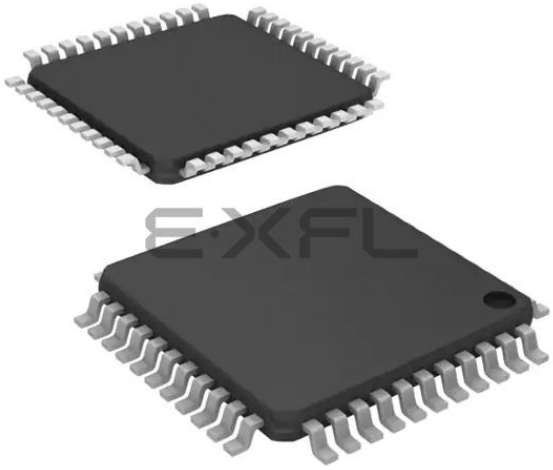


Welcome to E-XFL.COM

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	PIC
Core Size	16-Bit
Speed	32MHz
Connectivity	I²C, IrDA, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, HLVD, POR, PWM, WDT
Number of I/O	39
Program Memory Size	16KB (5.5K x 24)
Program Memory Type	FLASH
EEPROM Size	512 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 16x12b
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/pic24f16ka304t-i-pt

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Web site at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Web site; **http://www.microchip.com**
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our web site at **www.microchip.com** to receive the most current information on all of our products.

TABLE 1-3: PIC24FV32KA304 FAMILY PINOUT DESCRIPTIONS (CONTINUED)

Function	F					FV					I/O	Buffer	Description
	Pin Number					Pin Number							
	20-Pin PDIP/ SSOP/ SOIC	28-Pin SPDIP/ SSOP/ SOIC	28-Pin QFN	44-Pin QFN/ TQFP	48-Pin UQFN	20-Pin PDIP/ SSOP/ SOIC	28-Pin SPDIP/ SSOP/ SOIC	28-Pin QFN	44-Pin QFN/ TQFP	48-Pin UQFN			
CTPLS	16	24	21	11	12	16	24	21	11	12	O	—	CTMU Pulse Output
HLVDIN	15	23	20	10	11	15	23	20	10	11	I	ST	High/Low-Voltage Detect Input
IC1	14	19	16	6	6	11	19	16	6	6	I	ST	Input Capture 1 Input
IC2	13	18	15	5	5	13	18	15	5	5	I	ST	Input Capture 2 Input
IC3	15	23	20	13	14	15	23	20	13	14	I	ST	Input Capture 3 Input
INT0	11	16	13	43	47	11	16	13	43	47	I	ST	Interrupt 0 Input
INT1	17	25	22	14	15	17	25	22	14	15	I	ST	Interrupt 1 Input
INT2	14	20	17	7	7	15	23	20	10	11	I	ST	Interrupt 2 Input
$\overline{\text{MCLR}}$	1	1	26	18	19	1	1	26	18	19	I	ST	Master Clear (Device Reset) Input (active-low)
OC1	14	20	17	7	7	11	16	13	43	47	O	—	Output Compare/PWM1 Output
OC2	4	22	19	4	4	4	22	19	4	4	O	—	Output Compare/PWM2 Output
OC3	5	21	18	12	13	5	21	18	12	13	O	—	Output Compare/PWM3 Output
OCFA	17	25	22	14	15	17	25	22	14	15	O	—	Output Compare Fault A
OFCB	16	24	21	32	35	16	24	21	32	35	O	—	Output Compare Fault B
OSCI	7	9	6	30	33	7	9	6	30	33	I	ANA	Main Oscillator Input
OSCO	8	10	7	31	34	8	10	7	31	34	O	ANA	Main Oscillator Output
PGEC1	5	5	2	22	24	5	5	2	22	24	I/O	ST	ICSP™ Clock 1
PCED1	4	4	1	21	23	4	4	1	21	23	I/O	ST	ICSP Data 1
PGEC2	2	22	19	19	10	2	22	19	19	10	I/O	ST	ICSP Clock 2
PGED2	3	21	18	8	9	3	21	18	8	9	I/O	ST	ICSP Data 2
PGEC3	10	15	12	42	46	10	15	12	42	46	I/O	ST	ICSP Clock 3
PGED3	9	14	11	41	45	9	14	11	41	45	I/O	ST	ICSP Data 3

