



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	25MHz
Connectivity	UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	16
Program Memory Size	4KB (2K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	256 x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 7x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VQFN Exposed Pad
Supplier Device Package	28-QFN (6x6)
Purchase URL	<a href="https://www.e-xfl.com/product-detail/microchip-technology/pic18f1220-e-ml">https://www.e-xfl.com/product-detail/microchip-technology/pic18f1220-e-ml</a>

---

[illegible]

<b>Legend:</b>	TTL = TTL compatible input	CMOS = CMOS compatible input or output
	ST = Schmitt Trigger input with CMOS levels	I = Input
	O = Output	P = Power
	OD = Open-drain (no P diode to VDD)	

**TABLE 2-2: CAPACITOR SELECTION FOR CRYSTAL OSCILLATOR**

Osc Type	Crystal Freq.	Typical Capacitor Values Tested:	
		C1	C2
LP	32 kHz	33 pF	33 pF
	200 kHz	15 pF	15 pF
XT	1 MHz	33 pF	33 pF
	4 MHz	27 pF	27 pF
HS	4 MHz	27 pF	27 pF
	8 MHz	22 pF	22 pF
	20 MHz	15 pF	15 pF

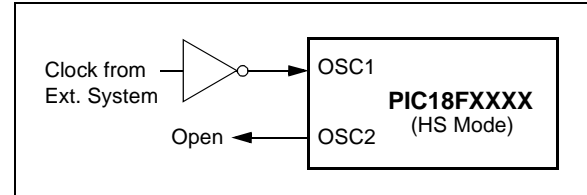
**Capacitor values are for design guidance only.**  
 These capacitors were tested with the crystals listed below for basic start-up and operation. **These values are not optimized.**  
 Different capacitor values may be required to produce acceptable oscillator operation. The user should test the performance of the oscillator over the expected VDD and temperature range for the application.  
 See the notes following this table for additional information.

Crystals Used:	
32 kHz	4 MHz
200 kHz	8 MHz
1 MHz	20 MHz

- Note 1:** Higher capacitance increases the stability of oscillator, but also increases the start-up time.
- 2:** When operating below 3V VDD, or when using certain ceramic resonators at any voltage, it may be necessary to use the HS mode or switch to a crystal oscillator.
- 3:** Since each resonator/crystal has its own characteristics, the user should consult the resonator/crystal manufacturer for appropriate values of external components.
- 4:** RS may be required to avoid overdriving crystals with low drive level specification.
- 5:** Always verify oscillator performance over the VDD and temperature range that is expected for the application.

An external clock source may also be connected to the OSC1 pin in the HS mode, as shown in Figure 2-2.

**FIGURE 2-2: EXTERNAL CLOCK INPUT OPERATION (HS OSC CONFIGURATION)**



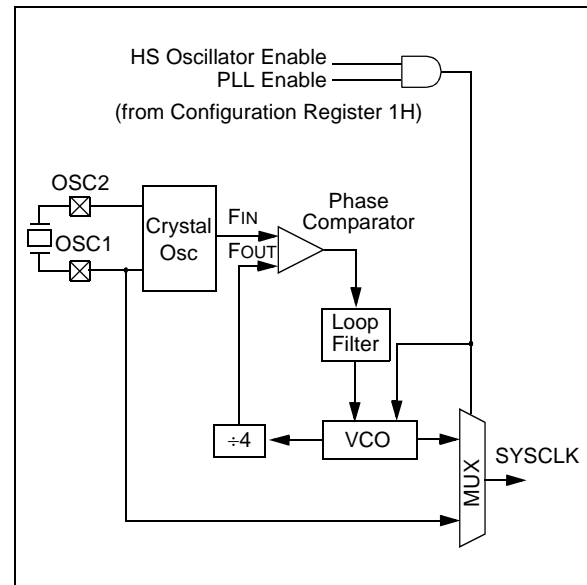
## 2.3 HSPLL

A Phase Locked Loop (PLL) circuit is provided as an option for users who wish to use a lower frequency crystal oscillator circuit, or to clock the device up to its highest rated frequency from a crystal oscillator. This may be useful for customers who are concerned with EMI due to high-frequency crystals.

The HSPLL mode makes use of the HS mode oscillator for frequencies up to 10 MHz. A PLL then multiplies the oscillator output frequency by 4 to produce an internal clock frequency up to 40 MHz.

The PLL is enabled only when the oscillator Configuration bits are programmed for HSPLL mode. If programmed for any other mode, the PLL is not enabled.

**FIGURE 2-3: PLL BLOCK DIAGRAM**



## 3.1.2 ENTERING POWER MANAGED MODES

In general, entry, exit and switching between power managed clock sources requires clock source switching. In each case, the sequence of events is the same.

Any change in the power managed mode begins with loading the OSCCON register and executing a `SLEEP` instruction. The SCS1:SCS0 bits select one of three power managed clock sources; the primary clock (as defined in Configuration Register 1H), the secondary clock (the Timer1 oscillator) and the internal oscillator block (used in RC modes). Modifying the SCS bits will have no effect until a `SLEEP` instruction is executed. Entry to the power managed mode is triggered by the execution of a `SLEEP` instruction.

Figure 3-5 shows how the system is clocked while switching from the primary clock to the Timer1 oscillator. When the `SLEEP` instruction is executed, clocks to the device are stopped at the beginning of the next instruction cycle. Eight clock cycles from the new clock source are counted to synchronize with the new clock source. After eight clock pulses from the new clock source are counted, clocks from the new clock source resume clocking the system. The actual length of the pause is between eight and nine clock periods from the new clock source. This ensures that the new clock source is stable and that its pulse width will not be less than the shortest pulse width of the two clock sources.

Three bits indicate the current clock source: OSTS and IOFS in the OSCCON register and T1RUN in the T1CON register. Only one of these bits will be set while in a power managed mode. When the OSTS bit is set, the primary clock is providing the system clock. When the IOFS bit is set, the INTOSC output is providing a stable 8 MHz clock source and is providing the system clock. When the T1RUN bit is set, the Timer1 oscillator is providing the system clock. If none of these bits are set, then either the INTRC clock source is clocking the system, or the INTOSC source is not yet stable.

If the internal oscillator block is configured as the primary clock source in Configuration Register 1H, then both the OSTS and IOFS bits may be set when in PRI\_RUN or PRI\_IDLE modes. This indicates that the primary clock (INTOSC output) is generating a stable 8 MHz output. Entering an RC power managed mode (same frequency) would clear the OSTS bit.

**Note 1:** Caution should be used when modifying a single IRCF bit. If VDD is less than 3V, it is possible to select a higher clock speed than is supported by the low VDD. Improper device operation may result if the VDD/FOSC specifications are violated.

**2:** Executing a `SLEEP` instruction does not necessarily place the device into Sleep mode; executing a `SLEEP` instruction is simply a trigger to place the controller into a power managed mode selected by the OSCCON register, one of which is Sleep mode.

