dsPIC30F Family Reference Manual Errata Sheet

The dsPIC30F Family Reference Manual (DS70046B) correctly describes the function of the dsPIC30F devices, except for the anomalies described below.

All of the issues listed will be addressed in future releases of the Family Reference Manual.

Errata Summary

The following list summarizes the errata described in further detail through the remainder of this document:

1. Typographical errors in several register descriptions, bit descriptions, and source code examples have been corrected.
2. The conditions leading to a Stack Error Trap have been clarified.
3. Restrictions on the instructions that can be used at or near the end of a DO loop have been clarified.
4. The timing for Program Space Visibility (PSV) operations has been clarified and the effect of instruction stalls on PSV has been corrected.
5. The function of the NVMADRU register has been documented.
6. Run-Time Self Programming (RTSP) can be used to program 32 instruction locations at one time, not 4 instructions as the current documentation indicates.
7. Causes of Address Error Traps have been clarified.
8. Control bits for tuning the 8 MHz RC oscillator have been documented.
9. Clock switching operation has been clarified.
10. Differences in QEI DFLTCON register among device variants has been documented.
11. The 10-bit A/D sampling requirements have been changed.
12. The 12-bit A/D sampling requirements have been changed.
13. Wake-up operation from Sleep and Idle modes has been clarified.

1. Page 2-2, Section 2.1 Introduction

On page 2-2, Section 2.1 Introduction, paragraph 5 should be replaced with the following:

> The upper 32 Kbytes of the data space memory map can optionally be mapped into program space at any 16K program word boundary defined by the 8-bit Program Space Visibility Page (PSVPAG) register. The program to data space mapping feature lets any instruction access program space as if it were data space. Furthermore, RAM may be connected to the program memory bus **on devices with an external bus** and used to extend the internal data RAM.

2. Page 2-10, Section 2.3.3 Stack Pointer Overflow

On page 2-10, Section 2.3.3 Stack Pointer Overflow, the last sentence in paragraph 2 should be replaced by the following:

> If the contents of the Stack Pointer (W15) are greater than the contents of the SPLIM register by 2 and a push operation is performed, a Stack Error Trap will not occur. The Stack Error Trap will occur on a subsequent push operation. Thus, for example, if it is desirable to cause a Stack Error Trap when the stack grows beyond address `0x2000` in RAM, initialize the SPLIM with the value, `0x1FFE`.

> **Note:** A Stack Error Trap may be caused by any instruction that uses the contents of the W15 register to generate an effective address (EA). Thus, if the contents of W15 are greater than the contents of the SPLIM register by 2, and a CALL instruction is executed, or if an interrupt occurs, a Stack Error Trap will be generated.

3. Page 2-34, Section 2.9.2.5 DO Loop Restrictions

On page 2-34, Section 2.9.2.5 DO Loop Restrictions, the following paragraph and bullets should be added to the end of this section:

> The instruction that is executed two instructions before the last instruction in a `DO` loop should not modify any of the following:
>
> - CPU priority level governed by the IPL (SR<7:5>) bits
> - Peripheral Interrupt Enable bits governed by the IEC0, IEC1 and IEC2 registers
> - Peripheral Interrupt Priority bits governed by the IPC0 through IPC11 registers
>
> If the restrictions above are not followed, the `DO` loop may execute incorrectly.

4. Page 2-34, Section 2.9.2.5.1 Last Instruction Restrictions

On page 2-34, Section 2.9.2.5.1 Last Instruction Restrictions, the following bullet should be added to the end of the bulleted list:

> 6. `DISI` instruction

5. Page 2-37, Section 2.10.2.4 Instruction Stalls and Program Space Visibility

On page 2-37, Section 2.10.2.4 Instruction Stalls and Program Space Visibility should be replaced with the following:

2.10.2.4 Instruction Stalls and Program Space Visibility (PSV)

When program space (PS) is mapped to data space by enabling the PSV (CORCON<2>) bit, and the X space EA falls within the visible program space window, the read or write cycle is redirected to the address in program space. Accessing data from program space takes up to 3 instruction cycles.

Instructions operating in PSV address space are subject to RAW data dependencies and consequent instruction stalls, just like any other instruction.

Consider the following code segment:

```
ADD W0,[W1],[W2++]; PSV = 1, W1=0x8000, PSVPAG=0xAA
MOV [W2],[W3]
```

This sequence of instructions would take 5 instruction cycles to execute. 2 instruction cycles are added to perform the PSV access via W1. Furthermore, an instruction stall cycle is inserted to resolve the RAW data dependency caused by W2.

6. Page 2-39, Table 2-8 dsPIC30F Core Register Map (Continued)

On page 2-39, Table 2-8 dsPIC30F Core Register Map (Continued), in the first row of the table, correct bits names for bit 8 through bit 12 in the CORCON register as follows:

TABLE 1: CORCON BIT NAMES

Bit Location in CORCON	Bit Name (Incorrect)	Bit Name (Please read as)
12	-	US
11	US	EDT
10	EDT	DL2

7. Page 3-19, Register 3-1: MODCON: Modulo and Bit-Reversed Addressing Control Register, bit 7-4

On page 3-19, Register 3-1: MODCON: Modulo and Bit-Reversed Addressing Control Register, the description for bit4 through bit7 should be modified as follows. All other bit definitions do not change and hence are not described:

Register 3-1: MODCON: Modulo and Bit-Reversed Addressing Control Register

Upper Byte:							
R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
XMODEN	YMODEN	—	—	BWM<3:0>			
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
YWM<3:0>				XWM<3:0>			
bit 7							bit 0

bit 7-4 **YWM<3:0>:** Y AGU W Register Select for Modulo Addressing bits

`1111` = Modulo addressing disabled
`1110` = W14 selected for modulo addressing
`1101` = W13 selected for **modulo** addressing
 ¦ ¦
`0000` = W0 selected for modulo addressing

Note: A write to the MODCON register should not be followed by an instruction that performs an indirect read operation using a W register. Unexpected results may occur. Some instructions perform an implicit indirect read. These are: `POP`, `RETURN`, `RETFIE`, `RETLW` and `ULNK`.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

8. Page 4-10, Section 4.4.3 PSV Timing

On page 4-10, Section 4.4.3 PSV Timing should be replaced with the following:

4.4.3 PSV Timing

Instructions that use PSV will require two extra instruction cycles to complete execution, except the following instructions that require only one extra cycle to complete execution:

-The `MAC` class of instructions with data pre-fetch operands

-All `MOV` instructions including the `MOV.D` instruction

The additional instruction cycles are used to fetch the PSV data on the program memory bus.

4.4.3.1 Using PSV in a REPEAT Loop

Instructions that use PSV within a `REPEAT` loop eliminate the extra instruction cycle(s) required for the data access from program memory, hence incurring no overhead in execution time. However, the following iterations of the `REPEAT` loop will incur an overhead of two instruction cycles to complete execution:

-The first iteration

-The last iteration

-Instruction execution prior to exiting the loop due to an interrupt

-Instruction execution upon re-entering the loop after an interrupt is serviced

4.4.3.2 PSV and Instruction Stalls

Refer to **Item 5 in this FRM Errata Document** for more information about instruction stalls using PSV.

9. Page 5-5, Table 5-1 NVMCON Register Values

On page 5-5, Table 5-1 NVMCON Register Values should be replaced with the following:

Table 5-1: NVMCON Register Values

NVMCON Register Values for RTSP Program and Erase Operations			
Memory Type	**Operation**	**Data Size**	**NVMCON Value**
Flash PM	Erase	1 row (32 instr. words)	`0x4041`
	Program	**1 row (32 instr. words)**	**`0x4001`**
Data EEPROM	Erase	1 data word	`0x4044`
		16 data words	`0x4045`
	Program	1 data word	`0x4004`
		16 data words	`0x4005`
Configuration Register	Write**(see Note)**	1 config. register	`0x4008`

Note: The device configuration registers may be written to a new value without performing an erase cycle.