FIGURE 3-1: PIC24F CPU CORE BLOCK DIAGRAM

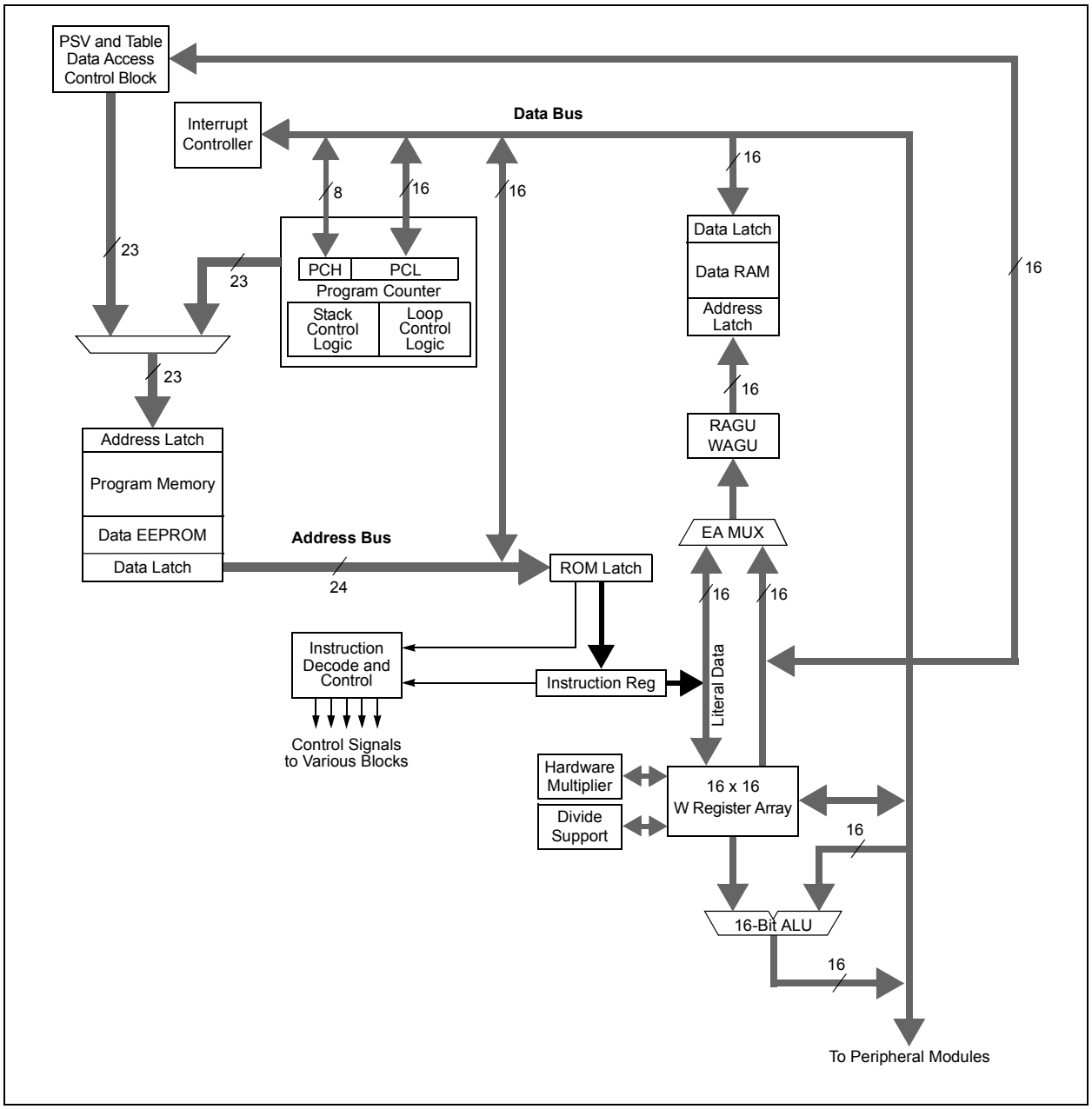


TABLE 3-1: CPU CORE REGISTERS

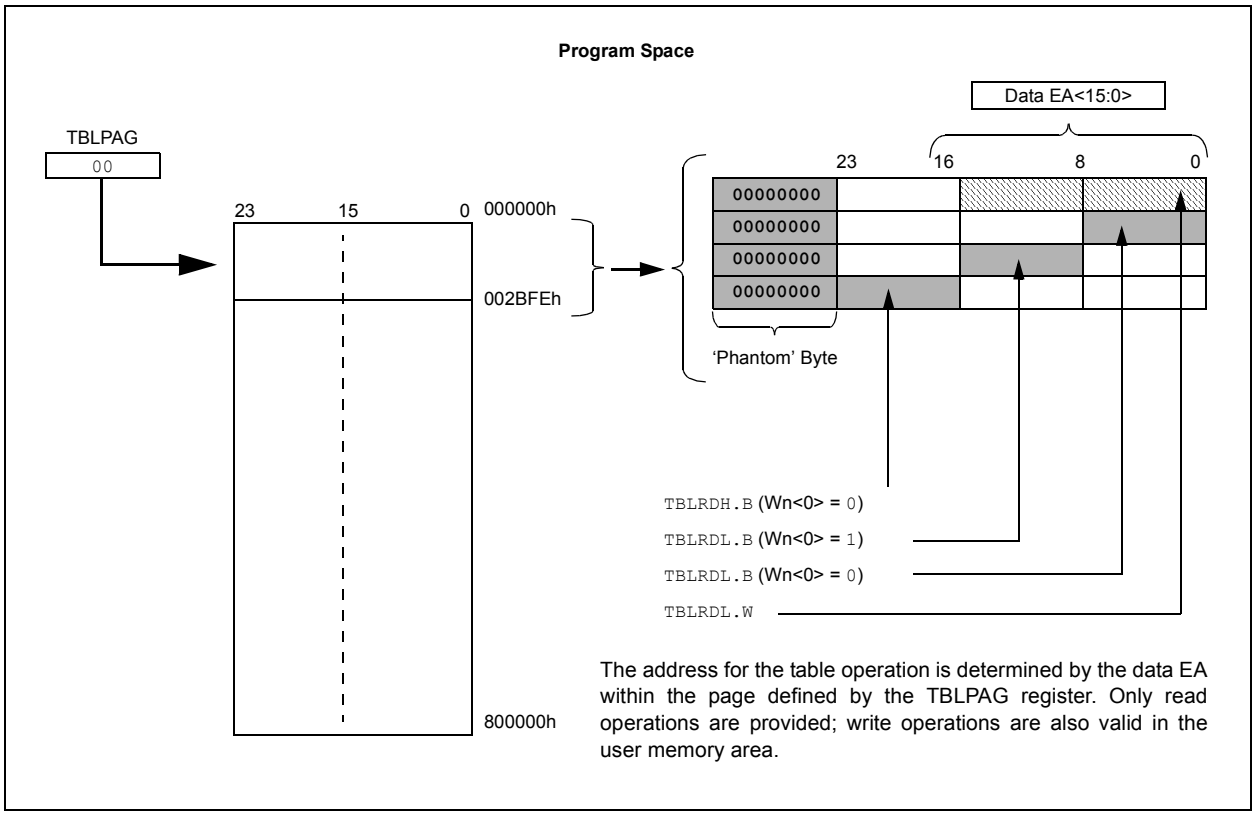
Register(s) Name	Description
W0 through W15	Working Register Array
PC	23-Bit Program Counter
SR	ALU STATUS Register
SPLIM	Stack Pointer Limit Value Register
TBLPAG	Table Memory Page Address Register
PSVPAG	Program Space Visibility Page Address Register
RCOUNT	Repeat Loop Counter Register
CORCON	CPU Control Register

In a similar fashion, two table instructions, TBLWTH and TBLWTL, are used to write individual bytes or words to a program space address. The details of their operation are explained in **Section 5.0 "Flash Program Memory"**.

For all table operations, the area of program memory space to be accessed is determined by the Table Memory Page Address register (TBLPAG). TBLPAG covers the entire program memory space of the device, including user and configuration spaces. When TBLPAG<7> = 0, the table page is located in the user memory space. When TBLPAG<7> = 1, the page is located in configuration space.

Note:	Only Table Read operations will execute in the configuration memory space, and only then, in implemented areas, such as the Device ID. Table write operations are not allowed.

FIGURE 4-6: ACCESSING PROGRAM MEMORY WITH TABLE INSTRUCTIONS



6.4.1 ERASE DATA EEPROM

The data EEPROM can be fully erased, or can be partially erased, at three different sizes: one word, four words or eight words. The bits, NVMOP<1:0> (NVMCON<1:0>), decide the number of words to be erased. To erase partially from the data EEPROM, the following sequence must be followed:

1. Configure NVMCON to erase the required number of words: one, four or eight.
2. Load TBLPAG and WREG with the EEPROM address to be erased.
3. Clear the NVMIF status bit and enable the NVM interrupt (optional).
4. Write the key sequence to NVMKEY.
5. Set the WR bit to begin the erase cycle.
6. Either poll the WR bit or wait for the NVM interrupt (NVMIF is set).

A typical erase sequence is provided in Example 6-2. This example shows how to do a one-word erase. Similarly, a four-word erase and an eight-word erase can be done. This example uses C library procedures to manage the Table Pointer (`builtin_tblpage` and `builtin_tbloffset`) and the Erase Page Pointer (`builtin_tblwtl`). The memory unlock sequence (`builtin_write_NVM`) also sets the WR bit to initiate the operation and returns control when complete.

EXAMPLE 6-2: SINGLE-WORD ERASE

```
int __attribute__ ((space(eedata))) eeData = 0x1234;
/*-----------------------------------------------------------------------------------------------
The variable eeData must be a Global variable declared outside of any method

the code following this comment can be written inside the method that will execute the erase
-----------------------------------------------------------------------------------------------
*/
    unsigned int offset;

    // Set up NVMCON to erase one word of data EEPROM
    NVMCON = 0x4058;

    // Set up a pointer to the EEPROM location to be erased
    TBLPAG = __builtin_tblpage(&eeData);            // Initialize EE Data page pointer
    offset = __builtin_tbloffset(&eeData);          // Initizlize lower word of address
    __builtin_tblwtl(offset, 0);                    // Write EEPROM data to write latch

    asm volatile ("disi #5");                       // Disable Interrupts For 5 Instructions
    __builtin_write_NVM();                          // Issue Unlock Sequence & Start Write Cycle
    while(NVMCONbits.WR=1);                         // Optional: Poll WR bit to wait for
                                                    // write sequence to complete
```

TABLE 8-1: TRAP VECTOR DETAILS

Vector Number	IVT Address	AIVT Address	Trap Source
0	000004h	000104h	Reserved
1	000006h	000106h	Oscillator Failure
2	000008h	000108h	Address Error
3	00000Ah	00010Ah	Stack Error
4	00000Ch	00010Ch	Math Error
5	00000Eh	00010Eh	Reserved
6	000010h	000110h	Reserved
7	000012h	000112h	Reserved

