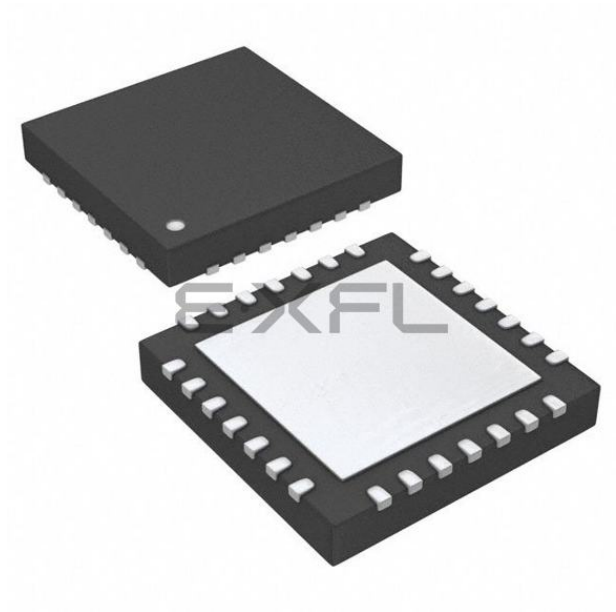


Welcome to E-XFL.COM

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	40MHz
Connectivity	UART/USART
Peripherals	Brown-out Detect/Reset, LVD, POR, PWM, WDT
Number of I/O	16
Program Memory Size	8KB (4K 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 ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VQFN Exposed Pad
Supplier Device Package	28-QFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f1320-i-ml

4.0 RESET

The PIC18F1220/1320 devices differentiate between various kinds of Reset:

a) Power-on Reset (POR)
b) $\overline{\text{MCLR}}$ Reset during normal operation
c) $\overline{\text{MCLR}}$ Reset during Sleep
d) Watchdog Timer (WDT) Reset (during execution)
e) Programmable Brown-out Reset (BOR)
f) RESET Instruction
g) Stack Full Reset
h) Stack Underflow Reset

Most registers are unaffected by a Reset. Their status is unknown on POR and unchanged by all other Resets. The other registers are forced to a "Reset state", depending on the type of Reset that occurred.

Most registers are not affected by a WDT wake-up, since this is viewed as the resumption of normal operation. Status bits from the RCON register (Register 5-2), $\overline{\text{RI}}$, $\overline{\text{TO}}$, $\overline{\text{PD}}$, $\overline{\text{POR}}$ and $\overline{\text{BOR}}$, are set or cleared differently in different Reset situations, as indicated in Table 4-2. These bits are used in software to determine the nature of the Reset. See Table 4-3 for a full description of the Reset states of all registers.

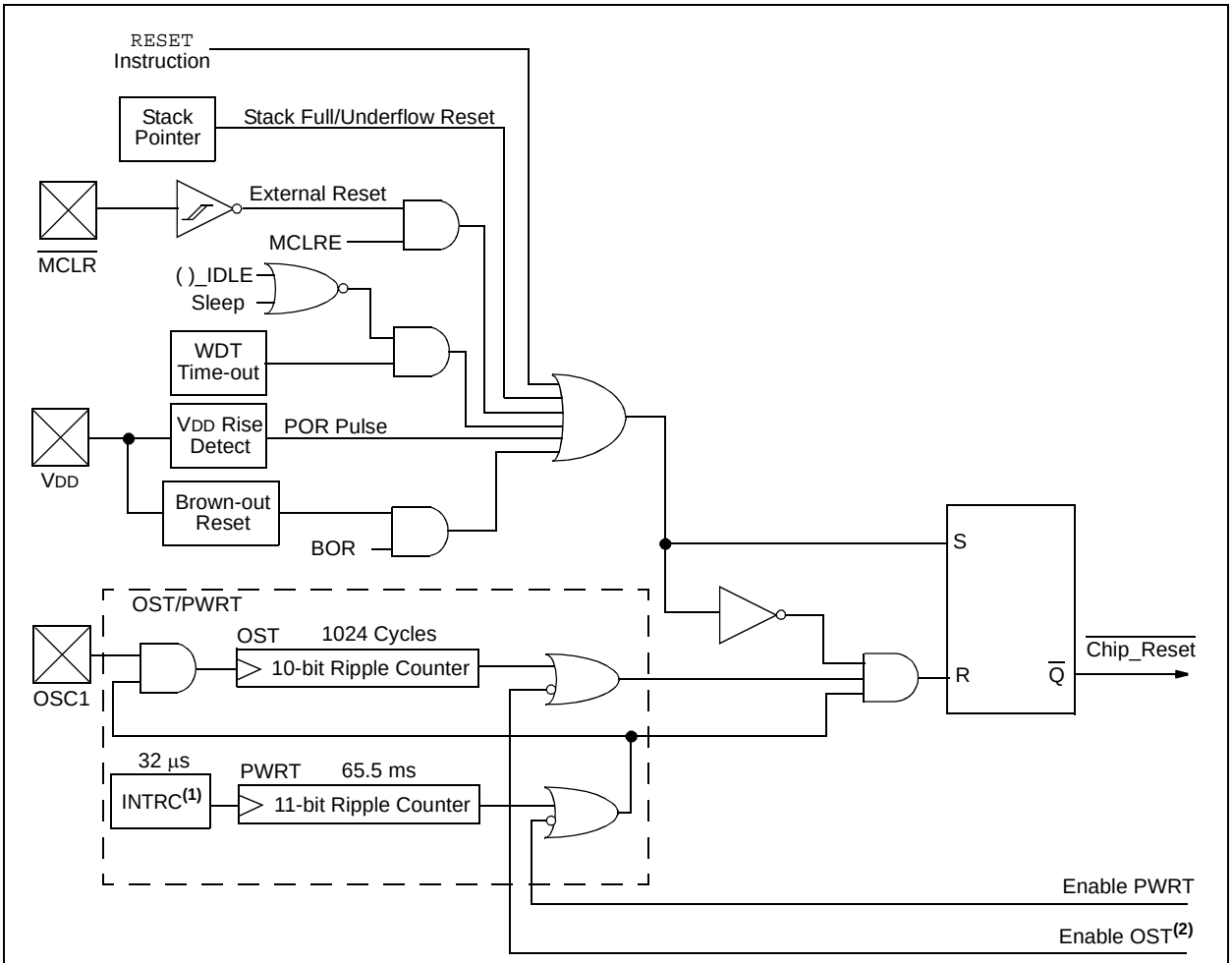
A simplified block diagram of the On-Chip Reset Circuit is shown in Figure 4-1.

The Enhanced MCU devices have a $\overline{\text{MCLR}}$ noise filter in the $\overline{\text{MCLR}}$ Reset path. The filter will detect and ignore small pulses.

The $\overline{\text{MCLR}}$ pin is not driven low by any internal Resets, including the WDT.

The $\overline{\text{MCLR}}$ input provided by the $\overline{\text{MCLR}}$ pin can be disabled with the MCLRE bit in Configuration Register 3H (CONFIG3H<7>).

FIGURE 4-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



Note 1: This is a separate oscillator from the RC oscillator of the CLKI pin.
2: See Table 4-1 for time-out situations.

