dsPIC30F2011/2012 Family Silicon Errata and Data Sheet Clarification

The dsPIC30F2011/2012 family devices that you have received conform functionally to the current Device Data Sheet (DS70139**F**), except for the anomalies described in this document.

The silicon issues discussed in the following pages are for silicon revisions with the Device and Revision IDs listed in Table 1. The silicon issues are summarized in Table 2.

The errata described in this document will be addressed in future revisions of the dsPIC30F2011/2012 silicon.

Note:	This document summarizes all silicon errata issues from all revisions of silicon, previous as well as current. Only the issues indicated in the last column of Table 2 apply to the current silicon revision (**A1**).

Data Sheet clarifications and corrections start on page 14, following the discussion of silicon issues.

The silicon revision level can be identified using the current version of MPLAB® IDE and Microchip's programmers, debuggers and emulation tools, which are available at the Microchip corporate web site (www.microchip.com).

For example, to identify the silicon revision level using MPLAB IDE in conjunction with MPLAB ICD 2 or PICkit™ 3:

1. Using the appropriate interface, connect the device to the MPLAB ICD 2 programmer/debugger or PICkit 3.
2. From the main menu in MPLAB IDE, select *Configure>Select Device*, and then select the target part number in the dialog box.
3. Select the MPLAB hardware tool (*Debugger>Select Tool*).
4. Perform a "Connect" operation to the device (*Debugger>Connect*). Depending on the development tool used, the part number *and* Device Revision ID value appear in the **Output** window.

Note:	If you are unable to extract the silicon revision level, please contact your local Microchip sales office for assistance.

The Device and Revision ID values for the various dsPIC30F2011/2012 silicon revisions are shown in Table 1.

TABLE 1: SILICON DEVREV VALUES

Part Number	Device ID(1)	Revision ID for Silicon Revision(2)
		A1
dsPIC30F2011	0x0240	0x1001
dsPIC30F2012	0x0241	

Note 1: The Device and Revision IDs (DEVID and DEVREV) are located at the last two implemented addresses in program memory.

2: Refer to the *"dsPIC30F Flash Programming Specification"* (DS70102) for detailed information on Device and Revision IDs for your specific device.

TABLE 2: SILICON ISSUE SUMMARY

Module	Feature	Item Number	Issue Summary	Affected Revisions(1)
				A1
—	—	1.	This issue was removed in the “B” revision of this document (DS80450B).	—
CPU	MAC Class Instructions with ±4 Address Modification	2.	Sequential MAC instructions, which prefetch data from Y data space using ±4 address modification, will cause an address error trap.	X
CPU	DAW.b Instruction	3.	The Decimal Adjust instruction, DAW.b, may improperly clear the Carry bit, C (SR<0>).	X
CPU	DISI Instruction	4.	The DISI instruction will not disable interrupts if a DISI instruction is executed in the same instruction cycle that the DISI counter decrements to zero.	X
Interrupt Controller	—	5.	An interrupt occurring immediately after modifying the CPU IPL, interrupt IPL, interrupt enable or interrupt flag may cause an address error trap.	X
Output Compare	—	6.	The Output Compare module will produce a glitch on the output when an I/O pin is initially set high and the module is configured to drive the pin low at a specified time.	X
Output Compare	PWM Mode	7.	Output compare will produce a glitch when loading 0% duty cycle in PWM mode. It will also miss the next compare after the glitch.	X
ADC	Sleep Mode	8.	ADC event triggers from the INT0 pin will not wake-up the device from Sleep mode if the SMPI bits are non-zero.	X
PLL	—	9.	If 4x or 8x PLL mode is used, the input frequency range is 5 MHz-10 MHz instead of 4 MHz-10 MHz.	X
Sleep Mode	—	10.	Execution of the Sleep instruction (PWRSAV #0) may cause incorrect program operation after the device wakes up from Sleep. The current consumption during Sleep may also increase beyond the specifications listed in the device data sheet.	X
Timer	Sleep Mode	11.	Clock switching prevents the device from waking up from Sleep.	X
PLL	Lock Status bit	12.	The PLL LOCK Status bit (OSCCON<5>) can occasionally get cleared and generate an oscillator failure trap even when the PLL is still locked and functioning correctly.	X
PSV Operations	—	13.	An address error trap occurs in certain addressing modes when accessing the first four bytes of any Program Space Visibility (PSV) page.	X
I/O	Port Pin Multiplexed with IC1	14.	The Port I/O pin multiplexed with the Input Capture 1 (IC1) function cannot be used as a digital input pin when the UART auto-baud feature is enabled.	X
I^2C™	10-bit Addressing	15.	The 10-bit slave does not set the RBF flag or load the I2CxRCV register on address match if the Least Significant bits (LSbs) of the address are the same as the 7-bit reserved addresses.	X
I^2C	Slave Mode	16.	The I^2C module loses incoming data bytes when operating as an I^2C slave.	X

Note 1: Only those issues indicated in the last column apply to the current silicon revision.

