EFM8 Busy Bee
EFM8BB3 Errata

This document contains information on the EFM8BB3 errata. The latest available revision of this device is revision D.

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: November, 2020.

1. Errata Summary

The table below lists all known errata for the EFM8BB3 and all unresolved errata in revision D of the EFM8BB3.

Table 1.1. Errata Overview

Designator	Title/Problem	Workaround Exists	Exists on Revision:			
			A	B	C	D
CLU_E101	CLU Wake-Up Sources are Level Triggered	No	X	—	—	—
DAC_E101	DAC1 and DAC3 outputs are not updated	Yes	X	X	X	X
I2CSLAVE_E101	I2CSLAVE0 Cannot Distinguish Between Multiple Addresses	No	X	—	—	—
PKG_E101	Top Marking Right Justified	No	X	—	—	—
POR_E102	P0.3 Drives Low During Startup	No	—	X	—	X
RST_E101	VREF/P0.0 Not Retained through Power-On Reset	No	X	—	—	—
TIMER_E101	Timer 3/4 Chaining Mode in Suspend	Yes	X	—	—	—
TIMER_E102	High Byte Overflow Flag and Low Byte Overflow Flag are not cleared	Yes	X	X	X	X
UART1_E101	Some Data Patterns Cause Inadvertent LIN Break Detection	Yes	X	X	X	—
WDT_E101	Restrictions on Watchdog Timer Refresh Interval	Yes	X	X	X	X
WDT_E102	Restrictions on changing Watchdog Timer Interval	Yes	X	X	X	X
XOSC_E101	Crystal Mode in External Oscillator Not Available	No	X	—	X	—
XOSC_E102	External Oscillator XFCN = 111 Setting Unavailable when XOSCMD = CMOS_DIV_2	Yes	—	X	—	X

2. Current Errata Descriptions

2.1 DAC_E101 – DAC1 and DAC3 outputs are not updated

Description of Errata
When DAC1/DAC3 is configured to use non-alternating mode (D1AMEN = NORMAL, D3AMEN = NORMAL in DACGCF0) and DAC1/DAC3 is configured to use another channel as its input (D1SRC = DAC0 or DAC0_INVERT, D3SRC = DAC2 or DAC2_INVERT in DACGCF0), then the DAC1/DAC3 outputs are not updated.
Affected Conditions / Impacts
Systems using DAC1/DAC3 in normal mode (non-alternating) and that use other channels as the source inputs (DAC0/DAC2) for DAC1/DAC3 will not see changes to the DAC1/DAC3 outputs, regardless of what is written to DAC0L/H and DAC2L/H.
Workaround
For the DAC1 and DAC3 outputs to be updated, D1UDIS/D3UDIS in DACGCF1 must be toggled or DAC1H/DAC3H needs to be written.
Resolution
There is currently no resolution for this issue.

2.2 POR_E102 – P0.3 Drives Low During Startup

Description of Errata
On some devices, during power-on reset (POR) with a slow power-on ramp, P0.3, the External Oscillator Output (EXTOSC) pin, may drive low until VDD rises above V_{VDDM} max and the device begins code execution. The normal behavior for P0.3 during POR is that the pin is pulled logic high via weak pull-up.
Affected Conditions / Impacts
External devices and peripherals connected to P0.3 requiring a logic high via weak pull-up during POR may fail to operate properly.
Workaround
There is currently no workaround for this issue.
Resolution
This issue is resolved in revision A and revision C devices.

2.3 TIMER_E102 – High Byte Overflow Flag and Low Byte Overflow Flag are not cleared

Description of Errata
When TIMERn is enabled and firmware writes TMRnCN0 to 0, the TMRnCN0_TFxH and TMRnCN0_TFxL flags are not always cleared.
Affected Conditions / Impacts
When TIMERn is enabled, a high byte/low byte overflow event can happen in the same cycle as the firmware is writing a 0 to TMRnCN0. It is possible that TMRnCN0_TFxH and TMRnCN0_TFxL are still set by hardware even after firmware writes to clear them.
Workaround
TIMERn must be completely stopped before attempting to clear TMRnCN0. This guarantees that the TMRnCN0_TFxH and TMRnCN0_TFxL flags are cleared as expected.
Resolution
There is currently no resolution for this issue.

