# MAX32650–MAX32652
# ERRATA SHEET

## Revision A1 Errata

*The errata listed below describe situations where components of this revision perform differently than expected or differently than described in the data sheet. Analog Devices may, at its own discretion, take future steps to correct these errata when the opportunity to redesign the product presents itself. Prior to that, Analog Devices has determined the following potential workarounds that customers may want to consider when addressing one of the situations described below.*

*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 the Maxim website at* ***www.maximintegrated.com/errata****.*

### 1) DATA WRITES TO GCR_MEMCKCN[31:6] ARE MISALIGNED

**Description:**
Data written to GCR_MEMCKCN[31:6] is unintentionally shifted three positions to the left.

**Workaround:**
Shift the 32 bits of the intended data three positions to the right before the write operation. Bits GCR_ MEMCKCN[15:3] are reserved for future use, and shifting data through these bits does not affect device operation. After the write operation, an OR operation must be used to program the Program Flash Wait State bits, GCR_MEMCKCN[2:0], to the desired nonzero value.

The following example puts System RAM 5 and 6 in light sleep mode and sets the Program Flash Wait States to 3:

```
GCR_MEMCKCN = (0x00600000 >> 3) | 0x3;
```

### 2) GCR_MEMCKCN[5:3] MIRROR GCR_MEMCKCN[2:0]

**Description:**
GCR_MEMCKCN[5:3] maintain their reset value of 000b unless explicitly changed. Reads of these bits, however, returns the value of GCR_MEMCKCN[2:0] instead.

**Workaround:**
None required. GCR_MEMCKCN[5:3] are RFU.

*419-100011; Rev 4; 4/22*

### 3) SMART DMA CANNOT OPERATE AS INTENDED IF RESET OCCURS DURING SDMA LOW-POWER OPERATION

**Description:**
The SDMA can become unresponsive if a system reset occurs while the SDMA is in its low-power mode. The only way to recover from this situation is to perform a power-on reset of the device.

**Workaround:**
Characterize the system operation to determine the most likely circumstances under which a system reset can occur, and avoid placing the SDMA in its low power mode during those periods.

### 4) I$^2$S IN SLAVE MODE CAN RECORD INCORRECT DATA IF A PARTIAL WORD IS RECEIVED DURING LEFT CHANNEL RECEPTION

**Description:**
While in slave mode, receipt of a partial/truncated word in the left data channel loads incorrect data into the I$^2$S Rx FIFO.

**Workaround:**
1) Do not enable the I$^2$S peripheral while the I$^2$S master is transmitting.
2) Ensure that the I$^2$S master begins all transmissions with a complete word.

### 5) SYSTEM CANNOT OPERATE AS INTENDED WHEN USING THE 32kHz CRYSTAL OR INTERNAL 7MCLK CLOCK AND THE DEVICE IS RESET

**Description:**
The device does not operate as expected if the RSTN pin is asserted or a watchdog timer occurs while the 32kHz crystal or internal 7MCLK clock is selected as the system clock source.

**Workaround:**
Use the internal high frequency oscillator as the system clock source and select a large divisor if low-speed/low-power operation is required.

### 6) HYP_CK AND HYP_CKN DRIVE STRENGTH IS FIXED AT 1x

**Description:**
The drive strength of the HYP_CK and HYP_CKN pins is fixed at the 1x setting regardless of the control bits.

**Workaround:**
None. Ensure the interface to the HYP_CK and HYP_CKN pins is compatible with the relevant electrical specifications in the data sheet.

### 7) DEVICE DOES NOT EXIT BACKGROUND MODE IF SYSTEM CLOCK FREQUENCY IS FASTER THAN THE LPCLK FREQUENCY

**Description:**
The device does not exit Background mode if the system clock frequency is faster than the LPCLK frequency.

**Workaround:**
The software workaround for this erratum has been implemented in the appropriate Analog Devices-supplied API.

### 8) HYP_CS1 SIGNAL IS NOT DRIVEN INACTIVE WHILE THE HYPERBUS IS IDLE

**Description:**
The output driver on the HYP_CS1 pin is disabled when the hyperbus is idle. Devices connected to the HYP_CS1 signal are not guaranteed to enter their deselected state.

**Workaround:**
Configure the GPIO associated with the HYP_CS1 pin to input mode with the internal strong/normal pull-up enabled. This pulls the HYP_CS1 signal to its inactive state while the hyperbus is idle.

### 9) $V_{DDA}$ NOT SUPPORTED AS ADC REFERNCE

**Description:**
The $V_{DDA}$ supply is not available as an internal reference for the ADC.

**Workaround:**
None. Ignore descriptions that mention $V_{DDA}$ as a reference for the ADC.

### 10) $V_{DDIO}$ AND $V_{DDIOH}$ POWER MONITORS DISABLED

**Description:**
The $V_{DDIO}$ and $V_{DDIOH}$ power monitors are disabled by default. Writes to LP_CTRL[24:23] have no effect.

