



Welcome to [E-XFL.COM](#)

What is "[Embedded - Microcontrollers](#)"?

"[Embedded - Microcontrollers](#)" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "[Embedded - Microcontrollers](#)"

Details

Product Status	Active
Core Processor	dsPIC
Core Size	16-Bit
Speed	30 MIPS
Connectivity	I ² C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, Motor Control PWM, QEI, POR, PWM, WDT
Number of I/O	30
Program Memory Size	24KB (8K x 24)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	2.5V ~ 5.5V
Data Converters	A/D 9x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/dspic30f3011-30i-pt

1.0 DEVICE OVERVIEW

Note: This data sheet summarizes features of this group of dsPIC30F devices and is not intended to be a complete reference source. For more information on the CPU, peripherals, register descriptions and general device functionality, refer to the “*dsPIC30F Family Reference Manual*” (DS70046). For more information on the device instruction set and programming, refer to the “*16-bit MCU and DSC Programmer’s Reference Manual*” (DS70157).

This document contains device-specific information for the dsPIC30F3010/3011 device. The dsPIC30F devices contain extensive Digital Signal Processor (DSP) functionality within a high-performance 16-bit microcontroller (MCU) architecture. Figure 1-1 and Figure 1-2 illustrate device block diagrams for the dsPIC30F3011 and dsPIC30F3010 devices.

dsPIC30F3010/3011

FIGURE 1-1: dsPIC30F3011 BLOCK DIAGRAM

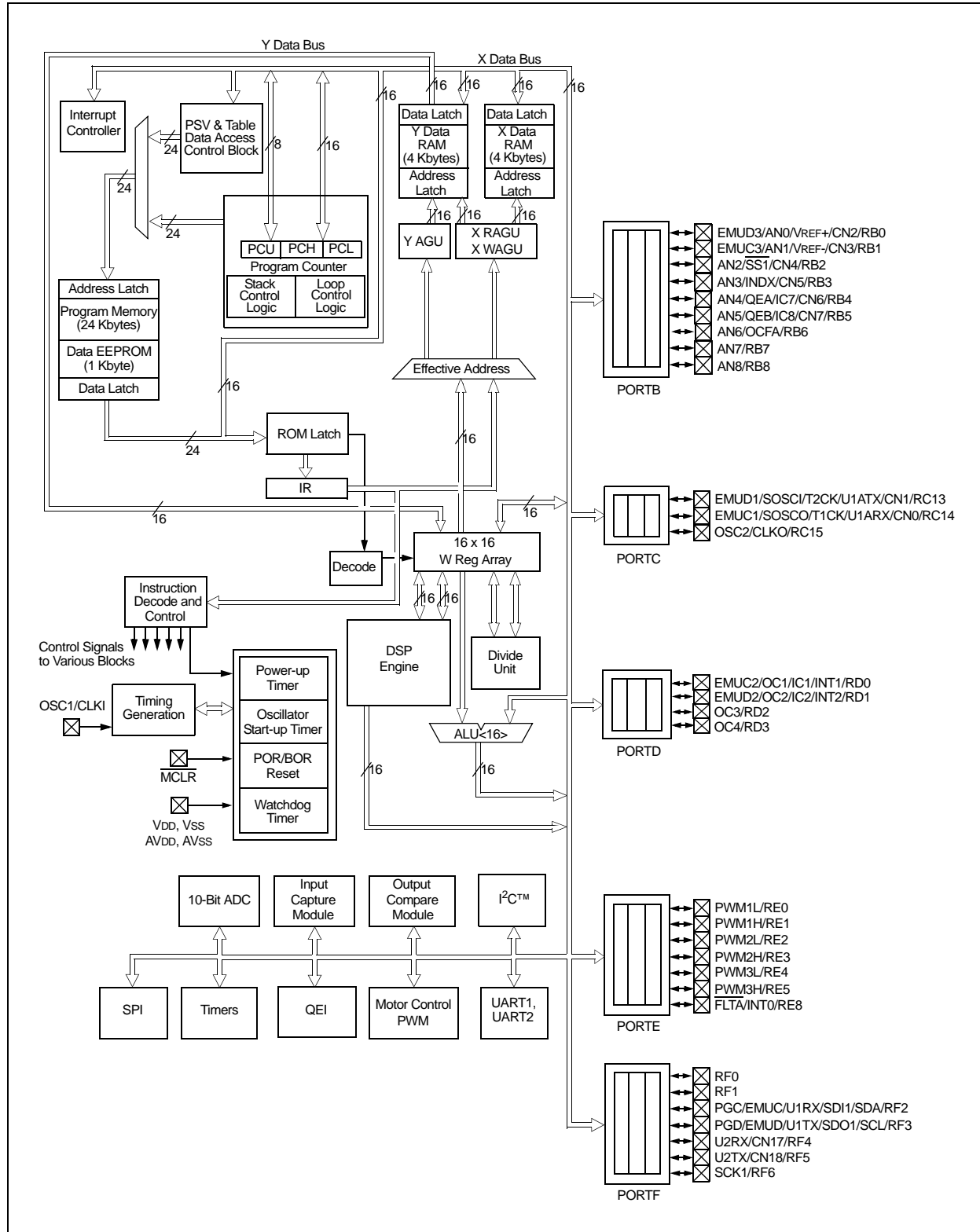
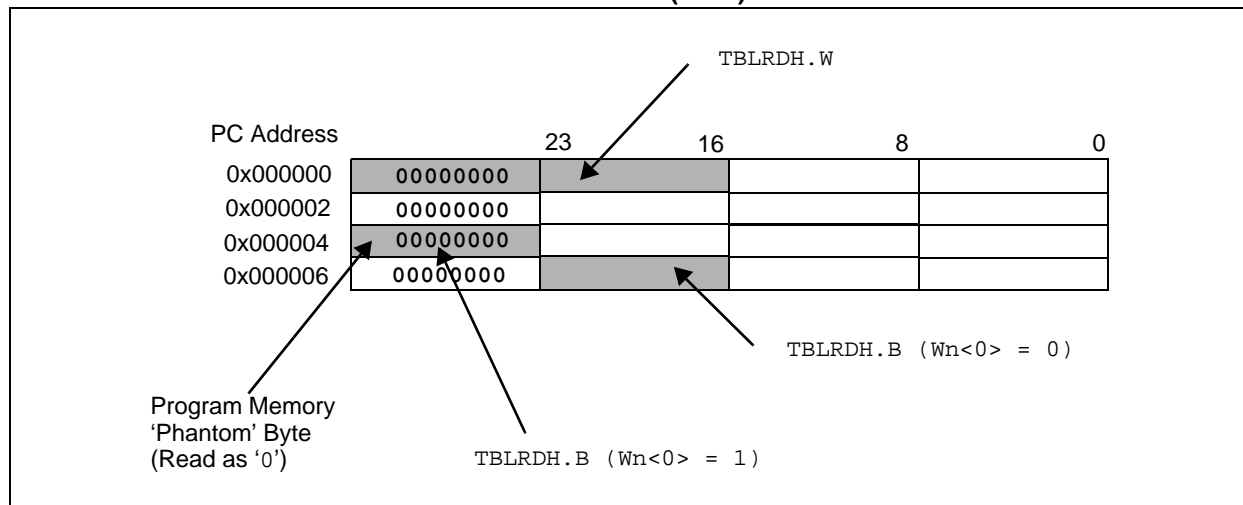


