MicroConverter® 12-Bit ADCs and DACs with Embedded High Speed 62-kB Flash MCU

Silicon Anomaly List — ADuC841/ADuC842/ADuC843

A. This Anomaly List represents the known bugs, anomalies and work-arounds for the ADuC841/ADuC842/ADuC843 MicroConverter parts.

B. The Anomaly listed, apply to all ADuC841/ADuC842/ADuC843 packaged material branded as follows:

Third Line: **E20 <date code>**

C. Analog Devices Inc. is committed, through future silicon revisions to continuously improve silicon functionality. Analog Devices Inc. will use its best endeavors to ensure that these future silicon revisions remain compatible with your present software/systems implementing the recommended work-arounds outlined in this document.

D. ADuC841/ADuC842/ADuC843 Silicon Anomaly List Revision History :

Revision	Date	Relevance	Silicon Status	# of Bugs Reported
B	May 2004	All Silicon branded Third Line: E20 <date code>	Release	10 Errata

REV B May 2004

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 www.analog.com
Fax: 781.326.8703

1. MODE 0 UART OPERATION [er001]:

Background:
UART mode 0 allows the UART to function in an 8-Bit shift register mode

Issue:
UART mode 0 is non-functional on the ADuC841/ADuC842/ADuC843

Work-Around:
None

Related Issues:
None

2. USE OF THE EXTENDED STACK POINTER [er002]:

Background:
The Extended Stack Pointer allows the Stack to overflow into internal XRAM

Issue:
A PUSH onto the extended stack when it is the first instruction within a subroutine results in the return address being overwritten.

Work-Around:
For Assembly code insert a 'NOP' as the first instruction in any subroutine.
For C code there is currently no workaround.

Related Issues:
None

3. USE OF I2C IN SLAVE MODE WITH STOP INTERRUPT ENABLED [er003]:

Background:
In SLAVE mode, the I2C interface can be configured to generate an interrupt due to a START, REPEATED START, DATA or STOP condition. The I2C interrupt decode bits (I2CID0 and I2CID1) in I2CCON indicate the source of the interrupt. If the STOP interrupt is enabled via the I2CSI bit an interrupt will be generated when the SLAVE receives a STOP condition.

Issue A:
In the I2C interrupt service routine if the I2CI bit or I2CDAT register is accessed during a STOP interrupt the I2C bus will fail to respond to further I2C communication.

Work-Around A:
When a STOP interrupt is detected the user should reset the I2C bus using the I2CRS bit.

Issue B:
When the STOP interrupt is enabled, on occasion the I2C interrupt decode bits will indicate that a START, REPEATED START, or DATA interrupt occurred when the source was in fact a STOP interrupt. If this happens the user may try to clear I2CI or read I2CDAT resulting in the bus failing to respond to further I2C communication.

Work-Around B:
Tie the SCLOCK pin to an I/O pin, this will allow the state of SCLOCK to be read. SCLOCK will only be high during an interrupt if the source is a STOP interrupt.

Related Issues:
4. Use of I2C in slave mode with stop interrupt disabled.

4. USE OF I2C IN SLAVE MODE WITH STOP INTERRUPT DISABLED [er004]:

Background:
In SLAVE mode, the I2C interface can be configured to generate an interrupt due to a START, REPEATED START, DATA or STOP condition. The I2C interrupt decode bits (I2CID0 and I2CID1) in I2CCON indicate the source of the interrupt.

Issue:
Once one REPEATED START is detected by the I2C interface, all subsequent START conditions will be detected as a REPEATED START even if a STOP bit has been received between data transfers.

Work-Around:
None.

Related Issues:
None

5. I2C DATA TRANSFER [er005]:

Background:
The I2CDAT register is used to read or write data to the I2C bus. The I2CDAT register has an SFR address of 9AH.

Issue A:
During an I2C transfer if a user accesses the RAM address 9AH the contents of the I2CDAT SFR can be modified.

Work-Around A:
For Assembly code: Do not use memory location 9AH
For C code: Assign a dummy variable to location 9AH using the following code

```
idata unsigned int ui32Dummy[2] _at_ 0x9A;
```

Issue B:
During an I2C transfer if a user executes either of the following instructions the contents of the I2CDAT SFR can be modified.

```
MOV dest,#9AH
SUBB A,R2
```

Work-Around B:
To prevent code from changing the contents of the I2CDAT SFR make sure that neither of these instructions are executed during an I2C transfer.