TABLE 2: SILICON ISSUE SUMMARY (CONTINUED)

Module	Feature	Item Number	Issue Summary	Affected Revisions(1)
				A1
I^2C	10-bit Addressing	17.	When the I^2C module is configured for 10-bit addressing using the same address bits (A10 and A9) as other I^2C devices, the A10 and A9 bits may not work as expected.	X
I^2C	10-bit Addressing	18.	When the I^2C module is configured as a 10-bit slave with an address of 0x102, the I2CxRCV register content for the lower address byte is 0x01 rather than 0x02.	X
I^2C	Bus Collision	19.	When the I^2C module is enabled, the dsPIC® DSC device generates a glitch on the SDA and SCL pins, causing a false communication start in a single-master configuration or a bus collision in a multi-master configuration.	X
ADC	Current Consumption in Sleep Mode	20.	If the ADC module is in an enabled state when the device enters Sleep Mode, the power-down current (I_{PD}) of the device may exceed the device data sheet specifications.	X

Note 1: Only those issues indicated in the last column apply to the current silicon revision.

Silicon Errata Issues

Note:	This document summarizes all silicon errata issues from all revisions of silicon, previous as well as current. Only the issues indicated by the shaded column in the following tables apply to the current silicon revision (**A1**).

1. Module: N/A

This issue was removed in the "B" revision of this document (DS80450B).

2. Module: CPU

Sequential MAC class instructions, which prefetch data from Y data space using ±4 address modification will cause an address error trap. The trap occurs only when all the following conditions are true:

1. Two sequential MAC class instructions (or a MAC class instruction executed in a REPEAT or DO loop) that prefetch from Y data space.
2. Both instructions prefetch data from Y data space using the + = 4 or - = 4 address modification.
3. Neither of the instructions uses an accumulator write back.

Work around

The problem described above can be avoided by using any of the following methods:

1. Inserting any other instruction between the two MAC class instructions.
2. Adding an accumulator write back (a dummy write back if needed) to either of the MAC class instructions.
3. Do not use the + = 4 or - = 4 address modification.
4. Do not prefetch data from Y data space.

Affected Silicon Revisions

A1							
X							

3. Module: CPU

The Decimal Adjust instruction, DAW.b, may improperly clear the Carry bit, C (SR<0>), when executed.

Work arounds

Work around 1:

Check the state of the Carry bit prior to executing the DAW.b instruction. If the Carry bit is set, set the Carry bit again after executing the DAW.b instruction. Example 1 shows how the application should process the Carry bit during a BCD addition operation.

EXAMPLE 1: CHECK CARRY BIT BEFORE DAW.b

```
    .include "p30f3013.inc"
    .......
    MOV.b   #0x80, w0   ;First BCD number
    MOV.b   #0x80, w1   ;Second BCD number
    ADD.b   w0, w1, w2  ;Perform addition
    BRA     NC, L0      ;If C set go to L0
    DAW.b   w2          ;If not,do DAW and
    BSET.b   SR, #C     ;set the carry bit
    BRA     L1          ;and exit
L0:DAW.b    w2
L1: ....
```

Work around 2: For C Language Source Code

For applications using C language, MPLAB C30 versions 1.20.04 or higher provide the following command-line switch that implements a work around for the erratum.

```
-merrata=psv
```

Refer to the "readme.txt" file in the MPLAB C30 v1.20.04 toolsuite for further details.

Affected Silicon Revisions

A1							
X							

4. Module: CPU

When a user executes a DISI #7, for example, this will disable interrupts for 7 + 1 cycles (7 + the DISI instruction itself). In this case, the DISI instruction uses a counter which counts down from 7 to 0. The counter is loaded with 7 at the end of the DISI instruction.

