Document identifier: MPC5645S_0N29D
Rev. 30, 7/2023

MPC5645S_0N29D

Mask Set Errata

Mask Set Errata for Mask 0N29D

Revision History

This report applies to mask 0N29D for these products:

- MPC5645S

Table 1. Revision History

Revision	Date	Significant Changes
30 JUL 2023	7/2023	The following errata were removed. • ERR011235 • ERR050575
30 SEP 2022	9/2022	The following errata were revised. • ERR011235 • ERR007394
28 FEB 2022	1/2022	The following errata were revised. • ERR050575
30 JULY 2021	8/2021	The following errata were added. • ERR007589 • ERR009682 • ERR003010 • ERR003223 • ERR008970 • ERR050782 • ERR011235 • ERR009764 • ERR009976 • ERR009978 • ERR010755 • ERR050575 • ERR011295 • ERR011294 • ERR007688 • ERR011293 • ERR004186 • ERR008933

Table continues on the next page...

Table 1. Revision History (continued)

Revision	Date	Significant Changes
		• ERR008951 The following errata were revised. • ERR003853 • ERR050459
20 APR 2020	4/2020	The following errata were added. • ERR050459 • ERR011214
20 FEB 2015	4/2020	Initial Revision

Errata and Information Summary

Table 2. Errata and Information Summary

Erratum ID	Erratum Title
ERR003010	ADC: conversion chain failing after ABORT chain
ERR003223	PDI does not follow ITU656 spec for Internal Sync Mode
ERR003526	Flash : Prefetch at flash boundary causes unexpected ECC errors
ERR003574	MC_RGM: A non-monotonic ramp on the VDD_HV/BV supply can cause the RGM module to clear all flags in the DES register
ERR003624	RLE: Decode operation freezes when configured for very small data transfers
ERR003659	FLASH: Resuming after a suspend during an Erase may prevent the erase from completing.
ERR003664	Flash may not enter normal mode after reset or on wake from STANDBY
ERR003853	FLASH: Flash array access may be incorrect after sleep mode exit or after power up
ERR004114	MPC5645S RM does not correctly describe slew rate control bits
ERR004168	ADC: Abort switch aborts the ongoing injected channel as well as the upcoming normal channel
ERR004186	ADC: Do not trigger ABORT or ABORTCHAIN prior to the start of CTU triggered ADC conversions and do not trigger ABORTCHAIN prior to the start of INJECTED triggered ADC conversions.
ERR004340	LINFlexD: Buffer overrun can not be detected in UART Rx FIFO mode
ERR005099	MC_ME: possibility of the machine check on low power mode exit (MPC5645S only)
ERR006026	DSPI: Incorrect SPI Frame Generated in Combined Serial Interface Configuration
ERR006082	LINFlexD : LINS bits in LIN Status Register(LINSR) are not usable in UART mode.
ERR006481	NZ4C3/NZ7C3: Erroneous Resource Full Message under certain conditions
ERR006726	NPC: MCKO clock may be gated one clock period early when MCKO frequency is programmed as SYS_CLK/8.and gating is enabled

Table continues on the next page...

Table 2. Errata and Information Summary (continued)

Erratum ID	Erratum Title
ERR006976	MC_ME: SAFE mode not entered immediately on hardware-triggered SAFE mode request during STOP0 mode
ERR007026	FLASH: Continuously cycling from STANDBY to RUN mode may disturb flash array content.
ERR007120	NZxC3: DQTAG implemented as variable length field in DQM message
ERR007274	LINFlexD: Consecutive headers received by LIN Slave triggers the LIN FSM to an unexpected state
ERR007394	MC_ME: Incorrect mode may be entered on low-power mode exit.
ERR007589	LINFlexD: Spurious timeout error when switching from UART to LIN mode or when resetting LINTCSR[MODE] bit in LIN mode
ERR007688	RTC: An API interrupt may be triggered prematurely after programming the API timeout value
ERR007953	ME: All peripherals that will be disabled in the target mode must have their interrupt flags cleared prior to target mode entry
ERR008933	LINFlexD: Inconsistent sync field may cause an incorrect baud rate and the Sync Field Error Flag may not be set
ERR008951	I2C: Attempting a start cycle while the bus is busy may generate a short clock pulse
ERR008970	LINFlexD: Spurious bit error in extended frame mode may cause an incorrect Idle State
ERR009066	PAD RING: The Device may lock-up into reset loop while exiting from standby mode or at power up
ERR009682	SPI: Inconsistent loading of shift register data into the receive FIFO following an overflow event
ERR009764	SARADC : DMA interface limitation depending on PBRIDGE/SARADC clock ratio
ERR009976	DSPI: Incorrect data received by master with Modified transfer format enabled when using Continuous serial communication clock mode
ERR009978	eMIOS: Unexpected channel flag assertion during GPIO to MCB mode transition
ERR010755	DSPI: Transmit and Receive FIFO fill flags in status register is not cleared when DMA is improperly configured
ERR011214	[MDDRC]: Do not perform unaligned address transfers to MDDRC from different bus masters (e.g: PowerPC core)
ERR011293	EMIOS: For any UC operating in OPWFMB mode the Channel Count register should not be written with a value greater than Channel B Data register value
ERR011294	EMIOS: OPWFMB and MCB mode counter rollover resets the counter to 0x0 instead of 0x1 as mentioned in the specification
ERR011295	EMIOS: In OPWFMB mode, A1/B1 registers do not get reloaded with A2/B2 register values if counter value returns 0x1 after counter wrap condition
ERR050459	SXOSC: Clock output may contain extra clock pulses in Normal mode
ERR050782	e200: Time Base TBU register contains wrong value during TBL rollover

Known Errata

ERR003010: ADC: conversion chain failing after ABORT chain

Description

During a chain conversion while the ADC is in scan mode when ADC_MCR[ABORTCHAIN] is asserted the current chain will be aborted as expected. However, in the next scan the first conversion of the chain is performed twice and the last conversion of the chain is not performed.

Workaround

When aborting a chain conversion enable ADC_MCR[ABORTCHAIN] and disable ADC_MCR[START].

ADC_MCR[START] can be enabled when the abort is complete.

ERR003223: PDI does not follow ITU656 spec for Internal Sync Mode

Description

Internal synchronisation is a means of reducing the pincount required by the interface between a video source and the PDI module. The timing signals (HSYNC, VSYNC, CLK and DataEnable) are encoded within the data signal, meaning they do not require their own dedicated pins.