FIGURE 3-4: PROGRAM DATA TABLE ACCESS (MSB)



3.1.2 DATA ACCESS FROM PROGRAM MEMORY USING PROGRAM SPACE VISIBILITY

The upper 32 Kbytes of data space may optionally be mapped into any 16K word program space page. This provides transparent access of stored constant data from X data space, without the need to use special instructions (i.e., TBLRDL/H, TBLWTL/H instructions).

Program space access through the data space occurs if the MSb of the data space, EA, is set and program space visibility is enabled, by setting the PSV bit in the Core Control register (CORCON). The functions of CORCON are discussed in **Section 2.4 “DSP Engine”**.

Data accesses to this area add an additional cycle to the instruction being executed, since two program memory fetches are required.

Note that the upper half of addressable data space is always part of the X data space. Therefore, when a DSP operation uses program space mapping to access this memory region, Y data space should typically contain state (variable) data for DSP operations, whereas X data space should typically contain coefficient (constant) data.

Although each data space address, 0x8000 and higher, maps directly into a corresponding program memory address, as shown in Figure 3-5, only the lower 16 bits of the 24-bit program word are used to contain the data. The upper 8 bits should be programmed to force an illegal instruction to maintain machine robustness. Refer to the “16-bit MCU and DSC Programmer’s Reference Manual” (DS70157) for details on instruction encoding.

Note that by incrementing the PC by 2 for each program memory word, the 15 LSbs of data space addresses directly map to the 15 LSbs in the corresponding program space addresses. The remaining bits are provided by the Program Space Visibility Page register, PSVPAG<7:0>, as shown in Figure 3-5.

Note: PSV access is temporarily disabled during table reads/writes.

For instructions that use PSV which are executed outside a REPEAT loop:

- The following instructions will require one instruction cycle in addition to the specified execution time:
 - MAC class of instructions with data operand prefetch
 - MOV instructions
 - MOV.D instructions
- All other instructions will require two instruction cycles in addition to the specified execution time of the instruction.

For instructions that use PSV which are executed inside a REPEAT loop:

- The following instances will require two instruction cycles in addition to the specified execution time of the instruction:
 - Execution in the first iteration
 - Execution in the last iteration
 - Execution prior to exiting the loop due to an interrupt
 - Execution upon re-entering the loop after an interrupt is serviced
- Any other iteration of the REPEAT loop will allow the instruction, accessing data using PSV, to execute in a single cycle.

TABLE 3-3: CORE REGISTER MAP⁽¹⁾

SFR Name	Address (Home)	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State	
W0	0000	W0/WREG																0000 0000 0000 0000	
W1	0002	W1																0000 0000 0000 0000	
W2	0004	W2																0000 0000 0000 0000	
W3	0006	W3																0000 0000 0000 0000	
W4	0008	W4																0000 0000 0000 0000	
W5	000A	W5																0000 0000 0000 0000	
W6	000C	W6																0000 0000 0000 0000	
W7	000E	W7																0000 0000 0000 0000	
W8	0010	W8																0000 0000 0000 0000	
W9	0012	W9																0000 0000 0000 0000	
W10	0014	W10																0000 0000 0000 0000	
W11	0016	W11																0000 0000 0000 0000	
W12	0018	W12																0000 0000 0000 0000	
W13	001A	W13																0000 0000 0000 0000	
W14	001C	W14																0000 0000 0000 0000	
W15	001E	W15																0000 1000 0000 0000	
SPLIM	0020	SPLIM																0000 0000 0000 0000	
ACCAL	0022	ACCAL																0000 0000 0000 0000	
ACCAH	0024	ACCAH																0000 0000 0000 0000	
ACCAU	0026	Sign-Extension (ACCA<39>)										ACCAU						0000 0000 0000 0000	
ACCBL	0028	ACCBL																0000 0000 0000 0000	
ACCBH	002A	ACCBH																0000 0000 0000 0000	
ACCBU	002C	Sign-Extension (ACCB<39>)										ACCBU						0000 0000 0000 0000	
PCL	002E	PCL																0000 0000 0000 0000	
PCH	0030	—	—	—	—	—	—	—	—	—	PCH							0000 0000 0000 0000	
TBLPAG	0032	—	—	—	—	—	—	—	—	TBLPAG								0000 0000 0000 0000	
PSVPAG	0034	—	—	—	—	—	—	—	—	PSVPAG								0000 0000 0000 0000	
RCOUNT	0036	RCOUNT																uuuu uuuu uuuu uuuu	
DCOUNT	0038	DCOUNT																uuuu uuuu uuuu uuuu	
DOSTARTL	003A	DOSTARTL																0	uuuu uuuu uuuu uu0
DOSTARTH	003C	—	—	—	—	—	—	—	—	—	DOSTARTH							0000 0000 0uuu uuuu	
DOENDL	003E	DOENDL																0	uuuu uuuu uuuu uu0
DOENDH	0040	—	—	—	—	—	—	—	—	—	DOENDH							0000 0000 0uuu uuuu	
SR	0042	OA	OB	SA	SB	OAB	SAB	DA	DC	IPL2	IPL1	IPL0	RA	N	OV	Z	C	0000 0000 0000 0000	

Legend: u = uninitialized bit; — = unimplemented bit, read as '0'

Note 1: Refer to the “dsPIC30F Family Reference Manual” (DS70046) for descriptions of register bit fields.

