Ultra Low Power ARM Cortex-M3 MCU with Integrated Power Management

Silicon Anomaly List

ADuCM3027/9

ABOUT ADuCM3027/9 SILICON ANOMALIES

These anomalies represent the currently known differences between revisions of the ADuCM3027/9 product(s) and the functionality specified in the ADuCM3027/9 data sheet(s) and the Hardware Reference book(s).

SILICON REVISIONS

A silicon revision number with the form "-x.x" is branded on all parts. The silicon revision can be electronically determined by reading bits <3:0> of the **SYS_CHIPID** register.

Silicon REVISION	SYS_CHIPID.REV
1.2	0x2

ANOMALY LIST REVISION HISTORY

The following revision history lists the anomaly list revisions and major changes for each anomaly list revision.

Date	Anomaly List Revision	Data Sheet Revision	Additions and Changes
03/03/2017	D	0	Removed Silicon Revision 1.0 Added Silicon Revision 1.2 Added Anomalies: 21000015, 21000016, 21000017
02/19/2016	C	PrF	Removed Anomaly: 21000012
12/04/2015	B	PrE	Removed Deprecated Part Numbers ADuCM3023/5
11/26/2014	A	PrC	Initial Version

One Technology Way, P.O.Box 9106, Norwood, MA 02062-9106 U.S.A.
Tel: 781.329.4700
Technical Support www.analog.com

SUMMARY OF SILICON ANOMALIES

The following table provides a summary of ADuCM3027/9 anomalies and the applicable silicon revision(s) for each anomaly.

No.	ID	Description	Rev 1.2
1	21000011	I2C Master Mode Fails to Generate Clock when Clock Dividers Are Too Small	x
2	21000015	Pin P2_11 Is Not Retained After Shutdown Wake Up	x
3	21000016	Possible Receive Data Loss with I2C Automatic Clock Stretching	x
4	21000017	SPI Read Command Mode Does Not Work Properly When SPI_CNT Is 1 and DMA Is Enabled	x

Key: x = anomaly exists in revision
. = Not applicable

DETAILED LIST OF SILICON ANOMALIES

The following list details all known silicon anomalies for the ADuCM3027/9 including a description, workaround, and identification of applicable silicon revisions.

1. 21000011 - I2C Master Mode Fails to Generate Clock when Clock Dividers Are Too Small:

DESCRIPTION:
When the I2C clock dividers are configured in Master mode such that the sum of the **I2C_DIV.LOW** and **I2C_DIV.HIGH** register bit fields is less than 16, the I2C fails to generate a clock.

WORKAROUND:
Program the I2C clock dividers such that **I2C_DIV.LOW** + **I2C_DIV.HIGH** >= 16.

APPLIES TO REVISION(S):
1.2

2. 21000015 - Pin P2_11 Is Not Retained After Shutdown Wake Up:

DESCRIPTION:
The state of pin P2_11 is not retained after waking up from shutdown mode.

WORKAROUND:
None. To retain the pin state through shutdown, use any other GPIO pin instead of P2_11.

APPLIES TO REVISION(S):
1.2

3. 21000016 - Possible Receive Data Loss with I2C Automatic Clock Stretching:

DESCRIPTION:
When the I2C RX FIFO is full and new I2C data is received, a data overflow occurs. When automatic clock stretching is enabled, the transaction is paused by holding the SCL line low. This functions as expected when the next read happens after the clock is stretched (i.e., after the overflow is detected); however, if the read occurs after the last bit of the I2C data is received but before the clock is stretched, the received data is not written to the RX FIFO and is lost.

WORKAROUND:
To avoid waiting for the lost data, use the timeout feature when enabling automatic clock stretching. Configure **I2C_ASTRETCH_SCL.SLV** (for slave mode) and **I2C_ASTRETCH_SCL.MST** (for master mode) with values between 1 and 14. Identify the data loss (**I2C_ASTRETCH_SCL.SLVTMO** = 1 or **I2C_ASTRETCH_SCL.MSTTMO** = 1) and request for data retransmission.

APPLIES TO REVISION(S):
1.2

4. 21000017 - SPI Read Command Mode Does Not Work Properly When SPI_CNT Is 1 and DMA Is Enabled:

DESCRIPTION:

When SPI master is enabled and uses the DMA mode with **SPI_CNT** = 1, the Read command mode may not function properly. Consider the following configurations:

1. **SPI_RD_CTL** = 0x07.
2. **SPI_CNT** = 1.
3. The TX and RX DMA channels are configured for 1 half-word.

In this configuration, the read command sent in the first byte on the MOSI output is repeated in the second byte (in the address slot); thus, the slave device will respond on the MISO line with whatever content is at the address equivalent to the read command value (for example, if the read command is 0xB, the response will be the data read from slave address 0xB).

WORKAROUND:

The following workarounds can be used:

1. Utilize the overlap mode to align the transmit/receive SPI operations and discard the junk bytes, as follows:
   a. Set **SPI_RD_CTL.OVERLAP** = 1 to enable overlap mode.
   b. Set **SPI_RD_CTL.TXBYTES** = 1 to configure a single transmit byte (8-bit address register).
   c. Set **SPI_CNT.VALUE** = 3 to configure the transfer count: one byte for the address register, one byte for the command, and one dummy byte to obtain the read value.
   d. On the receive side, discard the first two junk bytes received during the transfer of the address and command bytes before processing the actual read value in the 3rd byte.

2. Do not use TX DMA operation on the SPI transmit side, as follows:
   a. Enable only SPI RX DMA requests.
   b. Fill the SPI TX FIFO by using core accesses to write the **SPI_TX** register.
   c. Perform a dummy read of the **SPI_RX** register to kick off the SPI transfers.

APPLIES TO REVISION(S):

1.2