - Fixed rules, containing: 10.8, 10.3.
    - Fixed the build warning issue.
    - Fixed the bug that PCS would temporarily pull down for a while during master initialization.
    - Fixed compiling error of undefined identifier g_dspiDummyData.

    - Changed the type of kDSPI_AllStatusFlag and kDSPI_AllInterruptEnable from int to uint32_t to fix Pe068 warning of integer conversion.
    - Eliminated IAR Pa082 warning in DSPI_MasterTransferDMA.
- 2.2.1
  - Bug Fixes
    - Fixed the bug for double execution of transfer complete callback in master interrupt transfer mode. In the interrupt routine, the DSPI interrupt may drop in the situation of the interrupt pending by itself while receiving the last frame, adding check to the transfer state to execute the callback function.
    - Fixed wrong logic in DSPI_SetFifoEnable().
    - MISRA C-2012 issue fixed.
      - Fixed rules, containing: rule-12.1, rule-17.7, rule-16.4, rule-14.4, rule-10.4, rule-10.8, rule-10.3, rule-10.1, rule-10.6, rule-13.5, rule-11.3, rule-13.2, rule-8.3, and rule-8.5.
- 2.2.0
  - New Features
    - Added gasket feature for SPI EDMA driver, which reduces one channel used in the EDMA master transfer. With this feature support, only two channels are needed. For example, if the gasket feature is supported, we could use the DSPI_MasterTransferCreateHandleEDMA function like below: DSPI_MasterTransferCreateHandleEDMA(EXAMPLE_DSPI_MASTER_BASEADDR, &g_dspi_edma_m_handle, DSPI_MasterUserCallback, &userData, &dspiEdmaMasterRxRegToRxDataHandle, NULL, &dspiEdmaMasterIntermediaryToTxRegHandle);
    - Added dummy data setup API to allow users to configure the dummy data to be transferred.
    - Added new APIs for half-duplex transfer function. Users can send and receive data by one API in the polling/interrupt/EDMA way, and they can choose either to transmit first or to receive first. Additionally, the PCS pin can be configured as assert status in transmission (between transmit and receive) by setting the isPcsAssertInTransfer to true.
- 2.1.4
  - Bug Fixes
    - DSPI EDMA driver: the DSPI instance has been separated, so the DMA request source can now transfer up to 32767 Bytes data in one DSPI_MasterTransferEDMA() transfer.
- 2.1.3
  - Bug Fixes
    - DSPI EDMA driver can no longer support the case that the transfer data size is odd, but the bitsPerFrame is greater than 8.
  - Improvements
    - Added #ifndef/#endif to allow users to change the default TX value at compile time.
- 2.1.2
  - Bug Fixes
    - DSPI_MasterTransferBlocking function would hang in some corner cases (for example, some cases with bitsPerFrame is 4,6 and kDSPI_MasterPcsContinuous transfer mode).
- 2.1.1
  - Bug Fixes
    - Set the EOQ (End Of Queue) bit to TRUE for the last transfer in transactional APIs.

- 2.1.0
  - New Features
    - Added Transfer prefix in transactional APIs.
- 2.0.0
  - Initial version.

EDMA

The current eDMA driver version is 2.3.2.

- 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 2.3.0.

- 2.3.0
  - Bug Fixes
    - Fixed the issue that clause 45 MDIO read/write API doesn't check the transmission over status between two transmissions.
    - Fixed violations of the MISRA C-2012 rules 2.2,10.3,10.4,10.7,11.6,11.8,13.5,14.4,15.-7,17.7.
  - New Features
    - Added APIs to support send/receive frame with Zero-Copy.
  - Improvements
    - Separated the clock configuration from module configuration when init and deinit.
    - Added functions to set second level interrupt handler.
    - Provided new function to get 1588 timer count without disabling interrupt.
    - Improved timestamp controlling, deleted all old timestamp management APIs and data structures.
    - Merged the single/multiple ring(s) APIs, now these APIs can handle both.
    - Used base and index to control buffer descriptor, aligned with qos and lpc enet driver.
- 2.2.6
  - Bug Fixes
    - Updated MII speed formula referring to the manual.
- 2.2.5
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.1, 10.3, 10.4, 10.6, 10.7, 11.6, 11.9, 13.5, 14.4, 16.4, 17.7, 21.15, 3.1, 8.4.
    - Changed to use ARRAY_SIZE(s_enetBases) as the array size for s_ENETHandle, fixed the hardfault issue for using some ENET instance when ARRAY_SIZE(s_enetBases) is not same as FSL_FEATURE_SOC_ENET_COUNT.

- 2.2.4
  - Improvements
    * Added call to Data Synchronization Barrier instruction before activating Tx/Rx buffer descriptor to ensure previous data update is completed.
    * Improved ENET_TransmitIRQHandler to store timestamps for multiple transmit buffer descriptors.
    * Bug Fixes
    * Fixed the issue that ENET_Ptp1588GetTimer did not handle the timer wrap situation.
- 2.2.3
  - Improvements
    * Improved data buffer cache maintenance in the ENET driver.
- 2.2.2
  - New Features
    * Added APIs for extended multi-ring support.
    * Added the AVB configure API for extended AVB feature support.
- 2.2.1
  - Improvements
    * Changed the input data pointer attribute to const in ENET_SendFrame().
- 2.1.1
  - New Features
    * Added the extended MDIO IEEE802.3 Clause 45 MDIO format SMI command APIs.
    * Added the extended interrupt coalescing feature.
  - Improvements
    * Combined all storage operations in the ENET_Init to ENET_SetHandler API.
- 2.0.1
  - Bug Fixes
    * Used direct transmit busy check when doing data transmit.
  - Miscellaneous Changes
    * Updated IRQ handler work flow.
    * Changed the TX/RX interrupt macro from kENET_RxByteInterrupt to kENET_RxBuffer-Interrupt, from kENET_TxByteInterrupt to kENET_TxBufferInterrupt.
    * Deleted unnecessary parameters in ENET handler.
- 2.0.0
  - Initial version.

EWM

The current EWM driver version is 2.0.3.

- 2.0.3
  - Bug Fixes
    * Fixed violation of MISRA C-2012 rules: 10.1, 10.3.
- 2.0.2
  - Bug Fixes

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

FLASH

Current FLASH driver version is 3.0.2.