## 3.1.3 MULTIPLE SLEEP COMMANDS

The power managed mode that is invoked with the `SLEEP` instruction is determined by the settings of the IDLEN and SCS bits at the time the instruction is executed. If another `SLEEP` instruction is executed, the device will enter the power managed mode specified by these same bits at that time. If the bits have changed, the device will enter the new power managed mode specified by the new bit settings.

## 3.1.4 COMPARISONS BETWEEN RUN AND IDLE MODES

Clock source selection for the Run modes is identical to the corresponding Idle modes. When a `SLEEP` instruction is executed, the SCS bits in the OSCCON register are used to switch to a different clock source. As a result, if there is a change of clock source at the time a `SLEEP` instruction is executed, a clock switch will occur.

In Idle modes, the CPU is not clocked and is not running. In Run modes, the CPU is clocked and executing code. This difference modifies the operation of the WDT when it times out. In Idle modes, a WDT time-out results in a wake from power managed modes. In Run modes, a WDT time-out results in a WDT Reset (see Table 3-2).

During a wake-up from an Idle mode, the CPU starts executing code by entering the corresponding Run mode until the primary clock becomes ready. When the primary clock becomes ready, the clock source is automatically switched to the primary clock. The IDLEN and SCS bits are unchanged during and after the wake-up.

Figure 3-2 shows how the system is clocked during the clock source switch. The example assumes the device was in SEC\_IDLE or SEC\_RUN mode when a wake is triggered (the primary clock was configured in HSPLL mode).

## REGISTER 6-1: EECN1: EEPROM CONTROL 1 REGISTER

R/W-x	R/W-x	U-0	R/W-0	R/W-x	R/W-0	R/S-0	R/S-0
EEPGD	CFGS	—	FREE	WRERR <sup>(1)</sup>	WREN	WR	RD
bit 7							bit 0

### Legend:

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
S = Bit can only be set	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	HC = Bit is cleared by hardware

- bit 7 **EEPGD:** Flash Program or Data EEPROM Memory Select bit  
1 = Access program Flash memory  
0 = Access data EEPROM memory
- bit 6 **CFGS:** Flash Program/Data EEPROM or Configuration Select bit  
1 = Accesses Configuration, User ID and Device ID Registers  
0 = Accesses Flash Program or data EEPROM Memory
- bit 5 **Unimplemented:** Read as '0'
- bit 4 **FREE:** Flash Row Erase Enable bit  
1 = Erase the program memory row addressed by TBLPTR on the next WR command  
(cleared by completion of erase operation – TBLPTR<5:0> are ignored)  
0 = Perform write only
- bit 3 **WRERR:** EEPROM Error Flag bit<sup>(1)</sup>  
1 = A write operation was prematurely terminated (any Reset during self-timed programming)  
0 = The write operation completed normally
- bit 2 **WREN:** Program/Erase Enable bit  
1 = Allows program/erase cycles  
0 = Inhibits programming/erasing of program Flash and data EEPROM
- bit 1 **WR:** Write Control bit  
1 = Initiates a data EEPROM erase/write cycle or a program memory erase cycle or write cycle.  
(The operation is self-timed and the bit is cleared by hardware once write is complete. The WR bit can only be set (not cleared) in software.)  
0 = Write cycle completed
- bit 0 **RD:** Read Control bit  
1 = Initiates a memory read  
(Read takes one cycle. RD is cleared in hardware. The RD bit can only be set (not cleared) in software. RD bit cannot be set when EEGPD = 1.)  
0 = Read completed

**Note 1:** When a WRERR occurs, the EEGPD and CFGS bits are not cleared. This allows tracing of the error condition.

# PIC18F1220/1320

## REGISTER 9-5: PIR2: PERIPHERAL INTERRUPT REQUEST REGISTER 2

R/W-0/0	U-0	U-0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	U-0
OSCFIF	—	—	EEIF	—	LVDIF	TMR3IF	—
bit 7							bit 0

### Legend:

R = Readable bit

W = Writable bit

U = Unimplemented bit, read as '0'

u = Bit is unchanged

x = Bit is unknown

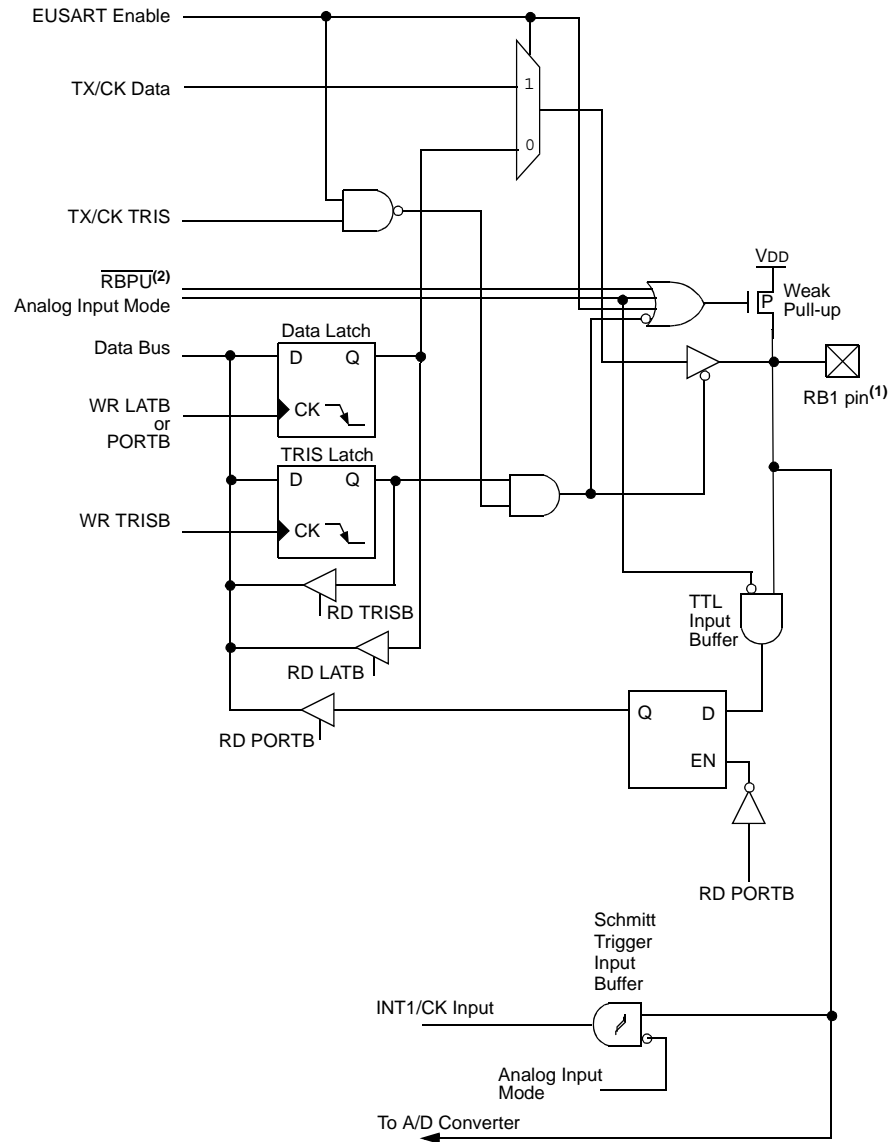
-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