TABLE 3-3: CORE REGISTER MAP⁽¹⁾ (CONTINUED)

SFR Name	Address (Home)	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State
CORCON	0044	—	—	—	US	EDT	DL2	DL1	DL0	SATA	SATB	SATDW	ACCSAT	IPL3	PSV	RND	IF	0000 0000 0010 0000
MODCON	0046	XMODEN	YMODEN	—	—	BWM<3:0>				YWM<3:0>				XWM<3:0>				0000 0000 0000 0000
XMODSRT	0048	XS<15:1>															0	uuuu uuuu uuuu uuu0
XMODEND	004A	XE<15:1>															1	uuuu uuuu uuuu uuu1
YMODSRT	004C	YS<15:1>															0	uuuu uuuu uuuu uuu0
YMODEND	004E	YE<15:1>															1	uuuu uuuu uuuu uuu1
XBREV	0050	BREN	XB<14:0>															uuuu uuuu uuuu uuuu
DISICNT	0052	—	—	DISICNT<13:0>													0000 0000 0000 0000	

Legend: u = uninitialized bit; — = unimplemented bit, read as '0'

Note 1: Refer to the “dsPIC30F Family Reference Manual” (DS70046) for descriptions of register bit fields.

NOTES:

4.1.3 MOVE AND ACCUMULATOR INSTRUCTIONS

Move instructions and the DSP Accumulator class of instructions provide a greater degree of addressing flexibility than other instructions. In addition to the addressing modes supported by most MCU instructions, move and accumulator instructions also support Register Indirect with Register Offset Addressing mode, also referred to as Register Indexed mode.

Note: For the `MOV` instructions, the addressing mode specified in the instruction can differ for the source and destination EA. However, the 4-bit `Wb` (Register Offset) field is shared between both source and destination (but typically only used by one).

In summary, the following addressing modes are supported by move and accumulator instructions:

- Register Direct
- Register Indirect
- Register Indirect Post-Modified
- Register Indirect Pre-Modified
- Register Indirect with Register Offset (Indexed)
- Register Indirect with Literal Offset
- 8-bit Literal
- 16-bit Literal

Note: Not all instructions support all the addressing modes given above. Individual instructions may support different subsets of these addressing modes.

4.1.4 MAC INSTRUCTIONS

The dual source operand DSP instructions (`CLR`, `ED`, `EDAC`, `MAC`, `MPY`, `MPY.N`, `MOVSAC` and `MSC`), also referred to as `MAC` instructions, utilize a simplified set of addressing modes to allow the user to effectively manipulate the Data Pointers through register indirect tables.

The two source operand prefetch registers must be a member of the set {`W8`, `W9`, `W10`, `W11`}. For data reads, `W8` and `W9` will always be directed to the X RAGU and `W10` and `W11` will always be directed to the Y AGU. The effective addresses generated (before and after modification) must, therefore, be valid addresses within X data space for `W8` and `W9` and Y data space for `W10` and `W11`.

Note: Register Indirect with Register Offset Addressing is only available for `W9` (in X space) and `W11` (in Y space).

In summary, the following addressing modes are supported by the `MAC` class of instructions:

- Register Indirect
- Register Indirect Post-Modified by 2
- Register Indirect Post-Modified by 4
- Register Indirect Post-Modified by 6
- Register Indirect with Register Offset (Indexed)

4.1.5 OTHER INSTRUCTIONS

Besides the various addressing modes outlined above, some instructions use literal constants of various sizes. For example, `BRA` (branch) instructions use 16-bit signed literals to specify the branch destination directly, whereas the `DISI` instruction uses a 14-bit unsigned literal field. In some instructions, such as `ADD ACC`, the source of an operand or result is implied by the opcode itself. Certain operations, such as `NOP`, do not have any operands.

4.2 Modulo Addressing

Modulo Addressing is a method of providing an automated means to support circular data buffers using hardware. The objective is to remove the need for software to perform data address boundary checks when executing tightly looped code, as is typical in many DSP algorithms.

Modulo Addressing can operate in either data or program space (since the Data Pointer mechanism is essentially the same for both). One circular buffer can be supported in each of the X (which also provides the pointers into program space) and Y data spaces. Modulo Addressing can operate on any W register pointer. However, it is not advisable to use `W14` or `W15` for Modulo Addressing, since these two registers are used as the Stack Frame Pointer and Stack Pointer, respectively.

In general, any particular circular buffer can only be configured to operate in one direction, as there are certain restrictions on the buffer start address (for incrementing buffers) or end address (for decrementing buffers) based upon the direction of the buffer.

The only exception to the usage restrictions is for buffers which have a power-of-2 length. As these buffers satisfy the start and end address criteria, they may operate in a Bidirectional mode, (i.e., address boundary checks will be performed on both the lower and upper address boundaries).

5.2 Reset Sequence

A Reset is not a true exception, because the interrupt controller is not involved in the Reset process. The processor initializes its registers in response to a Reset, which forces the PC to zero. The processor then begins program execution at location 0x000000. A GOTO instruction is stored in the first program memory location, immediately followed by the address target for the GOTO instruction. The processor executes the GOTO to the specified address and then begins operation at the specified target (start) address.

5.2.1 RESET SOURCES

There are 6 sources of error which will cause a device reset.

- Watchdog Time-out:
The watchdog has timed out, indicating that the processor is no longer executing the correct flow of code.
- Uninitialized W Register Trap:
An attempt to use an uninitialized W register as an Address Pointer will cause a Reset.
- Illegal Instruction Trap:
Attempted execution of any unused opcodes will result in an illegal instruction trap. Note that a fetch of an illegal instruction does not result in an illegal instruction trap if that instruction is flushed prior to execution due to a flow change.
- Brown-out Reset (BOR):
A momentary dip in the power supply to the device has been detected, which may result in malfunction.
- Trap Lockout:
Occurrence of multiple trap conditions simultaneously will cause a Reset.

5.3 Traps

Traps can be considered as non-maskable interrupts, indicating a software or hardware error, which adhere to a predefined priority as shown in Figure 5-1. They are intended to provide the user a means to correct erroneous operation during debug and when operating within the application.

Note: If the user does not intend to take corrective action in the event of a trap error condition, these vectors must be loaded with the address of a default handler that simply contains the RESET instruction. If, on the other hand, one of the vectors containing an invalid address is called, an address error trap is generated.

Note that many of these trap conditions can only be detected when they occur. Consequently, the questionable instruction is allowed to complete prior to trap exception processing. If the user chooses to recover from the error, the result of the erroneous action that caused the trap may have to be corrected.