- 3.0.2
  - Bug Fixes — MISRA C-2012 issue fixed: rule 8.4, 17.7, 10.4, 16.1, 21.15, 11.3, 10.7 — building warning -Wnull-dereference on arm compiler v6
- 3.0.1
  - New Features
    - Added support FlexNVM alias for (kw37/38/39).
- 3.0.0
  - Improvements
    - Reorganized FTFx flash driver source file.
    - Extracted flash cache driver from FTFx driver.
    - Extracted flexnvm flash driver from FTFx driver.
- 2.3.1
  - Bug Fixes
    - Unified Flash IFR design from K3.
    - New encoding rule for K3 flash size.
- 2.3.0
  - New Features
    - Added support for device with LP flash (K3S/G).
    - Added flash prefetch speculation APIs.
  - Improvements
    - Refined flash_cache_clear function.
    - Reorganized the member of flash_config_t struct.
- 2.2.0
  - New Features
    - Supported FTFL device in FLASH_Swap API.
    - Supported various pflash start addresses.
    - Added support for KV58 in cache clear function.
    - Added support for device with secondary flash (KW40).
  - Bug Fixes
    - Compiled execute-in-ram functions as PIC binary code for driver use.
    - Added missed flexram properties.
    - Fixed unaligned variable issue for execute-in-ram function code array.
- 2.1.0

  - Improvements
    - Updated coding style to align with KSDK 2.0.
    - Different-alignment-size support for pflash and flexnvm.
    - Improved the implementation of execute-in-ram functions.
- 2.0.0
  - Initial version

FLEXCAN

The current FLEXCAN driver version is 2.5.2.

- 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 F-

LEXCAN_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.

FTM

The current FTM driver version is 2.2.3.

- 2.2.3
    - Bug Fixes
        * MISRA C-2012 issue fixed: rule 14.4 and 17.7.
- 2.2.2
    - Bug Fixes
        * Fixed the issue that when FTM instance has only TPM features cannot be initialized by FTM_Init() function. By added function macro to assert FTM is TPM only instance.
- 2.2.1
    - Bug Fixes
        * MISRA C-2012 issue fixed: rule 10.1, 10.3, 10.4, 10.6, 10.7 and 11.9.
- 2.2.0
    - Bug Fixes
        * Fixed the issue of comparison between signed and unsigned integer expressions.
    - Improvements
        * Added support of complementary mode in FTM_SetupPWM() and FTM_SetupPwmMode() APIs.
        * Added new parameter "enableDeadtime" in structure ftm_chnl_pwm_signal_param_t.
- 2.1.1
    - Bug Fixes
        * Fixed COVERITY integer handing issue where the right operand of a left bit shift statement should not be a negative value. This appears in FTM_SetReloadPoints().
- 2.1.0
    - Improvements
        * Added a new API FTM_SetupPwmMode() to allow the user to set the channel match value in units of timer ticks. New configure structure called ftm_chnl_pwm_config_param_t was added to configure the channel's PWM parameters. This API is similar with FTM_SetupPwm() API, but the new API will not set the timer period(MOD value), it will be useful for users to set the PWM parameters without changing the timer period.
    - Bug Fixes
        * Added feature macro to enable/disable the external trigger source configuration.
- 2.0.4

  - Improvements
    - Added a new API to enable DMA transfer:
      - FTM_EnableDmaTransfer()
- 2.0.3
  - Bug Fixes
    - Updated the FTM driver to enable fault input after configuring polarity.
- 2.0.2
  - Improvements
    - Added support to Quad Decoder feature with new APIs:
      - FTM_GetQuadDecoderFlags()
      - FTM_SetQuadDecoderModuloValue()
      - FTM_GetQuadDecoderCounterValue()
      - FTM_ClearQuadDecoderCounterValue()
- 2.0.1
  - Bug Fixes
    - Updated the FTM driver to fix write to ELSA and ELSB bits.
    - FTM combine mode: set the COMBINE bit before writing to CnV register.
- 2.0.0
  - Initial version.

GPIO

The current driver version is 2.5.1.

- 2.5.1
  - Bug Fixes
    - Fixed wrong macro definition.
    - Fixed MISRA C-2012 rule issues in the FGPIO_CheckAttributeBytes() function.
    - Defined the new macro to separate the scene when the width of registers is different.
    - Removed some redundant macros.
  - New Features
    - Added some APIs to get/clear the interrupt status flag when the port doesn't control pins' interrupt.
- 2.4.1
  - Improvements
    - Improved GPIO_CheckAttributeBytes() function to support 8 bits width GACR register.
- 2.4.0
  - Improvements
    - API interface added:
      - New APIs were added to configure the GPIO interrupt clear settings.
- 2.3.2
  - Bug Fixes
    - Fixed the issue for MISRA-2012 check.
      - Fixed rule 3.1, 10.1, 8.6, 10.6, and 10.3.

- 2.3.1
  - Improvements
    - Removed deprecated APIs.
- 2.3.0
  - New Features
    - Updated the driver code to adapt the case of interrupt configurations in GPIO module. New APIs were added to configure the GPIO interrupt settings if the module has this feature on it.
- 2.2.1
  - Improvements
    - API interface changes:
      - Refined naming of APIs while keeping all original APIs by marking them as deprecated. The original APIs will be removed in next release. The main change is updating APIs with prefix of _PinXXX() and _PortXXX.
- 2.1.1
  - Improvements
    - API interface changes:
      - Added an API for the check attribute bytes.
- 2.1.0
  - Improvements
    - API interface changes:
      - Added "pins" or "pin" to some APIs' names.
      - Renamed "_PinConfigure" to "GPIO_PinInit".

I2C

The current I2C driver version is 2.0.8.

- 2.0.8
  - Bug Fixes
    - Fixed the bug that DFEN bit of I2C Status register 2 could not be set in I2C_MasterInit.
    - MISRA C-2012 issue fixed: rule 14.2, 15.7, and 16.4.
    - Eliminated IAR Pa082 warnings from I2C_MasterTransferDMA and I2C_MasterTransferCallbackDMA by assigning volatile variables to local variables and using local variables instead.
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 11.9, 14.4, 15.7, 17.7.
  - Improvements
    - Improved timeout mechanism when waiting certain state in transfer API.
    - Updated the I2C_WAIT_TIMEOUT macro to unified name I2C_RETRY_TIMES.
    - Moved the master manually acknowledge byte operation into static function I2C_MasterAckByte.
    - Fixed control/status clean flow issue inside I2C_MasterReadBlocking to avoid potential issue that pending status is cleaned before it's proceeded.