When using the PDI in internal sync mode the PDI will incorrectly reject streams which conform to the ITU656 standard. Further information can be found in Engineering Bulletin EB736.

Workaround

Use external sync mode to provide the HSYNC, VSYNC, CLK (and optionally, DataEnable) signals or use workaround provided in EB736.

ERR003526: Flash : Prefetch at flash boundary causes unexpected ECC errors

Description

For flash modules, where the implemented array size is less than the maximum allowed size of the flash revision (2 MB for FL2 or 3 MB for FL3) then if prefetch from the flash is enabled via the configuration bits in the Platform Flash Configuration Register (PFCR), accesses to the last page in the flash may cause an unexpected ECC error flagged by the ECC Event Error bit (EER) in the Module Configuration Register (MCR) of the flash module.

As 1 page is 4 words + ECC bits (146 bits in total), the addresses affected are the upper most 16 bytes in the flash address space.

Workaround

Either :

(1) Don't use prefetch

or

(2) Do not use last page in flash array when prefetch is enabled;

or

(3) Ignore the setting of the Flash module MCR.EER register bit on the last page if you are using prefetch.

ERR003574: MC_RGM: A non-monotonic ramp on the VDD_HV/BV supply can cause the RGM module to clear all flags in the DES register

Description

During power up, if there is non-monotonicity in power supply ramp with a voltage drop > 100 mV due to external factors, such as battery cranking or weak board regulators, the SoC may show a no flag condition (F_POR == LVD12 == LVD27 == 0).

Under these situations, it is recommended that customers use a workaround to detect a POR.

In all cases, initialization of the device will complete normally.

Workaround

The software workaround need only be applied when neither the F_POR, LVD27 nor LVD12 flag is set and involves checking SRAM contents and monitoring for ECC errors during this process. In all cases, an ECC error is assumed to signify a power-on reset (POR).

Three suggestions are made for software workarounds. In each case, if POR is detected all RAM should be initialized otherwise no power-on condition is detected and it is possible to initialize only needed parts of RAM while preserving required information.

Software workaround #1 :

An area of RAM can be reserved by the compiler into which a KEY, such as 0x3EC1_9678, is written. This area can be checked at boot and if the KEY is incorrect or an ECC error occurs, POR can be assumed and the KEY should be set. Use of a KEY increases detection rate to 31 bits(=<10e-9) or 23 bits (= < 5.10e-6) instead of 7 bit linked to ECC (=< 10e-2)

Software workaround #2 :

When runtime data should be retained and RAM only fully re-initialized in the case of POR, a checksum should be calculated on the runtime data area after each data write. In the event of a reset where no flags are set, the checksum should be read and compared with one calculated across the data area. If reading the checksum and the runtime data area succeeds without an ECC error, and the checksums match, it is assumed than no POR occurred. The checksum could be a CRC, a CMAC or any other suitable hash.

Software workaround #3 :

Perform a read of memory space that is expected to be retained across an LVD reset. If there are no ECC errors, it can be assumed that an LVD reset occurred rather than a POR.

ERR003624: RLE: Decode operation freezes when configured for very small data transfers

Description

When the RLE decode module is configured to decode small blocks of encoded data it may enter a state where an operation either never completes or completes the current operation but causes the next decode operation to fail. The fail condition depends on the configuration of the RLE module and the eDMA channel used to copy data into the module.

Workaround

Use at least 32 byte images (compressed and non-compressed) when using DMA modes on RLE.

ERR003659: FLASH: Resuming after a suspend during an Erase may prevent the erase from completing.

Description

If an erase suspend (including the flash put into sleep or disabled mode) is done on any block in the low Address Space (LAS) or the Mid-Address Space (MAS) except the 16 KB blocks, or if a suspend is done with multiple non-adjacent blocks (including the High Address Space [HAS]), the flash state machine may not set the FLASH_MCR[DONE] bit in the flash Module Control Register.

This condition only occurs if the suspend occurs during certain internal flash erase operations. The likelihood of an issue occurring is reduced by limiting the frequency of suspending the erase operation.

Workaround

If the suspend feature (including disable and sleep modes) of the flash is used, then software should ensure that if the maximum time allowed for an erase operation occurs without a valid completion flag from the flash (FLASH_MCR[DONE] = 1), the software should abort the erase operation (by first clearing the Enable High Voltage (FLASH_MCR[EHV]) bit, then clearing the Erase read/Write bit (FLASH_MCR[ERS] bit) and the erase operation should be restarted.

Note: The cycle count of the sector is increased by this abort and restart operation.

ERR003664: Flash may not enter normal mode after reset or on wake from STANDBY

Description

One of the actions performed during a microcontroller reset is to reset the internal flash block. This takes place on all microcontroller resets except those where the short sequence reset option is selected. The short reset is controlled by the RGM_FESS register and applies to a limited number of functional resets.

The internal flash is also reset when the microcontroller wakes from STANDBY mode.

The expected behavior is that the flash block will return to normal operation after receiving an internal reset, however, this intermittently fails to occur.

The behavior of the microcontroller during this condition depends on the circumstances of the internal reset.

If the condition occurs on a microcontroller reset then the Reset Generation Module will stall the reset process while waiting for the internal flash to boot. As part of its normal operation the RGM will enable the Software Watchdog Timer and so a second watchdog reset will occur approximately 16 ms after the failed reset. It is likely that this second reset will behave normally.

There are two possible scenarios when waking from STANDBY mode.

If the device is configured to restart from flash on waking from STANDBY then the device will intermittently stall while waiting for the flash and will be unable to execute code. This stall can only be recovered by a microcontroller reset.

If the device is configured to restart from RAM on waking from STANDBY then the behavior depends on how the flash is configured in the ME_DRUN_MC register. If the flash is configured to be in normal mode then the device will intermittently stall while waiting for the flash and will be unable to execute code. This stall can only be recovered by a reset. If the flash is configured to be in power-down or low-power mode the device will execute code from RAM as expected however any attempt to change mode to place the flash into normal mode will intermittently fail and that mode entry transition will never complete.

Workaround

The original source of a reset can be determined by examining the bits in the RGM_DES and RGM_FES register. If no other flags are set then the SWT is the original reset source. If other flags are set then it is possible that they are the true source of reset. To simplify this detection ensure that the source flags in the RGM_DES and RGM_FES registers are cleared once the reset source is confirmed. This will ensure that the flag registers only contain flags related to unhandled resets.