'0' = Bit is cleared

- bit 7      **OSCFIF:** Oscillator Fail Interrupt Flag bit  
1 = System oscillator failed, clock input has changed to INTOSC (must be cleared in software)  
0 = System clock operating
- bit 6-5      **Unimplemented:** Read as '0'
- bit 4      **EEIF:** Data EEPROM/Flash Write Operation Interrupt Flag bit  
1 = The write operation is complete (must be cleared in software)  
0 = The write operation is not complete or has not been started
- bit 3      **Unimplemented:** Read as '0'
- bit 2      **LVDIF:** Low-Voltage Detect Interrupt Flag bit  
1 = A low-voltage condition occurred (must be cleared in software)  
0 = The device voltage is above the Low-Voltage Detect trip point
- bit 1      **TMR3IF:** TMR3 Overflow Interrupt Flag bit  
1 = TMR3 register overflowed (must be cleared in software)  
0 = TMR3 register did not overflow
- bit 0      **Unimplemented:** Read as '0'

**FIGURE 10-8: BLOCK DIAGRAM OF RB1/AN5/TX/CK/INT1 PIN**



**Note 1:** I/O pins have diode protection to VDD and VSS.

**2:** To enable weak pull-ups, set the appropriate TRIS bit(s) and clear the  $\overline{\text{RBP}}$  bit (INTCON2<7>).

## 11.1 Timer0 Operation

Timer0 can operate as a timer or as a counter.

Timer mode is selected by clearing the T0CS bit. In Timer mode, the Timer0 module will increment every instruction cycle (without prescaler). If the TMR0 register is written, the increment is inhibited for the following two instruction cycles. The user can work around this by writing an adjusted value to the TMR0 register.

Counter mode is selected by setting the T0CS bit. In Counter mode, Timer0 will increment either on every rising or falling edge of pin RA4/T0CKI. The incrementing edge is determined by the Timer0 Source Edge Select bit (T0SE). Clearing the T0SE bit selects the rising edge.

When an external clock input is used for Timer0, it must meet certain requirements. The requirements ensure the external clock can be synchronized with the internal phase clock (Tosc). Also, there is a delay in the actual incrementing of Timer0 after synchronization.

## 11.2 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The prescaler is not readable or writable.

The PSA and T0PS2:T0PS0 bits determine the prescaler assignment and prescale ratio.

Clearing bit PSA will assign the prescaler to the Timer0 module. When the prescaler is assigned to the Timer0 module, prescale values of 1:2, 1:4, ..., 1:256 are selectable.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (e.g., CLRF TMR0, MOVWF TMR0, BSF TMR0, x, ..., etc.) will clear the prescaler count.

**Note:** Writing to TMR0 when the prescaler is assigned to Timer0 will clear the prescaler count, but will not change the prescaler assignment.

### 11.2.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control (i.e., it can be changed “on-the-fly” during program execution).

## 11.3 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h in 8-bit mode, or FFFFh to 0000h in 16-bit mode. This overflow sets the TMR0IF bit. The interrupt can be masked by clearing the TMR0IE bit. The TMR0IF bit must be cleared in software by the Timer0 module Interrupt Service Routine before re-enabling this interrupt. The TMR0 interrupt cannot awaken the processor from Low-Power Sleep mode, since the timer requires clock cycles even when T0CS is set.

## 11.4 16-Bit Mode Timer Reads and Writes

TMR0H is not the high byte of the timer/counter in 16-bit mode, but is actually a buffered version of the high byte of Timer0 (refer to Figure 11-2). The high byte of the Timer0 counter/timer is not directly readable nor writable. TMR0H is updated with the contents of the high byte of Timer0 during a read of TMR0L. This provides the ability to read all 16 bits of Timer0, without having to verify that the read of the high and low byte were valid due to a rollover between successive reads of the high and low byte.

A write to the high byte of Timer0 must also take place through the TMR0H Buffer register. Timer0 high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16 bits of Timer0 to be updated at once.

**TABLE 11-1: REGISTERS ASSOCIATED WITH TIMER0**

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
TMR0L	Timer0 Module Low Byte Register								xxxx xxxx	uuuu uuuu
TMR0H	Timer0 Module High Byte Register								0000 0000	0000 0000
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
T0CON	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0	1111 1111	1111 1111
TRISA	RA7 <sup>(1)</sup>	RA6 <sup>(1)</sup>	—	PORTA Data Direction Register					11-1 1111	11-1 1111

**Legend:** x = unknown, u = unchanged, — = unimplemented locations read as ‘0’. Shaded cells are not used by Timer0.

**Note 1:** RA6 and RA7 are enabled as I/O pins, depending on the oscillator mode selected in Configuration Word 1H.



## 15.5 Enhanced PWM Mode

The Enhanced PWM Mode provides additional PWM output options for a broader range of control applications. The module is an upwardly compatible version of the standard CCP module and offers up to four outputs, designated P1A through P1D. Users are also able to select the polarity of the signal (either active-high or active-low). The module's output mode and polarity are configured by setting the P1M1:P1M0 and CCP1M3:CCP1M0 bits of the CCP1CON register (CCP1CON<7:6> and CCP1CON<3:0>, respectively).

Figure 15-3 shows a simplified block diagram of PWM operation. All control registers are double-buffered and are loaded at the beginning of a new PWM cycle (the period boundary when Timer2 resets) in order to prevent glitches on any of the outputs. The exception is the PWM Delay register, ECCP1DEL, which is loaded at either the duty cycle boundary or the boundary period (whichever comes first). Because of the buffering, the module waits until the assigned timer resets instead of starting immediately. This means that Enhanced PWM waveforms do not exactly match the standard PWM waveforms, but are instead offset by one full instruction cycle (4 TOSC).

As before, the user must manually configure the appropriate TRIS bits for output.

