MCUXpresso SDK Release Notes Supporting evkmimxrt595

Change Logs

NXP Semiconductors

Contents

Title Page No.

Title Page No.

Component Change Log

1 Driver Change Log

CLOCK

The current CLOCK driver version is 2.7.0.

- 2.7.0
  - API changes
    - Added CLOCK_FroTuneToFreq and CLOCK_EnableFroClkFreq API.
- 2.6.1
  - Improvements
    - Added lost comments for some enumerations.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.1.
- 2.6.0
  - API changes
    - Added CLOCK_EnableFroClkRange API.
    - FRO clock name changed from FRO192M, FRO96M, FRO48M, FRO24M, FRO12M to FRO_DIV1,FRO_DIV2, FRO_DIV4, FRO_DIV8, FRO_DIV16.
  - Bug Fixes
    - Added kAUX0_PLL_to_MIPI_DPHYESC_CLK, kAUX1_PLL_to_MIPI_DPHYESC_-CLK for clock_attach_id_t.
    - Fixed the error usage of macro in CLOCK_DeinitSysPfd() function.
- 2.5.1
  - Bug Fixes
    - Updated enum sys_pll_mult_t and audio_pll_mult_t to fix the supported MULT values for PLLs.
    - Added kHCLK_to_OSTIMER_CLK for clock_attach_id_t.
    - Fixed the calculation of main_pll_clk, dsp_pll_clk, aux0_pll_clk, aux0_pll_clk.
    - Renamed "kFRO192M_to_CLKOUT" to "kFRO96M_to_CLKOUT" to align with RM.
- 2.5.0
  - API change
    - Added CLOCK_SetClkinFreq API.
  - Other Changes
    - Macro "CLK_CLK_IN" changed to "CLK_EXT_CLKIN".
- 2.4.0
  - API change
    - Added enableLowPower parameter in CLOCK_EnableSysOscClk().
  - Other Changes
    - Fixed C++ build errors.
    - Added assert in CLOCK_SetFRGClock(), the FRG DIV should be always set to 0xFF according to Reference Manual.
- 2.3.1
  - Other Changes:

    - Updated register access per the header file's change.
- 2.3.0
  - New feature:
    - Moved SDK_DelayAtLeastUs function from clock driver to common driver.
- 2.2.2
  - Bug Fixes
    - Avoided waiting REQFLAG when divider configured to HALT in CLOCK_SetClkDiv().
- 2.2.1
  - Added CLOCK_EnableLpOscClk() and CLOCK_EnableFroClk() API
- 2.2.0
  - New feature
    - Added Deinit PLL&PFD API.
  - API change
    - Added delay_us parameter in CLOCK_EnableSysOscClk()
- 2.1.0
  - New feature
    - Adding new API CLOCK_DelayAtLeastUs() implemented by DWT to allow users set delay in unit of microsecond.
- 2.0.1
  - Updated clock_attach_id_t elements, removing the FRG(Fractional Generator) clock source selection from CLOCK_AttachClk.
  - Users need call CLOCK_SetFRGClock to set FRG clock source.
- 2.0.0
  - initial version.

POWER

The current POWER driver version is 2.4.0.

- 2.4.0
  - Bug Fixes
    - Removed HSPAD related configurations.
- 2.3.3
  - Bug Fixes
    - Fixed MISRA issue in function POWER_GetLibVersion.
    - Cleared bit PDSLEEPCFG0[PMCREF_LP] when FRO or PLL enabled during deep sleep in function POWER_EnterDeepSleep.
- 2.3.2
  - Bug Fixes
    - Added PORCORE_LP bitfield in macro PCFG0_DEEP_SLEEP.
    - Updated pSlowSwitches calculation in function countPartitionSwitches.
    - Added PMC internal clock divider config in POWER_ApplyPD to decrease the PMC register access delay, incase the divider was enabled before.
- 2.3.1

  - Updated powerFreqLevel array for B2 sample.
- 2.3.0
  - Set MEMSEQNUM to 0x3F to turn on all partitions in parallel to decrease deep sleep wakeup time.
  - Updated power_pad_vrange_val_t for supported PAD voltage range.
- 2.2.2
  - Supported dual FRO frequency in deep sleep.
- 2.2.1
  - Optimized MAINCLKSAFETY calculation.
  - Used FRO48M instead of FRO192M as main clock source in deep sleep.
- 2.2.0
  - Moved power lib implementaion to fsl_power.c
- 2.1.0
  - Added LVD APIs.
  - Added POWER_UpdatePmicRecoveryTime() API.
- 2.0.4
  - Supported OTP switch RBB in deep sleep.
- 2.0.3
  - Improved XIP recovery in deep sleep wakeup.
- 2.0.2
  - Added POWER_SetDeepSleepClock() API to allow main clock source selection in deep sleep.
  - Added POWER_SetPadVolRange() API
- 2.0.1
  - Added POWER_UpdateOscSettlingTime() API to set on-board system osc settling time.
- 2.0.0
  - initial version.

RESET

The current RESET driver version is 2.0.1.

- 2.0.1
  - Bug Fixes
    * Fixed MISRA C-2012 rule 10.6 and rule 16.4.
- 2.0.0
  - initial version.

DSP

The current DSP driver version is 2.0.1.

- 2.0.1
  - Fixed Misra issue.
- 2.0.0

    - initial version.

MIPI DSI SOC level driver

Current MIPI DSI SOC level driver version is 2.0.0

- 2.0.0
  - Initial version.

ACMP

The current ACMP driver version is 2.0.6.

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

CACHE

The current CACHE driver version is 2.0.6.

- 2.0.6
  - Bug Fixes
    - Fixed overflow for CACHE64_GetInstanceByAddr()/CACHE64_CleanCacheByRange()/CACHE64_InvalidateCacheByRange() APIs.
- 2.0.5
  - Improvement
    - Made use of FSL_FEATURE_CACHE64_CTRL_HAS_NO_WRITE_BUF feature
- 2.0.4
  - Improvement
    - Disable cache policy feature on SoC without CACHE64_POLSEL IP.
  - Bug Fixes
    - Fixed doxygen issue.
- 2.0.3
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 10.3.
- 2.0.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 10.1, 10.3, 10.4 and 14.4.
    - Fixed doxygen issue.
- 2.0.1
  - Improvements
    - Moved CLCR register configuration out of the while loop, it's unnecessary to repeat this operation.
- 2.0.0
  - Initial version.

COMMON

The current COMMON driver version is 2.4.0.

- 2.4.0
  - New Features
    - Added EnableIRQWithPriority, IRQ_SetPriority, and IRQ_ClearPendingIRQ for ARM.
    - Added MSDK_EnableCpuCycleCounter, MSDK_GetCpuCycleCount for ARM.
- 2.3.3
  - New Features
    - Added NETC into status group.
- 2.3.2
  - Improvements
    - Make driver aarch64 compatible
- 2.3.1
  - Bug Fixes
    - Fixed MAKE_VERSION overflow on 16-bit platforms.
- 2.3.0

- Improvements
  - Split the driver to common part and CPU architecture related part.
- 2.2.10
  - Bug Fixes
    - Fixed the ATOMIC macros build error in cpp files.
- 2.2.9
  - Bug Fixes
    - Fixed MISRA C-2012 issue, 5.6, 5.8, 8.4, 8.5, 8.6, 10.1, 10.4, 17.7, 21.3.
    - Fixed SDK_Malloc issue that not allocate memory with required size.
- 2.2.8
  - Improvements
    - Included stddef.h header file for MDK tool chain.
  - New Features:
    - Added atomic modification macros.
- 2.2.7
  - Other Change
    - Added MECC status group definition.
- 2.2.6
  - Other Change
    - Added more status group definition.
  - Bug Fixes
    - Undef __VECTOR_TABLE to avoid duplicate definition in cmsis_clang.h
- 2.2.5
  - Bug Fixes
    - Fixed MISRA C-2012 rule-15.5.
- 2.2.4
  - Bug Fixes
    - Fixed MISRA C-2012 rule-10.4.
- 2.2.3
  - New Features
    - Provided better accuracy of SDK_DelayAtLeastUs with DWT, use macro SDK_DELA-Y_USE_DWT to enable this feature.
    - Modified the Cortex-M7 delay count divisor based on latest tests on RT series boards, this setting lets result be closer to actual delay time.
- 2.2.2
  - New Features
    - Added include RTE_Components.h for CMSIS pack RTE.
- 2.2.1
  - Bug Fixes
    - Fixed violation of MISRA C-2012 Rule 3.1, 10.1, 10.3, 10.4, 11.6, 11.9.
- 2.2.0
  - New Features
    - Moved SDK_DelayAtLeastUs function from clock driver to common driver.
- 2.1.4
  - New Features

    * Added OTFAD into status group.
- 2.1.3
  - Bug Fixes
    * MISRA C-2012 issue fixed.
      · Fixed the rule: rule-10.3.
- 2.1.2
  - Improvements
    * Add SUPPRESS_FALL_THROUGH_WARNING() macro for the usage of suppressing fallthrough warning.
- 2.1.1
  - Bug Fixes
    * Deleted and optimized repeated macro.
- 2.1.0
  - New Features
    * Added IRQ operation for XCC toolchain.
    * Added group IDs for newly supported drivers.
- 2.0.2
  - Bug Fixes
    * MISRA C-2012 issue fixed.
      · Fixed the rule: rule-10.4.
- 2.0.1
  - Improvements
    * Removed the implementation of LPC8XX Enable/DisableDeepSleepIRQ() function.
    * Added new feature macro switch "FSL_FEATURE_HAS_NO_NONCACHEABLE_S-ECTION" for specific SoCs which have no noncacheable sections, that helps avoid an unnecessary complex in link file and the startup file.
    * Updated the align(x) to **attribute**(aligned(x)) to support MDK v6 armclang compiler.
- 2.0.0
  - Initial version.

CTIMER

The current CTimer driver version is 2.3.1.

- 2.3.1
  - Bug Fixes
    * MISRA C-2012 issue fixed: rule 10.7 and 12.2.
- 2.3.0
  - Improvements
    * Added the CTIMER_SetPrescale(), CTIMER_GetCaptureValue(), CTIMER_Enable-ResetMatchChannel(), CTIMER_EnableStopMatchChannel(), CTIMER_EnableRising-EdgeCapture(), CTIMER_EnableFallingEdgeCapture(), CTIMER_SetShadowValue(),A-PIs Interface to reduce code complexity.
- 2.2.2

  - Bug Fixes
    - Fixed SetupPwm() API only can use match 3 as period channel issue.
- 2.2.1
  - Bug Fixes
    - Fixed use specified channel to setting the PWM period in SetupPwmPeriod() API.
    - Fixed Coverity Out-of-bounds issue.
- 2.2.0
  - Improvements
    - Updated three API Interface to support Users to flexibly configure the PWM period and PWM output.
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 8.4.
- 2.1.0
  - Improvements
    - Added the CTIMER_GetOutputMatchStatus() API Interface.
    - Added feature macro for FSL_FEATURE_CTIMER_HAS_NO_CCR_CAP2 and FSL_-FEATURE_CTIMER_HAS_NO_IR_CR2INT.
- 2.0.3
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.3, 10.4, 10.6, 10.7 and 11.9.
- 2.0.2
  - New Features
    - Added new API "CTIMER_GetTimerCountValue" to get the current timer count value.
    - Added a control macro to enable/disable the RESET and CLOCK code in current driver.
    - Added a new feature macro to update the API of CTimer driver for lpc8n04.
- 2.0.1
  - Improvements
    - API Interface Change
      - Changed API interface by adding CTIMER_SetupPwmPeriod API and CTIMER-_UpdatePwmPulsePeriod API, which both can set up the right PWM with high resolution.
- 2.0.0
  - Initial version.

DMIC

The current DMIC driver version is 2.3.0.

- 2.3.0
  - Improvements
    - Added new apis DMIC_ResetChannelDecimator/DMIC_EnableChannelGlobalSync/D-MIC_DisableChannelGlobalSync.
- 2.2.1
  - Bug Fixes

    * Fixed violations of the MISRA C-2012 rules 14.4, 17.7, 10.4, 10.3, 10.8, 14.3.
- 2.2.0
  - Bug Fixes
    * Corrected the usage of feature FSL_FEATURE_DMIC_IO_HAS_NO_BYPASS.
- 2.1.1
  - Improvements
    * Added feature FSL_FEATURE_DMIC_HAS_NO_IOCFG for IOCFG register.
- 2.1.0
  - New Features
    * Added API DMIC_EnbleChannelInterrupt/DMIC_EnbleChannelDma to replace API DMIC_SetOperationMode.
    * Added API DMIC_SetIOCFG and marked DMIC_ConfigIO as deprecated.
    * Added API DMIC_EnableChannelSignExtend to support sign extend feature.
- 2.0.5
  - Improvements
    * Changed some parameters' value of DMIC_FifoChannel API, such as enable, resetn, and trig_level. This is not possible for the current code logic, so it improves the DMIC_FifoChannel logic and fixes incorrect math logic.
- 2.0.4
  - Bug Fixes
    * Fixed the issue that DMIC DMA driver(ver2.0.3) did not support calling DMIC_TransferReceiveDMA in DMA callback as it did before version 2.0.3. But calling DMIC_TransferReceiveDMA in callback is not recommended.
- 2.0.3
  - New Features
  - Supported linked transfer in DMIC DMA driver.
  - Added new API DMIC_EnableChannelFifo/DMIC_DoFifoReset/DMIC_InstallDMADescriptor.
- 2.0.2
  - New Features
    * Supported more channels in driver.
- 2.0.1
  - New Features
    * Added a control macro to enable/disable the RESET and CLOCK code in current driver.
- 2.0.0
  - Initial version.

DMIC_DMA

The current DMIC_DMA driver version is 2.3.1.

- 2.3.1
  - Bug Fixes
    * Fixed violations of the MISRA C-2012 rules 10.3.

- 2.3.0
  - Refer DMIC driver change log 2.0.1 to 2.3.0

FLEXCOMM

The current FLEXCOMM driver version is 2.0.2.

- 2.0.2
  - Bug Fixes
    - Fixed typos in FLEXCOMM15_DriverIRQHandler().
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 10.8, 11.3, 11.6, 11.8, 11.9, 13.5.
  - Improvements
    - Added instance calculation in FLEXCOMM16_DriverIRQHandler() to align with Flexcomm 14 and 15.
- 2.0.1
  - Improvements
    - Added more IRQHandler code in drivers to adapt new devices.
- 2.0.0
  - Initial version.

I2C

The current I2C driver version is 2.3.1.

- 2.3.1
  - Improvement
    - Before master transfer with transactional APIs, enable master function while disable slave function and vise versa for slave transfer to avoid the one affecting the other.
  - Bug Fixes
    - Fixed bug in I2C_SlaveEnable that the slave enable/disable should not affect the other register bits.
- 2.3.0
  - Improvement
    - Added new return codes kStatus_I2C_EventTimeout and kStatus_I2C_SclLowTimeout, and added the check for event timeout and SCL timeout in I2C master transfer.
    - Fixed bug in slave transfer that the address match event should be invoked before not after slave transmit/receive event.
- 2.2.0
  - New Features
    - Added enumeration _i2c_status_flags to include all previous master and slave status flags, and added missing status flags.
    - Modified I2C_GetStatusFlags to get all I2C flags.
    - Added API I2C_ClearStatusFlags to clear all clearable flags not just master flags.

    - Modified master transactional APIs to enable bus event timeout interrupt during transfer, to avoid glitch on bus causing transfer hangs indefinitely.
  - Bug Fixes
    - Fixed bug that status flags and interrupt enable masks share the same enumerations by adding enumeration _i2c_interrupt_enable for all master and slave interrupt sources.
- 2.1.0
  - Bug Fixes
    - Fixed bug that during master transfer, when master is nacked during slave probing or sending subaddress, the return status should be kStatus_I2C_Addr_Nak rather than k-Status_I2C_Nak.
  - Bug Fixes
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.4, 13.5.
  - New Features
    - Added macro I2C_MASTER_TRANSMIT_IGNORE_LAST_NACK, so that user can configure whether to ignore the last byte being nacked by slave during master transfer.
- 2.0.8
  - Bug Fixes
    - Fixed I2C_MasterSetBaudRate issue that MSTSCLLOW and MSTSCLHIGH are incorrect when MSTTIME is odd.
- 2.0.7
  - Bug Fixes
    - Two dividers, CLKDIV and MSTTIME are used to configure baudrate. According to reference manual, in order to generate 400kHz baudrate, the clock frequency after CLK-DIV must be less than 2mHz. Fixed the bug that, the clock frequency after CLKDIV may be larger than 2mHz using the previous calculation method.
    - Fixed MISRA 10.1 issues.
    - Fixed wrong baudrate calculation when feature FSL_FEATURE_I2C_PREPCLKFRG_-8MHZ is enabled.
- 2.0.6
  - New Features
    - Added master timeout self-recovery support for feature FSL_FEATURE_I2C_TIMEOU-T_RECOVERY.
  - Bug Fixes
    - Eliminated IAR Pa082 warning.
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 10.8, 11.3, 11.6, 11.8, 11.9, 13.5.
- 2.0.5
  - Bug Fixes
    - Fixed wrong assignment for datasize in I2C_InitTransferStateMachineDMA.
    - Fixed wrong working flow in I2C_RunTransferStateMachineDMA to ensure master can work in no start flag and no stop flag mode.
    - Fixed wrong working flow in I2C_RunTransferStateMachine and added kReceiveData-BeginState in _i2c_transfer_states to ensure master can work in no start flag and no stop flag mode.