10. Page 5-6, Section 5.3.2 NVMADR Register

On page 5-6, Section 5.3.2 NVMADR Register should be replaced with the following:

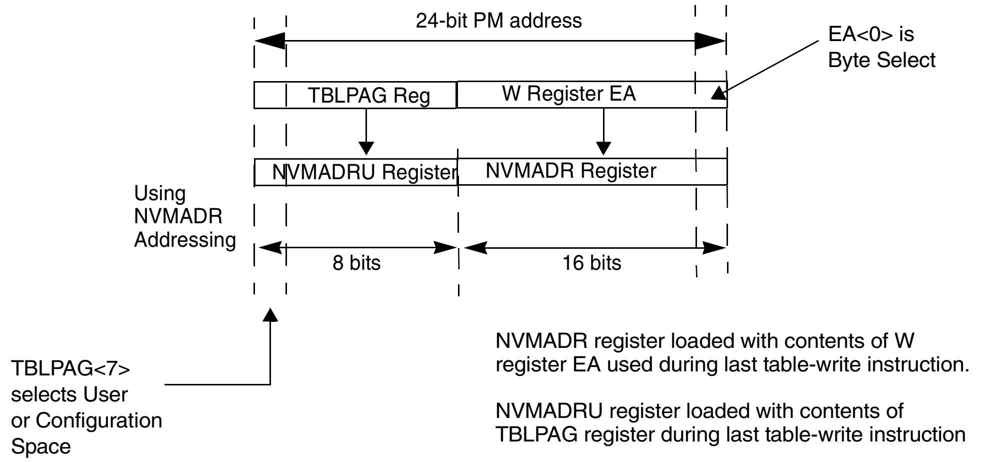
5.3.2 NVM Address Registers

There are two NVM Address Registers - NVMADRU and NVMADR. These two registers when concatenated form the 24-bit effective address (EA) of the selected row or word for programming operations. The NVMADRU register is used to hold the upper 8 bits of the EA, while the NVMADR register is used to hold the lower 16 bits of the EA.

The register pair, NVMADRU:NVMADR, capture the EA<23:0> of the last table-write instruction that has been executed and select the row of Flash or EEPROM memory to write/erase. Figure 5-2 shows how the program memory EA is formed for programming and erase operations.

Although the NVMADRU and NVMADR registers are automatically loaded by the table-write instructions, the user can also directly modify their contents before the programming operation begins. A write to these registers will be required prior to an erase operation, because no table-write instructions are required for any erase operation.

Figure 5-2: NVM Addressing with TBLPAG and NVM Address Registers


24-bit PM address
EA<0> is Byte Select
TBLPAG Reg
W Register EA
NVMADRU Register
NVMADR Register
Using NVMADR Addressing
8 bits
16 bits
TBLPAG<7> selects User or Configuration Space
NVMADR register loaded with contents of W register EA used during last table-write instruction.
NVMADRU register loaded with contents of TBLPAG register during last table-write instruction

11. Page 5-7, Register 5-1 NVMCON

On page 5-7, Register 5-1: NVMCON: Non-Volatile Memory Control Register, the definition of the WREN and PROGOP bits should be modified as follows. All other bit definitions do not change and hence are not described here.

Register 5-1: NVMCON: Non-Volatile Memory Control Register

Upper Byte:							
R/S-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
WR	WREN	WRERR	—	—	—	—	—
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PROGOP<7:0>							
bit 7							bit 0

bit 14 **WREN:** Write (Erase or Program) Enable bit
`1` = Enable an erase or program operation
`0` = No operation allowed **(Device clears this bit on completion of the write/erase operation)**

bit 7-0 **PROGOP<7:0>:** Programming Operation Command Byte bits

Erase Operations:
`0x41` = Erase 1 row (32 instruction words) from 1 panel of program Flash
`0x44` = Erase 1 data word from data Flash
`0x45` = Erase 1 row (16 data words) from data Flash

Programming Operations:
`0x01` = Program 1 row (32 instruction words) into Flash program memory
`0x04` = Program 1 data word into data EEPROM
`0x05` = Program 1 row (16 data words) into data EEPROM
`0x08` = Program 1 data word into device configuration register

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
S = Settable bit	-n = Value at POR	‘1’ = Bit is set
‘0’ = Bit is cleared	x = Bit is unknown	

12. Page 5-8, Register 5-2: NVMADR: Non-Volatile Memory Address Register, bit 15-0

On page 5-8, Register 5-2: NVMADR: Non-Volatile Memory Address Register, bit 15-0 should be replaced with the following**:** :

Upper Byte:							
R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
NVMADR<15:8>							
bit 15							bit 8

Lower Byte:							
R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
NVMADR<7:0>							
bit 7							bit 0

bit 15-0 **NVMADR<15:0>:** NV Memory Write Address bits
Selects the location to program **or erase** in program or data Flash memory.
This register may be read or written by user. This register will contain the address of EA<15:0> of the last table write instruction executed, until written by the user.

Note: The NVMADRU register function is similar to the NVMADR register and holds the upper 8 bits of the location to be programmed or erased. The value of the TBLPAG register is automatically loaded into the NVMADRU register during a table write instruction.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

13. Page 5-10, Section 5.4.2.1 Flash Program Memory Programming Algorithm

On page 5-10, Section 5.4.2.1 Flash Program Memory Programming Algorithm should be replaced with the following:

5.4.2.1 Flash Program Memory Programming Algorithm

The user can erase and program Flash Program Memory by rows (32 instruction words). The general process is as follows:

1. Read one row of program Flash (32 instruction words) and store into data RAM as a data “image”. The RAM image must be read from an even 32-word program memory address boundary.
2. Update the RAM data image with the new program memory data.
3. Erase program Flash row.
   - Setup NVMCON register to erase 1 row of Flash program memory.
   - Write address of row to be erased into NVMADRU and NVMADR registers.
   - Disable interrupts.
   - Write the key sequence to NVMKEY to enable the erase.
   - Set the WR bit. This will begin erase cycle.
   - CPU will stall for the duration of the erase cycle.
   - The WR bit is cleared when erase cycle ends.
   - Re-enable interrupts.
4. Write 32 instruction words of data from RAM into the Flash program memory write latches.
5. Program 32 instruction words into program Flash.
   - Setup NVMCON to program one row of Flash program memory.
   - Disable interrupts.
   - Write the key sequence to NVMKEY to enable the program cycle.
   - Set the WR bit. This will begin the program cycle.
   - CPU will stall for duration of the program cycle.
   - The WR bit is cleared by the hardware when program cycle ends.
   - Re-enable interrupts.
6. Repeat steps 1 through 6, as needed, to program the desired amount of Flash program memory

> **Note:** The user should remember that the minimum amount of program memory that can be modified using RTSP is 32 instruction word locations. Therefore, it is important that an image of these locations be stored in general purpose RAM before an erase cycle is initiated. An erase cycle must be performed on any previously written locations before any programming is done.

14. Page 5-11, Section 5.4.2.2 Erasing a Row of Program Memory

On page 5-11, Section 5.4.2.2 Erasing a Row of Program Memory should be replaced with the following:

5.4.2.2 Erasing a Row of Program Memory

The following is a code sequence that can be used to erase a row (32 instructions) of program memory. The NVMCON register is configured to erase one row of program memory. The NVMADRU and NVMADR registers are loaded with the address of the row to be erased. The program memory must be erased at 'even' row boundaries. Therefore, the 6 LSbits of the value written to the NVMADR register have no effect when a row is erased.

The erase operation is initiated by writing a special unlock, or key sequence to the NVMKEY register before setting the WR control bit (NVMCON<15>). The unlock sequence needs to be executed in the exact order shown without interruption. Therefore, interrupts should be disabled prior to writing the sequence.