TABLE 4-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Applicable Devices		Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets WDT Reset RESET Instruction Stack Resets	Wake-up via WDT or Interrupt
TOSU	1220	1320	---0 0000	---0 0000	---0 uuuu(3)
TOSH	1220	1320	0000 0000	0000 0000	uuuu uuuu(3)
TOSL	1220	1320	0000 0000	0000 0000	uuuu uuuu(3)
STKPTR	1220	1320	00-0 0000	00-0 0000	uu-u uuuu(3)
PCLATU	1220	1320	---0 0000	---0 0000	---u uuuu
PCLATH	1220	1320	0000 0000	0000 0000	uuuu uuuu
PCL	1220	1320	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	1220	1320	--00 0000	--00 0000	--uu uuuu
TBLPTRH	1220	1320	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	1220	1320	0000 0000	0000 0000	uuuu uuuu
TABLAT	1220	1320	0000 0000	0000 0000	uuuu uuuu
PRODH	1220	1320	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	1220	1320	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	1220	1320	0000 000x	0000 000u	uuuu uuuu(1)
INTCON2	1220	1320	1111 -1-1	1111 -1-1	uuuu -u-u(1)
INTCON3	1220	1320	11-0 0-00	11-0 0-00	uu-u u-uu(1)
INDF0	1220	1320	N/A	N/A	N/A
POSTINC0	1220	1320	N/A	N/A	N/A
POSTDEC0	1220	1320	N/A	N/A	N/A
PREINC0	1220	1320	N/A	N/A	N/A
PLUSW0	1220	1320	N/A	N/A	N/A
FSR0H	1220	1320	---- 0000	---- 0000	---- uuuu
FSR0L	1220	1320	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	1220	1320	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	1220	1320	N/A	N/A	N/A
POSTINC1	1220	1320	N/A	N/A	N/A
POSTDEC1	1220	1320	N/A	N/A	N/A
PREINC1	1220	1320	N/A	N/A	N/A
PLUSW1	1220	1320	N/A	N/A	N/A
FSR1H	1220	1320	---- 0000	---- 0000	---- uuuu
FSR1L	1220	1320	xxxx xxxx	uuuu uuuu	uuuu uuuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition. Shaded cells indicate conditions do not apply for the designated device.

Note 1: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

4: See Table 4-2 for Reset value for specific condition.

5: Bits 6 and 7 of PORTA, LATA and TRISA are enabled, depending on the Oscillator mode selected. When not enabled as PORTA pins, they are disabled and read '0'.

6: Bit 5 of PORTA is enabled if $\overline{\text{MCLR}}$ is disabled.

5.12 Indirect Addressing, INDF and FSR Registers

Indirect addressing is a mode of addressing data memory, where the data memory address in the instruction is not fixed. An FSR register is used as a pointer to the data memory location that is to be read or written. Since this pointer is in RAM, the contents can be modified by the program. This can be useful for data tables in the data memory and for software stacks. Figure 5-8 shows how the fetched instruction is modified prior to being executed.

Indirect addressing is possible by using one of the INDF registers. Any instruction, using the INDF register, actually accesses the register pointed to by the File Select Register, FSR. Reading the INDF register itself, indirectly (FSR = 0), will read 00h. Writing to the INDF register indirectly, results in a no operation (NOP). The FSR register contains a 12-bit address, which is shown in Figure 5-9.

The INDFn register is not a physical register. Addressing INDFn actually addresses the register whose address is contained in the FSRn register (FSRn is a pointer). This is indirect addressing.

Example 5-5 shows a simple use of indirect addressing to clear the RAM in Bank 1 (locations 100h-1FFh) in a minimum number of instructions.

EXAMPLE 5-5: HOW TO CLEAR RAM (BANK 1) USING INDIRECT ADDRESSING

```
        LFSR   FSR0,0x100  ;
NEXT    CLRF   POSTINC0    ; Clear INDF
                           ; register then
                           ; inc pointer
        BTFSS  FSR0H, 1    ; All done with
                           ; Bank1?
        GOTO   NEXT        ; NO, clear next
CONTINUE                   ; YES, continue
```

There are three indirect addressing registers. To address the entire data memory space (4096 bytes), these registers are 12-bit wide. To store the 12 bits of addressing information, two 8-bit registers are required:

1. FSR0: composed of FSR0H:FSR0L
2. FSR1: composed of FSR1H:FSR1L
3. FSR2: composed of FSR2H:FSR2L

In addition, there are registers INDF0, INDF1 and INDF2, which are not physically implemented. Reading or writing to these registers activates indirect addressing, with the value in the corresponding FSR register being the address of the data. If an instruction writes a value to INDF0, the value will be written to the address pointed to by FSR0H:FSR0L. A read from INDF1 reads the data from the address pointed to by FSR1H:FSR1L. INDFn can be used in code anywhere an operand can be used.

If INDF0, INDF1 or INDF2 are read indirectly via an FSR, all '0's are read (zero bit is set). Similarly, if INDF0, INDF1 or INDF2 are written to indirectly, the operation will be equivalent to a NOP instruction and the Status bits are not affected.

5.12.1 INDIRECT ADDRESSING OPERATION

Each FSR register has an INDF register associated with it, plus four additional register addresses. Performing an operation using one of these five registers determines how the FSR will be modified during indirect addressing.

When data access is performed using one of the five INDFn locations, the address selected will configure the FSRn register to:

- Do nothing to FSRn after an indirect access (no change) – INDFn
- Auto-decrement FSRn after an indirect access (post-decrement) – POSTDECn
- Auto-increment FSRn after an indirect access (post-increment) – POSTINCn
- Auto-increment FSRn before an indirect access (pre-increment) – PREINCn
- Use the value in the WREG register as an offset to FSRn. Do not modify the value of the WREG or the FSRn register after an indirect access (no change) – PLUSWn

When using the auto-increment or auto-decrement features, the effect on the FSR is not reflected in the Status register. For example, if the indirect address causes the FSR to equal '0', the Z bit will not be set.

Auto-incrementing or auto-decrementing an FSR affects all 12 bits. That is, when FSRnL overflows from an increment, FSRnH will be incremented automatically.