TABLE 8-2: IMPLEMENTED INTERRUPT VECTORS

Interrupt Source	Vector Number	IVT Address	AIVT Address	Interrupt Bit Locations		
				Flag	Enable	Priority
ADC1 Conversion Done	13	00002Eh	00012Eh	IFS0<13>	IEC0<13>	IPC3<6:4>
Comparator Event	18	000038h	000138h	IFS1<2>	IEC1<2>	IPC4<10:8>
CRC Generator	67	00009Ah	00019Ah	IFS4<3>	IEC4<3>	IPC16<14:12>
CTMU	77	0000AEh	0001AEh	IFS4<13>	IEC4<13>	IPC19<6:4>
External Interrupt 0	0	000014h	000114h	IFS0<0>	IEC0<0>	IPC0<2:0>
External Interrupt 1	20	00003Ch	00013Ch	IFS1<4>	IEC1<4>	IPC5<2:0>
External Interrupt 2	29	00004Eh	00014Eh	IFS1<13>	IEC1<13>	IPC7<6:4>
I2C1 Master Event	17	000036h	000136h	IFS1<1>	IEC1<1>	IPC4<6:4>
I2C1 Slave Event	16	000034h	000134h	IFS1<0>	IEC1<0>	IPC4<2:0>
I2C2 Master Event	50	000078h	000178h	IFS3<2>	IEC3<2>	IPC12<10:8>
I2C2 Slave Event	49	000076h	000176h	IFS3<1>	IEC3<1>	IPC12<6:4>
Input Capture 1	1	000016h	000116h	IFS0<1>	IEC0<1>	IPC0<6:4>
Input Capture 2	5	00001Eh	00011Eh	IFS0<5>	IEC0<5>	IPC1<6:4>
Input Capture 3	37	00005Eh	00015Eh	IFS2<5>	IEC2<5>	IPC9<6:4>
Input Change Notification	19	00003Ah	00013Ah	IFS1<3>	IEC1<3>	IPC4<14:12>
HLVD (High/Low-Voltage Detect)	72	0000A4h	0001A4h	IFS4<8>	IEC4<8>	IPC17<2:0>
NVM – NVM Write Complete	15	000032h	000132h	IFS0<15>	IEC0<15>	IPC3<14:12>
Output Compare 1	2	000018h	000118h	IFS0<2>	IEC0<2>	IPC0<10:8>
Output Compare 2	6	000020h	000120h	IFS0<6>	IEC0<6>	IPC1<10:8>
Output Compare 3	25	000046h	000146h	IFS1<9>	IEC1<9>	IPC6<6:4>
Real-Time Clock/Calendar	62	000090h	000190h	IFS3<14>	IEC3<14>	IPC15<10:8>
SPI1 Error	9	000026h	000126h	IFS0<9>	IEC0<9>	IPC2<6:4>
SPI1 Event	10	000028h	000128h	IFS0<10>	IEC0<10>	IPC2<10:8>
SPI2 Error	32	000054h	000154h	IFS2<0>	IEC2<2>	IPC8<2:0>
SPI2 Event	33	000056h	000156h	IFS2<1>	IEC2<1>	IPC8<6:4>
Timer1	3	00001Ah	00011Ah	IFS0<3>	IEC0<3>	IPC0<14:12>
Timer2	7	000022h	000122h	IFS0<7>	IEC0<7>	IPC1<14:12>
Timer3	8	000024h	000124h	IFS0<8>	IEC0<8>	IPC2<2:0>
Timer4	27	00004Ah	00014Ah	IFS1<11>	IEC1<11>	IPC6<14:12>
Timer5	28	00004Ch	00015Ch	IFS1<12>	IEC1<12>	IPC7<2:0>
UART1 Error	65	000096h	000196h	IFS4<1>	IEC4<1>	IPC16<6:4>
UART1 Receiver	11	00002Ah	00012Ah	IFS0<11>	IEC0<11>	IPC2<14:12>
UART1 Transmitter	12	00002Ch	00012Ch	IFS0<12>	IEC0<12>	IPC3<2:0>
UART2 Error	66	000098h	000198h	IFS4<2>	IEC4<2>	IPC16<10:8>
UART2 Receiver	30	000050h	000150h	IFS1<14>	IEC1<14>	IPC7<10:8>
UART2 Transmitter	31	000052h	000152h	IFS1<15>	IEC1<15>	IPC7<14:12>
Ultra Low-Power Wake-up	80	0000B4h	0001B4h	IFS5<0>	IEC5<0>	IPC20<2:0>

REGISTER 8-23: IPC6: INTERRUPT PRIORITY CONTROL REGISTER 6

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	T4IP2	T4IP1	T4IP0	—	—	—	—
bit 15							bit 8

U-0	R/W-1	R/W-0	R/W-0	U-0	U-0	U-0	U-0
—	OC3IP2	OC3IP1	OC3IP0	—	—	—	—
bit 7							bit 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

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

bit 14-12 **T4IP<2:0>:** Timer4 Interrupt Priority bits
111 = Interrupt is Priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is Priority 1
000 = Interrupt source is disabled

bit 11-7 **Unimplemented:** Read as '0'

bit 6-4 **OC3IP<2:0>:** Output Compare Channel 3 Interrupt Priority bits
111 = Interrupt is Priority 7 (highest priority interrupt)
•
•
•
001 = Interrupt is Priority 1
000 = Interrupt source is disabled

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

9.0 OSCILLATOR CONFIGURATION

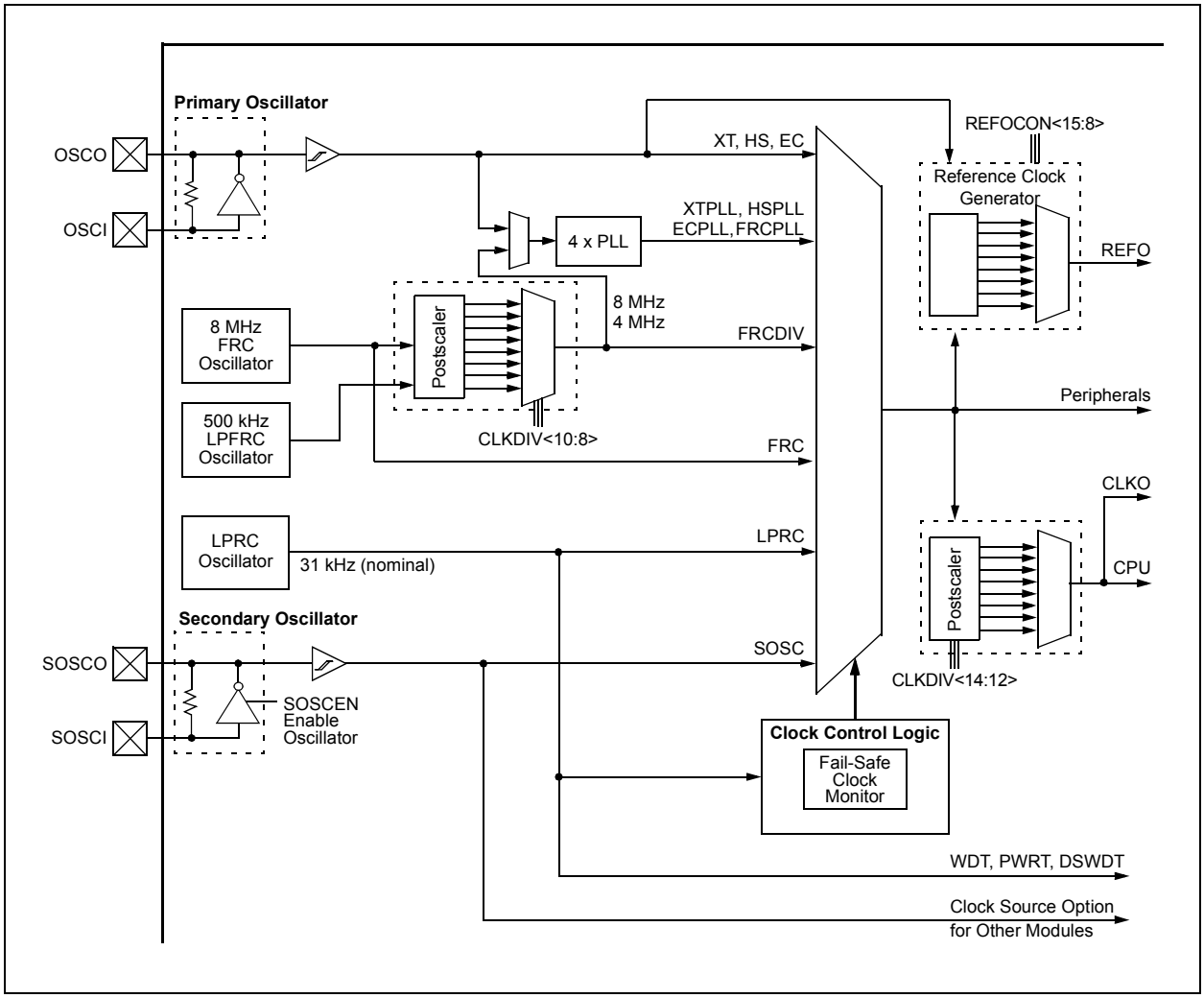
Note: This data sheet summarizes the features of this group of PIC24F devices. It is not intended to be a comprehensive reference source. For more information on Oscillator Configuration, refer to the *"PIC24F Family Reference Manual"*, **Section 38. "Oscillator with 500 kHz Low-Power FRC"** (DS39726).