If suitable for the application then configure functional resets to use the short sequence.

There are four workarounds possible for use in STANDBY mode:

(1) Do not use STANDBY mode. Instead use STOP mode and configure it to the lowest current configuration.

(2) Configure the device to recover to flash and configure the SWT watchdog to be active from reset by setting the flash NVUSR0[WATCHDOG_EN] bit. In software track entry into and out of STANDBY mode using a value in the RAM. If the SWT watchdog flag is present in the RGM_FES register then the value in RAM will indicate that the reset was caused by recovery from STANDBY as distinct from a normal SWT watchdog failure while executing code.

(3) Configure the device to recover into RAM and then perform a mode transition to place the flash into normal mode in a RUNn mode (for example RUN0). If the mode transition does not complete after a suitable period (for example 100 µs) then abort the

transition by performing a mode transition back to DRUN mode and in software set a value in RAM before executing a mode transition into RESET mode. This will cause the device to reset and the flash to recover. The software can distinguish between this reset and another by examining the flags in the RGM_FES register and the value in RAM.

(4) Configure the device to recover into RAM and then perform a mode transition to place the flash into normal mode in a RUNn mode (for example RUN0). If the mode transition does not complete after a suitable period (for example 100 µs) then abort the transition by performing a mode transition back to DRUN mode. The flash can be recovered by cycling through STANDBY mode again. Configure a wakeup using the API set to a suitable period (for example 500 µs) and then perform another mode transition into STANDBY mode. When the device recovers perform the flash mode transition once more. If this fails on multiple attempts perform a reset per step (3). It is unlikely that this will fail more than once in a row.

ERR003853: FLASH: Flash array access may be incorrect after sleep mode exit or after power up

Description

After exiting sleep mode or power up, a flash read may return incorrect data or a program/erase operation may fail. The condition will persist from sleep mode exit or power up until the next reset assertion.

Workaround

Flash read: Software Watchdog Timer (SWT) needs to be enabled at the time the micro exits sleep mode and exits its reset sequence after power up which provides a recovery mechanism if code execution fails.

Flash program/erase: After exiting sleep mode and power up, the device must be reset before performing flash program/erase operations. However, through proper usage of software mechanisms such as EE emulation algorithms in addition to Error Correction Code (ECC), the probability of observing a failure due to ineffective programing or erasing will be greatly reduced.

ERR004114: MPC5645S RM does not correctly describe slew rate control bits

Description

The fast pads in MPC5645S do not have the slew rate control bits in the same physical location as the medium and slow pads. Instead of the SRC[1:0] control being in bits PCR[12:13] they are in PCR[8:9] instead. PCR[12:13] are writable but have no effect on the behaviour of the pad. The following fast pads exist on MPC5645S: GPIO[127], MCKO, GPIO[160], GPIO[97], GPIO[119] & GPIO[85].

Workaround

Use bits PCR[8:9] for configuring slew rate for fast pads.

ERR004168: ADC: Abort switch aborts the ongoing injected channel as well as the upcoming normal channel

Description

If an Injected chain (jch1,jch2,jch3) is injected over a Normal chain (nch1,nch2,nch3,nch4) the Abort switch does not behave as expected.

Expected behavior:

Correct Case (without SW Abort on jch3): Nch1- Nch2(aborted) -Jch1 - Jch2 - Jch3 - Nch2(restored) - Nch3 - Nch4

Correct Case(with SW Abort on jch3): Nch1 - Nch2(aborted) -Jch1 - Jch2 - Jch3(aborted) - Nch2(restored) - Nch3 - Nch4

Observed unexpected behavior:

Fault1 (without SW abort on jch3): Nch1 - Nch2(aborted) - Jch1 - Jch2 - Jch3 - Nch3 - Nch4 (Nch2 not restored)

Fault2 (with SW abort on jch3): Nch1- Nch2 (aborted) - Jch1 - Jch2 - Jch3(aborted) - Nch4 (Nch2 not restored & Nch3 conversion skipped)

Workaround

It is possible to detect the unexpected behavior by using the CEOCFRx register. The CEOCFRx fields will not be set for a not restored or skipped channel, which indicates this issue has occurred. The CEOCFRx fields need to be checked before the next Normal chain execution (in scan mode). The CEOCFRx fields should be read by every ECH interrupt at the end of every chain execution.

ERR004186: ADC: Do not trigger ABORT or ABORTCHAIN prior to the start of CTU triggered ADC conversions and do not trigger ABORTCHAIN prior to the start of INJECTED triggered ADC conversions.

Description

When ADC_MCR[ABORT] or ADC_MCR[ABORTCHAIN] is set prior to the ADC receiving a CTU trigger, the next CTU triggered ADC conversion will not be performed and further CTU triggered ADC conversions will be blocked.

When ADC_MCR[ABORTCHAIN] is set prior to the ADC receiving an INJECTED trigger, the next INJECTED ADC conversion will not be performed. Following the ABORTCHAIN command the MCU behaviour does not meet the specification as ADC_ISR[JECH] is not set and ADC_MCR[ABORTCHAIN] is not cleared.

Workaround

Do not program ADC_MCR[ABORT] or ADC_MCR[ABORTCHAIN] before the start of ADC conversions.

The case when CTU triggered ADC conversions are blocked should be avoided however it is possible to reactivate CTU conversions by clearing and setting ADC_MCR[CTUEN].

ERR004340: LINFlexD: Buffer overrun can not be detected in UART Rx FIFO mode

Description

When the LINFlexD is configured in UART Receive (Rx) FIFO mode, the Buffer Overrun Flag (BOF) bit of the UART Mode Status Register (UARTSR) register is cleared in the subsequent clock cycle after being asserted.

User software can not poll the BOF to detect an overflow.

The LINFlexD Error Combined Interrupt can still be triggered by the buffer overrun. This interrupt is enabled by setting the Buffer Overrun Error Interrupt Enable (BOIE) bit in the LIN Interrupt enable register (LINIER). However, the BOF bit will be cleared when the interrupt routine is entered, preventing the user from identifying the source of error.

Workaround

Buffer overrun errors in UART FIFO mode can be detected by enabling only the Buffer Overrun Interrupt Enable (BOIE) in the LIN interrupt enable register (LINIER).