- 2.0.7
  - Bug Fixes
    - Fixed the issue for MISRA-2012 check.
      - Fixed rule 11.9 ,15.7 ,14.4 ,10.4 ,10.8 ,10.3, 10.1, 10.6, 13.5, 11.3, 13.2, 17.7, 5.7, 8.3, 8.5, 11.1, 16.1.
    - Fixed Coverity issue of unchecked return value in I2C_RTOS_Transfer.
    - Fixed variable redefine issue by moving i2cBases from fsl_i2c.h to fsl_i2c.c.
  - Improvements
    - Added I2C_MASTER_FACK_CONTROL macro to enable FACK control for master transfer receive flow with IP supporting double buffer, then master could hold the SCL by manually setting TX AK/NAK during data transfer.
- 2.0.6
  - Bug Fixes
    - Fixed the issue that I2C Master transfer APIs(blocking/non-blocking) did not support the situation of master transfer with subaddress and transfer data size being zero, which means no data followed by the subaddress.
- 2.0.5
  - Improvements
    - Added I2C_WATI_TIMEOUT macro to allow the user to specify the timeout times for waiting flags in functional API and blocking transfer API.
- 2.0.4
  - Bug Fixes
    - Added a proper handle for transfer config flag kI2C_TransferNoStartFlag to support transmit with kI2C_TransferNoStartFlag flag. Support write only or write+read with no start flag; does not support read only with no start flag.
- 2.0.3
  - Bug Fixes
    - Removed enableHighDrive member in the master/slave configuration structure because the operation to HDRS bit is useless, the user need to use DSE bit in port register to configure the high drive capability.
    - Added register reset operation in I2C_MasterInit and I2C_SlaveInit APIs. Fixed issue where I2C could not switch between master and slave mode.
    - Improved slave IRQ handler to handle the corner case that stop flag and address match flag come synchronously.
- 2.0.2
  - Bug Fixes
    - Fixed issue in master receive and slave transmit mode with no stop flag. The master could not succeed to start next transfer because the master could not send out re-start signal.
    - Fixed the out-of-order issue of data transfer due to memory barrier.
    - Added hold time configuration for slave. By leaving the SCL divider and MULT reset values when configured to slave mode, the setup and hold time of the slave is then reduced outside of spec for lower baudrates. This can cause intermittent arbitration loss on the master side.
  - New Features

        * Added address nak event for master.
        * Added general call event for slave.
- 2.0.1
  - New Features
    * Added double buffer enable configuration for SoCs which have the DFEN bit in S2 register.
    * Added flexible transmit/receive buffer size support in I2C_SlaveHandleIRQ.
    * Added start flag clear, address match, and release bus operation in I2C_SlaveWrite/-ReadBlocking API.
  - Bug Fixes
    * Changed the kI2C_SlaveRepeatedStartEvent to kI2C_SlaveStartEvent.
- 2.0.0
  - Initial version.

LLWU

The current LLWU driver version is 2.0.5.

- 2.0.5
  - Bug Fixes
    * Fixed violations of the MISRA C-2012 rules 10.3.
    * Fixed the issue that function LLWU_SetExternalWakeupPinMode() does not work on 32-bit width platforms.
- 2.0.4
  - Bug Fixes
    * Fixed violations of the MISRA C-2012 rules 10.3, 10.4, 10.6, 10.7, 11.3.
    * Fixed issue that LLWU_ClearExternalWakeupPinFlag may clear other filter flags by mistake on platforms with 32-bit LLWU registers.
- 2.0.3
  - Bug Fixes
    * Fixed MISRA-2012 rules.
      · Rule 16.4.
- 2.0.2
  - Improvements
    * Corrected driver function LLWU_SetResetPinMode parameter name.
  - Bug Fixes
    * Fixed MISRA-2012 rules.
      · Rule 14.4, 10.8, 10.4, 10.3.
- 2.0.1
  - Other Changes
    * Updates for KL8x.
- 2.0.0
  - Initial version.

LPTMR

The current LPTMR driver version is 2.1.1.

- 2.1.1
  - Improvements
    - Updated the characters from "PTMR" to "LPTMR" in "FSL_FEATURE_PTMR_HAS_NO_PRESCALER_CLOCK_SOURCE_1_SUPPORT" feature definition.
- 2.1.0
  - Improvements
    - Implement for some special devices' not supporting for all clock sources.
  - Bug Fixes
    - Fixed issue when accessing CMR register.
- 2.0.2
  - Bug Fixes
    - Fixed MISRA-2012 issues.
      - Rule 10.1.
- 2.0.1
  - Improvements
    - Updated the LPTMR driver to support 32-bit CNR and CMR registers in some devices.
- 2.0.0
  - Initial version.

PDB

The current PDB driver version is 2.0.4.

- 2.0.4
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.1 and 10.4.
- 2.0.3
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 17.7.
- 2.0.2
  - Improvement:
    - Used macros in feature file instead of that in IO map.
- 2.0.1
  - Changed PDB register base array to const.
- 2.0.0
  - Initial version.

PIT

The current PIT driver version is 2.0.3.

- 2.0.3
  - Bug Fixes
    - Clear all status bits for all channels to make sure the status of all TCTRL registers is clean.
- 2.0.2
  - Bug Fixes
    - Fixed MISRA-2012 issues.
      - Rule 10.1.
- 2.0.1
  - Bug Fixes
    - Cleared timer enable bit for all channels in function PIT_Init() to make sure all channels stay in disable status before setting other configurations.
    - Fixed MISRA-2012 rules.
      - Rule 14.4, rule 10.4.
- 2.0.0
  - Initial version.

PMC

The current PMC driver version is 2.0.3.