There are 8 fixed priority levels for traps: Level 8 through Level 15, which implies that the IPL3 is always set during processing of a trap.

If the user is not currently executing a trap, and he sets the IPL<3:0> bits to a value of '0111' (Level 7), then all interrupts are disabled, but traps can still be processed.

5.3.1 TRAP SOURCES

The following traps are provided with increasing priority. However, since all traps can be nested, priority has little effect.

Math Error Trap:

The math error trap executes under the following four circumstances:

1. Should an attempt be made to divide by zero, the divide operation will be aborted on a cycle boundary and the trap taken.
2. If enabled, a math error trap will be taken when an arithmetic operation on either accumulator A or B causes an overflow from bit 31 and the accumulator guard bits are not utilized.
3. If enabled, a math error trap will be taken when an arithmetic operation on either accumulator A or B causes a catastrophic overflow from bit 39 and all saturation is disabled.
4. If the shift amount specified in a shift instruction is greater than the maximum allowed shift amount, a trap will occur.

dsPIC30F3010/3011

6.6.3 LOADING WRITE LATCHES

Example 6-2 shows a sequence of instructions that can be used to load the 96 bytes of write latches. 32 TBLWTL and 32 TBLWTH instructions are needed to load the write latches selected by the Table Pointer.

EXAMPLE 6-2: LOADING WRITE LATCHES

```
; Set up a pointer to the first program memory location to be written
; program memory selected, and writes enabled
MOV    #0x0000,W0                                ;
MOV    W0,TBLPAG                                ; Initialize PM Page Boundary SFR
MOV    #0x6000,W0                                ; An example program memory address
; Perform the TBLWT instructions to write the latches
; 0th_program_word
MOV    #LOW_WORD_0,W2                            ;
MOV    #HIGH_BYTE_0,W3                          ;
TBLWTL W2,[W0]                                   ; Write PM low word into program latch
TBLWTH W3,[W0++]                                 ; Write PM high byte into program latch
; 1st_program_word
MOV    #LOW_WORD_1,W2                            ;
MOV    #HIGH_BYTE_1,W3                          ;
TBLWTL W2,[W0]                                   ; Write PM low word into program latch
TBLWTH W3,[W0++]                                 ; Write PM high byte into program latch
; 2nd_program_word
MOV    #LOW_WORD_2,W2                            ;
MOV    #HIGH_BYTE_2,W3                          ;
TBLWTL W2,[W0]                                   ; Write PM low word into program latch
TBLWTH W3,[W0++]                                 ; Write PM high byte into program latch
.
.
.
; 31st_program_word
MOV    #LOW_WORD_31,W2                          ;
MOV    #HIGH_BYTE_31,W3                        ;
TBLWTL W2,[W0]                                   ; Write PM low word into program latch
TBLWTH W3,[W0++]                                 ; Write PM high byte into program latch
```

Note: In Example 6-2, the contents of the upper byte of W3 have no effect.

6.6.4 INITIATING THE PROGRAMMING SEQUENCE

For protection, the write initiate sequence for NVMKEY must be used to allow any erase or program operation to proceed. After the programming command has been executed, the user must wait for the programming time until programming is complete. The two instructions following the start of the programming sequence should be NOPs.

EXAMPLE 6-3: INITIATING A PROGRAMMING SEQUENCE

```
DISI    #5                                ; Block all interrupts with priority <7
                                              ; for next 5 instructions

MOV     #0x55,W0                          ;
MOV     W0,NVMKEY                         ; Write the 0x55 key
MOV     #0xAA,W1                          ;
MOV     W1,NVMKEY                         ; Write the 0xAA key
BSET    NVMCON,#WR                        ; Start the erase sequence
NOP                                           ; Insert two NOPs after the erase
NOP                                           ; command is asserted
```

TABLE 14-1: QEI REGISTER MAP⁽¹⁾

SFR Name	Addr.	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State
QEICON	0122	CNTERR	—	QEISIDL	INDX	UPDN	QEIM2	QEIM1	QEIM0	SWPAB	—	TQGATE	TQCKPS1	TQCKPS0	POSRES	TQCS	UPDN_SRC	0000 0000 0000 0000
DFLTCON	0124	—	—	—	—	—	IMV1	IMV0	CEID	QEOUT	QECK2	QECK1	QECK0	—	—	—	—	0000 0000 0000 0000
POSCNT	0126	Position Counter<15:0>																0000 0000 0000 0000
MAXCNT	0128	Maximum Count<15:0>																1111 1111 1111 1111
ADPCFG	02A8	—	—	—	—	—	—	—	PCFG8	PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	0000 0000 0000 0000

Legend: — = unimplemented bit, read as '0'

Note 1: Refer to the “dsPIC30F Family Reference Manual” (DS70046) for descriptions of register bit fields.

15.1 PWM Time Base

The PWM time base is provided by a 15-bit timer with a prescaler and postscaler. The time base is accessible via the PTMR SFR. The PTDIR bit (PTMR<15>) is a read-only status bit that indicates the present count direction of the PWM time base. If the PTDIR bit is cleared, PTMR is counting upward. If the PTDIR bit is set, PTMR is counting downward. The PWM time base is configured via the PTCN SFR. The time base is enabled/disabled by setting/clearing the PTEN bit in the PTCN SFR. PTMR is not cleared when the PTEN bit is cleared in software.

The PTPER SFR sets the counting period for PTMR. The user must write a 15-bit value to PTPER<14:0>. When the value in PTMR<14:0> matches the value in PTPER<14:0>, the time base will either reset to 0, or reverse the count direction on the next occurring clock cycle. The action taken depends on the operating mode of the time base.

Note: If the Period register is set to 0x0000, the timer will stop counting, and the interrupt and the Special Event Trigger will not be generated, even if the special event value is also 0x0000. The module will not update the Period register if it is already at 0x0000; therefore, the user must disable the module in order to update the Period register.

The PWM time base can be configured for four different modes of operation:

- Free-Running mode
- Single-Shot mode
- Continuous Up/Down Count mode
- Continuous Up/Down Count mode with interrupts for double updates

These four modes are selected by the PTMOD<1:0> bits in the PTCN SFR. The Continuous Up/Down Count modes support center-aligned PWM generation. The Single-Shot mode allows the PWM module to support pulse control of certain Electronically Commutative Motors (ECMs).