### 15.5.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the equation:

#### EQUATION 15-1: PWM PERIOD

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1/[PWM period]. When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP1 pin is set (if PWM duty cycle = 0%, the CCP1 pin will not be set)
- The PWM duty cycle is copied from CCPR1L into CCPR1H

**Note:** The Timer2 postscaler (see **Section 13.0 "Timer2 Module"**) is not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

### 15.5.2 PWM DUTY CYCLE

The PWM duty cycle is specified by writing to the CCPR1L register and to the CCP1CON<5:4> bits. Up to 10-bit resolution is available. The CCPR1L contains the eight MSBs and the CCP1CON<5:4> contains the two LSBs. This 10-bit value is represented by CCPR1L:CCP1CON<5:4>. The PWM duty cycle is calculated by the equation:

#### EQUATION 15-2: PWM DUTY CYCLE

$$\text{PWM Duty Cycle} = (\text{CCPR1L:CCP1CON<5:4>}) \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

CCPR1L and CCP1CON<5:4> can be written to at any time, but the duty cycle value is not copied into CCPR1H until a match between PR2 and TMR2 occurs (i.e., the period is complete). In PWM mode, CCPR1H is a read-only register.

The CCPR1H register and a 2-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitchless PWM operation. When the CCPR1H and 2-bit latch match TMR2, concatenated with an internal 2-bit Q clock or two bits of the TMR2 prescaler, the CCP1 pin is cleared. The maximum PWM resolution (bits) for a given PWM frequency is given by the equation:

#### EQUATION 15-3: PWM RESOLUTION

$$\text{PWM Resolution (max)} = \frac{\log\left(\frac{F_{OSC}}{F_{PWM}}\right)}{\log(2)} \text{ bits}$$

**Note:** If the PWM duty cycle value is longer than the PWM period, the CCP1 pin will not be cleared.

### 15.5.3 PWM OUTPUT CONFIGURATIONS

The P1M1:P1M0 bits in the CCP1CON register allow one of four configurations:

- Single Output
- Half-Bridge Output
- Full-Bridge Output, Forward mode
- Full-Bridge Output, Reverse mode

The Single Output mode is the Standard PWM mode discussed in **Section 15.5 "Enhanced PWM Mode"**. The Half-Bridge and Full-Bridge Output modes are covered in detail in the sections that follow.

The general relationship of the outputs in all configurations is summarized in Figure 15-4.

**TABLE 15-4: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz**

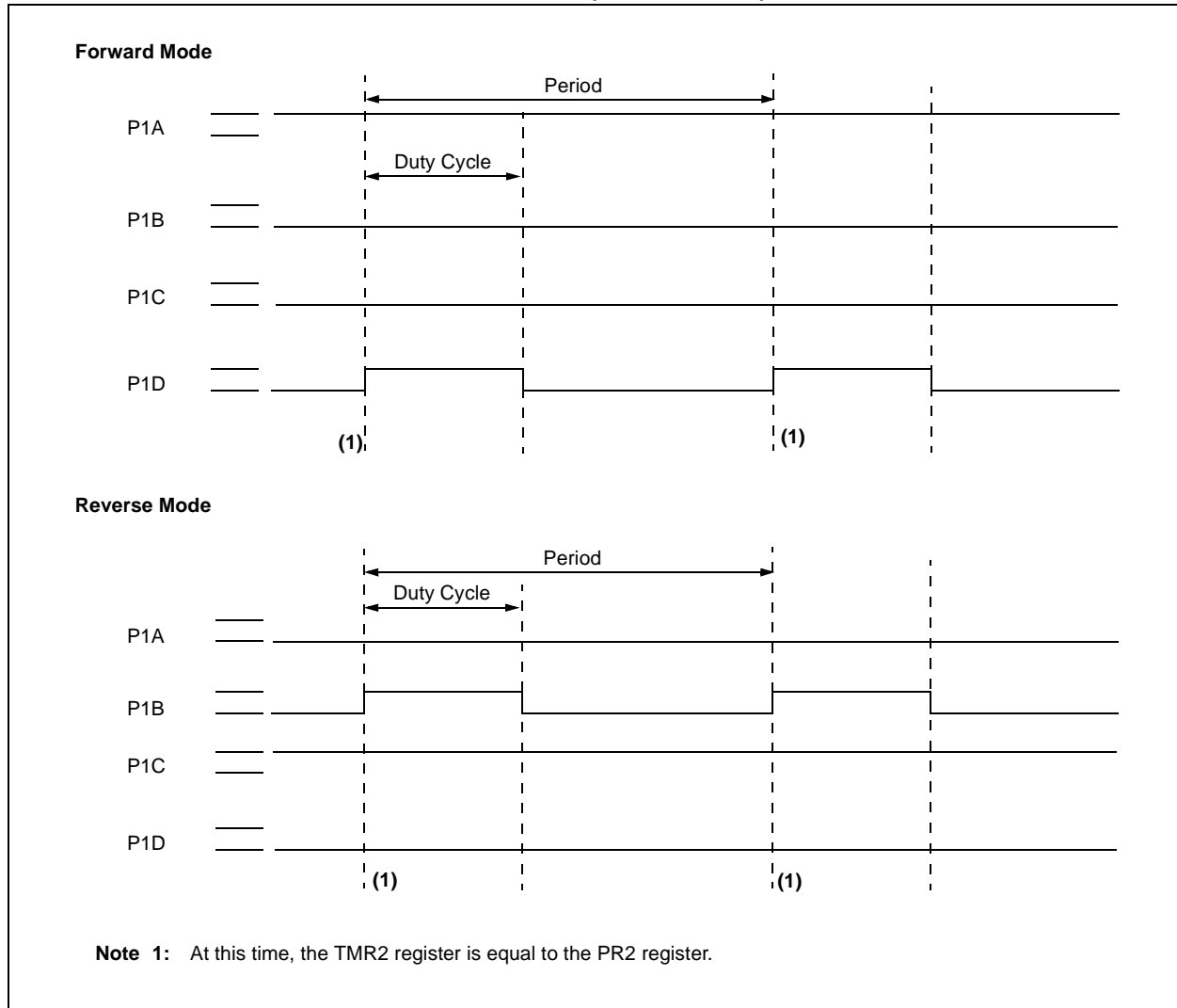
PWM Frequency	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	FFh	FFh	FFh	3Fh	1Fh	17h
Maximum Resolution (bits)	10	10	10	8	7	6.58

## 15.5.5 FULL-BRIDGE MODE

In Full-Bridge Output mode, four pins are used as outputs; however, only two outputs are active at a time. In the Forward mode, pin RB3/CCP1/P1A is continuously active and pin RB7/PGD/T1OSI/P1D/KBI3 is modulated. In the Reverse mode, pin RB6/PGC/T1OSO/T13CKI/P1C/KBI2 is continuously active and pin RB2/P1B/INT2 is modulated. These are illustrated in Figure 15-8.

The TRISB<3:2> and TRISB<7:6> bits must be cleared to make the P1A, P1B, P1C and P1D pins output.

**FIGURE 15-8: FULL-BRIDGE PWM OUTPUT (ACTIVE-HIGH)**



## 15.5.6 PROGRAMMABLE DEAD-BAND DELAY

In half-bridge applications where all power switches are modulated at the PWM frequency at all times, the power switches normally require more time to turn off than to turn on. If both the upper and lower power switches are switched at the same time (one turned on and the other turned off), both switches may be on for a short period of time until one switch completely turns off. During this brief interval, a very high current (shoot-through current) may flow through both power switches, shorting the bridge supply. To avoid this potentially destructive shoot-through current from flowing during switching, turning on either of the power switches is normally delayed to allow the other switch to completely turn off.

