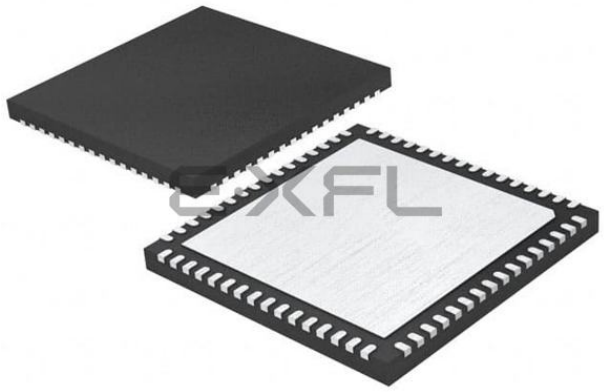


Welcome to E-XFL.COM

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	8-Bit
Speed	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	53
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 16x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-VFQFN Exposed Pad
Supplier Device Package	64-VQFN (9x9)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f65k22t-i-mr

PIC18F87K22 FAMILY

64/80-Pin, High-Performance, 1-Mbit Enhanced Flash MCUs with 12-Bit A/D and nanoWatt XLP Technology

Low-Power Features:

- Power-Managed modes:
  - Run: CPU on, peripherals on
  - Idle: CPU off, peripherals on
  - Sleep: CPU off, peripherals off
- Two-Speed Oscillator Start-up
- Fail-Safe Clock Monitor
- Power-Saving Peripheral Module Disable (PMD)
- Ultra Low-Power Wake-up
- Fast Wake-up, 1 μs Typical
- Low-Power WDT, 300 nA Typical
- Ultra Low 50 nA Input Leakage
- Run mode Currents Down to 5.5 μA, Typical
- Idle mode Currents Down to 1.7 μA Typical
- Sleep mode Currents Down to Very Low 20 nA, Typical
- RTCC Current Downs to Very Low 700 nA, Typical

Special Microcontroller Features:

- Operating Voltage Range: 1.8V to 5.5V
- On-Chip 3.3V Regulator
- Operating Speed up to 64 MHz
- Up to 128 Kbytes On-Chip Flash Program Memory
- Data EEPROM of 1,024 Bytes
- 4K x 8 General Purpose Registers (SRAM)
- 10,000 Erase/Write Cycle Flash Program Memory, Minimum
- 1,000,000 Erase/write Cycle Data EEPROM Memory, Typical
- Flash Retention: 40 Years, Minimum
- Three Internal Oscillators: LF-INTRC (31 kHz), MF-INTOSC (500 kHz) and HF-INTOSC (16 MHz)
- Self-Programmable under Software Control
- Priority Levels for Interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
  - Programmable period from 4 ms to 4,194s (about 70 minutes)
- In-Circuit Serial Programming™ (ICSP™) via Two Pins
- In-Circuit Debug via Two Pins
- Programmable:
  - BOR
  - LVD

Device	Program Memory		Data Memory		I/O	12-Bit A/D (ch)	CCP/ ECCP (PWM)	MSSP			EUSART	Comparators	Timers 8/16-Bit	External Bus	CTMU	RTCC
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)					SPI	Master I^2C™						
PIC18F65K22	32K	16,383	2K	1K	53	16	5/3	2	Y	Y	2	3	4/4	N	Y	Y
PIC18F66K22	64K	32,768	4K	1K	53	16	7/3	2	Y	Y	2	3	6/5	N	Y	Y
PIC18F67K22	128K	65,536	4K	1K	53	16	7/3	2	Y	Y	2	3	6/5	N	Y	Y
PIC18F85K22	32K	16,383	2K	1K	69	24	5/3	2	Y	Y	2	3	4/4	Y	Y	Y
PIC18F86K22	64K	32,768	4K	1K	69	24	7/3	2	Y	Y	2	3	6/5	Y	Y	Y
PIC18F87K22	128K	65,536	4K	1K	69	24	7/3	2	Y	Y	2	3	6/5	Y	Y	Y

3.8 Effects of Power-Managed Modes on the Various Clock Sources

When PRI_IDLE mode is selected, the designated primary oscillator continues to run without interruption. For all other power-managed modes, the oscillator using the OSC1 pin is disabled. The OSC1 pin (and OSC2 pin if used by the oscillator) will stop oscillating.

In secondary clock modes (SEC_RUN and SEC_IDLE), the SOSC oscillator is operating and providing the device clock. The SOSC oscillator may also run in all power-managed modes if required to clock SOSC.

In RC_RUN and RC_IDLE modes, the internal oscillator provides the device clock source. The 31 kHz LF-INTOSC output can be used directly to provide the clock and may be enabled to support various special features, regardless of the power-managed mode (see **Section 28.2 "Watchdog Timer (WDT)"** through **Section 28.5 "Fail-Safe Clock Monitor"** for more information on WDT, Fail-Safe Clock Monitor and Two-Speed Start-up).

If Sleep mode is selected, all clock sources are stopped. Since all the transistor switching currents have been stopped, Sleep mode achieves the lowest current consumption of the device (only leakage currents).

Enabling any on-chip feature that will operate during Sleep will increase the current consumed during Sleep. The INTOSC is required to support WDT operation. The SOSC oscillator may be operating to support a Real-Time Clock (RTC). Other features may be operating that do not require a device clock source (i.e., MSSP slave, INTx pins and others). Peripherals that may add significant current consumption are listed in **Section 31.2 "DC Characteristics: Power-Down and Supply Current PIC18F87K22 Family (Industrial/ Extended)"**.

3.9 Power-up Delays

Power-up delays are controlled by two timers, so that no external Reset circuitry is required for most applications. The delays ensure that the device is kept in Reset until the device power supply is stable under normal circumstances and the primary clock is operating and stable. For additional information on power-up delays, see **Section 5.6 "Power-up Timer (PWRT)"**.

The first timer is the Power-up Timer (PWRT), which provides a fixed delay on a power-up time of about 64 ms (Parameter 33, Table 31-13); it is always enabled.

The second timer is the Oscillator Start-up Timer (OST), intended to keep the chip in Reset until the crystal oscillator is stable (HS, XT or LP modes). The OST does this by counting 1,024 oscillator cycles before allowing the oscillator to clock the device.