The interrupt signals generated by the PWM time base depend on the mode selection bits (PTMOD<1:0>) and the postscaler bits (PTOPS<3:0>) in the PTCN SFR.

15.1.1 FREE-RUNNING MODE

In the Free-Running mode, the PWM time base counts upwards until the value in the Time Base Period register (PTPER) is matched. The PTMR register is reset on the following input clock edge and the time base will continue to count upwards as long as the PTEN bit remains set.

When the PWM time base is in the Free-Running mode (PTMOD<1:0> = 00), an interrupt event is generated each time a match with the PTPER register occurs and the PTMR register is reset to zero. The postscaler selection bits may be used in this mode of the timer to reduce the frequency of the interrupt events.

15.1.2 SINGLE-SHOT MODE

In the Single-Shot mode, the PWM time base begins counting upwards when the PTEN bit is set. When the value in the PTMR register matches the PTPER register, the PTMR register will be reset on the following input clock edge and the PTEN bit will be cleared by the hardware to halt the time base.

When the PWM time base is in the Single-Shot mode (PTMOD<1:0> = 01), an interrupt event is generated when a match with the PTPER register occurs, the PTMR register is reset to zero on the following input clock edge, and the PTEN bit is cleared. The postscaler selection bits have no effect in this mode of the timer.

15.1.3 CONTINUOUS UP/DOWN COUNT MODES

In the Continuous Up/Down Count modes, the PWM time base counts upwards until the value in the PTPER register is matched. The timer will begin counting downwards on the following input clock edge. The PTDIR bit in the PTCN SFR is read-only and indicates the counting direction. The PTDIR bit is set when the timer counts downwards.

In the Continuous Up/Down Count mode (PTMOD<1:0> = 10), an interrupt event is generated each time the value of the PTMR register becomes zero and the PWM time base begins to count upwards. The postscaler selection bits may be used in this mode of the timer to reduce the frequency of the interrupt events.

TABLE 16-1: SPI1 REGISTER MAP⁽¹⁾

SFR Name	Addr.	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State
SPI1STAT	0220	SPIEN	—	SPISIDL	—	—	—	—	—	—	SPIROV	—	—	—	—	SPITBF	SPIRBF	0000 0000 0000 0000
SPI1CON	0222	—	FRMEN	SPIFSD	—	DISSDO	MODE16	SMP	CKE	SSEN	CKP	MSTEN	SPRE2	SPRE1	SPRE0	PPRE1	PPRE0	0000 0000 0000 0000
SPI1BUF	0224	Transmit and Receive Buffer																0000 0000 0000 0000

Legend: — = unimplemented bit, read as '0'

Note 1: Refer to the “dsPIC30F Family Reference Manual” (DS70046) for descriptions of register bit fields.

18.2 Enabling and Setting Up UART

18.2.1 ENABLING THE UART

The UART module is enabled by setting the UARTEN bit in the UxMODE register (where x = 1 or 2). Once enabled, the UxTX and UxRX pins are configured as an output and an input respectively, overriding the TRIS and LATCH register bit settings for the corresponding I/O port pins. The UxTX pin is at logic '1' when no transmission is taking place.

18.2.2 DISABLING THE UART

The UART module is disabled by clearing the UARTEN bit in the UxMODE register. This is the default state after any Reset. If the UART is disabled, all I/O pins operate as port pins under the control of the LATCH and TRIS bits of the corresponding port pins.

Disabling the UART module resets the buffers to empty states. Any data characters in the buffers are lost and the baud rate counter is reset.

All error and status flags associated with the UART module are reset when the module is disabled. The URXDA, OERR, FERR, PERR, UTXEN, UTXBRK and UTXBF bits are cleared, whereas RIDLE and TRMT are set. Other control bits, including ADDEN, URXISEL<1:0>, UTXISEL, as well as the UxMODE and UxBRG registers, are not affected.

Clearing the UARTEN bit while the UART is active will abort all pending transmissions and receptions and reset the module as defined above. Re-enabling the UART will restart the UART in the same configuration.

18.2.3 ALTERNATE I/O

The alternate I/O function is enabled by setting the ALTIO bit (U1MODE<10>). If ALTIO = 1, the UxATX and UxARX pins (alternate transmit and alternate receive pins, respectively) are used by the UART module instead of the UxTX and UxRX pins. If ALTIO = 0, the UxTX and UxRX pins are used by the UART module.

18.2.4 SETTING UP DATA, PARITY AND STOP BIT SELECTIONS

Control bits, PDSEL<1:0> in the UxMODE register, are used to select the data length and parity used in the transmission. The data length may either be 8 bits with even, odd or no parity, or 9 bits with no parity.

The STSEL bit determines whether one or two Stop bits will be used during data transmission.

The default (power-on) setting of the UART is 8 bits, no parity, 1 Stop bit (typically represented as 8, N, 1).

18.3 Transmitting Data

18.3.1 TRANSMITTING IN 8-BIT DATA MODE

The following steps must be performed in order to transmit 8-bit data:

1. Set up the UART:
First, the data length, parity and number of Stop bits must be selected. Then, the transmit and receive interrupt enable and priority bits are set up in the UxMODE and UxSTA registers. Also, the appropriate baud rate value must be written to the UxBRG register.
2. Enable the UART by setting the UARTEN bit (UxMODE<15>).
3. Set the UTXEN bit (UxSTA<10>), thereby enabling a transmission.
4. Write the byte to be transmitted to the lower byte of UxTXREG. The value will be transferred to the Transmit Shift register (UxTSR) immediately and the serial bit stream will start shifting out during the next rising edge of the baud clock. Alternatively, the data byte may be written while UTXEN = 0, following which, the user may set UTXEN. This will cause the serial bit stream to begin immediately because the baud clock will start from a cleared state.
5. A transmit interrupt will be generated depending on the value of the interrupt control bit, UTXISEL (UxSTA<15>).

18.3.2 TRANSMITTING IN 9-BIT DATA MODE

The sequence of steps involved in the transmission of 9-bit data is similar to 8-bit transmission, except that a 16-bit data word (of which the upper 7 bits are always clear) must be written to the UxTXREG register.

18.3.3 TRANSMIT BUFFER (UxTXB)

The transmit buffer is 9 bits wide and 4 characters deep. Including the Transmit Shift register (UxTSR), the user effectively has a 5-deep FIFO (First In First Out) buffer. The UTXBF Status bit (UxSTA<9>) indicates whether the transmit buffer is full.