    - Fixed wrong handle state in I2C_MasterTransferDMAHandleIRQ. After all the data has been transfered or nak is returned, handle state should be changed to idle.
  - Improvements
    - Rounded up the calculated divider value in I2C_MasterSetBaudRate.
- 2.0.4
  - Improvements
    - Updated the I2C_WATI_TIMEOUT macro to unified name I2C_RETRY_TIMES
    - Updated the "I2C_MasterSetBaudRate" API to support baudrate configuration for feature QN9090.
  - Bug Fixes
    - Fixed build warnning caused by uninitialized variable.
    - Fixed COVERITY issue of unchecked return value in I2C_RTOS_Transfer.
- 2.0.3
  - Improvements
    - Unified the component full name to FLEXCOMM I2C(DMA/FREERTOS) driver.
- 2.0.2
  - Improvements
    - In slave IRQ:
      1. Changed slave receive process to first set the I2C_SLVCTL_SLVCONTINUE_MASK to acknowledge the received data, then do data receive.
      2. Improved slave transmit process to set the I2C_SLVCTL_SLVCONTINUE_MASK immediately after writing the data.
- 2.0.1
  - Improvements
    - Added I2C_WATI_TIMEOUT macro to allow users to specify the timeout times for waiting flags in functional API and blocking transfer API.
- 2.0.0
  - Initial version.

I2S

The current I2S driver version is 2.3.2

- 2.3.2
  - Bug Fixes
    - Fixed warning for comparison between pointer and integer.
- 2.3.1
  - Bug Fixes
    - Updated the value of TX/RX software transfer state machine after transfer contents are submitted to avoid race condition.
- 2.3.0 Improvements
  - Added api I2S_InstallDMADescriptorMemory/I2S_TransferSendLoopDMA/I2S_TransferReceiveLoopDMA to support loop transfer.
  - Added api I2S_EmptyTxFifo to support blocking flush tx fifo.

  - Updated api I2S_TransferAbortDMA by removed the blocking flush tx fifo from this function.

Bug Fixes

- Removed the while loop in abort transfer function to fix the dead loop issue under specific user case.

2.2.2

- Bug Fixes
  - Fixed violations of the MISRA C-2012 rules 8.4.

2.2.1

- Improvements
  - Added feature FSL_FEATURE_FLEXCOMM_INSTANCE_I2S_SUPPORT_SECONDARY_CHANNELn for the SOC has parts of instance support secondary channel.
- Bug Fixes
  - Added volatile statement for the state variable of i2s_handle and enable the mainline channel pair before enable interrupt to avoid the issue of code excution reordering which may cause the interrupt generated unexpectedly.

2.2.0

- Improvements
  - Added 8/16/24 bits mono data format transfer support in I2S driver.
  - Added new apis I2S_SetBitClockRate.
- Bug Fixes
  - Fixed the PA082 build warning.
  - Fixed the sign-compare warning.
  - Fixed violations of the MISRA C-2012 rules 10.4, 10.8, 11.9, 10.1, 11.3, 13.5, 11.8, 10.3, 10.7.
  - Fixed the Operand don't affect result Coverity issue.

2.1.0

- Improvements
  - Added a feature for the FLEXCOMM which supports I2S and has interconnection with DMIC.
  - Used a feature to control PDMDATA instead of I2S_CFG1_PDMDATA.
  - Added member bytesPerFrame in i2s_dma_handle_t, used for DMA transfer width configure, instead of using sizeof(uint32_t) hardcode.
  - Used the macro provided by DMA driver to define the I2S DMA descriptor.
- Bug Fixes
  - Fixed the issue that I2S DMA driver always generated duplicate callback.

2.0.3

- New Features
  - Added a feature to remove configuration for the second channel on LPC51U68.

2.0.2

- New Features
  - Added ENABLE_IRQ handle after register I2S interrupt handle.

2.0.1

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

2.0.0

- Initial version.

SPI

The current SPI driver version is 2.2.2

- 2.2.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules.
- 2.2.1
  - Bug Fixes
    - Fixed MISRA 2012 10.4 issue.
    - Added code to clear FIFOs before transfer using DMA.
- 2.2.0
  - Bug Fixes
    - Fixed bug that slave gets stuck during interrupt transfer.
- 2.1.1
  - Improvements
    - Added timeout mechanism when waiting certain states in transfer driver.
  - Bug Fixes
    - Fixed MISRA 10.1, 5.7 issues.
- 2.1.0
  - Bug Fixes
    - Fixed Coverity issue of incrementing null pointer in SPI_TransferHandleIRQInternal.
    - Eliminated IAR Pa082 warnings.
    - Fixed MISRA issues.
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 10.8, 11.3, 11.6, 11.8, 11.9, 13.5.
  - New Features
    - Modified the definition of SPI_SSELPOL_MASK to support the socs that have only 3 SSEL pins.
- 2.0.4
  - Bug Fixes
    - Fixed the bug of using read only mode in DMA transfer. In DMA transfer mode, if transfer->txData is NULL, code attempts to read data from the address of 0x0 for configuring the last frame.
    - Fixed wrong assignment of handle->state. During transfer handle->state should be kSP-I_Busy rather than kStatus_SPI_Busy.
  - Improvements
    - Rounded up the calculated divider value in SPI_MasterSetBaud.
- 2.0.3

  - Improvements
    - Added "SPI_FIFO_DEPTH(base)" with more definition.
- 2.0.2
  - Improvements
    - Unified the component full name to FLEXCOMM SPI(DMA/FREERTOS) driver.
- 2.0.1
  - Changed the data buffer from uint32_t to uint8_t which matches the real applications for SPI DMA driver.
  - Added dummy data setup API to allow users to configure the dummy data to be transferred.
  - Added new APIs for half-duplex transfer function. Users can not only send and receive data by one API in polling/interrupt/DMA way, but choose either to transmit first or to receive first. Besides, the PCS pin can be configured as assert status in transmission (between transmit and receive) by setting the isPcsAssertInTransfer to true.
- 2.0.0
  - Initial version.

USART

The current USART driver version is 2.8.0.

- 2.8.0
  - New Features
    - Added the rx timeout interrupts and status flags of bus status.
    - Added new rx timeout configuration item in usart_config_t.
    - Added API USART_SetRxTimeoutConfig for rx timeout configuration.
  - Improvements
    - When the calculated baudrate cannot meet user's configuration, lower OSR value is allewed to use.
- 2.7.0
  - New Features
    - Added the missing interrupts and status flags of bus status.
    - Added the check of tx error, noise error framing error and parity error in interrupt handler.
- 2.6.0
  - Improvements
    - Used separate data for TX and RX in usart_transfer_t.
  - Bug Fixes
    - Fixed bug that when ring buffer is used, if some data is received in ring buffer first before calling USART_TransferReceiveNonBlocking, the received data count returned by USART_TransferGetReceiveCount is wrong.
  - New Features
    - Added missing API USART_TransferGetSendCountDMA get send count using DMA.
- 2.5.0
  - New Features
    - Added APIs USART_GetRxFifoCount/USART_GetTxFifoCount to get rx/tx FIFO data

count.
    * Added APIs USART_SetRxFifoWatermark/USART_SetTxFifoWatermark to set rx/tx F-IFO water mark.
  - Bug Fixes
    * Fixed DMA transfer blocking issue by enabling tx idle interrupt after DMA transmission finishes.
- 2.4.0
  - New Features
    * Modified usart_config_t, USART_Init and USART_GetDefaultConfig APIs so that the hardware flow control can be enabled during module initialization.
  - Bug Fixes
    * Fixed MISRA 10.4 violation.
- 2.3.1
  - Bug Fixes
    * Fixed bug that operation on INTENSET, INTENCLR, FIFOINTENSET and FIFOINTE-NCLR should use bitwise operation not 'or' operation.
    * Fixed bug that if rx interrupt occurrs before TX interrupt is enabled and after txDataSize is configured, the data will be sent early by mistake, thus TX interrupt will be enabled after data is sent out.
  - Improvements
    * Added check for baud rate's accuracy that returns kStatus_USART_BaudrateNotSupport when the best achieved baud rate is not within 3% error of configured baud rate.
- 2.3.0
  - New Features
    * Added APIs to configure 9-bit data mode, set slave address and send address.
    * Modified USART_TransferReceiveNonBlocking and USART_TransferHandleIRQ to use 9-bit mode in multi-slave system.
- 2.2.0
  - New Features
    * Added the feature of supporting USART working at 32 kHz clocking mode.
  - Improvements
    * Modified USART_TransferHandleIRQ so that txState will be set to idle only when all data has been sent out to bus.
    * Modified USART_TransferGetSendCount so that this API returns the real byte count that USART has sent out rather than the software buffer status.
    * Added timeout mechanism when waiting for certain states in transfer driver.
  - Bug Fixes
    * Fixed MISRA 10.1 issues.
    * Fixed bug that operation on INTENSET, INTENCLR, FIFOINTENSET and FIFOINTE-NCLR should use bitwise operation not 'or' operation.
    * Fixed bug that if rx interrupt occurrs before TX interrupt is enabled and after txDataSize is configured, the data will be sent early by mistake, thus TX interrupt will be enabled after data is sent out.
- 2.1.1
  - Improvements

        - Added check for transmitter idle in USART_TransferHandleIRQ and USART_TransferSendDMACallback to ensure all the data would be sent out to bus.
        - Modified USART_ReadBlocking so that if more than one receiver errors occur, all status flags will be cleared and the most severe error status will be returned.
    - Bug Fixes
        - Eliminated IAR Pa082 warnings.
        - Fixed MISRA issues.
            - Fixed rules 10.1, 10.3, 10.4, 10.7, 10.8, 11.3, 11.6, 11.8, 11.9, 13.5.
- 2.1.0
    - New Features
        - Added features to allow users to configure the USART to synchronous transfer(master and slave) mode.
    - Bug Fixes
        - Modified USART_SetBaudRate to get more acurate configuration.
- 2.0.3
    - New Features
        - Added new APIs to allow users to enable the CTS which determines whether CTS is used for flow control.
- 2.0.2
    - Bug Fixes
        - Fixed the bug where transfer abort APIs could not disable the interrupts. The FIFOINTENSET register should not be used to disable the interrupts, so use the FIFOINTENCLR register instead.
- 2.0.1
    - Improvements
        - Unified the component full name to FLEXCOMM USART (DMA/FREERTOS) driver.
- 2.0.0
    - Initial version.

USART_DMA

The current USART_DMA driver version is 2.6.0.

- 2.6.0
    - Refer USART driver change log 2.0.1 to 2.6.0

USART_FREERTOS

The current USART_FREERTOS driver version is 2.6.0.

- 2.6.0
    - Refer USART driver change log 2.0.1 to 2.6.0

FLEXIO

The current FLEXIO driver version is 2.1.0.

- 2.1.0
  - Improvements
    - Added API FLEXIO_SetClockMode to set flexio channel counter and source clock.
- 2.0.4
  - Bug Fixes
    - Fixed MISRA 8.4 issues.
- 2.0.3
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
- 2.0.2
  - Improvements
    - Split FLEXIO component which combines all flexio/flexio_uart/flexio_i2c/flexio_i2s drivers into several components: FlexIO component, flexio_uart component, flexio_i2c_-master component, and flexio_i2s component.
  - Bug Fixes
    - Fixed MISRA issues
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
- 2.0.1
  - Bug Fixes
    - Fixed the dozen mode configuration error in FLEXIO_Init API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.

FLEXIO_UART

The current FLEXIO_UART driver version is 2.4.0.

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

- 2.2.0
  - Improvements
    - Added timeout mechanism when waiting for certain states in transfer driver.
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
- 2.1.6
  - Bug Fixes
    - Fixed IAR Pa082 warnings.
    - Fixed MISRA issues
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
- 2.1.5
  - Improvements
    - Triggered user callback after all the data in ringbuffer were received in FLEXIO_UART-_TransferReceiveNonBlocking.
- 2.1.4
  - Improvements
    - Unified component full name to FLEXIO UART(DMA/EDMA) Driver.
- 2.1.3
  - Bug Fixes
    - The following modifications support FLEXIO using multiple instances:
      - Removed FLEXIO_Reset API in module Init APIs.
      - Updated module Deinit APIs to reset the shifter/timer configuration instead of disabling module and clock.
      - Updated module Enable APIs to only support enable operation.
- 2.1.2
  - Bug Fixes
    - Fixed the transfer count calculation issue in FLEXIO_UART_TransferGetReceiveCount, FLEXIO_UART_TransferGetSendCount, FLEXIO_UART_TransferGetReceiveCount-DMA, FLEXIO_UART_TransferGetSendCountDMA, FLEXIO_UART_TransferGet-ReceiveCountEDMA and FLEXIO_UART_TransferGetSendCountEDMA.
    - Fixed the Dozen mode configuration error in FLEXIO_UART_Init API. For enableIn-Doze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
    - Added code to report errors if the user sets a too-low-baudrate which FLEXIO cannot reach.
    - Disabled FLEXIO_UART receive interrupt instead of all NVICs when reading data from ring buffer. If ring buffer is used, receive nonblocking will disable all NVIC interrupts to protect the ring buffer. This had negative effects on other IPs using interrupt.
- 2.1.1
  - Bug Fixes
    - Changed the API name FLEXIO_UART_StopRingBuffer to FLEXIO_UART_Transfer-StopRingBuffer to align with the definition in C file.
- 2.1.0
  - New Features
    - Added Transfer prefix in transactional APIs.

    - Added txSize/rxSize in handle structure to record the transfer size.
  - Bug Fixes
    - Added an error handle to handle the situation that data count is zero or data buffer is NULL.

FLEXIO_I2C

The current FLEXIO_I2C driver version is 2.5.0.

- 2.5.0
  - Improvements
    - Split some functions, fixed CCM problem in file fsl_flexio_i2c_master.c.
- 2.4.0
  - Improvements
    - Added delay of 1 clock cycle in FLEXIO_I2C_MasterTransferRunStateMachine to ensure that bus would be idle before next transfer if master is nacked.
    - Fixed issue that the restart setup time is less than the time in I2C spec by adding delay of 1 clock cycle before restart signal.
- 2.3.0
  - Improvements
    - Used 3 timers instead of 2 to support transfer which is more than 14 bytes in single transfer.
    - Improved FLEXIO_I2C_MasterTransferGetCount so that the API can check whether the transfer is still in progress.
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
- 2.2.0
  - New Features
    - Added timeout mechanism when waiting certain state in transfer API.
    - Added an API for checking bus pin status.
  - Bug Fixes
    - Fixed COVERITY issue of useless call in FLEXIO_I2C_MasterTransferRunState-Machine.
    - Fixed MISRA issues
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
    - Added codes in FLEXIO_I2C_MasterTransferCreateHandle to clear pending NVIC IRQ, disable internal IRQs before enabling NVIC IRQ.
    - Modified code so that during master's nonblocking transfer the start and slave address are sent after interrupts being enabled, in order to avoid potential issue of sending the start and slave address twice.
- 2.1.7
  - Bug Fixes
    - Fixed the issue that FLEXIO_I2C_MasterTransferBlocking did not wait for STOP bit sent.
    - Fixed COVERITY issue of useless call in FLEXIO_I2C_MasterTransferRunState-

    Machine.
    * Fixed the issue that I2C master did not check whether bus was busy before transfer.
- 2.1.6
  - Bug Fixes
    * Fixed the issue that I2C Master transfer APIs(blocking/non-blocking) did not support the situation of master transfer with subaddress and transfer data size being zero, which means no data followed the subaddress.
- 2.1.5
  - Improvements
    * Unified component full name to FLEXIO I2C Driver.
- 2.1.4
  - Bug Fixes
    * The following modifications support FlexIO using multiple instances:
      · Removed FLEXIO_Reset API in module Init APIs.
      · Updated module Deinit APIs to reset the shifter/timer config instead of disabling module/clock.
      · Updated module Enable APIs to only support enable operation.
- 2.1.3
  - Improvements
    * Changed the prototype of FLEXIO_I2C_MasterInit to return kStatus_Success if initialized successfully or to return kStatus_InvalidArgument if "(srcClock_Hz / masterConfig->baudRate_Bps) / 2 - 1" exceeds 0xFFU.
- 2.1.2
  - Bug Fixes
    * Fixed the FLEXIO I2C issue where the master could not receive data from I2C slave in high baudrate.
    * Fixed the FLEXIO I2C issue where the master could not receive NAK when master sent non-existent addr.
    * Fixed the FLEXIO I2C issue where the master could not get transfer count successfully.
    * Fixed the FLEXIO I2C issue where the master could not receive data successfully when sending data first.
    * Fixed the Dozen mode configuration error in FLEXIO_I2C_MasterInit API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
    * Fixed the issue that FLEXIO_I2C_MasterTransferBlocking API called FLEXIO_I2C_-MasterTransferCreateHandle, which lead to the s_flexioHandle/s_flexioIsr/s_flexioType variable being written. Then, if calling FLEXIO_I2C_MasterTransferBlocking API multiple times, the s_flexioHandle/s_flexioIsr/s_flexioType variable would not be written any more due to it being out of range. This lead to the following situation: NonBlocking transfer APIs could not work due to the fail of register IRQ.
- 2.1.1
  - Bug Fixes
    * Implemented the FLEXIO_I2C_MasterTransferBlocking API which is defined in header file but has no implementation in the C file.
- 2.1.0

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

FLEXIO_SPI

The current FLEXIO_SPI driver version is 2.3.0.

- 2.3.0
  - New Features
    - Supported FLEXIO_SPI slave transfer with continuous master CS signal and CPHA=0.
    - Supported FLEXIO_SPI master transfer with continuous CS signal.
    - Support 32 bit transfer width.
  - Bug Fixes
    - Fixed wrong timer compare configuration for dma/edma transfer.
    - Fixed wrong byte order of rx data if transfer width is 16 bit, since the we use shifter buffer bit swapped/byte swapped register to read in received data, so the high byte should be read from the high bits of the register when MSB.
- 2.2.1
  - Bug Fixes
    - Fixed bug in FLEXIO_SPI_MasterTransferAbortEDMA that when aborting EDMA transfer EDMA_AbortTransfer should be used rather than EDMA_StopTransfer.
- 2.2.0
  - Improvements
    - Added timeout mechanism when waiting certain states in transfer driver.
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
    - Added codes in FLEXIO_SPI_MasterTransferCreateHandle and FLEXIO_SPI_SlaveTransferCreateHandle to clear pending NVIC IRQ before enabling NVIC IRQ, to fix issue of pending IRQ interfering the on-going process.
- 2.1.3
  - Improvements
    - Unified component full name to FLEXIO SPI(DMA/EDMA) Driver.
  - Bug Fixes
    - Fixed MISRA issues
      - Fixed rules 10.1, 10.3, 10.4, 10.7, 11.6, 11.9, 14.4, 17.7.
- 2.1.2
  - Bug Fixes
    - The following modification support FlexIO using multiple instances:
      - Removed FLEXIO_Reset API in module Init APIs.
      - Updated module Deinit APIs to reset the shifter/timer config instead of disabling module/clock.
      - Updated module Enable APIs to only support enable operation.
- 2.1.1

  - Bug Fixes
    - Fixed bug where FLEXIO SPI transfer data is in 16 bit per frame mode with eDMA.
    - Fixed bug when FLEXIO SPI works in eDMA and interrupt mode with 16-bit per frame and Lsbfirst.
    - Fixed the Dozen mode configuration error in FLEXIO_SPI_MasterInit/FLEXIO_SPI_-SlaveInit API. For enableInDoze = true, the configuration should be 0; for enableInDoze = false, the configuration should be 1.
  - Improvements
    - Added #ifndef/#endif to allow users to change the default TX value at compile time.
- 2.1.0
  - New Features
    - Added Transfer prefix in transactional APIs.
    - Added transferSize in handle structure to record the transfer size.
  - Bug Fixes
    - Fixed the error register address return for 16-bit data write in FLEXIO_SPI_GetTxData-RegisterAddress.
    - Provided independent IRQHandler/transfer APIs for Master and slave to fix the baudrate limit issue.