The oscillator system for the PIC24FV32KA304 family of devices has the following features:

- A total of five external and internal oscillator options as clock sources, providing 11 different clock modes.
- On-chip 4x Phase Locked Loop (PLL) to boost internal operating frequency on select internal and external oscillator sources.
- Software-controllable switching between various clock sources.
- Software-controllable postscaler for selective clocking of CPU for system power savings.
- System frequency range declaration bits for EC mode. When using an external clock source, the current consumption is reduced by setting the declaration bits to the expected frequency range.
- A Fail-Safe Clock Monitor (FSCM) that detects clock failure and permits safe application recovery or shutdown.

A simplified diagram of the oscillator system is shown in Figure 9-1.

FIGURE 9-1: PIC24FV32KA304 FAMILY CLOCK DIAGRAM



10.5 Doze Mode

Generally, changing clock speed and invoking one of the power-saving modes are the preferred strategies for reducing power consumption. There may be circumstances, however, where this is not practical. For example, it may be necessary for an application to maintain uninterrupted synchronous communication, even while it is doing nothing else. Reducing system clock speed may introduce communication errors, while using a power-saving mode may stop communications completely.

Doze mode is a simple and effective alternative method to reduce power consumption while the device is still executing code. In this mode, the system clock continues to operate from the same source and at the same speed. Peripheral modules continue to be clocked at the same speed, while the CPU clock speed is reduced. Synchronization between the two clock domains is maintained, allowing the peripherals to access the SFRs while the CPU executes code at a slower rate.

Doze mode is enabled by setting the DOZEN bit (CLKDIV<11>). The ratio between peripheral and core clock speed is determined by the DOZE<2:0> bits (CLKDIV<14:12>). There are eight possible configurations, from 1:1 to 1:128, with 1:1 being the default.

It is also possible to use Doze mode to selectively reduce power consumption in event driven applications. This allows clock-sensitive functions, such as synchronous communications, to continue without interruption. Meanwhile, the CPU Idles, waiting for something to invoke an interrupt routine. Enabling the automatic return to full-speed CPU operation on interrupts is enabled by setting the ROI bit (CLKDIV<15>). By default, interrupt events have no effect on Doze mode operation.

10.6 Selective Peripheral Module Control

Idle and Doze modes allow users to substantially reduce power consumption by slowing or stopping the CPU clock. Even so, peripheral modules still remain clocked, and thus, consume power. There may be cases where the application needs what these modes do not provide: the allocation of power resources to CPU processing, with minimal power consumption from the peripherals.

PIC24F devices address this requirement by allowing peripheral modules to be selectively disabled, reducing or eliminating their power consumption. This can be done with two control bits:

- The Peripheral Enable bit, generically named, “XXXEN”, located in the module’s main control SFR.
- The Peripheral Module Disable (PMD) bit, generically named, “XXXMD”, located in one of the PMD Control registers.

Both bits have similar functions in enabling or disabling its associated module. Setting the PMDx bits for a module disables all clock sources to that module, reducing its power consumption to an absolute minimum. In this state, the control and status registers associated with the peripheral will also be disabled, so writes to those registers will have no effect, and read values will be invalid. Many peripheral modules have a corresponding PMDx bit.

In contrast, disabling a module by clearing its XXXEN bit, disables its functionality, but leaves its registers available to be read and written to. Power consumption is reduced, but not by as much as the PMDx bits are used. Most peripheral modules have an enable bit; exceptions include capture, compare and RTCC.

To achieve more selective power savings, peripheral modules can also be selectively disabled when the device enters Idle mode. This is done through the control bit of the generic name format, “XXXIDL”. By default, all modules that can operate during Idle mode will do so. Using the disable on Idle feature disables the module while in Idle mode, allowing further reduction of power consumption during Idle mode, enhancing power savings for extremely critical power applications.

14.1.2 CASCADED (32-BIT) MODE

By default, each module operates independently with its own 16-bit timer. To increase resolution, adjacent even and odd modules can be configured to function as a single 32-bit module. (For example, Modules 1 and 2 are paired, as are Modules 3 and 4, and so on.) The odd numbered module (ICx) provides the Least Significant 16 bits of the 32-bit register pairs, and the even numbered module (ICy) provides the Most Significant 16 bits. Wraparounds of the ICx registers cause an increment of their corresponding ICy registers.

Cascaded operation is configured in hardware by setting the IC32 bit (ICxCON2<8>) for both modules.

14.2 Capture Operations

The input capture module can be configured to capture timer values and generate interrupts on rising edges on ICx or all transitions on ICx. Captures can be configured to occur on all rising edges or just some (every 4th or 16th). Interrupts can be independently configured to generate on each event or a subset of events.

To set up the module for capture operations:

1. If Synchronous mode is to be used, disable the Sync source before proceeding.
2. Make sure that any previous data has been removed from the FIFO by reading ICxBUF until the ICBNE bit (ICxCON1<3>) is cleared.
3. Set the SYNCSELx bits (ICxCON2<4:0>) to the desired Sync/trigger source.
4. Set the ICTSELx bits (ICxCON1<12:10>) for the desired clock source. If the desired clock source is running, set the ICTSELx bits before the input capture module is enabled, for proper synchronization with the desired clock source.
5. Set the ICIx bits (ICxCON1<6:5>) to the desired interrupt frequency.
6. Select Synchronous or Trigger mode operation:
   a) Check that the SYNCSELx bits are not set to '00000'.
   b) For Synchronous mode, clear the ICTRIG bit (ICxCON2<7>).
   c) For Trigger mode, set ICTRIG and clear the TRIGSTAT bit (ICxCON2<6>).
7. Set the ICMx bits (ICxCON1<2:0>) to the desired operational mode.
8. Enable the selected Sync/trigger source.

For 32-bit cascaded operations, the setup procedure is slightly different:

1. Set the IC32 bits for both modules (ICyCON2<8> and (ICxCON2<8>), enabling the even numbered module first. This ensures the modules will start functioning in unison.
2. Set the ICTSELx and SYNCSELx bits for both modules to select the same Sync/trigger and time base source. Set the even module first, then the odd module. Both modules must use the same ICTSELx and SYNCSELx bit settings.
3. Clear the ICTRIG bit of the even module (ICyCON2<7>). This forces the module to run in Synchronous mode with the odd module, regardless of its trigger setting.
4. Use the odd module's ICIx bits (ICxCON1<6:5>) to the desired interrupt frequency.
5. Use the ICTRIG bit of the odd module (ICxCON2<7>) to configure Trigger or Synchronous mode operation.