In the Half-Bridge Output mode, a digitally programmable dead-band delay is available to avoid shoot-through current from destroying the bridge power switches. The delay occurs at the signal transition from the non-active state to the active state. See Figure 15-6 for an illustration. The lower seven bits of the PWM1CON register (Register 15-2) sets the delay period in terms of microcontroller instruction cycles (T<sub>CY</sub> or 4 T<sub>OSC</sub>).

## 15.5.7 ENHANCED PWM AUTO-SHUTDOWN

When the ECCP is programmed for any of the Enhanced PWM modes, the active output pins may be configured for auto-shutdown. Auto-shutdown immediately places the Enhanced PWM output pins into a defined shutdown state when a shutdown event occurs.

A shutdown event can be caused by the INT0, INT1 or INT2 pins (or any combination of these three sources). The auto-shutdown feature can be disabled by not selecting any auto-shutdown sources. The auto-shutdown sources to be used are selected using the ECCPAS2:ECCPAS0 bits (bits <6:4> of the ECCPAS register).

When a shutdown occurs, the output pins are asynchronously placed in their shutdown states, specified by the PSSAC1:PSSAC0 and PSSBD1:PSSBD0 bits (ECCPAS<3:0>). Each pin pair (P1A/P1C and P1B/P1D) may be set to drive high, drive low or be tri-stated (not driving). The ECCPASE bit (ECCPAS<7>) is also set to hold the Enhanced PWM outputs in their shutdown states.

The ECCPASE bit is set by hardware when a shutdown event occurs. If automatic restarts are not enabled, the ECCPASE bit is cleared by firmware when the cause of the shutdown clears. If automatic restarts are enabled, the ECCPASE bit is automatically cleared when the cause of the auto-shutdown has cleared.

If the ECCPASE bit is set when a PWM period begins, the PWM outputs remain in their shutdown state for that entire PWM period. When the ECCPASE bit is cleared, the PWM outputs will return to normal operation at the beginning of the next PWM period.

<b>Note:</b>	Writing to the ECCPASE bit is disabled while a shutdown condition is active.
--------------	--

## 16.4 EUSART Synchronous Master Mode

The Synchronous Master mode is entered by setting the CSRC bit (TXSTA<7>). In this mode, the data is transmitted in a half-duplex manner (i.e., transmission and reception do not occur at the same time). When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit, SYNC (TXSTA<4>). In addition, enable bit, SPEN (RCSTA<7>), is set in order to configure the RB1/AN5/TX/CK/INT1 and RB4/AN6/RX/DT/KBI0 I/O pins to CK (clock) and DT (data) lines, respectively.

The Master mode indicates that the processor transmits the master clock on the CK line. Clock polarity is selected with the SCKP bit (BAUDCTL<5>); setting SCKP sets the Idle state on CK as high, while clearing the bit sets the Idle state as low. This option is provided to support Microwire devices with this module.

### 16.4.1 EUSART SYNCHRONOUS MASTER TRANSMISSION

The EUSART transmitter block diagram is shown in Figure 16-2. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The shift register obtains its data from the Read/Write Transmit Buffer register, TXREG. The TXREG register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREG (if available).

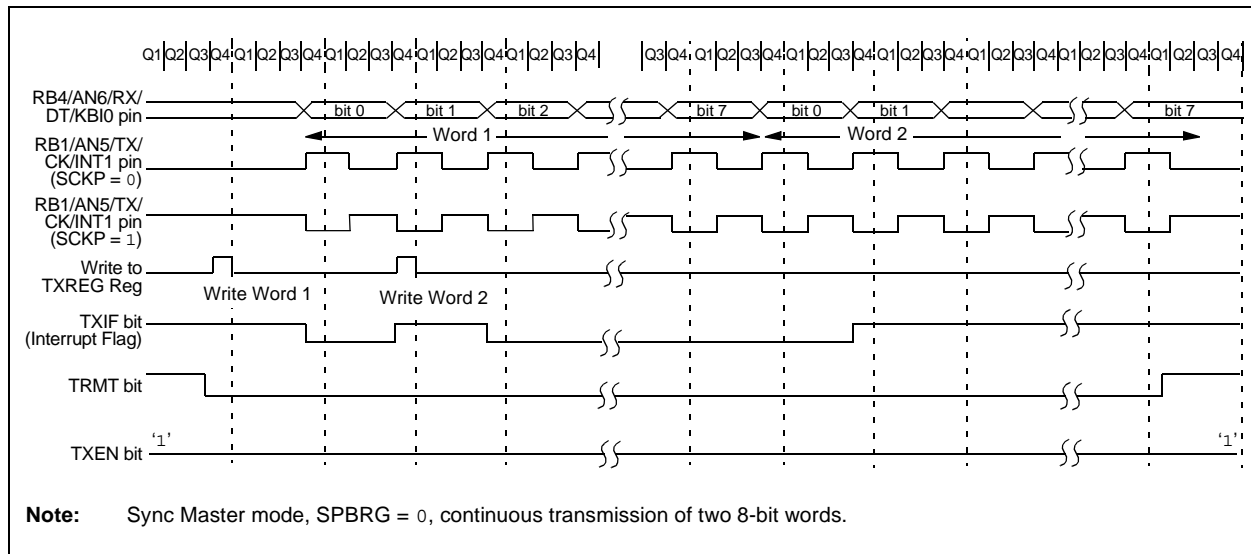
Once the TXREG register transfers the data to the TSR register (occurs in one T<sub>CYCLE</sub>), the TXREG is empty and interrupt bit, TXIF (PIR1<4>), is set. The interrupt can be enabled/disabled by setting/clearing enable bit, TXIE (PIE1<4>). Flag bit, TXIF, will be set, regardless of the state of enable bit, TXIE and cannot be cleared in software. It will reset only when new data is loaded into the TXREG register.

While flag bit, TXIF, indicates the status of the TXREG register, another bit, TRMT (TXSTA<1>), shows the status of the TSR register. TRMT is a read-only bit, which is set when the TSR is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory, so it is not available to the user.

To set up a Synchronous Master Transmission:

1. Initialize the SPBRGH:SPBRG registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the synchronous master serial port by setting bits SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set bit TX9.
5. Enable the transmission by setting bit TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Start transmission by loading data to the TXREG register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

**FIGURE 16-10: SYNCHRONOUS TRANSMISSION**



## 19.0 SPECIAL FEATURES OF THE CPU

PIC18F1220/1320 devices include several features intended to maximize system reliability, minimize cost through elimination of external components and offer code protection. These are:

- Oscillator Selection
- Resets:
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
  - Brown-out Reset (BOR)
- Interrupts
- Watchdog Timer (WDT)
- Fail-Safe Clock Monitor
- Two-Speed Start-up
- Code Protection
- ID Locations
- In-Circuit Serial Programming

Several oscillator options are available to allow the part to fit the application. The RC oscillator option saves system cost, while the LP crystal option saves power. These are discussed in detail in **Section 2.0 “Oscillator Configurations”**.

