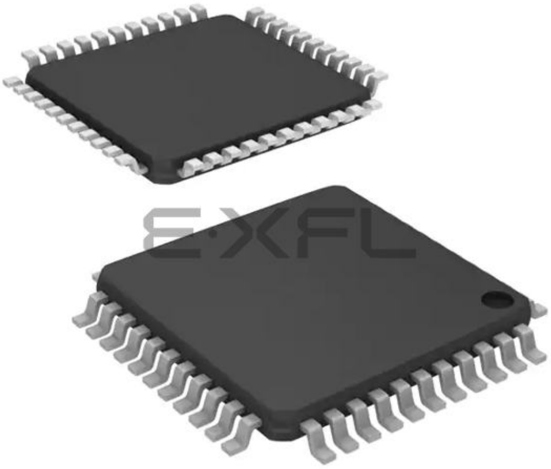


Welcome to E-XFL.COM

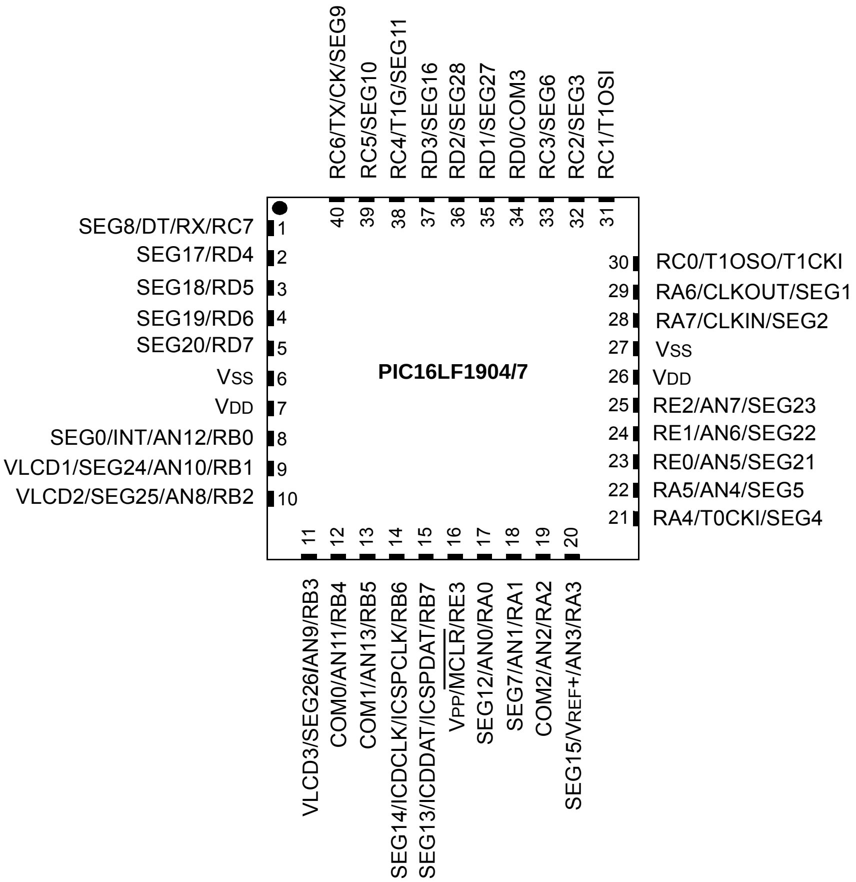
What is "Embedded - Microcontrollers"?

"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	20MHz
Connectivity	LINbus, UART/USART
Peripherals	Brown-out Detect/Reset, LCD, POR, PWM, WDT
Number of I/O	36
Program Memory Size	14KB (8K x 14)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	512 x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 14x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-TQFP
Supplier Device Package	44-TQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf1907t-i-pt

FIGURE 5: 40-PIN UQFN (5X5) PACKAGE DIAGRAM FOR PIC16LF1904/7



3.2.1.1 STATUS Register

The STATUS register, shown in Register 3-1, contains:

- the arithmetic status of the ALU
- the Reset status

The STATUS register can be the destination for any instruction, like any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the $\overline{\text{TO}}$ and $\overline{\text{PD}}$ bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, `CLRF STATUS` will clear the upper three bits and set the Z bit. This leaves the STATUS register as '`000u u1uu`' (where u = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register, because these instructions do not affect any Status bits. For other instructions not affecting any Status bits (Refer to **Section 21.0 "Instruction Set Summary"**).

Note:	The C and DC bits operate as $\overline{\text{Borrow}}$ and $\overline{\text{Digit Borrow}}$ out bits, respectively, in subtraction.

REGISTER 3-1: STATUS: STATUS REGISTER

U-0	U-0	U-0	R-1/q	R-1/q	R/W-0/u	R/W-0/u	R/W-0/u
—	—	—	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC[(1)]	C[(1)]
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	q = Value depends on condition

bit 7-5 **Unimplemented:** Read as '`0`'

bit 4 **$\overline{\text{TO}}$:** Time-out bit
`1` = After power-up, `CLRWDT` instruction or `SLEEP` instruction
`0` = A WDT time-out occurred

bit 3 **$\overline{\text{PD}}$:** Power-Down bit
`1` = After power-up or by the `CLRWDT` instruction
`0` = By execution of the `SLEEP` instruction

bit 2 **Z:** Zero bit
`1` = The result of an arithmetic or logic operation is zero
`0` = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit Carry/$\overline{\text{Digit Borrow}}$ bit (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)[(1)]
`1` = A carry-out from the 4th low-order bit of the result occurred
`0` = No carry-out from the 4th low-order bit of the result

bit 0 **C:** Carry/$\overline{\text{Borrow}}$ bit[(1)] (`ADDWF`, `ADDLW`, `SUBLW`, `SUBWF` instructions)[(1)]
`1` = A carry-out from the Most Significant bit of the result occurred
`0` = No carry-out from the Most Significant bit of the result occurred

Note 1: For $\overline{\text{Borrow}}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high-order or low-order bit of the source register.