If the user code executes another DISI on the instruction cycle where the DISI counter has become zero, the new DISI count is loaded, but the DISI state machine does not properly re-engage and continue to disable interrupts. At this point, all interrupts are enabled. The next time the user code executes a DISI instruction, the feature will act normally and block interrupts.

In summary, it is only when DISI execution is coincident with the current DISI count = 0, that the issue occurs. Executing a DISI instruction before the DISI counter reaches zero will not produce this error. In this case, the DISI counter is loaded with the new value and interrupts remain disabled until the counter becomes zero.

Work around

When executing multiple DISI instructions within the source code, make sure that subsequent DISI instructions have at least one instruction cycle between the time that the DISI counter decrements to zero and the next DISI instruction. Alternatively, make sure that subsequent DISI instructions are called before the DISI counter decrements to zero.

Affected Silicon Revisions

A1							
X							

5. Module: Interrupt Controller

The following sequence of events will lead to an address error trap. The generic term “Interrupt 1” is used to represent any enabled dsPIC30F interrupt.

1. User software performs one of the following operations:
   - CPU IPL is raised to Interrupt 1 IPL level or higher, or
   - Interrupt 1 IPL is lowered to CPU IPL level or lower, or
   - Interrupt 1 is disabled (Interrupt 1 IE bit set to ‘0’), or
   - Interrupt 1 flag is cleared
2. Interrupt 1 occurs between 2 and 4 instruction cycles after any of the operations listed above.

Work arounds

Work around 1: For Assembly Language Source Code

The user may disable interrupt nesting, disable interrupts before modifying the Interrupt 1 setting or execute a DISI instruction before modifying the CPU IPL or Interrupt 1. A minimum DISI value of 4 is required if the DISI instruction is executed immediately before the CPU IPL or Interrupt 1 is modified, as shown in Example 2. It is necessary to have DISI active for four cycles after the CPU IPL or Interrupt 1 is modified.

EXAMPLE 2: USING DISI

```
.include "p30fxxxx.inc"
...
DISI #4 ; protect the disable
; of INT1
BCLR IEC1, #INT1IE ; disable interrupt 1
... ; next instruction
;protected by DISI
```

Work around 2: For C Language Source Code

For applications using the C language, MPLAB C30 versions 1.32 and higher provide several macros for modifying the CPU IPL. The SET_CPU_IPL macro provides the ability to safely modify the CPU IPL, as shown in Example 3.

EXAMPLE 3: USING SET_CPU_IPL MACRO

```
// Note: Macro defined in device include
// files
#define SET_CPU_IPL (ipl){ \
int DISI_save; \
\
DISI_save = DISICNT; \
asm volatile ("disi #0x3FFF");\
SRbits.IPL = ipl; \
__builtin_nop();   \
__builtin_nop();   \
DISICNT = DISI_save; } (void) 0;

#include "p30fxxxx.h"
. . .
SET_CPU_IPL (3)
. . .
```

There is one level of DISI, so this macro saves and restores the DISI state. For temporarily modifying and restoring the CPU IPL, the macros SET_AND_SAVE_CPU_IPL and RESTORE_CPU_IPL can be used, as shown in Example 4. These macros also make use of the SET_CPU_IPL macro.

EXAMPLE 4: USING SET_AND_SAVE_CPU_IPL AND RESTORE_CPU_IPL MACROS

```
// Note: Macros defined in device include files
#define SET_AND_SAVE_CPU_IPL (save_to, ipl){ \
save_to = SRbits.IPL; \
SET_CPU_IPL (ipl); } (void) 0;

#define RESTORE_CPU_IPL (saved_to) SET_CPU_IPL (saved_to)

#include "p30fxxxx.h"
. . .
int save_to;
SET_AND_SAVE_CPU_IPL (save_to, 3)
. . .
RESTORE_CPU_IPL (save_to)
```

For modification of the Interrupt 1 setting, the INTERRUPT_PROTECT macro can be used. This macro disables interrupts before executing the desired expression, as shown in Example 5. This macro is not distributed with the compiler.

EXAMPLE 5: USING INTERRUPT_PROTECT MACRO

```
#define INTERRUPT_PROTECT (x) { \
int save_sr; \
SET_AND_SAVE_CPU_IPL (save_sr, 7);\
x; \
RESTORE_CPU_IPL (save_sr); } (void) 0;

. . .
INTERRUPT_PROTECT (IEC0bits.U1TXIE=0);
```

Note:	If you are using a MPLAB C30 compiler version earlier than version 1.32, you may still use the macros by adding them to your application.