ERR005099: MC_ME: possibility of the machine check on low power mode exit (MPC5645S only)

Description

When executing from the flash and entering a Low-Power Mode (LPM) where the flash is in low-power or power-down mode, 2-4 clock cycles exist at the beginning of the RUNx to LPM transition during which a wakeup or interrupt will generate a checkstop due to the flash not being available on RUNx mode re-entry. This will cause either a checkstop reset or machine check interrupt.

Workaround

Workaround 1

The premise of Workaround 1 is to deal with the issue as it occurs: that is, enter and exit LPM as normal executing application code from flash memory. However, in the event the MCU exits LPM and the flash memory is not available, the next instruction fetch to flash memory will result in the core checkstop state as a result of a bus error. The checkstop state will generate a checkstop reset unless the machine check exception is enabled at the core. In the case of this problem occurring and the machine check exception being enabled, the application software can handle this event if the exception handler code is placed in RAM.

Workaround 2

The application can be configured to avoid the checkstop reset or machine check interrupt. To do this the code initiating the LPM transition has be executed in RAM. By this means, if the LPM mode is exited prior to the flash memory being returned to normal mode, any instruction fetch will be from the available RAM rather than the unavailable flash memory. The application software can then wait until flash is ready by only checking flash status flags in RAM. When the flash memory is ready, code execution can then return to the flash memory.

Workaround 3

For MPC5645S devices it is not necessary to run a work around from RAM, when the machine check is present due to a premature low power mode abort, user may prepare code to verify the flash status in the machine check exception subroutine. MPC5645S devices will cause a few attempts until the flash is successfully restarted and the user code is executed, after this user may return to run from flash again.

Refer to engineering bulleting EB770 on the web, for test case and work around examples.

ERR006026: DSPI: Incorrect SPI Frame Generated in Combined Serial Interface Configuration

Description

In the Combined Serial Interface (CSI) configuration of the Deserial Serial Peripheral Interface (DSPI) where data frames are periodically being sent (Deserial Serial Interface, DSI), a Serial Peripheral Interface (SPI) frame may be transmitted with incorrect framing.

The incorrect frame may occur in this configuration if the user application writes SPI data to the DSPI Push TX FIFO Register (DSPI_PUSHR) during the last two peripheral clock cycles of the Delay-after-Transfer (DT) phase. In this case, the SPI frame is corrupted.

Workaround

Workaround 1: Perform SPI FIFO writes after halting the DSPI.

To prevent writing to the FIFO during the last two clock cycles of DT, perform the following steps every time a SPI frame is required to be transmitted:

Step 1: Halt the DSPI by setting the HALT control bit in the Module Configuration Register (DSPI_MCR[HALT]).

Step 2: Poll the Status Register's Transmit and Receive Status bit (DSPI_SR[TXRXS]) to ensure the DSPI has entered the HALT state and completed any in-progress transmission. Alternatively, if continuous polling is undesirable in the application, wait for a fixed time interval such as 35 baud clocks to ensure completion of any in-progress transmission and then check once for DSPI_SR[TXRXS].

Step 3: Perform the write to DSPI_PUSHR for the SPI frame.

Step 4: Clear bit DSPI_MCR[HALT] to bring the DSPI out of the HALT state and return to normal operation.

Workaround 2: Do not use the CSI configuration. Use the DSPI in either DSI-only mode or SPI-only mode.

Workaround 3: Use the DSPI's Transfer Complete Flag (TCF) interrupt to reduce worst-case wait time of Workaround 1.

Step 1: When a SPI frame is required to be sent, halt the DSPI as in Step 1 of Workaround 1 above.

Step 2: Enable the TCF interrupt by setting the DSPI DMA/Interrupt Request Select and Enable Register's Transmission Complete Request Enable bit (DSPI_RSER[TCF_RE])

Step 3: In the TCF interrupt service routine, clear the interrupt status (DSPI_SR[TCF]) and the interrupt request enable (DSPI_RSER[TCF_RE]). Confirm that DSPI is halted by checking DSPI_SR[TXRXS] and then write data to DSPI_PUSHR for the SPI frame. Finally, clear bit DSPI_MCR[HALT] to bring the DSPI out of the HALT state and return to normal operation.

ERR006082: LINFlexD : LINS bits in LIN Status Register(LINSR) are not usable in UART mode.

Description

When the LINFlexD module is used in the Universal Asynchronous Receiver/Transmitter (UART) mode, the LIN state bits (LINS3:0]) in LIN Status Register (LINSR) always indicate the value zero. Therefore, these bits cannot be used to monitor the UART state.

Workaround

LINS bits should be used only in LIN mode.

ERR006481: NZ4C3/NZ7C3: Erroneous Resource Full Message under certain conditions

Description

The e200zx core Nexus interface may transmit an erroneous Resource Full Message (RFM) following a Program Trace history message with a full history buffer. The History information for both of the messages are the same and the RFM should have not been transmitted. This occurs when the instruction following the indirect branch instruction (which triggers the Program Trace History message) would have incremented the History field. The instructions must be executed in back to back cycles for this problem to occur. This is the only case that cases this condition.

Workaround

There are three possible workarounds for this issue.

(1) Tools can check to see if the Program Trace History message and proceeding Resource Full Message (RFM) have the same history information. If the history is the same, then the RFM is extraneous and can be ignored.

(2) Code can be rewritten to avoid the History Resource Full Messages at certain parts of the code. Insert 2 NOP instructions between problematic code. Or inserting an "isync" or a indirect branch some where in the code sequence to breakup/change the flow.

(3) If possible, use Traditional Program Trace mode can be used to avoid the issue completely. However, depending on other conditions (Nexus port width, Nexus Port speed, and other enabled trace types), overflows of the port could occur.

ERR006726: NPC: MCKO clock may be gated one clock period early when MCKO frequency is programmed as SYS_CLK/8.and gating is enabled

Description

The Nexus auxiliary message clock (MCKO) may be gated one clock period early when the MCKO frequency is programmed as SYS_CLK/8 in the Nexus Port Controller Port Configuration Register (NPC_PCR[MCKO_DIV]=111) and the MCKO gating function is enabled (NPC_PCR[MCKO_GT]=1). In this case, the last MCKO received by the tool prior to the gating will correspond to the END_MESSAGE state. The tool will not receive an MCKO to indicate the transition to the IDLE state, even though the NPC will transition to the IDLE state internally. Upon re-enabling of MCKO, the first MCKO edge will drive the Message Start/End Output (MSEO=11) and move the tool's state to IDLE.

Workaround

Expect to receive the MCKO edge corresponding to the IDLE state upon re-enabling of MCKO after MCKO has been gated.