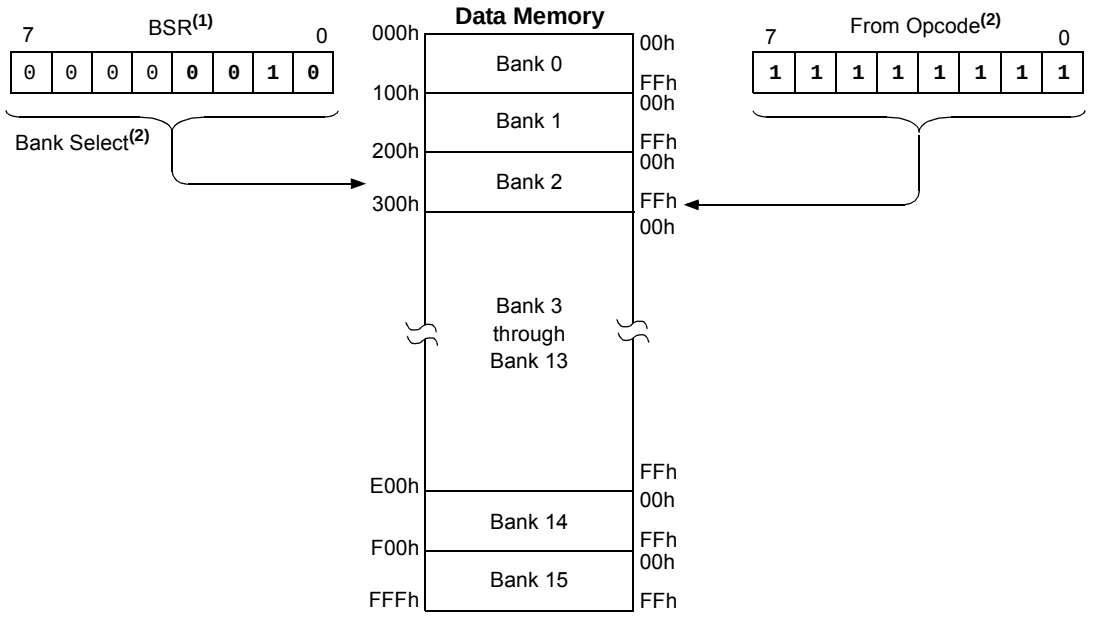
There is a delay of interval, TCSD (Parameter 38, Table 31-13), following POR, while the controller becomes ready to execute instructions.

TABLE 3-4: OSC1 AND OSC2 PIN STATES IN SLEEP MODE

Oscillator Mode	OSC1 Pin	OSC2 Pin
EC, ECPLL	Floating, pulled by external clock	At logic low (clock/4 output)
HS, HSPLL	Feedback inverter is disabled at quiescent voltage level	Feedback inverter is disabled at quiescent voltage level
INTOSC, INTPLL1/2	I/O pin, RA6, direction is controlled by TRISA<6>	I/O pin, RA6, direction is controlled by TRISA<7>

Note: See **Section 5.0 "Reset"** for time-outs due to Sleep and $\overline{\text{MCLR}}$ Reset.

FIGURE 6-7: USE OF THE BANK SELECT REGISTER (DIRECT ADDRESSING)



Note 1: The Access RAM bit of the instruction can be used to force an override of the selected bank (BSR<3:0>) to the registers of the Access Bank.

2: The MOVFF instruction embeds the entire 12-bit address in the instruction.

6.3.2 ACCESS BANK

While the use of the BSR, with an embedded 8-bit address, allows users to address the entire range of data memory, it also means that the user must ensure that the correct bank is selected. If not, data may be read from, or written to, the wrong location. This can be disastrous if a GPR is the intended target of an operation, but an SFR is written to instead. Verifying and/or changing the BSR for each read or write to data memory can become very inefficient.

To streamline access for the most commonly used data memory locations, the data memory is configured with an Access Bank, which allows users to access a mapped block of memory without specifying a BSR. The Access Bank consists of the first 96 bytes of memory (00h-5Fh) in Bank 0 and the last 160 bytes of memory (60h-FFh) in Bank 15. The lower half is known as the “Access RAM” and is composed of GPRs. The upper half is where the device’s SFRs are mapped. These two areas are mapped contiguously in the Access Bank and can be addressed in a linear fashion by an eight-bit address (Figure 6-6).

The Access Bank is used by core PIC18 instructions that include the Access RAM bit (the ‘a’ parameter in the instruction). When ‘a’ is equal to ‘1’, the instruction uses the BSR and the 8-bit address included in the opcode for the data memory address. When ‘a’ is ‘0’, however, the instruction is forced to use the Access Bank address map. In that case, the current value of the BSR is ignored entirely.

Using this “forced” addressing allows the instruction to operate on a data address in a single cycle without updating the BSR first. For 8-bit addresses of 60h and above, this means that users can evaluate and operate on SFRs more efficiently. The Access RAM below 60h is a good place for data values that the user might need to access rapidly, such as immediate computational results or common program variables.

Access RAM also allows for faster and more code efficient context saving and switching of variables.

The mapping of the Access Bank is slightly different when the extended instruction set is enabled (XINST Configuration bit = 1). This is discussed in more detail in **Section 6.6.3 “Mapping the Access Bank in Indexed Literal Offset Mode”**.

6.3.3 GENERAL PURPOSE REGISTER FILE

PIC18 devices may have banked memory in the GPR area. This is data RAM which is available for use by all instructions. GPRs start at the bottom of Bank 0 (address 000h) and grow upwards towards the bottom of the SFR area. GPRs are not initialized by a Power-on Reset and are unchanged on all other Resets.

6.3.4 SPECIAL FUNCTION REGISTERS

The Special Function Registers (SFRs) are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. SFRs start at the top of data memory (FFFh) and extend downward to occupy all of Bank 15 (F00h to FFFh) and the top part of Bank 14 (EF4h to EFFh).

A list of these registers is given in Table 6-1 and Table 6-2.

The SFRs can be classified into two sets: those associated with the “core” device functionality (ALU, Resets and interrupts) and those related to the peripheral functions. The Reset and Interrupt registers are described in their respective chapters, while the ALU’s STATUS register is described later in this section. Registers related to the operation of the peripheral features are described in the chapter for that peripheral.

The SFRs are typically distributed among the peripherals whose functions they control. Unused SFR locations are unimplemented and read as ‘0’s.