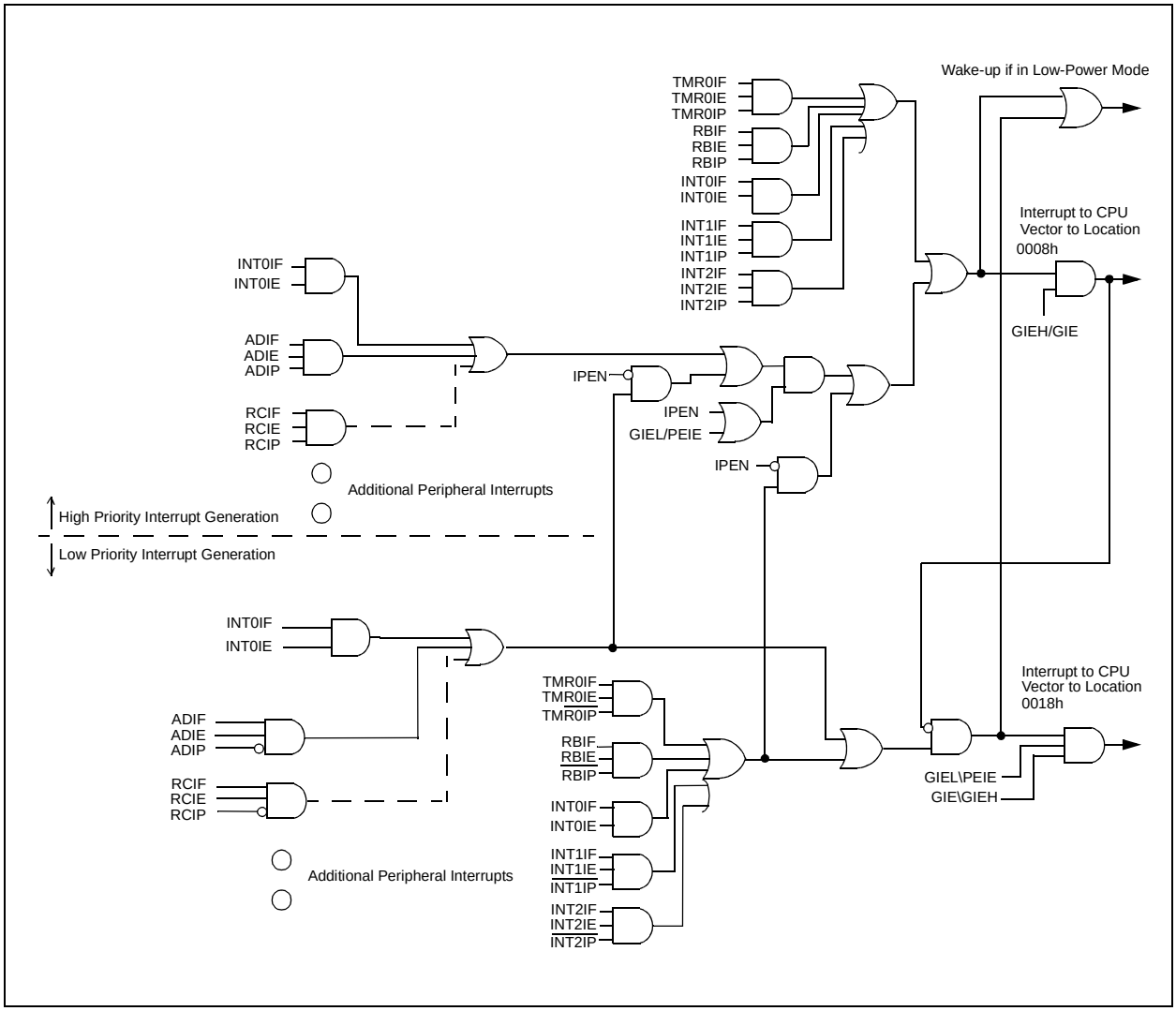
Adding these features allows the FSRn to be used as a stack pointer, in addition to its uses for table operations in data memory.

Each FSR has an address associated with it that performs an indexed indirect access. When a data access to this INDFn location (PLUSWn) occurs, the FSRn is configured to add the signed value in the WREG register and the value in FSR to form the address before an indirect access. The FSR value is not changed. The WREG offset range is -128 to +127.

If an FSR register contains a value that points to one of the INDFn, an indirect read will read 00h (zero bit is set), while an indirect write will be equivalent to a NOP (Status bits are not affected).

If an indirect addressing write is performed when the target address is an FSRnH or FSRnL register, the data is written to the FSR register, but no pre- or post-increment/decrement is performed.

FIGURE 9-1: INTERRUPT LOGIC



9.2 PIR Registers

The PIR registers contain the individual flag bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are two Peripheral Interrupt Request (Flag) registers (PIR1, PIR2).

Note 1: 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, GIE (INTCON<7>).

2: User software should ensure the appropriate interrupt flag bits are cleared prior to enabling an interrupt and after servicing that interrupt.

REGISTER 9-4: PIR1: PERIPHERAL INTERRUPT REQUEST REGISTER 1

U-0	R/W-0/0	R-0/0	R-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	ADIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF
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 **Unimplemented:** Read as ‘0’

bit 6 **ADIF:** A/D Converter Interrupt Flag bit
1 = An A/D conversion completed (must be cleared in software)
0 =The A/D conversion is not complete

bit 5 **RCIF:** EUSART Receive Interrupt Flag bit
1 = The EUSART receive buffer, RCREG, is full (cleared when RCREG is read)
0 = The EUSART receive buffer is empty

bit 4 **TXIF:** EUSART Transmit Interrupt Flag bit
1 = The EUSART transmit buffer, TXREG, is empty (cleared when TXREG is written)
0 = The EUSART transmit buffer is full

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **CCP1IF:** CCP1 Interrupt Flag bit
Capture mode:
1 = A TMR1 register capture occurred (must be cleared in software)
0 = No TMR1 register capture occurred
Compare mode:
1 = A TMR1 register compare match occurred (must be cleared in software)
0 = No TMR1 register compare match occurred
PWM mode:
Unused in this mode.

bit 1 **TMR2IF:** TMR2 to PR2 Match Interrupt Flag bit
1 = TMR2 to PR2 match occurred (must be cleared in software)
0 = No TMR2 to PR2 match occurred

bit 0 **TMR1IF:** TMR1 Overflow Interrupt Flag bit
1 = TMR1 register overflowed (must be cleared in software)
0 = TMR1 register did not overflow

9.4 IPR Registers

The IPR registers contain the individual priority bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are two Peripheral Interrupt Priority registers (IPR1, IPR2). Using the priority bits requires that the Interrupt Priority Enable (IPEN) bit be set.

REGISTER 9-8: IPR1: PERIPHERAL INTERRUPT PRIORITY REGISTER 1

U-0	R/W-1/1	R/W-1/1	R/W-1/1	U-0	R/W-1/1	R/W-1/1	R/W-1/1
—	ADIP	RCIP	TXIP	—	CCP1IP	TMR2IP	TMR1IP
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 **Unimplemented:** Read as ‘0’

bit 6 **ADIP:** A/D Converter Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **RCIP:** EUSART Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TXIP:** EUSART Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **CCP1IP:** CCP1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR2IP:** TMR2 to PR2 Match Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR1IP:** TMR1 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

TABLE 10-3: PORTB FUNCTIONS

Name	Bit#	Buffer	Function
RB0/AN4/INT0	bit 0	TTL(1)/ST(2)	Input/output port pin, analog input or external interrupt input 0.
RB1/AN5/TX/CK/INT1	bit 1	TTL(1)/ST(2)	Input/output port pin, analog input, Enhanced USART Asynchronous Transmit, Addressable USART Synchronous Clock or external interrupt input 1.
RB2/P1B/INT2	bit 2	TTL(1)/ST(2)	Input/output port pin or external interrupt input 2. Internal software programmable weak pull-up.
RB3/CCP1/P1A	bit 3	TTL(1)/ST(3)	Input/output port pin or Capture1 input/Compare1 output/ PWM output. Internal software programmable weak pull-up.
RB4/AN6/RX/DT/KBI0	bit 4	TTL(1)/ST(4)	Input/output port pin (with interrupt-on-change), analog input, Enhanced USART Asynchronous Receive or Addressable USART Synchronous Data.
RB5/PGM/KBI1	bit 5	TTL(1)/ST(5)	Input/output port pin (with interrupt-on-change). Internal software programmable weak pull-up. Low-Voltage ICSP™ enable pin.
RB6/PGC/T1OSO/T13CKI/ P1C/KBI2	bit 6	TTL(1)/ST(5,6)	Input/output port pin (with interrupt-on-change), Timer1/ Timer3 clock input or Timer1oscillator output. Internal software programmable weak pull-up. Serial programming clock.
RB7/PGD/T1OSI/P1D/KBI3	bit 7	TTL(1)/ST(5)	Input/output port pin (with interrupt-on-change) or Timer1 oscillator input. Internal software programmable weak pull-up. Serial programming data.