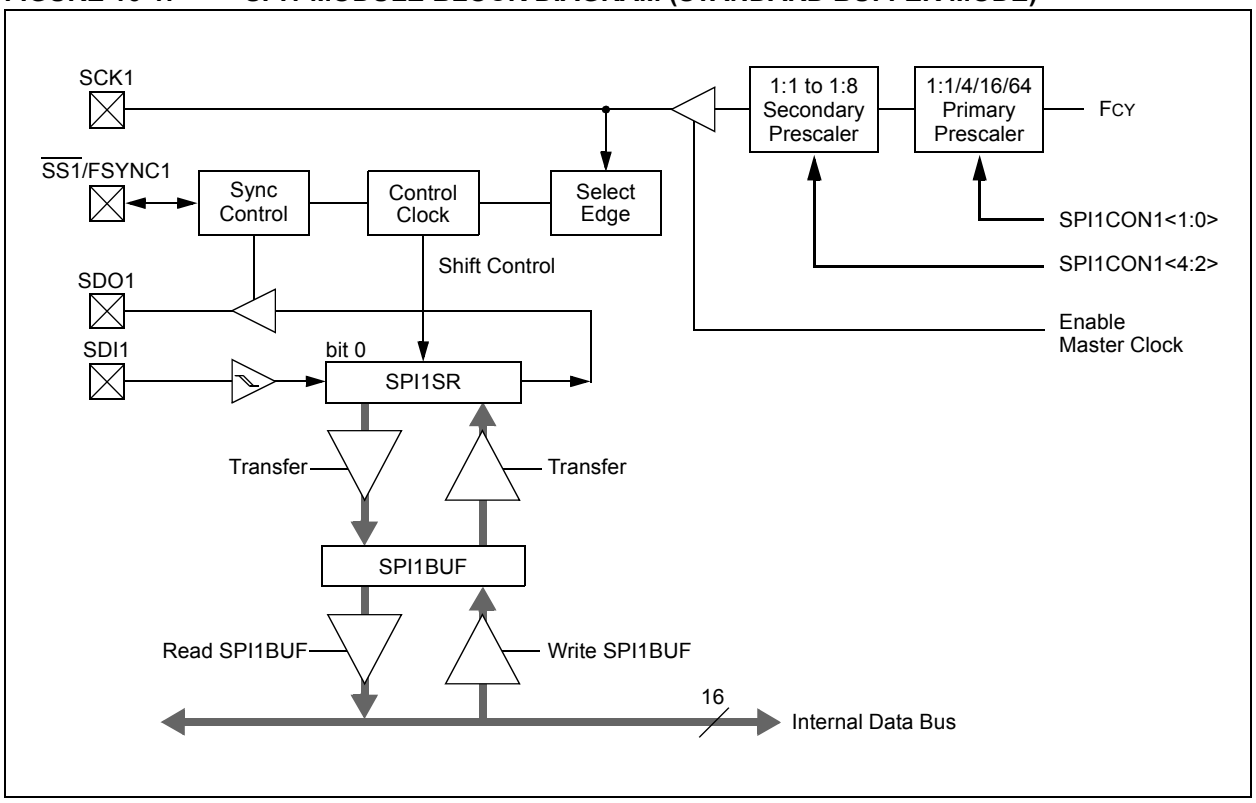
Note:	For Synchronous mode operation, enable the Sync source as the last step. Both input capture modules are held in Reset until the Sync source is enabled.

6. Use the ICMx bits of the odd module (ICxCON1<2:0>) to set the desired capture mode.

The module is ready to capture events when the time base and the Sync/trigger source are enabled. When the ICBNE bit (ICxCON1<3>) becomes set, at least one capture value is available in the FIFO. Read input capture values from the FIFO until the ICBNE clears to '0'.

For 32-bit operation, read both the ICxBUF and ICyBUF for the full 32-bit timer value (ICxBUF for the lsw, ICyBUF for the msw). At least one capture value is available in the FIFO buffer when the odd module's ICBNE bit (ICxCON1<3>) becomes set. Continue to read the buffer registers until ICBNE is cleared (performed automatically by hardware).

FIGURE 16-1: SPI1 MODULE BLOCK DIAGRAM (STANDARD BUFFER MODE)



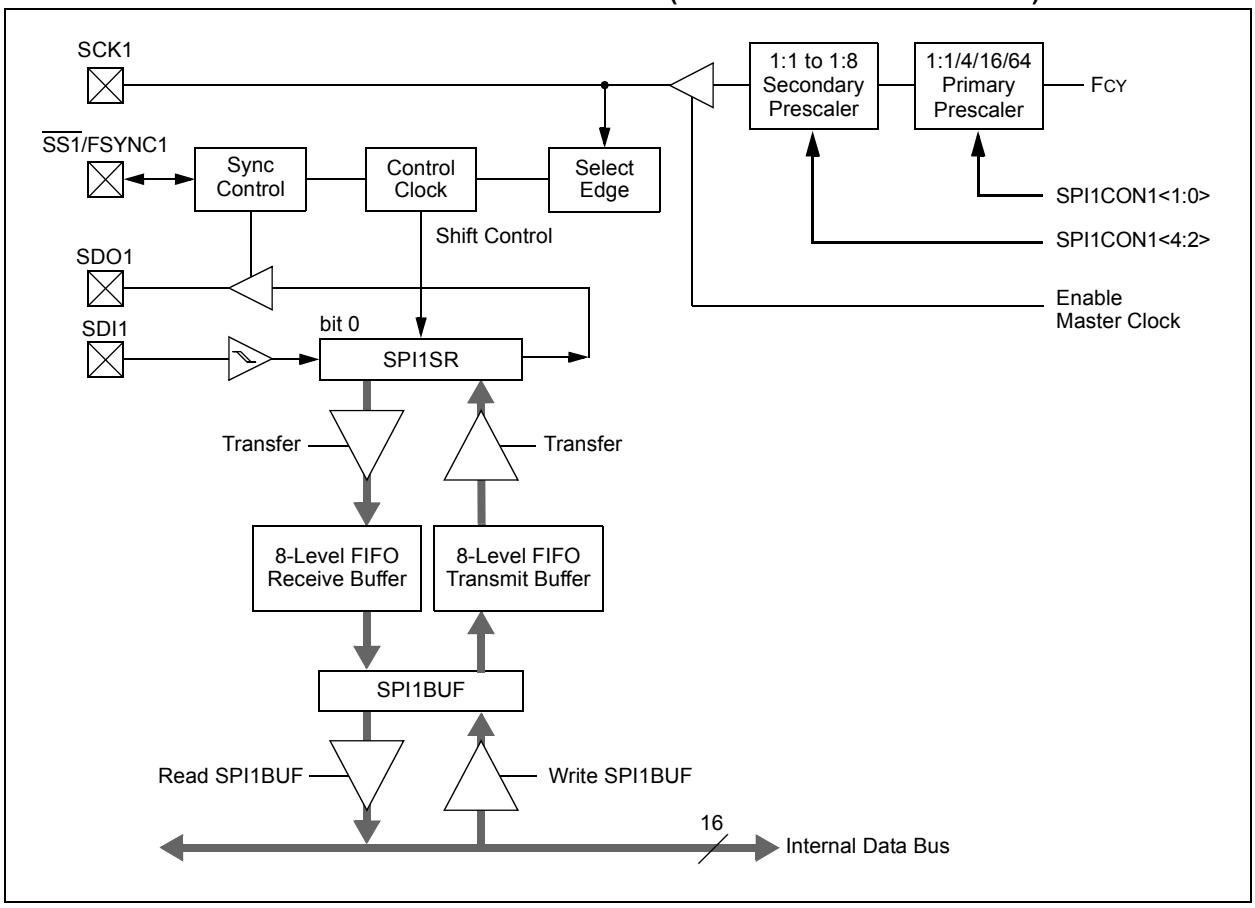
To set up the SPI1 module for the Enhanced Buffer Master (EBM) mode of operation:

1. If using interrupts:
   a) Clear the SPI1IF bit in the IFS0 register.
   b) Set the SPI1IE bit in the IEC0 register.
   c) Write the respective SPI1IPx bits in the IPC2 register.
2. Write the desired settings to the SPI1CON1 and SPI1CON2 registers with the MSTEN bit (SPI1CON1<5>) = 1.
3. Clear the SPIROV bit (SPI1STAT<6>).
4. Select Enhanced Buffer mode by setting the SPIBEN bit (SPI1CON2<0>).
5. Enable SPI operation by setting the SPIEN bit (SPI1STAT<15>).
6. Write the data to be transmitted to the SPI1BUF register. Transmission (and reception) will start as soon as data is written to the SPI1BUF register.

To set up the SPI1 module for the Enhanced Buffer Slave mode of operation:

1. Clear the SPI1BUF register.
2. If using interrupts:
   a) Clear the SPI1IF bit in the IFS0 register.
   b) Set the SPI1IE bit in the IEC0 register.
   c) Write the respective SPI1IPx bits in the IPC2 register to set the interrupt priority.
3. Write the desired settings to the SPI1CON1 and SPI1CON2 registers with the MSTEN bit (SPI1CON1<5>) = 0.
4. Clear the SMP bit.
5. If the CKE bit is set, then the SSEN bit must be set, thus enabling the $\overline{SS1}$ pin.
6. Clear the SPIROV bit (SPI1STAT<6>).
7. Select Enhanced Buffer mode by setting the SPIBEN bit (SPI1CON2<0>).
8. Enable SPI operation by setting the SPIEN bit (SPI1STAT<15>).

FIGURE 16-2: SPI1 MODULE BLOCK DIAGRAM (ENHANCED BUFFER MODE)



REGISTER 16-2: SPIxCON1: SPIx CONTROL REGISTER 1 (CONTINUED)

bit 1-0 **PPRE<1:0>:** Primary Prescale bits (Master mode)
11 = Primary prescale 1:1
10 = Primary prescale 4:1
01 = Primary prescale 16:1
00 = Primary prescale 64:1

Note 1: The CKE bit is not used in the Framed SPI modes. The user should program this bit to '0' for the Framed SPI modes (FRMEN = 1).

REGISTER 16-3: SPIxCON2: SPIx CONTROL REGISTER 2

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
FRMEN	SPIFSD	SPIFPOL	—	—	—	—	—
bit 15							bit 8

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
—	—	—	—	—	—	SPIFE	SPIBEN
bit 7							bit 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

bit 15 **FRMEN:** Framed SPIx Support bit
1 = Framed SPIx support is enabled
0 = Framed SPIx support is disabled

bit 14 **SPIFSD:** SPIx Frame Sync Pulse Direction Control on $\overline{SSx}$ Pin bit
1 = Frame Sync pulse input (slave)
0 = Frame Sync pulse output (master)

bit 13 **SPIFPOL:** SPIx Frame Sync Pulse Polarity bit (Frame mode only)
1 = Frame Sync pulse is active-high
0 = Frame Sync pulse is active-low

bit 12-2 **Unimplemented:** Read as '0'

bit 1 **SPIFE:** SPIx Frame Sync Pulse Edge Select bit
1 = Frame Sync pulse coincides with the first bit clock
0 = Frame Sync pulse precedes the first bit clock

bit 0 **SPIBEN:** SPIx Enhanced Buffer Enable bit
1 = Enhanced buffer is enabled
0 = Enhanced buffer is disabled (Legacy mode)

REGISTER 19-1: RCFGCAL: RTCC CALIBRATION AND CONFIGURATION REGISTER(1) (CONTINUED)

bit 7-0 **CAL<7:0>:** RTC Drift Calibration bits

01111111 = Maximum positive adjustment; adds 508 RTC clock pulses every one minute
.
.
.
00000001 = Minimum positive adjustment; adds 4 RTC clock pulses every one minute
00000000 = No adjustment
11111111 = Minimum negative adjustment; subtracts 4 RTC clock pulses every one minute
.
.
.
10000000 = Maximum negative adjustment; subtracts 512 RTC clock pulses every one minute

Note 1: The RCFGCAL register is only affected by a POR.

2: A write to the RTCEN bit is only allowed when RTCWREN = 1.

3: This bit is read-only; it is cleared to '0' on a write to the lower half of the MINSEC register.

REGISTER 20-1: CRCCON1: CRC CONTROL REGISTER 1

R/W-0	U-0	R/W-0	R-0	R-0	R-0	R-0	R-0
CRCEN	—	CSIDL	VWORD4	VWORD3	VWORD2	VWORD1	VWORD0
bit 15							bit 8

R-0, HSC	R-1, HSC	R/W-0	R/W-0, HC	R/W-0	U-0	U-0	U-0
CRCFUL	CRCMPT	CRCISEL	CRCGO	LENDIAN	—	—	—
bit 7							bit 0

Legend:	HC = Hardware Clearable bit	HSC = Hardware Settable/Clearable bit	
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

bit 15 **CRCEN**: CRC Enable bit
1 = Module is enabled
0 = Module is enabled
All state machines, pointers and CRCWDAT/CRCDAT registers are reset; other SFRs are NOT reset.

bit 14 **Unimplemented:** Read as '0'

bit 13 **CSIDL:** CRC Stop in Idle Mode bit
1 = Discontinues module operation when device enters Idle mode
0 = Continues module operation in Idle mode

bit 12-8 **VWORD<4:0>:** Pointer Value bits
Indicates the number of valid words in the FIFO, which has a maximum value of 8 when PLEN<4:0> > 7 or 16 when PLEN<4:0> ≤ 7.

bit 7 **CRCFUL**: CRC FIFO Full bit
1 = FIFO is full
0 = FIFO is not full

bit 6 **CRCMPT:** CRC FIFO Empty Bit
1 = FIFO is empty
0 = FIFO is not empty

bit 5 **CRCISEL:** CRC interrupt Selection bit
1 = Interrupt on FIFO is empty; CRC calculation is not complete
0 = Interrupt on shift is complete and CRCWDAT result is ready

bit 4 **CRCGO:** Start CRC bit
1 = Starts CRC serial shifter
0 = CRC serial shifter is turned off

bit 3 **LENDIAN:** Data Shift Direction Select bit
1 = Data word is shifted into the CRC, starting with the LSb (little endian)
0 = Data word is shifted into the CRC, starting with the MSb (big endian)

bit 2-0 **Unimplemented:** Read as '0'

REGISTER 22-1: AD1CON1: A/D CONTROL REGISTER 1

R/W-0	U-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0
ADON	—	ADSIDL	—	—	MODE12	FORM1	FORM0
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0, HSC	R/C-0, HSC
SSRC3	SSRC2	SSRC1	SSRC0	—	ASAM	SAMP	DONE
bit 7							bit 0