TABLE 3-5: SPECIAL FUNCTION REGISTER SUMMARY (CONTINUED)

Addr	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
Bank 2											
10Ch	LATA	PORTA Data Latch								xxxx xxxx	uuuu uuuu
10Dh	LATB	PORTB Data Latch								xxxx xxxx	uuuu uuuu
10Eh	LATC	PORTC Data Latch								xxxx xxxx	uuuu uuuu
10Eh	LATD(3)	PORTD Data Latch								xxxx xxxx	uuuu uuuu
10Eh	LATE(3)	—	—	—	—	—	LATE2	LATE1	LATE0	---- -xxx	---- -uuu
111h to 115h	—	Unimplemented								—	—
116h	BORCON	SBOREN	BORFS	—	—	—	—	—	BORRDY	10-- ---q	uu-- ---u
117h	FVRCON	FVREN	FVRRDY	TSEN	TSRNG	—	—	ADFVR1	ADFVR0	0q00 --00	0q00 --00
118h to 11Fh	—	Unimplemented								—	—
Bank 3											
18Ch	ANSELA	—	—	ANSA5	—	ANSA3	ANSA2	ANSA1	ANSA0	--1- 1111	--11 1111
18Dh	ANSELB	—	—	ANSB5	ANSB4	ANSB3	ANSB2	ANSB1	ANSB0	--11 1111	--11 1111
18Eh	—	Unimplemented								—	—
18Fh	—	Unimplemented								—	—
190h	ANSELE(3)	—	—	—	—	—	ANSE2	ANSE1	ANSE0	---- -111	---- -111
191h	PMADRL	Program Memory Address Register Low Byte								0000 0000	0000 0000
192h	PMADRH	—(2)	Program Memory Address Register High Byte							1000 0000	1000 0000
193h	PMDATL	Program Memory Read Data Register Low Byte								xxxx xxxx	uuuu uuuu
194h	PMDATH	—	—	Program Memory Read Data Register High Byte						--xx xxxx	--uu uuuu
195h	PMCON1	—(2)	CFGS	LWLO	FREE	WRERR	WREN	WR	RD	1000 x000	1000 q000
196h	PMCON2	Program Memory Control Register 2								0000 0000	0000 0000
197h	—	Unimplemented								—	—
198h	—	Unimplemented								—	—
199h	RCREG	USART Receive Data Register								0000 0000	0000 0000
19Ah	TXREG	USART Transmit Data Register								0000 0000	0000 0000
19Bh	SPBRG	BRG<7:0>								0000 0000	0000 0000
19Ch	SPBRGH	BRG<15:8>								0000 0000	0000 0000
19Dh	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
19Eh	TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	0000 0010	0000 0010
19Fh	BAUD1CON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-0 0-00	01-0 0-00
Bank 4											
20Ch	—	Unimplemented								—	—
20Dh	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0	1111 1111	1111 1111
20Eh	—	Unimplemented								—	—
20Fh	—	Unimplemented								—	—
210h	WPUE	—	—	—	—	WPUE3	—	—	—	---- 1---	---- 1---
211h to 21Fh	—	Unimplemented								—	—
Bank 5											
28Ch — 29Fh	—	Unimplemented								—	—
Bank 6											
30Ch — 31Fh	—	Unimplemented								—	—

Legend: x = unknown, u = unchanged, q = value depends on condition, - = unimplemented, read as ‘0’, r = reserved.
Shaded locations are unimplemented, read as ‘0’.

Note 1: These registers can be addressed from any bank.
2: Unimplemented, read as ‘1’.
3: PIC16LF1904/7 only.

7.1 Operation

Interrupts are disabled upon any device Reset. They are enabled by setting the following bits:

- GIE bit of the INTCON register
- Interrupt Enable bit(s) for the specific interrupt event(s)
- PEIE bit of the INTCON register (if the Interrupt Enable bit of the interrupt event is contained in the PIE1 and PIE2 registers)

The INTCON, PIR1 and PIR2 registers record individual interrupts via interrupt flag bits. Interrupt flag bits will be set, regardless of the status of the GIE, PEIE and individual interrupt enable bits.

The following events happen when an interrupt event occurs while the GIE bit is set:

- Current prefetched instruction is flushed
- GIE bit is cleared
- Current Program Counter (PC) is pushed onto the stack
- Critical registers are automatically saved to the shadow registers (See **Section 7.5 “Automatic Context Saving”**)
- PC is loaded with the interrupt vector 0004h

The firmware within the Interrupt Service Routine (ISR) should determine the source of the interrupt by polling the interrupt flag bits. The interrupt flag bits must be cleared before exiting the ISR to avoid repeated interrupts. Because the GIE bit is cleared, any interrupt that occurs while executing the ISR will be recorded through its interrupt flag, but will not cause the processor to redirect to the interrupt vector.

The `RETFIE` instruction exits the ISR by popping the previous address from the stack, restoring the saved context from the shadow registers and setting the GIE bit.

For additional information on a specific interrupt’s operation, refer to its peripheral chapter.

Note 1: Individual interrupt flag bits are set, regardless of the state of any other enable bits.

2: All interrupts will be ignored while the GIE bit is cleared. Any interrupt occurring while the GIE bit is clear will be serviced when the GIE bit is set again.

7.2 Interrupt Latency

Interrupt latency is defined as the time from when the interrupt event occurs to the time code execution at the interrupt vector begins. The latency for synchronous interrupts is three or four instruction cycles. For asynchronous interrupts, the latency is three to five instruction cycles, depending on when the interrupt occurs. See Figure 7-2 and Figure 7.3 for more details.

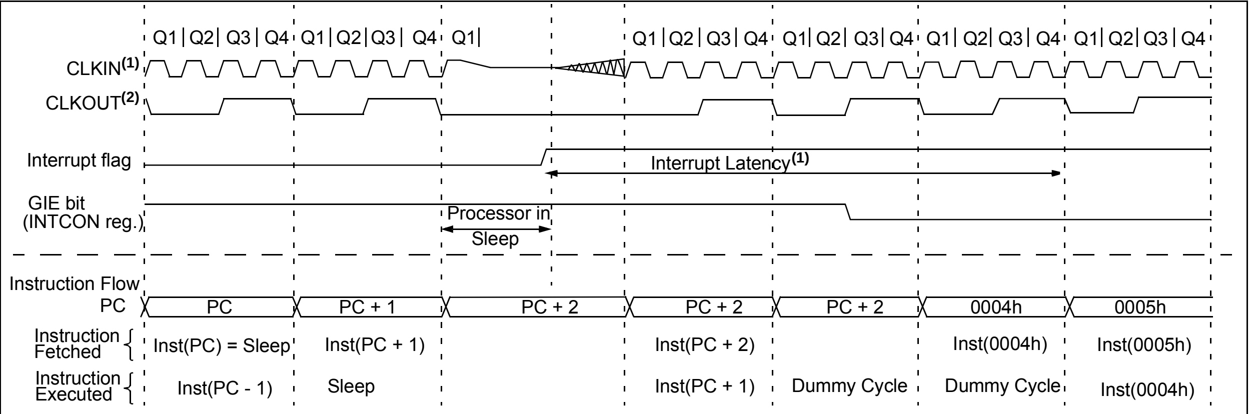
8.1.1 WAKE-UP USING INTERRUPTS

When global interrupts are disabled (GIE cleared) and any interrupt source has both its interrupt enable bit and interrupt flag bit set, one of the following will occur:

- If the interrupt occurs **before** the execution of a SLEEP instruction
  - SLEEP instruction will execute as a NOP.
  - WDT and WDT prescaler will not be cleared
  - $\overline{\text{TO}}$ bit of the STATUS register will not be set
  - $\overline{\text{PD}}$ bit of the STATUS register will not be cleared.
- If the interrupt occurs **during or after** the execution of a SLEEP instruction
  - SLEEP instruction will be completely executed
  - Device will immediately wake-up from Sleep
  - WDT and WDT prescaler will be cleared
  - $\overline{\text{TO}}$ bit of the STATUS register will be set
  - $\overline{\text{PD}}$ bit of the STATUS register will be cleared.

Even if the flag bits were checked before executing a SLEEP instruction, it may be possible for flag bits to become set before the SLEEP instruction completes. To determine whether a SLEEP instruction executed, test the $\overline{\text{PD}}$ bit. If the $\overline{\text{PD}}$ bit is set, the SLEEP instruction was executed as a NOP.

FIGURE 8-1: WAKE-UP FROM SLEEP THROUGH INTERRUPT



Note 1: GIE = 1 assumed. In this case after wake-up, the processor calls the ISR at 0004h. If GIE = 0, execution will continue in-line.

TABLE 8-1: SUMMARY OF REGISTERS ASSOCIATED WITH POWER-DOWN MODE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	65
IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0	110
IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0	110
IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0	110
PIE1	TMR1GIE	ADIE	RCIE	TXIE	—	—	—	TMR1IE	66
PIE2	—	—	—	—	—	LCDIE	—	—	67
PIR1	TMR1GIF	ADIF	RCIF	TXIF	—	—	—	TMR1IF	68
PIR2	—	—	—	—	—	LCDIF	—	—	69
STATUS	—	—	—	$\overline{\text{TO}}$	$\overline{\text{PD}}$	Z	DC	C	21
WDTCON	—	—	WDTPS<4:0>					SWDTEN	75

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used in Power-down mode.

EXAMPLE 10-3: WRITING TO FLASH PROGRAM MEMORY

```
; This write routine assumes the following:
; 1. 64 bytes of data are loaded, starting at the address in DATA_ADDR
; 2. Each word of data to be written is made up of two adjacent bytes in DATA_ADDR,
;    stored in little endian format
; 3. A valid starting address (the least significant bits = 00000) is loaded in ADDRH:ADDRL
; 4. ADDRH and ADDRL are located in shared data memory 0x70 - 0x7F (common RAM)
;
        BCF        INTCON,GIE     ; Disable ints so required sequences will execute properly
        BANKSEL    PMADRH         ; Bank 3
        MOVF       ADDRH,W        ; Load initial address
        MOVWF      PMADRH         ;
        MOVF       ADDRL,W        ;
        MOVWF      PMADRL         ;
        MOVLW      LOW DATA_ADDR  ; Load initial data address
        MOVWF      FSR0L          ;
        MOVLW      HIGH DATA_ADDR ; Load initial data address
        MOVWF      FSR0H          ;
        BCF        PMCON1,CFGS    ; Not configuration space
        BSF        PMCON1,WREN    ; Enable writes
        BSF        PMCON1,LWLO    ; Only Load Write Latches
LOOP
        MOVIW      FSR0++         ; Load first data byte into lower
        MOVWF      PMDATL         ;
        MOVIW      FSR0++         ; Load second data byte into upper
        MOVWF      PMDATH         ;

        MOVF       PMADRL,W       ; Check if lower bits of address are '00000'
        XORLW      0x1F           ; Check if we're on the last of 32 addresses
        ANDLW      0x1F           ;
        BTFSC      STATUS,Z       ; Exit if last of 32 words,
        GOTO       START_WRITE    ;

  Required Sequence
        MOVLW      55h            ; Start of required write sequence:
        MOVWF      PMCON2         ; Write 55h
        MOVLW      0AAh           ;
        MOVWF      PMCON2         ; Write AAh
        BSF        PMCON1,WR      ; Set WR bit to begin write
        NOP                       ; NOP instructions are forced as processor
                                  ; loads program memory write latches
        NOP                       ;

        INCF       PMADRL,F       ; Still loading latches Increment address
        GOTO       LOOP           ; Write next latches

START_WRITE
        BCF        PMCON1,LWLO    ; No more loading latches - Actually start Flash program
                                  ; memory write

  Required Sequence
        MOVLW      55h            ; Start of required write sequence:
        MOVWF      PMCON2         ; Write 55h
        MOVLW      0AAh           ;
        MOVWF      PMCON2         ; Write AAh
        BSF        PMCON1,WR      ; Set WR bit to begin write
        NOP                       ; NOP instructions are forced as processor writes
                                  ; all the program memory write latches simultaneously
        NOP                       ; to program memory.
                                  ; After NOPs, the processor
                                  ; stalls until the self-write process in complete
                                  ; after write processor continues with 3rd instruction
        BCF        PMCON1,WREN    ; Disable writes
        BSF        INTCON,GIE     ; Enable interrupts
```

REGISTER 11-5: PORTB: PORTB REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
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 7-0 **RB<7:0>**: PORTB General Purpose I/O Pin bits(1)
`1` = Port pin is > VIH
`0` = Port pin is < VIL

Note 1: Writes to PORTB are actually written to the corresponding LATB register. Reads from the PORTB register is return of actual I/O pin values.

REGISTER 11-6: TRISB: PORTB TRI-STATE REGISTER

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
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 7-0 **TRISB<7:0>:** PORTB Tri-State Control bits
`1` = PORTB pin configured as an input (tri-stated)
`0` = PORTB pin configured as an output

REGISTER 11-7: LATB: PORTB DATA LATCH REGISTER

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0
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 7-0 **LATB<7:0>**: PORTB Output Latch Value bits(1)

Note 1: Writes to PORTB are actually written to the corresponding LATB register. Reads from the PORTB register is return of actual I/O pin values.