Legend: TTL = TTL input, ST = Schmitt Trigger input

Note 1: This buffer is a TTL input when configured as a port input pin.

2: This buffer is a Schmitt Trigger input when configured as the external interrupt.

3: This buffer is a Schmitt Trigger input when configured as the CCP1 input.

4: This buffer is a Schmitt Trigger input when used as EUSART receive input.

5: This buffer is a Schmitt Trigger input when used in Serial Programming mode.

6: This buffer is a TTL input when used as the T13CKI input.

TABLE 10-4: SUMMARY OF REGISTERS ASSOCIATED WITH PORTB

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
PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxq qqqq	uuuu uuuu
LATB	LATB Data Output Register								xxxx xxxx	uuuu uuuu
TRISB	PORTB Data Direction Register								1111 1111	1111 1111
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP	1111 -1-1	1111 -1-1
INTCON3	INT2IP	INT1IP	—	INT2IE	INT1IE	—	INT2IF	INT1IF	11-0 0-00	11-0 0-00
ADCON1	—	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0	-000 0000	-000 0000

Legend: x = unknown, u = unchanged, q = value depends on condition. Shaded cells are not used by PORTB.

16.2 EUSART Baud Rate Generator (BRG)

The BRG is a dedicated 8-bit or 16-bit generator, that supports both the Asynchronous and Synchronous modes of the EUSART. By default, the BRG operates in 8-bit mode; setting the BRG16 bit (BAUDCTL<3>) selects 16-bit mode.

The SPBRGH:SPBRG register pair controls the period of a free running timer. In Asynchronous mode, bits BRGH (TXSTA<2>) and BRG16 also control the baud rate. In Synchronous mode, bit BRGH is ignored. Table 16-1 shows the formula for computation of the baud rate for different EUSART modes which only apply in Master mode (internally generated clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRGH:SPBRG registers can be calculated using the formulas in Table 16-1. From this, the error in baud rate can be determined. An example calculation is shown in Example 16-1. Typical baud rates and error values for the various asynchronous modes are shown in Table 16-2. It may be advantageous to use the high baud rate (BRGH = 1), or the 16-bit BRG to reduce the baud rate error, or achieve a slow baud rate for a fast oscillator frequency.

Writing a new value to the SPBRGH:SPBRG registers causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

16.2.1 POWER MANAGED MODE OPERATION

The system clock is used to generate the desired baud rate; however, when a power managed mode is entered, the clock source may be operating at a different frequency than in PRI_RUN mode. In Sleep mode, no clocks are present and in PRI_IDLE mode, the primary clock source continues to provide clocks to the Baud Rate Generator; however, in other power managed modes, the clock frequency will probably change. This may require the value in SPBRG to be adjusted.

If the system clock is changed during an active receive operation, a receive error or data loss may result. To avoid this problem, check the status of the RCIDL bit and make sure that the receive operation is Idle before changing the system clock.

16.2.2 SAMPLING

The data on the RB4/AN6/RX/DT/KBI0 pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX pin.

TABLE 16-1: BAUD RATE FORMULAS

Configuration Bits			BRG/EUSART Mode	Baud Rate Formula
SYNC	BRG16	BRGH		
0	0	0	8-bit/Asynchronous	FOSC/[64 (n + 1)]
0	0	1	8-bit/Asynchronous	FOSC/[16 (n + 1)]
0	1	0	16-bit/Asynchronous	
0	1	1	16-bit/Asynchronous	FOSC/[4 (n + 1)]
1	0	x	8-bit/Synchronous	
1	1	x	16-bit/Synchronous	

Legend: x = Don't care, n = value of SPBRGH:SPBRG register pair

EXAMPLE 16-1: CALCULATING BAUD RATE ERROR

For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, 8-bit BRG:
Desired Baud Rate= FOSC/(64 ([SPBRGH:SPBRG] + 1))
Solving for SPBRGH:SPBRG:
X = ((FOSC/Desired Baud Rate)/64) – 1
= ((16000000/9600)/64) – 1
= [25.042] = 25
Calculated Baud Rate=16000000/(64 (25 + 1))
= 9615
Error = (Calculated Baud Rate – Desired Baud Rate)/Desired Baud Rate
= (9615 – 9600)/9600 = 0.16%

To set up an Asynchronous 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 asynchronous serial port by clearing bit SYNC and setting bit SPEN.
3. If interrupts are desired, set enable bit TXIE.
4. If 9-bit transmission is desired, set transmit bit TX9. Can be used as address/data bit.
5. Enable the transmission by setting bit TXEN, which will also set bit TXIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit TX9D.
7. Load data to the TXREG register (starts transmission).

If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 16-2: EUSART TRANSMIT BLOCK DIAGRAM