Legend:	C = Clearable bit	U = Unimplemented bit, read as '0'	
R = Readable bit	W = Writable bit	HSC = Hardware Settable/Clearable bit	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15 **ADON:** A/D Operating Mode bit
1 = A/D Converter module is operating
0 = A/D Converter is off

bit 14 **Unimplemented:** Read as '0'

bit 13 **ADSIDL:** A/D Stop in Idle Mode bit
1 = Discontinues module operation when device enters Idle mode
0 = Continues module operation in Idle mode

bit 12-11 **Unimplemented:** Read as '0'

bit 10 **MODE12:** 12-Bit Operation Mode bit
1 = 12-bit A/D operation
0 = 10-bit A/D operation

bit 9-8 **FORM<1:0>:** Data Output Format bits (see the following formats)
11 = Fractional result, signed, left-justified
10 = Absolute fractional result, unsigned, left-justified
01 = Decimal result, signed, right-justified
00 = Absolute decimal result, unsigned, right-justified

bit 7-4 **SSRC<3:0>:** Sample Clock Source Select bits
1111 = Not available; do not use
•
•
•
1000 = Not available; do not use
0111 = Internal counter ends sampling and starts conversion (auto-convert)
0110 = Not available; do not use
0101 = Timer1 event ends sampling and starts conversion
0100 = CTMU event ends sampling and starts conversion
0011 = Timer5 event ends sampling and starts conversion
0010 = Timer3 event ends sampling and starts conversion
0001 = INT0 event ends sampling and starts conversion
0000 = Clearing the SAMP bit in software ends sampling and begins conversion

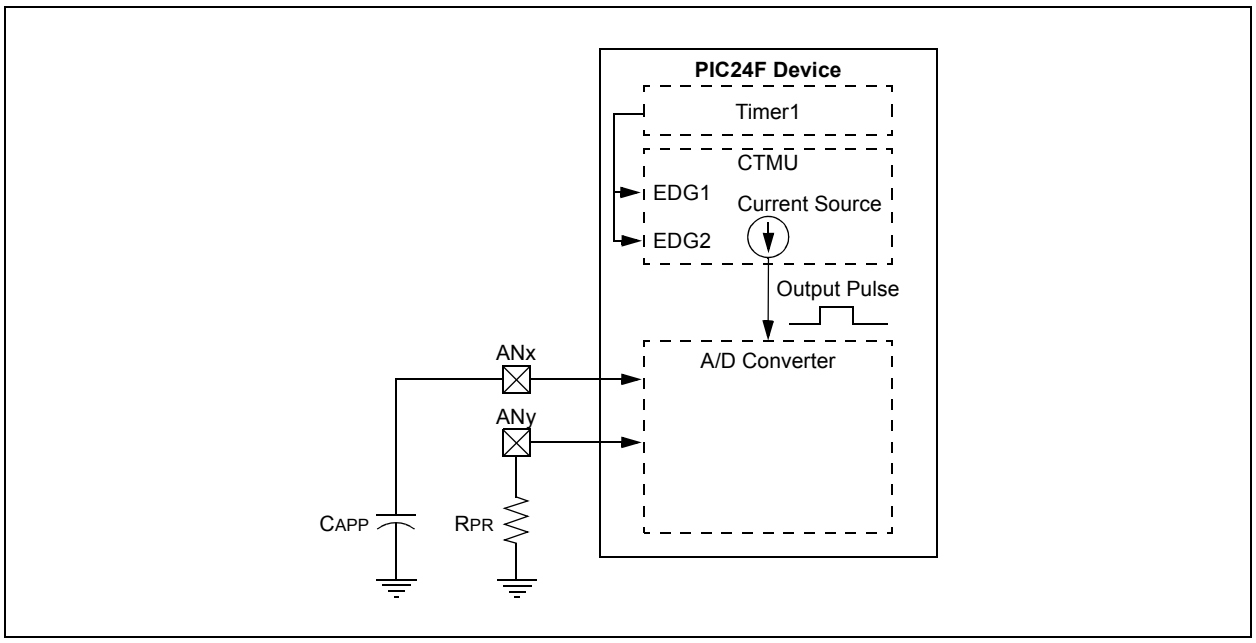
bit 3 **Unimplemented:** Read as '0'

bit 2 **ASAM:** A/D Sample Auto-Start bit
1 = Sampling begins immediately after the last conversion; SAMP bit is auto-set
0 = Sampling begins when the SAMP bit is manually set

bit 1 **SAMP:** A/D Sample Enable bit
1 = A/D Sample-and-Hold amplifiers are sampling
0 = A/D Sample-and-Hold amplifiers are holding

bit 0 **DONE:** A/D Conversion Status bit
1 = A/D conversion cycle has completed
0 = A/D conversion cycle has not started or is in progress

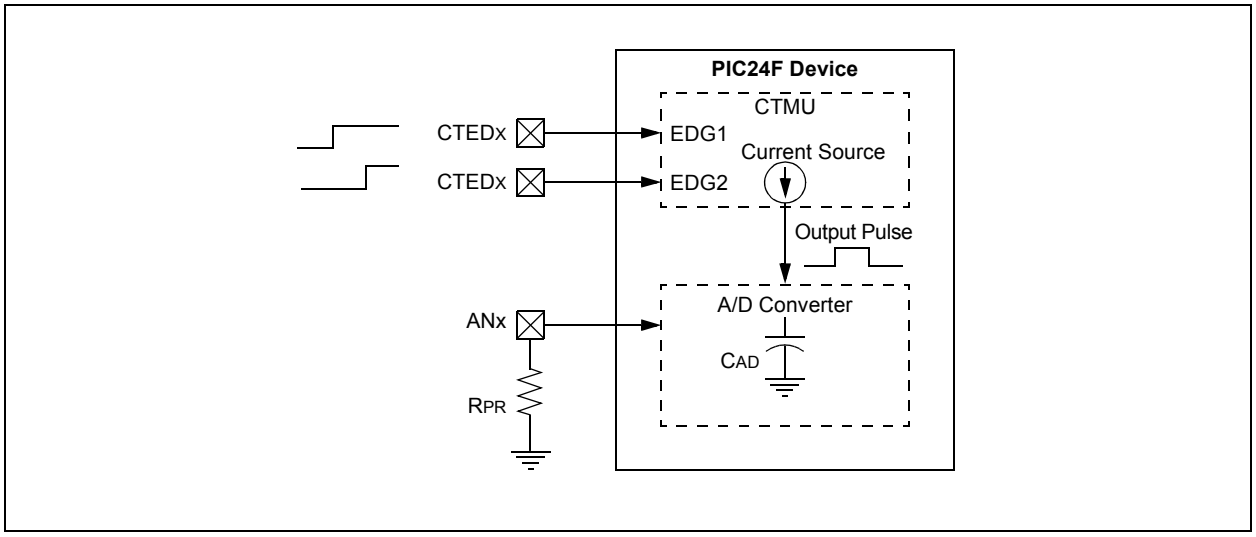
FIGURE 25-1: TYPICAL CONNECTIONS AND INTERNAL CONFIGURATION FOR CAPACITANCE MEASUREMENT



25.2 Measuring Time

Time measurements on the pulse width can be similarly performed using the A/D module's Internal Capacitor (C_{AD}) and a precision resistor for current calibration. Figure 25-2 displays the external connections used for time measurements, and how the CTMU and A/D modules are related in this application. This example also shows both edge events coming from the external CTEDx pins, but other configurations using internal edge sources are possible.

FIGURE 25-2: TYPICAL CONNECTIONS AND INTERNAL CONFIGURATION FOR TIME MEASUREMENT



REGISTER 25-2: CTMUCON2: CTMU CONTROL REGISTER 2 (CONTINUED)

bit 6 **EDG2POL:** Edge 2 Polarity Select bit
1 = Edge 2 is programmed for a positive edge
0 = Edge 2 is programmed for a negative edge

bit 5-2 **EDG2SEL<3:0>:** Edge 2 Source Select bits
1111 = Edge 2 source is Comparator 3 output
1110 = Edge 2 source is Comparator 2 output
1101 = Edge 2 source is Comparator 1 output
1100 = Unimplemented; do not use
1011 = Edge 2 source is IC3
1010 = Edge 2 source is IC2
1001 = Edge 2 source is IC1
1000 = Edge 2 source is CTED13(2)
0111 = Edge 2 source is CTED12(1,2)
0110 = Edge 2 source is CTED11(1,2)
0101 = Edge 2 source is CTED10
0100 = Edge 2 source is CTED9
0011 = Edge 2 source is CTED1
0010 = Edge 2 source is CTED2
0001 = Edge 2 source is OC1
0000 = Edge 2 source is Timer1

bit 1-0 **Unimplemented:** Read as '0'

Note 1: Edge sources, CTED11 and CTED12, are not available on PIC24FV32KA302 devices.
2: Edge sources, CTED3, CTED11, CTED12 and CTED13, are not available on PIC24FV32KA301 devices.