Affected Silicon Revisions

A1							
X							

6. Module: Output Compare

A glitch will be produced on an output compare pin under the following conditions:

- The user software initially drives the I/O pin high using the Output Compare module or a write to the associated PORT register.
- The Output Compare module is configured and enabled to drive the pin low at some point in later time (OCxCON = 0x0002 or OCxCON = 0x0003).

When these events occur, the Output Compare module will drive the pin low for one instruction cycle (TCY) after the module is enabled.

Work around

None. However, the user may use a timer interrupt and write to the associated PORT register to control the pin manually.

Affected Silicon Revisions

A1							
X							

7. Module: Output Compare

If the desired duty cycle is '0' (OCxRS = 0), the module will generate a high level glitch of 1 TCY. The second problem is that on the next cycle after the glitch, the OC pin does not go high, or in other words, it misses the next compare for any value written on OCxRS.

Work around

There are two possible solutions to this problem:

1. Load a value greater than '0' to the OCxRS register when operating in PWM mode. In this case, no 0% duty cycle is achievable.
2. If the application requires 0% duty cycles, the Output Compare module can be disabled for 0% duty cycles, and re-enabled for non-zero percent duty cycles.

Affected Silicon Revisions

A1							
X							

8. Module: ADC

ADC event triggers from the INT0 pin will not wake-up the device from Sleep mode if the SMPI bits are non-zero. This means that if the ADC is configured to generate an interrupt after a certain number of INT0 triggered conversions, the ADC conversions will not be triggered and the device will remain in Sleep. The ADC will perform conversions and wake-up the device only if it is configured to generate an interrupt after each INT0 triggered conversion (SMPI<3:0> = 0000).

Work around

None. If ADC event trigger from the INT0 pin is required, initialize SMPI<3:0> to '0000' (interrupt on every conversion).

Affected Silicon Revisions

A1							
X							

9. Module: PLL

If 4x or 8x PLL mode is used, the input frequency range is 5 MHz-10 MHz instead of 4 MHz-10 MHz.

Work around

None. If 4x or 8x PLL is used, make sure the input crystal or clock frequency is 5 MHz or greater.

Affected Silicon Revisions

A1							
X							

10. Module: Sleep Mode

Execution of the Sleep instruction (PWRSAV #0) may cause incorrect program operation after the device wakes up from Sleep. The current consumption during Sleep may also increase beyond the specifications listed in the device data sheet.

Work arounds

To avoid this issue, implement any of the following three work arounds, depending on the application requirements.

Work around 1:

Ensure that the PWRSAV #0 instruction is located at the end of the last row of program Flash memory available on the target device and fill the remainder of the row with NOP instructions.

This can be accomplished by replacing all occurrences of the PWRSAV #0 instruction with a function call to a suitably aligned subroutine. The address( ) attribute provided by the MPLAB ASM30 assembler can be utilized to correctly align the instructions in the subroutine. For an application written in C, the function call would be GotoSleep( ), while for an assembly language application, the function call would be CALL _GotoSleep.

The address error Trap Service Routine (TSR) software can then replace the invalid return address saved on the stack with the address of the instruction immediately following the _GotoSleep or GotoSleep( ) function call. This ensures that the device continues executing the correct code sequence after waking up from Sleep mode.

Example 6 demonstrates the work around described above.

EXAMPLE 6:

```
; ------------------------------------------------------------------------------------------------
.global __reset
.global _main
.global _GotoSleep
.global __AddressError
.global __INT1Interrupt
; ------------------------------------------------------------------------------------------------
    .section *, code
_main:
    BSET    INTCON2, #INT1EP  ; Set up INT pins to detect falling edge
    BCLR    IFS1, #INT1IF     ; Clear interrupt pin interrupt flag bits
    BSET    IEC1, #INT1IE     ; Enable ISR processing for INT pins
    CALL    _GotoSleep        ; Call function to enter SLEEP mode
_continue:
    BRA  _continue
; ------------------------------------------------------------------------------------------------
; Address Error Trap
__AddressError:
    BCLR    INTCON1, #ADDRERR
    ; Set program memory return address to _continue
    POP.D   W0
    MOV.B   #tblpage (_continue), W1
    MOV     #tbloffset (_continue), W0
    PUSH.D  W0
    RETFIE
; ------------------------------------------------------------------------------------------------
__INT1Interrupt:
    BCLR    IFS1, #INT1IF         ; Ensure flag is reset
    RETFIE                        ; Return from Interrupt Service Routine
; ------------------------------------------------------------------------------------------------
    .section *, code, address (0x1FC0)
_GotoSleep:
; fill remainder of the last row with NOP instructions
    .rept 31
        NOP
    .endr
; Place SLEEP instruction in the last word of program memory
    PWRSAV #0
```