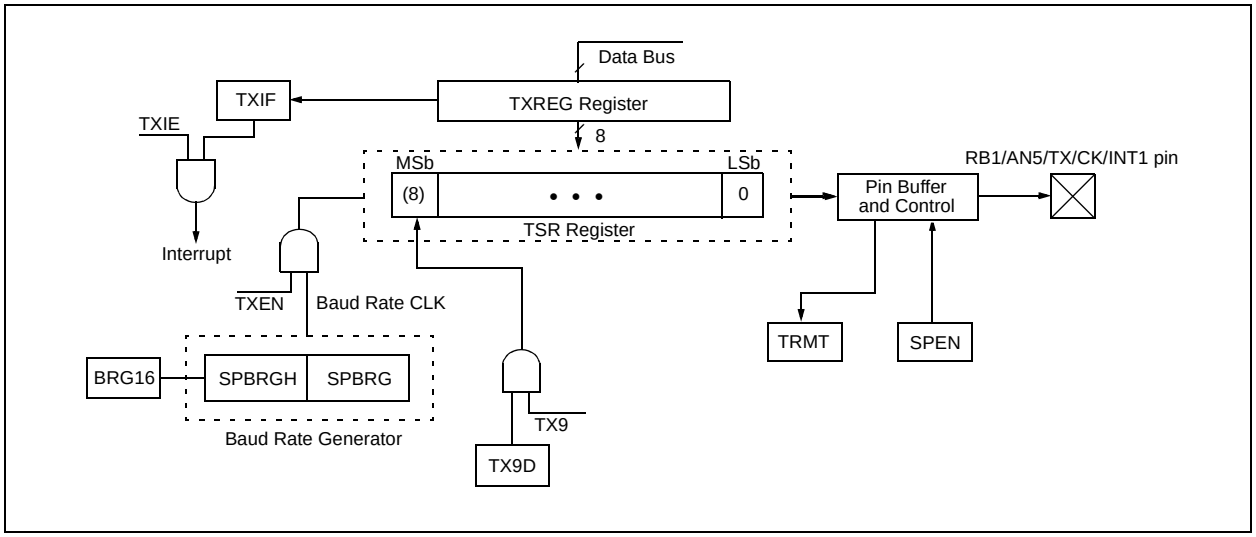
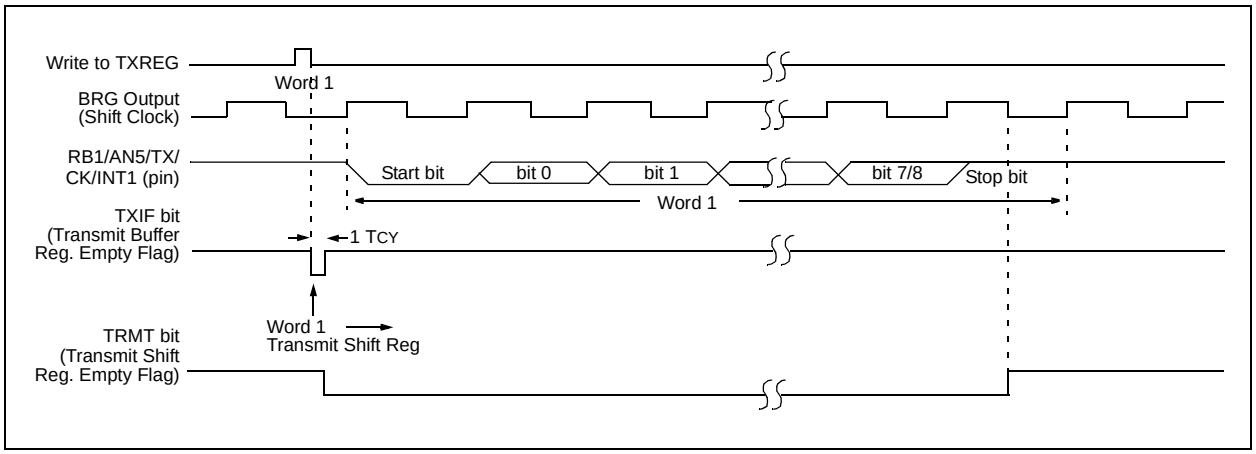


FIGURE 16-3: ASYNCHRONOUS TRANSMISSION



16.5.2 EUSART SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical, except in the case of Sleep, or any Idle mode and bit SREN, which is a “don’t care” in Slave mode.

If receive is enabled by setting the CREN bit prior to entering Sleep or any Idle mode, then a word may be received while in this low-power mode. Once the word is received, the RSR register will transfer the data to the RCREG register; if the RCIE enable bit is set, the interrupt generated will wake the chip from low-power mode. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Reception:

1. Enable the synchronous master serial port by setting bits SYNC and SPEN and clearing bit CSRC.
2. If interrupts are desired, set enable bit RCIE.
3. If 9-bit reception is desired, set bit RX9.
4. To enable reception, set enable bit CREN.
5. Flag bit, RCIF, will be set when reception is complete. An interrupt will be generated if enable bit, RCIE, was set.
6. Read the RCSTA register to get the ninth bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREG register.
8. If any error occurred, clear the error by clearing bit CREN.
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

TABLE 16-10: REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE RECEPTION

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
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR1	—	ADIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF	-000 -000	-000 -000
PIE1	—	ADIE	RCIE	TXIE	—	CCP1IE	TMR2IE	TMR1IE	-000 -000	-000 -000
IPR1	—	ADIP	RCIP	TXIP	—	CCP1IP	TMR2IP	TMR1IP	-111 -111	-111 -111
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
RCREG	EUSART Receive Register								0000 0000	0000 0000
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
BAUDCTL	—	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	-1-1 0-00	-1-1 0-00
SPBRGH	Baud Rate Generator Register High Byte								0000 0000	0000 0000
SPBRG	Baud Rate Generator Register Low Byte								0000 0000	0000 0000

Legend: x = unknown, – = unimplemented, read as ‘0’. Shaded cells are not used for synchronous slave reception.

17.0 10-BIT ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) converter module has seven inputs for the PIC18F1220/1320 devices. This module allows conversion of an analog input signal to a corresponding 10-bit digital number.

A new feature for the A/D converter is the addition of programmable acquisition time. This feature allows the user to select a new channel for conversion and to set the GO/$\overline{\text{DONE}}$ bit immediately. When the GO/$\overline{\text{DONE}}$ bit is set, the selected channel is sampled for the programmed acquisition time before a conversion is actually started. This removes the firmware overhead that may have been required to allow for an acquisition (sampling) period (see Register 17-3 and **Section 17.3 "Selecting and Configuring Automatic Acquisition Time"**).

The module has five registers:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)
- A/D Control Register 2 (ADCON2)

The ADCON0 register, shown in Register 17-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 17-2, configures the functions of the port pins. The ADCON2 register, shown in Register 17-3, configures the A/D clock source, programmed acquisition time and justification.

18.0 LOW-VOLTAGE DETECT

In many applications, the ability to determine if the device voltage (V_{DD}) is below a specified voltage level is a desirable feature. A window of operation for the application can be created, where the application software can do “housekeeping tasks”, before the device voltage exits the valid operating range. This can be done using the Low-Voltage Detect module.

This module is a software programmable circuitry, where a device voltage trip point can be specified. When the voltage of the device becomes lower then the specified point, an interrupt flag is set. If the interrupt is enabled, the program execution will branch to the interrupt vector address and the software can then respond to that interrupt source.

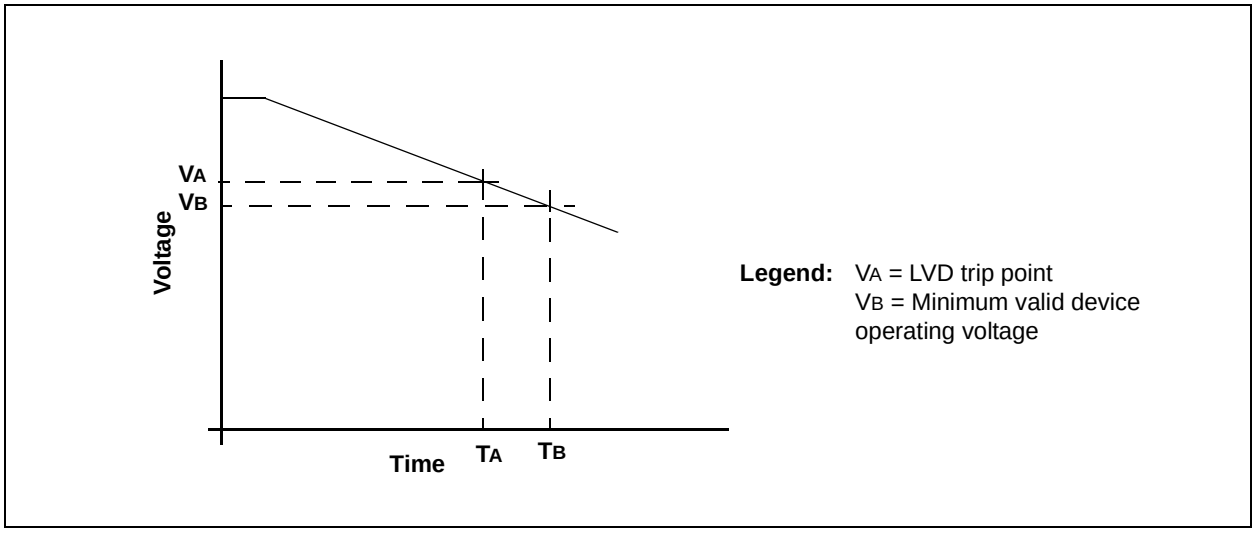
The Low-Voltage Detect circuitry is completely under software control. This allows the circuitry to be turned off by the software, which minimizes the current consumption for the device.