FLEXIO_MCU_LCD

The current FLEXIO_MCU_LCD driver version is 2.0.7.

- 2.0.7
  - Bug Fixes
    - Fixed bug that FLEXIO_MCULCD_Init return kStatus_Success even with invalid parameter.
- 2.0.6
  - Bug Fixes
    - Fixed MISRA 10.4 issues when FLEXIO_MCULCD_DATA_BUS_WIDTH defined as signed value.
- 2.0.5
  - Improvements
    - Changed FLEXIO_MCULCD_WriteDataArrayBlocking's data parameter to const type.
- 2.0.4
  - Bug Fixes
    - Fixed MISRA 10.4 issues.
- 2.0.3
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.1, 10.3, 10.4, 10.6, 14.4, 17.7.
- 2.0.2
  - Improvements
    - Unified component full name to FLEXIO_MCU_LCD (EDMA) driver.
- 2.0.1

- Bug Fixes
  - The following modification to support FlexIO using multiple instances:
    - Removed FLEXIO_Reset API in module Init APIs.
    - Updated module Deinit APIs to reset the shifter/timer configuration instead of disabling module and clock.
    - Updated module Enable APIs to only support enable operation.
- 2.0.0
  - Initial version.

FLEXIO_MCU_LCD_SMARTDMA

The current FLEXIO_MCU_LCD_SMARTDMA driver version is 2.0.2.

- 2.0.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.1, 10.3, 10.4, 10.6, 14.4, 17.7.
- 2.0.1
  - Other Changes
    - Update driver implementation due to SMARTDMA driver update.
- 2.0.0
  - Initial version.

FLEXSPI

The current FLEXSPI driver version is 2.5.0.

- 2.5.0
  - Improvements
    - Supported word un-aligned access for write/read blocking/non-blocking API functions.
    - Fixed dead loop issue in DLL update function when using FRO clock source.
    - Fixed violations of the MISRA C-2012 Rule 10.3.
- 2.4.0
  - Improvements
    - Isolated IP command parallel mode and AHB command parallel mode using feature MACRO.
    - Supported new column address shift feature for external memory.
- 2.3.5
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 14.2.
- 2.3.4
  - Bug Fixes
    - Updated flexspi_config_t structure and FlexSPI_Init to support new feature FSL_FEATURE_FLEXSPI_HAS_NO_MCR0_CONBINATION.
- 2.3.3

- Bug Fixes
  - Removed feature FSL_FEATURE_FLEXSPI_DQS_DELAY_PS for DLL delay setting. Changed to use feature FSL_FEATURE_FLEXSPI_DQS_DELAY_MIN to set slave delay target as 0 for DLL enable and clock frequency higher than 100MHz.
- 2.3.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 8.4, 8.5, 10.1, 10.3, 10.4, 11.6 and 14.4.
- 2.3.1
  - Bug Fixes
    - Wait for bus to be idle before using it as access to external flash with new setting in FLEXSPI_SetFlashConfig() API.
    - Fixed the potential buffer overread and Tx FIFO overwrite issue in FLEXSPI_WriteBlocking.
- 2.3.0
  - New Features
    - Added new API FLEXSPI_UpdateDllValue for users to update DLL value after updating flexspi root clock.
    - Corrected grammatical issues for comments.
    - Added support for new feature FSL_FEATURE_FLEXSPI_DQS_DELAY_PS in DLL configuration.
- 2.2.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 10.1, 10.3 and 10.4.
    - Updated _flexspi_command from named enumerator into anonymous enumerator.
- 2.2.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 10.1, 10.3, 10.4, 10.8, 11.9, 14.4, 15.7, 16.4, 17.7, 7.3.
    - Fixed IAR build warning Pe167.
    - Fixed the potential buffer overwrite and Rx FIFO overread issue in FLEXSPI_ReadBlocking.
- 2.2.0
  - Bug Fixes
    - Fixed flag name typos: kFLEXSPI_IpTxFifoWatermarkEmpltyFlag to kFLEXSPI_IpTxFifoWatermarkEmptyFlag; kFLEXSPI_IpCommandExcutionDoneFlag to kFLEXSPI_IpCommandExecutionDoneFlag.
    - Fixed comments typos such as sequencen->sequence, levle->level.
    - Fixed FLSHCR2[ARDSEQID] field clean issue.
    - Updated flexspi_config_t structure and FlexSPI_Init to support new feature FSL_FEATURE_FLEXSPI_HAS_NO_MCR0_ATDFEN and FSL_FEATURE_FLEXSPI_HAS_NO_MCR0_ARDFEN.
    - Updated flexspi_flags_t structure to support new feature FSL_FEATURE_FLEXSPI_HAS_INTEN_AHBBUSERROREN.
- 2.1.1
  - Improvements

    - ∗ Defaulted enable prefetch for AHB RX buffer configuration in FLEXSPI_GetDefault-Config, which is align with the reset value in AHBRXBUFxCR0.
    - ∗ Added software workaround for ERR011377 in FLEXSPI_SetFlashConfig; added some delay after DLL lock status set to ensure correct data read/write.
- 2.1.0
  - New Features
    - ∗ Added new API FLEXSPI_UpdateRxSampleClock for users to update read sample clock source after initialization.
    - ∗ Added reset peripheral operation in FLEXSPI_Init if required.
- 2.0.5
  - Bug Fixes
    - ∗ Fixed FLEXSPI_UpdateLUT cannot do partial update issue.
- 2.0.4
  - Bug Fixes
    - ∗ Reset flash size to zero for all ports in FLEXSPI_Init; fixed the possible out-of-range flash access with no error reported.
- 2.0.3
  - Bug Fixes
    - ∗ Fixed AHB receive buffer size configuration issue. The FLEXSPI_AHBRXBUFCR0_-BUFSZ field should configure 64 bits size, and currently the AHB receive buffer size is in bytes which means 8-bit, so the correct configuration should be config->ahbConfig.-buffer[i].bufferSize / 8.
- 2.0.2
  - New Features
    - ∗ Supported DQS write mask enable/disable feature during set FLEXSPI configuration.
    - ∗ Provided new API FLEXSPI_TransferUpdateSizeEDMA for users to update eDMA transfer size(SSIZE/DSIZE) per DMA transfer.
  - Bug Fixes
    - ∗ Fixed invalid operation of FLEXSPI_Init to enable AHB bus Read Access to IP RX FIFO.
    - ∗ Fixed incorrect operation of FLEXSPI_Init to configure IP TX FIFO watermark.
- 2.0.1
  - Bug Fixes
    - ∗ Fixed the flag clear issue and AHB read Command index configuration issue in FLEXSP-I_SetFlashConfig.
    - ∗ Updated FLEXSPI_UpdateLUT function to update LUT table from any index instead of previous command index.
    - ∗ Added bus idle wait in FLEXSPI_SetFlashConfig and FLEXSPI_UpdateLUT to ensure bus is idle before any change to FlexSPI controller.
    - ∗ Updated interrupt API FLEXSPI_TransferNonBlocking and interrupt handle flow FLEX-SPI_TransferHandleIRQ.
    - ∗ Updated eDMA API FLEXSPI_TransferEDMA.
- 2.0.0
  - Initial version.

FLEXSPI DMA Driver

- 2.2.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 10.1, 10.3, 10.4, 10.8.
- 2.2.0
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 Rule 10.1, 10.3.
  - New Features
    - Updated name of FLEXSPI_TransferGetTransferCountDMA API.
- 2.1.1
  - New Features
    - Updated driver to support feature FSL_FEATURE_FLEXSPI_DMA_MULTIPLE_DES.
- 2.1.0
  - Bug Fixes
    - Updated enumaration flexspi_dma_transfer_nsize_t and remove the unsupported items.
  - New Features
    - Updated driver for deprecating the multiple linked descriptors inside FLEXSPI_TransferDMA, only up to one linked descriptor is needed according to hardware update.
- 2.0.0
  - Initial version.

FMEAS

The current FMEAS driver version is 2.1.1.

- 2.1.1
  - Bug Fixes
    - MISRA C-2012 issues fixed: rule 10.4, rule 10.8.
- 2.1.0
  - Updated "FMEAS_GetFrequency","FMEAS_StartMeasure","FMEAS_IsMeasureComplete" API and add definition to match ASYNC_SYSCON.
- 2.0.0
  - Initial version ported from LPCOpen.

I3C

The current I3C driver version is 2.9.0.

- 2.9.0
  - Improvements
    - Added adaptive termination for master blocking transfer. Set termination with start signal when receiving bytes less than 256.
- 2.8.1

  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 17.7.
- 2.8.0
  - Improvements
    - Added API I3C_MasterProcessDAASpecifiedBaudrate for temporary baud rate adjustment when I3C master assigns dynamic address.
- 2.7.1
  - Bug Fixes
    - Fixed the issue that I3C slave handle STOP event before finishing data transmission.
- 2.7.0
  - Fixed the CCM problem in file fsl_i3c.c.
  - Fixed the FSL_FEATURE_I3C_HAS_NO_SCONFIG_IDRAND usage issue in I3C_GetDefaultConfig and I3C_Init.
- 2.6.0
  - Fixed the FSL_FEATURE_I3C_HAS_NO_SCONFIG_IDRAND usage issue in fsl_i3c.h.
  - Changed some static functions in fsl_i3c.c as non-static and define the functions in fsl_i3c.h to make I3C DMA driver reuse:
    - I3C_GetIBIType
    - I3C_GetIBIAddress
    - I3C_SlaveCheckAndClearError
  - Changed the handle pointer parameter in IRQ related funtions to void * type to make it reuse in I3C DMA driver.
  - Added new API I3C_SlaveRequestIBIWithSingleData for slave to request single data byte, this API could be used regardless slave is working in non-blocking interrupt or non-blocking dma.
  - Added new API I3C_MasterGetDeviceListAfterDAA for master application to get the device information list built up in DAA process.
- 2.5.4
  - Improved I3C driver to avoid setting state twice in the SendCommandState of I3C_RunTransferStateMachine.
  - Fixed MISRA violation of rule 20.9.
  - Fixed the issue that I3C_MasterEmitRequest did not use Type I3C SDR.
- 2.5.3
  - Updated driver for new feature FSL_FEATURE_I3C_HAS_NO_SCONFIG_BAMATCH and FSL_FEATURE_I3C_HAS_NO_SCONFIG_IDRAND.
- 2.5.2
  - Updated driver for new feature FSL_FEATURE_I3C_HAS_NO_MERRWARN_TERM.
  - Fixed the issue that call to I3C_MasterTransferBlocking API did not generate STOP signal when NAK status was returned.
- 2.5.1
  - Improved the receive terminate size setting for interrupt transfer read, now it's set at beginning of transfer if the receive size is less than 256 bytes.
- 2.5.0
  - Added new API I3C_MasterRepeatedStartWithRxSize to send repeated start signal with receive terminate size specified.

- Fixed the status used in I3C_RunTransferStateMachine, changed to use pending interrupts as status to be handled in the state machine.
- Fixed MISRA 2012 violation of rule 10.3, 10.7.

- 2.4.0
  - Bug Fixes
    * Fixed kI3C_SlaveMatchedFlag interrupt is not properly handled in I3C_SlaveTransfer-HandleIRQ when it comes together with interrupt kI3C_SlaveBusStartFlag.
    * Fixed the inaccurate I2C baudrate calculation in I3C_MasterSetBaudRate.
    * Added new API I3C_MasterGetIBIRules to get registered IBI rules.
    * Added new variable isReadTerm in struct _i3c_master_handle for transfer state routine to check if MCTRL.RDTERM is configured for read transfer.
    * Changed to emit Auto IBI in transfer state routine for slave start flag assertion.
    * Fixed the slave maxWriteLength and maxReadLength does not be configured into SMA-XLIMITS register issue.
    * Fixed incorrect state for IBI in I3C master interrupt transfer IRQ handle routine.
    * Added isHotJoin in i3c_slave_config_t to request hot-join event during slave init.
- 2.3.2
  - Bug Fixes
    * Fixed violations of the MISRA C-2012 rules 8.4, 17.7.
    * Fixed incorrect HotJoin event index in I3C_GetIBIType.
- 2.3.1
  - Bug Fixes
    * Fixed the issue that call of I3C_MasterTransferBlocking/I3C_MasterTransferNon-Blocking fails for the case which receive length 1 byte of data.
    * Fixed the issue that STOP signal is not sent when NAK status is detected during execution of I3C_MasterTransferBlocking function.
- 2.3.0
  - Improvements
    * Added I3C common driver APIs to initialize I3C with both master and slave configuration.
    * Updated I3C master transfer callback to function set structure to include callback invoke for IBI event and slave2master event.
    * Updated I3C master non-blocking transfer model and always enable the interrupts to be able to re-act to the slave start event and handle slave IBI.
- 2.2.0
  - Bug Fixes
    * Fixed the issue that I3C transfer size limit to 255 bytes.
- 2.1.2
  - Bug Fixes
    * Reset default hkeep value to kI3C_MasterHighKeeperNone in I3C_MasterGetDefault-Config
- 2.1.1
  - Bug Fixes
    * Fixed incorrect FIFO reset operation in I3C Master Transfer APIs.
    * Fixed i3c slave IRQ handler issue, slave transmit could be underrun because tx FIFO is not filled in time right after start flag detected.

- 2.1.0
  - Added definitions and APIs for I3C slave functionality, updated previous I3C APIs to support I3C functionality.
- 2.0.0
  - Initial version.

IAP

The current IAP driver version is 2.1.3.

- 2.1.3
  - Bug Fixes
    - Fixed misra issue.
- 2.1.2
  - Bug Fixes
    - Fixed some macro undefined issue.
    - Put IAP_FlexspiNorInit API into RAM.
- 2.1.1
  - Bug Fixes
    - Fixed misra issue.
- 2.1.0
  - New Features
    - Added IAP_RunBootLoader() API
- 2.0.2
  - Bug Fixes
    - Fixed doxygen issue.
- 2.0.1
  - Bug Fixes
    - Minor update for MISRA issue fix.
- 2.0.0
  - Initial version.

INPUTMUX

The current INPUTMUX driver version is 2.0.4.

- 2.0.5
  - Bug Fixes
    - Fixed build error because some devices has no sct.
- 2.0.4
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rule 10.4, 12.2 in INPUTMUX_EnableSignal() function.
- 2.0.3

- Bug Fixes
  - Fixed violations of the MISRA C-2012 rules 10.4, 10.7, 12.2.
- 2.0.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.4, 12.2.
- 2.0.1
  - Support channel mux setting in INPUTMUX_EnableSignal().
- 2.0.0
  - Initial version.

LCDIF

The current LCDIF driver version is 2.1.1.

- 2.1.1
  - Improvements
    - Added memory address conversion to support buffers which could only be accessed using alias address by non-core masters.
  - Bug Fixes
    - Fix MISRA-C 2012 issues.
- 2.1.0
  - Bug Fixes
    - Corrected the frame buffer pixel format name.
- 2.0.0
  - Initial version.

LPADC

The current LPADC driver version is 2.6.1.

- 2.6.1
  - Improvements
    - Updated the use of macros in the driver code.
- 2.6.0
  - Improvements
    - Added the API LPADC_SetOffset12BitValue() to configure 12bit ADC conversion offset trim value manually.
    - Added the API LPADC_SetOffset16BitValue() to configure 16bit ADC conversion offset trim value manually.
    - Added API to set offset calibration mode.
    - Added configuration of alternate channel.
    - Updated auto calibration API and added calibration value conversion API.
  - New feature
    - Added API LPADC_EnableHardwareTriggerCommandSelection() to enable trigger

commands controlled by ADC_ETC.

- 2.5.1
  - Bug Fixes
    - Fixed some typos in Lpadc driver comments.
- 2.5.0
  - Improvements
    - Added missing items to enable trigger interrupts.
- 2.4.0
  - New features
    - Added APIs to get/clear trigger status flags.
- 2.3.0
  - Improvements
    - Removed LPADC_MeasureTemperature() function for the LPADC supports different temperature sensor calculation equations.
- 2.2.1
  - Improvements
    - Optimized LPADC_MeasureTemperature() function to support the specific series with flash solidified calibration value.
    - Clean doxygen warnings.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.3, rule 10.8 and rule 17.7.
- 2.2.0
  - New Feature
    - Added API LPADC_MeasureTemperature() to get correct temperature from the internal sensor.
  - Improvements
    - Separated lpadc_conversion_resolution_mode_t with related feature macro.
  - Bug Fixes
    - Fixed the violations of MISRA C-2012 rules:
      - Rule 10.3, 10.4, 10.6, 10.7 and 17.7.
- 2.1.1
  - Improvements
    - Updated the gain calibration formula.
    - Used feature to segregate the new item kLPADC_TriggerPriorityPreemptSubsequently.
- 2.1.0
  - New Features
    - Added the API LPADC_SetOffsetValue() to support configure offset trim value manually.
    - Added the API LPADC_DoOffsetCalibration() to do offset calibration independently.
  - Improvements
    - Improved the usage of macros and removed invalid macros.
- 2.0.2
  - Improvements
    - Added support for platforms with 2 FIFOs and different calibration measures.
- 2.0.1
  - Bug Fixes

    - Ensured the API LPADC_SetConvCommandConfig configure related registers correctly.
- 2.0.0
  - Initial version.