TABLE 6-1: SPECIAL FUNCTION REGISTER MAP FOR PIC18F87K22 FAMILY

Addr.	Name	Addr.	Name	Addr.	Name	Addr.	Name	Addr.	Name	Addr.	Name(4)
FFFh	TOSU	FDFh	INDF2(1)	FBFh	ECCP1AS	F9Fh	IPR1	F7Fh	EECON1	F5Fh	RTCCFG
FFEh	TOSH	FDEh	POSTINC2(1)	FBEh	ECCP1DEL	F9Eh	PIR1	F7Eh	EECON2	F5Eh	RTCCAL
FFDh	TOSL	FDDh	POSTDEC2(1)	FBDh	CCPR1H	F9Dh	PIE1	F7Dh	TMR5H	F5Dh	RTCVALH
FFCh	STKPTR	FDCh	PREINC2(1)	FBCh	CCPR1L	F9Ch	PSTR1CON	F7Ch	TMR5L	F5Ch	RTCVALL
FFBh	PCLATU	FDBh	PLUSW2(1)	FBBh	CCP1CON	F9Bh	OSCTUNE	F7Bh	T5CON	F5Bh	ALRMCFG
FFAh	PCLATH	FDAh	FSR2H	FBAh	PIR5	F9Ah	TRISJ(2)	F7Ah	T5GCON	F5Ah	ALRMRPT
FF9h	PCL	FD9h	FSR2L	FB9h	PIE5	F99h	TRISH(2)	F79h	CCPR4H	F59h	ALRMVALH
FF8h	TBLPTRU	FD8h	STATUS	FB8h	IPR4	F98h	TRISG	F78h	CCPR4L	F58h	ALRMVALL
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	PIR4	F97h	TRISF	F77h	CCP4CON	F57h	CTMUCONH
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	PIE4	F96h	TRISE	F76h	CCPR5H	F56h	CTMUCONL
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD	F75h	CCPR5L	F55h	CTMUICONH
FF4h	PRODH	FD4h	SPBRGH1	FB4h	CMSTAT	F94h	TRISC	F74h	CCP5CON	F54h	CM1CON
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB	F73h	CCPR6H	F53h	PADCFG1
FF2h	INTCON	FD2h	IPR5	FB2h	TMR3L	F92h	TRISA	F72h	CCPR6L	F52h	ECCP2AS
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	LATJ(2)	F71h	CCP6CON	F51h	ECCP2DEL
FF0h	INTCON3	FD0h	RCON	FB0h	T3GCON	F90h	LATH(2)	F70h	CCPR7H	F50h	CCPR2H
FEFh	INDF0(1)	FCFh	TMR1H	FAFh	SPBRG1	F8Fh	LATG	F6Fh	CCPR7L	F4Fh	CCPR2L
FEEh	POSTINC0(1)	FCEh	TMR1L	FAEh	RCREG1	F8Eh	LATF	F6Eh	CCP7CON	F4Eh	CCP2CON
FEDh	POSTDEC0(1)	FCDh	T1CON	FADh	TXREG1	F8Dh	LATE	F6Dh	TMR4	F4Dh	ECCP3AS
FECh	PREINC0(1)	FCCh	TMR2	FACh	TXSTA1	F8Ch	LATD	F6Ch	PR4	F4Ch	ECCP3DEL
FEBh	PLUSW0(1)	FCBh	PR2	FABh	RCSTA1	F8Bh	LATC	F6Bh	T4CON	F4Bh	CCPR3H
FEAh	FSR0H	FCAh	T2CON	FAAh	T1GCON	F8Ah	LATB	F6Ah	SSP2BUF	F4Ah	CCPR3L
FE9h	FSR0L	FC9h	SSP1BUF	FA9h	IPR6	F89h	LATA	F69h	SSP2ADD	F49h	CCP3CON
FE8h	WREG	FC8h	SSP1ADD	FA8h	HLVDCON	F88h	PORTJ(2)	F68h	SSP2STAT	F48h	CCPR8H
FE7h	INDF1(1)	FC7h	SSP1STAT	FA7h	PSPCON	F87h	PORTH(2)	F67h	SSP2CON1	F47h	CCPR8L
FE6h	POSTINC1(1)	FC6h	SSP1CON1	FA6h	PIR6	F86h	PORTG	F66h	SSP2CON2	F46h	CCP8CON
FE5h	POSTDEC1(1)	FC5h	SSP1CON2	FA5h	IPR3	F85h	PORTF	F65h	BAUDCON1	F45h	CCPR9H(3)
FE4h	PREINC1(1)	FC4h	ADRESH	FA4h	PIR3	F84h	PORTE	F64h	OSCCON2	F44h	CCPR9L(3)
FE3h	PLUSW1(1)	FC3h	ADRESL	FA3h	PIE3	F83h	PORTD	F63h	EEADRH	F43h	CCP9CON(3)
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC	F62h	EEADR	F42h	CCPR10H(3)
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB	F61h	EEDATA	F41h	CCPR10L(3)
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA	F60h	PIE6	F40h	CCP10CON(3)

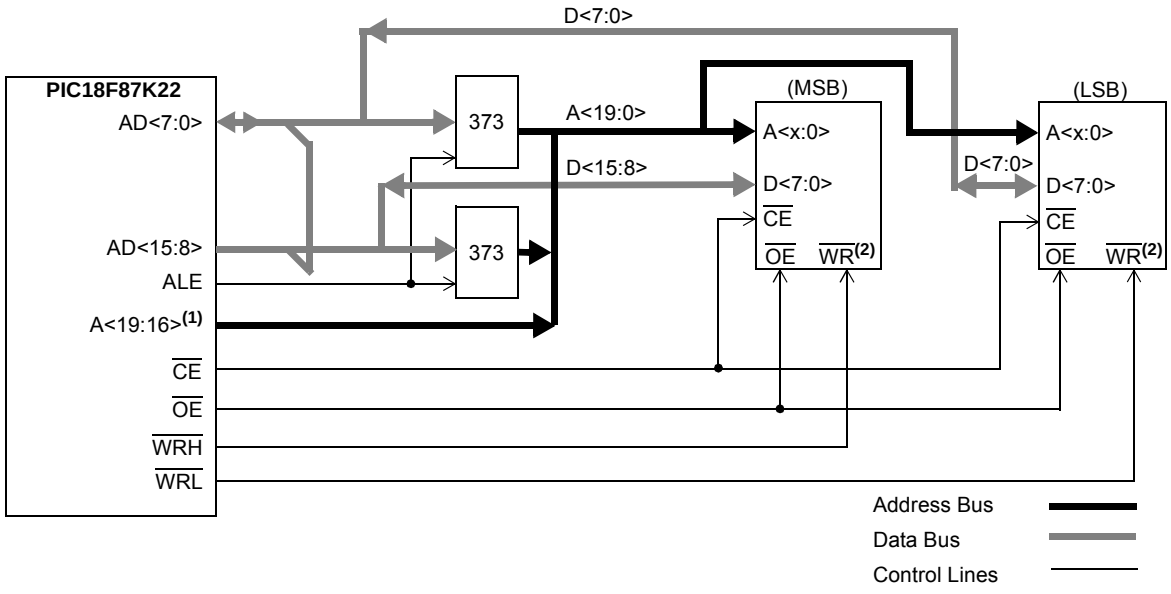
Note 1: This is not a physical register.
2: Unimplemented on 64-pin devices (PIC18F6XK22), read as ‘0’.
3: This register is not available on devices with a program memory of 32 Kbytes (PIC18FX5K22).
4: Addresses, F16h through F5Fh, are also used by SFRs, but are not part of the Access RAM. To access these registers, users must always load the proper BSR value.