REGISTER 11-18: LATE: PORTE DATA LATCH REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u
—	—	—	—	—	LATE2[(2)]	LATE1[(2)]	LATE0[(2)]
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 7-3 **Unimplemented:** Read as ‘0’

bit 2-0 **LATE<2:0>**: PORTE Output Latch Value bits[(1)]

Note 1: Writes to PORTE are actually written to the corresponding LATE register. Reads from the PORTE register is return of actual I/O pin values.

2: LATE<2:0> are not implemented on the PIC16LF1906. Read as ‘0’.

REGISTER 11-19: ANSELE: PORTE ANALOG SELECT REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-1/1	R/W-1/1	R/W-1/1
—	—	—	—	—	ANSE2[(2)]	ANSE1[(2)]	ANSE0[(2)]
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 7-3 **Unimplemented:** Read as ‘0’

bit 2-0 **ANSE<2:0>**: Analog Select between Analog or Digital Function on pins RE<2:0>, respectively
0 = Digital I/O. Pin is assigned to port or digital special function.
1 = Analog input. Pin is assigned as analog input[(1)]. Digital input buffer disabled.

Note 1: When setting a pin to an analog input, the corresponding TRIS bit must be set to Input mode in order to allow external control of the voltage on the pin.

2: ANSE<2:0> are not implemented on the PIC16LF1906. Read as ‘0’.

15.2.5 A/D CONVERSION PROCEDURE

This is an example procedure for using the ADC to perform an Analog-to-Digital conversion:

1. Configure Port:
   - Disable pin output driver (Refer to the TRIS register)
   - Configure pin as analog (Refer to the ANSEL register)
2. Configure the ADC module:
   - Select ADC conversion clock
   - Configure voltage reference
   - Select ADC input channel
   - Turn on ADC module
3. Configure ADC interrupt (optional):
   - Clear ADC interrupt flag
   - Enable ADC interrupt
   - Enable peripheral interrupt
   - Enable global interrupt**(1)**
4. Wait the required acquisition time**(2)**.
5. Start conversion by setting the GO/$\overline{\text{DONE}}$ bit.
6. Wait for ADC conversion to complete by one of the following:
   - Polling the GO/$\overline{\text{DONE}}$ bit
   - Waiting for the ADC interrupt (interrupts enabled)
7. Read ADC Result.
8. Clear the ADC interrupt flag (required if interrupt is enabled).

Note 1: The global interrupt can be disabled if the user is attempting to wake-up from Sleep and resume in-line code execution.

2: Refer to **Section 15.3 "A/D Acquisition Requirements"**.

EXAMPLE 15-1: A/D CONVERSION

```
;This code block configures the ADC
;for polling, Vdd and Vss references, Frc
;clock and AN0 input.
;
;Conversion start & polling for completion
; are included.
;
BANKSEL   ADCON1       ;
MOVLW     B'11110000'  ;Right justify, Frc
                       ;clock
MOVWF     ADCON1       ;Vdd and Vss Vref
BANKSEL   TRISA        ;
BSF       TRISA,0      ;Set RA0 to input
BANKSEL   ANSEL        ;
BSF       ANSEL,0      ;Set RA0 to analog
BANKSEL   ADCON0       ;
MOVLW     B'00000001'  ;Select channel AN0
MOVWF     ADCON0       ;Turn ADC On
CALL      SampleTime   ;Acquisiton delay
BSF       ADCON0,ADGO  ;Start conversion
BTFSC     ADCON0,ADGO  ;Is conversion done?
GOTO      $-1          ;No, test again
BANKSEL   ADRESH       ;
MOVF      ADRESH,W     ;Read upper 2 bits
MOVWF     RESULTHI     ;store in GPR space
BANKSEL   ADRESL       ;
MOVF      ADRESL,W     ;Read lower 8 bits
MOVWF     RESULTLO     ;Store in GPR space
```

18.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module is a serial I/O communications peripheral. It contains all the clock generators, shift registers and data buffers necessary to perform an input or output serial data transfer independent of device program execution. The EUSART, also known as a Serial Communications Interface (SCI), can be configured as a full-duplex asynchronous system or half-duplex synchronous system. Full-Duplex mode is useful for communications with peripheral systems, such as CRT terminals and personal computers. Half-Duplex Synchronous mode is intended for communications with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs or other microcontrollers. These devices typically do not have internal clocks for baud rate generation and require the external clock signal provided by a master synchronous device.

The EUSART module includes the following capabilities:

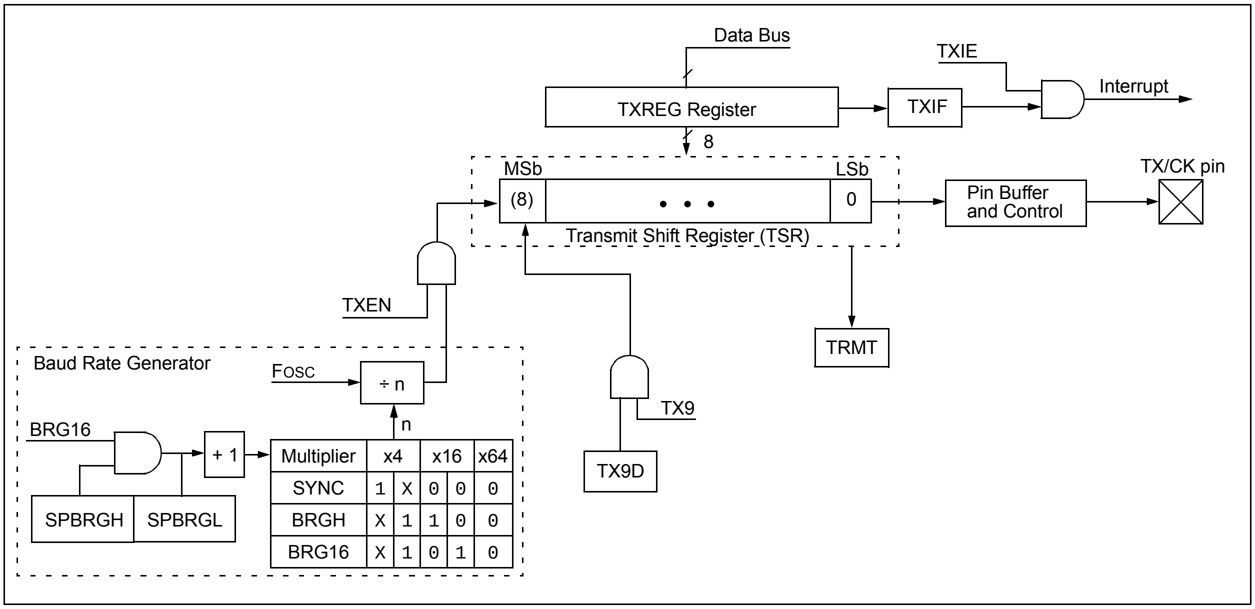
- Full-duplex asynchronous transmit and receive
- Two-character input buffer
- One-character output buffer
- Programmable 8-bit or 9-bit character length
- Address detection in 9-bit mode
- Input buffer overrun error detection
- Received character framing error detection
- Half-duplex synchronous master
- Half-duplex synchronous slave
- Programmable clock and data polarity