2.4 WDT_E101 – Restrictions on Watchdog Timer Refresh Interval

Description of Errata

If the Watchdog Timer (WDT) is enabled, firmware will periodically write an 0xA5 value to the WDTCN register to refresh the timer and prevent the watchdog reset from occurring. However, if firmware writes to WDTCN more than once during the same LFOSC0 clock period, the refresh signal may be canceled, resulting in an unintended watchdog reset when the timer expires.

Affected Conditions / Impacts

If firmware refreshes the watchdog more than once in the same LFOSC0 clock period, an unexpected watchdog reset can occur.

Workaround

Systems using the Watchdog Timer (WDT) should ensure that the WDT is refreshed no more than once per LFOSC0 clock period.

Firmware can do this by using timers to count LFOSC0 clock periods. There are three methods to accomplish this:

1. If Timer 3 is not already in use, set it up to capture on the LFOSC0 clock. In this mode, the value of the Timer 3 reload registers does not matter. Instead, the WDT refresh function should check for the 16-bit timer flag (TF3H) to be set in the reset watchdog function, which indicates that a capture event occurred. If the device has another timer that can capture on the LFOSC0 clock, then that timer may be used instead of Timer 3.

```
void refresh_wdt()
{
   // Only refresh if TF3H is set
   if (TMR3CN0 & (0x80))
   {
      WDTCN = 0xA5;
      TMR3CN0 &= ~(0x80);
   }
}
```

2. If any timer is already in use, is clocked from the LFOSC0, and the low overflow flag is not already in use, firmware can check the low byte overflow flag (TFnL) to ensure at least one clock period has passed. For example, using Timer 3:

```
void init_wdt()
{
  // whatever code needed to initialized watchdog

  // intentionally set the TF3L flag (assuming SFRPAGE is correct)
  TMR3CN0 |= 0x40;
}

void refresh_wdt()
{
  static uint8_t last_tmr3l = 0;

  if ( (TMR3CN0 & 0x40) || (last_tmr3l != TMR3L) )
  {
    WDTCN = 0xA5;
    TMR3CN0 &= ~0x40;
    last_tmr3l = TMR3L;
  }
}
```

3. If the application already has an accurate and reliable time base, use that timer to establish a minimum WDT refresh interval that is longer than one LFOSC0 clock period in duration, similar to method (2) above as appropriate.

See the Knowledge Base article on this errata for more information, including examples of these firmware workarounds: https://www.silabs.com/community/mcu/8-bit/knowledge-base.entry.html/2016/11/28/wdt_e101_-_restricti-Vqe5.

Note: The LFOSC0 does not halt while debugging. This can cause the timer overflow flag to be set more quickly than expected when debugging the watchdog refresh function.

Resolution

There is currently no resolution for this issue.

2.5 WDT_E102 – Restrictions on changing Watchdog Timer Interval

Description of Errata
A watchdog reset can occur when the Watchdog Timer (WDT) is disabled.
Affected Conditions / Impacts
If the WDT timeout interval is changed from a higher interval to a lower interval, regardless if the WDT is enabled or disabled, a watchdog reset can occur
Workaround
This can be resolved by refreshing and disabling the WDT before changing the WDT timeout interval from a higher interval to lower interval. Following is the sequence of code that needs to be followed when changing the WDT interval. **Note:** User must insert the code to wait. It is not explicity added in the above sequence as it depends on the divided LFOSC0 clock and the SYSCLK clock selected by the user.
Resolution
There is currently no resolution for this issue.

```
void change_interval()
{
   WDTCN = 0xA5;          // Refresh WDT
   // Insert code to wait for 2 divided LFOSC0 clock periods
   WDTCN = 0xDE;          // Disable WDT (first key)
   WDTCN = 0xAD;          // Disable WDT (second key)
   // Insert code to wait for 3 divided LFOSC0 clock periods
   WDTCN = WDT_interval   // Change the current WDT interval to a lower interval with the MSB cleared to 0
   // Insert code to wait for 1 SYSCLK clock period
}
```

2.6 XOSC_E102 – External Oscillator XFCN = 111 Setting Unavailable when XOSCMD = CMOS_DIV_2

Description of Errata
When the external oscillator is configured for external CMOS clock mode with divide by 2 (XOSCMD = CMOS_DIV_2 in XOSC0CN), it is not possible to use the 111 frequency control setting (XFCN = 111 in XOSC0CN).
Affected Conditions / Impacts
Systems that need to use the XFCN = 111 setting cannot do so in the CMOS_DIV_2 (external clock divided-by-2) configuration.
Workaround
Use other clock control options to achieve the desired divide-by-2 when using an external oscillator, e.g. set CLKDIV = SYSCLK_DIV_2 in CLKSEL.
Resolution
This issue is resolved in revision A and revision C devices.