8.6.1 16-BIT BYTE WRITE MODE

Figure 8-1 shows an example of 16-Bit Byte Write mode for PIC18F87K22 family devices. This mode is used for two separate 8-bit memories connected for 16-bit operation. This generally includes basic EPROM and Flash devices. It allows table writes to byte-wide external memories.

During a TBLWT instruction cycle, the TABLAT data is presented on the upper and lower bytes of the AD<15:0> bus. The appropriate $\overline{WRH}$ or $\overline{WRL}$ control line is strobed on the LSb of the TBLPTR.

FIGURE 8-1: 16-BIT BYTE WRITE MODE EXAMPLE



Note 1: Upper order address lines are used only for 20-bit address widths.

2: This signal only applies to table writes. See **Section 7.1 "Table Reads and Table Writes"**.

REGISTER 11-3: INTCON3: INTERRUPT CONTROL REGISTER 3

R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
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 7 **INT2IP:** INT2 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **INT1IP:** INT1 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **INT3IE:** INT3 External Interrupt Enable bit
1 = Enables the INT3 external interrupt
0 = Disables the INT3 external interrupt

bit 4 **INT2IE:** INT2 External Interrupt Enable bit
1 = Enables the INT2 external interrupt
0 = Disables the INT2 external interrupt

bit 3 **INT1IE:** INT1 External Interrupt Enable bit
1 = Enables the INT1 external interrupt
0 = Disables the INT1 external interrupt

bit 2 **INT3IF:** INT3 External Interrupt Flag bit
1 = The INT3 external interrupt occurred (must be cleared in software)
0 = The INT3 external interrupt did not occur

bit 1 **INT2IF:** INT2 External Interrupt Flag bit
1 = The INT2 external interrupt occurred (must be cleared in software)
0 = The INT2 external interrupt did not occur

bit 0 **INT1IF:** INT1 External Interrupt Flag bit
1 = The INT1 external interrupt occurred (must be cleared in software)
0 = The INT1 external interrupt did not occur

Note: Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

12.5 PORTD, TRISD and LATD Registers

PORTD is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISD and LATD.

All pins on PORTD are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Note: These pins are configured as digital inputs on any device Reset.

Each of the PORTD pins has a weak internal pull-up. A single control bit can turn off all the pull-ups. This is performed by setting bit, RDPU (PADCFG1<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on all device Resets.

On 80-pin devices, PORTD is multiplexed with the system bus as part of the external memory interface. The I/O port and other functions are only available when the interface is disabled by setting the EBDIS bit (MEMCON<7>). When the interface is enabled, PORTD is the low-order byte of the multiplexed address/data bus (AD<7:0>). The TRISD bits are also overridden.

PORTD can also be configured as an 8-bit wide microprocessor port (Parallel Slave Port) by setting control bit, PSPMODE (TRISE<4>). In this mode, the input buffers are TTL. For additional information, see **Section 12.11 "Parallel Slave Port"**.

The PORTD also has the I^2C and SPI functionality on RD4, RD5 and RD6. The pins for SPI are also configurable for open-drain output. Open-drain configuration is selected by setting bit, SSP2OD (ODCON1<0>).

RD0 has a CTMU functionality. RD1 has the functionality for the Timer5 clock input and Timer7 external clock gate input.

EXAMPLE 12-4: INITIALIZING PORTD

```
CLRF    PORTD   ; Initialize PORTD by
                ; clearing output
                ; data latches
CLRF    LATD    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISD   ; Set RD<3:0> as inputs
                ; RD<5:4> as outputs
                ; RD<7:6> as inputs
```

TABLE 12-7: PORTD FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RD0/PSP0/ AD0/CTPLS	RD0	0	O	DIG	LATD<0> data output.
		1	I	ST	PORTD<0> data input.
	PSP0(1)	x	I/O	TTL	Parallel Slave Port data.
	AD0(2)	x	I/O	TTL	External Memory Address/Data 0.
	CTPLS	x	O	DIG	CTMU pulse generator output.
RD1/PSP1/ AD1/T5CKI/ T7G	RD1	0	O	DIG	LATD<1> data output.
		1	I	ST	PORTD<1> data input.
	PSP1(1)	x	I/O	TTL	Parallel Slave Port data.
	AD1(2)	x	I/O	TTL	External Memory Address/Data 1.
	T5CKI	x	I	ST	Timer5 clock input.
	T7G	x	I	ST	Timer7 external clock gate input.
RD2/PSP2/AD2	RD2	0	O	DIG	LATD<2> data output.
		1	I	ST	PORTD<2> data input.
	PSP2(1)	x	I/O	TTL	Parallel Slave Port data.
	AD2(2)	x	I/O	TTL	External Memory Address/Data 2.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, I^2C = I^2C™/SMBus Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

Note 1: The Parallel Slave Port (PSP) is available only in Microcontroller mode.

2: This feature is available only on PIC18F8XK22 devices.

15.0 TIMER2 MODULE

The Timer2 module incorporates the following features:

- Eight-bit Timer and Period registers (TMR2 and PR2, respectively)
- Both registers are readable and writable
- Software programmable prescaler (1:1, 1:4 and 1:16)
- Software programmable postscaler (1:1 through 1:16)
- Interrupt on TMR2 to PR2 match
- Optional use as the shift clock for the MSSP modules

This module is controlled through the T2CON register (Register 15-1) that enables or disables the timer, and configures the prescaler and postscaler. Timer2 can be shut off by clearing control bit, TMR2ON (T2CON<2>), to minimize power consumption.