CRC

The current CRC driver version is 2.1.1.

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

DMA

The current DMA driver version is 2.5.0.

- 2.5.0
  - Improvements
    - Added a new api DMA_SetChannelXferConfig to set DMA xfer config.
- 2.4.4
  - Bug Fixes
    - Fixed the issue that DMA_IRQHandle might generate redundant callbacks.
    - Fixed the issue that DMA driver cannot support channel bigger then 32.
    - Fixed violation of the MISRA C-2012 rule 13.5.
- 2.4.3
  - Improvements
    - Added features FSL_FEATURE_DMA_DESCRIPTOR_ALIGN_SIZEn/FSL_FEATU-RE_DMA0_DESCRIPTOR_ALIGN_SIZE/FSL_FEATURE_DMA1_DESCRIPTOR_-ALIGN_SIZE to support the descriptor align size not constant in the two instances.
- 2.4.2
  - Bug Fixes
    - Fixed violation of the MISRA C-2012 rule 8.4.
- 2.4.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 5.7, 8.3.
- 2.4.0

  - Improvements
    - Added new APIs DMA_LoadChannelDescriptor/DMA_ChannelIsBusy to support polling transfer case.
  - Bug Fixes
    - Added address alignment check for descriptor source and destination address.
    - Added DMA_ALLOCATE_DATA_TRANSFER_BUFFER for application buffer allocation.
    - Fixed the sign-compare warning.
    - Fixed violations of the MISRA C-2012 rules 18.1, 10.4, 11.6, 10.7, 14.4, 16.3, 20.7, 10.8, 16.1, 17.7, 10.3, 3.1, 18.1.
- 2.3.0
  - Bug Fixes
    - Removed DMA_HandleIRQ prototype definition from header file.
    - Added DMA_IRQHandle prototype definition in header file.
- 2.2.5
  - Improvements
    - Added new API DMA_SetupChannelDescriptor to support configuring wrap descriptor.
    - Added wrap support in function DMA_SubmitChannelTransfer.
- 2.2.4
  - Bug Fixes
    - Fixed the issue that macro DMA_CHANNEL_CFER used wrong parameter to calculate DSTINC.
- 2.2.3
  - Bug Fixes
    - Improved DMA driver Deinit function for correct logic order.
  - Improvements
    - Added API DMA_SubmitChannelTransferParameter to support creating head descriptor directly.
    - Added API DMA_SubmitChannelDescriptor to support ping pong transfer.
    - Added macro DMA_ALLOCATE_HEAD_DESCRIPTOR/DMA_ALLOCATE_LINK_-DESCRIPTOR to simplify DMA descriptor allocation.
- 2.2.2
  - Bug Fixes
    - Do not use software trigger when hardware trigger is enabled.
- 2.2.1
  - Bug Fixes
    - Fixed Coverity issue.
- 2.2.0
  - Improvements
    - Changed API DMA_SetupDMADescriptor to non-static.
    - Marked APIs below as deprecated.
      - DMA_PrepareTransfer.
      - DMA_Submit transfer.
    - Added new APIs as below:
      - DMA_SetChannelConfig.

      - DMA_PrepareChannelTransfer.
      - DMA_InstallDescriptorMemory.
      - DMA_SubmitChannelTransfer.
      - DMA_SetChannelConfigValid.
      - DMA_DoChannelSoftwareTrigger.
      - DMA_LoadChannelTransferConfig.
- 2.0.1
  - Improvements
    - Added volatile for DMA descriptor member xfercfg to avoid optimization.
- 2.0.0
  - Initial version.

GPIO

The current GPIO driver version is 2.1.7.

- 2.1.7
  - Improvements
    - Enhanced GPIO_PinInit to enable clock internally.
- 2.1.6
  - Bug Fixes
    - Clear bit before set it within GPIO_SetPinInterruptConfig() API.
- 2.1.5
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 3.1, 10.6, 10.7, 17.7.
- 2.1.4
  - Improvements
    - Added API GPIO_PortGetInterruptStatus to retrieve interrupt status for whole port.
    - Corrected typos in header file.
- 2.1.3
  - Improvements
    - Updated "GPIO_PinInit" API. If it has DIRCLR and DIRSET registers, use them at set 1 or clean 0.
- 2.1.2
  - Improvements
    - Removed deprecated APIs.
- 2.1.1
  - Improvements
    - API interface changes:
      - Refined naming of APIs while keeping all original APIs, marking them as deprecated. Original APIs will be removed in next release. The mainin change is updating APIs with prefix of _PinXXX() and _PorortXXX
- 2.1.0
  - New Features

    - Added GPIO initialize API.
- 2.0.0
  - Initial version.

IOPCTL

The current IOPCTL driver version is 2.0.0.

- 2.0.0
  - Initial version.

RTC

The current RTC driver version is 2.1.3.

- 2.1.3
  - Bug Fixes
    - Fixed issue that RTC_GetWakeupCount may return wrong value.
- 2.1.2
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.1, 10.4 and 10.7.
- 2.1.1
  - Bug Fixes
    - MISRA C-2012 issue fixed: rule 10.3 and 11.9.
- 2.1.0
  - Bug Fixes
    - Created new APIs for the RTC driver.
      - RTC_EnableTimer
      - RTC_EnableWakeUpTimerInterruptFromDPD
      - RTC_EnableAlarmTimerInterruptFromDPD
      - RTC_EnableWakeupTimer
      - RTC_GetEnabledWakeupTimer
      - RTC_SetSecondsTimerMatch
      - RTC_GetSecondsTimerMatch
      - RTC_SetSecondsTimerCount
      - RTC_GetSecondsTimerCount
    - deprecated legacy APIs for the RTC driver.
      - RTC_StartTimer
      - RTC_StopTimer
      - RTC_EnableInterrupts
      - RTC_DisableInterrupts
      - RTC_GetEnabledInterrupts
- 2.0.0
  - Initial version.

MIPI_DSI

The current MIPI_DSI driver version is 2.1.4.

- 2.1.4
  - Bug Fixes
    - Fixed the MISRA issues.
- 2.1.3
  - Bug Fixes
    - Fixed the DPI horizontal timing setting issue.
- 2.1.2
  - Improvements
    - Supported long package read.
  - Bug Fixes
    - Fixed the bug that runs to hardfault when sending long packet with 4-byte unaligned address.
- 2.1.1
  - Improvements
    - Some SOC compatibility improvement.
- 2.1.0
  - Improvements
    - Improved for the platforms which does not support ULPS.
- 2.0.6
  - Bug Fixes
    - Fixed the timing issue that non-continuous HS clock mode does not work.
- 2.0.5
  - Bug Fixes
    - Fixed kDSI_InterruptGroup1BtaTo and kDSI_InterruptGroup1HtxTo definition error.
  - Improvements
    - Changed to override MIPI_DriverIRQHandler instead of MIPI_IRQHandler.
- 2.0.4
  - Bug Fixes
    - Fixed MISRA C-2012 issues: 10.1, 10.3, 10.4, 10.4, 10.6, 10.7, 10.8, 11.3, 11.8, 12.2, 14.4, 16.4, 17.7.
- 2.0.3
  - Improvement
    - Updated for combo phy header file.
- 2.0.2
  - New Features
    - Supported sending separate DSI command from TX data array.
  - Bug Fixes
    - Disabled all interrupts in DSI_Init.
- 2.0.1
  - Improvements
    - Updated to support the DPHY which does not have internal DPHY PLL.
- 2.0.0

- Initial version.

MIPI_DSI_SMARTDMA

The current MIPI_DSI driver version is 2.3.0.

- 2.3.0
  - New Features
    - Updated DSI_TransferWriteMemorySMARTDMA, dsi_smartdma_write_mem_transfer-_t and dsi_smartdma_handle_t to support 2-dimensional data transfer for interleaved pixels.
- 2.2.1
  - Bug Fixes
    - Fixed MISRA C-2012 issues: 10.1, 10.3, 11.3, 11.8, 14.4, 17.7.
- 2.2.0
  - Improvements
    - Supported swap or don't swap the pixel byte before written to MIPI DSI FIFO.
- 2.1.0
  - Improvements
    - Supported frame buffer format XRGB8888.
    - Added virtual channel setting in dsi_smartdma_write_mem_transfer_t, current driver only support channel 0, added for future enhancement.
- 2.0.1
  - Bug Fixes
    - Fixed the issue that driver handle not set to busy during transfer.
- 2.0.0
  - Initial version.

MRT

The current MRT driver version is 2.0.4.

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

- 2.0.0
  - Initial version.

MU

The Current MU driver version is 2.1.1.

- 2.1.1
  - Bug Fixes
    - Fixed general interrupt comment typo.
- 2.1.0
  - Improvements
    - Added new enum mu_msg_reg_index_t.
- 2.0.7
  - Bug Fixes
    - Fixed MU_GetInterruptsPending bug that can not get general interrupt status.
- 2.0.6
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 17.7.
- 2.0.5
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 14.4, 15.5.
- 2.0.4
  - Improvements
    - Improved for the platforms which don't support reset assert interrupt and get the other core power mode.
- 2.0.3
  - Bug fixes
    - MISRA C-2012 issue fixed.
      - Fixed rules, containing: rule-10.3, rule-14.4, rule-15.5.
- 2.0.2
  - Improvements
    - Added support for MIMX8MQx.
- 2.0.1
  - Improvements
    - Added support for MCIMX7Ux_M4.
- 2.0.0
  - Initial version.

OSTIMER

The current OSTIMER driver version is 2.2.0.

- 2.2.0

  - Improvements
    - Move the PMC operation out of the OSTIMER driver to board specific files.
    - Added low level APIs to control OSTIMER MATCH and interrupt.
- 2.1.2
  - Bug Fixes
    - Fixed MISRA-2012 rule 10.8.
- 2.1.1
  - Bug Fixes
    - removes the suffix 'n' for some register names and bit fields' names
  - Improvements
    - Added HW CODE GRAY feature supported by CODE GRAY in SYSCTRL register group.
- 2.1.0
  - Bug Fixes
    - Added a workaround to fix the issue that no interrupt was reported when user set smaller period.
    - Fixed violation of MISRA C-2012 rule 10.3 and 11.9.
  - Improvements
    - Added return value for the two APIs to set match value.
      - OSTIMER_SetMatchRawValue
      - OSTIMER_SetMatchValue
- 2.0.3
  - Bug Fixes
    - Fixed violation of MISRA C-2012 rule 10.3, 14.4, 17.7.
- 2.0.2
  - Improvements
    - Added support for OSTIMER0
- 2.0.1
  - Improvements
    - Removed the software reset function out of the initialization API.
    - Enabled interrupt directly instead of enabling deep sleep interrupt. Users need to enable the deep sleep interrupt in application code if needed.
- 2.0.0
  - Initial version.

OTFAD

The current driver version is 2.1.4.

- 2.1.4
  - Bug fixes
    - Fixed MISRA 2012 issue: 10.1.
- 2.1.3
  - Bug fixes

    - Fixed the error that waiting for both FLEXSPI AHB idle and SEQ idle.
- 2.1.2
  - Bug fixes
    - Fixed MISRA 2012 issue: 10.4.
- 2.1.1
  - Improvements:
    - Hided some bits in CR and SR registers for selected platforms.
    - Fixed doxygen issues.
- 2.1.0
  - Improvements:
    - Used boolean type to define 1-bit field concepts.
- 2.0.0
  - Initial version.

PINT

The current PINT driver version is 2.1.9.

- 2.1.9
  - Bug Fixes
    - Fixed MISRA-2012 rule 8.4.
- 2.1.8
  - Bug Fixes
    - Fixed MISRA-2012 rule 10.1 rule 10.4 rule 10.8 rule 18.1 rule 20.9.
- 2.1.7
  - Improvements
    - Added fully support for the SECPINT, making it can be used just like PINT.
- 2.1.6
  - Bug Fixes
    - Fixed the bug of not enabling common pint clock when enabling security pint clock.
- 2.1.5
  - Bug Fixes
    - Fixed issue for MISRA-2012 check.
      - Fixed rule 10.1 rule 10.3 rule 10.4 rule 10.8 rule 14.4.
    - Changed interrupt init order to make pin interrupt configuration more reasonable.
- 2.1.4
  - Improvements
    - Added feature to control distinguish PINT/SECPINT relevant interrupt/clock configurations for PINT_Init and PINT_Deinit API.
    - Swapped the order of clearing PIN interrupt status flag and clearing pending NVIC interrupt in PINT_EnableCallback and PINT_EnableCallbackByIndex function.
    - Bug Fixes
      - Fixed build issue caused by incorrect macro definitions.
- 2.1.3

  - Bug fix:
    - Updated PINT_PinInterruptClrStatus to clear PINT interrupt status when the bit is asserted and check whether was triggered by edge-sensitive mode.
    - Write 1 to IST corresponding bit will clear interrupt status only in edge-sensitive mode and will switch the active level for this pin in level-sensitive mode.
    - Fixed MISRA c-2012 rule 10.1, rule 10.6, rule 10.7.
    - Added FSL_FEATURE_SECPINT_NUMBER_OF_CONNECTED_OUTPUTS to distinguish IRQ relevant array definitions for SECPINT/PINT on lpc55s69 board.
    - Fixed PINT driver c++ build error and remove index offset operation.
- 2.1.2
  - Improvement:
    - Improved way of initialization for SECPINT/PINT in PINT_Init API.
- 2.1.1
  - Improvement:
    - Enabled secure pint interrupt and add secure interrupt handle.
- 2.1.0
  - Added PINT_EnableCallbackByIndex/PINT_DisableCallbackByIndex APIs to enable/disable callback by index.
- 2.0.2
  - Added control macro to enable/disable the RESET and CLOCK code in current driver.
- 2.0.1
  - Bug fix:
    - Updated PINT driver to clear interrupt only in Edge sensitive.
- 2.0.0
  - Initial version.

POWERQUAD

The current POWERQUAD driver version is 2.1.0.

- 2.1.0
  - Improvements
    - Fixed typo issue for biquad related function name.
    - Changed operator from "%" into "&" to reduce heavy cycle for biquad functions.
- 2.0.5
  - Improvements
    - Added a note in driver for FIR that powerquad has a hardware limitation, when using it for FIR increment calculation, the address of pSrc needs to be a continuous address.
- 2.0.4
  - Improvements
    - Supported the platforms which don't have PowerQuad clock and reset control.
- 2.0.3
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 8.4, 10.1, 10.3, 10.4, 10.6, and so on.

- 2.0.2
  - Bug Fixes
    - Fixed array size issue in fsl_powerquad_data.h file.
    - Fixed vector function pipeline issue.
- 2.0.1
  - Bug Fixes
    - Fixed build error in C++ mode.
- 2.0.0
  - Initial version.

PUF

The current PUF driver version is 2.1.6.

- 2.1.6
  - Changed wait time in PUF_Init(), when initialization fails it will try PUF_Powercycle() with shorter time. If this shorter time will also fail, initialization will be tried with worst case time as before.
- 2.1.5
  - Use common SDK delay in puf_wait_usec().
- 2.1.4
  - Replace register uint32_t ticksCount with volatile uint32_t ticksCount in puf_wait_usec() to prevent optimization out delay loop.
- 2.1.3
  - Fix MISRA C-2012 issue.
- 2.1.2
  - Update: Add automatic big to little endian swap for user (pre-shared) keys destinated to secret hardware bus (PUF key index 0).
- 2.1.1
  - Fix ARMGCC build warning .
- 2.1.0
  - Align driver with PUF SRAM controller registers on LPCXpresso55s16.
  - Update initizalition logic .
- 2.0.3
  - Fix MISRA C-2012 issue.
- 2.0.2
  - New feature:
    - Add PUF configuration structure and support for PUF SRAM controller.
  - Improvements:
    - Remove magic constants.
- 2.0.1
  - Bug Fixes:
    - Fixed puf_wait_usec function optimization issue.
- 2.0.0

  - Initial version.

SCTIMER

The current SCTimer driver version is 2.4.8.

- 2.4.8
  - Bug Fixes
    - Fixed the issue that the SCTIMER_UpdatePwmDutycycle() can't writes MATCH_H bit and RELOADn_H.
- 2.4.7
  - Bug Fixes
    - Fixed the issue that the SCTIMER_UpdatePwmDutycycle() can't configure 100% duty cycle PWM.
- 2.4.6
  - Bug Fixes
    - Fixed the issue where the H register was not written as a word along with the L register.
    - Fixed the issue that the SCTIMER_SetCOUNTValue() is not configured with high 16 bits in unify mode.
- 2.4.5
  - Bug Fixes
    - Fix SCT_EV_STATE_STATEMSKn macro build error.
- 2.4.4
  - Bug Fixes
    - Fix MISRA C-2012 issue 10.8.
- 2.4.3
  - Bug Fixes
    - Fixed the wrong way of writing CAPCTRL and REGMODE registers in SCTIMER_-SetupCaptureAction.
- 2.4.2
  - Bug Fixes
    - Fixed SCTIMER_SetupPwm 100% duty cycle issue.
- 2.4.1
  - Bug Fixes
    - Fixed the issue that MATCHn_H bit and RELOADn_H bit could not be written.
- 2.4.0
- 2.3.0
  - Bug Fixes
    - Fixed the potential overflow issue of pulseperiod variable in SCTIMER_SetupPwm/SCT-IMER_UpdatePwmDutycycle API.
    - Fixed the issue of SCTIMER_CreateAndScheduleEvent API does not correctly work with 32 bit unified counter.
    - Fixed the issue of position of clear counter operation in SCTIMER_Init API.
  - Improvements

        - Update SCTIMER_SetupPwm/SCTIMER_UpdatePwmDutycycle to support generate 0% and 100% PWM signal.
        - Add SCTIMER_SetupEventActiveDirection API to configure event activity direction.
        - Update SCTIMER_StartTimer/SCTIMER_StopTimer API to support start/stop low counter and high counter at the same time.
        - Add SCTIMER_SetCounterState/SCTIMER_GetCounterState API to write/read counter current state value.
        - Update APIs to make it meaningful.
            - SCTIMER_SetEventInState
            - SCTIMER_ClearEventInState
            - SCTIMER_GetEventInState
- 2.2.0
    - Improvements
        - Updated for 16-bit register access.
- 2.1.3
    - Bug Fixes
        - Fixed the issue of uninitialized variables in SCTIMER_SetupPwm.
        - Fixed the issue that the Low 16-bit and high 16-bit work independently in SCTIMER driver.
    - Improvements
        - Added an enumerable macro of unify counter for user.
            - kSCTIMER_Counter_U
        - Created new APIs for the RTC driver.
            - SCTIMER_SetupStateLdMethodAction
            - SCTIMER_SetupNextStateActionwithLdMethod
            - SCTIMER_SetCOUNTValue
            - SCTIMER_GetCOUNTValue
            - SCTIMER_SetEventInState
            - SCTIMER_ClearEventInState
            - SCTIMER_GetEventInState
        - Deprecated legacy APIs for the RTC driver.
            - SCTIMER_SetupNextStateAction
- 2.1.2
    - Bug Fixes
        - MISRA C-2012 issue fixed: rule 10.3, 10.4, 10.6, 10.7, 11.9, 14.2 and 15.5.
- 2.1.1
    - Improvements
        - Updated the register and macro names to align with the header of devices.
- 2.1.0
    - Bug Fixes
        - Fixed issue where SCT application level Interrupt handler function is occupied by SCT driver.
        - Fixed issue where wrong value for INSYNC field inside SCTIMER_Init function.
        - Fixed issue to change Default value for INSYNC field inside SCTIMER_GetDefault-Config.