If the WDT is enabled in hardware (FWDTEN<1:0> = 11), it will continue to run during Sleep or Idle modes. When the WDT time-out occurs, the device will wake and code execution will continue from where the PWRSAV instruction was executed. The corresponding SLEEP or IDLE bit (RCON<3:2>) will need to be cleared in software after the device wakes up.

The WDT Flag bit, WDTO (RCON<4>), is not automatically cleared following a WDT time-out. To detect subsequent WDT events, the flag must be cleared in software.

Note: The CLRWDT and PWRSAV instructions clear the prescaler and postscaler counts when executed.

26.3.1 WINDOWED OPERATION

The Watchdog Timer has an optional Fixed Window mode of operation. In this Windowed mode, CLRWDT instructions can only reset the WDT during the last 1/4 of the programmed WDT period. A CLRWDT instruction executed before that window causes a WDT Reset, similar to a WDT time-out.

Windowed WDT mode is enabled by programming the Configuration bit, WINDIS (FWDT<6>), to '0'.

26.3.2 CONTROL REGISTER

The WDT is enabled or disabled by the FWDTEN<1:0> Configuration bits. When both the FWDTEN<1:0> Configuration bits are set, the WDT is always enabled.

The WDT can be optionally controlled in software when the FWDTEN<1:0> Configuration bits have been programmed to '10'. The WDT is enabled in software by setting the SWDTEN control bit (RCON<5>). The SWDTEN control bit is cleared on any device Reset. The software WDT option allows the user to enable the WDT for critical code segments, and disable the WDT during non-critical segments, for maximum power savings. When the FWTEN<1:0> bits are set to '01', the WDT is only enabled in Run and Idle modes, and is disabled in Sleep. Software control of the SWDTEN bit (RCON<5>) is disabled with this setting.

FIGURE 26-2: WDT BLOCK DIAGRAM

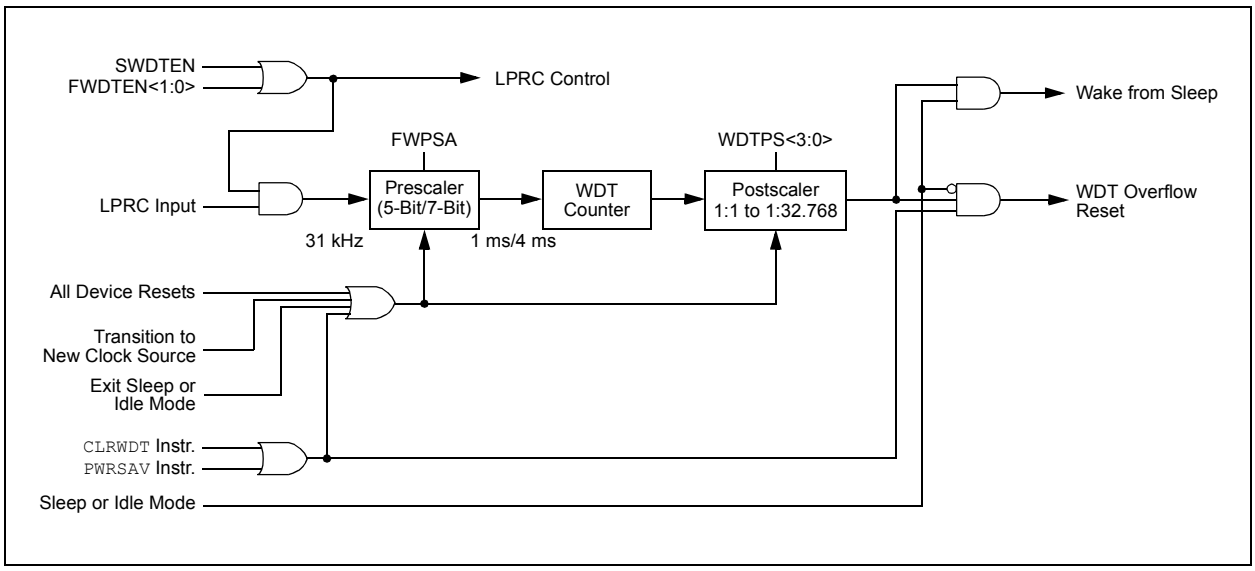


FIGURE 30-18: TYPICAL ΔIBOR vs. VDD

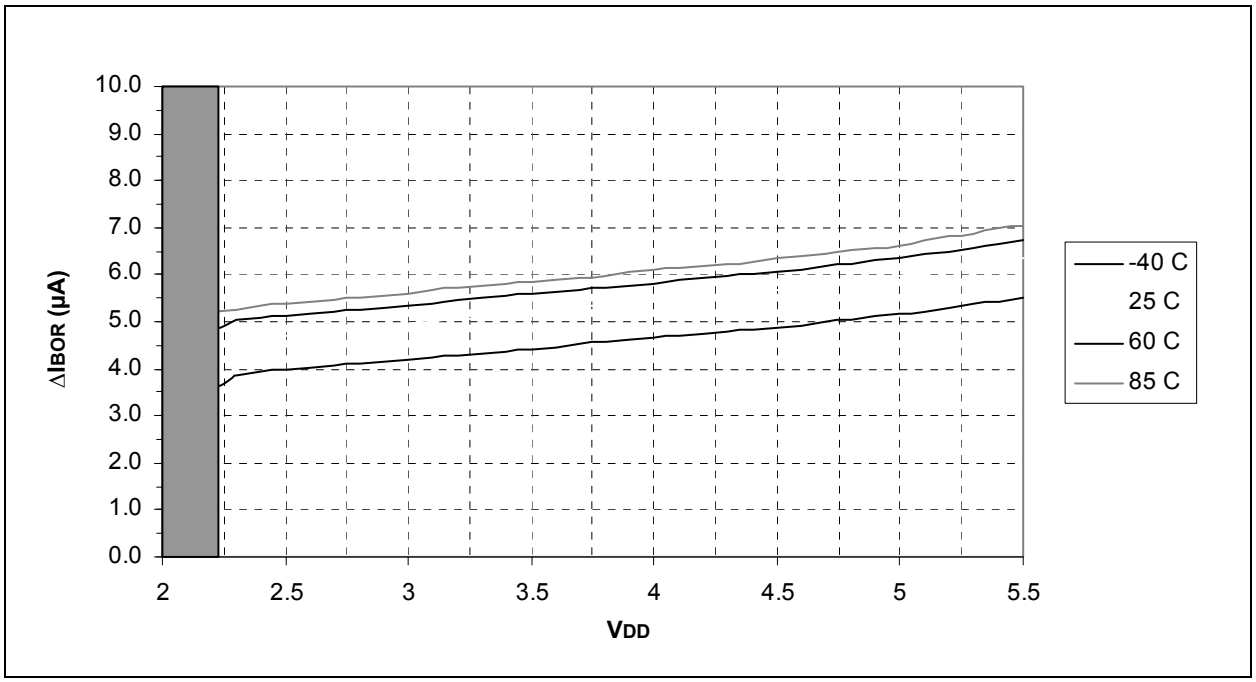


FIGURE 30-19: TYPICAL ΔIWDT vs. VDD