MAX32620
ERRATA SHEET

Revision A4 Errata

The errata listed below describe situations where components of this revision perform differently than expected or differently than described in the data sheet. Maxim Integrated Products, Inc., intends to correct these errata when the opportunity to redesign the product presents itself.

This errata sheet only applies to components of this revision. These components are branded on the topside of the package with a six-digit code in the form yywwRR, where yy and ww are two-digit numbers representing the year and work week of manufacture, respectively, and RR is the revision of the component. To obtain an errata sheet on other die revisions, visit our website at ***www.maximintegrated.com/errata****.*

1) CRC-32 HARDWARE BLOCK DOES NOT CALCULATE CORRECTLY IN 8- OR 16-BIT MODES

Description:
The values returned by the CRC-32 hardware do not match the expected value when used in 16- or 8-bit modes. Data is processed by the CRC-32 hardware in big-endian format, rather than the little-endian format used by the Cortex-M CPU memory.

Workaround:
The CRC-32 engine only works for 32-bit mode and must be loaded in blocks of 4 bytes.

Convert the CRC-32 input to big-endian format before loading the CRC-32 hardware. After the calculation is complete, the CRC-32 output must be converted back to little-endian format.

2) SPIX FREQUENCY MUST BE SAME AS CM4 CLOCK FREQUENCY

Description:
The SPI execute in place (SPIX) clock frequency is incorrect unless the clock frequency matches the CM4 clock frequency.

Workaround:
Calculate the value of CLKMAN_SYS_CLK_CNTL_2_SPIX that provides the correct SPIX frequency. The same value must be written to the CLKMAN_SYS_CLK_CNTL_0_CM4 register.

3) SPIX HARDWARE LEAVES SLAVE ACTIVE AFTER COMPLETING EXTERNAL READ

Description:
The SPI execute in place (SPIX) peripheral does not return the slave select signal to the idle state after program execution returns to the internal memory. This causes some external memories to remain active, increasing power consumption even when the memory is not being accessed.

Workaround:
If the external memory only serves as additional data storage, use the SPI master peripheral for access. When executing code from the external memory, disable the SPIX module upon returning to the internal memory, reenabling it for each new access. This does have the side effect of incurring a time penalty as the SPIX module repeats its initialization phase each time the peripheral is enabled.

4) SPI MASTER MAY GENERATE EXTRA CLOCKS AFTER FIFO EMPTY

Description:
The SPI master may generate additional SCK clocks after the last byte in the FIFO is transmitted.

Workaround:
The user should avoid creating a stall by keeping data in the transaction FIFO until the normal completion of the transfer.

5) SPI MASTER GENERATES TX_READY INTERRUPT WHEN HEADER IS WRITTEN TO FIFO

Description:
The TX_READY interrupt occurs as soon as the header is written to the FIFO instead of when the FIFO is empty and SSEL is deasserted.

Workaround:
Clear the first TX_READY interrupt immediately following the header write to the transmit FIFO.

Note: This software workaround is implemented in the latest version of the MAX32620 SDK/API. No workaround is necessary if the supplied drivers are used.

6) DEVICE MAY NOT RESUME FROM LP1 MODE

Description:
The device does not exit LP1 mode correctly if dynamic clock gating is enabled for the flash memory controller. Dynamic clock gating is enabled by default following a POR.

Workaround:
Set CLKMAN_CLK_GATE_CTRL_0 [7:6] = 0x3 before entering LP1 mode.

Note: This software workaround is implemented in the latest version of the MAX32620 SDK/API. No workaround is necessary if the supplied drivers are used.

7) ADC OUTPUT MAY BE INCORRECT IF V_{AIN} IS APPROXIMATELY 0V or V_{REF}

Description:
If (V_{AIN} + ADC's Offset) > V_{REF}, or (V_{AIN} + ADC's offset) < 0V, the ADC output wraps around 0x000 or 0x3FF and produces an incorrect data value.

Workaround:
Store FTR_TRIM_REG_12[31:28] to memory. Call this variable "AdcOffsetTrim".

Set FTR_TRIM_REG_12[31:28] = 0x0 before starting the ADC. This forces the ADC to use a trim of 0V.

After the ADC sample is complete, apply the code clamping to create the correct result "AdcDataFinal":

```
if ((ADC_Status.AdcOvrFlw == 1) (
  if (AdcOffsetTrim >=0)
    AdcDataFinal = 0x3FF;
  else
    AdcDataFinal = 0x3FF + AdcOffsetTrim;
  )


else (
  if (ADC_Data.AdcData + AdcOffsetTrim < 0x0)
    AdcDataFinal = 0x0;
  else if ((ADC_Data.AdcData + AdcOffsetTrim) > 0x3FF)
    AdcDataFinal = 0x3FF;
  else
    AdcDataFinal = ADC_Data.AdcData + AdcOffsetTrim;
  )
```

Note: This software workaround is implemented in the latest version of the MAX32620 SDK/API. No workaround is necessary if the supplied drivers are used.

8) ADC CHANNEL 9 MUST USE INTERNAL DIVIDE BY TWO SETTING

Description:
Measurements on ADC channel 9 are incorrect, unless the ADC is using the internal divide by two mode.

Workaround:
Configure channel 9 to use the ADC internal divide by 2 mode. Measurement accuracy can be affected if $V_{RTC} > V_{DD18} + 0.2V$.

Note: This software workaround is implemented in the latest version of the MAX32620 SDK/API. No workaround is necessary if the supplied drivers are used.

9) BASE_PART_NUMBER FIELD READS 0x7F67

Description:
The value in base_part_number field for this device is expected to be 0x7F6C (32620). Instead the value 0x7F67 (32615) is returned.

Workaround:
None required.

10) USB HARDWARE CANNOT DETECT SUBSEQUENT SUSPEND EVENTS AFTER INITIAL SUSPEND/RESUME CYCLE

Description:
The USB peripheral properly responds to an initial suspend/resume cycle from the USB host. However, it fails to detect any subsequent suspend events from the host. If the device wakes up from something other than bus activity, it does need to send the remote wakeup signal before going back to sleep. It is only necessary to send the remote wakeup signal when the bus activity from the host resumes.

Workaround:
Set the SIGRWU bit when the device wakes up from bus activity. This resolves the issue and the device again detects a suspend condition. Note that this has the side effect of generating a remote wakeup signal to the host.

Revision History

REVISION NUMBER	REVISION DATE	DESCRIPTION	PAGES CHANGED
0	12/15	Initial release	—