A complete discussion of device Resets and interrupts is available in previous sections of this data sheet.

In addition to their Power-up and Oscillator Start-up Timers provided for Resets, PIC18F1220/1320 devices have a Watchdog Timer, which is either permanently enabled via the Configuration bits, or software controlled (if configured as disabled).

The inclusion of an internal RC oscillator also provides the additional benefits of a Fail-Safe Clock Monitor (FSCM) and Two-Speed Start-up. FSCM provides for background monitoring of the peripheral clock and automatic switchover in the event of its failure. Two-Speed Start-up enables code to be executed almost immediately on start-up, while the primary clock source completes its start-up delays.

All of these features are enabled and configured by setting the appropriate Configuration register bits.

### 19.1 Configuration Bits

The Configuration bits can be programmed (read as '0'), or left unprogrammed (read as '1'), to select various device configurations. These bits are mapped starting at program memory location 300000h.

The user will note that address 300000h is beyond the user program memory space. In fact, it belongs to the configuration memory space (300000h-3FFFFh), which can only be accessed using table reads and table writes.

Programming the Configuration registers is done in a manner similar to programming the Flash memory. The EECON1 register WR bit starts a self-timed write to the Configuration register. In normal operation mode, a TBLWT instruction, with the TBLPTR pointing to the Configuration register, sets up the address and the data for the Configuration register write. Setting the WR bit starts a long write to the Configuration register. The Configuration registers are written a byte at a time. To write or erase a configuration cell, a TBLWT instruction can write a '1' or a '0' into the cell. For additional details on Flash programming, refer to **Section 6.5 “Writing to Flash Program Memory”**.

**TABLE 19-1: CONFIGURATION BITS AND DEVICE IDS**

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprogrammed Value
300001h CONFIG1H	IESO	FSCM	—	—	FOSC3	FOSC2	FOSC1	FOSC0	11-- 1111
300002h CONFIG2L	—	—	—	—	BORV1	BORV0	BOR	PWRTEN	---- 1111
300003h CONFIG2H	—	—	—	WDTPS3	WDTPS2	WDTPS1	WDTPS0	WDT	---1 1111
300005h CONFIG3H	MCLRE	—	—	—	—	—	—	—	1--- ----
300006h CONFIG4L	DEBUG	—	—	—	—	LVP	—	STVR	1--- -1-1
300008h CONFIG5L	—	—	—	—	—	—	CP1	CP0	---- --11
300009h CONFIG5H	CPD	CPB	—	—	—	—	—	—	11-- ----
30000Ah CONFIG6L	—	—	—	—	—	—	WRT1	WRT0	---- --11
30000Bh CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—	111- ----
30000Ch CONFIG7L	—	—	—	—	—	—	EBTR1	EBTR0	---- --11
30000Dh CONFIG7H	—	EBTRB	—	—	—	—	—	—	-1-- ----
3FFFFEh DEVID1 <sup>(1)</sup>	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	xxxx xxxx <sup>(1)</sup>
3FFFFFh DEVID2 <sup>(1)</sup>	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	0000 0111

**Legend:** x = unknown, u = unchanged, — = unimplemented. Shaded cells are unimplemented, read as '0'.

**Note 1:** See Register 19-12 for DEVID1 values. DEVID registers are read-only and cannot be programmed by the user.

# PIC18F1220/1320

**REGISTER 19-6: CONFIG5L: CONFIGURATION REGISTER 5 LOW (BYTE ADDRESS 300008h)**

U-0	U-0	U-0	U-0	U-0	U-0	R/C-1	R/C-1
—	—	—	—	—	—	CP1	CP0
bit 7						bit 0	

**Legend:**

R = Readable bit                      W = Writable bit                      U = Unimplemented bit, read as '0'  
u = Bit is unchanged                      x = Bit is unknown                      -n/n = Value at POR and BOR/Value at all other Resets  
'1' = Bit is set                      '0' = Bit is cleared                      C = Clearable bit

- bit 7-2                      **Unimplemented:** Read as '0'
- bit 1                      **CP1:** Code Protection bit (PIC18F1320)  
1 = Block 1 (001000-001FFFh) not code-protected  
0 = Block 1 (001000-001FFFh) code-protected
- bit 0                      **CP0:** Code Protection bit (PIC18F1320)  
1 = Block 0 (00200-000FFFh) not code-protected  
0 = Block 0 (00200-000FFFh) code-protected
- bit 1                      **CP1:** Code Protection bit (PIC18F1220)  
1 = Block 1 (000800-000FFFh) not code-protected  
0 = Block 1 (000800-000FFFh) code-protected
- bit 0                      **CP0:** Code Protection bit (PIC18F1220)  
1 = Block 0 (000200-0007FFh) not code-protected  
0 = Block 0 (000200-0007FFh) code-protected

**REGISTER 19-7: 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:**

R = Readable bit                      W = Writable bit                      U = Unimplemented bit, read as '0'  
u = Bit is unchanged                      x = Bit is unknown                      -n/n = Value at POR and BOR/Value at all other Resets  
'1' = Bit is set                      '0' = Bit is cleared                      C = Clearable bit

- bit 7                      **CPD:** Data EEPROM Code Protection bit  
1 = Data EEPROM not code-protected  
0 = Data EEPROM code-protected
- bit 6                      **CPB:** Boot Block Code Protection bit  
1 = Boot Block (000000-0001FFFh) not code-protected  
0 = Boot Block (000000-0001FFFh) code-protected
- bit 5-0                      **Unimplemented:** Read as '0'

## 20.0 INSTRUCTION SET SUMMARY

The PIC18 instruction set adds many enhancements to the previous PIC instruction sets, while maintaining an easy migration from these PIC instruction sets.

Most instructions are a single program memory word (16 bits), but there are three instructions that require two program memory locations.

Each single-word instruction is a 16-bit word divided into an opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into four basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal** operations
- **Control** operations

The PIC18 instruction set summary in Table 20-1 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 20-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The file register (specified by 'f')
2. The destination of the result (specified by 'd')
3. The accessed memory (specified by 'a')

The file register designator 'f' specifies which file register is to be used by the instruction.

The destination designator 'd' specifies where the result of the operation is to be placed. If 'd' is zero, the result is placed in the WREG register. If 'd' is one, the result is placed in the file register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The file register (specified by 'f')
2. The bit in the file register (specified by 'b')
3. The accessed memory (specified by 'a')

The bit field designator 'b' selects the number of the bit affected by the operation, while the file register designator 'f' represents the number of the file in which the bit is located.

