Wireless Gecko Multi-Protocol Module MGM240P Errata

This document contains information on the MGM240P errata. The latest available revision of this device is revision V3.

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

Errata effective date: December, 2022.

1. Errata Summary

The table below lists all known errata for the MGM240P and all unresolved errata of the MGM240P.

Table 1.1. Errata Overview

Designator	Title/Problem	Workaround Exists	Exists on Revision:		
			V1	V2	V3
CUR_E302	Extra EM1 Current if FPU is Disabled	Yes	X	X	X
CUR_E303	Active Charge Pump Clock Causes High Current	Yes	X	—	—
DCDC_E302	DCDC Interrupts Block EM2/3 Entry or Cause Unexpected Wake-up	Yes	X	—	—
EMU_E304	Higher Than Expected EM2 Current	No	X	—	—
EUSART_E302	Synchronous EUSART Module Disable Lockup	Yes	X	X	X
EUSART_E303	EUSART Receiver Enters Lockup State when Using Low Frequency IrDA Mode	Yes	X	X	X
EUSART_E304	Incorrect Stop Bits Lock Receiver	Yes	X	X	X
IADC_E306	Changing Gain During a Scan Sequence Causes an Erroneous IADC Result	Yes	X	X	X
KEYSCAN_E301	Unused Rows Are Not Properly Gated Off	Yes	X	X	X
RADIO_E304	Zigbee Signal Identifier False Detection	No	X	—	—
RADIO_E305	Channel Clear Detection	No	X	—	—
RADIO_E307	BLE 2 Mbps and IEEE 802.15.4 Sensitivity and Selectivity Degradation with Crystals Below 39 MHz	Yes	X	X	X
SE_E301	Bricked Device After SE Firmware Upgrade or Bootloader Upgrade	No	X	X	—
USART_E304	PRS Transmit Unavailable in Synchronous Secondary Mode	No	X	X	X

2. Current Errata Descriptions

2.1 CUR_E302 – Extra EM1 Current if FPU is Disabled

Description of Errata
When the Floating Point Unit (FPU) is disabled, the on-demand Fast Startup RC Oscillator (FSRCO) remains on after an energy mode transition from EM0 to EM1 is complete. This leads to higher current consumption in EM1.
Affected Conditions / Impacts
The enabled FSRCO increases EM1 current consumption by approximately 500 µA.
Workaround
Always enable the FPU at the beginning of code execution via the Coprocessor Access Control Register (CPACR) in the System Control Block (SCB) as shown below: `SCB->CPACR |= ((3 << 20) | (3 << 22));`
Resolution
There is currently no resolution for this issue.

2.2 EUSART_E302 — Synchronous EUSART Module Disable Lockup

Description of Errata
The EUSART freezes and does not function if firmware: 1. Initializes the EUSART in synchronous main mode. 2. Disables the EUSART and reconfigures it to either synchronous secondary or asynchronous mode. 3. Re-enables the ESUART. 4. Transfers data. 5. Disables the EUSART. This issue is caused by the failure of a handshake signal that is required by the EUSART disabling logic to fully propagate when leaving synchronous main mode.
Affected Conditions / Impacts
Systems that use the EUSART in synchronous main mode cannot simply switch to another mode as this causes the module to freeze. This ocurs only when firmware attempts to switch from synchronous main mode to another mode. Switching between all other modes is unaffected.
Workaround

Firmware can manually generate additional clock edges after the module is disabled to fully propagate the handshake signal and allow the next disable sequence to happen as usual.
Example code
`//Work-around code//`
`uint32_t i;`
`for (i=0;i<4;i++) {`
`EUSART0->CFG2 |= EUSART_CFG2_CLKPHA;`
`EUSART0->CFG2 &= ~EUSART_CFG2_CLKPHA;`
`}`
`//Work-around code - END//`

Resolution
There is currently no resolution for this issue.