A simplified block diagram of the module is shown in Figure 15-1.

15.1 Timer2 Operation

In normal operation, TMR2 is incremented from 00h on each clock (FOSC/4). A four-bit counter/prescaler on the clock input gives the prescale options of direct input, divide-by-4 or divide-by-16. These are selected by the prescaler control bits, T2CKPS<1:0> (T2CON<1:0>).

The value of TMR2 is compared to that of the Period register, PR2, on each clock cycle. When the two values match, the comparator generates a match signal as the timer output. This signal also resets the value of TMR2 to 00h on the next cycle and drives the output counter/postscaler. (See **Section 15.2 "Timer2 Interrupt"**.)

The TMR2 and PR2 registers are both directly readable and writable. The TMR2 register is cleared on any device Reset, while the PR2 register initializes at FFh. Both the prescaler and postscaler counters are cleared on the following events:

- A write to the TMR2 register
- A write to the T2CON register
- Any device Reset – Power-on Reset (POR), MCLR Reset, Watchdog Timer Reset (WDTR) or Brown-out Reset (BOR)

TMR2 is not cleared when T2CON is written.

Note:	The CCP and ECCP modules use Timers, 1 through 8, for some modes. The assignment of a particular timer to a CCP/ECCP module is determined by the Timer to CCP enable bits in the CCPTMRSx registers. For more details, see Register 20-2, Register 19-2 and Register 19-3.

REGISTER 15-1: T2CON: TIMER2 CONTROL REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
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 7 **Unimplemented:** Read as '0'

bit 6-3 **T2OUTPS<3:0>:** Timer2 Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMR2ON:** Timer2 On bit
1 = Timer2 is on
0 = Timer2 is off

bit 1-0 **T2CKPS<1:0>:** Timer2 Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

TABLE 16-5: REGISTERS ASSOCIATED WITH TIMER3/5/7 AS A TIMER/COUNTER

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
PIR5	TMR7GIF(1)	TMR12IF(1)	TMR10IF(1)	TMR8IF	TMR7IF(1)	TMR6IF	TMR5IF	TMR4IF
IPR5	TMR7GIP(1)	TMR12IP(1)	TMR10IP(1)	TMR8IP	TMR7IP(1)	TMR6IP	TMR5IP	TMR4IP
PIE5	TMR7GIE(1)	TMR12IE(1)	TMR10IE(1)	TMR8IE	TMR7IE(1)	TMR6IE	TMR5IE	TMR4IE
PIE3	TMR5GIE	—	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
IPR3	TMR5GIP	—	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
PIR3	TMR5GIF	—	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
PIE2	OSCFIE	—	SSP2IE	BCL2IE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE
PIR2	OSCFIF	—	SSP2IF	BCL2IF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF
IPR2	OSCFIP	—	SSP2IP	BCL2IP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP
TMR3H	Timer3 Register High Byte							
TMR3L	Timer3 Register Low Byte							
T3GCON	TMR3GE	T3GPOL	T3GTM	T3GSPM	T3GGO/$\overline{\text{T3DONE}}$	T3GVAL	T3GSS1	T3GSS0
T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{\text{T3SYNC}}$	RD16	TMR3ON
TMR5H	Timer5 Register High Byte							
TMR5L	Timer5 Register Low Byte							
T5GCON	TMR5GE	T5GPOL	T5GTM	T5GSPM	T5GGO/$\overline{\text{T5DONE}}$	T5GVAL	T5GSS1	T5GSS0
T5CON	TMR5CS1	TMR5CS0	T5CKPS1	T5CKPS0	SOSCEN	$\overline{\text{T5SYNC}}$	RD16	TMR5ON
TMR7H(1)	Timer7 Register High Byte							
TMR7L(1)	Timer7 Register Low Byte							
T7GCON(1)	TMR7GE	T7GPOL	T7GTM	T7GSPM	T7GGO/$\overline{\text{T7DONE}}$	T7GVAL	T7GSS1	T7GSS0
T7CON(1)	TMR7CS1	TMR7CS0	T7CKPS1	T7CKPS0	SOSCEN	$\overline{\text{T7SYNC}}$	RD16	TMR7ON
OSCCON2	—	SOSCRUN	—	—	SOSCGO	—	MFIOFS	MFIOSEL
PMD1	PSPMD	CTMUMD	RTCCMD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	EMBMD
PMD2	TMR10MD(1)	TMR8MD	TMR7MD(1)	TMR6MD	TMR5MD	CMP3MD	CMP2MD	CMP2MD

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the Timer3/5/7 module.

Note 1: Unimplemented on devices with a program memory of 32 Kbytes (PIC18FX5K22).

REGISTER 18-17: ALRMHR: ALARM HOURS VALUE REGISTER(1)

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
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 7-6 **Unimplemented:** Read as '0'

bit 5-4 **HRTEN<1:0>:** Binary Coded Decimal Value of Hour's Tens Digit bits
Contains a value from 0 to 2.

bit 3-0 **HRONE<3:0>:** Binary Coded Decimal Value of Hour's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 18-18: ALRMMIN: ALARM MINUTES VALUE REGISTER

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
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 7 **Unimplemented:** Read as '0'

bit 6-4 **MINTEN<2:0>:** Binary Coded Decimal Value of Minute's Tens Digit bits
Contains a value from 0 to 5.

bit 3-0 **MINONE<3:0>:** Binary Coded Decimal Value of Minute's Ones Digit bits
Contains a value from 0 to 9.

REGISTER 18-19: ALRMSEC: ALARM SECONDS VALUE REGISTER

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
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 7 **Unimplemented:** Read as '0'

bit 6-4 **SECTEN<2:0>:** Binary Coded Decimal Value of Second's Tens Digit bits
Contains a value from 0 to 5.

bit 3-0 **SECONE<3:0>:** Binary Coded Decimal Value of Second's Ones Digit bits
Contains a value from 0 to 9.

