# EFM32 Gecko
# EFM32TG11 Errata

This document contains information on the EFM32TG11 errata. The latest available revision of this device is revision B.

Errata that have been resolved remain documented and can be referenced for previous revisions of this device.

The device data sheet explains how to identify the chip revision, either from package marking or electronically.

Errata effective date: September, 2020.

# 1. Errata Summary

The table below lists all known errata for the EFM32TG11 and all unresolved errata in revision B of the EFM32TG11.

**Table 1.1. Errata Overview**

| Designator | Title/Problem | Workaround Exists | Exists on Revision: | |
|---|---|---|---|---|
| | | | A | B |
| ADC_E224 | ADC Warm-Up Ready Can Cause IDAC, ACMP, or CSEN to Not Function | Yes | X | — |
| ADC_E225 | Using the ADC in High-Accuracy Bias Mode Will Force All Analog Peripherals to High-Accuracy Bias Mode | No | X | — |
| ADC_E228 | Limited ADC Sampling Frequency in EM2 | No | X | — |
| BU_E201 | Extra Current from BUVDD to VREGVDD in BUMODE when Pulling Main Supply Low | No | X | — |
| BU_E202 | Extra Current from DVDD to GND in BUMODE when Pulling BUVDD Low | Yes | X | — |
| CMU_E203 | Peak Detector May Not Trip | Yes | X | — |
| CMU_E204 | Initial Oscillator Calibration | Yes | X | — |
| CSEN_E201 | CSEN_DATA in Debug Mode | Yes | X | X |
| CSEN_E202 | CSEN Baseline DMA Transfers | Yes | X | X |
| CUR_E204 | Extra EM4S Current When ANASW Set to 1 | Yes | X | X |
| DBG_E204 | Debug Recovery with JTAG Does Not Work | Yes | X | X |
| DBG_E205 | Incorrect Debug AP Base Address Register Value | Yes | X | — |
| EMU_E214 | Device Erase Cannot Occur if Voltage Scaling Level is Too Low | Yes | X | — |
| EMU_E220 | DECBOD Reset During Voltage Scaling After EM2 or EM3 Wake-up | Yes | X | X |
| I2C_E202 | Race Condition Between Start Detection and Timeout | Yes | X | — |
| I2C_E203 | I2C Received Data Can be Shifted | Yes | X | — |
| I2C_E205 | Go Idle Bus Idle Timeout Does Not Bring Device to Idle State | Yes | X | — |
| I2C_E206 | Slave Holds SCL Low After Losing Arbitration | Yes | X | X |
| I2C_E207 | I2C Fails to Indicate New Incoming Data | Yes | X | X |
| LCD_E201 | LCD Boost Mode Does Not Work | No | X | — |
| LES_E201 | LFPRESC Can Extend Channel Start-Up Delay | Yes | X | X |
| RMU_E202 | External Debug Access Not Available After Watchdog or Lockup Full Reset | Yes | X | X |
| RMU_E203 | AVDD Ramp Issue | Yes | X | — |
| TRNG_E201 | Possible Triggering of Noise Alarm | Yes | X | — |
| TRNG_E202 | Standards Non-compliance | No | X | — |
| TIMER_E202 | Continuous Overflow and Underflow Interrupts in Quadrature Counting Mode | Yes | X | X |
| USART_E204 | IrDA Modulation and Transmission of PRS Input Data | Yes | X | X |

| Designator | Title/Problem | Workaround Exists | Exists on Revision: | |
|---|---|---|---|---|
| | | | A | B |
| USART_E205 | Possible Data Transmission on Wrong Edge in Synchronous Mode | Yes | X | X |
| USART_E206 | Additional SCLK Pulses Can Be Generated in USART Synchronous Mode | Yes | X | X |
| WDOG_E201 | Clear Command is Lost Upon EM2 Entry | Yes | X | X |
| WTIMER_E201 | Continuous Overflow and Underflow Interrupts in Quadrature Counting Mode | Yes | X | X |

# 2. Current Errata Descriptions

## 2.1 CSEN_E201 – CSEN_DATA in Debug Mode

| ***Description of Errata*** |
|---|
| Reading CSEN_DATA in debug mode inadvertently clears pending CSEN data DMA requests. |
| ***Affected Conditions / Impacts*** |
| Reads of CSEN_DATA clear pending receive data DMA requests. This would be expected in normal operation as the DMA reads CSEN_DATA to transfer newly acquired results. These reads are intentional, but any read of CSEN_DATA, including while in debug mode, has the same effect. Thus, viewing the CSEN module registers in a debugger, such as in Simplicity Studio, can inadvertently clear pending CSEN DMA requests resulting in subsequent data being received out of order and with insertions of random data. |
| ***Workaround*** |
| Do not use a debugger to read the CSEN registers while DMA is enabled. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