2.3 EUSART_E303 — EUSART Receiver Enters Lockup State when Using Low Frequency IrDA Mode

Description of Errata
When low frequency IrDA mode is enabled (EUSART_IRLFCFG_IRLFEN = 1), the receiver can block incoming traffic if it receives either a… • 0 if EUSART_CFG0_RXINV = 0 or • 1 if EUSART_CFG0_RXINV = 1 …before… • the EUSART module is enabled (EUSART_EN_EN =1), • the receiver is enabled (EUSART_CMD_RXEN =1), and • the write to enable the receiver (RXEN = 1) has been synchronized (EUSART_SYNCBUSY_RXEN = 0).
Affected Conditions / Impacts
Incoming traffic will be blocked at the EUSART receiver and subsequent interrupts and status flags will not be set correctly.
Workaround

To avoid entering the lockup state, use one of the workarounds mentioned below:

- When the receiver (RX) input is routed through the PRS:

  Force the input to the IrDA demodulator to high by using the PRS before enabling EUSART. Keep it this way until the receiver has been enabled and EUSART_CMD_RXEN bit is synchronized. See the following code sequence for an example of how to do this:

```
// Output logic 0 through PRS Channel that is connected to EUSART RX GPIO
PRS->ASYNC_CH[0].CTRL = PRS_ASYNC_CH_CTRL_FNSEL_LOGICAL_ZERO |
                    PRS_ASYNC_CH_CTRL_SOURCESEL_GPIO | PRS_ASYNC_CH_CTRL_SIGSEL_GPIOPIN0;

// Select PRS as input to RX.
EUSART0->CFG1_SET = EUSART_CFG1_RXPRSEN;

// Enable EUSART to configure Rx
EUSART0->EN_SET = EUSART_EN_EN;

// Enable Rx
EUSART0->CMD = EUSART_CMD_RXEN;

// Wait until Rx enable is synchronized
while ((EUSART0->SYNCBUSY & EUSART_SYNCBUSY_RXEN) != 0U) {}

// Output EUSART RX pin through PRS Channel
PRS->ASYNC_CH[0].CTRL = (PRS->ASYNC_CH[0].CTRL & ~_PRS_ASYNC_CH_CTRL_FNSEL_MASK) |
                                    PRS_ASYNC_CH_CTRL_FNSEL_A;
```

  Note: EUSART_CTRL_RXINV = 1 in this workaround because the receiver input must be inverted for proper IrDA RZI operation.

- When the receiver (RX) input is not routed through the PRS:

  Force the input to the IrDA demodulator to high by using a GPIO pin other than the current EUSART RX pin before enabling the EUSART. Keep it this way until the receiver has been enabled and EUSART_CMD_RXEN bit is synchronized. See the following code sequence for an example of how to do this:

```
// Configure alternate GPIO (PA00) used for workaround to output 0
GPIO_PinModeSet(gpioPortA, 0, gpioModePushPull, 0);

// Route EUSART0 Rx to the alternate GPIO (PA00)
GPIO->EUSARTROUTE[0].ROUTEEN = (GPIO->EUSARTROUTE[0].ROUTEEN & ~GPIO_EUSART_ROUTEEN_RXPEN);
GPIO->EUSARTROUTE[0].RXROUTE = (gpioPortA << _GPIO_EUSART_RXROUTE_PORT_SHIFT) | (0 <<
_GPIO_EUSART_RXROUTE_PIN_SHIFT);
GPIO->EUSARTROUTE[0].ROUTEEN |= GPIO_EUSART_ROUTEEN_RXPEN;

// Enable EUSART0 to configure Rx
EUSART0->EN_SET = EUSART_EN_EN;

// Enable Rx
EUSART0->CMD = EUSART_CMD_RXEN;

// Wait until Rx enable is synchronized
while ((EUSART0->SYNCBUSY & EUSART_SYNCBUSY_RXEN) != 0U) {}

// Route EUSART Rx to EUSART_RX GPIO(EUSART_RX_PORT & EUSART_RX_PIN)
GPIO->EUSARTROUTE[0].ROUTEEN = (GPIO->EUSARTROUTE[0].ROUTEEN & ~GPIO_EUSART_ROUTEEN_RXPEN);
GPIO->EUSARTROUTE[0].RXROUTE = (EUSART_RX_PORT << _GPIO_EUSART_RXROUTE_PORT_SHIFT) | (EUSART_RX_PORT <<
_GPIO_EUSART_RXROUTE_PIN_SHIFT);
GPIO->EUSARTROUTE[0].ROUTEEN |= GPIO_EUSART_ROUTEEN_RXPEN;

// Disable alternate GPIO (PA00) used for workaround
GPIO_PinModeSet(gpioPortA, 0, gpioModeDisabled, 0);
```

  Note: EUSART_CTRL_RXINV = 1 in this workaround because the receiver input must be inverted for proper IrDA RZI operation.