If a user attempts to write to a full buffer, the new data will not be accepted into the FIFO, and no data shift will occur within the buffer. This enables recovery from a buffer overrun condition.

The FIFO is reset during any device Reset, but is not affected when the device enters or wakes up from a power-saving mode.

19.8 A/D Acquisition Requirements

The analog input model of the 10-bit ADC is shown in Figure 19-3. The total sampling time for the ADC is a function of the internal amplifier settling time, device V_{DD} and the holding capacitor charge time.

For the ADC 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 (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, 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, R_S , is 5 k Ω . 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.

The user must allow at least 1 TAD period of sampling time, T_{SAMP} , between conversions to allow each sample to be acquired. This sample time may be controlled manually in software by setting/clearing the SAMP bit, or it may be automatically controlled by the ADC. In an automatic configuration, the user must allow enough time between conversion triggers so that the minimum sample time can be satisfied. Refer to the **Section 23.0 “Electrical Characteristics”** for TAD and sample time requirements.

FIGURE 19-3: ADC ANALOG INPUT MODEL

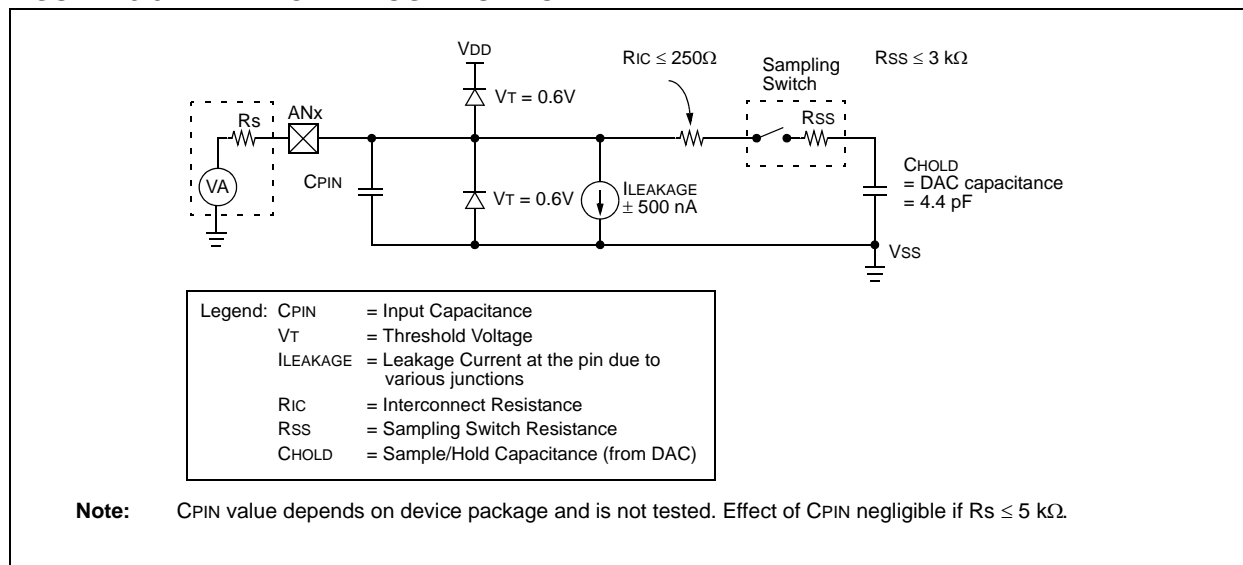


TABLE 21-1: SYMBOLS USED IN OPCODE DESCRIPTIONS (CONTINUED)

Field	Description
Wb	Base W register $\in \{W0..W15\}$
Wd	Destination W register $\in \{Wd, [Wd], [Wd++] , [Wd--], [++Wd], [--Wd] \}$
Wdo	Destination W register $\in \{Wnd, [Wnd], [Wnd++] , [Wnd--], [++Wnd], [--Wnd], [Wnd+Wb] \}$
Wm, Wn	Dividend, Divisor Working register pair (direct addressing)
Wm*Wm	Multiplicand and Multiplier working register pair for Square instructions $\in \{W4*W4, W5*W5, W6*W6, W7*W7\}$
Wm*Wn	Multiplicand and Multiplier working register pair for DSP instructions $\in \{W4*W5, W4*W6, W4*W7, W5*W6, W5*W7, W6*W7\}$
Wn	One of 16 working registers $\in \{W0..W15\}$
Wnd	One of 16 destination working registers $\in \{W0..W15\}$
Wns	One of 16 source working registers $\in \{W0..W15\}$
WREG	W0 (working register used in File register instructions)
Ws	Source W register $\in \{Ws, [Ws], [Ws++] , [Ws--], [++Ws], [--Ws] \}$
Wso	Source W register $\in \{Wns, [Wns], [Wns++] , [Wns--], [++Wns], [--Wns], [Wns+Wb] \}$
Wx	X data space Prefetch Address register for DSP instructions $\in \{[W8] += 6, [W8] += 4, [W8] += 2, [W8], [W8] -= 6, [W8] -= 4, [W8] -= 2, [W9] += 6, [W9] += 4, [W9] += 2, [W9], [W9] -= 6, [W9] -= 4, [W9] -= 2, [W9+W12], \text{none}\}$
Wxd	X data space Prefetch Destination register for DSP instructions $\in \{W4..W7\}$
Wy	Y data space Prefetch Address register for DSP instructions $\in \{[W10] += 6, [W10] += 4, [W10] += 2, [W10], [W10] -= 6, [W10] -= 4, [W10] -= 2, [W11] += 6, [W11] += 4, [W11] += 2, [W11], [W11] -= 6, [W11] -= 4, [W11] -= 2, [W11+W12], \text{none}\}$
Wyd	Y data space Prefetch Destination register for DSP instructions $\in \{W4..W7\}$

TABLE 21-2: INSTRUCTION SET OVERVIEW (CONTINUED)

Base Instr #	Assembly Mnemonic	Assembly Syntax	Description	# of words	# of cycles	Status Flags Affected
80	TBLWTL	TBLWTL <i>Ws</i> , <i>Wd</i>	Write <i>Ws</i> to Prog<15:0>	1	2	None
81	ULNK	ULNK	Unlink Frame Pointer	1	1	None
82	XOR	XOR <i>f</i>	$f = f .XOR. WREG$	1	1	N,Z
		XOR <i>f</i> , <i>WREG</i>	$WREG = f .XOR. WREG$	1	1	N,Z
		XOR <i>#lit10</i> , <i>Wn</i>	$Wd = lit10 .XOR. Wd$	1	1	N,Z
		XOR <i>Wb</i> , <i>Ws</i> , <i>Wd</i>	$Wd = Wb .XOR. Ws$	1	1	N,Z
		XOR <i>Wb</i> , <i>#lit5</i> , <i>Wd</i>	$Wd = Wb .XOR. lit5$	1	1	N,Z
83	ZE	ZE <i>Ws</i> , <i>Wnd</i>	<i>Wnd</i> = Zero-Extend <i>Ws</i>	1	1	C,Z,N