Figure 18-1 shows a possible application voltage curve (typically for batteries). Over time, the device voltage decreases. When the device voltage equals voltage V_A, the LVD logic generates an interrupt. This occurs at time T_A. The application software then has the time, until the device voltage is no longer in valid operating range, to shut down the system. Voltage point V_B is the minimum valid operating voltage specification. This occurs at time T_B. The difference, $T_B - T_A$, is the total time for shutdown.

The block diagram for the LVD module is shown in Figure 18-2 (following page). A comparator uses an internally generated reference voltage as the set point. When the selected tap output of the device voltage crosses the set point (is lower than), the LVDIF bit is set.

Each node in the resistor divider represents a “trip point” voltage. The “trip point” voltage is the minimum supply voltage level at which the device can operate before the LVD module asserts an interrupt. When the supply voltage is equal to the trip point, the voltage tapped off of the resistor array is equal to the 1.2V internal reference voltage generated by the voltage reference module. The comparator then generates an interrupt signal setting the LVDIF bit. This voltage is software programmable to any one of 16 values (see Figure 18-2). The trip point is selected by programming the LVDL3:LVDL0 bits (LVDCON<3:0>).

FIGURE 18-1: TYPICAL LOW-VOLTAGE DETECT APPLICATION



19.5 Program Verification and Code Protection

The overall structure of the code protection on the PIC18 Flash devices differs significantly from other PIC devices.

The user program memory is divided into three blocks. One of these is a boot block of 512 bytes. The remainder of the memory is divided into two blocks on binary boundaries.

Each of the three blocks has three protection bits associated with them. They are:

- Code-Protect bit (CPn)
- Write-Protect bit (WRTn)
- External Block Table Read bit (EBTRn)

Figure 19-5 shows the program memory organization for 4 and 8-Kbyte devices and the specific code protection bit associated with each block. The actual locations of the bits are summarized in Table 19-3.

FIGURE 19-5: CODE-PROTECTED PROGRAM MEMORY FOR PIC18F1220/1320

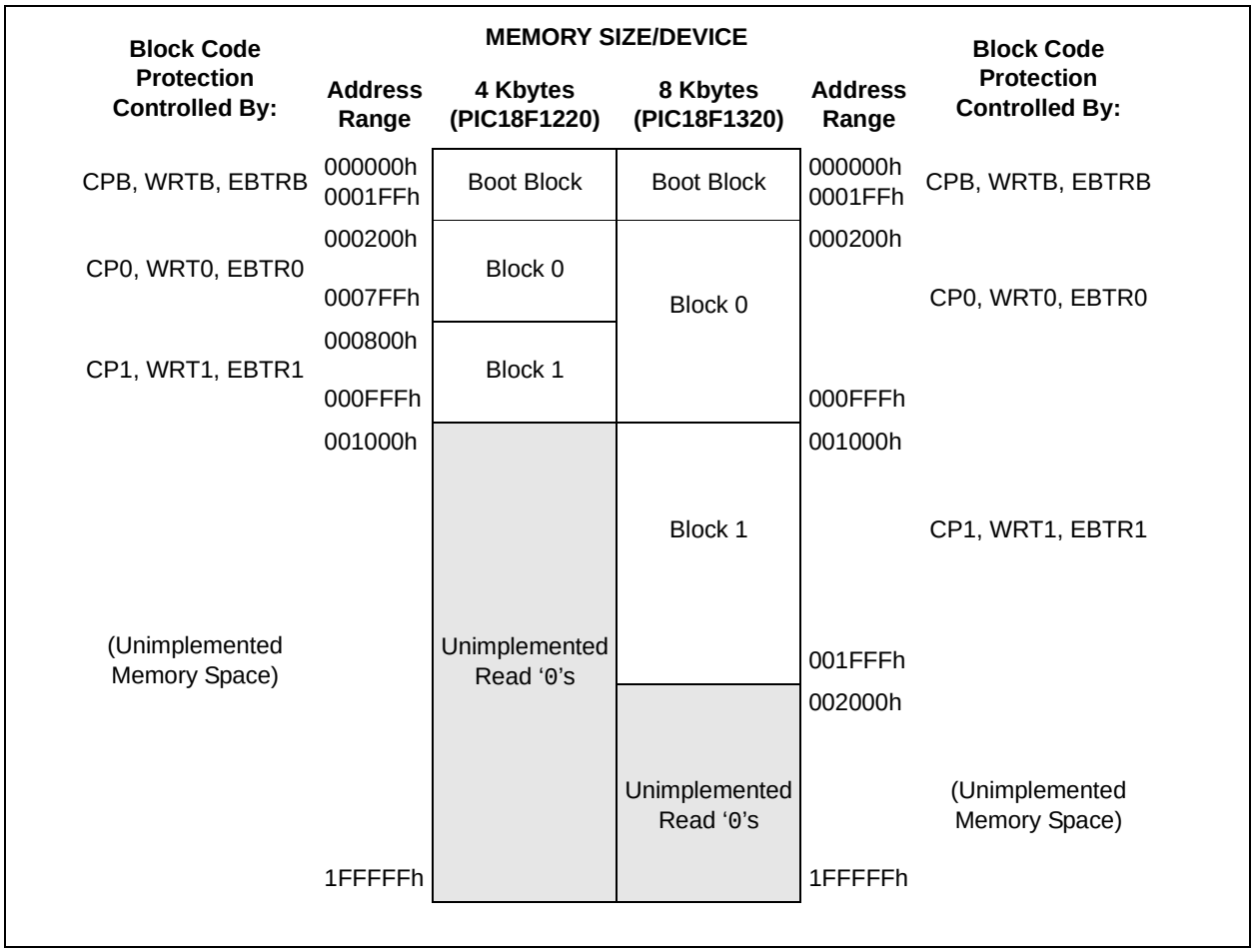


TABLE 19-3: SUMMARY OF CODE PROTECTION REGISTERS

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
300008h	CONFIG5L	—	—	—	—	—	—	CP1	CP0
300009h	CONFIG5H	CPD	CPB	—	—	—	—	—	—
30000Ah	CONFIG6L	—	—	—	—	—	—	WRT1	WRT0
30000Bh	CONFIG6H	WRTD	WRTB	WRTC	—	—	—	—	—
30000Ch	CONFIG7L	—	—	—	—	—	—	EBTR1	EBTR0
30000Dh	CONFIG7H	—	EBTRB	—	—	—	—	—	—

Legend: Shaded cells are unimplemented.