The **literal** instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by 'k')
- The desired FSR register to load the literal value into (specified by 'f')
- No operand required (specified by '—')

The **control** instructions may use some of the following operands:

- A program memory address (specified by 'n')
- The mode of the **CALL** or **RETURN** instructions (specified by 's')
- The mode of the table read and table write instructions (specified by 'm')
- No operand required (specified by '—')

All instructions are a single word, except for three double-word instructions. These three instructions were made double-word instructions so that all the required information is available in these 32 bits. In the second word, the 4 MSBs are '1's. If this second word is executed as an instruction (by itself), it will execute as a **NOP**.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true, or the program counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles, with the additional instruction cycle(s) executed as a **NOP**.

The double-word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1  $\mu$ s. If a conditional test is true, or the program counter is changed as a result of an instruction, the instruction execution time is 2  $\mu$ s. Two-word branch instructions (if true) would take 3  $\mu$ s.

Figure 20-1 shows the general formats that the instructions can have.

All examples use the format 'nnh' to represent a hexadecimal number, where 'h' signifies a hexadecimal digit.

The Instruction Set Summary, shown in Table 20-1, lists the instructions recognized by the Microchip Assembler (MPASM™). **Section 20.2 "Instruction Set"** provides a description of each instruction.

### 20.1 Read-Modify-Write Operations

Any instruction that specifies a file register as part of the instruction performs a Read-Modify-Write (R-M-W) operation. The register is read, the data is modified and the result is stored according to either the instruction or the destination designator 'd'. A read operation is performed on a register even if the instruction writes to that register.

For example, a "**BCF** PORTB,1" instruction will read PORTB, clear bit 1 of the data, then write the result back to PORTB. The read operation would have the unintended result that any condition that sets the RBIF flag would be cleared. The R-M-W operation may also copy the level of an input pin to its corresponding output latch.

## BZ Branch if Zero

Syntax: [ *label* ] BZ n

Operands:  $-128 \leq n \leq 127$

Operation: if Zero bit is '1'  
(PC) + 2 + 2n → PC

Status Affected: None

Encoding: 

1110	0000	nnnn	nnnn
------	------	------	------

Description: If the Zero bit is '1', then the program will branch. The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

**Example:**                      HERE                      BZ                      Jump

Before Instruction

PC = address (HERE)

After Instruction

If Zero = 1;

PC = address (Jump)

If Zero = 0;

PC = address (HERE + 2)

## CALL Subroutine Call

Syntax: [ *label* ] CALL k [,s]

Operands:  $0 \leq k \leq 1048575$   
s ∈ [0,1]

Operation: (PC) + 4 → TOS,  
k → PC<20:1>,  
if s = 1  
(W) → WS,  
(Status) → STATUSS,  
(BSR) → BSRS

Status Affected: None

Encoding: 

1110	110s	k <sub>7</sub> kkk	kkkk <sub>0</sub>
1111	k <sub>19</sub> kkk	kkkk	kkkk <sub>8</sub>

Description: Subroutine call of entire 2-Mbyte memory range. First, return address (PC + 4) is pushed onto the return stack. If 's' = 1, the W, Status and BSR registers are also pushed into their respective shadow registers, WS, STATUSS and BSRS. If 's' = 0, no update occurs (default). Then, the 20-bit value 'k' is loaded into PC<20:1>. CALL is a 2-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	Push PC to stack	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

**Example:**                      HERE                      CALL                      THERE, FAST

Before Instruction

PC = address (HERE)

After Instruction

PC = address (THERE)  
TOS = address (HERE + 4)  
WS = W  
BSRS = BSR  
STATUSS = Status



GOTO		Unconditional Branch			
Syntax:	[ <i>label</i> ] GOTO k				
Operands:	$0 \leq k \leq 1048575$				
Operation:	$k \rightarrow PC<20:1>$				
Status Affected:	None				
Encoding:					
1st word (k<7:0>)	1110	1111	k <sub>7</sub> kkk	kkkk <sub>0</sub>	
2nd word(k<19:8>)	1111	k <sub>19</sub> kkk	kkkk	kkkk <sub>8</sub>	
Description:	GOTO allows an unconditional branch anywhere within the entire 2-Mbyte memory range. The 20-bit value 'k' is loaded into PC<20:1>. GOTO is always a 2-cycle instruction.				
Words:	2				
Cycles:	2				
Q Cycle Activity:					
	Q1	Q2	Q3	Q4	
	Decode	Read literal 'k'<7:0>,	No operation	Read literal 'k'<19:8>, Write to PC	
	No operation	No operation	No operation	No operation	

**Example:** GOTO THERE

After Instruction  
PC = Address (THERE)

INCF		Increment f						
Syntax:	[ <i>label</i> ] INCF f [,d [,a]]							
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]							
Operation:	(f) + 1 → dest							
Status Affected:	C, DC, N, OV, Z							
Encoding:	<table border="1"><tr><td>0010</td><td>10da</td><td>ffff</td><td>ffff</td></tr></table>				0010	10da	ffff	ffff
0010	10da	ffff	ffff					
Description:	The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).							
Words:	1							
Cycles:	1							
Q Cycle Activity:								
	Q1	Q2	Q3	Q4				
	Decode	Read register 'f'	Process Data	Write to destination				

**Example:** INCF CNT

Before Instruction

CNT	=	0xFF
Z	=	0
C	=	?
DC	=	?

After Instruction

CNT	=	0x00
Z	=	1
C	=	1
DC	=	1

RETFIE		Return from Interrupt							
Syntax:	[ <i>label</i> ] RETFIE [s]								
Operands:	s ∈ [0,1]								
Operation:	(TOS) → PC, 1 → GIE/GIEH or PEIE/GIEL, if s = 1 (WS) → W, (STATUS) → Status, (BSRS) → BSR, PCLATU, PCLATH are unchanged.								
Status Affected:	GIE/GIEH, PEIE/GIEL.								
Encoding:	<table border="1"><tr><td>0000</td><td>0000</td><td>0001</td><td>000s</td></tr></table>					0000	0000	0001	000s
0000	0000	0001	000s						
Description:	Return from interrupt. Stack is popped and Top-of-Stack (TOS) is loaded into the PC. Interrupts are enabled by setting either the high or low priority Global Interrupt Enable bit. If 's' = 1, the contents of the shadow registers, WS, STATUS and BSRS, are loaded into their corresponding registers, W, Status and BSR. If 's' = 0, no update of these registers occurs (default).								
Words:	1								
Cycles:	2								
Q Cycle Activity:									

Q1	Q2	Q3	Q4
Decode	No operation	No operation	Pop PC from stack Set GIEH or GIEL
No operation	No operation	No operation	No operation