Two `NOP` instructions should be inserted in the code at the point where the CPU will resume operation. Finally, interrupts can be enabled (if required).

```
; Setup NVMCON to erase one row of Flash program memory
    MOV    #0x4041,W0
    MOV    W0,NVMCON
; Setup address pointer to row to be ERASED
    MOV    #tblpage(PROG_ADDR),W0
    MOV    W0,NVMADRU
    MOV    #tbloffset(PROG_ADDR),W0
    MOV    W0,NVMADR
; Disable interrupts, if enabled
    PUSH   SR
    MOV    #0x00E0,W0
    IOR    SR
; Write the KEY sequence
    MOV        #0x55,W0
    MOV W0,    NVMKEY
    MOV #0xAA, W0
    MOV W0,    NVMKEY
; Start the erase operation
    BSET   NVMCON,#WR
; Insert two NOPs after the erase cycle (required)
    NOP
    NOP
; Re-enable interrupts, if needed
    POP    SR
```

Note: When erasing a row of program memory, the user writes the upper 8 bits of the erase address directly to the NVMADRU and NVMADR registers. Together, the contents of the NVMADRU and NVMADR registers form the complete address of the program memory row to be erased.

The NVMADRU and NVMADR registers specify the address for all Flash erase and program operations. However, these two registers do not have to be directly written by the user for Flash program operations. This is because the table write instructions used to write the program memory data automatically transfers the TBLPAG register contents and the table write address into the NVMADRU and NVMADR registers.

The above code example could be modified to perform a 'dummy' table write operation to capture the program memory erase address.

15. Page 5-12, Section 5.4.2.3 Loading Write Latches

On page 5-12, Section 5.4.2.3 Loading Write Latches should be replaced with the following:

5.4.2.3 Loading Write Latches

The following is a sequence of instructions that can be used to load the 768-bits of write latches (32 instruction words). Four `TBLWTL` and four `TBLWTH` instructions are needed to load the write latches selected by the table pointer.

The TBLPAG register is loaded with the 8 MSbits of the program memory address. The user does not need to write the NVMADRU:NVMADR register-pair for a Flash programming operation. The 24-bits of the program memory address are automatically captured into the NVMADRU:NVMADR register-pair when each table write instruction is executed. The program memory must be programmed at an ‘even’ 32 instruction word address boundary. In effect, the 6 LSbits of the value captured in the NVMADR register are not used during the programming operation.

The row of 32 instruction words do not necessarily have to be written in sequential order. The 6 LSbits of the table write address determine which of the latches will be written. However, all 32 instruction words should be written for each programming cycle to overwrite old data.

> **Note:** The following code example is the ‘Load_Write_Latch’ code referred to in subquent examples.

```
; Set up a pointer to the first program memory location to be written.
MOV     #tblpage(PROG_ADDR),W0
MOV     W0,TBLPAG
MOV     #tbloffset(PROG_ADDR),W0

; Perform the TBLWT instructions to write the latches
; W0 is incremented in the TBLWTH instruction to point to the
; next instruction location.

MOV     #LOW_WORD_0,W2
MOV     #HIGH_BYTE_0,W3
TBLWTL  W2,[W0]
TBLWTH  W3,[W0++]       ; 1st_program_word
MOV     #LOW_WORD_1,W2
MOV     #HIGH_BYTE_1,W3
TBLWTL  W2,[W0]
TBLWTH  W3,[W0++]       ; 2nd_program_word
MOV     #LOW_WORD_2,W2
MOV     #HIGH_BYTE_2,W3
TBLWTL  W2, [W0]
TBLWTH  W3, [W0++]      ; 3rd_program_word
MOV     #LOW_WORD_3,W2
MOV     #HIGH_BYTE_3,W3
TBLWTL  W2,[W0]
TBLWTH  W3,[W0++]       ; 4th_program_word
........
........
MOV     #LOW_WORD_31,W2
MOV     #HIGH_BYTE_31,W3
TBLWTL  W2,[W0]
TBLWTH  W3,[W0++]       ; 32nd_program_word
```

16. Page 6-6, Section 6.2 Non-Maskable Traps

On page 6-6, Section 6.2 Non-Maskable Traps, the second paragraph should be replaced with the following:

> The dsPIC30F has **four** implemented sources of non-maskable traps:
>
> - Oscillator Failure Trap
> - Stack Error Trap
> - Address Error Trap
> - Arithmetic Error Trap

17. Page 6-7, Section 6.2.2 Hard Traps

On page 6-7, Section 6.2.2 Hard Traps, paragraph 1 should be replaced with the following:

> Hard traps include exceptions of priority level 13 through level 15, inclusive. The **address** error (level 13) and oscillator error (level 14) traps fall into this category.

18. Page 6-8, Section 6.2.2.3 Address Error Trap (Hard Trap, Level 13)

On page 6-8, Section 6.2.2.3 Address Error Trap (Hard Trap, Level 13), the following additional circumstances under which an Address Error Trap may occur, should be included in the numbered list of items:

> 4. Execution of a "`BRA #literal`" instruction or a "`GOTO #literal`" instruction, where `literal` is an unimplemented program memory address.
> 5. Executing instructions after modifying the PC to point to unimplemented program memory addresses. The PC may be modified by loading a value into the stack and executing a `RETURN` instruction.

19. Page 6-10, Section 6.2.5 Wake-up from SLEEP and IDLE

On page 6-10, Section 6.2.5 Wake-up from SLEEP and IDLE, the existing note should be replaced with the following:

> **Note:** User interrupt sources that are assigned to CPU priority level 0 cannot wake the CPU from SLEEP or IDLE mode, because the interrupt source is effectively disabled. To use an interrupt as a wake-up source, the CPU priority level for the interrupt must be assigned to CPU priority level 1 or greater.

20. Page 6-43, Table 6-3: Special Function Registers Associated with Interrupt Controller

On page 6-43, Table 6-3: Special Function Registers Associated with Interrupt Controller, row 3, Bit 0, SFR Name: IFS0, should be replaced with the following:

INT0IF

21. Page 7-7, Register 7-1 OSCCON

On page 7-7, Register 7-1: OSCCON, the register description should be modified as follows to include the TUN<3:0> bits for the on-chip Fast RC oscillator. All other bit definitions do not change and hence are not described.

Register 7-1: OSCCON: Oscillator Control Register

Upper Byte:

R/W-0	R/W-0	R-y	R-y	U-0	U-0	R/W-y	R/W-y
TUN3	**TUN2**	COSC<1:0>		**TUN1**	**TUN0**	NOSC<1:0>	
bit 15							bit 8

Lower Byte:

R/W-0	R/W-0	R-0	U-0	R/W-0	U-0	R/W-0	R/W-0
POST<1:0>		LOCK	—	CF	—	LPOSCEN	OSWEN
bit 7							bit 0

bit 15-14 **TUN<3:2>:** Upper 2 bits of the TUN bit-field. Refer to the description of TUN<1:0> (OSCCON<11:10>) bits for details.

bit 11-10 **TUN<1:0>:** Lower 2 bits of the TUN bit-field.
The four bit field specified by TUN<3:0> allows the user to tune the internal fast RC oscillator which has a nominal frequency of 8 MHz. The user may be able to tune the frequency of the FRC oscillator within a range of +/- 12% (or 960 kHz) in steps of 1.5% around the factory-calibrated frequency setting, as follows:
TUN<3:0> = `0111` provides the highest frequency
......
TUN<3:0> = `0000` provides the factory-calibrated frequency
......
TUN<3:0> = `1000` provides the lowest frequency

Note: Refer to the device-specific data sheet for the tuning range and tuning step size for the FRC oscillator on your device.

Legend:
R = Readable bit W = Writable bit U = Unimplemented bit, read as ‘0’
-n = Value at POR ‘1’ = Bit is set ‘0’ = Bit is cleared x = Bit is unknown
y = Value set from configuration bits on POR or BOR

Note: The OSCCON register description and functionality may vary depending on the clock sources available on the selected device. Please refer to the specific device data sheet for additional details on this register..

22. Page 7-23, Section 7.17.3 Clock Switching Tips

On page 7-23, Section 7.17.3 Clock Switching Tips, add the following note to the end:

> **Note:** The application should not attempt to switch to a clock of frequency lower than 100 KHz when the fail-safe clock monitor is enabled. If such clock switching is performed, the device may generate an oscillator fail trap and switch to the Fast RC oscillator.