FIGURE 23-8: TIMERQ (QE1 MODULE) EXTERNAL CLOCK TIMING CHARACTERISTICS

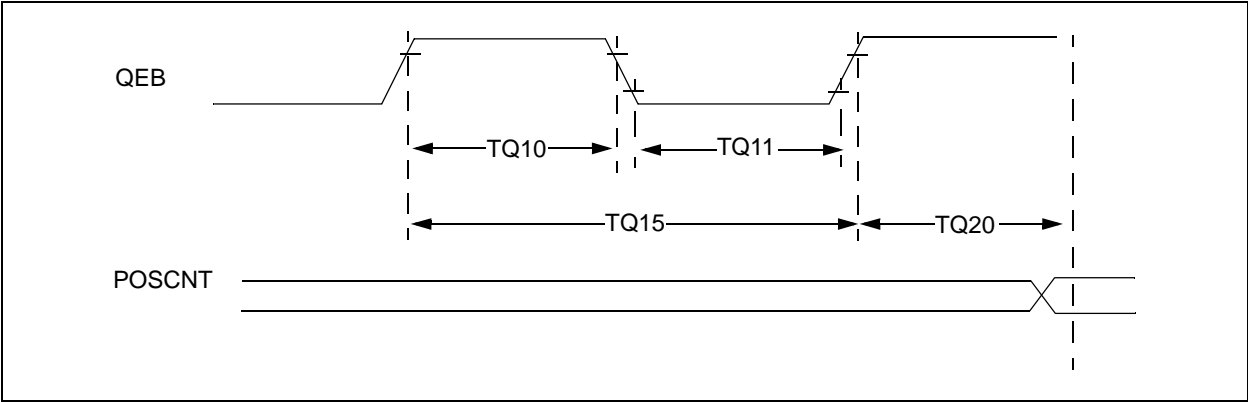


TABLE 23-25: QE1 MODULE EXTERNAL CLOCK TIMING REQUIREMENTS

AC CHARACTERISTICS				Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	Symbol	Characteristic ⁽¹⁾		Min	Typ	Max	Units	Conditions
TQ10	TtQH	TQCK High Time	Synchronous, with prescaler	TcY + 20	—	—	ns	Must also meet parameter TQ15
TQ11	TtQL	TQCK Low Time	Synchronous, with prescaler	TcY + 20	—	—	ns	Must also meet parameter TQ15
TQ15	TtQP	TQCP Input Period	Synchronous, with prescaler	2 * TcY + 40	—	—	ns	
TQ20	TCKEXTMRL	Delay from External TQCK Clock Edge to Timer Increment		0.5 TcY	—	1.5 TcY	—	

Note 1: These parameters are characterized but not tested in manufacturing.

TABLE 23-36: I²C™ BUS DATA TIMING REQUIREMENTS (MASTER MODE)

AC CHARACTERISTICS				Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended			
Param No.	Symbol	Characteristic		Min ⁽¹⁾	Max	Units	Conditions
IM10	TLO:SCL	Clock Low Time	100 kHz mode	Tcy/2 (BRG + 1)	—	μs	
			400 kHz mode	Tcy/2 (BRG + 1)	—	μs	
			1 MHz mode ⁽²⁾	Tcy/2 (BRG + 1)	—	μs	
IM11	THI:SCL	Clock High Time	100 kHz mode	Tcy/2 (BRG + 1)	—	μs	
			400 kHz mode	Tcy/2 (BRG + 1)	—	μs	
			1 MHz mode ⁽²⁾	Tcy/2 (BRG + 1)	—	μs	
IM20	TF:SCL	SDA and SCL Fall Time	100 kHz mode	—	300	ns	Cb is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 Cb	300	ns	
			1 MHz mode ⁽²⁾	—	100	ns	
IM21	TR:SCL	SDA and SCL Rise Time	100 kHz mode	—	1000	ns	Cb is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 Cb	300	ns	
			1 MHz mode ⁽²⁾	—	300	ns	
IM25	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	
			400 kHz mode	100	—	ns	
			1 MHz mode ⁽²⁾	—	—	ns	
IM26	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	μs	
			1 MHz mode ⁽²⁾	—	—	ns	
IM30	TSU:STA	Start Condition Setup Time	100 kHz mode	Tcy/2 (BRG + 1)	—	μs	Only relevant for Repeated Start condition
			400 kHz mode	Tcy/2 (BRG + 1)	—	μs	
			1 MHz mode ⁽²⁾	Tcy/2 (BRG + 1)	—	μs	
IM31	THD:STA	Start Condition Hold Time	100 kHz mode	Tcy/2 (BRG + 1)	—	μs	After this period the first clock pulse is generated
			400 kHz mode	Tcy/2 (BRG + 1)	—	μs	
			1 MHz mode ⁽²⁾	Tcy/2 (BRG + 1)	—	μs	
IM33	TSU:STO	Stop Condition Setup Time	100 kHz mode	Tcy/2 (BRG + 1)	—	μs	
			400 kHz mode	Tcy/2 (BRG + 1)	—	μs	
			1 MHz mode ⁽²⁾	Tcy/2 (BRG + 1)	—	μs	
IM34	THD:STO	Stop Condition Hold Time	100 kHz mode	Tcy/2 (BRG + 1)	—	ns	
			400 kHz mode	Tcy/2 (BRG + 1)	—	ns	
			1 MHz mode ⁽²⁾	Tcy/2 (BRG + 1)	—	ns	
IM40	TAA:SCL	Output Valid From Clock	100 kHz mode	—	3500	ns	
			400 kHz mode	—	1000	ns	
			1 MHz mode ⁽²⁾	—	—	ns	
IM45	TBF:SDA	Bus Free Time	100 kHz mode	4.7	—	μs	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	μs	
			1 MHz mode ⁽²⁾	—	—	μs	
IM50	CB	Bus Capacitive Loading		—	400	pF	

Note 1: BRG is the value of the I²C™ Baud Rate Generator. Refer to **Section 21. “Inter-Integrated Circuit (I²C)”** in the “dsPIC30F Family Reference Manual” (DS70046).