## 2.2 CSEN_E202 – CSEN Baseline DMA Transfers

| ***Description of Errata*** |
|---|
| DMA transfers to CSEN_DMBASELINE do not occur in the expected order. |
| ***Affected Conditions / Impacts*** |
| When using delta modulation, a baseline value must be written to CSEN_DMBASELINE before each conversion. However, when DMA is used to do this, these writes occur after the desired conversion instead of before the conversion as is required. This means that in a given sequence of conversions serviced by DMA, the write to CSEN_DMBASELINE that should happen before conversion N is actually written in advance of conversion N + 1, leading to potentially erroneous results. |
| ***Workaround*** |
| Manually write the first value to CSEN_DMBASELINE and then use the DMA to perform subsequent baseline writes. Therefore, in the case of a sequence consisting of four conversions, the first baseline value would be written to CSEN_DMBASELINE under software control (e.g., before the conversion trigger occurs). The next three values can be written by the DMA after the first and each subsequent conversion occurs.<br><br>After the final conversion, which would be the fourth in this example, the DMA will service a final write request to CSEN_DMBASELINE. This final transfer can be (1) a dummy value if no further conversions are required, (2) the initial baseline value in the case where conversions are repeated in a loop, or (3) the initial baseline value for a new, yet-to-be-triggered series of conversions. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.3 CUR_E204 – Extra EM4S Current When ANASW Set to 1

| ***Description of Errata*** |
|---|
| After entering EM4S while analog peripherals are powered from DVDD (ANASW = 1 in the EMU_PWRCTRL register), the device will see an additional 30-300 µA of current consumption, depending on the AVDD voltage. |
| ***Affected Conditions / Impacts*** |
| Systems using EM4S will see an additional 30-300 µA of current draw if ANASW in EMU_PWRCTRL is set to 1. |
| ***Workaround*** |
| Firmware can clear ANASW to 0 before entering EM4S to reduce current consumption. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.4 DBG_E204 – Debug Recovery with JTAG Does Not Work

| ***Description of Errata*** |
|---|
| The debug recovery algorithm of holding down pin reset, issuing a System Bus Stall AAP instruction, and releasing the reset pin does not work when using the JTAG debug interface. When using the JTAG debug interface, the core will continue to execute code as soon as the reset pin is released. |
| ***Affected Conditions / Impacts*** |
| The debug recovery sequence will not work when using the JTAG debug interface. |
| ***Workaround*** |
| Use the Serial Wire debug interface to implement the debug recovery sequence. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.5 EMU_E220 – DECBOD Reset During Voltage Scaling After EM2 or EM3 Wakeup

| ***Description of Errata*** |
|---|
| An infrequent, asynchronous and unrelated internal event can intermittently delay normal BOD state-machine transition sequencing during voltage scaling from VSCALE0 (1.0 Vdc) to VSCALE2 (1.2 Vdc) when emerging from EM2/EM3 to EM0. This delay can cause erroneous DECBOD resets on some devices. |
| ***Affected Conditions / Impacts*** |
| Systems operating with core voltage scaling can experience a decouple voltage brownout reset (DECBOD) when exiting EM2 or EM3. |
| ***Workaround*** |
| Systems that use core voltage scaling need to enter EM2 or EM3 via a RAM executed wait for interrupt instruction with interrupts disabled. Additionally, the EMU writes shown below should be added around EM2/EM3 entry and exit and after voltage scaling completes. This prevents the BOD state-machine transition signals from being delayed. This workaround adds 2.7 µs to the voltage scaling operation.<br>**Note:** This workaround is included in `em_emu.c` in the v2.7.4.0 or later of the Gecko SDK. It is recommended to workaround this issue by using the latest Gecko SDK version. |

```
// Execute from RAM with interrupts disabled
*(uint32_t *) (EMU_BASE + 0x1A4) |= 0x1f << 10;
__WFI();
*(uint32_t *) (EMU_BASE + 0x14C) |=  0x01 << 31;
// Enable Interrupts and return to flash execution

. . . .

// After voltage scaling is complete
*(uint32_t *) (EMU_BASE + 0x14C) &= ~(0x01 << 31);
EMU->IFC = 0xFFFFFFFF;
```

| ***Resolution*** |
|---|
| There is currently no resolution for this issue. |

### 2.6 I2C_E206 – Slave Holds SCL Low After Losing Arbitration

| ***Description of Errata*** |
|---|
| If, while transmitting data as a slave, arbitration is lost, SCL is unintentionally held low for an indefinite period of time. |
| ***Affected Conditions / Impacts*** |
| The winner of arbitration cannot use the bus because SCL is never released. |
| ***Workaround*** |
| If the $I^2C$ arbitration lost flag is asserted (I2C_IF_ARBLOST = 1) in slave mode (I2C_STATE_MASTER = 0), application software needs to wait for at least one SCL high time and then issue the transmission abort command (set I2C_CMD_ABORT = 1), thus releasing SCL. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.7 I2C_E207 – I²C Fails to Indicate New Incoming Data