23. Page 7-24, Section 7.17.7.2 Aborting a Clock Switch

On page 7-24, Section 7.17.7.2 Aborting a Clock Switch, should be replaced with the following:

7.17.7.2 Aborting a Clock Switch

The following code sequence would be used to ABORT an unsuccessful clock switch:

```
MOV      #OSCCON,W1          ; pointer to OSCCON
MOV.B    #0x46,W2            ; first unlock code
MOV.B    #0x57,W3            ; second unlock code
MOV.B    W2, [W1]            ; write first unlock code
MOV.B    W3, [W1]            ; write second unlock code
BCLR     OSCCON,#OSWEN       ; ABORT the switch
```

24. Page 10-4, Section 10.3.6 Wake-up from SLEEP on Interrupt

On page 10-4, Section 10.3.6 Wake-up from SLEEP on Interrupt, the following text should be added:

User interrupt sources that are assigned to CPU priority level 0 cannot wake the CPU from SLEEP mode, because the interrupt source is effectively disabled. To use an interrupt as a wake-up source, the CPU priority level for the interrupt must be assigned to CPU priority level 1 or greater.

25. Page 10-4, Section 10.4 IDLE Mode

On page 10-4, Section 10.4 IDLE Mode, the following text should be added:

User interrupt sources that are assigned to CPU priority level 0 cannot wake the CPU from IDLE mode, because the interrupt source is effectively disabled. To use an interrupt as a wake-up source, the CPU priority level for the interrupt must be assigned to CPU priority level 1 or greater.

26. Page 15-6, Register 15-2: PTMR: PWM Time Base Register, bit 14-0

On page 15-6, Register 15-2: PTMR: PWM Time Base Register, the description of bit 0 through bit 14 should be replaced as follows. All other bit definitions do not change and hence are not described.

Register 15-2: PTMR: PWM Time Base Register

Upper Byte:							
R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTDIR	PTMR <14:8>						
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PTMR <7:0>							
bit 7							bit 0

bit 14-0 **PTMR <14:0>:** PWM Timebase Register Count Value

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

27. Page 15-32, Section 15.10 PWM Fault Pins

On page 15-32, Section 15.10 PWM Fault Pins, paragraph 2 should be replaced with the following:

> The fault pins may have other multiplexed functions depending on the dsPIC device variant. When used as a fault input, each fault pin is readable via its corresponding PORT register. The $\overline{\text{FLTA}}$ and $\overline{\text{FLTB}}$ pins function as active low inputs so that it is easy to wire-OR many sources to the same input through an external pull-up resistor. When not used with the PWM module, these pins may be used as general purpose I/O or **another multiplexed function**. Each fault pin has its own interrupt vector, interrupt flag bit, interrupt enable bit, and interrupt priority bits associated with it.

28. Page 15-32, Section 15.10.1 Fault Pin Enable Bits

On page 15-32, Section 15.10.1 Fault Pin Enable Bits, paragraph 2 should be replaced with the following:

> If all enable bits are cleared in the FLTACON or FLTBCON registers, then that fault input pin has no effect on the PWM module and **no fault interrupts will be produced**.

29. Page 17-9, Register 17-4 ADCHS

On page 17-9, Section 17-4 ADCHS, the following note should be added to the ADCHS register description:

> **Note:** The ADCHS register description and functionality will vary depending on the number of A/D inputs available on the selected device. Please refer to the specific device datasheet for additional details on this register.

30. Page 17-14, Section 17.7 Selecting the A/D Conversion Clock

On page 17-14, Section 17.7 Selecting the A/D Conversion Clock, paragraph 3 should be replaced with the following:

> For correct A/D conversions, the A/D conversion clock (T_{AD}) must be selected to ensure a minimum T_{AD} time of 154 nsec (for V_{DD} = 5V).

Also remove Table 17-1: Typical T_{AD} vs. Device Operating Frequencies.

31. Page 17-37, Table 17-5 Converting Three Inputs, Four Times and Four Inputs, One time Per Interrupt

On page 17-37, Table 17-5 Converting Three Inputs, Four Times and Four Inputs, One Timer Per Interrupt, the initialization of the SMPI <3:0> (ADCON2<5:2>) bits should be modified as follows:

SMPI<3:0> = 0011

32. Page 17-46, Section 17.16 A/D Sampling Requirements

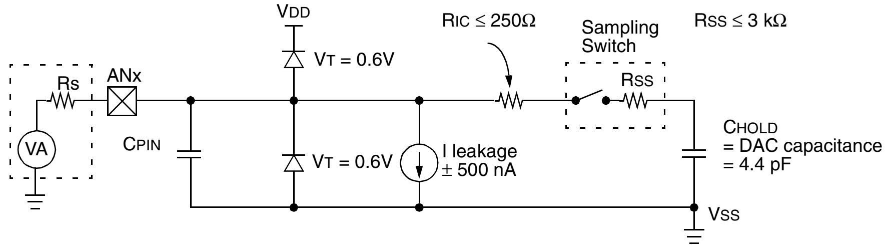
The entire Section 17.16 should be replaced with the following text and figures:

The analog input model of the 10-bit A/D converter is shown in Figure 17-21. The total sampling time for the A/D is a function of the internal amplifier settling time and the holding capacitor charge time.

For the A/D converter to meet its specified accuracy, the charge holding capacitor (C_{HOLD}) must be allowed to fully charge to the voltage level on the analog input pin. The source impedance (R_S), the interconnect impedance (R_{IC}), and the internal sampling switch (R_{SS}) impedance combine to directly affect the time required to charge the capacitor C_{HOLD}. The combined impedance of the analog sources must therefore be small enough to fully charge the holding capacitor within the chosen sample time. To minimize the effects of pin leakage currents on the accuracy of the A/D converter, the maximum recommended source impedance, R_S, is 5 kΩ. After the analog input channel is selected (changed), this sampling function must be completed prior to starting the conversion. The internal holding capacitor will be in a discharged state prior to each sample operation.

At least 1 T_{AD} time period should be allowed between conversions for the sample time. For more details, see the device electrical specifications.

Figure 17-21: 10-bit A/D Converter Analog Input Model



Legend: C_{PIN} = input capacitance
V_T = threshold voltage
I leakage = leakage current at the pin due to various junctions
R_{IC} = interconnect resistance
R_{SS} = sampling switch resistance
C_{HOLD} = sample/hold capacitance (from DAC)

Note: C_{PIN} value depends on device package and is not tested. Effect of C_{PIN} negligible if Rs ≤ 5 kΩ.

33. Page 18-12, Section 18.7 Selecting the A/D Conversion Clock

On page 18-12, Section 18.7 Selecting the A/D Conversion Clock, paragraph 3 should be replaced with the following:

For correct A/D conversions, the A/D conversion clock (T_{AD}) must be selected to ensure a minimum T_{AD} time of 667 nsec (for V_{DD} = 5V).

Also remove Table 18-1: Typical T_{AD} vs. Device Operating Frequencies.

34. Page 18-27, Section 18.15 A/D Sampling Requirements

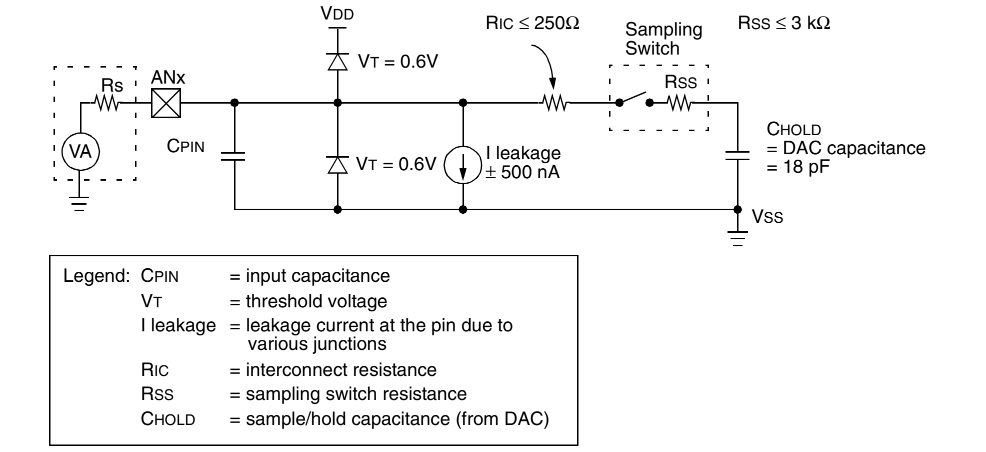
The entire Section 18.15 should be replaced with the following text and figures:

The analog input model of the 12-bit A/D converter is shown in Figure 18-11. The total sampling time for the A/D is a function of the internal amplifier settling time and the holding capacitor charge time.