Work around 2:

Instead of executing a `PWRSAV #0` instruction to put the device into Sleep mode, perform a clock switch to the 512 kHz Low-Power RC (LPRC) Oscillator with a 64:1 postscaler mode. This enables the device to operate at 0.002 MIPS, thereby significantly reducing the current consumption of the device. Similarly, instead of using an interrupt to wake-up the device from Sleep mode, perform another clock switch back to the original oscillator source to resume normal operation. Depending on the device, refer to **Section 7. "Oscillator"** (DS70054) or **Section 29. "Oscillator"** (DS70268) in the *"dsPIC30F Family Reference Manual"* (DS70046) for more details on performing a clock switch operation.

Note:	The above work around is recommended for users for whom application hardware changes are not possible.

Work around 3:

Instead of executing a `PWRSAV #0` instruction to put the device into Sleep mode, perform a clock switch to the 32 kHz Low-Power (LP) Oscillator with a 64:1 postscaler mode. This enables the device to operate at 0.000125 MIPS, thereby significantly reducing the current consumption of the device. Similarly, instead of using an interrupt to wake-up the device from Sleep mode, perform another clock switch back to the original oscillator source to resume normal operation. Depending on the device, refer to **Section 7. "Oscillator"** (DS70054) or **Section 29. "Oscillator"** (DS70268) in the *"dsPIC30F Family Reference Manual*" (DS70046) for more details on performing a clock switch operation.

Note:	The above work around is recommended for users for whom application hardware changes are possible, and also for users whose application hardware already includes a 32 kHz LP Oscillator crystal.

Affected Silicon Revisions

A1							
X							

11. Module: Timer

When the timer is being operated in Asynchronous mode using the secondary oscillator (32.768 kHz) and the device is put into Sleep mode, a clock switch to any other oscillator mode before putting the device to Sleep prevents the timer from waking the device from Sleep.

Work around

Do not clock switch to any other oscillator mode if the timer is being used in Asynchronous mode using the secondary oscillator (32.768 kHz).

Affected Silicon Revisions

A1							
X							

12. Module: PLL

The PLL LOCK Status bit (OSCCON<5>) can occasionally get cleared and generate an oscillator failure trap even when the PLL is still locked and functioning correctly.

Work around

The user application must include an oscillator failure trap service routine. In the trap service routine, first inspect the status of the Clock Failure Status bit (OSCCON<3>). If this bit is clear, return from the trap service routine immediately and continue program execution.

Affected Silicon Revisions

A1							
X							

13. Module: PSV Operations

An address error trap occurs in certain addressing modes when accessing the first four bytes of an Program Space Visibility (PSV) page. This only occurs when using the following addressing modes:

- MOV.D
- Register Indirect Addressing (word or byte mode) with pre/post-decrement

Work around

Do not perform PSV accesses to any of the first four bytes using the above addressing modes. For applications using the C language, MPLAB C30 version 3.11 or higher, provides the following command-line switch that implements a work around for the erratum.

```
-merrata=psv_trap
```

Refer to the `readme.txt` file in the MPLAB C30 v3.11 tool suite for further details.

Affected Silicon Revisions

A1							
X							

14. Module: I/O

If the user application enables the auto-baud feature in the UART module, the I/O pin multiplexed with the IC1 (Input Capture) pin cannot be used as a digital input. However, the external interrupt function (INT1) can be used.

Work around

None.

Affected Silicon Revisions

A1							
X							

15. Module: I²C

When the I²C module is configured as a slave, either in single-master or multi-master mode, the I²C receiver buffer is filled whether a valid slave address is detected or not. Therefore, an I²C receiver overflow condition occurs and this condition is indicated by the I2COV flag in the I2CSTAT register.