To exit the lockup state, disable the EUART and force the input to the IrDA demodulator to 1 before re-enabling the EUART by using steps mentioned above.

Resolution

There is currently no resolution for this issue.

2.4 EUSART_E304 — Incorrect Stop Bits Lock Receiver

Description of Errata
When low frequency IrDA mode is enabled (EUSART_IRLFCFG_IRLFEN = 1), the receiver can block incoming traffic if it receives either a… • 0 if EUSART_CFG0_RXINV = 0 or • 1 if EUSART_CFG0_RXINV = 1 …when it is expecting a stop bit.
Affected Conditions / Impacts
Incoming traffic will be blocked at the EUSART receiver. Subsequent interrupts and status flags will not be set correctly.
Workaround

To avoid receiver lock-up in the application firmware caused by formatting errors in the received data, change the receiver GPIO pin routing to force the input to the IrDA demodulator to 1 for the anticipated period of time during which such data can be received.

To exit the lockup state, disable the EUSART and force the input to the IrDA demodulator to 1 before re-enabling the EUSART by using one of the workarounds mentioned below:

- When the receiver (RX) input is routed through the PRS:

  Force the input to the IrDA demodulator to high by using the PRS before enabling EUSART. Keep it this way until the receiver has been enabled and EUSART_CMD_RXEN bit is synchronized. See the following code sequence for an example of how to do this:

```
// Output logic 0 through PRS Channel that is connected to EUSART RX GPIO
PRS->ASYNC_CH[0].CTRL = PRS_ASYNC_CH_CTRL_FNSEL_LOGICAL_ZERO |
                        PRS_ASYNC_CH_CTRL_SOURCESEL_GPIO | PRS_ASYNC_CH_CTRL_SIGSEL_GPIOPIN0;

// Select PRS as input to Rx
EUSART0->CFG1_SET = EUSART_CFG1_RXPRSEN;

// Enable EUSART to configure Rx
EUSART0->EN_SET = EUSART_EN_EN;

// Enable Rx
EUSART0->CMD = EUSART_CMD_RXEN;

// Wait until Rx enable is synchronized
while ((EUSART0->SYNCBUSY & EUSART_SYNCBUSY_RXEN) != 0U) {}

// Output EUSART RX through PRS Channel
PRS->ASYNC_CH[0].CTRL = (PRS->ASYNC_CH[0].CTRL & ~_PRS_ASYNC_CH_CTRL_FNSEL_MASK) |
                        PRS_ASYNC_CH_CTRL_FNSEL_A;
```

  Note: EUSART_CTRL_RXINV = 1 in this workaround because the receiver input must be inverted for proper IrDA RZI operation.

- When the receiver (RX) input is not routed through the PRS:
  Force the input to the IrDA demodulator to high by using a GPIO pin other than the current EUSART RX pin before enabling the EUSART. Keep it this way until the receiver has been enabled and EUSART_CMD_RXEN bit is synchronized. See the following code sequence for an example of how to do this:

```
// Configure alternate GPIO (PA00) used for workaround to output 0
GPIO_PinModeSet(gpioPortA, 0, gpioModePushPull, 0);

// Route EUSART0 Rx to the alternate GPIO (PA00)
GPIO->EUSARTROUTE[0].RXROUTE = (gpioPortA << _GPIO_EUSART_RXROUTE_PORT_SHIFT) | (0 <<
_GPIO_EUSART_RXROUTE_PIN_SHIFT);

// Enable EUSART0 to configure Rx
EUSART0->EN_SET = EUSART_EN_EN;

// Enable Rx
EUSART0->CMD = EUSART_CMD_RXEN;

// Wait until Rx enable is synchronized
while ((EUSART0->SYNCBUSY & EUSART_SYNCBUSY_RXEN) != 0U) {}

// Route EUSART Rx to EUSART_RX GPIO(EUSRT_RX_PORT & EUSART_RX_PIN)
GPIO->EUSARTROUTE[0].RXROUTE = (EUSART_RX_PORT << _GPIO_EUSART_RXROUTE_PORT_SHIFT) | (EUSART_RX_PIN <<
_GPIO_EUSART_RXROUTE_PIN_SHIFT);

// Disable alternate GPIO (PA00) used for workaround
GPIO_PinModeSet(gpioPortA, 0, gpioModeDisabled, 0);
```

  Note: EUSART_CTRL_RXINV = 1 in this workaround because the receiver input must be inverted for proper IrDA RZI operation.