- 2.0.3
  - Bug Fixes
    - Fixed the violation of MISRA C-2012 rule 11.3.
- 2.0.2
  - Bug Fixes
    - Fixed the violations of MISRA 2012 rules:
      - Rule 10.3.
- 2.0.1
  - Bug Fixes
    - Fixed MISRA issues.
      - Rule 10.8, Rule 10.3.
- 2.0.0
  - Initial version.

PORT

The current PORT driver version is 2.1.0.

- 2.1.0
  - New Features
    - Updated the driver code to adapt the case of the interrupt configurations in GPIO module. Will move the pin configuration APIs to GPIO module.
- 2.0.2

  - Other Changes
    - Added feature guard macros in the driver.
- 2.0.1
  - Other Changes
    - Added "const" in function parameter.
    - Updated some enumeration variables' names.

RCM

The current RCM driver version is 2.0.4.

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

RTC

The current RTC driver version is 2.2.1.

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

  - Improvements
    - Added feature macro check for many features.
- 2.0.0
  - Initial version.

SAI

The current SAI driver version is 2.3.1

- 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.

SDHC

The current SDHC driver version is 2.1.11.

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

- 2.1.3
  - Improvements
    - Modified some definitions to be compatible with middleware adapter.
- 2.1.2
  - Bug Fixes
    - Used function pointer for interrupt handler to reduce code size.
    - Bad status bit check behavior when waiting for initialization of an SD card.
    - Added support NON-WORD aligned data size transfer mode for SDIO card.
- 2.1.1
  - Bug Fixes
    - Fixed the compile error when ADMA1 is enabled.
- 2.1.0
  - New Features
    - Added a host descriptor to contain SDHC related attributes.
  - Bug Fixes
    - Removed clock auto gated function because of that it is a hardware issue.
  - Other Changes
    - Added more SDIO card related command type.
    - Changed the callback mechanism in the non-blocking transaction API.
    - Merged the two ADMA configuration functions to be one.
    - Changed the transaction API's name.

SIM

The current SIM driver version is 2.1.1.

- 2.1.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.1, 10.4
- 2.1.0
  - Improvements
    - Added new APIs: SIM_GetRfAddr() and SIM_EnableSystickClock().
- 2.0.0
  - Initial version.

SMC

The current SMC driver version is 2.0.7.

- 2.0.7
  - Bug Fixes
    - Fixed MISRA-2012 issue 10.3.
- 2.0.6
  - Bug Fixes

        - Fixed issue for MISRA-2012 check.
            - Fixed rule 10.3, rule 11.3.
- 2.0.5
  - Bug Fixes
    - Fixed issue for MISRA-2012 check.
      - Fixed rule 15.7, rule 14.4, rule 10.3, rule 10.1, rule 10.4.
- 2.0.4
  - Bug Fixes
    - When entering stop modes, used RAM function for the flash synchronization issue. Application should make sure that, the RW data of fsl_smc.c is located in memory region which is not powered off in stop modes.
- 2.0.3
  - Improvements
    - Added APIs SMC_PreEnterStopModes, SMC_PreEnterWaitModes, SMC_PostExitWaitModes, and SMC_PostExitStopModes.
- 2.0.2
  - Bug Fixes
    - Added DSB before WFI while ISB after WFI.
  - Other Changes
    - Updated SMC_SetPowerModeVlpw implementation.
- 2.0.1
  - Other Changes
    - Updated for KL8x.
- 2.0.0
  - Initial version.

UART

The current UART driver version is 2.3.0.

- 2.3.0
  - Bug Fixes
    - Fixed the bug that, when framing/parity/noise/overflow flag or idle line detect flag is set, receive FIFO should be flushed to avoid FIFO pointer being in unknown state, since FIFO has no valid data.
  - Improvements
    - Modified UART_TransferHandleIRQ so that txState will be set to idle only when all data has been sent out to bus.
    - Modified UART_TransferGetSendCount so that this API returns the real byte count that UART has sent out rather than the software buffer status.
    - Added timeout mechanism when waiting for certain states in transfer driver.
- 2.2.0
  - New Features
    - Added UART hardware FIFO enable/disable API.

  - Improvements
    - Added check for kUART_TransmissionCompleteFlag in UART_TransferHandleIRQ, UART_SendEDMACallback and UART_TransferSendDMACallback to ensure all the data would be sent out to bus.
  - Bug Fixes
    - Eliminated IAR Pa082 warnings from UART_TransferGetRxRingBufferLength, UART_GetEnabledInterrupts, UART_GetStatusFlags and UART_TransferHandleIRQ.
    - Added code in UART_ReadBlocking so that if more than one receiver errors occur, all status flags will be cleared and the most severe error status will be returned.
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 14.4, 11.6, 17.7.
- 2.1.6
  - Bug Fixes
    - Fixed the issue of register's being in repeatedly reading status while performing the IRQ routine.
- 2.1.5
  - Improvements
    - Added hardware flow control function support.
    - Added idle-line-detecting feature in UART_TransferNonBlocking function. If an idle line is detected, a callback will be triggered with status kStatus_UART_IdleLineDetected returned. This feature may be useful when the number of received bytes is less than the expected receive data size. Before triggering the callback, data in the FIFO is read out (if it has FIFO), and no interrupt will be disabled except for the case that the receive data size reaches 0.
    - Enabled the RX FIFO watermark function. With the idle-line-detecting feature enabled, you can set the watermark value to whatever you want (should not be bigger than the RX FIFO size). Data is then received and a callback will be triggered when data receive ends.
- 2.1.4
  - Improvements
    - Changed parameter type in UART_RTOS_Init() struct rtos_uart_config –> uart_rtos_config_t.
  - Bug Fixes
    - Disabled UART receive interrupt instead of global interrupt when reading data from ring buffer. With ring buffer used, receive nonblocking will disable global interrupt to protect the ring buffer. This has a negative effect on other IPs using interrupt.
- 2.1.3
  - New Features
    - Added RX framing error and parity error status check when using interrupt transfer.
- 2.1.2
  - Bug Fixes
    - Fixed baud rate fine adjust bug to make the computed baud rate more accurate.
- 2.1.1
  - Bug Fixes
    - Removed needless check of event flags and assert in UART_RTOS_Receive.

        * Always waited for RX event flag in UART_RTOS_Receive.
- 2.1.0
  - Improvements
    * Added transactional API.
- 2.0.0
  - Initial version.