**Example:** RETFIE 1

After Interrupt

PC	=	TOS
W	=	WS
BSR	=	BSRS
Status	=	STATUS
GIE/GIEH, PEIE/GIEL	=	1

RETLW	Return Literal to W				
Syntax:	[ <i>label</i> ] RETLW k				
Operands:	$0 \leq k \leq 255$				
Operation:	$k \rightarrow W$ , (TOS) $\rightarrow$ PC, PCLATU, PCLATH are unchanged				
Status Affected:	None				
Encoding:	<table><tr><td>0000</td><td>1100</td><td>kkkk</td><td>kkkk</td></tr></table>	0000	1100	kkkk	kkkk
0000	1100	kkkk	kkkk		
Description:	W is loaded with the 8-bit literal 'k'. The program counter is loaded from the top of the stack (the return address). The high address latch (PCLATH) remains unchanged.				
Words:	1				
Cycles:	2				

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Pop PC from stack, Write to W
No operation	No operation	No operation	No operation

**Example:**

```
CALL TABLE ; W contains table
              ; offset value
              ; W now has
              ; table value
:
TABLE
  ADDWF PCL ; W = offset
  RETLW k0 ; Begin table
  RETLW k1 ;
:
  RETLW kn ; End of table
```

Before Instruction

W = 0x07

After Instruction

W = value of kn

# PIC18F1220/1320

## RETURN Return from Subroutine

**Syntax:** [ *label* ] RETURN [ *s* ]

**Operands:**  $s \in [0,1]$

**Operation:** (TOS) → PC,  
if  $s = 1$   
(WS) → W,  
(STATUS) → Status,  
(BSRS) → BSR,  
PCLATU, PCLATH are unchanged

**Status Affected:** None

**Encoding:**

0000	0000	0001	001s
------	------	------	------

**Description:** Return from subroutine. The stack is popped and the top of the stack is loaded into the program counter. If 's' = 1, the contents of the shadow registers, WS, STATUS and BSR, are loaded into their corresponding registers, W, Status and BSR. If 's' = 0, no update of these registers occurs (default).

**Words:** 1

**Cycles:** 2

**Q Cycle Activity:**

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	Pop PC from stack
No operation	No operation	No operation	No operation

**Example:** RETURN

After Interrupt  
PC = TOS

## RLCF Rotate Left f through Carry

**Syntax:** [ *label* ] RLCF f [,d [,a]]

**Operands:**  $0 \leq f \leq 255$   
 $d \in [0,1]$   
 $a \in [0,1]$

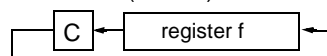
**Operation:** (f<n>) → dest<n + 1>,  
(f<7>) → C,  
(C) → dest<0>

**Status Affected:** C, N, Z

**Encoding:**

0011	01da	ffff	ffff
------	------	------	------

**Description:** The contents of register 'f' are rotated one bit to the left through the Carry flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f' (default). If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' = 1, then the bank will be selected as per the BSR value (default).



**Words:** 1

**Cycles:** 1

**Q Cycle Activity:**

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

**Example:** RLCF REG, W

Before Instruction

REG = 1110 0110  
C = 0

After Instruction

REG = 1110 0110  
W = 1100 1100  
C = 1

# PIC18F1220/1320

## 22.4 AC (Timing) Characteristics

### 22.4.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

1. TppS2ppS
2. TppS
3. TCC:ST (I<sup>2</sup>C specifications only)
4. Ts (I<sup>2</sup>C specifications only)

T		T	
F	Frequency	T	Time

Lowercase letters (pp) and their meanings:

pp		osc	OSC1
cc	CCP1	rd	$\overline{RD}$
ck	CLKO	rw	$\overline{RD}$ or $\overline{WR}$
cs	$\overline{CS}$	sc	SCK
di	SDI	ss	$\overline{SS}$
do	SDO	t0	T0CKI
dt	Data in	t1	T13CKI
io	I/O port	wr	$\overline{WR}$
mc	$\overline{MCLR}$		

Uppercase letters and their meanings:

S		P	Period
F	Fall	R	Rise
H	High	V	Valid
I	Invalid (High-Impedance)	Z	High-Impedance
L	Low		
I <sup>2</sup> C only		High	High
AA	output access	Low	Low
BUF	Bus free		

TCC:ST (I<sup>2</sup>C specifications only)

CC		SU	Setup
HD	Hold		
ST		STO	Stop condition
DAT	DATA input hold		
STA	Start condition		

# PIC18F1220/1320

**TABLE 22-16: DC CHARACTERISTICS FOR I<sub>DD</sub> SPECIFICATIONS FOR PIC18F1220/1320-H (High Temp.)**

Symbol	Device Characteristics	Min.	Typ.	Max.	Units	Condition	
						V <sub>DD</sub>	Note
I <sub>PD</sub>	Power Down Current	—	—	220	μA	5.0	Sleep mode
I <sub>DD</sub>	Supply Current	—	—	250	μA	5.0	Fosc = 31 kHz (RC_Run mode, Internal Oscillator source)
I <sub>DD</sub>	Supply Current	—	—	800	μA	5.0	Fosc = 1 MHz (RC_Run mode, Internal Oscillator source)
I <sub>DD</sub>	Supply Current	—	—	2.0	mA	5.0	Fosc = 4 MHz (RC_Run mode, Internal Oscillator source)
I <sub>DD</sub>	Supply Current	—	—	240	μA	5.0	Fosc = 31 kHz (RC_Idle mode, Internal Oscillator source)
I <sub>DD</sub>	Supply Current	—	—	600	μA	5.0	Fosc = 1 MHz (RC_Idle mode, Internal Oscillator source)
I <sub>DD</sub>	Supply Current	—	—	1.0	mA	5.0	Fosc = 4 MHz (RC_Idle mode, Internal Oscillator source)
I <sub>DD</sub>	Supply Current	—	—	1.2	mA	5.0	Fosc = 1 MHz (PRI_Run mode, EC Oscillator)
I <sub>DD</sub>	Supply Current	—	—	2.2	mA	5.0	Fosc = 4 MHz (PRI_Run mode, EC Oscillator)
I <sub>DD</sub>	Supply Current	—	—	10.0	mA	5.0	Fosc = 20 MHz (PRI_Run mode, EC Oscillator)
I <sub>DD</sub>	Supply Current	—	—	500	μA	5.0	Fosc = 1 MHz (PRI_Idle mode, EC Oscillator)
I <sub>DD</sub>	Supply Current	—	—	1.0	mA	5.0	Fosc = 4 MHz (PRI_Idle mode, EC Oscillator)
I <sub>DD</sub>	Supply Current	—	—	3.5	mA	5.0	Fosc = 20 MHz (PRI_Idle mode, EC Oscillator)

**TABLE 22-17: ADC CHARACTERISTICS FOR PIC18F1220/1320-H (High Temp.)**

Param No.	Sym.	Characteristic	Min.	Typ.	Max.	Units	Conditions
D026	A/D Conv		—	—	30	μA	V <sub>DD</sub> = 5.0V, A/D on, Not converting