This overflow condition inhibits the ability to set the I²C receive interrupt flag (SI2CF) when the last valid data byte is received. Therefore, the I²C slave Interrupt Service Routine (ISR) is not called and the I²C receiver buffer is not read prior receiving the next data byte.

Work arounds

To avoid this issue, either of the following two work arounds can be implemented, depending on the application requirements.

Work around 1:

For applications in which the I²C receiver interrupt is not required, the following procedure can be used to receive valid data bytes:

1. Wait until the RBF flag is set.
2. Poll the I²C receiver interrupt SI2CIF flag.
3. If SI2CF is not set in the corresponding Interrupt Flag Status register (IFSx), a valid address or data byte has not been received for the current slave. Execute a dummy read of the I²C receiver buffer, I2CRCV; this will clear the RBF flag. Go back to step 1 until SI2CF is set and then continue to Step 4.
4. If the SI2CF is set in the corresponding Interrupt Flag Status register (IFSx), valid data has been received. Check the D_A flag to verify that an address or a data byte has been received.
5. Read the I2CRCV buffer to recover valid data bytes. This will also clear the RBF flag.
6. Clear the I²C receiver interrupt flag SI2CF.
7. Go back to step 1 to continue receiving incoming data bytes.

Work around 2:

Use this work around for applications in which the I²C receiver interrupt is required. Assuming that the RBF and the I2COV flags in the I2CSTAT register are set due to previous data transfers in the I²C bus (i.e., between master and other slaves); the following procedure can be used to receive valid data bytes:

1. When a valid slave address byte is detected, SI2CF bit is set and the I²C slave interrupt service routine is called; however, the RBF and I2COV bits are already set due to data transfers between other I²C nodes.
2. Check the status of the D_A flag and the I2COV flag in the I2CSTAT register when executing the I²C slave service routine.
3. If the D_A flag is cleared and the I2COV flag are set, an invalid data byte was received but a valid address byte was received. The overflow condition occurred because the I²C receive buffer was overflowing with previous I²C data transfers between other I²C nodes. This condition only occurs after a valid slave address was detected.
4. Clear the I2COV flag and perform a dummy read of the I²C receiver buffer, I2CRCV, to clear the RBF bit and recover the valid address byte. This action will also avoid the loss of the next data byte due to an overflow condition.
5. Verify that the recovered address byte matches the current slave address byte. If they match, the next data to be received is a valid data byte.
6. If the D_A flag and the I2COV flag are both set, a valid data byte was received and a previous valid data byte was lost. It will be necessary to code for handling this overflow condition.

Affected Silicon Revisions

A1							
X							

16. Module: I²C

If there are two I²C devices on the bus, one of them is acting as the Master receiver and the other as the Slave transmitter. If both devices are configured for 10-bit addressing mode, and have the same value in the A10 and A9 bits of their addresses, then when the Slave select address is sent from the Master, both the Master and Slave acknowledge it. When the Master sends out the read operation, both the Master and the Slave enter into Read mode and both of them transmit the data. The resultant data will be the ANDing of the two transmissions.

Work around

In all I²C devices, the addresses as well as bits A10 and A9 should be different.

Affected Silicon Revisions

A1							
X							

17. Module: I²C

In 10-bit Addressing mode, some address matches do not set the RBF flag or load the receive register I2CxRCV, if the lower address byte matches the reserved addresses. In particular, these include all addresses with the form XX0000XXXX and XX1111XXXX, with the following exceptions:

- 001111000X
- 011111001X
- 101111010X
- 111111011X

Work around

Ensure that the lower address byte in 10-bit Addressing mode does not match any 7-bit reserved addresses.

Affected Silicon Revisions

A1							
X							

18. Module: I²C

When the I²C module is configured as a 10-bit slave with and address of 0x102, the I2CxRCV register content for the lower address byte is 0x01 rather than 0x02; however, the module acknowledges both address bytes.

Work around

None.

Affected Silicon Revisions

A1							
X							

19. Module: I²C

When the I²C module is enabled by setting the I2CEN bit in the I2CCON register, the dsPIC DSC device generates a glitch on the SDA and SCL pins. This glitch falsely indicates "Communication Start" to all devices on the I²C bus, and can cause a bus collision in a multi-master configuration.

Additionally, when the I2CEN bit is set, the S and P bits of the I²C module are set to values '1' and '0', respectively, which indicate a "Communication Start" condition.