For the A/D converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the voltage level on the analog input pin. The source impedance (RS), the interconnect impedance (RIC), and the internal sampling switch (RSS) impedance combine to directly affect the time required to charge the capacitor CHOLD. The combined impedance of the analog sources must therefore be small enough to fully charge the holding capacitor within the chosen sample time. To minimize the effects of pin leakage currents on the accuracy of the A/D converter, the maximum recommended source impedance, RS, is 2.5 kΩ. After the analog input channel is selected (changed), this sampling function must be completed prior to starting the conversion. The internal holding capacitor will be in a discharged state prior to each sample operation.

At least 1 TAD time period should be allowed between conversions for the sample time. For more details, see the device electrical specifications.

Figure 18-11: 12-bit A/D Converter Analog Input Model


VDD
RIC ≤ 250Ω
Sampling Switch
RSS ≤ 3 kΩ
VT = 0.6V
Rs
ANx
RSS
VA
CPIN
VT = 0.6V
I leakage
± 500 nA
CHOLD
= DAC capacitance
= 18 pF
VSS
Legend: CPIN = input capacitance
VT = threshold voltage
I leakage = leakage current at the pin due to various junctions
RIC = interconnect resistance
RSS = sampling switch resistance
CHOLD = sample/hold capacitance (from DAC)

Note: CPIN value depends on device package and is not tested. Effect of CPIN negligible if Rs ≤ 2.5 kΩ.

35. Page 19-10, Section 19.4.1 Enabling the UART, Page 19-13, Section 19.5.3 Setup for UART Transmit and Page 19-20, Section 19.9 Initialization

Add the following note to these subsections:

> **Note:** The UTXEN bit should not be set until the UARTEN bit has been set. Otherwise, UART transmissions will not be enabled.

36. Page 19-13, Section 19.5.3 Setup for UART Transmit

On page 19-13, Section 19.5.3 Setup for UART Transmit, delete Step 3 in this section.

37. Page 19-17, Section 19.6.4 Setup for UART Reception

On page 19-17, Section 19.6.4 Setup for UART Reception, delete Step 3 in this section.

38. Page 19-20, Section 19.9 Initialization

On page 19-20, Section 19.9 Initialization, Example 19-2 and Example 19-3 should be replaced with the following:

Example 19-2: 8-bit Transmit/Receive (UART1)

```
MOV    #baudrate,W0    ; Set Baudrate
MOV    W0,U1BRG

BSET   IPC2,#U1TXIP2   ; Set UART TX interrupt priority
BCLR   IPC2,#U1TXIP1   ;
BCLR   IPC2,#U1TXIP0   ;
BSET   IPC2,#U1RXIP2   ; Set UART RX interrupt priority
BCLR   IPC2,#U1RXIP1   ;
BCLR   IPC2,#U1RXIP0   ;

CLR    U1STA

MOV    #0x8800,W0      ; Enable UART for 8-bit data,
                       ; no parity, 1 STOP bit,
                       ; no wakeup
MOV    W0,U1MODE

BSET   U1STA,#UTXEN    ; Enable transmit

BSET   IEC0,#U1TXIE    ; Enable transmit interrupts
BSET   IEC0,#U1RXIE    ; Enable receive interrupts
```

Example 19-3: 8-bit Transmit/Receive (UART1), Address Detect Enabled

```
    MOV     #baudrate,W0     ; Set Baudrate
    MOV     W0,U1BRG

    BSET    IPC2,#U1TXIP2    ; Set UART TX interrupt priority
    BCLR    IPC2,#U1TXIP1    ;
    BCLR    IPC2,#U1TXIP0    ;
    BSET    IPC2,#U1RXIP2    ; Set UART RX interrupt priority
    BCLR    IPC2,#U1RXIP1    ;
    BCLR    IPC2,#U1RXIP0    ;

    BSET    U1STA,#ADDEN     ; Enable address detect

    MOV     #0x8883,W0       ; UART1 enabled for 9-bit data,
                             ; no parity, 1 STOP bit,
                             ; wakeup enabled
    MOV     W0,U1MODE

    BSET    U1STA,#UTXEN     ; Enable transmit

    BSET    IEC0,#U1TXIE     ; Enable transmit interrupts
    BSET    IEC0,#U1RXIE     ; Enable receive interrupts
```

39. Page 21-9, Register 21-1 I2CCON

On page 21-9, Register 21-1: I^{2}C Control Register, the description of the ACKDT bit should be corrected as follows. All other bit definitions do not change and hence are not described.

Register 21-1: I2CCON: I^{2}C Control Register

Upper Byte:							
R/W-0	U-0	R/W-0	R/W-1 HC	R/W-0	R/W-0	R/W-0	R/W-0
I2CEN	—	I2CSIDL	SCLREL	IPMIEN	A10M	DISSLW	SMEN
bit 15							bit 8

Lower Byte:							
R/W-0	R/W-0	R/W-0	R/W-0 HC	R/W-0 HC	R/W-0 HC	R/W-0 HC	R/W-0 HC
GCEN	STREN	**ACKDT**	ACKEN	RCEN	PEN	RSEN	SEN
bit 7							bit 0

bit 5 **ACKDT:** Acknowledge Data bit (When operating as I^{2}C Master. Applicable during master receive.)
Value that will be transmitted when the software initiates an Acknowledge sequence.
`1` = Send NACK during acknowledge
`0` = Send ACK during acknowledge

Legend:		
R = Readable	C = Clearable bit	U = Unimplemented bit, read as ‘0’
W = Writable	HS = Set by Hardware	S = Settable bit
HC = Cleared by Hardware	‘0’ = Bit cleared at POR	x = Bit is unknown at POR
‘1’ = Bit is set at POR		

40. I^{2}C Interrupt Bit Names

Table 2 below, shows changes to the nomenclature of the I^{2}C module’s interrupt enable, status flag and priority bits. These changes should be applied to the entire document.

TABLE 2: INTERRUPT CONTROLLER REGISTER MAP: I^{2}C BIT NAMES

Bit Location in SFR	Bit Name (Incorrect)	Bit Name (Please read as)
IFS0<14>	BCLIF	MI2CIF
IFS0<13>	I2CIF	SI2CIF
IEC0<14>	BCLIE	MI2CIE
IEC0<13>	I2CIE	SI2CIE
IPC3<10:8>	BCLIP<2:0>	MI2CIP<2:0>
IPC3<6:4>	I2CIP<2:0>	SI2CIP<2:0>

41. Page 22-12, Equation 22-1 DCI Bit Clock Generator Value

On page 22-12, Equation 22-1: DCI Bit Clock Generator Value, should be replaced by the following:

Equation 22-1: DCI Bit Clock Generator Value

$$\mathrm{BCG<11:0>} = \frac{f_{CY}}{2\, f_{CSCK}} - 1$$

42. Page 22-24, Section 22.5.5.3 I^{2}S Data Justification

On page 22-24, Section 22.5.5.3 I^{2}S Data Justification, paragraph 2 should be replaced by the following:

> If DJST = 1, the I^{2}S data transfers will be MSb left justified. The MSb of the data word will be presented on the CSDO pin during the same serial clock cycle as the rising or falling edge of the FS signal. After the data word has been transmitted, the state of the CSDO pin is dictated by the CSDOM (DCICON1<6>) bit.

43. Page 23-38, Section 23.5.2 Disable Mode

On page 23-38, Section 23.5.2 Disable Mode, add the following note to the end.

> **Note:** Typically, if the CAN module is allowed to transmit in a particular mode of operation and a transmission is requested immediately after the CAN module has been placed in that mode of operation, the module waits for 11 consecutive recessive bits on the bus before starting transmission. If the user switches to Disable Mode within this 11-bit period, then this transmission is aborted and the corresponding TXABT bit is set and TXREQ bit is cleared.