The EUSART module implements the following additional features, making it ideally suited for use in Local Interconnect Network (LIN) bus systems:

- Automatic detection and calibration of the baud rate
- Wake-up on Break reception
- 13-bit Break character transmit

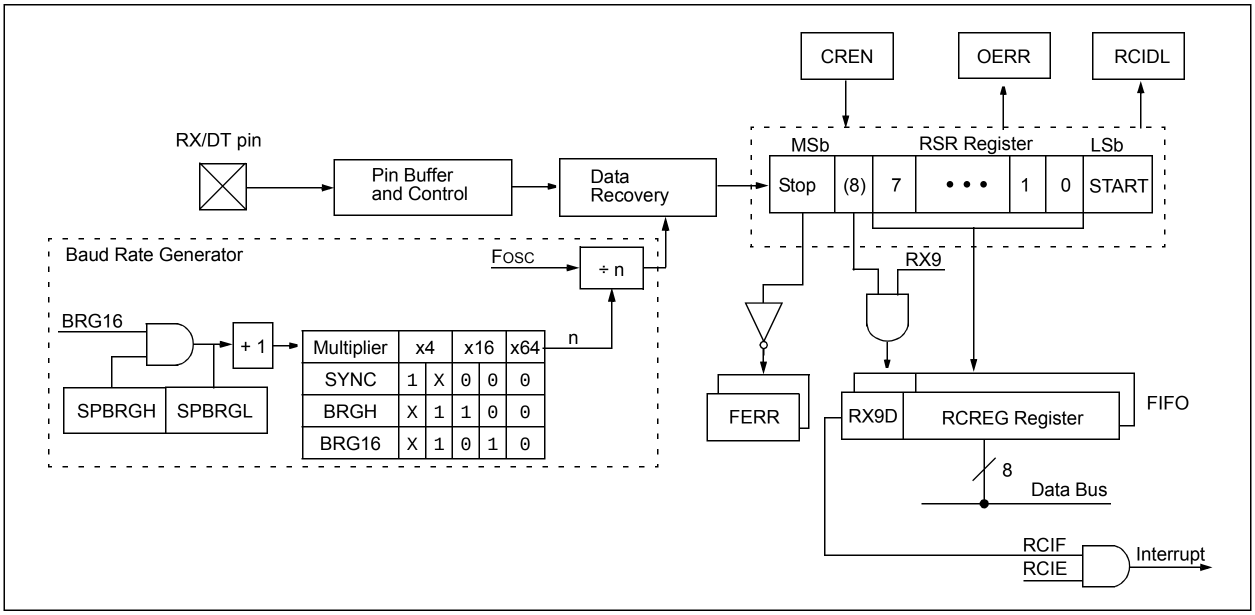
Block diagrams of the EUSART transmitter and receiver are shown in Figure 18-1 and Figure 18-2.

FIGURE 18-1: EUSART TRANSMIT BLOCK DIAGRAM



Multiplier	x4		x16		x64
SYNC	1	X	0	0	0
BRGH	X	1	1	0	0
BRG16	X	1	0	1	0

FIGURE 18-2: EUSART RECEIVE BLOCK DIAGRAM



Multiplier	x4		x16		x64
SYNC	1	X	0	0	0
BRGH	X	1	1	0	0
BRG16	X	1	0	1	0

The operation of the EUSART module is controlled through three registers:

- Transmit Status and Control (TXSTA)
- Receive Status and Control (RCSTA)
- Baud Rate Control (BAUDCON)

These registers are detailed in Register 18-1, Register 18-2 and Register 18-3, respectively.

For all modes of EUSART operation, the TRIS control bits corresponding to the RX/DT and TX/CK pins should be set to ‘1’. The EUSART control will automatically reconfigure the pin from input to output, as needed.

When the receiver or transmitter section is not enabled then the corresponding RX/DT or TX/CK pin may be used for general purpose input and output.

TABLE 18-2: REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
BAUD1CON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	153
BAUD2CON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	153
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	65
PIE1	TMR1GIE	ADIE	RCIE(1)	TXIE(1)	—	—	—	TMR1IE	66
PIR1	TMR1GIF	ADIF	RCIF(1)	TXIF(1)	—	—	—	TMR1IF	68
RCREG	EUSART Receive Register								147*
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	152
SPBRGL	EUSART Baud Rate Generator, Low Byte								154*
SPBRGH	EUSART Baud Rate Generator, High Byte								154*
TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	101
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	151

Legend: — = unimplemented locations, read as ‘0’. Shaded bits are not used for asynchronous reception.
* Page provides register information.

Note 1: PIC16LF1904/7 only.

REGISTER 19-1: LCDCON: LIQUID CRYSTAL DISPLAY (LCD) CONTROL REGISTER

R/W-0/0	R/W-0/0	R/C-0/0	U-0	R/W-0/0	R/W-0/0	R/W-1/1	R/W-1/1
LCDEN	SLPEN	WERR	—	CS<1:0>		LMUX<1:0>	
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 = Only clearable bit

bit 7 **LCDEN:** LCD Driver Enable bit
1 = LCD driver module is enabled
0 = LCD driver module is disabled

bit 6 **SLPEN:** LCD Driver Enable in Sleep Mode bit
1 = LCD driver module is disabled in Sleep mode
0 = LCD driver module is enabled in Sleep mode

bit 5 **WERR:** LCD Write Failed Error bit
1 = LCDDATAn register written while the WA bit of the LCDPS register = 0 (must be cleared in software)
0 = No LCD write error

bit 4 **Unimplemented:** Read as ‘0’

bit 3-2 **CS<1:0>:** Clock Source Select bits
00 = FOSC/256
01 = T1OSC (Timer1)
1x = LFINTOSC (31 kHz)

bit 1-0 **LMUX<1:0>:** Commons Select bits

LMUX<1:0>	Multiplex	Maximum Number of Pixels		Bias
		PIC16LF1906	PIC16LF1904/7	
00	Static (COM0)	19	29	Static
01	1/2 (COM<1:0>)	38	58	1/2 or 1/3
10	1/3 (COM<2:0>)	57	87	1/2 or 1/3
11	1/4 (COM<3:0>)	72[(1)]	116	1/3

Note 1: On these devices, COM3 and SEG15 are shared on one pin, limiting the device from driving 72 segments.