| ***Description of Errata*** |
|---|
| A race condition exists in which the I²C fails to indicate reception of new data when both user software attempts to read data from and the I²C hardware attempts to write data to the I2C_RXFIFO in the same cycle. |
| ***Affected Conditions / Impacts*** |
| When this race condition occurs, the RXFIFO enters an invalid state in which both I2C_STATUS_RXDATAV = 0 and I2C_STATUS_RXFULL = 1. This causes the I²C to discard new incoming data bytes because RXFULL = 1 and would otherwise prevent user software from reading last byte written by the I²C hardware to RXFIFO because RXDATAV = 0. |
| ***Workaround*** |
| User software can recognize and clear this invalid RXDATAV = 0 and RXFULL = 1 condition by performing a dummy read of the RXFIFO (I2C_RXDATA). This restores the expected RXDATAV = 1 and RXFULL = 0 condition. The data from this read can be discarded, and user software can now read the last byte written by the I²C hardware to the RXFIFO (the byte which caused the invalid RXDATAV = 0 and RXFULL = 1 condition).<br><br>No data will be lost as long as user software completes this recovery procedure (performing the dummy read and then reading the remaining valid byte in the RXFIFO) before the I²C hardware receives the next incoming data byte. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.8 LES_E201 — LFPRESC Can Extend Channel Start-Up Delay

| ***Description of Errata*** |
|---|
| Setting LESENSE_TIMCTRL_LFPRESC to a value other than DIV1 may delay channel start-up longer than the number of $LFACLK_{LESENSE}$ clock cycles specified by LESENSE_TIMCTRL_STARTDLY. |
| ***Affected Conditions / Impacts*** |
| Delaying channel start-up delays the subsequent excitation and measurement phases and may have an impact on the data returned by the LESENSE. |
| ***Workaround*** |
| If a channel start-up delay is used (LESENSE_TIMCTRL_STARTDLY > 0), LESENSE_TIMCTRL_LFPRESC must be set to DIV1. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.9 RMU_E202 – External Debug Access Not Available After Watchdog or Lockup Full Reset

| ***Description of Errata*** |
|---|
| When a reset is triggered in full-reset mode, a debugger will not be able to read AHB-AP or ARM core registers. |
| ***Affected Conditions / Impacts*** |
| Systems using the full reset mode for watchdog or lockup resets will see limited debugging capability after one of these resets triggers. |
| ***Workaround*** |
| There are three possible workarounds:<br>• Software should configure peripherals to either LIMITED or EXTENDED mode if full debugger functionality is needed after a watchdog or lockup reset.<br>• When using FULL reset mode, appending at least 9 idle clock cycles to the last debug command will allow the transaction to complete.<br>• A power cycle or hard pin reset will restore normal operation. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.10 TIMER_E202 — Continuous Overflow and Underflow Interrupts in Quadrature Counting Mode

| ***Description of Errata*** |
|---|
| When the TIMER is configured to operate in quadrature decoder mode with the overflow interrupt enabled and the counter value (TIMER_CNT) reaches the top value (TIMER_TOP), the overflow interrupt is requested continuously even if the interrupt flag (TIMER_IF_OF) is cleared. Similarly, if the underflow interrupt is enabled and the counter value reaches zero, the underflow interrupt is requested continuously even if the interrupt flag (TIMER_IF_UF) is cleared. The interrupt can be cleared only after the counter value has incremented or decremented so that the overflow or underflow condition no longer applies. |
| ***Affected Conditions / Impacts*** |
| Because the counter is clocked by its CC0 and CC1 inputs in quadrature decoder mode and not the prescaled HFPERCLK, overflow and underflow events remain latched as long as TIMER_CNT remains at the value that triggered the overflow or underflow condition. Until the counter is no longer in the overflow or underflow condition, it is not possible to clear the associated interrupt flag. |
| ***Workaround*** |
| Short of disabling the relevant interrupts, the simplest workaround is to manually change TIMER_CNT so that the overflow or underflow condition no longer exists. Insert the following or similar code in the interrupt handler for the timer in question (TIMER0 in this case) to do this:<br>`uint32 intFlags = TIMER_IntGet(TIMER0);`<br><br>`if((intFlags & TIMER_IF_OF) && (TIMER0->CNT == TIMER0->TOP))`<br>`  TIMER0->CNT = 0;`<br><br>`if((intFlags & TIMER_IF_UF) && (TIMER0->CNT == 0x0))`<br>`  TIMER0->CNT = TIMER0->TOP;`<br><br>It may be necessary for firmware to account for this adjustment in calculations that include the counter value. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.11 USART_E204 — IrDA Modulation and Transmission of PRS Input Data