- 2.0.1
  - New Features
    - Added control macro to enable/disable the RESET and CLOCK code in current driver.
- 2.0.0
  - Initial version.

SEMA42

The current SEMA42 driver version is 2.0.3.

- 2.0.3
  - Improvements
    - Changed to implement SEMA42_Lock base on SEMA42_TryLock.
- 2.0.2
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 17.7.
- 2.0.1
  - Bug Fixes
    - Fixed violations of the MISRA C-2012 rules 10.3, 10.4, 14.4, 18.1.
- 2.0.0
  - Initial version.

SMARTDMA

The current SMARTDMA driver version is 2.7.0.

- 2.7.0
  - New Features
    - Supported data transfer in 2-dimensional way.
    - Supported data transfer in XRGB8888 format and rotate 180 degree.
    - Supported to fill data in whenever there is room in MIPI controller's FIFO rather than using the tx FIFO in double buffered way.
- 2.6.3
  - Bug Fixes
    - Fixed EZH_MIPIDSI_RGB565_DMA, EZH_MIPIDSI_RGB888_DMA, EZH_MIPIDS-I_ARGB8888toRGB888_DMA issues that don't support some length.
- 2.6.2
  - Bug Fixes
    - Fixed MISRA C-2012 issues: 8.4, 11.6, 17.7.
- 2.6.1
  - Improvements
    - Optimized MIPI DSI APIs performance.
- 2.6.0
  - Improvements

    - Optimized MIPI DSI APIs performance.
  - New Features
    - Added new APIs to send MIPI DSI frame with 180 degree rotation.
- 2.5.0
  - Improvements
    - Supported swap or don't swap the pixel byte before written to MIPI DSI FIFO.
    - Updated MIPI DSI firmware, make sure data has been sent out before calling callback function.
- 2.4.0
  - Improvements
    - Added new APIs for MIPI DSI kSMARTDMA_MIPI_XRGB2RGB_DMA.
- 2.3.0
  - Improvements
    - Added new APIs for FlexIO one SHIFTBUF, kSMARTDMA_FlexIO_DMA_ONELAN-E.
  - Bug Fixes
    - Fixed kSMARTDMA_MIPI_RGB565_DMA color bias issue.
- 2.2.0
  - Improvements
    - Added new APIs for MIPI DSI, kSMARTDMA_MIPI_RGB565_DMA and kSMARTD-MA_MIPI_RGB888_DMA.
    - Supported install firmware and callback function dynamically.
- 2.1.0
  - Improvements
    - Removed test APIs, including kSMARTDMA_LightOn, kSMARTDMA_LightOff, kS-MARTDMA_Notify, and kSMARTDMA_Test.
    - Added new APIs, including kSMARTDMA_FlexIO_DMA_Reverse, kSMARTDMA-_FlexIO_DMA_ARGB2RGB, kSMARTDMA_FlexIO_DMA_ARGB2RGB_Endian_-Swap, and kSMARTDMA_FlexIO_DMA_ARGB2RGB_Endian_Swap_Reverse.
- 2.0.0
  - Initial version.

USDHC

The current USDHC driver version is 2.8.2.

- 2.8.2
  - Improvements
    - Added feature macro FSL_FEATURE_USDHC_HAS_NO_VOLTAGE_SELECT.
- 2.8.1
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9.
- 2.8.0
  - Improvements

        - Fixed the mmc boot transfer failed issue which is caused by the Dma complete interrupt not enabled.
        - Marked api USDHC_AdjustDelayForManualTuning as deprecated and added new api USDHC_SetTuingDelay/USDHC_GetTuningDelayStatus.
        - Improved the manual tuning flow accroding to specification.
        - Added memory address conversion to support buffers which could only be accessed using alias address by non-core masters.
        - Fixed violations of MISRA C-2012 rule 10.4.
- 2.7.0
    - Improvements
        - Added api USDHC_TransferScatterGatherADMANonBlocking to support scatter gather transfer.
        - Added feature FSL_FEATURE_USDHC_REGISTER_HOST_CTRL_CAP_HAS_NO_-RETUNING_TIME_COUNTER for re-tuning time counter field in HOST_CTRL_CAP register.
    - Bug Fixes
        - Fixed violations of MISRA C-2012 rule 11.9, 10.1, 10.3, 10.4, 8.4.
- 2.6.0
    - Improvements
        - Added api USDHC_SetStandardTuningCounter to support adjust tuning counter of Standard tuning.
- 2.5.1
    - Improvements
        - Used different status code for command and data interrupt callback.
        - Added cache line invalidate for receive buffer in driver IRQ handler to fix CM7 speculative access issue.
- 2.5.0
    - Improvements
        - Added new api USDHC_SetStrobeDllOverride for HS400 strobe dll override mode delay taps configurations.
        - Corrected the STROBE DLL configurations sequence.
- 2.4.0
    - Improvements
        - Added feature macro for read/write burst length.
            - Disabled redundant interrupt per different transfer request.
            - Disabled interrupt and reset command/data pointer in handle when transfer completes.
    - Bug Fixes
        - Fixed violations of MISRA C-2012 rule 11.9, 15.7, 4.7, 16.4, 10.1, 10.3, 10.4, 11.3, 14.4, 10.6, 17.7, 16.1, 16.3.
        - Fixed PA082 build warning.
        - Fixed logically dead code Coverity issue.
- 2.3.0
    - Improvements
        - Added USDHC_SetDataConfig API to support manual tuning.
        - Removed the limitaion that source clock must be bigger than the target in function USD-

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

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

UTICK

The current UTICK driver version is 2.0.5.

- 2.0.5
  - Improvements
    - Improved for SOC RW610.
- 2.0.4
  - Bug Fixes
    - Fixed compile fail issue of no-supporting PD configuration in utick driver.
- 2.0.3
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rules: 8.4, 14.4, 17.7
- 2.0.2
  - Added new feature definition macro to enable/disable power control in drivers for some devices have no power control function.
- 2.0.1
  - Added control macro to enable/disable the CLOCK code in current driver.
- 2.0.0
  - Initial version.

WWDT

The current WWDT driver version is 2.1.9.

- 2.1.9
  - Bug Fixes
    - Fixed violation of the MISRA C-2012 rule 10.4.
- 2.1.8
  - Improvements
    - Updated the "WWDT_Init" API to add wait operation. Which can avoid the TV value read by CPU still be 0xFF (reset value) after WWDT_Init function returns.
- 2.1.7
  - Bug Fixes
    - Fixed the issue that the watchdog reset event affected the system from PMC.
    - Fixed the issue of setting watchdog WDPROTECT field without considering the backwards compatibility.
    - Fixed the issue of clearing bit fields by mistake in the function of WWDT_ClearStatus-Flags.
- 2.1.5
  - Bug Fixes
    - deprecated a unusable API in WWWDT driver.
      - WWDT_Disable
- 2.1.4
  - Bug Fixes
    - Fixed violation of the MISRA C-2012 rules Rule 10.1, 10.3, 10.4 and 11.9.
    - Fixed the issue of the inseparable process interrupted by other interrupt source.
      - WWDT_Init
- 2.1.3
  - Bug Fixes
    - Fixed legacy issue when initializing the MOD register.
- 2.1.2
  - Improvements
    - Updated the "WWDT_ClearStatusFlags" API and "WWDT_GetStatusFlags" API to match QN9090. WDTOF is not set in case of WD reset. Get info from PMC instead.
- 2.1.1
  - New Features
    - Added new feature definition macro for devices which have no LCOK control bit in MOD register.
    - Implemented delay/retry in WWDT driver.
- 2.1.0
  - Improvements
    - Added new parameter in configuration when initializing WWDT module. This parameter, which must be set, allows the user to deliver the WWDT clock frequency.
- 2.0.0
  - Initial version.

2 Middleware Change Log

DSP Audio Streamer

The current version of DSP Audio Streamer is 2.6p2.

- 2.6p2
  - Update to version 2.6p2 (2.6 patch)
- 2.6p1
  - Update to version 2.6p1 (2.6 patch release 1)
- 2.6
  - Update to version 2.6 GA
- 2.0
  - Initial version of DSP Audio Streamer

NatureDSP

The current version of NatureDSP is 1.2.0

- 1.2.0
  - please find the release notes in the doc/release_notes.txt
- 1.0.1
  - Initial version of NatureDSP from Cadence for FusionF1 DSP.

emWin library

The currently supported version is 6.28

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

  - upgraded to v6.10f

FatFs for MCUXpresso SDK

Current version is FatFs R0.14b_rev0.

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

LigJpeg for KSDK

Current version is LigJpeg 9b.

- 9b_rev1
  - New Feature:
    - The configuration file libjpeg/inc/jmorecfg.h could include user defined header file to

override pixel format configuration.
- 9b_rev0
  - Initial version. Changes when integrate with SDK:
    - In libjpeg/inc/jinclude.h line 88-96, map JFREAD and JFWRITE to FATFS f_read and f_write
    - In libjpeg/inc/jmorecfg.h line 397-406, change RGB color offset.
    - In libjpeg/src/jerror.c line 79-81, don't call function exit.

lwIP for MCUXpresso SDK

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

- 2.2.0_rev7
  - New features:
    - Ported lwIP 2.2.0.dev (2022-05-09, branch: master, SHA-1: 239918ccc173cb2c2a62f41a40fd893f57f to MCUXpresso SDK.
    - Added function ethernetif_probe_link() which reads actual link, speed and duplex settings from phy and passes them to driver. Stack could be set to call this function periodically by setting ETH_LINK_POLLING_INTERVAL_MS to value higher than zero.
    - Added helper functions ethernetif_wait_linkup() and ethernetif_wait_ipv4_valid() to allow blocking of RTOS task or bare metal application until link is up or IPv4 address becomes valid.
    - Added NETC adaptation layer.
    - Processing of rx packets under RTOS moved from ISR to a separate task to improve system reaction times. Switch back to old behavior can be done by setting ETH_DO_R-X_IN_SEPARATE_TASK macro to 0.
  - Bug fixes:
    - port: Fixed copying of pbuf contents. Previous code was using an incorrect end condition and could result in the overrun of the destination buffer if more packets were on the queue.
    - port: Delegating pbuf_free calls to tcpip_thread via pbuf_free_callback where possible (RTOS), ensured pbuf_free is not called from interrupt context when LWIP_ALLOW_-MEM_FREE_FROM_OTHER_CONTEXT is not set (bare metal).
    - port/enet_ethernetif_qos.c - Fixed ENET_RXBD_NUM which was used instead of ENE-T_TXBD_NUM.
    - port/enet_ethernetif_qos.c - Fixed buffer alignment to be at least 64.
    - src/apps/lwiperf: Fixed IPv6 TCP TX throughput lower than IPv4 by modifying maximum segment size to avoid sending two segments instead of one.
    - src/apps/lwiperf: Out-of-order datagrams in UDP RX server mode are counted to the throughput.

        * src/apps/httpsrv: Implemented receive timeouts on sockets.
        * src/apps/httpsrv: Don't assert on HTTP session task creation failure.
        * src/apps/httpsrv: Fixed build with IPv6 enabled.
        * src/apps/httpsrv: Updated endianess macros required for websocket SHA generation.
        * src/apps/httpsrv: Added missing includes.
- 2.2.0_rev6
    - New features:
        * Ported lwIP 2.2.0.dev (2022-03-25, branch: master, SHA-1: 124dc0a64ef5d7c14a27e3115e5888df65 to MCUXpresso SDK.
        * Implemented leaving of multicast groups on ENET and ENET QOS.
- 2.2.0_rev5
    - New features:
        * Ported lwIP 2.2.0.dev (2021-05-11, branch: master, SHA-1: 7ec4e9be304e7f8953740f10b2c810a292 to MCUXpresso SDK.
        * LPC ENET adaptation layer allocates more buffers for frame reception now. Previously the number of receive buffers was determined by ENET_RXBD_NUM, which defaults to 5. It is determined by ENET_RXBUFF_NUM now, which is 2 ∗ ENET_RXBD_NUM by default. Increase was needed because the actual version of LPC ENET driver always hold ENET_RXBD_NUM number of buffers and few additional buffers are needed for passing zero-copy frame data to lwIP. If this takes too much memory in your application, you can counteract by decreasing PBUF_POOL_SIZE, since PBUF_POOL is used only for transmission when LPC ENET, Kinetis ENET or ENET QOS is used.
- 2.2.0_rev4
    - New features:
        * Ported lwIP 2.2.0.dev (2021-03-05, branch: master, SHA-1: 0056522cc974d2be2005c324f37187b5b3 to KSDK 2.0.0.
        * LWIP_DHCP_DOES_ACD_CHECK option default changed to 0 (disabled):
            · Although the ACD check makes getting IP address from DHCP more robust, it added several seconds delay at startup of all applications which use DHCP.
            · This feature was not present in earlier versions of lwIP.
        * ENET QOS adaptation layer - implemented zero-copy on receive.
        * Kinetis ENET and ENET QOS adaptation layers allocate more buffers for frame reception now. Previously the number of receive buffers was determined by ENET_RXBD_NUM, which defaults to 5. It is determined by ENET_RXBUFF_NUM now, which is 2 ∗ ENET_RXBD_NUM by default. Increase was needed because the actual version of Kinetis ENET and ENET QOS drivers always hold ENET_RXBD_NUM number of buffers and few additional buffers are needed for passing zero-copy frame data to lwIP. If this takes too much memory in your application, you can counteract by decreasing PBUF_POOL_SIZE, since PBUF_POOL is used only for transmission when Kinetis ENET or ENET QOS is used.
        * Removed ethernetif_config_t.non_dma_memory field which was required to configure memory ranges unusable by ENET DMA on LPC devices. The setting has been replaced by BOARD_ENET_NON_DMA_MEMORY_ARRAY macro.
- 2.2.0_rev3
    - New features:

    - Ported lwIP 2.2.0.dev (2020-07-07, branch: master, SHA-1: c385f31076b27efb8ee37f00cb5568783a
      to KSDK 2.0.0.
- 2.2.0_rev2
  - New features:
    - Kinetis ENET adaptation layer - implemented zero-copy on receive.
    - lwiperf - counter of transferred bytes extended from 32 to 64 bit
  - Bug fixes:
    - Fixed restarting Auto IP from DHCP.
- 2.2.0_rev1
  - New features:
    - Ported lwIP 2.2.0.dev (2019-12-12, branch: master, SHA-1: 555812dcec38c9a2ef1ef9b31816291549
      to KSDK 2.0.0.
    - Implemented LWIP_ASSERT_CORE_LOCKED related functions in sys_arch.c. It can be enabled in lwipopts.h:
      - `#define LWIP_ASSERT_CORE_LOCKED() sys_check_core_locking()`
      - `#define LWIP_MARK_TCPIP_THREAD() sys_mark_tcpip_thread() // if NO_SYS == 0`
      - `#define LOCK_TCPIP_CORE() sys_lock_tcpip_core() // if NO_SYS == 0 and LWIP_TCPIP_CORE_LOCKING == 1`
      - `#define UNLOCK_TCPIP_CORE() sys_unlock_tcpip_core() // if NO_SYS == 0 and LWIP_TCPIP_CORE_LOCKING == 1`
- 2.1.2_rev5
  - New features:
    - Implemented TCP_USER_TIMEOUT socket option.
    - Implemented SIOCOUTQ ioctl.
- 2.1.2_rev4
  - New features:
    - Ported lwIP 2.1.3.dev (2019-02-27, branch: STABLE-2_1_x, SHA-1: 1bb6e7f52de1cd86be0eed31e3
      to KSDK 2.0.0.
    - Updated sys_thread_new implementation and comment.
    - Kinetis ENET adaptation layer - reading frames into a pbuf chain is conditionally compiled only when a single pbuf from pool cannot hold maximum frame size (PBUF_POOL_BUFSIZE >= maximum frame size). Avoiding this code also reduces stack size requirements by about 1.5 kilobytes.
  - Bug fixes:
    - Fixes in ethernetif_linkoutput() in enet_ethernetif_lpc.c:
      - Removed access to possibly freed pbuf.
      - Call pbuf_free() when transmit buffers not available.
      - When copying pbuf chain, updating the number of necessary transmit buffers to wait for, which can be often smaller in the copy.
    - When CGI script is reading POST data by chunks, the loop in httpsrv_read() may cause blocking in receive function waiting for more data at the end of the stream
      - HTTPSRV_cgi_read() - added limiting of the last chunk length according to content length to avoid undesired blocking
    - Applied AUTOIP patch `https://savannah.nongnu.org/patch/?9847` - with

modification to support multiple network interfaces.
    * Fixed buffer overflow in httpsrv when application provided CGI script does not handle the whole content of POST request
  - Removed LwipMibCompiler contrib application as it contained LGPL licensed files in 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 mbed TLS 2.28.1 branch released 2021-7-11

- 2.28.1
  - New features:
    - Ported mbedTLS 2.28.1 to SDK.
- 2.28.0
  - New features:
    - Ported mbedTLS 2.28.0 to SDK.
- 2.27.0
  - New features:
    - Ported mbedTLS 2.27.0 to SDK.
- 2.26.0
  - New features:
    - Ported mbedTLS 2.26.0 to SDK.