ERR006976: MC_ME: SAFE mode not entered immediately on hardware-triggered SAFE mode request during STOP0 mode

Description

If a SAFE mode request is generated by the Reset Generation Module (MC_RGM) while the chip is in STOP0 mode, the chip does not immediately enter SAFE mode if STOP0 is configured as follows in the STOP0 Mode Configuration register (ME_STOP0_MC):

- the system clock is disabled (ME_STOP0_MC[SYSCLK] = 0b1111)

- the internal RC oscillator is enabled (ME_STOP0_MC[IRCON] = 0b1)

In this case, the chip will remain in STOP0 mode until an interrupt request or wakeup event occurs, causing the chip to return to its previous RUNx mode, after which the still pending SAFE mode request will cause the chip to enter SAFE mode.

Workaround

There are two possibilities.

1. Configure the internal RC oscillator to be disabled during STOP0 mode (ME_STOP0_MC[IRCON] = 0b0) if the device supports it.

2. Prior to entering STOP0 mode, configure all hardware-triggered SAFE mode requests that need to cause an immediate transition from STOP0 to SAFE mode to be interrupt requests. This is done in the MC_RGM's ‘Functional' Event Alternate Request register (RGM_FEAR).

ERR007026: FLASH: Continuously cycling from STANDBY to RUN mode may disturb flash array content.

Description

Repetitively power cycling the flash with STANDBY mode may disturb flash bits, causing them to flip from a 1 (erased state) to a 0 (programmed state) when switching from STANDBY back to RUN mode when 80mV < VDD12 < 150mV. This voltage range is determined by characterization on a small sample size for typical devices.

For reference, a minimum 2.95M wakeups cycles from STANDBY to RUN with 80mV < VDD12 < 150mV are needed to disturb the flash contents.

Workaround

When exiting STANDBY mode, ensure that the VDD12 pins are below or above the defined range, from 80mV to 150mV, before entering a RUN mode.

Based on typical application conditions, Freescale recommends that VDD12 pins are below the lowest range value (80mV) with the following options:

1. Increase the time the MCU remains in STANDBY

2. Use a pull down resistor on VDD12

These options are recommendations or guidelines that each customer needs to analyze and adjust to their specific application.

Refer to Engineering Bulletin EB795, for a detailed analysis on the workaround options.

ERR007120: NZxC3: DQTAG implemented as variable length field in DQM message

Description

The e200zx core implements the Data Tag (DQTAG) field of the Nexus Data Acquisition Message (DQM) as a variable length packet instead of an 8-bit fixed length packet. This may result in an extra clock ("beat") in the DQM trace message depending on the Nexus port width selected for the device.

Workaround

Tools should decode the DQTAG field as a variable length packet instead of a fixed length packet.

ERR007274: LINFlexD: Consecutive headers received by LIN Slave triggers the LIN FSM to an unexpected state

Description

As per the Local Interconnect Network (LIN) specification, the processing of one frame should be aborted by the detection of a new header sequence and the LIN Finite State Machine (FSM) should move to the protected identifier (PID) state. In the PID state, the LIN FSM waits for the detection of an eight bit frame identifier value.

In LINFlexD, if the LIN Slave receives a new header instead of data response corresponding to a previous header received, it triggers a framing error during the new header's reception and returns to IDLE state.

Workaround

The following three steps should be followed -

1) Configure slave to Set the MODE bit in the LIN Time-Out Control Status Register (LINTCSR[MODE]) to '0'.

2) Configure slave to Set Idle on Timeout in the LINTCSR[IOT] register to '1'. This causes the LIN Slave to go to an IDLE state before the next header arrives, which will be accepted without any framing error.

3) Configure master to wait for Frame maximum time (T Frame_Maximum as per LIN specifications) before sending the next header.

Note:

THeader_Nominal = 34 * TBit

TResponse_Nominal = 10 * (NData + 1) * TBit

THeader_Maximum = 1.4 * THeader_Nominal

TResponse_Maximum = 1.4 * TResponse_Nominal

TFrame_Maximum = THeader_Maximum + TResponse_Maximum

where TBit is the nominal time required to transmit a bit and NData is number of bits sent.

ERR007394: MC_ME: Incorrect mode may be entered on low-power mode exit.

Description

For the case when the Mode Entry (MC_ME) module is transitioning from a run mode (RUN0/1/2/3) to a low power mode (HALT/STOP/STANDBY*) if a wake-up or interrupt is detected one clock cycle after the second write to the Mode Control (ME_MCTL) register, the MC_ME will exit to the mode previous to the run mode that initiated the low power mode transition.

Example correct operation DRUN->RUN1-> RUN3->STOP->RUN3

Example failing operation DRUN->RUN1-> RUN3->STOP->RUN1

Note STANDBY mode is not available on all MPC56xx microcontrollers

Workaround

To ensure the application software returns to the run mode (RUN0/1/2/3) prior to the low power mode (HALT/STOP/STANDBY*) it is required that the RUNx mode prior to the low power mode is entered twice.

The following example code shows RUN3 mode entry prior to a low power mode transition.

ME.MCTL.R = 0x70005AF0; /* Enter RUN3 Mode & Key */

ME.MCTL.R = 0x7000A50F; /* Enter RUN3 Mode & Inverted Key */

while (ME.GS.B.S_MTRANS) {} /* Wait for RUN3 mode transition to complete */

ME.MCTL.R = 0x70005AF0; /* Enter RUN3 Mode & Key */

ME.MCTL.R = 0x7000A50F; /* Enter RUN3 Mode & Inverted Key */

while (ME.GS.B.S_MTRANS) {} /* Wait for RUN3 mode transition to complete */

/* Now that run mode has been entered twice can enter low power mode */

/* (HALT/STOP/STANDBY*) when desired. */

ERR007589: LINFlexD: Spurious timeout error when switching from UART to LIN mode or when resetting LINTCSR[MODE] bit in LIN mode

Description

If the LINFlexD module is enabled in Universal Asynchronous Receiver/Transmitter (UART) mode and the value of the MODE bit of the LIN Timeout Control Status register (LINTCSR) is 0 (default value after reset), any activity on the transmit or receive pins will cause an unwanted change in the value of the 8-bit field Output Compare Value 2 (OC2) of the LIN Output Compare register (LINOCR).