Issue C:
A small percentage (<3%) of I2C STOP conditions received by a SLAVE will cause the I2C bus to enter a state where the subsequent START+matching address will not be acknowledged.

Work-Around C:
If a second START+matching address is issued following a NACK, this will be detected correctly.

Related Issues:
None.

6. SPI INTERFACE [er006]:

Background:
The SPI can either be used on the standard pins or can be moved to P3.3, P3.4 and P3.5 by setting the MSPI bit in CFG841/CFG842. When the MSPI bit is set P3.3 should be MISO, P3.4 MOSI and P3.5 SCLOCK.

Issue A:
By setting the MSPI bit the P3.3, P3.4 and P3.5 have the following configuration.
P3.3 = MISO, P3.4 = SCLOCK, P3.5 = MOSI

Work-Around A:
None.

Issue B:
When the ADuC841/ADuC842/ADuC843 is set up as an SPI slave the device may receive or transmit a small percentage of bytes incorrectly.

Work-Around B:
Incorporate checksums into all communication with the ADuC841/ADuC842/ADuC843 slave. This will allow the master devices to retransmit if an error occurs.

Related Issues:
None

7. READING/WRITING TO DATAFLASH/EE [er007]:

Background:
There are 4kB of DATAFLASH/EE which can be used for non-volatile storage.

Issue:
If an interrupt occurs during a DATAFLASH/EE read or write operation, code execution may resume at a random program memory address.

Work-Around:
Disable all interrupts prior to a read or write operation. This can be done by setting the EA bit to 0.

Related Issues:
None

8. PWM OPERATION [er008]:

Background:
The PWM output rate is determined by the PWMxH and PWMxL registers for the PWM0 and PWM1 outputs.

Issue:
Modifying RAM address 2EH will cause the PWM timer to be reloaded.

Work-Around:

For Assembly code: Do not use memory location 2EH
For C code: Assign a dummy variable to location 9AH using the following code

```
idata unsigned int ui32Dummy[2] _at_ 0x2E;
```

9. WDT OPERATION [er009]:

Background:

The ADuC841, ADuC842, and ADuC843 incorporate a Watchdog Timer. The purpose of the WDT is to ensure the part is never stuck in an endless loop by generating either a hardware reset or an interrupt event that vectors to the WDT ISR.

Issue:

If the WDT generates an interrupt as opposed to a hardware reset, and if the ISR subsequently sets up the WDT to time out to a hardware reset, the reset is ignored.

Work-Around:

Ensure that a double write to the WDCON is executed inside the ISR with the first write being a reset of the WDT. For example:

```
void isr_wdt( void ) interrupt 11
{
WDWR = 1;         // This first WDT write is required to get the WDT to work inside the ISR.
WDCON  = 0x60;// Reset WDT.
WDWR = 1;         // Now set the WDT to the required 1s timeout
WDCON  = 0x62;// select reset after 1000mS
while(1);
}

void main(void)
{
EA = 0;
WDWR = 1;         // Allow write to WDCON
WDCON = 0x6A;      // timeout=1000mS, WDT enable, WDT ISR Interrupt
while (1);
}
```

Related Issues:

None

10. LEVEL TRIGGERED EXTERNAL INTERRUPT OPERATION [er010]:

Background:
The ADuC841/ADuC842/ADuC843 incorporate two external interrupt sources (INT0 & INT1) that can be configured to respond to either an edge event or a level event.

Issue:
If an interrupt occurs on the INT0 or INT1 pins and is then removed within one core instruction cycle, the interrupt vector address that is generated may be incorrect resulting in a vector to 0000H. This effectively restarts code execution.

Work-Around:
To ensure that this does not occur all source for a level trigger must be kept low for a minimum of 9 core clock cycles.

Related Issues:
None

ADuC841/ADuC842/ADuC843 Silicon Anomaly Revision History

Errata #	Description	Status
er009	WDT OPERATION	**OPEN, May 2004**
er010	LEVEL TRIGGERED EXTERNAL INTERRUPT OPERATION	**OPEN, May 2004**