BNOV **Branch if Not Overflow**

Syntax: [*label*] BNOV n

Operands: -128 ≤ n ≤ 127

Operation: if Overflow bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0101	nnnn	nnnn

Description: If the Overflow bit is '0', 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 BNOV Jump

Before Instruction
PC = address (HERE)
After Instruction
If Overflow = 0;
PC = address (Jump)
If Overflow = 1;
PC = address (HERE + 2)

BNZ **Branch if Not Zero**

Syntax: [*label*] BNZ n

Operands: -128 ≤ n ≤ 127

Operation: if Zero bit is '0'
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0001	nnnn	nnnn

Description: If the Zero bit is '0', 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 BNZ Jump

Before Instruction
PC = address (HERE)
After Instruction
If Zero = 0;
PC = address (Jump)
If Zero = 1;
PC = address (HERE + 2)

DECFSZ **Decrement f, skip if 0**

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

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

Operation: (f) – 1 → dest,
skip if result = 0

Status Affected: None

Encoding:

0010	11da	ffff	ffff

Description: The contents of register 'f' are decremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).
If the result is '0', the next instruction, which is already fetched, is discarded and a NOP is executed instead, making it a 2-cycle instruction. 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(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	Write to destination

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    DECFSZ  CNT
        GOTO    LOOP
CONTINUE
```
Before Instruction
PC = Address (HERE)
After Instruction
CNT = CNT – 1
If CNT = 0;
PC = Address (CONTINUE)
If CNT ≠ 0;
PC = Address (HERE + 2)

DCFSNZ **Decrement f, skip if not 0**

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

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

Operation: (f) – 1 → dest,
skip if result ≠ 0

Status Affected: None

Encoding:

0100	11da	ffff	ffff

Description: The contents of register 'f' are decremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).
If the result is not '0', the next instruction, which is already fetched, is discarded and a NOP is executed instead, making it a 2-cycle instruction. 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(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	Write to destination

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    DCFSNZ TEMP
ZERO    :
NZERO   :
```
Before Instruction
TEMP = ?
After Instruction
TEMP = TEMP – 1,
If TEMP = 0;
PC = Address (ZERO)
If TEMP ≠ 0;
PC = Address (NZERO)

22.2 DC Characteristics: Power-Down and Supply Current PIC18F1220/1320 (Industrial) PIC18LF1220/1320 (Industrial) (Continued)

PIC18LF1220/1320 (Industrial)	Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial
PIC18F1220/1320 (Industrial, Extended)	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended

Param No.	Device	Typ.	Max.	Units	Conditions		
	Supply Current (IDD)(2,3)						
	All devices	3.2	4.1	mA	-40°C	VDD = 4.2 V	FOSC = 40 MHz (**PRI_IDLE** mode, EC oscillator)
		3.2	4.1	mA	+25°C		
		3.3	4.1	mA	+85°C		
	All devices	4.0	5.1	mA	-40°C	VDD = 5.0V	
		4.1	5.1	mA	+25°C		
		4.1	5.1	mA	+85°C		
	PIC18LF1220/1320	5.1	9	µA	-10°C	VDD = 2.0V	FOSC = 32 kHz(4) (**SEC_RUN** mode, Timer1 as clock)
		5.8	9	µA	+25°C		
		7.9	11	µA	+70°C		
	PIC18LF1220/1320	7.9	12	µA	-10°C	VDD = 3.0V	
		8.9	12	µA	+25°C		
		10.5	14	µA	+70°C		
	All devices	12.5	20	µA	-10°C	VDD = 5.0V	
		16.3	20	µA	+25°C		
		18.4	25	µA	+70°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.

The test conditions for all IDD measurements in active operation mode are:

OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;

$\overline{MCLR}$ = VDD; WDT enabled/disabled as specified.

3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

4: Standard low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

22.2 DC Characteristics: Power-Down and Supply Current PIC18F1220/1320 (Industrial) PIC18LF1220/1320 (Industrial) (Continued)

PIC18LF1220/1320 (Industrial)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial					
PIC18F1220/1320 (Industrial, Extended)		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial -40°C ≤ TA ≤ +125°C for extended					
Param No.	**Device**	**Typ.**	**Max.**	**Units**	**Conditions**		
	Supply Current (IDD)(2,3)						
	PIC18LF1220/1320	9.2	15	µA	-10°C	VDD = 2.0V	FOSC = 32 kHz(4) (**SEC_IDLE** mode, Timer1 as clock)
		9.6	15	µA	+25°C		
		12.7	18	µA	+70°C		
	PIC18LF1220/1320	22	30	µA	-10°C	VDD = 3.0V	
		21	30	µA	+25°C		
		20	35	µA	+70°C		
	All devices	50	80	µA	-10°C	VDD = 5.0V	
		45	80	µA	+25°C		
		45	80	µA	+70°C		

Legend: Shading of rows is to assist in readability of the table.

Note 1: The power-down current in Sleep mode does not depend on the oscillator type. Power-down current is measured with the part in Sleep mode, with all I/O pins in high-impedance state and tied to VDD or VSS and all features that add delta current disabled (such as WDT, Timer1 Oscillator, BOR, etc.).

2: The supply current is mainly a function of operating voltage, frequency and mode. Other factors, such as I/O pin loading and switching rate, oscillator type and circuit, internal code execution pattern and temperature, also have an impact on the current consumption.
The test conditions for all IDD measurements in active operation mode are:
OSC1 = external square wave, from rail-to-rail; all I/O pins tri-stated, pulled to VDD;
$\overline{\text{MCLR}}$ = VDD; WDT enabled/disabled as specified.

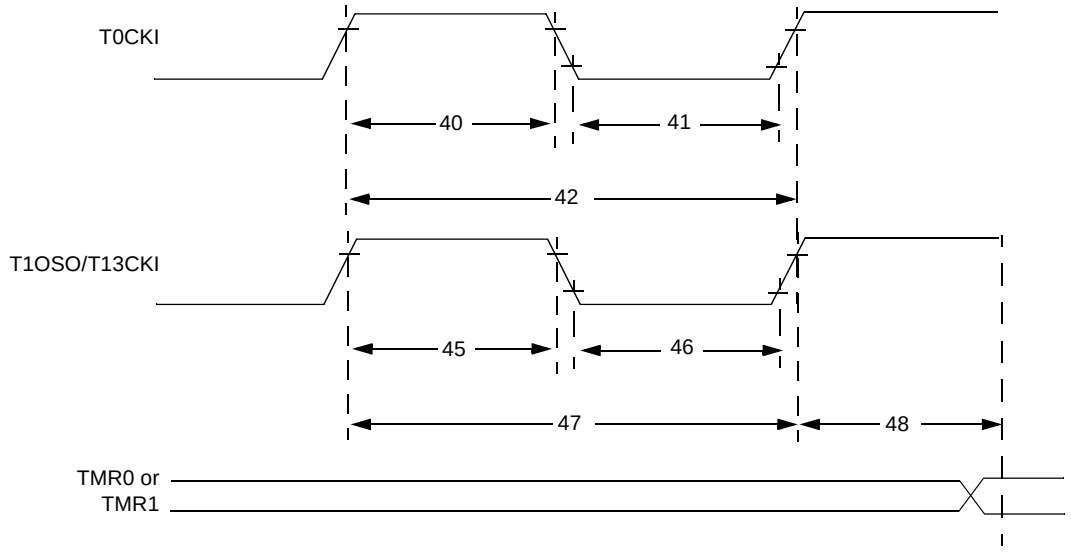
3: For RC oscillator configurations, current through REXT is not included. The current through the resistor can be estimated by the formula Ir = VDD/2REXT (mA) with REXT in kΩ.