VREF

The current VREF driver version is 2.1.2.

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

WDOG

The current WDOG driver version is 2.0.1.

- 2.0.1
  - Bug Fixes
    * MISRA C-2012 issue fixed: rule 10.3, 10.4, 10.6, 10.7 11.9 and 17.7.
- 2.0.0
  - Initial version.

Middleware Change Log

NXP Analog Middleware Layer

The current version is 1.4.3.

- 1.4.3
  - Bug fix:
    - Fixed compiler errors in MDK v6.
- 1.4.2
  - AML is based on MCUXpresso SDK 2.4 and S32 SDK 0.8.6 EAR.
  - Bug fix:
    - Fixed errors and warnings in MDK and IAR toolchains.
- 1.4.1
  - AML is based on MCUXpresso SDK 2.3 and S32 SDK 0.8.6 EAR.
- 1.4.0
  - AML is based on MCUXpresso SDK 2.3 and S32 SDK 0.8.4 EAR.
  - SDK_2_0 macro definition was replaced by SDK_KSDK.
  - API functions ADC_AML_InstallHandler, AML_GPIO_InstallHandler were removed as they were not used in any driver.
  - kStatus_Busy added to _generic_status enum.
  - New features:
    - CAN
  - Bug fix:
    - GPIO_AML_SetDirection: Fixed issue in KSDK AML. It used to both change the direction and also modify pin output value.
- 1.3.0
  - New features:
    - AML supports the newest S32K144 SDK (1.0.0 RTM).
    - SPI_AML: Bit count per frame was inserted into AML SPI for both master and slave configuration structures.
    - SPI_AML: For S32 SDK, interrupts were set as the transfer type.
  - Bug fix:
    - TMR_AML_Init: Fixed issue with FTM initialization if there is no channel for allocation.
    - TMR_AML_ResetTimer: Fixed issue with FTM, which did not leave this function.
  - AML status_t was replaced by aml_status_t because S32 SDK started to use global status_t.
- 1.2.0
  - Bug fix:
    - Fixed issue with timer initialization if there is no channel for allocation (e.g. KL25 calls assert if there was 0 channels).
- 1.1.0
  - New features:
    - WAIT

- 1.0.0
  - Initial version based on KSDK 2.0 and S32K144 SDK 0.8.1 EAR.
  - Supported peripherals:
    - Timer (TPM, FTM)
    - SPI (SPI, DSPI)
    - ADC
    - GPIO
    - I2C

IOT C SDK for KSDK

Current version is azure-iot-sdk-c 1.3.4_rev1.

- 1.3.4_rev1
  - Fixed lwIP SNTP called without lock.
- 1.3.4_rev0
  - update version
- 1.2.13_rev0
  - update version
  - add AMQP support
- 1.2.8_rev0
  - add Azure IOT snapshot from github.com/Azure/azure-iot-sdk-c
  - Create minimalistic snapshot without submodules
  - Changed files: c-utility\adapters\tlsio_mbedtls.c
    - remove deprecated USE_MBED_TLS conditional macro
    - include string.h, stdio.h, fsl_debug_console.h
    - comment out MBED_TLS_DEBUG_ENABLE definition
    - comment out mbedtls_ssl_conf_dbg and mbedtls_debug_set_threshold c-utility\inc\azure-_c_shared_utility\agenttime.h
    - Add sntp_get_current_timestamp declaration c-utility\inc\azure_c_shared_utility\crt_-abstractions.h
    - Add ENOMEM and EINVAL definitions (IAR specific) c-utility\inc\azure_c_shared_-utility\gbnetwork.h
    - Include lwip/sockets.h c-utility\inc\azure_c_shared_utility\tlsio_mbedtls.h
    - remove deprecated USE_MBED_TLS conditional macro
    - remove deprecated tls_config.h include c-utility\pal\freertos\lock.c
    - adapt freertos path to MSDK folder structure c-utility\pal\freertos\threadapi.c
    - adapt frertos path, use portTICK_PERIOD_MS c-utility\pal\freertos\tickcounter.c
    - adapt freertos path, use portTICK_PERIOD_MS c-utility\pal\lwip\sntp_lwip.c
    - include time.h
    - adapt SNTP_Init c-utility\pal\lwip\sntp_os.h
    - include lwip/apps/sntp.h. instead of apps/sntp/sntp.h deps\parson\parson.c
    - ignore functions requiring FILE_DESCRIPTOR iothub_client\src\iothub_client_-diagnostic.c

        * remove int64_t check, because of undefined type, observed for armgcc iothub_client\src\iothubtransport_mqtt_common.c
        * Comment out ResetConnectionIfNecessary, cause failure.
    - Renamed files (Fix conflicts with system libraries or other middlewares): c-utility\src\base64_azure.c c-utility\src\hmac_azure.c c-utility\src\sha1_azure.c iothub_client\src\version_azure.c
    - Removed: azure_c_shared_utility/tls_config.h
        * deprecated
    - New files: c-utility\adapters\agenttime_msdk.c
        * Add MSDK adaptation layer c-utility\adapters\platform_msdk.c
        * Add MSDK adaptation layer c-utility\adapters\socketio_berkeley_msdk.c
        * Add lwip adaptation layer

emWin library

The currently supported version is 6.10f

FatFs for MCUXpresso SDK

Current version is FatFs R0.14_rev0.

- 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 3.0.2. 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.

LittlevGL for KSDK

- 7.0.0_rev1
  - Integrate LittlevGL 7.0.0 to SDK.
  - Added PXP hardware acceleration initial version.
- 6.1.1_rev1
  - Integrate LittlevGL 6.1.1 to SDK.
- 5.3_rev1
  - Integrate LittlevGL 5.3 to SDK.

NXP Low Voltage H-Bridge driver for MCUXpresso SDK

The current version is 1.0.0.

- 1.0.0
  - Initial version.

lwIP for MCUXpresso SDK

The current version of lwIP is based on lwIP 2.2.0.dev.