| ***Description of Errata*** |
|---|
| If the USART IrDA modulator is configured to accept input from a PRS channel, the incoming data stream will not be transmitted because the required clock from the baud rate generator is never enabled. |
| ***Affected Conditions / Impacts*** |
| It is not possible for the USART IrDA modulator to directly transmit data from a source other than the USART's own transmitter. The USART_IRCTRL_IRPRSEN bit should remain at its reset state of 0. |
| ***Workaround*** |
| Assuming the data to be sent via the PRS is also data that could be received by the EFM32/EFR32 USART, then the data can be received using the USART's PRS RX feature (USART_INPUT_RXPRS = 1), stored in RAM (e.g., using DMA), and then transmitted with IrDA mode enabled. In cases where IrDA operation is transmit-only, the PRS RX data can be received on the same USART doing the transmission. If IrDA operation is bidirectional, then another USART must be used to receive the PRS data.<br><br>If the data to be sent is in some other format (e.g., pulses from a timer output), then there is no direct way to transmit it using the IrDA modulator. It would be necessary to capture the data in some other way and reformat it as serial data timed according to the clock generated by the USART. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.12 USART_E205 — Possible Data Transmission on Wrong Edge in Synchronous Mode

| ***Description of Errata*** |
|---|
| If the USART is configured to operate in synchronous mode with...<br><br>1. USART_CLKDIV_DIV = 0 (clock = $f_{HFPERCLK} \div 2$)<br>2. USART_CTRL_CLKPHA = 0<br>3. USART_TIMING_CSHOLD = 1<br><br>...and data is loaded into the transmit FIFO (say, by the LDMA) at the exact same time as the USART state machine begins to insert the requested one bit time extension of chip select hold time (USART_TIMING_CSHOLD = 1), the first bit of the new data word is incorrectly transmitted on the leading clock edge of the subsequent data bit and not the trailing clock edge of the current data bit. |
| ***Affected Conditions / Impacts*** |
| Reception of each data bit by the slave is tied to a specific clock edge, thus the late transmission by the master of the first bit of a word may cause the slave to receive the incorrect data, especially if the data setup time for the slave approaches or exceeds one half the shift clock period. |
| ***Workaround*** |
| Because there is no way to specifically time a write to the transmit FIFO such that it does not occur when the USART state machine changes state, use one of the following workarounds to avoid the risk for data corruption described above:<br><br>• Set USART_CLK_DIV > 0.<br>• Use USART_TIMING_CSHOLD = 0 or USART_TIMING_CSHOLD > 1.<br>• Use USART_CTRL_CLKPHA = 1. This option is particularly useful with SPI flash memories as many support operations in both the CLKPOL = CLKPHA = 0 and CLKPOL = CLKPHA = 1 modes. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.13 USART_E206 — Additional SCLK Pulses Can Be Generated in USART Synchronous Mode

| ***Description of Errata*** |
|---|
| When inter-character spacing is enabled (USART_TIMING_ICS > 0) and USART_CTRL_CLKPHA = 1 in synchronous master mode, an extra clock pulse is generated after each frame transmitted except the last (that frame which when sent results in both the transmit FIFO and transmit shift register being empty). |
| ***Affected Conditions / Impacts*** |
| The extra clock pulse generated at the end of the first frame would cause a slave device to clock in the first bit of the next frame it expects to receive even though the USART is not yet driving that data. The slave would lose synchronization with the master and erroneously receive all frames after the first. |
| ***Workaround*** |
| Do not enable inter-character spacing when CLKPHA = 1. If a delay between frames is necessary, insert one manually with a software delay loop. Data cannot be transmitted using DMA in this case. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.14 WDOG_E201 – Clear Command is Lost Upon EM2 Entry

| ***Description of Errata*** |
|---|
| If the device enters EM2 while the clear command is still being synchronized, the watchdog counter may not be cleared as expected. |
| ***Affected Conditions / Impacts*** |
| If the watchdog counter is not cleared as expected, the device can encounter a watchdog reset. |
| ***Workaround*** |
| Wait for WDOG_SYNCBUSY_CMD to clear before entering EM2.<br><br>Note that WDOG can be clocked from one of the low-frequency clock sources and will require additional time to enter EM2 when implementing this workaround. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

### 2.15 WTIMER_E201 — Continuous Overflow and Underflow Interrupts in Quadrature Counting Mode

| ***Description of Errata*** |
|---|
| When the WTIMER is configured to operate in quadrature decoder mode with the overflow interrupt enabled and the counter value (WTIMER_CNT) reaches the top value (WTIMER_TOP), the overflow interrupt is requested contiunously even if the interrupt flag (WTIMER_IF_OF) is cleared. Similarly, if the underflow interrupt is enabled and the counter value reaches zero, the underflow interrupt is requested contiunously even if the interrupt flag (WTIMER_IF_UF) is cleared. Only after the counter value has incremented or decremented so that the overflow or underflow condition no longer applies can the interrupt be cleared. |
| ***Affected Conditions / Impacts*** |
| Because the counter is clocked by its CC0 and CC1 inputs in quadrature decoder mode and not the prescaled HFPERCLK, overflow and underflow events remain latched as long WTIMER_CNT remains at the value that triggered the overflow or underflow condition. Until the counter is no longer in the overflow or underflow condition, it is not possible to clear the associated interrupt flag. |
| ***Workaround*** |
| Short of disabling the relevant interrupts, the simplest workaround is to manually change WTIMER_CNT so that the overflow or underflow condition no longer exists. Insert the following or similar code in the interrupt handler for the timer in question (WTIMER0 in this case) to do this:<br>`uint32 intFlags = TIMER_IntGet(WTIMER0);`<br><br>`if((intFlags & WTIMER_IF_OF) && (WTIMER0->CNT == WTIMER0->TOP))`<br>`  WTIMER0->CNT = 0;`<br><br>`if((intFlags & WTIMER_IF_UF) && (WTIMER0->CNT == 0x0))`<br>`  WTIMER0->CNT = WTIMER0->TOP;`<br><br>It may be necessary for firmware to account for this adjustment in calculations that include the counter value. |
| ***Resolution*** |
| There is currently no resolution for this issue. |