2.16.6_rev7

- Bug fixes:
  - Corrected definition of global variable g_isCryptoHWInitialized to be only internal static variable in sssapi_mbedtls.c file.

2.16.6_rev6

- Bug fixes:
  - Adding #ifdef in ecdsa.c to remove warning: "function "derive_mpi" was declared but never referenced", when alternative implementation of ECDSA sign and verify is used and not used Deterministic ECDSA, then was derive_mpi function never used.

Multicore SDK

The current version of Multicore SDK is 2.13.0

- 2.13.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.10.0
    - eRPC generator (erpcgen) v.1.10.0
    - Multicore Manager (MCMgr) v4.1.3
    - RPMsg-Lite v5.0.0
  - New features:

    * eRPC: MUTransport adaptation to new supported SoCs.
    * eRPC: Simplifying CI with installing dependencies using shell script, GitHub PR #267.
    * eRPC: Using event for waiting for sock connection in TCP python server, formatting python code, C specific includes, GitHub PR #269.
    * eRPC: Endianness agnostic update, GitHub PR #276.
    * eRPC: Assertion added for functions which are returning status on freeing memory, GitHub PR #277.
    * eRPC: Fixed closing arbitrator server in unit tests, GitHub PR #293.
    * eRPC: Makefile updated to reflect the correct header names, GitHub PR #295.
    * eRPC: Compare value length to used length() in reading data from message buffer, GitHub PR #297.
    * eRPC: Add TCP_NODELAY option to python, GitHub PR #298.
    * eRPC: Replace EXPECT_TRUE with EXPECT_EQ in unit tests, GitHub PR #318.
    * eRPC: Adapt rpmsg_lite based transports to changed rpmsg_lite_wait_for_link_up() API parameters.
    * eRPC, erpcgen: Better distuingish which file can and cannot by linked by C linker, GitHub PR #266.
    * eRPC, erpcgen: Stop checking if pointer is NULL before sending it to the erpc_free function, GitHub PR #275.
    * eRPC, erpcgen: Changed api to count with more interfaces, GitHub PR #304.
    * erpcgen: Check before reading from heap the buffer boundaries, GitHub PR #287.
    * erpcgen: Several fixes for tests and CI, GitHub PR #289.
    * erpcgen: Refactoring erpcgen code, GitHub PR #302.
    * erpcgen: Fixed assigning const value to enum, GitHub PR #309.
    * erpcgen: Enable runTesttest_enumErrorCode_allDirection, serialize enums as int32 instead of uint32.
    * MCMgr: mcmgr_mu_internal.c code adaptation to new supported SoCs.
    * RPMsg-Lite: Improveed debug check buffers implementation - instead of checking the pointer fits into shared memory check the presence in the VirtIO ring descriptors list.
    * RPMsg-Lite: Timeout parameter added to rpmsg_lite_wait_for_link_up API function.
    * RPMsg-Lite: VRING_SIZE is set based on number of used buffers now (as calculated in vring_init) - updated for all platforms that are not communicating to Linux rpmsg counterpart.
    * RPMsg-Lite: Fixed wrong RL_VRING_OVERHEAD macro comment in platform.h files.
    * RPMsg-Lite: Misra corrections.
- 2.12.0_imx93
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.9.1
    * eRPC generator (erpcgen) v.1.9.1
    * Multicore Manager (MCMgr) v4.1.2
    * RPMsg-Lite v4.0.1
  - New features:
    * RPMsg-Lite: Added porting layers for i.mx93 device.
- 2.12.0
  - Multicore SDK component versions:

      * embedded Remote Procedure Call (eRPC) v1.9.1
      * eRPC generator (erpcgen) v.1.9.1
      * Multicore Manager (MCMgr) v4.1.2
      * RPMsg-Lite v4.0.0
  - New features:
      * eRPC: Construct the USB CDC transport, rather than a client, GitHub PR #220.
      * eRPC: Fix premature import of package, causing failure when attempting installation of Python library in a clean environment, GitHub PR #38, #226.
      * eRPC: Improve python detection in make, GitHub PR #225.
      * eRPC: Fix several warnings with deprecated call in pytest, GitHub PR #227.
      * eRPC: Fix freeing union members when only default need be freed, GitHub PR #228.
      * eRPC: Fix making test under Linux, GitHub PR #229.
      * eRPC: Assert costumizing, GitHub PR #148.
      * eRPC: Fix corrupt clientList bug in TransportArbitrator, GitHub PR #199.
      * eRPC: Fix build issue when invoking g++ with -Wno-error=free-nonheap-object, GitHub PR #233.
      * eRPC: Fix inout cases, GitHub PR #237.
      * eRPC: Remove ERPC_PRE_POST_ACTION dependency on return type, GitHub PR #238.
      * eRPC: Adding NULL to ptr when codec function failed, fixing memcpy when fail is present during deserialization, GitHub PR #253.
      * eRPC: MessageBuffer usage improvement, GitHub PR #258.
      * eRPC: Get rid for serial and enum34 dependency (enum34 is in python3 since 3.4 (from 2014)), GitHub PR #247.
      * eRPC: Several MISRA violations addressed.
      * eRPC: Fix timeout for Freertos semaphore, GitHub PR #251.
      * eRPC: Use of rpmsg_lite_wait_for_link_up() in rpmsg_lite based transports, GitHub PR #223.
      * eRPC: Fix codec nullptr dereferencing, GitHub PR #264.
      * erpcgen: Fix two syntax errors in erpcgen Python output related to non-encapsulated unions, improved test for union, GitHub PR #206, #224.
      * erpcgen: Fix serialization of list/binary types, GitHub PR #240.
      * erpcgen: Fix empty list parsing, GitHub PR #72.
      * erpcgen: Fix templates for malloc errors, GitHub PR #110.
      * erpcgen: Get rid of encapsulated union declarations in global scale, improve enum usage in unions, GitHub PR #249, #250.
      * erpcgen: Fix compile error:UniqueIdChecker.cpp:156:104:'sort' was not declared, GitHub PR #265.
      * MCMgr: Update mcmgr_stop_core_internal() implementations to set core state to kMCMGR_ResetCoreState.
      * RPMsg-Lite: Introduce new rpmsg_lite_wait_for_link_up() API function - this allows to avoid using busy loops in rtos environments, GitHub PR #21.
      * RPMsg-Lite: Adjust rpmsg_lite_is_link_up() to return RL_TRUE/RL_FALSE.
- 2.11.1
  - Multicore SDK component versions:

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

- RPMsg-Lite: Updated RL_BUFFER_COUNT documentation.
- RPMsg-Lite: env_print macro adjusted to address MISRA 21.6 rule in MCUXpressoSDK projects.

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

config option to allow opposite side notification sending each time received buffers are consumed and put into the queue of available buffers.
    * RPMsg-Lite: Added environment layers for Threadx.

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

- New features:
    - eRPC: Improved support of const types.
    - eRPC: Fixed Mac build.
    - eRPC: Fixed serializing python list.
    - eRPC: Documentation update.
    - eRPC: Add missing doxygen comments for transports.
    - RPMsg-Lite: Added configuration macro RL_DEBUG_CHECK_BUFFERS.
    - RPMsg-Lite: Several MISRA violations fixed.
    - RPMsg-Lite: Added environment layers for QNX and Zephyr.
    - RPMsg-Lite: Allow environment context required for some environments (controlled by the RL_USE_ENVIRONMENT_CONTEXT configuration macro).
    - RPMsg-Lite: Data types consolidation.
    - MCMgr: Documentation updated to describe handshaking in a graphic form.
    - MCMgr: Minor code adjustments based on static analysis tool findings

- 2.5.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.7.1
    - eRPC generator (erpcgen) v.1.7.1
    - Multicore Manager (MCMgr) v4.0.2
    - RPMsg-Lite v2.0.2
  - New features:
    - RPMsg-Lite, MCMgr: Align porting layers to the updated MCUXpressoSDK feature files.
    - eRPC: Fixed semaphore in static message buffer factory.
    - erpcgen: Fixed MU received error flag.
    - erpcgen: Fixed tcp transport.
- 2.4.0
  - Multicore SDK component versions:
    - embedded Remote Procedure Call (eRPC) v1.7.0
    - eRPC generator (erpcgen) v.1.7.0
    - Multicore Manager (MCMgr) v4.0.1
    - RPMsg-Lite v2.0.1
  - New features:
    - eRPC: Improved code size of generated code.
    - eRPC: Generating crc value is optional.
    - eRPC: Fixed CMSIS Uart driver. Removed dependency on KSDK.
    - eRPC: List names are based on their types. Names are more deterministic.
    - eRPC: Service objects are as a default created as global static objects.
    - eRPC: Added missing doxygen comments.
    - eRPC: Forbid users use reserved words.
    - eRPC: Removed outByref for function parameters.
    - eRPC: Added support for 64bit numbers.
    - eRPC: Added support of program language specific annotations.
    - eRPC: Optimized code style of callback functions.
    - RPMsg-Lite: New API rpmsg_queue_get_current_size()
    - RPMsg-Lite: Fixed bug in interrupt handling for lpc5411x, lpc5410x

    * RPMsg-Lite: Code adjustments based on static analysis tool findings
- 2.3.1
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.6.0
    * eRPC generator (erpcgen) v.1.6.0
    * Multicore Manager (MCMgr) v4.0.0
    * RPMsg-Lite v1.2.0
  - New features:
    * eRPC: Improved code size of generated code.
    * eRPC: Improved eRPC nested calls.
    * eRPC: Improved eRPC list length variable serialization.
    * eRPC: Added @nullable support for scalar types.
    * MCMgr: Added new MCMGR_TriggerEventForce() API.
- 2.3.0
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.5.0
    * eRPC generator (erpcgen) v.1.5.0
    * Multicore Manager (MCMgr) v3.0.0
    * RPMsg-Lite v1.2.0
  - New features:
    * eRPC: Added support for unions type non-wrapped by structure.
    * eRPC: Added callbacks support.
    * eRPC: Added support @external annotation for functions.
    * eRPC: Added support @name annotation.
    * eRPC: Added Messaging Unit transport layer.
    * eRPC: Added RPMSG Lite RTOS TTY transport layer.
    * eRPC: Added version verification and IDL version verification between eRPC code and eRPC generated shim code.
    * eRPC: Added support of shared memory pointer.
    * eRPC: Added annotation to forbid generating const keyword for function parameters.
    * eRPC: Added python matrix multiply example.
    * eRPC: Added nested call support.
    * eRPC: Added struct member "byref" option support.
    * eRPC: Added support of forward declarations of structures
    * eRPC: Added Python RPMsg Multiendpoint kernel module support
    * eRPC: Added eRPC sniffer tool
    * MCMgr: Unused API removed
    * MCMgr: Added the ability for remote core monitoring and event handling
    * RPMsg-Lite: Several source files renamed to avoid conflicts with other middleware sw components
    * RPMsg-Lite: Added the ability to use Multicore Manager (MCMGR) as the IPC interrupts router
- 2.2.0
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.4.0

    * eRPC generator (erpcgen) v.1.4.0
    * Multicore Manager (MCMgr) v2.0.1
    * RPMsg-Lite v1.1.0
  - New features:
    * eRPC: win_flex_bison.zip for windows updated.
    * eRPC: Use one codec (instead of inCodec outCodec).
    * eRPC: New RPMsg-Lite Zero Copy (RPMsgZC) transport layer.
    * MCMgr: code updated to be Misra compliant.
    * RPMsg-Lite: Added macros for packed structures (compiler.h).
    * RPMsg-Lite: Improved interrupt handling in platform layer.
    * RPMsg-Lite: Changed RL_BUFFER_SIZE definition.
    * RPMsg-Lite: Fix of double initialization of vring shared data structure.
    * RPMsg-Lite: Support for the multi-instance.
- 2.1.0
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.3.0
    * eRPC generator (erpcgen) v.1.3.0
  - New features:
    * eRPC: New annotation types introduced (@length, @max_length, ...).
    * eRPC: Support for running both erpc client and erpc server on one side.
    * eRPC: New transport layers for (LP)UART, (D)SPI.
    * eRPC: Error handling support.
- 2.0.0
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.2.0
    * eRPC generator (erpcgen) v.1.2.0
    * Multicore Manager (MCMgr) v2.0.0
    * RPMsg-Lite v1.0.0
  - New features:
    * Multicore SDK support for lpcxpresso54114 board added.
    * RPMsg component of the Open-AMP framework re-implemented and the RPMsg-Lite version introduced.
    * eRPC source directory organization changed.
    * Many eRPC improvements.
- 1.1.0
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.1.0
    * Multicore Manager (MCMgr) v1.1.0
    * Open-AMP / RPMsg based on SHA1 ID 44b5f3c0a6458f3cf80 rev01
  - New features:
    * Multicore SDK 1.1.0 ported to KSDK 2.0.0.
    * Python support added into eRPC.
- 1.0.0
  - Multicore SDK component versions:
    * embedded Remote Procedure Call (eRPC) v1.0.0

    * Multicore Manager (MCMgr) v1.0.0
    * Open-AMP / RPMsg based on SHA1 ID 44b5f3c0a6458f3cf80 rev00

nghttp2 library for MCUXpresso SDK

Original package is available at https://github.com/nghttp2/nghttp2. The current version of nghttp2 library is 1.32.90.

- 1.32.90
  - Initial version of library for MCUXpresso SDK

Host USDHC driver for MCUXpresso SDK

The current driver version is 2.6.3.

- 2.6.3
  - Improvements
    * Added macro SDMMCHOST_SUPPORT_VOLTAGE_CONTROL.
- 2.6.2
  - Bug Fixes
    * Added clock force on during standard tuning to fix the card access not stable after initialization.
- 2.6.1
  - Improvements
    * Increased the delay after enable DAT3 detect card feature to fix the misdetect issue.
- 2.6.0
  - Improvements
    * Removed deprecated api in SDHC host driver.
    * Added SDMMCHOST_ConvertDataToLittleEndian api.
    * Added capability/maxBlockCount/maxBlockSize in host decriptior.
    * Improved the manual tuning flow according to specification.
    * Added mutual exclusive access for function init/deinit/reset/transfer function.
    * Fixed violations of MISRA C-2012 rule 10.1, 10.4, 16.3, 4.7.
- 2.5.3
  - Bug Fixes
    * Corrected the DAT3 detect card flow by PULL down the DAT3 pin firstly and then enable the host DAT3 function.
- 2.5.2
  - Improvements
    * Improved DAT3 card detect mechanism to avoid card false detection.
- 2.5.1
  - Improvements
    * Enabled DAT3 card detect interrupt in function SDMMCHOST_PollingCardDetectStatus to support DAT3 re-detect card.

- 2.5.0
  - Improvements
    - Added cache line size alignment maintain for the read transfer.
    - Added FSL_FEATURE_HAS_L1CACHE to enable cache maintain operation for the soc has LMEM cache.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9, 15.7, 4.7, 16.4, 10.1, 10.3, 10.4, 11.3, 14.4, 10.6, 17.7, 16.1, 16.3.
- 2.4.0
  - Improvements
    - Added cache maintain functionality in the host driver.
    - Enabled DAT3 card detect feature.
    - Increase the default STD tuning counter to 60 to cover range of the tuning window.
    - Added host instance capability macro.
    - Added clear card inserted/removed event when card removed/inserted interrupt generated.
- 2.3.0
  - Improvements
    - Merged the host controller driver from polling/freertos/interrupt to non_blocking/blocking.
    - Added SDMMC OSA layer to support muxtex access/event/delay.
- 2.2.14
  - Bug Fixes
    - Fixed uninitialized value Coverity issue.
- 2.0.0
  - Initial version

MMC Card driver for MCUXpresso SDK

The current driver version is 2.5.0.

- 2.5.0
  - Improvements
    - Added api MMC_SetSleepAwake to support enter/exit sleep state.
    - Added new api MMC_PollingCardStatusBusy for application polling card status.
    - Removed deprecated api in mmc driver and mark MMC_HostReset as deprecated.
    - Improved the read/write/erase function flow.
    - Added mutual exclusive access for init/deinit/read/write/erase function.
    - Fixed violations of MISRA C-2012 rule 4.7, 17.7, 10.7, 10.4, 13.5, 14.4, 10.6.
- 2.4.1
  - Improvements
    - Improved the voltage window argument of CMD1 according to host capabilty instead of use card ocr directly.
    - Added host HS200/HS400/8bit bus width capability validation during card initialization.
    - Used cache line size align buffer for MMC relate api.
    - Increased the CMD13 timeout count to avoid polling CMD13 time out issue.