If the LINFlexD module is enabled in LIN mode and the value of the MODE bit of the LIN Timeout Control Status register (LINTCSR) is changed from ‘1’ to ‘0’, then the old value of the Output Compare Value 1 (OC1) and Output Compare Value 2 (OC2) of the LIN Output Compare register (LINOCR) is retained.

As a consequence, if the module is reconfigured from UART to Local Interconnect Network (LIN) mode, or LINTCSR MODE bit is changed from ‘1’ to ‘0’, an incorrect timeout exception is generated when the LIN communication starts.

Workaround

If the LINFlexD module needs to be switched from UART mode to LIN mode, before writing UARTCR[UART] to 1, ensure that the LINTCSR[MODE] is first set to 1.

If the LINFlexD module is in LIN mode and LINTCSR[MODE] needs to be switched from 1 to 0 in between frames, the LINOCR must be set to 0xFFFF by software.

ERR007688: RTC: An API interrupt may be triggered prematurely after programming the API timeout value

Description

When the API is enabled (RTCC[APIEN]), the API interrupt flag is enabled (RTCC[APIIE]) and the API timeout value (RTCC[APIVAL]) is programmed the next API interrupt may be triggered before the programmed API timeout value. Successive API Interrupts will be triggered at the correct time interval.

Workaround

The user must not use the first API interrupt for critical timing tasks.

ERR007953: ME: All peripherals that will be disabled in the target mode must have their interrupt flags cleared prior to target mode entry

Description

Before entering the target mode, software must ensure that all interrupt flags are cleared for those peripheral that are programmed to be disabled in the target mode. A pending interrupt from these peripherals at target mode entry will block the mode transition or possibly lead to unspecified behaviour.

Workaround

For those peripherals that are to be disabled in the target mode the user has 2 options:

1. Mask those peripheral interrupts and clear the peripheral interrupt flags prior to the target mode request.

2. Through the target mode request ensure that all those peripheral interrupts can be serviced by the core.

ERR008933: LINFlexD: Inconsistent sync field may cause an incorrect baud rate and the Sync Field Error Flag may not be set

Description

When the LINFlexD module is configured as follows:

1. LIN (Local Interconnect Network) slave mode is enabled by clearing the Master Mode Enable bit in the LIN Control Register 1 (LINCR1[MME] = 0b0)

2. Auto synchronization is enabled by setting LIN Auto Synchronization Enable (LINCR1[LASE] = 0b1)

The LINFlexD module may automatically synchronize to an incorrect baud rate without setting the Sync Field Error Flag in the LIN Error Status register (LINESR[SFEF]) in case Sync Field value is not equal to 0x55, as per the Local Interconnect Network (LIN) specification.

The auto synchronization is only required when the baud-rate in the slave node can not be programmed directly in software and the slave node must synchronize to the master node baud rate.

Workaround

There are 2 possible workarounds.

Workaround 1:

When the LIN time-out counter is configured in LIN Mode by clearing the MODE bit of the LIN Time-Out Control Status register (LINTCSR[MODE]= 0x0]):

1. Set the LIN state Interrupt enable bit in the LIN Interrupt Enable register (LINIER[LSIE] = 0b1)

2. When the Data Reception Completed Flag is asserted in the LIN Status Register (LINSR[DRF] = 0b1) read the LIN State field (LINSR[LINS])

3. If LINSR[LINS]= 0b0101, read the Counter Value field of the LIN Time-Out Control Status register (LINTCSR[CNT]), otherwise repeat step 2

4. If LINTCSR[CNT] is greater than 0xA, discard the frame.

When the LIN Time-out counter is configured in Output Compare Mode by setting the LINTCSR[MODE] bit:

1. Set the LIN State Interrupt Enable bit in the LIN Interrupt Enable register (LINIER[LSIE])

2. When the Data Reception Completed flag bit is asserted in the LIN Status Register (LINSR[DRF] = 0b1), read the LINSR[LINS] field

3. If LINSR[LINS]= 0b0101, store LINTCSR[CNT] value in a variable (ValueA), otherwise repeat step 2

4. Clear LINSR[DRF] flag by writing LINSR[LINS] field with 0xF

5. Wait for LINSR[DRF] to become asserted again and read LINSR[LINS] field

6. If LINSR[LINS] = 0b0101, store LINTCSR[CNT] value in a variable (ValueB), else repeat step 4

7. If ValueB - ValueA is greater than 0xA, discard the frame

Workaround 2:

Do not use the auto synchronization feature (disable with LINCR1[LASE] = 0b0) in LIN slave mode.

ERR008951: I2C: Attempting a start cycle while the bus is busy may generate a short clock pulse

Description

When the I2C (Inter-Integrated Circuit) is operating in a multi-master network and a start cycle is attempted by the I2C device when the bus is busy, the attempting master will lose arbitration as expected but a short extra clock cycle is generated in the bus. After losing arbitration, the master switches to slave mode but it does not detect the short clock pulse. The acknowledge signal is expected at the ninth clock by the current bus master but it is not sent as expected due to the undetected short clock pulse.

Workaround

Software must ensure that the I2C BUS is idle by checking the bus busy bit in the I2C Bus Status Register (I2C_IBSR.IBB) before switching to master mode and attempting a Start cycle.

ERR008970: LINFlexD: Spurious bit error in extended frame mode may cause an incorrect Idle State

Description

The LINFlexD module may set a spurious Bit Error Flag (BEF) in the LIN Error Status Register (LINESR), when the LINFlexD module is configured as follows:

- Data Size greater than eight data bytes (extended frames) by configuring the Data Field Length (DFL) bitfield in the Buffer Identifier Register (BIDR) with a value greater than seven (eight data bytes)

- Bit error is able to reset the LIN state machine by setting Idle on Bit Error (IOBE) bit in the LIN Control Register 2 (LINCR2)

As consequence, the state machine may go to the Idle State when the LINFlexD module tries the transmission of the next eight bytes, after the first ones have been successfully transmitted and Data Buffer Empty Flag (DBEF) was set in the LIN Status Register (LINSR).

Workaround

Do not use the extended frame mode by configuring Data Field Length (DFL) bit-field with a value less than eight in the Buffer Identifier Register (BIDR) (BIDR[DFL] < 8)

ERR009066: PAD RING: The Device may lock-up into reset loop while exiting from standby mode or at power up

Description

The device may lock-up (core will not execute code) while exiting Standby mode or during power up of the VDD12 supply if the TCK pin is in floating or high state. The watchdog timer will continuously trigger a reset signal at the default 15 ms interval.