Work arounds

To avoid this issue, either of the following two work arounds can be implemented, depending on the application requirements.

Work around 1:

In a single-master environment, add a delay between enabling the I²C module and the first data transmission. The delay should be equal to or greater than the time it takes to transmit two data bits.

In the multi-master configuration, in addition to the delay, all other I²C masters should be synchronized and wait for the I²C module to be initialized before initiating any kind of communication.

Work around 2:

In dsPIC DSC devices in which the I²C module is multiplexed with other modules that have precedence in the use of the pin, it is possible to avoid this glitch by enabling the higher priority module before enabling the I²C module.

Use the following procedure to implement this work around:

1. Enable the higher priority peripheral module that is multiplexed on the same pins as the I²C module.
2. Set up and enable the I²C module.
3. Disable the higher priority peripheral module that was enabled in step 1.

Note: Work around 2 works only for devices that share the SDA and SCL pins with another peripheral that has a higher precedence over the port latch, such as the UART. The priority is shown in the pin diagram located in the data sheet. For example, if the SDA and SCL pins are shared with the UART and SPI pins, and the UART has higher precedence on the port latch pin.

Affected Silicon Revisions

A1							
X							

20. Module: ADC

If the ADC module is in an enabled state when the device enters Sleep mode as a result of executing a PWRSAV #0 instruction, the device power-down current (IPD) may exceed the specifications listed in the device data sheet. This may happen even if the ADC module is disabled by clearing the ADON bit prior to entering Sleep mode.

Work around

In order to remain within the IPD specifications listed in the device data sheet, the user software must completely disable the ADC module by setting the ADC Module Disable bit in the corresponding Peripheral Module Disable (PMDx) register, prior to executing a PWRSAV #0 instruction.

Affected Silicon Revisions

A1							
X							

Data Sheet Clarifications

The following typographic corrections and clarifications are to be noted for the latest version of the device data sheet (DS70139**F**):

Note:	Corrections are shown in **bold**. Where possible, the original bold text formatting has been removed for clarity.

1. Module: DC Characteristics: I/O Pin Input Specifications

The maximum value for parameter DI19 (V_{IL} specifications for SDAx and SCLx pins) and the minimum value for parameter DI29 (V_{IH} specifications for SDAx and SCLx pins) were stated incorrectly in Table 20-8 of the current device data sheet. The correct values are shown in bold type in Table 3.

TABLE 3: DC CHARACTERISTICS: I/O PIN INPUT SPECIFICATIONS

DC CHARACTERISTICS			Standard Operating Conditions: 3.3V and 5.0V (±10%) (unless otherwise stated) Operating temperature -40°C ≤TA ≤+85°C for Industrial -40°C ≤TA ≤+125°C for Extended				
Param No.	Symbol	Characteristic	Min	Typ	Max	Units	Conditions
DI19	V_{IL}	Input Low Voltage SDA, SCL	V_{SS}	—	**0.8**	V	SMbus enabled
DI29	V_{IH}	Input High Voltage SDA, SCL	**2.1**	—	V_{DD}	V	SMbus enabled

APPENDIX A: REVISION HISTORY

Rev A Document (4/2009)

Initial release of this document; issued for revision A1.

Includes silicon issues 1 (EMUC2 Pin), 2-4 (CPU), 5 (Interrupt Controller), 6-7 (Output Compare), 8 (ADC), 9 (PLL), 10 (Sleep Mode), 11 (Timer), 12 (PLL), 13 (PSV Operations), 14 (I/O) and 15-19 (I^2C).

This document replaces the following errata document:

- DS80273, *"dsPIC30F2011/2012 Rev. A1 Silicon Errata"*

Rev B Document (8/2009)

Removed silicon issue 1 (EMUC2 Pin). The numbering for existing issues does not change.

Updated silicon issue 5 (Interrupt Controller).

Rev C Document (2/2010)

Updated silicon issue 5 (Interrupt Controller).

Rev D Document (6/2010)

Added silicon issue 20 (ADC) and data sheet clarification 1 (DC Characteristics: I/O Pin Input Specifications).

NOTES:

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Trademarks

ISBN: 978-1-60932-251-9

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2002

Microchip received ISO/TS-16949:2002 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://support.microchip.com
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo
Kokomo, IN
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-6578-300
Fax: 886-3-6578-370

Taiwan - Kaohsiung
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

01/05/10