**Workaround:**
None.

### 11) DEVICE DOES NOT OPERATE AS EXPECTED IF CERTAIN GPIO PORT 2 PINS CHANGE STATE WHILE OPERATING FROM FLASH MEMORY

**Description:**
The device will execute incorrect instructions if the state of certain GPIO port 2 pins change while executing code from internal flash memory. The change of state can be caused by an external signal driven into the pin, or software changing the GPIO port registers associated with the affected pins. The affected pins are:
P2[7]
P2[8]
P2[14]
P2[15]
P2[16]
P2[24]
P2[28]

**Workaround:**

1) Do not use the affected pins and leave them unconnected while executing code from flash memory. Configure the GPIO with the strong pull-up connected to the external supply. Do not change the state of the corresponding bits in the GPIO2_OUT register from their default value. Alternately the pin can be connected to a static external signal if that signal remains at $V_{DDIO}$ while executing code from flash memory.
2) Copy the desired code into SRAM and ensure the device only executes code located SRAM by disabling interrupts. The affected port pins can be used as inputs or outputs without restrictions. Before resuming code execution from flash, ensure the signal on the device pin remains at a static $V_{DDIO}$.

### 12) SPI MODE 1 AND MODE 3 OPERATION MAY BE AFFECTED WHEN QSPIn_CLK_CFG.SCALE = 0

**Description:**
The following settings are invalid when operation the SPI is operating in mode 1 or mode 3: (13390)
SCALE = 0, CLOCKHI = 0, CLOCKLO = 0
SCALE = 0, CLOCKHI = 1, CLOCKLO = 1

**Workaround:**
Do not use the invalid settings. The operating speeds generated by all other field combinations are valid.

### 13) SPIXF BUS IDLE FEATURE CANNOT BE DISABLED

**Description:**
The SPIXF bus idle feature cannot be disabled. (14358)

**Workaround:**
Set the SPIXF_BUS_IDLE.busidle to 0x1 to approximate the operation of the SPIXF when the bus idle feature is disabled. This will have minimal effect on sequential code execution.

### 14) UART RECEIVER REQUIRES ONE EXTRA BIT TIME BETWEEN STOP AND START BITS

**Description:**
The UART requires an external transmitter to send at least one more stop bit than specified in the UART_CTRL0.stop field.

**Workaround:**
There are two workarounds:
1) Configure the transmitter to send at least one additional stop bit than specified in the UART_CTRL0.stop field.
2) Ensure the transmitter generates at least one idle bit time between byte transmissions.

### 15) DO NOT USE UART Tx AND Rx FUNCTIONALITY SIMULTANEOUSLY

**Description:**
The peripheral does not operate as expected if both the receive and transmit functions are used simultaneously.

**Workaround:**
None. Assign the transmit and receive functionality to different peripherals if full-duplex operation is required.

### 16) UART TRANSMITTER GENERATES SPURIOUS PULSE WHEN USING 7.3728MHz CLOCK SOURCE

**Description:**
The UART generates a negative pulse two PCLK periods wide on the UART_TX line one bit period before the falling edge of the start bit. (10689).

**Workaround:**
There are two workarounds:
1) Use PCLK as the clock source when transmitting.
2) Ensure the external receiver will reject the spurious pulse.

### 17) I$^2$S/SPIMSS AND SPI0 SHARE THE SPI0 IRQ HANDLER

**Description:**
Interrupt sources from both the I$^2$S/SPIMSS and SPI0 peripherals call the same SPI0_IRQHandler.

**Workaround:**
If using both I$^2$S/SPIMSS and SPI0 simultaneously, software must interrogate both the I$^2$S and SPI0 interrupt flags to determine which peripheral caused the interrupt. Both peripherals can generate interrupts.

### 18) I$^2$S SIGNAL INCORRECTLY TRANSMITTED ON P0.22 WHEN I$^2$S IS ENABLED AND SPI0 IS DISABLED

**Description:**
The I2S_LRCLK signal is incorrectly transmitted on P0.22 under the following conditions:

- P0.22 is configured as SPI0_SS0 (AF1).
- P2.5 is configured as I2S_LRCLK (AF1).
- I$^2$S is enabled in master mode.
- SPI0 is disabled.

**Workaround:**
To prevent the I2S_LRCLK signal from being transmitted on P0.22, configure P0.22 as GPIO.

# MAX32650–MAX32652
# REV A1 ERRATA

## Revision History

| REVISION NUMBER | REVISION DATE | DESCRIPTION | PAGES CHANGED |
|---|---|---|---|
| 0 | 5/18 | Initial release | — |
| 1 | 8/19 | Added errata 9, 10, and 11 | 3 |
| 2 | 12/20 | Added errata 12 and 13 | 4 |
| 3 | 6/21 | Added errata 14 and 15 | 4 |
| 4 | 4/22 | Added erratum 16, 17, and 18 | 4, 5 |

*granted by implication or otherwise under any patent or patent rights of Analog Devices. Trademarks and registered trademarks are the property of their respective owners.*

One Analog Way, Wilmington, MA 01887 U.S.A. | Tel: 781.329.4700 | © 2022 Analog Devices, Inc. All rights reserved.