TABLE 18-4: ALRMVAL REGISTER MAPPING

ALRMPTR<1:0>	Alarm Value Register Window	
	ALRMVALH	ALRMVALL
00	ALRMMIN	ALRMSEC
01	ALRMWD	ALRMHR
10	ALRMMNTH	ALRMDAY
11	—	—

18.2.9 CALIBRATION

The real-time crystal input can be calibrated using the periodic auto-adjust feature. When properly calibrated, the RTCC can provide an error of less than three seconds per month.

To perform this calibration, find the number of error clock pulses and store the value into the lower half of the RTCCAL register. The eight-bit, signed value, loaded into RTCCAL, is multiplied by four and will be either added or subtracted from the RTCC timer, once every minute.

To calibrate the RTCC module:

1. Use another timer resource on the device to find the error of the 32.768 kHz crystal.
2. Convert the number of error clock pulses per minute (see Equation 18-1).

EQUATION 18-1: CONVERTING ERROR CLOCK PULSES

(Ideal Frequency (32,758) – Measured Frequency) * 60 = Error Clocks per Minute

   - If the oscillator is *faster* than ideal (negative result from Step 2), the RCFGCALL register value needs to be negative. This causes the specified number of clock pulses to be subtracted from the timer counter, once every minute.
   - If the oscillator is *slower* than ideal (positive result from Step 2), the RCFGCALL register value needs to be positive. This causes the specified number of clock pulses to be added to the timer counter, once every minute.
3. Load the RTCCAL register with the correct value.

Writes to the RTCCAL register should occur only when the timer is turned off or immediately after the rising edge of the seconds pulse.

Note: In determining the crystal's error value, it is the user's responsibility to include the crystal's initial error from drift due to temperature or crystal aging.

18.3 Alarm

The Alarm features and characteristics are:

- Configurable from half a second to one year
- Enabled using the ALRMEN bit (ALRMCFG<7>, Register 18-4)
- Offers one-time and repeat alarm options

18.3.1 CONFIGURING THE ALARM

The alarm feature is enabled using the ALRMEN bit. This bit is cleared when an alarm is issued. The bit will not be cleared if the CHIME bit = 1 or if ALRMRPT ≠ 0.

The interval selection of the alarm is configured through the ALRMCFG bits (AMASK<3:0>); see Figure 18-5. These bits determine which, and how many, digits of the alarm must match the clock value for the alarm to occur.

The alarm can also be configured to repeat based on a preconfigured interval. The number of times this occurs, after the alarm is enabled, is stored in the ALRMRPT register.

Note: While the alarm is enabled (ALRMEN = 1), changing any of the registers, other than the RTCCAL, ALRMCFG and ALRMRPT registers and the CHIME bit, can result in a false alarm event leading to a false alarm interrupt. To avoid this, only change the timer and alarm values while the alarm is disabled (ALRMEN = 0). It is recommended that the ALRMCFG and ALRMRPT registers and CHIME bit be changed when RTCSYNC = 0.

REGISTER 20-2: CCPTMRS0: CCP TIMER SELECT 0 REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C3TSEL1	C3TSEL0	C2TSEL2	C2TSEL1	C2TSEL0	C1TSEL2	C1TSEL1	C1TSEL0
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 7-6 **C3TSEL<1:0>:** ECCP3 Timer Selection bits
00 = ECCP3 is based off of TMR1/TMR2
01 = ECCP3 is based off of TMR3/TMR4
10 = ECCP3 is based off of TMR3/TMR6
11 = ECCP3 is based off of TMR3/TMR8

bit 5-3 **C2TSEL<2:0>:** ECCP2 Timer Selection bits
000 = ECCP2 is based off of TMR1/TMR2
001 = ECCP2 is based off of TMR3/TMR4
010 = ECCP2 is based off of TMR3/TMR6
011 = ECCP2 is based off of TMR3/TMR8
100 = ECCP2 is based off of TMR3/TMR10: option reserved on the 32-Kbyte device variant; do not use
101 = Reserved; do not use
110 = Reserved; do not use
111 = Reserved; do not use

bit 2-0 **C1TSEL<2:0>:** ECCP1 Timer Selection bits
000 = ECCP1 is based off of TMR1/TMR2
001 = ECCP1 is based off of TMR3/TMR4
010 = ECCP1 is based off of TMR3/TMR6
011 = ECCP1 is based off of TMR3/TMR8
100 = ECCP1 is based off of TMR3/TMR10: option reserved on the 32-Kbyte device variant; do not use
101 = ECCP1 is based off of TMR3/TMR12: option reserved on the 32-Kbyte device variant; do not use
110 = Reserved; do not use
111 = Reserved; do not use

REGISTER 23-2: ADCON1: A/D CONTROL REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TRIGSEL1	TRIGSEL0	VCFG1	VCFG0	VNCFG	CHSN2	CHSN1	CHSN0
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 7-6 **TRIGSEL<1:0>:** Special Trigger Select bits
11 = Selects the special trigger from the RTCC
10 = Selects the special trigger from the Timer1
01 = Selects the special trigger from the CTMU
00 = Selects the special trigger from the ECCP2

bit 5-4 **VCFG<1:0>:** A/D V_{REF}+ Configuration bits
11 = Internal V_{REF}+ (4.096V)
10 = Internal V_{REF}+ (2.048V)
01 = External V_{REF}+
00 = AV_{DD}

bit 3 **VNCFG:** A/D V_{REF}- Configuration bit
1 = External V_{REF}
0 = AV_{SS}

bit 2-0 **CHSN<2:0>:** Analog Negative Channel Select bits
111 = Channel 07 (AN6)
110 = Channel 06 (AN5)
101 = Channel 05 (AN4)
100 = Channel 04 (AN3)
011 = Channel 03 (AN2)
010 = Channel 02 (AN1)
001 = Channel 01 (AN0)
000 = Channel 00 (AV_{SS})

27.5 Measuring Capacitance with the CTMU

There are two ways to measure capacitance with the CTMU. The absolute method measures the actual capacitance value. The relative method only measures for any change in the capacitance.

27.5.1 ABSOLUTE CAPACITANCE MEASUREMENT

For absolute capacitance measurements, both the current and capacitance calibration steps found in **Section 27.4 "Calibrating the CTMU Module"** should be followed.

To perform these measurements:

1. Initialize the A/D Converter.
2. Initialize the CTMU.
3. Set EDG1STAT.
4. Wait for a fixed delay, T.
5. Clear EDG1STAT.
6. Perform an A/D conversion.
7. Calculate the total capacitance, $C_{TOTAL} = (I * T)/V$, where:
   - I is known from the current source measurement step (**Section 27.4.1 "Current Source Calibration"**)
   - T is a fixed delay
   - V is measured by performing an A/D conversion
8. Subtract the stray and A/D capacitance (C_{OFFSET} from **Section 27.4.2 "Capacitance Calibration"**) from C_{TOTAL} to determine the measured capacitance.