44. Pages 23-22 through 23-27, Table 23-1 and Table 23-2 CAN1/CAN2 Register Map

On pages 23-22 through 23-27 in Tables 23-1 and 23-2, the nomenclature of the CAN module transmit and receive buffer registers as well as the addresses of the CAN2 module registers should be changed/corrected as follows:

TABLE 23-1: CAN1 REGISTER MAP

File Name	ADR	Bit																RESET
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
C1RXF0SID	300	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C1RXF0EIDH	302	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C1RXF0EIDL	304	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	306	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C1RXF1SID	308	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C1RXF1EIDH	30A	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C1RXF1EIDL	30C	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	30E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C1RXF2SID	310	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C1RXF2EIDH	312	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C1RXF2EIDL	314	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	316	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C1RXF3SID	318	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C1RXF3EIDH	31A	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C1RXF3EIDL	31C	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	31E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C1RXF4SID	320	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C1RXF4EIDH	322	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C1RXF4EIDL	324	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	326	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C1RXF5SID	328	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C1RXF5EIDH	32A	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C1RXF5EIDL	32C	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	32E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C1RXM0SID	330	—	—	—	SID<10:6>					SID<5:0>						—	MIDE	xxxx
C1RXM0EIDH	332	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C1RXM0EIDL	334	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	336	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C1RXM1SID	338	—	—	—	SID<10:6>					SID<5:0>						—	MIDE	xxxx
C1RXM1EIDH	33A	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C1RXM1EIDL	33C	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	33E	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx

TABLE 23-1: CAN1 REGISTER MAP (CONTINUED)

File Name	ADR	Bit																RESET
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
C1TX2SID	340	SID<10:6>					—	—	—	SID<5:0>						SRR	TX IDE	xxxx
C1TX2EID	342	EID<17:14>				—	—	—	—	EID<13:6>								xxxx
C1TX2DLC	342	EID<5:0>						TX RTR	TX RB1	TX RB0	DLC<3:0>				—	—	—	xxxx
C1TX2B1	346	Transmit Buffer 0 Byte 1								Transmit Buffer 0 Byte 0								xxxx
C1TX2B2	348	Transmit Buffer 0 Byte 3								Transmit Buffer 0 Byte 2								xxxx
C1TX2B3	34A	Transmit Buffer 0 Byte 5								Transmit Buffer 0 Byte 4								xxxx
C1TX2B4	34C	Transmit Buffer 0 Byte 7								Transmit Buffer 0 Byte 6								xxxx
C1TX2CON	34E	—	—	—	—	—	—	—	—	—	TX ABT	TX LARB	TX ERR	TX REQ	—	TXPRI[1:0]		0000
C1TX1SID	350	SID<10:6>					—	—	—	SID<5:0>						SRR	TX IDE	xxxx
C1TX1EID	352	EID<17:14>				—	—	—	—	EID<13:6>								xxxx
C1TX1DLC	352	EID<5:0>						TX RTR	TX RB1	TX RB0	DLC<3:0>				—	—	—	xxxx
C1TX1B1	356	Transmit Buffer 0 Byte 1								Transmit Buffer 0 Byte 0								xxxx
C1TX1B2	358	Transmit Buffer 0 Byte 3								Transmit Buffer 0 Byte 2								xxxx
C1TX1B3	35A	Transmit Buffer 0 Byte 5								Transmit Buffer 0 Byte 4								xxxx
C1TX1B4	35C	Transmit Buffer 0 Byte 7								Transmit Buffer 0 Byte 6								xxxx
C1TX1CON	35E	—	—	—	—	—	—	—	—	—	TX ABT	TX LARB	TX ERR	TX REQ	—	TXPRI[1:0]		0000
C1TX0SID	360	SID<10:6>					—	—	—	SID<5:0>						SRR	TX IDE	xxxx
C1TX0EID	362	EID<17:14>				—	—	—	—	EID<13:6>								xxxx
C1TX0DLC	362	EID<5:0>						TX RTR	TX RB1	TX RB0	DLC<3:0>				—	—	—	xxxx
C1TX0B1	366	Transmit Buffer 0 Byte 1								Transmit Buffer 0 Byte 0								xxxx
C1TX0B2	368	Transmit Buffer 0 Byte 3								Transmit Buffer 0 Byte 2								xxxx
C1TX0B3	36A	Transmit Buffer 0 Byte 5								Transmit Buffer 0 Byte 4								xxxx
C1TX0B4	36C	Transmit Buffer 0 Byte 7								Transmit Buffer 0 Byte 6								xxxx
C1TX0CON	36E	—	—	—	—	—	—	—	—	—	TX ABT	TX LARB	TX ERR	TX REQ	—	TXPRI[1:0]		0000

TABLE 23-1: CAN1 REGISTER MAP (CONTINUED)

File Name	ADR	Bit																RESET
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
C1RX1SID	370	—	—	—	SID<10:6>					SID<5:0>						SRR	RX IDE	xxxx
C1RX1EID	372	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C1RX1DLC	374	EID<0:5>						RX RTR	RX RB1	—	—	—	RX RB0	DLC[3:0]				xxxx
C1RX1B1	376	Receive Buffer 1 Byte 1								Receive Buffer 1 Byte 0								xxxx
C1RX1B2	378	Receive Buffer 1 Byte 3								Receive Buffer 1 Byte 2								xxxx
C1RX1B3	37A	Receive Buffer 1 Byte 5								Receive Buffer 1 Byte 4								xxxx
C1RX1B4	37C	Receive Buffer 1 Byte 7								Receive Buffer 1 Byte 6								xxxx
C1RX1CON	37E	—	—	—	—	—	—	—	—	RX FUL	—	—	RX ERR	RX RTR R0	FILHIT[2:0]			0000
C1RX1SID	380	—	—	—	SID<10:6>					SID<5:0>						SRR	RX IDE	xxxx
C1RX1EID	382	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C1RX1DLC	384	EID<0:5>						RX RTR	RX RB1	—	—	—	RX RB0	DLC[3:0]				xxxx
C1RX0B1	386	Receive Buffer 0 Byte 1								Receive Buffer 0 Byte 0								xxxx
C1RX0B2	388	Receive Buffer 0 Byte 3								Receive Buffer 0 Byte 2								xxxx
C1RX0B3	38A	Receive Buffer 0 Byte 5								Receive Buffer 0 Byte 4								xxxx
C1RX0B4	38C	Receive Buffer 0 Byte 7								Receive Buffer 0 Byte 6								xxxx
C1RX0CON	38E	—	—	—	—	—	—	—	—	RX FUL	—	—	RX ERR	RX RTR R0	RXB0 DBEN	JTOFF	FIL HIT 0	0000
C1CTRL	390	CAN CAP	—	C SIDL	ABAT	CAN CKS	REQOP[2:0]			OPMODE[2:0]			—	ICODE[2:0]			—	0480
C1CFG1	392	—	—	—	—	—	—	—	—	SJW[1:0]S		BRP[5:0]						0000
C1CFG2	394	—	WAK FIL	—	—	—	SEG2PH[2:0]			SEG2 PHTS	SAM	SEG1PH[2:0]			PRSEG[2:0]			0000
C1INTF	396	RXB0 OVR	RXB1 OVR	TXBO	TXBP	RXBP	TX WARN	RX WARN	E WARN	IVR IF	WAK IF	ERR IF	TXB2 IF	TXB1 IF	TXB0 IF	RXB1 IF	RXB0 IF	0000
C1INTE	398	—	—	—	—	—	—	—	—	IVR IE	WAK IE	ERR IE	TXB2 IE	TXB1 IE	TXB0 IE	RXB1 IE	RXB0 IE	0000
C1EC	39A	Transmit Error Counter								Receive Error Counter								0000
Reserved	39C 3FE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx

Legend: x = Unknown

TABLE 23-2: CAN2 REGISTER MAP

File Name	ADR	Bit																RESET
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
C2RXF0SID	3C0	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C2RXF0EIDH	3C2	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C2RXF0EIDL	3C4	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	3C6	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C2RXF1SID	3C8	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C2RXF1EIDH	3CA	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C2RXF1EIDL	3CC	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	3CE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C2RXF2SID	3D0	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C2RXF2EIDH	3D2	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C2RXF2EIDL	3D4	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	3D6	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C2RXF3SIDH	3D8	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C2RXF3EID	3DA	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C2RXF3EIDL	3DC	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	3DE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C2RXF4SID	3E0	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C2RXF4EIDH	3E2	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C2RXF4EIDL	3E4	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	3E6	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C2RXF5SID	3E8	—	—	—	SID<10:6>					SID<5:0>						—	EXIDE	xxxx
C2RXF5EIDH	3EA	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C2RXF5EIDL	3EC	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	3EE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C2RXM0SID	3F0	—	—	—	SID<10:6>					SID<5:0>						—	MIDE	xxxx
C2RXM0EIDH	3F2	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C2RXM0EIDL	3F4	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	3F6	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx
C2RXM1SID	3F8	—	—	—	SID<10:6>					SID<5:0>						—	MIDE	xxxx
C2RXM1EIDH	3FA	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C2RXM1EIDL	3FC	EID<5:0>						—	—	—	—	—	—	—	—	—	—	xxxx
unused	3FE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx

TABLE 23-2: CAN2 REGISTER MAP (CONTINUED)

File Name	ADR	Bit																RESET
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
C2TX2SID	400	SID<10:6>					—	—	—	SID<5:0>						SRR	TX IDE	xxxx
C2TX2EID	402	EID<17:14>				—	—	—	—	EID<13:6>								xxxx
C2TX2DLC	404	EID<5:0>						TX RTR	TX RB1	TX RB0	DLC<3:0>				—	—	—	xxxx
C2TX2B1	406	Transmit Buffer 0 Byte 1								Transmit Buffer 0 Byte 0								xxxx
C2TX2B2	408	Transmit Buffer 0 Byte 3								Transmit Buffer 0 Byte 2								xxxx
C2TX2B3	40A	Transmit Buffer 0 Byte 5								Transmit Buffer 0 Byte 4								xxxx
C2TX2B4	40C	Transmit Buffer 0 Byte 7								Transmit Buffer 0 Byte 6								xxxx
C2TX2CON	40E	—	—	—	—	—	—	—	—	—	TX ABT	TX LARB	TX ERR	TX REQ	—	TXPRI[1:0]		0000
C2TX1SID	410	SID<10:6>					—	—	—	SID<5:0>						SRR	TX IDE	xxxx
C2TX1EID	412	EID<17:14>				—	—	—	—	EID<13:6>								xxxx
C2TX1DLC	414	EID<5:0>						TX RTR	TX RB1	TX RB0	DLC<3:0>				—	—	—	xxxx
C2TX1B1	416	Transmit Buffer 0 Byte 1								Transmit Buffer 0 Byte 0								xxxx
C2TX1B2	418	Transmit Buffer 0 Byte 3								Transmit Buffer 0 Byte 2								xxxx
C2TX1B3	41A	Transmit Buffer 0 Byte 5								Transmit Buffer 0 Byte 4								xxxx
C2TX1B4	41C	Transmit Buffer 0 Byte 7								Transmit Buffer 0 Byte 6								xxxx
C2TX1CON	41E	—	—	—	—	—	—	—	—	—	TX ABT	TX LARB	TX ERR	TX REQ	—	TXPRI[1:0]		0000
C2TX0SID	420	SID<10:6>					—	—	—	SID<5:0>						SRR	TX IDE	xxxx
C2TX0EID	422	EID<17:14>				—	—	—	—	EID<13:6>								xxxx
C2TX0DLC	424	EID<5:0>						TX RTR	TX RB1	TX RB0	DLC<3:0>				—	—	—	xxxx
C2TX0B1	426	Transmit Buffer 0 Byte 1								Transmit Buffer 0 Byte 0								xxxx
C2TX0B2	428	Transmit Buffer 0 Byte 3								Transmit Buffer 0 Byte 2								xxxx
C2TX0B3	42A	Transmit Buffer 0 Byte 5								Transmit Buffer 0 Byte 4								xxxx
C2TX0B4	42C	Transmit Buffer 0 Byte 7								Transmit Buffer 0 Byte 6								xxxx
C2TX0CON	42E	—	—	—	—	—	—	—	—	—	TX ABT	TX LARB	TX ERR	TX REQ	—	TXPRI[1:0]		0000

TABLE 23-2: CAN2 REGISTER MAP (CONTINUED)

File Name	ADR	Bit																RESET
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
C2RX1SID	430	—	—	—	SID<10:6>					SID<5:0>						SRR	RX IDE	xxxx
C2RX1EID	432	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C2RX1DLC	434	EID<0:5>						RX RTR	RX RB1	—	—	—	RX RB0	DLC[3:0]				xxxx
C2RX1B1	436	Receive Buffer 1 Byte 1								Receive Buffer 1 Byte 0								xxxx
C2RX1B2	438	Receive Buffer 1 Byte 3								Receive Buffer 1 Byte 2								xxxx
C2RX1B3	43A	Receive Buffer 1 Byte 5								Receive Buffer 1 Byte 4								xxxx
C2RX1B4	43C	Receive Buffer 1 Byte 7								Receive Buffer 1 Byte 6								xxxx
C2RX1CON	43E	—	—	—	—	—	—	—	—	RX FUL	—	—	RX ERR	RX RTR R0	FILHIT[2:0]			0000
C2RX1SID	440	—	—	—	SID<10:6>					SID<5:0>						SRR	RX IDE	xxxx
C2RX1EID	442	—	—	—	—	EID<17:14>				EID<13:6>								xxxx
C2RX1DLC	444	EID<0:5>						RX RTR	RX RB1	—	—	—	RX RB0	DLC[3:0]				xxxx
C2RX0B1	446	Receive Buffer 0 Byte 1								Receive Buffer 0 Byte 0								xxxx
C2RX0B2	448	Receive Buffer 0 Byte 3								Receive Buffer 0 Byte 2								xxxx
C2RX0B3	44A	Receive Buffer 0 Byte 5								Receive Buffer 0 Byte 4								xxxx
C2RX0B4	44C	Receive Buffer 0 Byte 7								Receive Buffer 0 Byte 6								xxxx
C2RX0CON	44E	—	—	—	—	—	—	—	—	RX FUL	—	—	RX ERR	RX RTR R0	RXB0 DBEN	JTOFF	FIL HIT 0	0000
C2CTRL	450	CAN CAP	—	C SIDL	ABAT	CAN CKS	REQOP[2:0]			OPMODE[2:0]			—	ICODE[2:0]			—	0480
C2CFG1	452	—	—	—	—	—	—	—	—	SJW[1:0]S		BRP[5:0]						0000
C2CFG2	454	—	WAK FIL	—	—	—	SEG2PH[2:0]			SEG2 PHTS	SAM	SEG1PH[2:0]			PRSEG[2:0]			0000
C2INTF	456	RXB0 OVR	RXB1 OVR	TXBO	TXBP	RXBP	TX WARN	RX WARN	E WARN	IVR IF	WAK IF	ERR IF	TXB2 IF	TXB1 IF	TXB0 IF	RXB1 IF	RXB0 IF	0000
C2INTE	458	—	—	—	—	—	—	—	—	IVR IE	WAK IE	ERR IE	TXB2 IE	TXB1 IE	TXB0 IE	RXB1 IE	RXB0 IE	0000
C2EC	45A	Transmit Error Counter								Receive Error Counter								0000
Reserved	45C 4FE	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	xxxx

Legend: x = Unknown

45. Page 24-7, Section 24.3.4.1 General Code Segment Configuration Bit Group

On page 24-7, Section 24.3.4.1 General Code Segment Configuration Bit Group, add the following note to the end.

> **Note:** If the code protection configuration fuse group (FGS<GCP:GWRP>) bits have been programmed, an erase of the entire code-protected device is only possible at voltages, V_{DD} >= 4.5 volts.