3. Resolved Errata Descriptions

This section contains previous errata for EFM8BB3 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 CLU_E101 – CLU Wake-Up Sources are Level Triggered

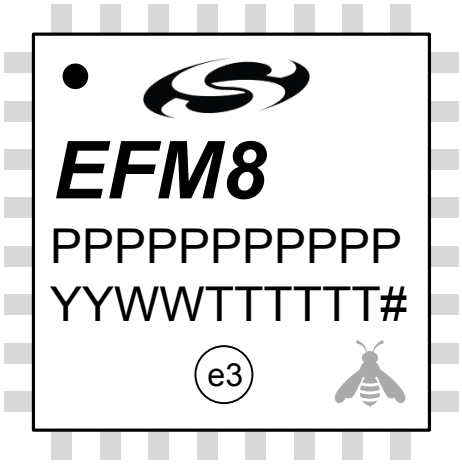
Description of Errata
The device reference manual describes the CLU interrupt-enabled Suspend or Snooze wake-up sources as edge triggered. However, these wake sources are level triggered on revision A devices.
Affected Conditions / Impacts
Systems using the CLU to wake from Suspend or Snooze must disable the CLU wake-up sources following a wake event if they remain high and firmware should not execute the ISR more than once.
Workaround
Since the CLU wake-up sources are level triggered, the CLU output causing the wake-up event will continously trigger interrupts if the output remains active. Systems not desiring this behavior should disable the CLU wake-up sources after a wake event to ensure the ISR is executed only once. The wake-up sources can be re-enabled before re-entering Suspend or Snooze.
Resolution
This issue is resolved in revision B devices.

3.2 I2CSLAVE_E101 – I2CSLAVE0 Cannot Distinguish Between Multiple Addresses

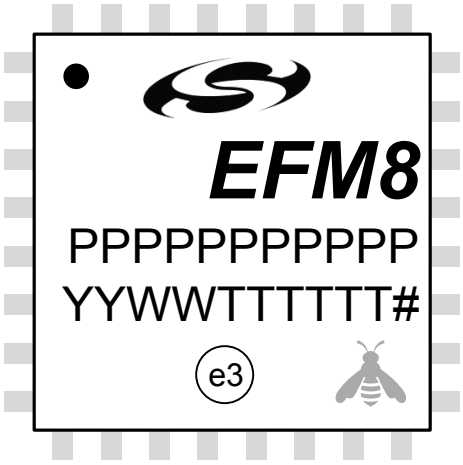
Description of Errata
The device reference manual mentions that the I2CSLAVE0 module can respond to multiple slave addresses and distinguish between these addresses using the receive FIFO when the ADDRCHK bit in I2C0CN0 is set to 1. The ADDRCHK bit is not available on revision A devices, and these devices cannot distinguish between multiple possible slave addresses that could have been used in the initiated transaction.
Affected Conditions / Impacts
Systems intending to use distinguish between multiple addresses cannot do so with revision A devices.
Workaround
There is currently no workaround for this issue.
Resolution
This issue is resolved in revision B devices.

3.3 PKG_E101 – Top Marking Right Justified

Description of Errata
The Package Marking diagrams indicate that the package marking is left justified. However, QFP32 devices with the 1531 date code have package markings that are right justified.



Correct QFP32
Package Marking



Incorrect QFP32
Package Marking

Affected Conditions / Impacts
Assembly houses expecting left-justified package marking may see some parts with right-justified package marking.
Workaround
There is currently no workaround for this issue.
Resolution
This issue only affects revision A QFP32 devices with a date code of 1531. All other devices should match the Package Marking diagrams in the device data sheet.

3.4 RST_E101 – VREF/P0.0 Not Retained through Power-On Reset

Description of Errata
The VREF/P0.0 pin output is expected to be retained on any non-POR reset if the RSTMD bit is set to the PERSIST state in the DACnCF0 register. However, the VREF/P0.0 output on revision A devices is not retained if the PINRETAIN bit in PCON1 is set (in addition to the RSTMD bit in the DAC) and drops from its level to 0 V on any non-POR reset. All DAC SFRs and REF0CN are properly retained. After a non-POR reset, if firmware clears the PINRETAIN bit to 0, the VREF/P0.0 pin will return back to its original level. This is true for both the 1.2 V and 2.4 V options.
Affected Conditions / Impacts
Systems intending to use the pin retain feature for VREF/P0.0 will not be able to do so without using the proposed workaround.
Workaround
Systems intending to use the pin retain feature for VREF/P0.0 must clear the PINRSTMD bit in the PCON1 register after a reset.
Resolution
This issue is resolved in revision B devices.