4: Standard low-cost 32 kHz crystals have an operating temperature range of -10°C to +70°C. Extended temperature crystals are available at a much higher cost.

TABLE 22-8: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET REQUIREMENTS

Param. No.	Symbol	Characteristic	Min.	Typ.	Max.	Units	Conditions
30	TmcL	$\overline{MCLR}$ Pulse Width (low)	2	—	—	µs	
31	TWDT	Watchdog Timer Time-out Period (No postscaler)	3.48	4.00	4.71	ms	
32	TOST	Oscillation Start-up Timer Period	1024 TOSC	—	1024 TOSC	—	TOSC = OSC1 period
33	TPWRT	Power-up Timer Period	—	65.5	132	ms	
34	TIOZ	I/O High-Impedance from $\overline{MCLR}$ Low or Watchdog Timer Reset	—	2	—	µs	
35	TBOR	Brown-out Reset Pulse Width	200	—	—	µs	VDD ≤ BVDD (see D005)
36	TIVRST	Time for Internal Reference Voltage to become stable	—	20	50	µs	
37	TLVD	Low-Voltage Detect Pulse Width	200	—	—	µs	VDD ≤ VLVD

FIGURE 22-10: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

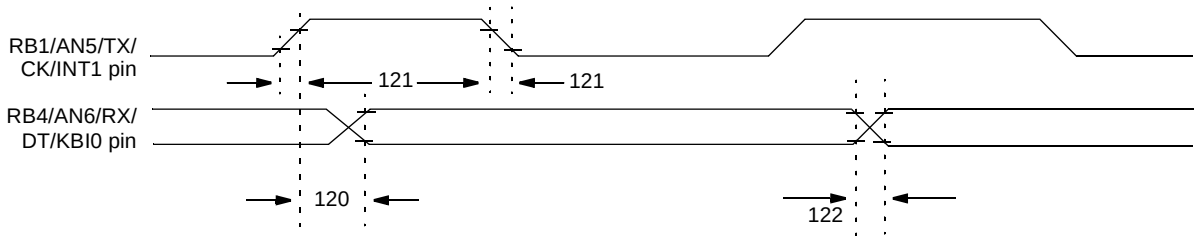


Note: Refer to Figure 22-5 for load conditions.

TABLE 22-10: CAPTURE/COMPARE/PWM REQUIREMENTS (ALL CCP MODULES)

Param. No.	Symbol	Characteristic			Min.	Max.	Units	Conditions
50	TccL	CCPx Input Low Time	No prescaler		0.5 TCY + 20	—	ns	
			With prescaler	PIC18F1X20	10	—	ns	
				PIC18LF1X20	20	—	ns	
51	TccH	CCPx Input High Time	No prescaler		0.5 TCY + 20	—	ns	
			With prescaler	PIC18F1X20	10	—	ns	
				PIC18LF1X20	20	—	ns	
52	TccP	CCPx Input Period			3 TCY + 40 / N	—	ns	N = prescale value (1, 4 or 16)
53	TccR	CCPx Output Fall Time		PIC18F1X20	—	25	ns	
				PIC18LF1X20	—	45	ns	
54	TccF	CCPx Output Fall Time		PIC18F1X20	—	25	ns	
				PIC18LF1X20	—	45	ns	

FIGURE 22-12: EUSART SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING

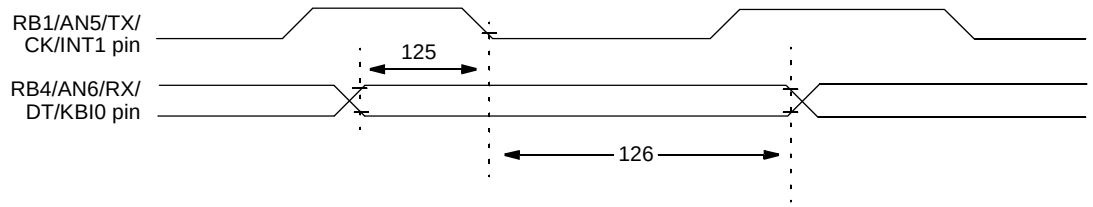


Note: Refer to Figure 22-5 for load conditions.

TABLE 22-11: EUSART SYNCHRONOUS TRANSMISSION REQUIREMENTS

Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
120	TckH2dtV	SYNC XMIT (MASTER & SLAVE) Clock High to Data Out Valid	PIC18F1X20	—	40	ns	
			PIC18LF1X20	—	100	ns	
121	Tckrf	Clock Out Rise Time and Fall Time (Master mode)	PIC18F1X20	—	20	ns	
			PIC18LF1X20	—	50	ns	
122	Tdtrf	Data Out Rise Time and Fall Time	PIC18F1X20	—	20	ns	
			PIC18LF1X20	—	50	ns	

FIGURE 22-13: EUSART SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING

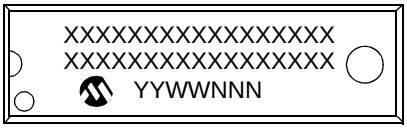


Note: Refer to Figure 22-5 for load conditions.

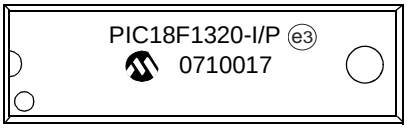
24.0 PACKAGING INFORMATION

24.1 Package Marking Information

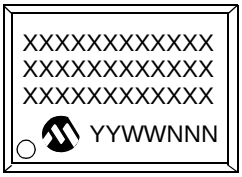
18-Lead PDIP



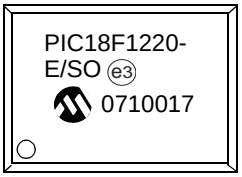
Example



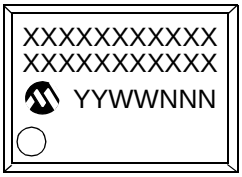
18-Lead SOIC



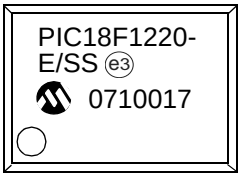
Example



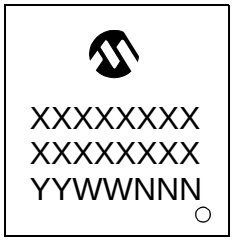
20-Lead SSOP



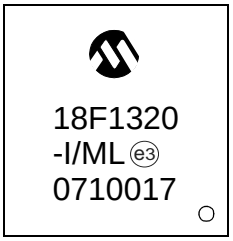
Example



28-Lead QFN



Example



Legend:	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week ‘01’)
	NNN	Alphanumeric traceability code
	(e3)	Pb-free JEDEC designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC designator ((e3)) can be found on the outer packaging for this package.

Note: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.