27.5.2 RELATIVE CHARGE MEASUREMENT

Not all applications require precise capacitance measurements. When detecting a valid press of a capacitance-based switch, only a relative change of capacitance needs to be detected.

In such an application, when the switch is open (or not touched), the total capacitance is the capacitance of the combination of the board traces, the A/D Converter and other elements. A larger voltage will be measured by the A/D Converter. When the switch is closed (or touched), the total capacitance is larger due to the addition of the capacitance of the human body to the above listed capacitances and a smaller voltage will be measured by the A/D Converter.

To detect capacitance changes simply:

1. Initialize the A/D Converter and the CTMU.
2. Set EDG1STAT.
3. Wait for a fixed delay.
4. Clear EDG1STAT.
5. Perform an A/D conversion.

The voltage measured by performing the A/D conversion is an indication of the relative capacitance. In this case, no calibration of the current source or circuit capacitance measurement is needed. (For a sample software routine for a capacitive touch switch, see Example 27-4.)

27.6 Measuring Time with the CTMU Module

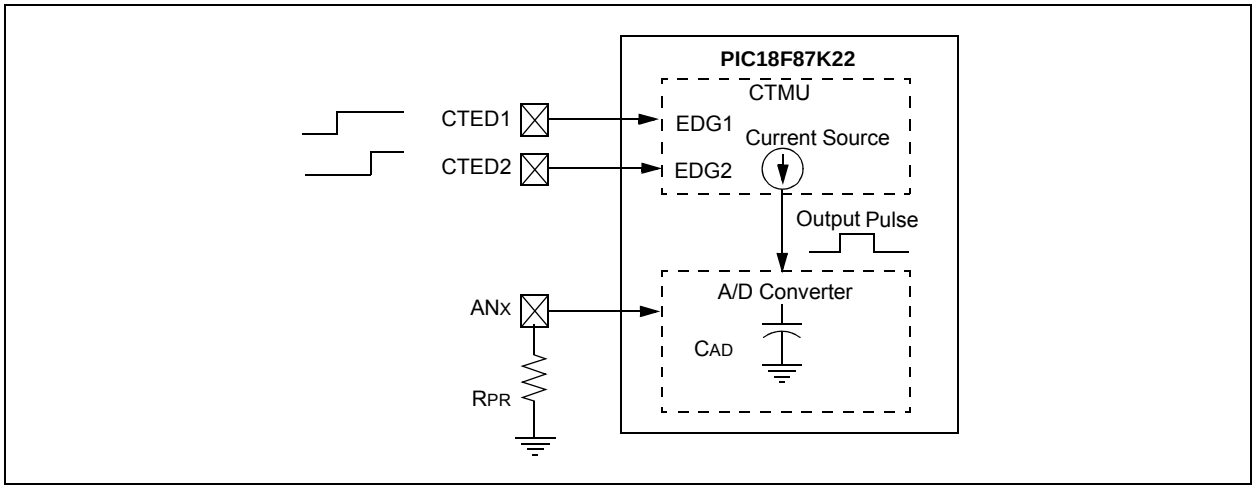
Time can be precisely measured after the ratio (C/I) is measured from the current and capacitance calibration step. To do that:

1. Initialize the A/D Converter and the CTMU.
2. Set EDG1STAT.
3. Set EDG2STAT.
4. Perform an A/D conversion.
5. Calculate the time between edges as $T = (C/I) * V$, where:
   - I is calculated in the current calibration step (**Section 27.4.1 "Current Source Calibration"**)
   - C is calculated in the capacitance calibration step (**Section 27.4.2 "Capacitance Calibration"**)
   - V is measured by performing the A/D conversion

It is assumed that the time measured is small enough that the capacitance, C_{OFFSET}, provides a valid voltage to the A/D Converter. For the smallest time measurement, always set the A/D Channel Select register (AD1CHS) to an unused A/D channel, the corresponding pin for which is not connected to any circuit board trace. This minimizes added stray capacitance, keeping the total circuit capacitance close to that of the A/D Converter itself (25 pF).

To measure longer time intervals, an external capacitor may be connected to an A/D channel and that channel selected whenever making a time measurement.

FIGURE 27-3: TYPICAL CONNECTIONS AND INTERNAL CONFIGURATION FOR TIME MEASUREMENT



REGISTER 28-8: CONFIG5L: CONFIGURATION REGISTER 5 LOW (BYTE ADDRESS 300008h)

R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1	R/C-1
CP7(1)	CP6(1)	CP5(1)	CP4(1)	CP3	CP2	CP1	CP0
bit 7							bit 0

Legend:	C = 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 7 **CP7:** Code Protection bit(1)
1 = Block 7 is not code-protected(2)
0 = Block 7 is code-protected(2)

bit 6 **CP6:** Code Protection bit(1)
1 = Block 6 is not code-protected(2)
0 = Block 6 is code-protected(2)

bit 5 **CP5:** Code Protection bit(1)
1 = Block 5 is not code-protected(2)
0 = Block 5 is code-protected(2)

bit 4 **CP4:** Code Protection bit(1)
1 = Block 4 is not code-protected(2)
0 = Block 4 is code-protected(2)

bit 3 **CP3:** Code Protection bit
1 = Block 3 is not code-protected(2)
0 = Block 3 is code-protected(2)

bit 2 **CP2:** Code Protection bit
1 = Block 2 is not code-protected(2)
0 = Block 2 is code-protected(2)

bit 1 **CP1:** Code Protection bit
1 = Block 1 is not code-protected(2)
0 = Block 1 is code-protected(2)

bit 0 **CP0:** Code Protection bit
1 = Block 0 is not code-protected(2)
0 = Block 0 is code-protected(2)

Note 1: This bit is available only on PIC18F67K22 and PIC18F87K22 devices.
2: For the memory size of the blocks, see Figure 28-6.