2: Maximum pin capacitance = 10 pF for all I²C pins (for 1 MHz mode only).

dsPIC30F3010/3011

TABLE 23-38: 10-BIT HIGH-SPEED ADC MODULE SPECIFICATIONS

AC CHARACTERISTICS			Standard Operating Conditions: 2.5V to 5.5V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for Industrial -40°C ≤ TA ≤ +125°C for Extended				
Param No.	Symbol	Characteristic ⁽¹⁾	Min.	Typ	Max.	Units	Conditions
Device Supply							
AD01	AVDD	Module VDD Supply	Greater of VDD – 0.3 or 2.7	—	Lesser of VDD + 0.3 or 5.5	V	
AD02	AVSS	Module VSS Supply	VSS – 0.3	—	VSS + 0.3	V	
Reference Inputs							
AD05	VREFH	Reference Voltage High	AVSS + 2.7	—	AVDD	V	
AD06	VREFL	Reference Voltage Low	AVSS	—	AVDD – 2.7	V	
AD07	VREF	Absolute Reference Voltage	AVSS – 0.3	—	AVDD + 0.3	V	
AD08	IREF	Current Drain	—	200 .001	300 3	μA μA	A/D operating A/D off
Analog Input							
AD10	VINH-VINL	Full-Scale Input Span	VREFL		VREFH	V	
AD12	—	Leakage Current	—	±0.001	±0.244	μA	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 5V Source Impedance = 5 kΩ
AD13	—	Leakage Current	—	±0.001	±0.244	μA	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V Source Impedance = 5 kΩ
AD17	RIN	Recommended Impedance of Analog Voltage Source	—	—	5K	Ω	
DC Accuracy⁽²⁾							
AD20	Nr	Resolution	10 data bits			bits	
AD21	INL	Integral Nonlinearity	—	±1	±1	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 5V
AD21A	INL	Integral Nonlinearity	—	±1	±1	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V
AD22	DNL	Differential Nonlinearity	—	±1	±1	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 5V
AD22A	DNL	Differential Nonlinearity	—	±1	±1	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V
AD23	GERR	Gain Error	±1	±5	±6	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 5V
AD23A	GERR	Gain Error	±1	±5	±6	LSb	VINL = AVSS = VREFL = 0V, AVDD = VREFH = 3V

Note 1: These parameters are characterized but not tested in manufacturing.

2: Measurements taken with external VREF+ and VREF- used as the ADC voltage references.

3: The A/D conversion result never decreases with an increase in the input voltage, and has no missing codes.

INDEX

Numerics

10-Bit HighSpeed A/D	
Connection Considerations.....	132
10-Bit High-Speed A/D	
A/D Acquisition Requirements	130
Aborting a Conversion	126
ADCHS	123
ADCON1	123
ADCON2	123
ADCON3	123
ADCSSL.....	123
ADPCFG	123
Configuring Analog Port Pins.....	132
Conversion Operation.....	125
Effects of a Reset.....	131
Operation During CPU Idle Mode	131
Operation During CPU Sleep Mode.....	131
Output Formats	131
Power-Down Modes.....	131
Programming the Start of Conversion Trigger	126
Register Map.....	133
Result Buffer	125
Sampling Requirements.....	130
Selecting the Conversion Sequence.....	125
10-Bit High-Speed Analog-to-Digital Converter (ADC) Module	
123	
16-Bit Up/Down Position Counter Mode	88
Count Direction Status	88
Error Checking.....	88

A

A/D	
1 Msp Configuration Guideline.....	128
600 ksp Configuration Guideline	129
Conversion Rate Parameters.....	127
Selecting the Conversion Clock	126
Voltage Reference Schematic	128
AC Characteristics	172
Load Conditions.....	172
AC Temperature and Voltage Specifications	172
ADC	
750 ksp Configuration Guideline	129
Conversion Speeds.....	127
Address Generator Units	37
Alternate 16-Bit Timer/Counter	89
Alternate Vector Table	47
Assembler	
MPASM Assembler.....	160
Automatic Clock Stretch.....	110
During 10-Bit Addressing (STREN = 1)	110
During 7-Bit Addressing (STREN = 1)	110
Receive Mode	110
Transmit Mode	110

B

Band Gap Start-up Time	
Requirements.....	179
Timing Characteristics	179
Barrel Shifter	24
Bit-Reversed Addressing	40
Example	40
Implementation	40
Modifier Values (table)	41

Sequence Table (16-Entry)	41
Block Diagrams	
10-Bit High-Speed ADC Functional	124
16-Bit Timer1 Module	66
16-Bit Timer4	76
16-Bit Timer5	77
32-Bit Timer4/5	75
Dedicated Port Structure	59
DSP Engine	21
dsPIC30F3010.....	11
dsPIC30F3011.....	10
External Power-on Reset Circuit	143
I ² C	108
Input Capture Mode.....	79
Oscillator System.....	137
Output Compare Mode	83
PWM Module.....	94
Quadrature Encoder Interface	87
Reset System	141
Shared Port Structure.....	60
SPI.....	104
SPI Master/Slave Connection.....	104
UART Receiver.....	116
UART Transmitter.....	115
BOR Characteristics	171
BOR. See Brown-out Reset.	
Brown-out Reset	
Timing Requirements	178
Brown-out Reset (BOR).....	135

C

C Compilers	
MPLAB C18.....	160
Center-Aligned PWM	97
CLKOUT and I/O Timing	
Characteristics.....	177
Requirements	177
Code Examples	
Data EEPROM Block Erase	56
Data EEPROM Block Write	58
Data EEPROM Read.....	55
Data EEPROM Word Erase	56
Data EEPROM Word Write	57
Erasing a Row of Program Memory	51
Initiating a Programming Sequence	52
Loading Write Latches.....	52
Code Protection	135
Complementary PWM Operation.....	98
Configuring Analog Port Pins.....	60
Control Registers	50
NVMADR	50
NVMADRU	50
NVMCON.....	50
NVMKEY	50
Core Overview	17
Core Register Map.....	33
Customer Change Notification Service.....	219
Customer Notification Service	219
Customer Support.....	219

D

Data Access from Program Memory	
Using Program Space Visibility	28
Data Accumulators and Adder/Subtractor	22