19.4 LCD Bias Internal Reference Ladder

The internal reference ladder can be used to divide the LCD bias voltage two or three equally spaced voltages that will be supplied to the LCD segment pins. To create this, the reference ladder consists of three matched resistors. Refer to Figure 19-3.

19.4.1 BIAS MODE INTERACTION

When in 1/2 Bias mode (BIASMD = 1), then the middle resistor of the ladder is shorted out so that only two voltages are generated. The current consumption of the ladder is higher in this mode, with the one resistor removed.

TABLE 19-3: LCD INTERNAL LADDER POWER MODES (1/3 BIAS)

Power Mode	Nominal Resistance of Entire Ladder	Nominal IDD
Low	3 Mohm	1 µA
Medium	300 kohm	10 µA
High	30 kohm	100 µA

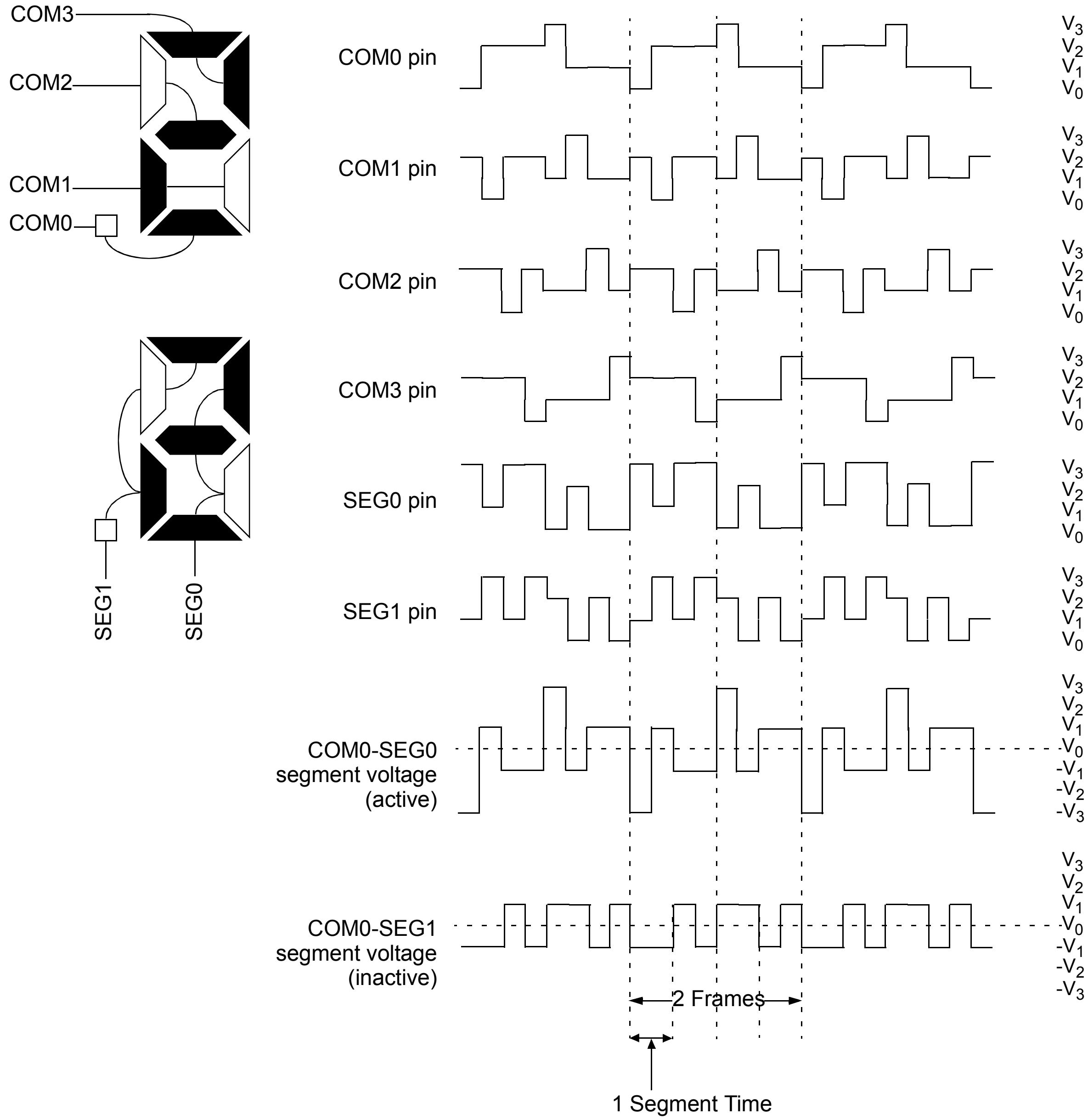
19.4.2 POWER MODES

The internal reference ladder may be operated in one of three power modes. This allows the user to trade off LCD contrast for power in the specific application. The larger the LCD glass, the more capacitance is present on a physical LCD segment, requiring more current to maintain the same contrast level.

Three different power modes are available, LP, MP and HP. The internal reference ladder can also be turned off for applications that wish to provide an external ladder or to minimize power consumption. Disabling the internal reference ladder results in all of the ladders being disconnected, allowing external voltages to be supplied.

Whenever the LCD module is inactive (LCDA = 0), the internal reference ladder will be turned off.

FIGURE 19-18: TYPE-B WAVEFORMS IN 1/4 MUX, 1/3 BIAS DRIVE



Note: 1 Frame = 2 single segment times.