# 3. Resolved Errata Descriptions

This section contains previous errata for EFM32TG11 devices.

For errata on the latest revision, refer to the beginning of this document. The device data sheet explains how to identify chip revision, either from package marking or electronically.

### 3.1 ADC_E224 – ADC Warm-Up Ready Can Cause IDAC, ACMP, or CSEN to Not Function

| ***Description of Errata*** |
|---|
| The IDAC, ACMP, or CSEN modules use the warm-up timing module in the ADC to determine when the peripherals are ready for use. However, if the ADC is enabled first, this timing module can fail to properly handshake with a low probability, causing the IDAC, ACMP, or CSEN modules to never finish warming up. The ADC is not affected by this issue and will always be available after it is enabled. |
| ***Affected Conditions / Impacts*** |
| Systems using the IDAC, ACMP, or CSEN modules in conjunction with the ADC can see intermittent failures where these modules do not operate. |
| ***Workaround*** |
| To work around this issue, enable the IDAC, ACMP, or CSEN modules before enabling the ADC. This will ensure the handshaking logic between the ADC and other modules functions correctly. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.2 ADC_E225 – Using the ADC in High-Accuracy Bias Mode Will Force All Analog Peripherals to High-Accuracy Bias Mode

| ***Description of Errata*** |
|---|
| Using the ADC in high-accuracy bias mode (GPBIASACC in ADCn_BIASPROG cleared to 0) forces all other analog peripherals into high-accuracy bias mode. These peripherals will then draw additional current.<br><br>The data sheet current consumption numbers are current specified with this additional current consumption included. When the updated devices are available, the device data sheet will be updated with the reduced current consumption specifications. |
| ***Affected Conditions / Impacts*** |
| Systems using the ADC in high-accuracy bias mode may see higher current consumption than expected when other analog peripherals not using high-accuracy bias mode are in use. |
| ***Workaround*** |
| There is currently no workaround for this issue. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.3 ADC_E228 – Limited ADC Sampling Frequency in EM2

| ***Description of Errata*** |
|---|
| ADC FIFO overflows occur at frequencies that are much lower than the ADC's maximum theoretical sampling rate. |
| ***Affected Conditions / Impacts*** |
| ADC sampling frequency is reduced in EM2. |
| ***Workaround*** |
| There is currently no workaround for this issue. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.4 BU_E201 — Extra Current from BUVDD to VREGVDD in BUMODE when Pulling Main Supply Low

| ***Description of Errata*** |
|---|
| While in Backup mode, pulling AVDD/IOVDD/VREGVDD/DVDD to 0 V and keeping BUVDD constant causes extra current from BUVDD to VREGVDD. This current starts occurring when AVDD = 0.6 V and is worst when AVDD = 0 V (~250 µA) and depends on the load on the VREGVDD pin (including the device itself). |
| ***Affected Conditions / Impacts*** |
| In most cases, even when replacing a battery, the supply should not drop below 0.6 V. If the supply does drop below this threshold, there will be some additional current draw depending on the load on the VREGVDD pin. |
| ***Workaround*** |
| There is currently no workaround for this issue. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.5 BU_E202 — Extra Current from DVDD to GND in BUMODE when Pulling BUVDD Low

| ***Description of Errata*** |
|---|
| While in Backup mode, pulling BUVDD low (but still above the BOD threshold) and keeping the main supply (AVDD/IOVDD/VREGVDD/DVDD) constant causes extra current from DVDD to GND. For example, with DVDD = 3.79 V, and BUVDD = 2 V, there is a current draw of ~128 µA current from DVDD to ground. |
| ***Affected Conditions / Impacts*** |
| Systems that enter Backup mode may see some additional current draw from DVDD to GND if BUVDD drops. |
| ***Workaround*** |
| For systems that do not use the DC-DC converter, there is currently no workaround for this issue.<br><br>For systems that do use the DC-DC converter, configure the chip to drive DVDD with the DC-DC converter (either BYPASS or regulation mode). When the device enters Backup mode, the DC-DC converter is turned off, leaving DVDD floating and removing the current leakage from DVDD to ground. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.6 CMU_E203 – Peak Detector May Not Trip