A destructive reset triggered by the watchdog or by any other reset source will not recover the device from the lock-up state.

Workaround

If a lock-up occurs, the device operation can be recovered by driving the TCK pin at logic level low and the device will recover at next reset de-assertion.

Use 4.7K ohm resistor to pull TCK pin down to prevent any occurrence of the failure.

ERR009682: SPI: Inconsistent loading of shift register data into the receive FIFO following an overflow event

Description

In the Serial Peripheral Interface (SPI) module, when both the receive FIFO and shift register are full (Receive FIFO Overflow Flag bit in Status Register is set (SR [RFOF] = 0b1)) and then the Clear Rx FIFO bit in Module Configuration Register (MCR [CLR_RXF]) is asserted to clear the receive FIFO, shift register data is loaded into the receive FIFO after the clear operation completes.

Workaround

1. Avoid a receive FIFO overflow condition (SR[RFOF] should never be 0b1). To do this, monitor the RX FIFO Counter field of the Status Register (SR[RXCTR]) which indicates the number of entries in receive FIFO and clear before the counter equals the FIFO depth.

2. Alternatively, after every receive FIFO clear operation (MCR[CLR_RXF] = 0b1) following a receive FIFO overflow (SR[RFOF] = 0b1) scenario, perform a single read from receive FIFO and discard the read data.

ERR009764: SARADC : DMA interface limitation depending on PBRIDGE/SARADC clock ratio

Description

The Successive Approximation Register Analog-to-Digital Converter (SARADC) modules can trigger a Direct Memory Access (DMA) request through the DMA Enable (DMAE) register interface.

When the SARADC clock (SAR_CLK) frequency is slower than half of the peripheral bridge (PBRIDGEx_CLK) clock frequency, the SARADC may trigger a spurious transfer request to the DMA module after the completion of a first valid transfer.

Workaround

Setting the DMA clear sequence enable (DCLR) bit in the DMAE register (DMAE[DCLR] = 1) forces the clearing of the DMA request on read access to the data register and therefore prevents the spurious DMA transfer request.

In case the Internal Channel Data Registers (ICDRn) are only accessed through DMA module (i.e. there are no bus accesses to ICDRn registers triggered by other than DMA bus master when the DMAE[DMAEN] bit is set), it is possible to configure DMAE[DCLR] bit to '1'. This will clear DMA transfer request on the first DMA read access, ensuring both that DMA triggered transfer will complete successfully and that no other spurious DMA request will be triggered.

This work-around can be applied when any of below condition can be met:

- frequency ratio PBRIDGEx_CLK/SAR_CLK <= 8/3

- PBRIDGEx_CLK is 40MHz and SAR_CLK >= 14MHz

ERR009976: DSPI: Incorrect data received by master with Modified transfer format enabled when using Continuous serial communication clock mode

Description

When the Deserial Serial Peripheral Interface (DSPI) module is configured as follows:

1. Master mode is enabled (Master/Slave Mode Select bit in Module Configuration Register is set (DSPI_MCR [MSTR] = 0b1))

2. Modified transfer format is enabled (Modified Transfer Format Enable bit in Module Configuration Register is set (DSPI_MCR [MTFE] = 0b1))

3. Continuous serial communication clock mode is enabled (Continuous SCK Enable bit in Module Configuration Register is set (DSPI_MCR [CONT_SCKE] = 0b1))

In this configuration if the frame size of the current frame is greater than the frame size of the next received frame, corrupt frames are received in two scenarios:

a) Continuous Peripheral Chip Select Enable bit in PUSH TX FIFO Register is set (DSPI_PUSHR [CONT] = 0b1)

b) DSPI_PUSHR [CONT] = 0b0 and lower significant bit of the frame is transferred first (LSB first bit in Clock and Transfer Attributes Register is set (DSPI_CTAR [LSBFE] =0b1))

Workaround

To receive correct frames:

a) When DSPI_PUSHR [CONT] = 0b1, configure the frame size of the current frame less than or equal to the frame size of the next frame (for all frames).

b) When DSPI_PUSHR [CONT] = 0b0, configure DSPI_CTAR [LSBFE] = 0b0. Alternatively, configure the frame size of the current frame less than or equal to the frame size of the next frame (for all frames).

Make sure that for all received frames, the bits are read equal to their respective frame sizes and any extra bits during POP operation are masked.

ERR009978: eMIOS: Unexpected channel flag assertion during GPIO to MCB mode transition

Description

When changing an Enhanced Modular IO Subsystem (eMIOS) channel mode from General Purpose Input/Output (GPIO) to Modulus Counter Buffered (MCB) mode, the channel flag in the eMIOS Channel Status register (eMIOS_Sn[FLAG]) may incorrectly be asserted. This will cause an unexpected interrupt or DMA request if enabled for that channel.

Workaround

In order to change the channel mode from GPIO to MCB without causing an unexpected interrupt or DMA request, perform the following steps:

(1) Clear the FLAG enable bit in the eMIOS Control register (eMIOS_Cn[FEN] = 0).

(2) Change the channel mode (eMIOS_Cn[MODE]) to the desired MCB mode.

(3) Clear the channel FLAG bit by writing '1' to the eMIOS Channel Status register FLAG field (eMIOS_Sn[FLAG] = 1).

(4) Set the FLAG enable bit (eMIOS_Cn[FEN] = 1) to re-enable the channel interrupt or DMA request reaction.

ERR010755: DSPI: Transmit and Receive FIFO fill flags in status register is not cleared when DMA is improperly configured

Description

The Deserial/Serial Peripheral Interface Transmit and Receive First In/First Out (FIFO) buffers can request additional information to be transferred via the Direct Memory Access (DMA) module when either the Transmit or Receive FIFO Fill/Drain Flags are set in the DSPI Status Register (SR[TFFF/RFDF]). However, the Transmit Fill Flag indicates that at least 1 location each (2 bytes each) in the Transmit and Command FIFOs is available to be written. It does not indicate that the FIFO is empty. Similarly, Receive FIFO fill flag only indicates at least 1 location (2 bytes) of the FIFO is available to be read. It does not indicate that the FIFO is full. If the DMA is configured to transfer more than 1 FIFO location size of data, the FIFO Fill/Drain Flags may not be properly cleared indicating that the FIFO is not full even when the FIFO is actually full (for Transmit FIFO) and not empty when the FIFO is actually empty (for Receive FIFO).

Workaround

Properly configure the DMA to fill/drain only 2 bytes to Transmit, Command and Receive FIFOs. Use the DMA loop to transfer more data if needed.