TABLE 19-8: SUMMARY OF REGISTERS ASSOCIATED WITH LCD OPERATION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE	PEIE	TMR0IE	INTE	IOCIE	TMR0IF	INTF	IOCIF	65
LCDCON	LCDEN	SLPEN	WERR	—	CS1	CS0	LMUX<1:0>		173
LCDCST	—	—	—	—	—	LCDCST<2:0>			176
LCDDATA0	SEG7 COM0	SEG6 COM0	SEG5 COM0	SEG4 COM0	SEG3 COM0	SEG2 COM0	SEG1 COM0	SEG0 COM0	177
LCDDATA1	SEG15 COM0	SEG14 COM0	SEG13 COM0	SEG12 COM0	SEG11 COM0	SEG10 COM0	SEG9 COM0	SEG8 COM0	177
LCDDATA2(1)	SEG23 COM0	SEG22 COM0	SEG21 COM0	SEG20 COM0	SEG19 COM0	SEG18 COM0	SEG17 COM0	SEG16 COM0	177
LCDDATA3	SEG7 COM1	SEG6 COM1	SEG5 COM1	SEG4 COM1	SEG3 COM1	SEG2 COM1	SEG1 COM1	SEG0 COM1	177
LCDDATA4	SEG15 COM1	SEG14 COM1	SEG13 COM1	SEG12 COM1	SEG11 COM1	SEG10 COM1	SEG9 COM1	SEG8 COM1	177
LCDDATA5(1)	SEG23 COM1	SEG22 COM1	SEG21 COM1	SEG20 COM1	SEG19 COM1	SEG18 COM1	SEG17 COM1	SEG16 COM1	177
LCDDATA6	SEG7 COM2	SEG6 COM2	SEG5 COM2	SEG4 COM2	SEG3 COM2	SEG2 COM2	SEG1 COM2	SEG0 COM2	177
LCDDATA7	SEG15 COM2	SEG14 COM2	SEG13 COM2	SEG12 COM2	SEG11 COM2	SEG10 COM2	SEG9 COM2	SEG8 COM2	177
LCDDATA8(1)	SEG23 COM2	SEG22 COM2	SEG21 COM2	SEG20 COM2	SEG19 COM2	SEG18 COM2	SEG17 COM2	SEG16 COM2	177
LCDDATA9	SEG7 COM3	SEG6 COM3	SEG5 COM3	SEG4 COM3	SEG3 COM3	SEG2 COM3	SEG1 COM3	SEG0 COM3	177
LCDDATA10	SEG15 COM3	SEG14 COM3	SEG13 COM3	SEG12 COM3	SEG11 COM3	SEG10 COM3	SEG9 COM3	SEG8 COM3	177
LCDDATA11(1)	SEG23 COM3	SEG22 COM3	SEG20 COM3	SEG19 COM3	SEG18 COM3	SEG17 COM3	SEG16 COM3	SEG15 COM3	177
LCDDATA12	—	—	—	SEG28 COM0	SEG27 COM0	SEG26 COM0	SEG25 COM0	SEG24 COM0	177
LCDDATA15	—	—	—	SEG28 COM1	SEG27 COM1	SEG26 COM1	SEG25 COM1	SEG24 COM1	177
LCDDATA18	—	—	—	SEG28 COM2	SEG27 COM2	SEG26 COM2	SEG25 COM2	SEG24 COM2	177
LCDDATA21	—	—	—	SEG28 COM3	SEG27 COM3	SEG26 COM3	SEG25 COM3	SEG24 COM3	177
LCDPS	WFT	BIASMD	LCDA	WA	LP<3:0>				174
LCDREF	LCDIRE	—	LCDIRI	—	VLCD3PE	VLCD2PE	VLCD1PE	—	175
LCDRL	LRLAP<1:0>		LRLBP<1:0>		—	LRLAT<2:0>			184
LCDSE0	SE<7:0>								177
LCDSE1	SE<15:8>								177
LCDSE2	SE<23:16>								177
LCDSE3	—	—	—	SE<28:24>					177
PIE2	—	—	—	—	—	LCDIE	—	—	67
PIR2	—	—	—	—	—	LCDIF	—	—	69
T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	—	TMR1ON	139

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by the LCD module.

Note 1: PIC16LF1904/7 only.

MOVWI Move W to INDFn

Syntax: [*label*] MOVWI ++FSRn
[*label*] MOVWI --FSRn
[*label*] MOVWI FSRn++
[*label*] MOVWI FSRn--
[*label*] MOVWI k[FSRn]

Operands: n ∈ [0,1]
mm ∈ [00,01, 10, 11]
$-32 \leq k \leq 31$

Operation: W → INDFn
Effective address is determined by

- FSR + 1 (preincrement)
- FSR - 1 (predecrement)
- FSR + k (relative offset)

After the Move, the FSR value will be either:

- FSR + 1 (all increments)
- FSR - 1 (all decrements)

Unchanged

Status Affected: None

Mode	Syntax	mm
Preincrement	++FSRn	00
Predecrement	--FSRn	01
Postincrement	FSRn++	10
Postdecrement	FSRn--	11

Description: This instruction is used to move data between W and one of the indirect registers (INDFn). Before/after this move, the pointer (FSRn) is updated by pre/post incrementing/decrementing it.

Note: The INDFn registers are not physical registers. Any instruction that accesses an INDFn register actually accesses the register at the address specified by the FSRn.

FSRn is limited to the range 0000h - FFFFh. Incrementing/decrementing it beyond these bounds will cause it to wrap-around.

The increment/decrement operation on FSRn WILL NOT affect any Status bits.

NOP No Operation

Syntax: [*label*] NOP

Operands: None

Operation: No operation

Status Affected: None

Description: No operation.

Words: 1

Cycles: 1

<u>Example:</u> `NOP`

OPTION Load OPTION_REG Register with W

Syntax: [*label*] OPTION

Operands: None

Operation: (W) → OPTION_REG

Status Affected: None

Description: Move data from W register to OPTION_REG register.

Words: 1

Cycles: 1

<u>Example:</u> `OPTION`

Before Instruction
OPTION_REG = 0xFF
W = 0x4F
After Instruction
OPTION_REG = 0x4F
W = 0x4F

RESET Software Reset

Syntax: [*label*] RESET

Operands: None

Operation: Execute a device Reset. Resets the $\overline{\text{RI}}$ flag of the PCON register.

Status Affected: None

Description: This instruction provides a way to execute a hardware Reset by software.