| ***Description of Errata*** |
|---|
| When PEAKDETTHR in HFXOCTRL1 is set to 4 or higher, the peak detector may not trip when the oscillating amplitude is higher than the target, which will cause calibration failure. |
| ***Affected Conditions / Impacts*** |
| If the device is not calibrated correctly (See CMU_E204), IBTRIMXOCORE will end up at a higher value than expected. Consequently, the oscillating amplitude and current consumption will also be higher than expected. |
| ***Workaround*** |
| PEAKDETTHR should be configured to be less than 4 to avoid potential peak detection failure. See CMU_E204 for guidance on how to program IBTRIMXOCORE. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.7 CMU_E204 – Initial Oscillator Calibration

***Description of Errata***

On power-up, a one-time calibration of oscillator amplitude is performed using the bias value specified by IBTRIMXOCORE in the HFXOSTEADYSTATECTRL register. The HFXO may not be able to start or settle at the target amplitude with the default value of IBTRIMXOCORE when crystal ESR is extraordinarily high.

***Affected Conditions / Impacts***

This typically manifests itself when performing a Pierce oscillator robustness test using a resistor that results in the HFXO seeing a series resistance of 3 or 5 times the maximum ESR specified for the crystal. The artificially inflated ESR may prevent the HFXO from starting or settling at the target amplitude with the default value of IBTRIMXOCORE.

***Workaround***

Three register fields can be modified to enhance oscillator start-up:

1. Increase the value of the IBTRIMXOCORE field in the HFXOSTEADYSTATECTRL register from the default value of 0x100 to 0x1ff.
2. Increase the value of the STEADYSTATETIMEOUT field in the HFXOTIMEOUTCTRL register from the default value of 0x04 to 0x08.
3. Decrease the value of the PEAKDETTHR field (bits [14:12]) in the previously undocumented HFXOCTRL1 register (at offset 0x28 in the CMU address space) from the default value of 0x04 to 0x02.

The following table summarizes the affected register fields on revision A and revision B devices:

**Table 3.1. CMU HFXO Register Field Defaults**

| Offset | Register | Bit Field | Default Value | |
|---|---|---|---|---|
| | | | Revision A | Revision B |
| 0x28 | HFXOCTRL1 | PEAKDETTHR | 0x4 | 0x2 |
| 0x34 | HFXOTIMEOUTCTRL | STEADYTIMEOUT | 0x4 | 0x8 |

Note that a user-specified value for IBTRIMXOCORE is going to be correlated with crystal frequency, ESR, and load capacitance, so its default value is not changing on revision B devices.

***Resolution***

This issue is resolved in revision B devices.

### 3.8 DBG_E205 – Incorrect Debug AP Base Address Register Value

| ***Description of Errata*** |
|---|
| According to the *ARM Debug Interface v5 Architecture Specification ADIv5.0 to ADIv5.2*, the Memory Access Port (MEM-AP) provides memory-mapped access to debug registers that are part of the complete Debug Access Port (DAP). One of the standard MEM-AP registers is the Debug Base Address register (BASE), which points either to a set of debug registers or a ROM table that, using a standardized format specified by ARM, describes the connected debug components.<br><br>EFM32TG11 contains a ROM table at address 0xF00FF000 that describes the connected debug components. However, the MEM-AP BASE register, which should point to this table, incorrectly points to address 0xE00FF000. |
| ***Affected Conditions / Impacts*** |
| If an external debugger uses the prescribed method of querying the BASE register to locate the ROM table and then querying the ROM table to determine which resources are present, it will not properly enumerate the full debug capabilities of the chip.<br><br>For example, EFM32TG11 contains the ARM CoreSight™ MTB-M0+, which provides simple execution tracing for devices with the Cortex-M0+ CPU. If the ROM table query is performed at address 0xE00FF000 as indicated by the BASE register, an external debugger will fail to determine that the MTB-M0+ is present. |
| ***Workaround*** |
| Debugger software targeting EFM32TG11 can properly locate all defined resources by explicitly querying the ROM table at address 0xF00FF000 using the defined procedure. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.9 EMU_E214 – Device Erase Cannot Occur if Voltage Scaling Level is Too Low

| ***Description of Errata*** |
|---|
| The device erase logic does not check the Voltage Scale Level prior to attempting a device erase. If using Voltage Scale Level 0 (1 V), the device may not be able to erase the flash. This results in a potentially unlockable device if operating at Voltage Scale Level 0 (1 V). |
| ***Affected Conditions / Impacts*** |
| It is possible that the flash is only partially erased when performing the operation at Voltage Scale Level 0 (1 V). If this results in the debug lock bit not clearing, a locked part doesn't unlock after the partial erasure (which it is intended to do), and the part remains locked. If subsequent erasures continue to fail, the part would remained locked. |
| ***Workaround*** |
| The voltage should be set to Voltage Scale Level 2 (1.2 V) before executing the device erase.<br><br>For systems that don't lock the debug interface, the user can follow the debug recovery procedure to halt the CPU before it has a chance to execute code in software to avoid the code scaling the voltage. The device erase can then be executed at Voltage Scale Level 2 (1.2 V) (the power-on default voltage of the device).<br><br>For systems that do lock the debug interface, firmware can implement a mechanism whereby it can voltage scale or unlock debug access if its defined authentication method is passed. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.10 I2C_E202 – Race Condition Between Start Detection and Timeout