Resolution

There is currently no resolution for this issue.

2.5 IADC_E306 – Changing Gain During a Scan Sequence Causes an Erroneous IADC Result

Description of Errata
Differences in the ANALOGGAIN setting within multiple IADC_CFGx groups during a scan sequence introduces a transient condition that may result in an inaccurate IADC conversion.
Affected Conditions / Impacts
The result of the IADC scan measurement may not match the expected result for the voltage present on the pin during the conversion.
Workaround
Both 1 and 2 shown below must be implemented. 1. If there is a difference in the ANALOGGAIN setting between IADC_CFGx groups during a scan sequence, the IADC_SCHEDx clock prescaler must also change to an appropriate setting. This forces a warmup state (5 µs delay) in between ANALOGGAIN changes. Note that the same IADC_SCHEDx clock prescaler value may be an appropriate setting for both ANALOGGAIN settings, but to force the warmup delay, the IADC_SCHEDx must have different values. 2. The first and last entry of a scan group should use IADC_CFG0, which is the default configuration of the IADC at the start and end of a scan conversion sequence. If CONFIG1 is used at the start and end of the scan group, erronous IADC results may occur.
Resolution
There is currently no resolution for this issue.

2.6 KEYSCAN_E301 – Unused Rows Are Not Properly Gated Off

Description of Errata
Unused KEYSCAN row inputs cause the KEY bit in the KEYSCAN_IF register to be set at all times indicating a key was pressed. This prevents the interrupt flag from clearing and stops the scan procedure.
Affected Conditions / Impacts
The KEY bit in the KEYSCAN_IF register is always set when rows are left unused.
Workaround

Configure the GPIO_KEYSCAN_ROWSENSEnROUTE registers for any unused row inputs to the same GPIO port and pin associated with any of the row inputs that are used. For example, if rows 0, 1, and 2 are used and routed to PA05, PA06, and PA07 respectively, and rows 3, 4, and 5 are unused, the configuration could be:
`// Routing GPIO pins PA05, PA06 and PA07 to rows 0, 1 and 2`
`GPIO->DBUSKEYPAD_ROWSENSE0ROUTE = 0 << _GPIO_DBUSKEYPAD_ROWSENSE0ROUTE_PORT_SHIFT |`
`5 << _GPIO_DBUSKEYPAD_ROWSENSE0ROUTE_PIN_SHIFT;`
`GPIO->DBUSKEYPAD_ROWSENSE1ROUTE = 0 << _GPIO_DBUSKEYPAD_ROWSENSE1ROUTE_PORT_SHIFT|`
`6 << _GPIO_DBUSKEYPAD_ROWSENSE1ROUTE_PIN_SHIFT;`
`GPIO->DBUSKEYPAD_ROWSENSE2ROUTE = 0 << _GPIO_DBUSKEYPAD_ROWSENSE2ROUTE_PORT_SHIFT|`
`7 << _GPIO_DBUSKEYPAD_ROWSENSE2ROUTE_PIN_SHIFT;`
 	
`// Workaround - Connect unused rows 3, 4, and 5 to row 2 (PA07), a single used row`
`GPIO->KEYSCANROUTE.ROWSENSE3ROUTE = 0 << _GPIO_KEYSCAN_ROWSENSE3ROUTE_PORT_SHIFT |`
`7 << _GPIO_KEYSCAN_ROWSENSE3ROUTE_PIN_SHIFT;`
`GPIO->KEYSCANROUTE.ROWSENSE4ROUTE = 0 << _GPIO_KEYSCAN_ROWSENSE4ROUTE_PORT_SHIFT |`
`7 << _GPIO_KEYSCAN_ROWSENSE4ROUTE_PIN_SHIFT;`
`GPIO->KEYSCANROUTE.ROWSENSE5ROUTE = 0 << _GPIO_KEYSCAN_ROWSENSE5ROUTE_PORT_SHIFT |`
`7 << _GPIO_KEYSCAN_ROWSENSE5ROUTE_PIN_SHIFT;`
 	
Note that KEYSCAN_STATUS.ROW will report the same values for used and unused rows that route to the same GPIO. In the scenario above, KEYSCAN_STATUS.ROW bits 2, 3, 4, and 5 will show the same values. The unused row bits in the KEYSCAN_STATUS field should be masked so that unused row bits are set to 1, indicating a key is not pressed.

Resolution
There is currently no resolution for this issue.