REGISTER 28-9: CONFIG5H: CONFIGURATION REGISTER 5 HIGH (BYTE ADDRESS 300009h)

R/C-1	R/C-1	U-0	U-0	U-0	U-0	U-0	U-0
CPD	CPB	—	—	—	—	—	—
bit 7							bit 0

Legend:	C = 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 7 **CPD:** Data EEPROM Code Protection bit
1 = Data EEPROM is not code-protected
0 = Data EEPROM is code-protected

bit 6 **CPB:** Boot Block Code Protection bit
1 = Boot block is not code-protected(1)
0 = Boot block is code-protected(1)

bit 5-0 **Unimplemented:** Read as '0'

Note 1: For the memory size of the blocks, see Figure 28-6. The boot block size changes with BBSIZ0.

BTFSC **Bit Test File, Skip if Clear**

Syntax: BTFSC f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: skip if (f<b>) = 0

Status Affected: None

Encoding:

1011	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '0', then the next instruction is skipped. If bit 'b' is '0', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```
Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (TRUE)
If FLAG<1> = 1;
PC = address (FALSE)

BTFSS **Bit Test File, Skip if Set**

Syntax: BTFSS f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

Operation: skip if (f<b>) = 1

Status Affected: None

Encoding:

1010	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '1', then the next instruction is skipped. If bit 'b' is '1', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```
Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (FALSE)
If FLAG<1> = 1;
PC = address (TRUE)

CPFSGT **Compare f with W, Skip if f > W**

Syntax: CPFSGT f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) > (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	010a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of the W by performing an unsigned subtraction.

If the contents of 'f' are greater than the contents of WREG, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE      CPFSGT REG, 0
NGREATER  :
GREATER   :
```
Before Instruction
PC = Address (HERE)
W = ?
After Instruction
If REG > W;
PC = Address (GREATER)
If REG ≤ W;
PC = Address (NGREATER)

CPFSLT **Compare f with W, Skip if f < W**

Syntax: CPFSLT f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) < (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	000a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.

If the contents of 'f' are less than the contents of W, then the fetched instruction is discarded and a NOP is executed instead, making this a two-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

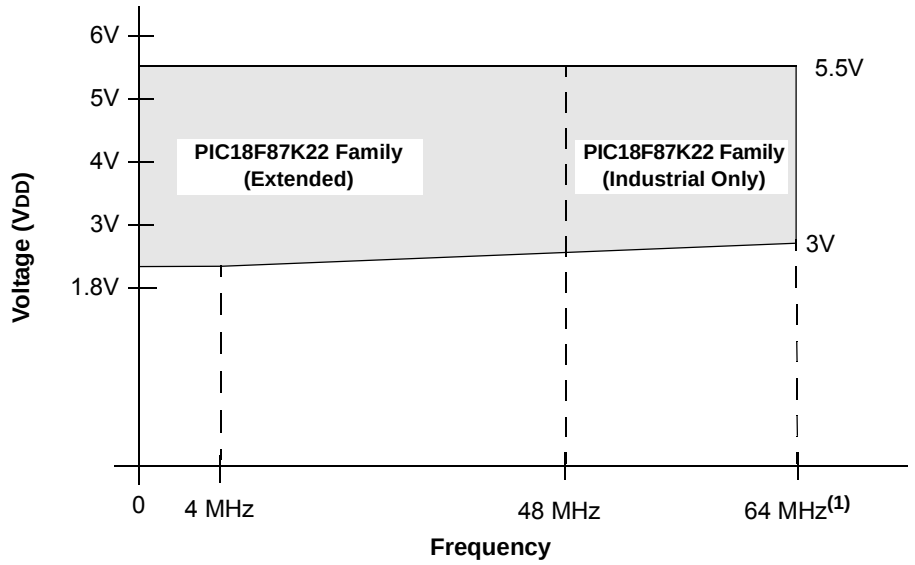
Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    CPFSLT REG, 1
NLESS   :
LESS    :
```
Before Instruction
PC = Address (HERE)
W = ?
After Instruction
If REG < W;
PC = Address (LESS)
If REG ≥ W;
PC = Address (NLESS)

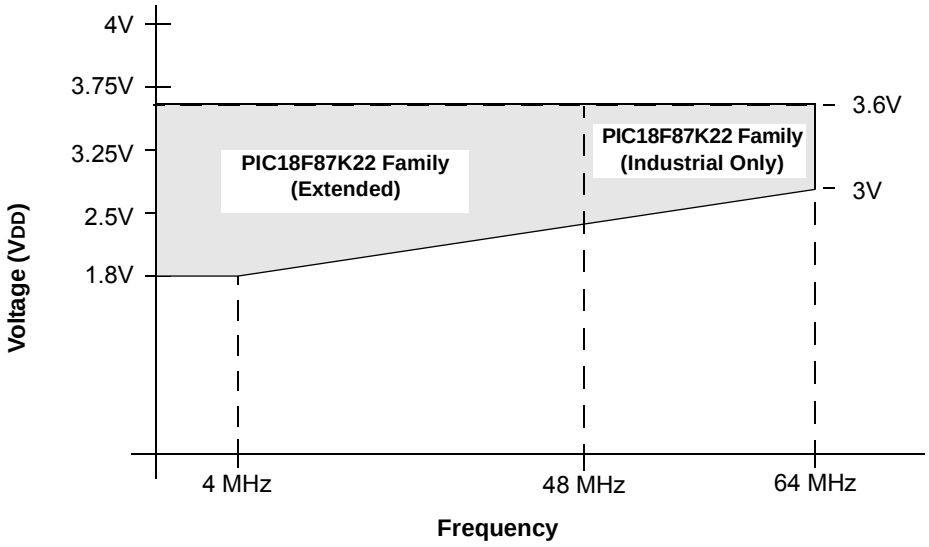
FIGURE 31-1: VOLTAGE-FREQUENCY GRAPH, REGULATOR ENABLED (INDUSTRIAL/EXTENDED)(1)



Note 1: FMAX = 25 MHz in 8-Bit External Memory mode. For VDD values, 1.8V to 3V, FMAX = (VDD – 1.72)/0.02 MHz.

2: FMAX = 64 MHz in all other modes. For VDD values, 1.8V to 3V, FMAX = (VDD – 1.72)/0.02 MHz.

FIGURE 31-2: VOLTAGE-FREQUENCY GRAPH, REGULATOR DISABLED (INDUSTRIAL/EXTENDED)(1,2)



Note 1: When the on-chip voltage regulator is disabled, VDD must be maintained so that VDD ≤ 3.6V.

2: For VDD values, 1.8V to 3V, FMAX = (VDD – 1.72)/0.02 MHz.