- 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 con-

tent 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 Sharp-SnmpLib.
- 2.1.2_rev3
  - New features:
    - lwiperf updated with UDP client/server support from the patch 9751 (https://savannah.nongnu.org/patch/?9751)
- 2.1.2_rev2
  - Bug fixes:
    - Fixed lwiperf_abort() in lwiperf.c to correctly close connections and free resources
- 2.1.2_rev1
  - New features:
    - Ported lwIP 2.1.2 (2018-11-22, SHA-1: 159e31b689577dbf69cf0683bbaffbd71fa5ee10) to KSDK 2.0.0.
    - Ported lwIP-contrib 2.1.0 (2018-09-24, SHA-1: 35b011d4cf4c4b480f8859c456587a884ec9d287) to KSDK 2.0.0.
- 2.0.3_rev1
  - New features:
    - Ported lwIP 2.0.3 (2017-09-15, SHA-1: 92f23d6ca0971a32f2085b9480e738d34174417b) to KSDK 2.0.0.
- 2.0.2_rev1
  - New features:
    - Ported lwIP 2.0.2 (2017-03-13, SHA-1: c0862d60746e2d1ceae69af4c6f24e469570ecef) to KSDK 2.0.0.
- 2.0.0_rev3
  - New features:
    - Ported lwIP 2.0.0 (2016-11-10, SHA-1: 216bf89491815029aa15463a18744afa04df58fe) to KSDK 2.0.0.
- 2.0.0_rev2
  - New features:
    - Ported lwIP 2.0.0 RC2 (2016-08-08, SHA-1: b1dfd00f9233d124514a36a8c8606990016f2ad4) to KSDK 2.0.0.
- 2.0.0_rev1
  - New features:
    - Ported lwIP 2.0.0 RC0 (2016-05-26) to KSDK 2.0.0.
    - Changed lwIP bare-metal examples to use poll-driven approach instead of interrupt-driven one.
- 1.4.1_rev2
  - New features:
    - Enabled critical sections in lwIP.
  - Bug fixes:

        - Fixed default lwIP packet-buffer size to be able to accept a maximum size frame from the ENET driver.
        - Fixed possible drop of multi-frame packets during transmission.
- 1.4.1_rev1
  - New features:
    - Ported lwIP 1.4.1 to KSDK 2.0.0.

mbedTLS for MCUXpresso SDK

The current version of mbedTLS is based on mbedTLS 2.16.6 released 2020-04-14

- 2.16.6
  - New features:
    - Ported mbedTLS 2.16.6 to SDK.
- 2.16.2_rev2
  - Bug fixes:
    - Add support for HASHCRYPT context switch check, Hashcrypt without context switch is not able to calculate SHA in parallel with AES. HW acceleration of SHA is disabled by default in MbedTLS integration, enabled on chip with context switch.
- 2.16.2_rev1
  - Bug fixes:
    - Add support for CTR_DRBG using AES-128 for crypto engines without AES-256 capability.
- 2.16.2
  - New features:
    - Ported mbedTLS 2.16.2 to SDK.
- 2.13.1_rev5
  - Bug fixes:
    - ecp_alt_ksdk.c fix CASPER port for ECJPAKE shortcut when points equal 1. This case is point addition and this shortcut follows original mbedtls_ecp_muladd() implementation which is required for ecjpake_ecp_add3().
- 2.13.1_rev4
  - New features:
    - Added support for NIST P-384 elliptic curve with CASPER driver.
- 2.13.1_rev3
  - Bug fixes:
    - Force align AES_CCM and AES_GCM self-test keys to fix unaligned key issue when using HW acceleration.
- 2.13.1_rev2
  - Bug fixes:
    - Disable default HW acceleration of SHA in parallel with AES.
- 2.13.1_rev1
  - Bug fixes:
    - Fixed incorrect macro check when skipping AES-192 or AES-256

- 2.13.1
  - New features:
    - Ported mbedTLS 2.13.1 to KSDK.
- 2.12.0_rev1
  - New features:
    - Added support for NIST P-256 elliptic curve with CASPER driver.
- 2.12.0
  - New features:
    - Ported mbedTLS 2.12.0 to KSDK.
- 2.9.0_rev2
  - New features:
    - Added support for Hashcrypt driver.
- 2.9.0_rev1
  - New features:
    - Added support for CASPER driver.
- 2.9.0
  - New features:
    - Ported mbedTLS 2.9.0 to KSDK.
- 2.6.0_rev2
  - Bug fixes:
    - ssl_cookie.c now uses SHA256 for COOKIE_MD (instead of original SHA224). Some hw crypto acceleration (such as CAU3) don't support SHA224 but all support SHA256.
- 2.6.0_rev1
  - Bug fixes:
    - ksdk_mbedtls.c bignum functions now read sign of input mbedtls_mpi at beginning of functions to properly support in place computations (when output bignum is the same as one of input bignums). Affected functions: mbedtls_mpi_mul_mpi(), mbedtls_mpi_mod_mpi(), ecp_mul_comb().
- 2.6.0
  - New features:
    - Ported mbedTLS 2.6.0 to KSDK.
    - Added MBEDTLS_FREESCALE_FREERTOS_CALLOC_ALT to allow alternate implementation of pvPortCalloc() when using /middleware/mbedtls/port/ksdk/ksdk_mbedtls.c.
- 2.5.1_rev1
  - New features:
    - Added support for DCP driver.
- 2.5.1
  - New features:
    - Ported mbedTLS 2.5.1 to KSDK.
- 2.4.2_rev2
  - New features:
    - Added Curve25519 support for CAU3.
    - Added MBEDTLS_ECP_MUL_MXZ_ALT configuration parameter enabling overloading of ecp_mul_mxz().