2.7 RADIO_E307 – BLE 2 Mbps and IEEE 802.15.4 Sensitivity and Selectivity Degradation with Crystals Below 39 MHz

Description of Errata
Sensitivity and selectivity degradation using the BLE 2 Mbps or 802.15.4 PHYs when using crystals below 39 MHz.
Affected Conditions / Impacts
The BLE 2 Mbps PHY and 802.15.4 PHY will show sensitivity degradation of approximately 8 dB at higher frequencies, and 3 dB up to 37 dB selectivity degradation based on the channel, when using crystals below 39 MHz. For the 38 MHz crystal, the sensitivity degradation will be seen at 2432 MHz and above. For the 38.4 MHz crystal, the sensitivity degradation will be seen at 2458 MHz and above. This problem does not exist with 39 MHz and above crystals. The BLE 1 Mbps and LR PHYs are unaffected.
Workaround
There is currently no workaround for either a 38 MHz or 38.4 MHz crystal with releases of Gecko SDK prior to 4.1.0. Use of a 39 MHz or 40 MHz crystal avoids the sensitivity and selectivity degradation when using earlier SDK releases.
Resolution
This issue is resolved by upgrading to Gecko SDK 4.1.0.

2.8 USART_E304 — PRS Transmit Unavailable in Synchronous Secondary Mode

Description of Errata
When the USART is configured for synchronous secondary operation, the transmit output (MISO) is not driven if the signal is routed to a pin using the PRS producer (e.g., SOURCESEL = 0x20 and SIGSEL = 0x4 for USART0).
Affected Conditions / Impacts
Systems cannot operate the USART in synchronous secondary mode if the PRS is used to route the transmit output to the RX (MISO) pin. Operation is not affected in main mode when the transmit output is routed to the TX (MOSI) pin using the PRS producer nor is operation affected in any mode when the GPIO_USARTn_RXROUTE and GPIO_USARTn_TXROUTE registers are used.
Workaround
There is currently no workaround for this issue.
Resolution
There is currently no resolution for this issue.

3. Resolved Errata Descriptions

This section contains previous errata for MGM240P 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 CUR_E303 – Active Charge Pump Clock Causes High Current

Description of Errata
When the ACMP0, ACMP1, or IADC0 peripherals are active, the clock to the internal analog mux charge pump may also be activated, resulting in extra supply current.
Affected Conditions / Impacts
• ACMP0 and ACMP1: The charge pump clock is activated whenever either module is enabled via the ACMPn_EN_EN bit or when enabled by the LESENSE state machine. • IADC0: The charge pump clock is activated when any portion of the IADC analog circuitry is on. When IADC_CTRL_WARMUP-MODE = KEEPINSTANDBY or KEEPWARM, the clock is activated as long as the IADC is enabled via the IADC_EN_EN bit. When IADC_CTRL_WARMUPMODE = NORMAL, the clock is activated only during warmup and conversion and will be shut down be-tween conversions. • The extra current is from a shared block and increases supply current by an approximate total of 25 µA when any of the above conditions are true.
Workaround
No workaround exists to entirely eliminate the extra current. The impact of the current can be reduced by duty-cycling the peripheral. The average system supply current increase depends on the total percentage of time the peripheral(s) is/are active. For example, if only ACMP0 is used and enabled for 10% of the time, the average supply current increase is about 2.5 µA.
Resolution
This issue is resolved in revision V2 devices.

3.2 DCDC_E302 – DCDC Interrupts Block EM2/3 Entry or Cause Unexpected Wake-up

Description of Errata
Regardless of the setting of the DCDC Interrupt Enable (DCDC_IEN) register, if the DCDC interrupt is enabled in the NVIC, the BYPSW, WARM, RUNNING, or TMAX interrupt requests can wake the device from EM2/3 or prevent it from entering EM2/3.
Affected Conditions / Impacts
The errata is limited to the BYPSW, WARM, RUNNING, or TMAX requests as reflected in the DCDC Interrupt Flag (DCDC_IF) regis-ter, which also function as wake-up sources from EM2/3. When the NVIC DCDC interrupt is enabled: • If the corresponding DCDC_IEN bit for one of these interrupt requests is 1 and that condition occurs, then an interrupt **will** occur, and the CPU will branch to the DCDC IRQ handler. • If the corresponding DCDC_IEN bit for one of these interrupt requests is 0 and that condition occurs, then an interrupt **will not** occur. • If any one of these four interrupt conditions occurs, regardless of the setting of its corresponding DCDC_IEN bit, the device **will** wake from EM2/3 and/or be prevented from entering EM2/3. If the corresponding IEN is 0, an interrupt **will not** occur even though the EM2/3 wakeup event has occurred.
Workaround
To prevent unwanted wake-up from or blocked entry into EM2/3, disable the DCDC interrupt using `NVIC_DisableIRQ(DCDC_IRQn)` before entering EM2/3 and re-enable the DCDC interrupt using `NVIC_EnableIRQ(DCDC_IRQn)` after EM2/3 wake-up.
Resolution
This issue is resolved in revision V2 devices.