3.5 TIMER_E101 – Timer 3/4 Chaining Mode in Suspend

Description of Errata
The Timer 3/4 32-bit counter on revision B devices will not switch to the low frequency oscillator (LFOSC0) after entering Suspend mode if the system clock divider is set to a value of divide-by-4 or greater.
Affected Conditions / Impacts
Systems using the Timer 3/4 32-bit counter in chained mode while in Suspend should use the recommended system clock divider settings to ensure proper operation.
Workaround
When using the Timer 3/4 32-bit counter in Suspend mode, set the system clock divider to the divide-by-1 or divide-by-2 settings before entering Suspend mode.
Resolution
This issue is resolved in revision B devices.

3.6 UART1_E101 – Some Data Patterns Cause Inadvertent LIN Break Detection

Description of Errata
If UART1 is used in LIN mode (LINMDE = 1 in UART1LIN), certain data patterns consisting of a byte whose MSBs resemble a start bit (e.g. 0b101xxxxx when accounting for LSB first transmission) followed by 0x0 are improperly detected as a LIN break sequence.
Affected Conditions / Impacts
Because LIN frames can have a variable length of 2, 4, or 8 bytes, the detection of a break when it is not actually sent on the LIN bus could result in application software expecting the arrival of a new frame. Furthermore, if autobaud is enabled (AUTOBDE = 1 in UART1LIN), the detection of a break at the wrong time would result in the interpretation of the next character as the sync character, which is not likely to be the required 0x55. By attempting to sync on the wrong character, the baud rate determined would be wrong, and communication with the master would be lost due to baud rate mismatch.
Workaround
There is currently no workaround for this issue. **Note:** The inadvertent triggering of the autobaud logic due to improper detection of break does not apply when UART1 is not being used in LIN mode. By following the procedure described in the reference manual, whereby it is enabled pending detection of the sync character and then disabled after the sync character is received, UART1 autobaud detection functions as expected.
Resolution
This issue is resolved in revision D devices.

3.7 XOSC_E101 – Crystal Mode in External Oscillator Not Available

Description of Errata
The data sheet mentions support for an external crystal oscillator. However, the crystal mode in the external oscillator option is not available on revision A devices.
Affected Conditions / Impacts
Systems intending to use the crystal mode in the external oscillator will not be able to do so. However, other external oscillator modes (CMOS or RC) are available.
Workaround
Systems needing to use an external oscillator should use the available external oscillator modes (CMOS or RC) with revision A devices.
Resolution
This issue is resolved in revision B and revision D devices.

4. Revision History

Revision 1.0

November, 2020

- Updated the latest revision to revision D.
- Corrected affected revisions for POR_E102 and XOSC_E101.
- Added DAC_E101, TIMER_E102 and XOSC_E102.
- Resolved UART1_E101 and moved to Resolved Errata Descriptions.
- Migrated to new errata document format.

Revision 0.9

May, 2019

- Added UART1_E101.

Revision 0.8

December, 2018

- Updated latest revision to C.
- now reflects the presence of POR_E102 on revision A devices.
- Moved POR_E102 from the errata to the errata history and changed XTAL2 pin reference to EXTOSC in order to conform with revision C documentation.

Revision 0.7

November, 2018

- Added WDT_E102.

Revision 0.6

August, 2018

- Merged errata history and errata into one document.
- Updated the second workaround in WDT_E101.
- Updated Knowledge Base article link in WDT_E101
- Added POR_E102.

Revision 0.5

September, 2016

- Added WDT_E101.

Revision 0.4

February, 2016

- Updated latest revision to B.
- Moved CLU_E101, I2CSLAVE_E101, PKG_E101, RST_E101, TIMER_E101, and XOSC_E101 from the errata to the errata history.

Revision 0.3

October, 2015

- Updated RST_E101, TIMER_E101, and XOSC_E101 with errata designators.
- Added CLU_E101, I2CSLAVE_E101, and PKG_E101.

Revision 0.1

June, 2015

- 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