ERR011214: [MDDRC]: Do not perform unaligned address transfers to MDDRC from different bus masters (e.g: PowerPC core)

Description

MDDRC does not support unaligned bus access on any of its slave port. Any unaligned address access reaching MDDRC is treated as aligned address access by ignoring the appropriate address LSBs. Unaligned address writes reaching MDDRC will perform write operation on corresponding aligned address while unaligned reads returns corresponding aligned address location contents of the DRAM. E.g. an unaligned 32-bit write access with address offset of 0x2 is treated as aligned double word write access at address offset 0x0.

Workaround

Do not perform unaligned address transfers from the rainbow bus masters e.g. PowerPC core.

ERR011293: EMIOS: For any UC operating in OPWFMB mode the Channel Count register should not be written with a value greater than Channel B Data register value

Description

For any Unified Channel (UC) running in Output Pulse-Width and Frequency Modulation Buffered (OPWFMB) mode, Channel Control Register MODE bitfield = 7'h1011000 or 7'h1011010, the internal counter runs from 0x1 to Channel B Data register value.

The internal counter can be overwritten by software using the Chanel Count register during 'freeze' operation.

If a counter wrap occurs due to overwriting of the counter with a value greater than its expiry value (B Data Register value); than the output signal behavior cannot be guaranteed.

Workaround

For any UC operating in OPWFMB mode the Channel Count register should not be written with a value greater than Channel B Data register value.

ERR011294: EMIOS: OPWFMB and MCB mode counter rollover resets the counter to 0x0 instead of 0x1 as mentioned in the specification

Description

When the enhanced Modular Input/Output System (eMIOS) is used in Output Pulse-Width and Frequency Modulation Buffered (OPWFMB) or Modulus Counter Buffered (MCB) modes, when the counter rolls over, the counter returns to 0x0 instead of 0x1 as specified in the reference manual.

Workaround

In order to avoid the counter wrap condition:

1. Make sure internal counter value is within the 0x1 to (B1 register) value range when the OPWFMB mode is entered.

2. Overwrite of Channel Count register by forcing 'freeze' in OPWFMB mode should not be outside the range of 0x1 to (B register) value.

ERR011295: EMIOS: In OPWFMB mode, A1/B1 registers do not get reloaded with A2/B2 register values if counter value returns 0x1 after counter wrap condition

Description

In Output Pulse-Width and Frequency Modulation Buffered (OPWFMB) mode, A1/B1 registers do not get reloaded with A2/B2 register values if counter value returns 0x1 after counter wrap condition.

In order to avoid the counter wrap condition make sure internal counter value is within the 0x1 to B1 register value range when the OPWFMB mode is entered. Also overwriting of Channel Count register by forcing 'freeze' in OPWFMB mode should not take internal counter outside 0x1 to B register value.

Workaround

In order to avoid the counter wrap condition:

1. Make sure internal counter value is within the 0x1 to (B1 register) value range when the OPWFMB mode is entered.

2. Overwrite of Channel Count register by forcing 'freeze' in OPWFMB mode should not be outside the range of 0x1 to (B register) value.

ERR050459: SXOSC: Clock output may contain extra clock pulses in Normal mode

Description

The 32 kHz Slow External Crystal Oscillator (SXOSC) may generate an extra clock pulse per clock period when configured in Normal mode using an external crystal. The SXOSC correctly generates positive clock edges aligned to the external crystal clock transitions but an extra clock pulse may be generated near the positive edge of the clock waveform. The SXOSC interface to the external crystal is not affected. This issue may only occur in the default Normal mode and operates as expected in Bypass mode (i.e. OSC_CTL[OSCBYP]=1).

The SXOSC clock source may be selected as the clock source for the Real Time Counter and Autonomous Periodic Interrupt (RTC/API) and also as the clock to be measured in the CMU Frequency Meter (via the CMU_FDR register). The RTC/API and CMU Frequency Meter counters may get an extra clock per SXOSC clock period causing a higher than expected count (affecting the calculated time or wakeup duration) or may theoretically cause a corrupted count value. The occurrence of the potential clock pulse may vary from clock period to clock period ranging from no extra clock pulse to one extra clock pulse per period. SXOSC may also be selected to be reflected to the CLKOUT pin. Worse case conditions to produce an extra clock pulse are lower temperatures.

Workaround

Use the SXOSC in Bypass mode instead of Normal mode by setting OSC_CTL[OSCBYP]. Bypass mode does not support an external crystal and thus an external clock source is needed.

Select other clock sources for the RTC/API and CMU Frequency Meter. The RTC/API supports other clock sources which include the 128 kHZ Slow Internal RC Oscillator (SIRC), 16 MHz Fast Internal RC Oscillator (FIRC), and 4-16 MHz Fast External Crystal Oscillator (FXOSC).

ERR050782: e200: Time Base TBU register contains wrong value during TBL rollover

Description

The e200 Time Base (TB) facility is a 64-bit structure provided for maintaining the time of day and operating interval timers. The TB consists of two 32-bit registers - time base upper (TBU) and time base lower (TBL). TBU and TBL are concatenated to provide a long-period 64-bit counter. TBL increments until its value becomes 0xFFFF_FFFF. The intended behavior is that at the next increment when the TBL value becomes 0x0000_0000 that the TBU value is incremented. But the actual behavior is that after the TBL value becomes 0x0000_0000, the TBU value will not increment until the transition of the TBL value to 0x0000_0001.

Workaround

Software will need to take care about the wrong TBU value during TBL rollover. Use the following sequence for reading TBU and TBL values:

loop:

mfspr r12, TBL

mfspr r3, TBU

mfspr r4, TBL

cmpl r4,r12

se_blt loop

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Suitability for use in automotive applications — This NXP product has been qualified for use in automotive applications. If this product is used by customer in the development of, or for incorporation into, products or services (a) used in safety critical applications or (b) in which failure could lead to death, personal injury, or severe physical or environmental damage (such products and services hereinafter referred to as "Critical Applications"), then customer makes the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, safety, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. As such, customer assumes all risk related to use of any products in Critical Applications and NXP and its suppliers shall not be liable for any such use by customer. Accordingly, customer will indemnify and hold NXP harmless from any claims, liabilities, damages and associated costs and expenses (including attorneys' fees) that NXP may incur related to customer's incorporation of any product in a Critical Application.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 7/2023
Document identifier: MPC5645S_0N29D