  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9, 15.7, 4.7, 16.4, 10.1, 10.3, 10.4, 11.3, 14.4, 10.6, 17.7, 16.1, 16.3.
- 2.4.0
  - Improvements
    - Added new apis MMC_EnableCacheControl/MMC_FlushCache to support cache feature.
- 2.3.1
  - Improvements
    - Removed the dead loop while polling DAT0 and CMD13 instead of using timeout mechanism.
    - Added card state check before switching to HS400 to improve the emmc initialization stability.
    - Removed the redundant operation of memset internal buffer in MMC_WrtiteBlocks function.
  - Bug Fixes
    - Fixed the sandisk emmc always busy while sending CMD1 without supported voltage provide in argument.
- 2.3.0
  - Improvements
    - Deprecated api MMC_PowerOnCard/MMC_PowerOffCard by api MMC_SetCardPower.
    - Added internalBuffer in mmc_card_t and removed rawCid/rawCsd/rawExtendedCsd.
    - Added retuning support during data transfer under HS200 mode.
    - Increased the read/write blocks failed retry times for stability.
    - Added delay while retry the CMD1 for stability.
    - Added legacy card support, the card not support CMD6, CMD8.
- 2.2.13
  - Improvements
    - Used the boot mode value instead of boot mode mask value as the parameter of MMC_-SetBootConfig to improve user experience.
    - Removed dynamic voltage switch feature for mmc, according to JEDEC standard, the voltage should be fixed after power up.
- 2.2.12
  - Improvement
    - Increased the CMD1 retry times in the MMC card driver to improve driver compatibility.
  - Bug Fixes
    - Fixed the build warning by changing the old style function declaration static status_t inline to static inline status_t(found by adding -Wold-style-declaration in armgcc build flag).
    - Fixed the fall through build warning by adding SUPPRESS_FALL_THROUGH_WARNING() in mmc driver.
- 2.2.7
  - Bug Fixes
    - Fixed MDK 66-D warning.
- 2.2.6
  - Improvements
    - Saved MMC OCR registers while sending CMD1 with argument 0.

- Bug Fixes
    - Added MMC_PowerOn function in which there is delay function after powerup sdcard. Otherwise, the card initialization by fail.
- 2.2.5
  - Improvements
    - Added SDMMC_ENABLE_SOFTWARE_TUNING to enable/disable software tuning and it is disabled by default.
- 2.2.4
  - Bug Fixes
    - Fixed DDR mode data sequence miss issue, which is caused by NIBBLE_POS.
  - Improvements
    - Increased g_sdmmc 512byte to improve the performance when application use a non-word align data buffer address.
    - Used OCR access mode bits to determine the mmccard high capacity flag.
- 2.2.3
  - Bug Fixes
    - Added response check for send operation condition command. If not checked, the card may occasionally init fail.
- 2.2.1
  - Improvements
    - Improved MMC Boot feature.
- 2.2.0
  - Improvements
    - Optimized tuning/mmc switch voltage/mmc select power class/mmc select timing function.
    - Added strobe dll for mmc HS400 mode.
    - Added write complete wait operation for MMC_Write to fix command timeout issue.
- 2.1.2
  - Improvements
    - Improved SDMMC to support eMMC v5.0.
  - Bug Fixes
    - Fixed incorrect comparison between count and length in MMC_ReadBlocks/MMC_-WriteBlocks.
- 2.1.1
  - Bug Fixes
    - Fixed the block range boundary error when transferring data to MMC card.
- 2.1.0
  - Improvements
    - Optimized the function of setting maximum data bus width for MMC card.
- 2.0.0
  - Initial version

SD Card driver for MCUXpresso SDK

The current driver version is 2.4.1.

- 2.4.1
  - Improvements
    - Added macro SDMMCHOST_SUPPORT_VOLTAGE_CONTROL for the host which not support voltage control.
- 2.4.0
  - Improvements
    - Removed deprecated api in sd driver.
    - Added new api SD_PollingCardStatusBusy for application polling card status.
    - Improved the read/write/erase function flow.
    - Improved the signal line voltage switch flow.
    - Added powerOnDelayMS/powerOffDelayMS in sd_usr_param_t to allow redefine the default power on/off delay.
    - Added mutual exclusive access for init/deinit/read/write/erase function.
    - Fixed the driver strength configurations missed when timing mode switch to non SDR50/-SDR104 mode.
    - Fixed violations of MISRA C-2012 rule 4.7, 17.7, 10.7, 10.4, 13.5, 14.4.
- 2.3.3
  - Improvements
    - Added host SDR timing mode capability validation during card initialization.
    - Added plling card ready for data status when transfer data failed.
    - Used cache line size align buffer for SD initialization api.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9, 15.7, 4.7, 16.4, 10.1, 10.3, 10.4, 11.3, 14.4, 10.6, 17.7, 16.1, 16.3.
- 2.3.2
  - Improvements
    - Moved power off function after card detect in SD_Init for DAT3 detect card feature.
- 2.3.1
  - Improvements
    - Removed the dead loop while polling DAT0 and CMD13 instead of using timeout mechanism.
- 2.3.0
  - Improvements
    - Marked api SD_HostReset/SD_PowerOnCard/SD_PowerOffCard/SD_WaitCardDetect-Status as deprecated.
    - Added new api SD_SetCardPower/SD_PollingCardDetectStatus/SD_HostDoReset.
    - Added internalBuffer in sd_card_t and removed rawCid/rawCsd/rawScr.
    - Added retuning support during data transfer under SDR50/SDR104 mode.
    - Increased the read/write blocks failed retry times for stability.
    - Added delay while retry the ACMD41 for stability.
- 2.2.12
  - Improvements

    - Increased the sd io driver strength for SD2.0 card.
  - Bug Fixes
    - Fixed the build warning by changing the old style function declaration static status_t inline to static inline status_t(found by adding -Wold-style-declaration in armgcc build flag).
- 2.2.10
  - Bug Fixes
    - Added event value check for all the FreeRTOS events to fix program hangs when a card event occurs before create.
- 2.2.7
  - Bug Fixes
    - Fixed MDK 66-D warning.
- 2.2.5
  - Improvements
    - Added SD_ReadStatus api to get 512bit SD status.
    - Added error log support in sdcard functions.
    - Added SDMMC_ENABLE_SOFTWARE_TUNING to enable/disable software tuning and it is disabled by default.
- 2.2.4
  - Bug Fixes
    - Fixed DDR mode data sequence miss issue, which is caused by NIBBLE_POS.
  - Improvements
    - Increased g_sdmmc 512byte to improve the performance when application use a non-word align data buffer address.
    - Enabled auto cmd12 for SD read/write.
- 2.2.3
  - Bug Fixes
    - Added response check for send operation condition command. If not checked, the card may occasionally init fail.
- 2.2.1
  - Improvements
    - Kept SD_Init function for forward compatibility.
- 2.2.0
  - Improvements
    - Separated the SD/MMC/SDIO init API to xxx_CardInit/xxx_HostInit.
    - SD_Init/SDIO_Init will be deprecated in the next version.
- 2.1.6
  - Improvements
    - Enhanced SD IO default driver strength.
- 2.1.5
  - Bug Fixes
    - Fixed Coverity issue.
    - Fixed SD v1.x card write fail issue. It was caused by the block length set error.
    - Fixed card cannot detect dynamically.
- 2.1.3
  - Bug Fixes

    - Fixed Non high-speed sdcard init fail at switch to high speed.
  - Improvements
    - Added Delay for SDCard power up.
- 2.1.2
  - Improvements
    - Improved SDMMC to support SD v3.0.
- 2.1.1
  - Bug Fixes
    - Fixed the bit mask error in the SD card switch to high speed function.
  - Improvements
    - Optimized the SD card initialization function.
- 2.1.0
  - Bug Fixes
    - Changed the callback mechanism when sending a command.
    - Fixed the performance low issue when transferring data.
  - Improvements
    - Changed the name of some error codes returned by internal function.
    - Merged all host related attributes to one structure.
- 2.0.0
  - Initial version.

SDIO Card driver for MCUXpresso SDK

The current driver version is 2.4.1.

- 2.4.1
  - Improvements
    - Added macro SDMMCHOST_SUPPORT_VOLTAGE_CONTROL for the host which not support voltage control.
- 2.4.0
  - Improvements
    - Removed deprecated api in sdio driver.
    - Improved the signal line voltage switch flow.
    - Added powerOnDelayMS/powerOffDelayMS in sdio_usr_param_t to allow redefine the default power on/off delay.
    - Added mutual exclusive access for init/deinit/direct/extend function.
    - Fixed violations of MISRA C-2012 rule 4.7, 17.7, 10.1, 12.2.
- 2.3.3
  - Bug Fixes
    - Fixed logical dead code coverity issue.
  - Improvements
    - Removed deprecated api in sdio driver.
- 2.3.2
  - Improvements

    - Added host SDR timing mode capability validation during card initialization.
    - Used cache line size align buffer for SDIO initialization api.
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 11.9, 15.7, 4.7, 16.4, 10.1, 10.3, 10.4, 11.3, 14.4, 10.6, 17.7, 16.1, 16.3.
- 2.3.1
  - Improvements
    - Moved power off function after card detect in SD_Init for DAT3 detect card feature.
- 2.3.0
  - Improvements
    - Marked api SDIO_HostReset/SDIO_PowerOnCard/SDIO_PowerOffCard/SDIO_WaitCardDetectStatus as deprecated.
    - Added new api SDIO_SetCardPower/SDIO_PollingCardDetectStatus/SDIO_HostDoReset.
    - Added internalBuffer in sdio_card_t for card register content extract and improve the data access efficiency.
    - Added retry function after switch to target timing failed in SDIO_SelectBusTiming.
    - Changed defalut bus clock from 400KHZ to 25MHZ.
- 2.2.13
  - Improvements
    - Removed the sdio card interrupt from sdio host initialization, since the card interrupt enablement should be determined by application.
  - Bug Fixes
    - Fixed Out-of-bounds write Coverity issue.
- 2.2.12
  - Improvements
    - Added manual tuning function for looking for the tuning window automatically.
    - Fixed the build warning by changing the old style function declaration static status_t inline to static inline status_t(found by adding -Wold-style-declaration in armgcc build flag).
    - Fixed the fall through build warning by adding SUPPRESS_FALL_THROUGH_WARNING() in sdio driver.
- 2.2.11
  - Bug Fixes
    - Added check card async interrupt capability in function SDIO_GetCardCapability.
    - Fixed OUT OF BOUNDS access in function SDIO_IO_Transfer.
- 2.2.10
  - Bug Fixes
    - Fixed SDIO card driver get an incorrect io number when the card io number is bigger than 2.
  - Improvements
    - Added SDIO 3.0 support.
    - Added API SDIO_IO_RW_Direct for direct read/write card register access.
- 2.2.9
  - Improvements
    - Added API SDIO_SetIOIRQHandler/SDIO_HandlePendingIOInterrupt to handle multi io

pending IRQ.

- 2.2.8
  - Improvements
    - Updated sdmmc to support SDIO interrupt.
    - Added API SDIO_GetPendingInterrupt to get the pending io interrupt.
- 2.2.7
  - Bug Fixes
    - Fixed MDK 66-D warning.
- 2.2.6
  - Improvements
    - Added an unify transfer interface for SDIO.
  - Bug Fixes
    - Fixed Wrong pointer address used by SDMMCHOST_Init.
- 2.1.5
  - Improvements
    - Improved SDIO card init sequence and add retry option for SDIO_SwitchToHighSpeed function.
- 2.1.4
  - Improvements
    - Added Go_Idle function for SDIO card.
- 2.0.0
  - Initial version.

USB stack for MCUXpresso SDK

The current version of USB stack is 2.8.4.

- 2.8.4
  - Improvement:
    - Add the new netc adatper for the new netc driver.
    - Fix issues for USB device dfu and usb device msc when enable the macro USB_DEVIC-E_CONFIG_RETURN_VALUE_CHECK.
    - Change the header file including order for usb.h header.
    - Update the USB host audio class driver to fix the wrong output log.
    - Add the workaround on dev_hid_mouse_bm case for the errata TN00071.
    - Enable ROOT2 macro in USB device stack.
    - Use an unified definiton for the base address of RTxxxx platforms.
- 2.8.3
  - Improvement:
    - Update the EHCI controller driver to support the address convert for TCM.
    - Update the USB host EHCI controller driver to make sure the mutual exclusion access under multiple tasks' environment.
- 2.8.2
  - Improvement:

    * Fix noise issue of UAC 3.1, UAC 5.1, UAC 7.1 on usb audio speaker demo.
    * Fix the issue that incorrect PC behavior when ejecting USB MSC devices.
    * Update the EHCI controller driver to support RW610 that does not reply on PHY driver, especially for low power feature.
    * Update the USB_HostHelperParseAlternateSetting to fix the wrong interface parse.
    * Update dev_composite_hid_audio_unified_bm demo to suppport independent mute/unmute and volume control.
- 2.8.1
  - Improvement:
    * update USB audio demos to use audio component (components).
    * Add the checking of function call return value.
    * Add audio multiple channels demo (usb_device_composite_audio_multi_ch_unified) on RT600 audio board.
    * Fix audio noise on sync mode and improve overflow/underflow checking method.
    * Support UAC 3.1, 5.1 and 7.1 on audio speaker demo.
    * Set USB device CDC demo not to depend on DTR setting from host.
    * Support MCUX toolchain on some RTxxxx platforms.
- 2.8.0
  - Improvement:
    * Fix the USB device stack vulnerability issues.
    * Update the audio PLL and FRO adjustment codes for audio examples in RTxxx, LP-C54xxx and LPC55xxx.
    * Improve the USB PD AMS collision avoidance.
    * Improve IP3511 controller driver's dedicated ram allocation.
    * Change the USB_DATA_ALIGN_SIZE to 4 because the controller driver uses the dedicated RAM to do memcpy.
  - New features:
    * Enable USB host audio recorder demo for mutilple boards.
- 2.7.0
  - Improvement:
    * Use new feeback solution and low latency playback for usb device speaker demo and unified demos. Add underflow and overflow protection.
    * Optimize hard code for usb audio demos.
    * Update Unconstrained Power field in the Sink Capabilities Message according to the external power state.
    * Fix CVE-2021-38258 and CVE-2021-38260
  - New features:
    * Enable USB host video demo for mutilple boards.
    * Enable USB device MTP demo for mutilple boards.
    * Add PPS message to usb pd stack.
- 2.6.1
  - Improvement:
    * rename sdcard as disk for all of sdcard demos. For ramdisk demos, they are not changed.
    * add wrapper for all of disk demos to support emmc.
- 2.6.0

  - Improvement:
    - Added more ufi event to support dynamic sdcard capacity.
    - Passed MISRA-2012 mandatory and required rules.
      - Except rule 17.2 in host hub and otg stack.
      - Except rule 5.1, rule 5.4, rule 21.1 and rule 21.2.
    - Re-implemented USB components and supported NPW.
    - Improved IP3511 controller driver's cancelling transfer function.
    - Enabled the audio2.0 defaultly for device audio demos.
    - Enabled the host audio2.0 function in host audio class driver and host audio speaker demo.
  - New features:
    - enable two USB controllers in one USB host mouse demo which named as host_hid_-mouse_dual.
    - enable UAC 5.1 for usb device audio speaker demo.
- 2.5.0
  - Improvement:
    - Integrated sdk components (OSA, Timer, GPIO and serial_manager) to USB stack and demos.
    - Improved the ip3511 driver throughput.
    - Improved audio initialization codes after SDK audio drivers update.
    - Improved auido to support the audio2.0 in win10.
    - Add one "enumeration fail" callback event to host stack.
- 2.4.2
  - Improvement:
    - Put the USB controller data and transfer buffer to noncache section, removed the setting that sets the whole ocram and sdram as noncached.
    - Separated composite audio examples' channel,sample rate,format parameters from commom macro to in dedicated macro and out dedicated macro.
    - replaced USB_PrepareData with USB_AudioRecorderGetBuffer.
- 2.4.1
  - New features:
    - Added enumeration fail callback to host stack when the attached device's enumeration failed.
- 2.4.0
  - Improvement:
    - Device Charger Detection (DCD) software architecture was refactored.
  - New features:
    - Enabled Device Charger Detection (DCD) on RT1060.
    - Enabled Device Charger Detection on RT600.
    - Enabled host battery charger function on RT600.
- 2.3.0
  - New features:
    - Added host video camera support. example: usb_host_video_camera
    - Added a new device example. example: usb_device_composite_cdc_hid_audio_unified
- 2.2.0
  - New features:

        - Added device DFU support.
        - Supported OM13790DOCK on LPCXpresso54018.
        - Added multiple logical unit support in msc class driver, updated usb_device_lba_-information_struct_t to support this.
        - Supported multiple transfers for host ISO on IP3516HS.
    - Bug fixes:
        - Fixed device ip3511 prime data length than maxpacket size issue.
        - Initialized interval attribute in usb_device_endpoint_struct_t/usb_device_endpoint_init_-struct_t.
        - Removed unnecessary header file in device CDC class driver, removed unnecessary usb_-echo, and added DEBUG macro for necessary usb_echo in device CDC class driver.
        - Fixed device IP3511HS unfinished interrupt transfer missing issue.
- 2.1.0
    - New features:
        - Added host RNDIS support. example: lwip_dhcp_usb
        - Enabled USB 3.0 support on device stack.
        - Power Delivery feature: Added OM13790HOST support; Added auto policy feature; Printed e-marked cable information;
- 2.0.1
    - Bug fixes:
        - Fixed some USB issues: Fixed MSC CV test failed in MSC examples.
        - Changed audio codec interfaces.
- 2.0.0
    - New features:
        - PTN5110N support.
    - Bug fix:
        - Added some comments, fixed some minor USB issues.
- 1.9.0
    - New features:
        - Examples:
            - usb_pd_alt_mode_dp_host
- 1.8.2
    - Updated license.
- 1.8.1
    - Bug fix:
        - Verified some hardware issues, support aruba_flashless.
- 1.8.0
    - New features:
        - Examples:
            - usb_device_composite_cdc_vcom_cdc_vcom
            - usb_device_composite_hid_audio_unified
            - usb_pd_sink_battery
            - Changed usb_pd_battery to usb_pd_charger_battery.
    - Bug fix:
        - Code clean up, removed some irrelevant code.

- 1.7.0
  - New features:
    - USB PD stack support.
  - Examples:
    - usb_pd
    - usb_pd_battery
    - usb_pd_source_charger
- 1.6.3
  - Bug fix: -IP3511_HS driver control transfer sequence issue, enabled 3511 ip cv test.
- 1.6.2
  - New features:
    - Multi instance support.
- 1.6.1
  - New features:
  - Changed the struct variable address method for device_video_virtual_camera and host_phdc-_manager.
- 1.6.0
  - New features:
    - Supported Device Charger Detect feature on usb_device_hid_mouse.
- 1.5.0
  - New features:
    - Supported controllers
      - OHCI (Full Speed, Host mode)
      - IP3516 (High Speed, Host mode)
      - IP3511 (High Speed, Device mode)
    - Examples:
      - usb_lpm_device_hid_mouse
      - usb_lpm_device_hid_mouse_lite
      - usb_lpm_host_hid_mouse
- 1.4.0
  - New features:
    - Examples:
      - usb_device_hid_mouse/freertos_static
      - usb_suspend_resume_device_hid_mouse_lite
- 1.3.0
  - New features:
    - Supported roles
      - OTG
    - Supported classes
      - CDC RNDIS
    - Examples
      - usb_otg_hid_mouse
      - usb_device_cdc_vnic
      - usb_suspend_resume_device_hid_mouse
      - usb_suspend_resume_host_hid_mouse

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

VGLite GPU Driver

The current version of the VGLite GPU Driver is 3.0.15_rev7.