| *Description of Errata* |
| --- |
| There is a race condition where the Bus Idle Timeout counter may clear the busy status of the I2C bus after a start condition. |
| ***Affected Conditions / Impacts*** |
| Software may attempt another I2C start if it thinks the bus is idle. This may disrupt the I2C bus. After the Bus Idle Timeout feature has triggered, it will not detect another idle condition. |
| ***Workaround*** |
| Software can wait for any of the following conditions before starting an I2C transaction:<br>• The received address match interrupt indicates that the I2C bus is busy. Software should serve this transaction and proceed accordingly. Software can ignore the wrong busy status.<br>• The SSTOPIF interrupt flag indicates that the I2C bus has returned to the idle state.<br>• A defined, system-dependent amount of time to wait after bus activity to ensure that the bus is in idle state. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.11 I2C_E203 – I2C Received Data Can be Shifted

| *Description of Errata* |
| --- |
| If SDA falls between detection of the start condition and the first rising edge of SCL, the I2C state machine clears the start condition that was just detected, causing the state machine counter to count the rising edge of SCL earlier than it was detected. This causes the received data to be out of sync and the acknowledge phase to occur one SCL clock cycle earlier than expected, thus corrupting the integrity of the I2C bus.<br><br>There are two ways in which the falling condition on SDA can potentially happen:<br>• In multi-master systems, one master initiates a start condition and then drives SDA high shortly before another master drives SDA low to indicate a start condition.<br>• In a single master system, if SDA is high from the last bit of the previous transaction, the master initiates a start condition and then drives SDA low because the MSB of the new address is low. |
| ***Affected Conditions / Impacts*** |
| I2C operation in slave mode or multi-master mode. |
| ***Workaround*** |
| This depends on whether the system is multi- or single-master. There is no workaround for multi-master cases. In a single-master system, the state of SDA may not change unless a new address is being sent, such that the falling condition on SDA would not be observed. Whether or not this is the case is dependent on the implementation of the particular I2C master. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.12 I2C_E205 – Go Idle Bus Idle Timeout Does Not Bring Device to Idle State

| ***Description of Errata*** |
|---|
| When the I2C is operating as a slave, if the bus idle timeout is active (`I2Cn_CTRL_BITO != 0`) and the go idle on bus timeout feature is enabled (`I2Cn_CTRL_GIBITO = 1`), the bus idle interrupt flag (`I2Cn_IF_BITO`) sets upon timeout, but the receiver does not enter the idle state. |
| ***Affected Conditions / Impacts*** |
| The I2C receiver needs to detect a START condition to recover from the bus idle timeout state. If there is other, undefined activity on the bus after the timeout, the receiver will not recover as expected. |
| ***Workaround*** |
| The `I2Cn_CTRL_EN` bit can be toggled from 1 to 0 and back to 1 again to resume normal operation. Alternatively, a START condition issued by any other master on the bus (including the EFM32/EFR32 device) will reset the receiver and return it to normal operation. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.13 LCD_E201 — LCD Boost Mode Does Not Work

| ***Description of Errata*** |
|---|
| It is not recommended to use the LCD boost mode on affected revisions of the device. |
| ***Affected Conditions / Impacts*** |
| Systems should not use the LCD boost mode feature. |
| ***Workaround*** |
| There is currently no workaround for this issue. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

### 3.14 RMU_E203 – AVDD Ramp Issue

***Description of Errata***

The device may not properly start during power-on or restart when a voltage droop (brown out) occurs on AVDD. The failure is intermittent.

For example configurations and waveforms that are more likely to result in this issue, see the following Knowledge Base article:

http://community.silabs.com/t5/32-bit-MCU-Knowledge-Base/RMU-E203-AVDD-Ramp-Issue/ta-p/197340

To detect this failure state, place a GPIO toggle at the beginning of `main()` in the device firmware. When this failure occurs, the pin will not be toggling as expected, as the device is not executing any code.

***Affected Conditions / Impacts***

Systems may intermittently see the device fail to start, reset, or respond. The current draw of the device in this state is ~100 µA and DECOUPLE will be fully powered (~1.2 V). The device will not execute any code in this state.