- 2.4.2_rev1
  - New features:
    - Added support for CAU3 driver.
    - Added new files:
    - /middleware/mbedtls/port/ksdk/des_alt.c - contains regular software implementation of DES algorithm with added MBEDTLS_DES3_SETKEY_DEC_ALT and MBEDTLS-_DES3_SETKEY_ENC_ALT config parameters.
    - /middleware/mbedtls/port/ksdk/des_alt.h - contains modified mbedtls_des_context and mbedtls_des3_context structures.
    - Added MBEDTLS_DES3_SETKEY_DEC_ALT configuration parameter enabling reloading of mbedtls_des3_set2key_dec() and mbedtls_des3_set3key_dec().
    - Added MBEDTLS_DES3_SETKEY_ENC_ALT configuration parameter enabling reloading of mbedtls_des3_set2key_enc() and mbedtls_des3_set3key_enc().
- 2.4.2
  - New features:
    - Ported mbedTLS 2.4.2 to KSDK 2.0.0.
    - Added CRYPTO_InitHardware() function.
    - Added new file:
      - /middleware/mbedtls/port/ksdk/ksdk_mbedtls.h - contains declaration of CRYPTO_InitHardware() function and should be included in applications.
- 2.3.0_rev1
  - New features:
    - Added support for CAAM driver.
    - In LTC-specific wrapper, allocate temporary integers from heap in one large block.
- 2.3.0
  - New features:
    - Ported mbedTLS 2.3.0 to KSDK 2.0.0.
- 2.2.1
  - New features:
    - Ported mbedTLS 2.2.1 to KSDK 2.0.0.
    - Added support of MMCAU cryptographic acceleration module. Accelerated MD5, SHA, AES, and DES.
    - Added support of LTC cryptographic acceleration module. Accelerated AES, DES, and PKHA.
    - Added new files:
    - /middleware/mbedtls/port/ksdk/ksdk_mbedtls.c - alternative implementation of cryptographic algorithm functions using LTC and MMCAU module drivers.
    - /middleware/mbedtls/port/ksdk/ksdk_mbedtls_config.h - configuration settings used by mbedTLS KSDK bare metal examples.
    - Added mbedTLS KSDK bare-metal examples:
      - /boards/<board name>/demo_apps/mbedtls/mbedtls_benchmark - KSDK mbedTLS benchmark application.
      - /boards/<board name>/demo_apps/mbedtls/mbedtls_selftest - KSDK mbedTLS self-test application.
    - Added MBEDTLS_GCM_CRYPT_ALT configuration parameter enabling reloading of

mbedtls_gcm_crypt_and_tag().
* Added MBEDTLS_ECP_MUL_COMB_ALT to enable alternate implementation of ecp_mul_comb().
* Added MBEDTLS_ECP_ADD_ALT configuration parameter enabling reloading of ecp_add().
* Added MBEDTLS_DES_SETKEY_DEC_ALT configuration parameter enabling reloading of mbedtls_des_setkey_dec(), mbedtls_des3_set2key_dec() and mbedtls_-des3_set3key_dec().
* Added MBEDTLS_DES_SETKEY_ENC_ALT configuration parameter enabling reloading of mbedtls_des_setkey_enc(), mbedtls_des3_set2key_enc() and mbedtls_-des3_set3key_enc().
* Added MBEDTLS_DES_CRYPT_CBC_ALT configuration parameter enabling reloading of mbedtls_des_crypt_cbc().
* Added MBEDTLS_DES3_CRYPT_CBC_ALT configuration parameter enabling reloading of mbedtls_des3_crypt_cbc().
* Added MBEDTLS_AES_CRYPT_CBC_ALT configuration parameter enabling reloading of mbedtls_aes_crypt_cbc().
* Added MBEDTLS_AES_CRYPT_CTR_ALT configuration parameter enabling reloading of mbedtls_aes_crypt_ctr().
* Added MBEDTLS_CCM_CRYPT_ALT configuration parameter enabling reloading of mbedtls_ccm_encrypt_and_tag() and mbedtls_ccm_auth_decrypt().
* Added MBEDTLS_MPI_ADD_ABS_ALT configuration parameter enabling reloading of mbedtls_mpi_add_abs().
* Added MBEDTLS_MPI_SUB_ABS_ALT configuration parameter enabling reloading of mbedtls_mpi_sub_abs().
* Added MBEDTLS_MPI_EXP_MOD_ALT configuration parameter enabling reloading of mbedtls_mpi_exp_mod().
* Added MBEDTLS_MPI_MUL_MPI_ALT configuration parameter enabling reloading of mbedtls_mpi_mul_mpi().
* Added MBEDTLS_MPI_MOD_MPI_ALT configuration parameter enabling reloading of mbedtls_mpi_mod_mpi().
* Added MBEDTLS_MPI_GCD_ALT configuration parameter enabling reloading of mbedtls_mpi_gcd().
* Added MBEDTLS_MPI_INV_MOD_ALT configuration parameter enabling reloading of mbedtls_mpi_inv_mod().
* Added MBEDTLS_MPI_IS_PRIME_ALT configuration parameter enabling reloading of mbedtls_mpi_is_prime().
* Added encrypt/decrypt mode to mbedtls_des_context and mbedtls_des3_context structure.
* Added carriage return '\r' for mbedtls_printf() in self test functions.

Multicore SDK

The current version of Multicore SDK is 2.8.0.

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

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

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

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

NTAG I2C plus library

The current version is 1.0.0.

- 1.0.0
  - initial release.

Host SDHC driver for MCUXpresso SDK

The current driver version is 2.3.0.

- 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 2.3.0.

- 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 2.3.0.

- 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 2.3.0.

- 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.

SD(SPI) Card driver for MCUXpresso SDK

The current driver version is 2.1.4.

- 2.1.4
  - Bug Fixes
    * Fixed MDK 66-D warning.
- 2.1.3
  - Improvements
    * Improved sdspi code size and performance.
- 2.0.0
  - Initial version.

Secure Element hostlib for KSDK

- 2.11.0
  - Added Support for SE050 and SSS API's
  - For Detailed change log please download the Plug & Trust MW package from https://www.nxp.com/products/:SE050?tab=Design_Tools_Tab and refer to the Changes Present in the User guide.
- 1.6.0
  - Host Library version goes from 01.40 to 01.41 (A71CH Host API has not been extended)
  - Conditional translation of i2c_Failed into i2c_NoAddrAck removed (this translation is no longer required as a NACK is no longer lumped into i2c_Failed by the I2C driver wrapper)

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

NXP Sigfox driver

The current version of Sigfox driver integrated in MCUXpresso SDK is 2.0.0. Version 2.1.0 can be found at NXP websites.

- 2.0.0
  - Sigfox driver determined for OL2385 with A1 firmware (version 1.12.0.2.8.20.17).
- 1.0.0
  - Initial version determined for OL2385 with A0 firmware.

USB stack for MCUXpresso SDK

The current version of USB stack is 2.6.0.