3.3 EMU_E304 – Higher Than Expected EM2 Current

Description of Errata
Current consumption in EM2 is higher than the datasheet specification.
Affected Conditions / Impacts
Systems operating in EM2 have higher than expected current consumption, regardless of whether the DCDC is enabled.
Workaround
There is currenty no workaround for this issue.
Resolution
This issue is resolved in revision V2 devices.

3.4 RADIO_E304 – Zigbee Signal Identifier False Detection

Description of Errata
The Zigbee signal identifier sometimes indicates a false detection when a BLE 1 Mbps or Continuous Wave (CW) signal is present.
Affected Conditions / Impacts
Systems that are exposed to BLE 1 Mbps or Continuous Wave (CW) signals sometimes falsely indicate the presence of non-existent Zigbee signals.
Workaround
There is currenty no workaround for this issue
Resolution
This issue is resolved in revision V2 devices.

3.5 RADIO_E305 – Channel Clear Detection

Description of Errata
The Listen Before Talk (LBT) and Carrier Sense Multiple Access-Collision Avoidance (CSMA-CA) algorithms always indicate that the channel is clear, even when this is not the case.
Affected Conditions / Impacts
The LBT and CSMA-CA algorithms cannot be used.
Workaround
There is currently no workaround for this issue.
Resolution
This issue is resolved in revision V2 devices.

3.6 SE_E301 – Bricked Device After SE Firmware Upgrade or Bootloader Upgrade

Description of Errata
Devices with a date code less than 2216 may become permanently disabled during an over-the-air (OTA) or over-the-wire (OTW) upgrade of the SE firmware or the bootloader in certain configurations.
Affected Conditions / Impacts
Devices with a date code before 2216 and SE firmware prior to version 2.1.8 will fail to upgrade properly and become permanently disabled at step #5 of the upgrade procedure (illustrated in Figures 4.1 in UG266: Silicon Labs Gecko Bootloader User's Guide for GSDK 3.2 and Lower and 5.1 in UG489: Silicon Labs Gecko Bootloader User's Guide for GSDK 4.0 and Higher) if the SE or bootloader upgrade image is staged such that any portion of it extends beyond a 512 kB (address offset 0x7D000) boundary. The default staging area for these operations in the Gecko Bootloader configuration tool is address offset 0x8000 (32768 decimal), which lies well below the 512 kB boundary. • Changing the staging offset for SE firmware and bootloader upgrade images is not common, and most projects are not expected to be impacted by this. • Devices with affected date codes that are upgraded to SE firmware version 2.1.8 or later will verify that the SE firmware or bootloader upgrade image does not extend beyond the 512 kB boundary and, in such cases, will not perform the upgrade and will return an error code, thus allowing the device to be recovered. • OTA and OTW (serial port) upgrades of the application firmware are not impacted. • Upgrades of SE firmware or the bootloader over the serial wire debug interface are also not impacted. • **Devices with date code 2216 or later are unaffected.**
Workaround
• Upgrade the bootloader over the debug interface with a configuration that uses the default staging offset at 0x8000. • Upgrade the SE firmware over the debug interface to version 2.1.8 (to be released in Gecko SDK v4.1.0) or later.
Resolution
Use one of the workarounds for affected devices.

4. Revision History

Revision 0.3

December, 2022

- Added EUSART_E303 and EUSART_E304.
- Fixed workaround routing for KEYSCAN_E301.

Revision 0.2

August, 2022

- Updated to module revisions V2 and V3.
- Added and resolved CUR_E303.
- Added IADC_E306.
- Added KEYSCAN_E301.
- Resolved RADIO_E304.
- Resolved RADIO_E305.
- Added RADIO_E307.
- Added and resolved SE_E301.

Revision 0.1

November, 2021

- 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

www.silabs.com