Mask Set Errata for Mask 1N52N

This report applies to mask 1N52N for these products:

- MKL28Z512VDC7
- MKL28Z512VLL7

Table 1. Errata and Information Summary

Erratum ID	Erratum Title
ERR050117	FAC: Execute-only access control feature has been deprecated
ERR009464	FIRC: A transfer error is received when writing to the SCG_FIRCSTAT register
ERR009380	FlexIO: Reading FlexIO register when FlexIO functional clock is disabled results in a bus hang
ERR009364	LPI2C: HS-mode signal on Repeated START uses Fast mode timing for the (Repeated) START hold time
ERR009365	LPI2C: Master does not always end the transfer when the NACK detect flag is set
ERR009881	LPIT: LPIT does not stop in the Debug or Low Power modes regardless of the DBG_EN or DOZE_EN bits setting in the LPITx_MCR register when Timer Reload On Trigger is set.
ERR010180	SCG: Clock switch may hang if SCG_RCCR is written to the switch system clock source with a different divide ratio while an external reset is asserted
ERR010181	SCG: Only clearing the SOSCEN bit of the SCG_SOSCCSR register cannot disable the SOSC analog circuit

Table 2. Revision History

Revision	Changes
6Sep2019	The following erratum was added. • ERR050117
14Apr2016	Initial release.

ERR050117: FAC: Execute-only access control feature has been deprecated

Description: The FAC feature is no longer recommended for use.

Workaround: Do not program the XACCn registers to use the FAC feature.

ERR009464: FIRC: A transfer error is received when writing to the SCG_FIRCSTAT register

Description: When writing to the SCG_FIRCSTAT register, a transfer error occurs that causes the code execution to stall at the write point. The SCG_FIRCSTAT register is used to trim FIRC, which is already trimmed at the factory.

Workaround: Do not use the SCG_FIRCSTAT register to trim FIRC as it is already trimmed at the factory.

ERR009380: FlexIO: Reading FlexIO register when FlexIO functional clock is disabled results in a bus hang

Description: Accessing a FlexIO register when the FlexIO functional clock is disabled (the clock source configured to 0 in PCC_FLEXIO0[PCS], or the selected clock source is disabled) will hang the bus and the access will stall forever.

Workaround: Always enable the FlexIO functional clock before accessing any FlexIO register.

ERR009364: LPI2C: HS-mode signal on Repeated START uses Fast mode timing for the (Repeated) START hold time

Description: The internal HS-mode signal on Repeated START updates after the (Repeated) START hold time, which causes the (Repeated) START hold time to use the Fast mode timing in LPI2Cx_MCCR0 instead of the HS-mode timing in LPI2Cx_MCCR1. This action only affects the (Repeated) START hold time and only on a Repeated START, that initiates HS-mode. It does not affect the subsequent HS-mode data transfer. This issue occurs when the first START is at the normal speed and the second START is in the HS-mode.

Workaround: This issue only lengthens a hold time delay in HS mode and does not affect the HS mode protocol.

ERR009365: LPI2C: Master does not always end the transfer when the NACK detect flag is set

Description: When the NACK detect flag is set, the LPI2C master should terminate the existing transfer and block a new transfer until the flag clears. However, when the NACK detect flag is set, and a Repeated START is queued as the next operation, then the LPI2C Master does not terminate the transfer and instead continues the transfer until a STOP condition is sent. This satisfies the requirement of the I2C specification, but does not halt the transfer when it detects an unexpected NACK.

Workaround: Confirm that both the NACK detect flag (NDF) and the end packet flag (EPF) in the LPI2Cx_MSR register are set before clearing the NACK detect flag. When both flags are set, write a STOP condition generation command (0x3FF) to the Transmit Data register and then clear the flags.

ERR009881: LPIT: LPIT does not stop in the Debug or Low Power modes regardless of the DBG_EN or DOZE_EN bits setting in the LPITx_MCR register when Timer Reload On Trigger is set.

Description: LPIT does not stop in the Debug or Low Power modes regardless of the DBG_EN or DOZE_EN bits setting in the LPITx_MCR register when Timer Reload On Trigger is set. For example, the LPIT reloading operation occurs in the Debug mode on every rise edge of the trigger signal when setting LPITx_MCR[DBG_EN] = 0. The LPIT reloading operation also occurs in the WAIT/STOP/VLPS mode on a trigger event when LPITx_MCR[DOZE_EN] = 0.

Workaround: Gate off the LPIT functional clock by clearing the PCC_LPIT0[PCS] bit just before entering the debug mode when LPITx_MCR[DBG_EN] = 0 and/or before entering the low power mode when LPITx_MCR[DOZE_EN] = 0.

For example, to stop LPIT in debug mode perform the following steps in the debugger.

1. Clear PCC_LPIT0 [PCS] in the Debugger Register window to gate off the LPIT functional clock when the code stops at a breakpoint.

2. Resume the normal debug operation.

To stop LPIT in WAIT/STOP/VLPS modes, add a code line to clear PCC_LPIT0 [PCS] to gate off the LPIT functional clock prior to entering these low power modes.

ERR010180: SCG: Clock switch may hang if SCG_RCCR is written to the switch system clock source with a different divide ratio while an external reset is asserted

Description: SCG: Clock switch may hang if SCG_RCCR is written to the switch system clock source with a different divide ratio while an external reset is asserted. For example, if SCG is in the FIRC mode and then configured to the SIRC mode,by writing the RCCR to switch system clock with a different divide ratio, during which the external reset is asserting, then the clock switch may hang.

Workaround: To recover the clock switch another reset must be issued.

ERR010181: SCG: Only clearing the SOSCEN bit of the SCG_SOSCCSR register cannot disable the SOSC analog circuit

Description: When SOSC is stable (after 4096 cycles) in crystal mode, only clearing the SOSCEN bit of the SCG_SOSCCSR register cannot disable the SOSC analog circuit. This may cause clock output unstable especially when switching from a low gain mode to a high gain mode.

Workaround: In the high gain mode, clearing both the SOSCEN and EREFS bits can disable SOSC digital and analog circuit within two crystal clocks.

In the low gain mode:

1. Switch to the high gain mode.

2. Wait for some time.

3. Recommend at least 8 NOP instructions to ensure that the crystal clock is stable.

4. Clear both the SOSCEN and EREFS bits to disable SOSC digital and analog circuit within two crystal clocks.

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 reserves the right to make changes without further notice to any products herein.

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.