46. Page 16-7, Section 16.2 DFLTCON Page 16-8, Section 16.3 Programmable Digital Noise Filters Page 16-14, Section 16.5.3.2 Index Pulse De-skew

The control bits used to control the QEI digital input filters may differ depending upon the device variant that is used. This affects the control bits found in the DFLTCON register. Please refer to the specific device data sheet to determine which DFLTCON control bits are implemented for the device you are using. Some devices have two sets of control bits to control the digital input filters. One set of control bits sets the digital filter characteristics for the INDX pin. The second set of control bits sets the digital filter characteristics for the QEA and QEB pins. The DS70063B document correctly depicts the DFLTCON control register for these device variants.

Other device variants have one set of control bits that set the digital filter characteristics for the INDX, QEA and QEB pins. The DFLTCON register for these device variants is provided in Register 16-1 below. If the device variant has this DFLTCON register, then paragraph 5 of Section 16.3 "Programmable Digital Noise Filters" is not applicable. Also, Section 16.5.3.2. "Index Pulse De-skew" is not applicable.

Register 16-1: DFLTCON: Digital Filter Control Register

Upper Half:

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	IMV<1:0>		CEID
bit 15							bit 8

Lower Half:

R/W-0	R/W-0	U-0	U-0	U-0	U-0
QEOUT	QECK<2:0>	—	—	—	—
bit 7					bit 0

bit 15-11 **Unimplemented:** Read as '0'

bit 10-9 **IMV<1:0>:** Index Match Value – These bits allow the user to specify the state of the QEA and QEB input pins during an Index pulse when the POSCNT register is to be reset.
In 4X Quadrature Count Mode:
IMV1= Required State of Phase B input signal for match on index pulse
IMV0= Required State of Phase A input signal for match on index pulse
In 2X Quadrature Count Mode:
IMV1= Selects Phase input signal for Index state match (0 = Phase A, 1 = Phase B)
IMV0= Required State of the selected Phase input signal for match on index pulse

bit 8 **CEID:** Count Error Interrupt Disable
1 = Interrupts due to count errors are disabled
0 = Interrupts due to count errors are enabled

bit 7 **QEOUT:** QEA/QEB/INDX pin Digital Filter Output Enable
1 = Digital filter outputs enabled
0 = Digital filter outputs disabled (normal pin operation)

bit 6-4 **QECK<2:0>:** QEA/QEB/INDX Digital Filter Clock Divide Select Bits
111 = 1:256 Clock Divide
110 = 1:128 Clock Divide
101 = 1:64 Clock Divide
100 = 1:32 Clock Divide
011 = 1:16 Clock Divide
010 = 1:4 Clock Divide
001 = 1:2 Clock Divide
000 = 1:1 Clock Divide

bit 3-0 **Unimplemented:** Read as '0'

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown
y = Value set from configuration bits on POR or BOR			

APPENDIX A: REVISION HISTORY

Revision A (10/2003)

Original version of the document.

Revision B (01/2004)

First revision of the document.

Revision C (02/2004)

Added clarification to QEI DFLTCON register.

Revision D (05/2004)

Additions made to Errata #21 and Errata #29. Added new Erratas #19, #24 and #25. New Errata #32 – updated document and Figure 17-21. Added new Figure 17-21a 65/2004)

Revision E (06/2004)

Additions made to Errata #30, Errata #32 and Errata #34. Errata #44 was updated. Added new Errata #33 and Errata #35.

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.

Information contained in this publication regarding device applications and the like is intended through suggestion only and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights.

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

Trademarks

♻ Printed on recycled paper.

Microchip received ISO/TS-16949:2002 quality system certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona and Mountain View, California in October 2003. The Company's quality system processes and procedures are for its PICmicro® 8-bit MCUs, 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.

MICROCHIP

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: 480-792-7627
Web Address: www.microchip.com

Atlanta

3780 Mansell Road, Suite 130
Alpharetta, GA 30022
Tel: 770-640-0034
Fax: 770-640-0307

Boston

2 Lan Drive, Suite 120
Westford, MA 01886
Tel: 978-692-3848
Fax: 978-692-3821

Chicago

333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071
Fax: 630-285-0075

Dallas

4570 Westgrove Drive, Suite 160
Addison, TX 75001
Tel: 972-818-7423
Fax: 972-818-2924

Detroit

Tri-Atria Office Building
32255 Northwestern Highway, Suite 190
Farmington Hills, MI 48334
Tel: 248-538-2250
Fax: 248-538-2260

Kokomo

2767 S. Albright Road
Kokomo, IN 46902
Tel: 765-864-8360
Fax: 765-864-8387

Los Angeles

18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888
Fax: 949-263-1338

San Jose

1300 Terra Bella Avenue
Mountain View, CA 94043
Tel: 650-215-1444
Fax: 650-961-0286

Toronto

6285 Northam Drive, Suite 108
Mississauga, Ontario L4V 1X5, Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Australia

Suite 22, 41 Rawson Street
Epping 2121, NSW
Australia
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing

Unit 706B
Wan Tai Bei Hai Bldg.
No. 6 Chaoyangmen Bei Str.
Beijing, 100027, China
Tel: 86-10-85282100
Fax: 86-10-85282104

China - Chengdu

Rm. 2401-2402, 24th Floor,
Ming Xing Financial Tower
No. 88 TIDU Street
Chengdu 610016, China
Tel: 86-28-86766200
Fax: 86-28-86766599

China - Fuzhou

Unit 28F, World Trade Plaza
No. 71 Wusi Road
Fuzhou 350001, China
Tel: 86-591-7503506
Fax: 86-591-7503521

China - Hong Kong SAR

Unit 901-6, Tower 2, Metroplaza
223 Hing Fong Road
Kwai Fong, N.T., Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Shanghai

Room 701, Bldg. B
Far East International Plaza
No. 317 Xian Xia Road
Shanghai, 200051
Tel: 86-21-6275-5700
Fax: 86-21-6275-5060

China - Shenzhen

Rm. 1812, 18/F, Building A, United Plaza
No. 5022 Binhe Road, Futian District
Shenzhen 518033, China
Tel: 86-755-82901380
Fax: 86-755-8295-1393

China - Shunde

Room 401, Hongjian Building, No. 2
Fengxiangnan Road, Ronggui Town, Shunde
District, Foshan City, Guangdong 528303, China
Tel: 86-757-28395507 Fax: 86-757-28395571

China - Qingdao

Rm. B505A, Fullhope Plaza,
No. 12 Hong Kong Central Rd.
Qingdao 266071, China
Tel: 86-532-5027355 Fax: 86-532-5027205

India

Divyasree Chambers
1 Floor, Wing A (A3/A4)
No. 11, O'Shaugnessey Road
Bangalore, 560 025, India
Tel: 91-80-22290061 Fax: 91-80-22290062

Japan

Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa, 222-0033, Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

Korea

168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea 135-882
Tel: 82-2-554-7200 Fax: 82-2-558-5932 or
82-2-558-5934

Singapore

200 Middle Road
#07-02 Prime Centre
Singapore, 188980
Tel: 65-6334-8870 Fax: 65-6334-8850

Taiwan

Kaohsiung Branch
30F - 1 No. 8
Min Chuan 2nd Road
Kaohsiung 806, Taiwan
Tel: 886-7-536-4818
Fax: 886-7-536-4803

Taiwan

Taiwan Branch
11F-3, No. 207
Tung Hua North Road
Taipei, 105, Taiwan
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

Austria

Durisolstrasse 2
A-4600 Wels
Austria
Tel: 43-7242-2244-399
Fax: 43-7242-2244-393

Denmark

Regus Business Centre
Lautrup hoj 1-3
Ballerup DK-2750 Denmark
Tel: 45-4420-9895 Fax: 45-4420-9910

France

Parc d'Activite du Moulin de Massy
43 Rue du Saule Trapu
Batiment A - ler Etage
91300 Massy, France
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany

Steinheilstrasse 10
D-85737 Ismaning, Germany
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy

Via Quasimodo, 12
20025 Legnano (MI)
Milan, Italy
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands

Waegenburghtplein 4
NL-5152 JR, Drunen, Netherlands
Tel: 31-416-690399
Fax: 31-416-690340

United Kingdom

505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44-118-921-5869
Fax: 44-118-921-5820

05/28/04