RETFIE	Return from Interrupt
Syntax:	[*label*] RETFIE
Operands:	None
Operation:	TOS → PC, 1 → GIE
Status Affected:	None
Description:	Return from Interrupt. Stack is POPed and Top-of-Stack (TOS) is loaded in the PC. Interrupts are enabled by setting Global Interrupt Enable bit, GIE (INTCON<7>). This is a 2-cycle instruction.
Words:	1
Cycles:	2

Example:

```
RETFIE
```

After Interrupt

```
PC  =  TOS
GIE =  1
```

RETLW	Return with literal in W
Syntax:	[*label*] RETLW k
Operands:	0 ≤ k ≤ 255
Operation:	k → (W); TOS → PC
Status Affected:	None
Description:	The W register is loaded with the 8-bit literal ‘k’. The program counter is loaded from the top of the stack (the return address). This is a 2-cycle instruction.
Words:	1
Cycles:	2

Example:

```
        CALL TABLE;W contains table
              ;offset value
        •     ;W now has table value
TABLE   •
        •
        ADDWF PC  ;W = offset
        RETLW k1  ;Begin table
        RETLW k2  ;
        •
        •
        •
        RETLW kn  ; End of table
```

Before Instruction

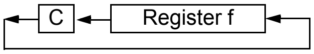
```
W  =  0x07
```

After Instruction

```
W  =  value of k8
```

RETURN	Return from Subroutine
Syntax:	[*label*] RETURN
Operands:	None
Operation:	TOS → PC
Status Affected:	None
Description:	Return from subroutine. The stack is POPed and the top of the stack (TOS) is loaded into the program counter. This is a 2-cycle instruction.

RLF	Rotate Left f through Carry
Syntax:	[*label*] RLF f,d
Operands:	0 ≤ f ≤ 127 d ∈ [0,1]
Operation:	See description below
Status Affected:	C
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 the W register. If ‘d’ is ‘1’, the result is stored back in register ‘f’.



Words:	1
Cycles:	1

Example:

```
RLF     REG1,0
```

Before Instruction

```
REG1   =   1110 0110
C      =   0
```

After Instruction

```
REG1   =   1110 0110
W      =   1100 1100
C      =   1
```

TABLE 22-3: POWER-DOWN CURRENTS (I_{PD})

PIC16LF1904/6/7		Standard Operating Conditions (unless otherwise stated)						
Param No.	Device Characteristics	Min.	Typ†	Max. +85°C	Max. +125°C	Units	Conditions	
							V_{DD}	Note
	Power-down Base Current (I_{PD})(2)							
D023		—	0.15	1.0	3.0	μA	1.8	WDT, BOR, FVR, and T1OSC disabled, all Peripherals Inactive
		—	0.16	2.0	4.0	μA	3.0	
		—	0.65	3.0	5.0	μA	3.6	
D024		—	0.27	2.0	4.0	μA	1.8	WDT Current **(Note 1)**
		—	0.56	3.0	5.0	μA	3.0	
		—	0.75	4.0	6.0	μA	3.6	
D025		—	17.5	31	35	μA	1.8	FVR current
		—	17.7	33	38	μA	3.0	
		—	17.8	35	41	μA	3.6	
D026		—	0.15	2.30	3.56	μA	3.0	LPBOR current
		—	0.21	3.40	4.70	μA	3.6	
D027		—	7.0	10	12	μA	3.0	BOR Current
		—	7.5	12	14	μA	3.6	
D028		—	0.50	2.0	4.0	μA	1.8	T1OSC Current
		—	0.60	3.0	5.0	μA	3.0	
		—	0.70	4.0	6.0	μA	3.6	
D029		—	0.40	2.0	4.0	μA	1.8	ADC Current **(Note 1, Note 3)**, no conversion in progress
		—	0.70	3.0	5.0	μA	3.0	
		—	0.90	4.0	6.0	μA	3.6	
D030		—	—	250	—	μA	1.8	ADC Current **(Note 1, Note 3)**, conversion in progress
		—	—	250	—	μA	3.0	
		—	—	250	—	μA	3.6	
D031	LCD Bias Ladder							
	Low power	—	1	2	6	μA	1.8	
	Medium Power	—	10	13	21	μA	3.0	
	High Power	—	100	111	120	μA	3.6	

* These parameters are characterized but not tested.

† Data in “Typ” column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: The peripheral current is the sum of the base I_{DD} or I_{PD} and the additional current consumed when this peripheral is enabled. The peripheral Δ current can be determined by subtracting the base I_{DD} or I_{PD} current from this limit. Max values should be used when calculating total current consumption.

2: 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 V_{DD}.

3: A/D oscillator source is F_{RC}.

TABLE 22-10: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET PARAMETERS

Standard Operating Conditions (unless otherwise stated)

Param No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
30	TMCL	$\overline{\text{MCLR}}$ Pulse Width (low)	2 5	— —	— —	μs μs	VDD = 3.0V, -40°C to +85°C VDD = 3.0V
31	FWDTLP	Low Frequency Internal Oscillator Frequency	19	33	52	kHz	
32	TOST	Oscillator Start-up Timer Period[(1)]	—	1024	—	Tosc	**(Note 2)**
33*	TPWRT	Power-up Timer Period, $\overline{\text{PWRTE}}$ = 0	—	2048	—	Tosc	Clocked by LFINTOSC
34*	TIOZ	I/O High-Impedance from $\overline{\text{MCLR}}$ Low or Watchdog Timer Reset	—	—	2.0	μs	
35	VBOR	Brown-out Reset Voltage: BORV = 0 BORV = 1	2.55 1.80	2.70 1.90	2.85 2.05	V V	
35A*	VHYST	Brown-out Reset Hysteresis	25 —	50 —	75 100	mV mV	-40°C to +85°C -40°C to +125°C
35B*	TBORDC	Brown-out Reset DC Response Time	1 —	3 —	5 10	μs μs	VDD ≤ VBOR, -40°C to +85°C VDD ≤ VBOR
35C	TBORAC	Brown-out Reset AC Response Time	—	100	—	ns	Transient Response immunity for a noise spike that goes from VDD to VSS and back with 10 ns rise and fall times. Guidance only.
36	TFVRS	Fixed Voltage Reference Turn-on Time	—	—	5	μs	Turn on to specified stability
37	VLPBOR	Low-Power Brown-out Reset Voltage	1.85	1.95	2.10	V	-40°C to +85°C
38*	VZPHYST	Zero-Power Brown-out Reset Hysteresis	0	25	60	mV	-40°C to +85°C
39*	TZPBPW	Zero-Power Brown-out Reset AC Response Time for BOR detection	10	—	500	nVs	VDD ≤ VBOR, -40°C to +85°C

* These parameters are characterized but not tested.

† Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at "min" values with an external clock applied to the OSC1 pin. When an external clock input is used, the "max" cycle time limit is "DC" (no clock) for all devices.

2: Period of the slower clock.

3: To ensure these voltage tolerances, VDD and VSS must be capacitively decoupled as close to the device as possible. 0.1 μF and 0.01 μF values in parallel are recommended.