- version 3.0.15_rev7
  - Fixed:
    - (MCUX-54842) Fixed build warnings
- version 3.0.15_rev6
  - Fixed:
    - Fixed incorrect scissoring issue in single thread mode
    - Optimized line stroking to reduce memory consumption
    - Extended blit output quality workaround to "vg_lite_blit_rect"
    - (IMX-3008) Fixed driver reporting incorrect version number
    - (IMX-2848) Allocated path stroking parameters dynamically
    - (IMX-3010) Fixed scissoring window check with large tessellation buffers
  - Changed:
    - (IMX-2907) Removed obsolete "vg_lite_perspective" API
- version 3.0.15_rev5
  - Fixed:
    - (IMX-2867) Fixed hang when processing vector paths with zero length
    - (IMX-2959) Fixed GPU using garbage data during image filtering
    - (IMX-2900) Restructured source code for better single thread & multithread modes maintenance
  - Changed:
    - (MCUX-52922) Disable GPU auto clock gating by default. Feature can be enabled from build config
- version 3.0.15_rev4

  - Changed:
    - (IMX-2900) Renamed build switch for disabling driver multithread support
- version 3.0.15_rev3
  - Fixed:
    - Relocated centerX/Y definitions in vg_lite.c
    - (IMX-2918) Reduced vg_lite_finish() delay when it has nothing to do
    - (IMX-2901) Fixed reversed red and blue channels in colour gradients fill colour
    - (IMX-2901) Fixed linear gradient matrix transformation error
    - (IMX-2901) Fixed radial gradient render error
  - Changed:
    - (IMX-2799) Enabled GPU auto clock gating by default
    - (IMX-2799) Added build switch to disable GPU auto clock gating
  - Added:
    - (IMX-2900) Added initial support for single thread mode
- version 3.0.15_rev2
  - Fixed:
    - (IMX-2881) Fixed memory leaks in vector path stroking implementation
    - (IMX-2863) Fixed stroked polygons rendering issue
    - (IMX-2842) Fixed system hang when drawing circular arcs
    - (MGG-897) Use OS heap instead of application heap for stroked vector polygons
    - (MGG-897) Use OS heap instead of application heap for ciecular arc rendering
  - Changed:
    - (IMX-2863) Allow users to configure fill colour for stroked & filled vector paths
- version 3.0.15_rev1
  - Fixed:
    - (IMX-2844) Fixed missing path descriptor initialization in "vg_lite_init_arc_path"
    - (IMX-2837) Fixed arc drawing direction
    - (IMX-2811) Added VGPE flush after buffer clear
  - Changed:
    - (IMX-2835) Optimized storage of radial gradients params to allow memory saving
  - Added:
    - Added dithering support for RT11xx platforms
    - Added color keying support for RT11xx platforms
    - (IMX-2817) Added vector path stroking
    - (IMX-2692) Added support for HW accelerated linear gradients on RT11xx platforms
- version 3.0.13_rev2
  - Fixed:
    - (MGG-793) Fixed clipping issue when using the RT500 blit output quality workaround
    - (MGG-830) Disabled RT500 blit output quality workaround for non-affine graphic transformations
    - (IMX-2701) Fixed memory leak in vector arc drawing API
    - (IMX-2699) Fixed build warnings in vector arc drawing API
    - (MGG-836) Fixed the font/text support via main VGLite driver API
  - Changed:
    - (IMX-1724) Changed image width 16 pixels alignment to stride 16 byte alignment

    * (MCUX-46210) Dropped useless "const" qualifier for the "name" attribute of "vg_lite_-font_params_t" data structure
    * (MGG-836) Reordered "vg_lite_draw_text" API arguments
- version 3.0.13_rev1
  - Fixed:
    * (IMX-2577) Fixed support for colour palettes (CLUT) in multithread mode
    * (MGG-735) Fixed Elementary library instability caused by using calloc/free in ElmWrap-Buffer
  - Changed:
    * (IMX-2600) Updated "vg_lite_finish" to wait for all frames previously submitted with "vg_lite_flush"
    * Aligned "vg_lite_radial_gradient_parameter" data struct with parameters in Elementary EVO object
  - Added:
    * Added support for drawing vector arcs/circles
    * Added support for i.MXRT6Q GPU
    * Added support for GCNanoliteV GPU Rev. 0x1322
    * Added vector arcs support in Elementary library
- version 3.0.11_rev3
  - Fixed:
    * Fix async event reset after being initialized
    * (IMX-2604) Fix polygon's rendering regression in multitasking scenarios
    * Avoid "vg_lite_blit" modifying user's transformation matrix
- version 3.0.11_rev2
  - Fixed:
    * (MGG-685) Added workaround to improve "blit" output quality for RT500
    * (MCUX-43004) Fixed clipping window regression issue introduced by VGLite 3.0.11.1
    * (MGG-764) Fixed VGLite heap useless splitting of memory nodes
    * (MGG-765) Fixed regression issue introduced by VGLite 3.0.11.1 when loading graphic resources using Elementary library
    * (IMX-2506) Fixed "vg_lite_update_rad_grad" not checking the result of memory allocation
    * (MCUX-42992) Fixed IAR toolchain not recognizing optimization directive
    * (MGG-763) Remove risk of out-of-bounds read in "vg_lite_update_rad_grad" function
  - Changed:
    * (IMX-2527) Improved memory footprint by using a common tessellation buffer for all drawing tasks
    * (MGG-712) Restructured OS abstraction layer to allow easier integration with popular OSes
- version 3.0.11_rev1
  - Fixed:
    * (IMX-2502) Fixed GPU command buffer overflow when copying context data
    * (IMX-2503) Fixed additional colour ring incorrectly appearing at the edge of radial gradients
    * (IMX-2487) Fixed risk of memory leak in "vg_lite_upload_path"

    - (IMX-2429) Fixed incorrect blending of A4 and A8 images (regression since VGLite 3.-0.4.x)
    - (MGG-687) Fixed build warning when VG_RENDER_TEXT feature is disabled
  - Changed:
    - (IMX-2354) Added support for dynamic command buffer size management
  - Added:
    - (IMX-2435) Added new API function - vg_lite_get_transform_matrix - to calculate parameters for 2D perspective transformations
    - (IMX-2411) Added support for radial gradients in Elementary library
    - (IMX-2026) Added support for images embedded in EVO data in Elementary library
    - (IMX-2026) Added support for patterns embedded in EVO data in Elementary library
- version 3.0.9_rev2
  - Fixed:
    - (MCUX-40557) Fixed build warnings
- version 3.0.9_rev1
  - Fixed:
    - (MGG-648) Fixed rendered text overlapping issue
    - (MGG-650) Fixed memory leak caused by failure to unload RLE font data
    - (IMX-2395) Fixed incorrect reporting of indexed images as "supported" for GC355 GPU (RT1170)
  - Changed:
    - (IMX-2370) Refactored GPU driver HAL and OS layers
    - (MGG-646) Configured a vector font as default font
- version 3.0.9
  - Fixed:
    - (IMX-2361) Fixed tessellation bounds computation error
  - Changed:
    - (IMX-2367) Enabled alpha channel premultiplication by default for GC355 GPU (R-T1170)
    - (IMX-2261) Added Elementary library input data address alignment verification
  - Added:
    - (IMX-2323) Added support for radial colour gradients for GC355 GPU (RT1170)
    - (IMX-2317) Upgraded the Elementary library to be thread safe
- version 3.0.6_rev4
  - Fixed:
    - (IMX-2357) Fixed rendering performance degradation since the implementation of the multithread/multicontext support
    - (MGG-576) Elementary: Fixed hard fault when resetting translation of EVO object
    - (MCUX-38672) Fixed font and text support build warnings
    - (MGG-596) Fixed memory leak in raster font loading
    - (MGG-596) Font and text support: Fixed out of range memory access in Elementary library
  - Changed:
    - (MGG-596) "VG_RENDER_TEXT=1" build symbol now required to enable font and text support

    * (MGG-594) Updated font and text support to allow easy decoupling from GPU driver and Elementary when not needed
    * (MGG-533) Removed "is_tspan" attribute from "vg_lite_font_attributes_t"
    * (MGG-533) Added new attribute "tspan_has_dx_dy" to "vg_lite_font_attributes_t"
    * (MGG-533) Added new argument "matrix" to "vg_lite_draw_text" API function
    * (MGG-592) Renamed "eFontTypes_t" enum to "eFontType_t"
    * (MGG-592) Renamed "eFontVectorType" identifier to "eFontTypeVector"
    * (MGG-592) Renamed "eFontRasterType" identifier to "eFontTypeRaster"
    * (MGG-596) Changed "vg_lite_draw_text" function return value from "int" to "vg_lite_-error_t"
  - Added:
    * (MGG-596) Added "vg_lite_find_font" API function
    * (MGG-596) Added 2 new error codes for "vg_lite_error_t": VG_LITE_ALREADY_EXISTS and VG_LITE_NOT_ALIGNED
    * (IMX-2357) Allow users to override command queue task priority at build time using QUEUE_TASK_PRIO build symbol
    * (MGG-551) Added text wrapping support for vector fonts
    * (MGG-533) Added support for text right alignment
- version 3.0.6_rev3
  - Added:
    * (MGG-551) Added support for font and text rendering
- version 3.0.6_rev2
  - Fixed:
    * (IMX-2292) Fixed command buffer flushing after draw
    * (IMX-2293) Fixed copy of register status when command buffer was not full
    * (IMX-2305) Fixed scissor window taking no effect
    * (IMX-2324) Fixed GPU feature table reset when calling "vg_lite_close"
    * (IMX-2358) Fixed misuse of address operator in checking colour channel premultiplication flag
    * (MGG-542) Cleaned up useless "memset" in "vg_lite_init"
- version 3.0.6_rev1
  - Fixed:
    * (IMX-2295) Initialize task context to zero in vg_lite_init()
- version 3.0.6
  - Fixed:
    * (MGG-525) Fixed "vg_lite_init_path" not properly initializating the "path" data structure
  - Changed:
    * (IMX-2255) Updated "vg_lite_set_scissor" arguments to (x, y, width, height) instead of (x0, y0, x1, y1)
  - Added:
    * (IMX-2104) Added API to enable/disable colour channel pre-multiplication at runtime on RT1170
- version 3.0.5
  - Fixed:
    * (IMX-2252) Reset global mutex when it is destroyed

        * (IMX-2252) Fixed reset of task local context in vg_lite_close()
    - Changed:
        * (MGG-333) Enabled scissoring for GC255 GPU (i.MXRT500)
    - Added:
        * (IMX-1729) Added support for drawing from multiple threads
- version 3.0.4_rev5
    - Changed:
        * (IMX-2104) Disabled by default colour channel pre-multiplication on RT1170 platform
        * (MGG-517) Updated "vg_lite_draw_pattern" function to return VG_LITE_NOT_SUPPORT for A4/A8 patterns
    - Fixed:
        * (IMX-2155) Fixed hard coded image mode in "vg_lite_draw_pattern"
        * (IMX-2153) Updated "vg_lite_draw_pattern" to take into account pattern transparency
        * (KPSDK-37093) Elementary library - Fixed bad free in "load_evo"
        * (KPSDK-37093) Elementary library - Avoid resource leak in "ElmCreateBuffer"
- version 3.0.4_rev4
    - Fixed:
        * Fixed empty function argument lists definition for scissoring related API functions
        * (IMX-1995) Extended RT500 image rotation fix to vg_lite_blit_rect, vg_lite_draw_pattern
        * (IMX-1995) Isolated RT500 image rotation fix effects to RT500 platform only
- version 3.0.4_rev3
    - Fixed:
        * (IMX-1995) Compensated for RT500 image shift effect when rotation is approaching multiples of 90 dgs
- version 3.0.4_rev2
    - Fixed:
        * Fixed integration issue of "vg_lite_mem_avail" API
- version 3.0.4_rev1
    - Changed:
        * (IMX-1768) Enabled users to query, at runtime, the support for VG_LITE_UPPER draw quality
    - Fixed:
        * (IMX-2074) Fixed GPU exception handling issue
    - Added:
        * (IMX-2045) Added API to provide available heap memory
- version 3.0.4
    - Changed:
        * (IMX-1957) Enabled users to query, at runtime, the support for BORDER_CULLING and SCISSOR features
        * Enable users to query, at runtime, the support for RGBA 2 bits-per-channel image formats
    - Fixed:
        * (IMX-1934) Fixed image stride alignment verification for TILED images
        * Fix GC355 GPU (i.MXRT1170) draw error when tessellation window width is not aligned to 128

- Added:
    - (MGG-333) Added support for GC355 GPU (i.MXRT1170) scissoring
- version 3.0.1_rev1
  - Fixed:
    - (MGG-250) Fixed GPU hang after a random time (mostly reproduced on RT1170 platforms)
    - (KPSDK-33132) Fixed Elementary library memory leaks in case of failed EBO loading
    - (MGG-336) Allow use of blend modes not affected by the border culling limitation
    - (MGG-18) Fixed Elementary library memory leaks when loading EVO/EBO/EGO objects
    - (MGG-353) Fixed linear colour gradient rendering error when loading EVOs using the Elementary library
- version 3.0.1
  - Changed:
    - Removed "vg_lite_blit2" API function due to lack of hardware support
    - Removed "vg_lite_scanline" API function due to lack of harware support
    - Aggregated "vg_lite_error.h" API header file content into "vg_lite.h"
    - Aggregated "vg_lite_features.h" API header file content into "vg_lite.h"
    - Aggregated "vg_lite_matrix.h" API header file content into "vg_lite.h"
    - Aggregated "vg_lite_path.h" API header file content into "vg_lite.h"
    - Aggregated "vg_lite_util.h" API header file content into "vg_lite.h"
    - (IMX-1861) Added return code to the "vg_lite_flush" API function
    - Changed VGLite GPU driver license from proprietary to MIT
  - Fixed:
    - Fixed definition of "elm_alloc" function in Elementary toolkit
    - (IMX-1869) Fixed initialization of aligned bytes in the command buffer
    - (IMX-1821) Fixed inverted background colours when using "vg_lite_draw_pattern"
    - Fixed hang when calling "vg_lite_flush" repeatedly
    - (IMX-1861) Fix propagation of return codes from "stall", "submit", "vg_lite_flush" function calls
- version 2.0.14_rev1
  - Changed:
    - (IMX-1809) Fixed misspelling of "vg_lite_buffer_transparency_mode"
    - (IMX-1778) Added verification of colour gradients parameters
    - (IMX-1813) Added return code to the "vg_lite_hal_allocate_contiguous" function
    - (MGG-204) Added return code to "vg_lite_finish"
  - Fixed:
    - (IMX-1808) Fixed "vg_lite_blit" failure on dynamically allocated buffers
    - (IMX-1773) Fixed failure to create 16 colours gradients
    - (IMX-1790) Fixed driver incorrectly reporting available heap space
    - (IMX-1810) Fixed verification of raster image stride alignment
    - (IMX-1810) Fixed verification of raster image colour depth
    - (IMX-1816) Fixed "vg_lite_close" not releasing memory allocated from OS heap
    - (MGG-201) Fixed hard fault caused by command buffer management
    - (MGG-202) Fixed "vg_lite_hal_wait_interrupt" function ignoring the timeout
    - (MGG-203) Fixed "vg_lite_draw" function always returning success

- version 2.0.13_rev2
  - Fixed:
    - (MGG-102) Fixed incorrect colour gradient clipping issue when using "vg_lite_draw_-gradient" API
    - (MGG-140) Fixed "vg_lite_draw_gradient" error when gradient is not covering the entire shape
- version 2.0.13_rev1
  - Added:
    - (MGG-88) Support for operating with BGRA2222, ABGR2222, ARGB2222 type images
    - (MGG-88) Support for operating with ABGR4444, ARGB4444 type images
    - (MGG-88) Support for operating with ABGR8888, ARGB8888 type images
    - (MGG-88) Support for operating with XBGR8888, XRGB8888 type images
    - (MGG-52) Improved GPU bus error reporting by using weak functions
  - Changed:
    - (MGG-66) Restructured GPU driver by exposing the HAL source code for easier integration with operating systems
  - Fixed:
    - (MGG-72) Fixed rough edges of vector artefacts when using the "vg_lite_draw_pattern" API
    - (MGG-58) Fixed "vg_lite_blit_rect" API not supporting a zero Y coordinate
- version 2.0.13_rev0

3 Component Change Log

CODEC

The current codec common driver version is 2.3.1.

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

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

WM8904

The current wm8904 driver version is 2.5.1.

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

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

New features

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

Bug Fixes

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

2.0.3

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

2.0.2

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

2.0.1

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

2.0.0

- Initial version.

CS42888

The current cs42888 driver version is 2.1.3

- 2.1.3

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

TFA9896

The current TFA9896 driver version is 6.0.2.

- 6.0.2
  - Bug Fixes
    - Fixed violations of MISRA C-2012 rule 10.4, 16.1, 16.3.
- 6.0.1
  - Bug Fixes
    - Fixed the coverity issue of error code overwritten.
- 6.0.0
  - Initial version.

SERIAL_MANAGER

The current Serial_Manager component version is 1.0.2.

- 1.0.2
  - Add SerialManager_WriteTimeDelay()/SerialManager_ReadTimeDelay() for serial manager's

read/write non-blocking mode.

- 1.0.1
  - Add prefixing fsl_component_xxx/fsl_adapter_xxx.
- 1.0.0
  - Initial version

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

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

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

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

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

© 2021 NXP B.V.