***Workaround***

This issue can be resolved with a hardware workaround where an external circuit holds the reset pin low during power-on or brown out until AVDD reaches 1.8 V. For brown out, the reset pin must be configured to hard reset mode. This can be accomplished as part of the firmware image programmed to the device (lock bits area) or using the following code:

```
// Clears the CLW0 bit to enable Hard reset
void enable_hardreset()
{
  uint32_t value;
  uint32_t newvalue;
  value = *(uint32_t *)0xFE041E8;
  newvalue = value & ~(1 << 2);
  MSC_WriteWord((uint32_t *)0xFE041E8, &newvalue, 4);
}
```

There is currently no software workaround for all potential failure mechanisms. The software workaround included in the Knowledge Base article will prevent failure in some scenarios. See the Knowledge Base article for more information:

http://community.silabs.com/t5/32-bit-MCU-Knowledge-Base/RMU-E203-AVDD-Ramp-Issue/ta-p/197340

***Resolution***

This issue is resolved in revision B devices.

### 3.15 TRNG_E201 — Possible Triggering of Noise Alarm

| ***Description of Errata*** |
|---|
| Operation of the TRNG with CMU_HFPERPRESC register PRESC field set to 0, may cause a higher than expected frequency of AIS31 preliminary alarms and noise alarms.<br><br>For some devices at extreme process variations, or at extremely cold temperature, the pre-conditioning entropy is not high enough to always pass the AIS31 start-up test or AIS31 online health tests. |
| ***Affected Conditions / Impacts*** |
| Systems using the TRNG may see a higher than expected frequency of AIS31 preliminary alarms and noise alarms.<br><br>Because of issues with the AIS31 test, the TRNG is not strictly AIS31 compliant at this time.<br><br>Any postconditioning data collected with no error flags will have a very high entropy. |
| ***Workaround*** |
| • When using TRNG, always set CMU_HFPERPRESC_PRESC to 1 or greater.<br>• Disable the start-up AIS31 tests, by setting TRNG_CONTROL register TRNG_CONTROL_BYPASSAIS31 bit.<br>• If more than 64 samples are required, the code should attempt to keep the FIFO from filling.<br>• The data should be discarded if any of the online health flag bits are set (ALMIF, PREIF, APT4096IF, APT64IF, or REPCOUNTIF).<br>• If the ALMIF bit is set, the TRNG should be reset by setting and then clearing the SOFTRESET bit in the TRNG_CONTROL register. |
| ***Resolution*** |
| This errata is deprecated by TRNG_E202. |

### 3.16 TRNG_E202 — Standards Non-compliance

| ***Description of Errata*** |
|---|
| The TRNG module may either fail to generate random numbers or generate random numbers with AIS-31 error flags. This is because the TRNG has two potential issues:<br>• Non-Compliance with NIST SP800-90B<br>The TRNG entropy source may not be sufficient to pass the start-up tests and will not place any data in the FIFO.<br>• Non-Compliance with AIS-31<br>The TRNG entropy source may be sufficient to pass the start-up tests, but insufficient to pass the AIS-31 test. It will place some data in the FIFO and indicate an AIS-31 error.<br><br>In both cases, the TRNG will cause the mbed TLS random number generator to return an error code and no data.<br><br>The TRNG module, therefore, is non-functional and should not be used. |
| ***Affected Conditions / Impacts*** |
| Application software that uses the mbed TLS random number generator may return no data either with or without an error code. Software that accesses the TRNG module directly by using the public CMSIS registers will either receive no data or data with AIS-31 error flags. |
| ***Workaround*** |
| There is no workaround that is NIST SP800-90B or AIS-31 compliant. |
| ***Resolution*** |
| This issue is resolved in revision B devices. |

# 4. Revision History

### Revision 0.5

September, 2020

- Added I2C_E207, USART_E206 and WDOG_E201.

### Revision 0.4

April 2020

- Added EMU_E220, LES_E201, TIMER_E202, USART_E205, and WTIMER_E201.
- Migrated to new errata document format.

### Revision 0.3

November, 2018

- Updated for device revision B.
- ADC_E224, ADC_E225, ADC_E228, BU_E201, BU_E202, CMU_E203, CMU_E204, DBG_E205, EMU_E214, I2C_E202, I2C_E203, I2C_E205, LCD_E201, RMU_E203, and TRNG_E202 resolved and moved to .
- Previously deprecated TRNG_E201 moved to because TRNG_E202 replaced TRNG_E201 and is resolved.
- Added CSEN_E201, CSEN_E202, DBG_E205, I2C_E202, I2C_E203, I2C_E205, I2C_E206, and USART_E204.
- USART_E201 has never been present on EFM32TG11 and has been removed.

### Revision 0.2

May, 2018

- Resolution status for multiple errata changed to "Fix Planned."
- Updated the workaround in RMU_E202.
- Deprecated TRNG_E201.
- Added TRNG_E202.

### Revision 0.1

February, 2018

- Initial release.

# Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

**IoT Portfolio**
www.silabs.com/IoT

**SW/HW**
www.silabs.com/simplicity

**Quality**
www.silabs.com/quality

**Support & Community**
www.silabs.com/community



SILICON LABS


Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

**http://www.silabs.com**