- 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.
- 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

QCA WiFi

The current version is 2.0.0.

- 2.0.0
  - Initial version.
    - Added QCA WiFi, ported from SDK 1.3, synchronized with latest MQX Qualcomm v3.3.5.
  - Known issues:
    - Low power mode may not work, require further investigation.
    - DHCP request requires some timeout to retrieve valid data.

RTOS Change Log

FreeRTOS for MCUXpresso SDK.

The current version is Amazon-FreeRTOS 202002.00 Original package is available at github.-com/aws/amazon-freertos.

- 202002.00_rev1
  - updated iot_tls.c as per latest SSS stack v2.14. All SSS/SE05x code under conditional compilation.
  - fix "#pragma weak"" issue caused by IAR update to version 8.50.5. Use __weak definition of vPortSetupTimerInterrupt instead of "#pragma weak" declaration.
- 202002.00_rev0
  - update aws_iot_network_manager
  - enable lowpower tickless for CA9, CM33
- 201908.00_rev0
  - update amazon freertos version
  - Fix freertos_tasks_c_additions.h - fix IAR build fail
  - update queue.c - add definition for pvBuffer necessary for segger sysview
  - iot_crypto.c - change include file to be possible include mbedtls config file defined by MBEDTLS_CONFIG_FILE macro
  - iot_mqtt_agent.h - extend MQTTAgentConnectParams_t structure - required by se_hostlib examples
  - Fixed build warnings:
    - aws_dev_mode_key_provisioning.c - some variables were declared but never referenced
    - aws_iot_network_manager.c - some functions were declared but never referenced
    - iot_device_metrics.c - add include
    - iot_pkcs11_mbedtls.c - incompatible pointer type, unused variable
    - iot_demo_freertos.c - macro expansion producing 'defined' has undefined behavior
    - iot_pkcs11_mbedtls.c - comparison of address not equal to a null pointer is always true
    - pkcs11.h - '__PASTE' macro redefined
  - changes required for se_hostlib examples
    - iot_default_root_certificates.h tlsCombi_ROOT_CERTIFICATE_PEM added for se_-hostlib cloud demos
    - iot_mqtt_agent.h modified structure to support for clouds which use usename and password
    - iot_mqtt_agent.c assigning structure to support for clouds which use usename and password
    - iot_crypto.c threading alt under MBEDTLS_THREADING_ALT
    - iot_tls.c Supoort for secure element handling
    - aws_pkcs11_pal.c pkcs11 handling for SE050
  - Introduced aws_ota_pal.c for RT1060 supporting ota_bootloader (mcu-boot)
  - updated lwIP template with options for core locking

- 1.4.9_rev0
  - Remove 3rd party libraries lwip, mbedtls (use MCUXpresso SDK versions).
  - Add missing comments to heap_useNewlib.c.
- 1.4.7_rev0
  - New features:
    - Add optional allocation scheme heap_useNewlib.c by D. Nadler.
    - Enable task aware debugging for cm33 platforms
    - Move tickless implementation to application layer
  - Other changes:
    - Fix other build warnings, errors
- 1.4.6_rev0
  - New features:
    - Update support of CM33 port with Trustzone, MPU, FPU support
    - Add support for AWS test for Cypress WiFi
    - Use lwip netif api to avoid lwIP raw API calls outside of tcpip thread in aws_wifi.c
  - Other changes:
    - Fix issues with mflash driver
    - Fix other build warnings, errors
- 1.4.0_rev1
  - New features:
    - Add implementation of vTaskEndScheduler for CM0 GCC port.
    - Support for CM33, CM33F architectures based on CM3, CM4F ports
- 1.4.0_rev0
  - New features:
    - Support for pkcs11 for several platforms, secure element host library under pkcs11/portable/nxp folder
    - Lwip, wifi_qca support for secure_sockets in secure_sockets/portable/nxp folder
    - Flash driver support for several platforms in third_party/mcu_vendor/nxp folder
    - Generic support for aws_wifi under wifi/portable/nxp/common folder
  - Other changes:
    - Fix several build warnings, errors

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

- 9.0.0_rev3
  - New features:
    - Tickless idle mode support for Cortex-A7. Add fsl_tickless_epit.c and fsl_tickless_generic.h in portable/IAR/ARM_CA9 folder.
    - Enabled float context saving in IAR for Cortex-A7. Added configUSE_TASK_FPU_SUPPORT macros. Modified port.c and portmacro.h in portable/IAR/ARM_CA9 folder.
  - Other changes:
    - Transformed ARM_CM core specific tickless low power support into generic form under freertos/Source/portable/low_power_tickless/.
- 9.0.0_rev2
  - New features:

        - Enabled MCUXpresso thread aware debugging. Add freertos_tasks_c_additions.h and configINCLUDE_FREERTOS_TASK_C_ADDITIONS_H and configFRTOS_MEMORY_SCHEME macros.
- 9.0.0_rev1
  - New features:
    - Enabled -flto optimization in GCC by adding **attribute**((used)) for vTaskSwitchContext.
    - Enabled KDS Task Aware Debugger. Apply FreeRTOS patch to enable configRECORD_STACK_HIGH_ADDRESS macro. Modified files are task.c and FreeRTOS.h.
- 9.0.0_rev0
  - New features:
    - Example freertos_sem_static.
    - Static allocation support RTOS driver wrappers.
  - Other changes:
    - Tickless idle rework. Support for different timers is in separated files (fsl_tickless_systick.c, fsl_tickless_lptmr.c).
    - Removed configuration option configSYSTICK_USE_LOW_POWER_TIMER. Low power timer is now selected by linking of apropriate file fsl_tickless_lptmr.c.
    - Removed configOVERRIDE_DEFAULT_TICK_CONFIGURATION in RVDS port. Use of **attribute**((weak)) is the preferred solution. Not same as _weak!
- 8.2.3
  - New features:
    - Tickless idle mode support.
    - Added template application for Kinetis Expert (KEx) tool (template_application).
  - Other changes:
    - Folder structure reduction. Keep only Kinetis related parts.

Component Change Log

SERIAL_MANAGER

The current Serial_Manager component version is 1.0.0.

- 1.0.0
  - Initial version

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document.

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

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