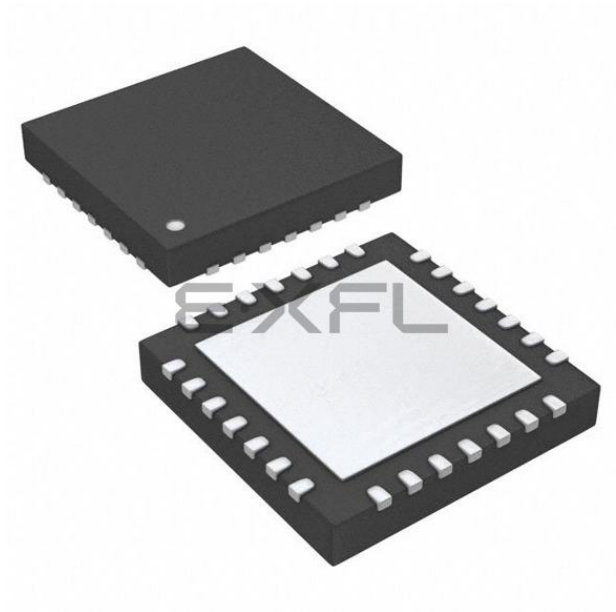


Welcome to E-XFL.COM

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	32MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	15
Program Memory Size	14KB (8K x 14)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 12x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-UFQFN Exposed Pad
Supplier Device Package	28-UQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16lf1847t-i-mv

PIN ALLOCATION TABLE

TABLE 1: 18/20/28-PIN SUMMARY (PIC16(L)F1847)

I/O	18-Pin PDIP/SOIC	20-Pin SSOP	28-Pin QFN/UQFN	ANSEL	ADC	Reference	Cap Sense	Comparator	SR Latch	Timers	CCP	EUSART	MSSP	Interrupt	Modulator	Pull-up	Basic
RA0	17	19	23	Y	AN0	—	CPS0	C12IN0-	—	—	—	—	SDO2	—	—	N	—
RA1	18	20	24	Y	AN1	—	CPS1	C12IN1-	—	—	—	—	$\overline{SS2}$	—	—	N	—
RA2	1	1	26	Y	AN2	VREF- DACOUT	CPS2	C12IN2- C12IN+	—	—	—	—	—	—	—	N	—
RA3	2	2	27	Y	AN3	VREF+	CPS3	C12IN3- C1IN+ C1OUT	SRQ	—	CCP3	—	—	—	—	N	—
RA4	3	3	28	Y	AN4	—	CPS4	C2OUT	SRNQ	T0CKI	CCP4	—	—	—	—	N	—
RA5	4	4	1	N	—	—	—	—	—	—	—	—	$\overline{SS1}^{(1)}$	—	—	Y(2)	$\overline{MCLR}$ VPP
RA6	15	17	20	N	—	—	—	—	—	—	P1D(1) P2B(1)	—	SDO1(1)	—	—	N	OSC2 CLKOUT CLKR
RA7	16	18	21	N	—	—	—	—	—	—	P1C(1) CCP2(1) P2A(1)	—	—	—	—	N	OSC1 CLKIN
RB0	6	7	7	N	—	—	—	—	SRI	T1G	CCP1(1) P1A(1) FLT0	—	—	INT IOC	—	Y	—
RB1	7	8	8	Y	AN11	—	CPS11	—	—	—	—	RX(1,3) DT(1,3)	SDA1 SDI1	IOC	—	Y	—
RB2	8	9	9	Y	AN10	—	CPS10	—	—	—	—	RX(1) DT(1) TX(1,3) CK(1,3)	SDA2 SDI2 SDO1(1,3)	IOC	MDMIN	Y	—
RB3	9	10	10	Y	AN9	—	CPS9	—	—	—	CCP1(1,3) P1A(1,3)	—	—	IOC	MDOUT	Y	—
RB4	10	11	12	Y	AN8	—	CPS8	—	—	—	—	—	SCL1 SCK1	IOC	MDCIN2	Y	—
RB5	11	12	13	Y	AN7	—	CPS7	—	—	—	P1B	TX(1) CK(1)	SCL2 SCK2 $\overline{SS1}^{(1,3)}$	IOC	—	Y	—
RB6	12	13	15	Y	AN5	—	CPS5	—	—	T1CKI T1OSCI	P1C(1,3) CCP2(1,3) P2A(1,3)	—	—	IOC	—	Y	ICSPCLK
RB7	13	14	16	Y	AN6	—	CPS6	—	—	T1OSCO	P1D(1,3) P2B(1,3)	—	—	IOC	MDCIN1	Y	ICSPDAT
VDD	14	15, 16	17, 19	—	—	—	—	—	—	—	—	—	—	—	—	—	VDD
Vss	5	5,6	3,5	—	—	—	—	—	—	—	—	—	—	—	—	—	Vss

Note 1: Pin functions can be moved using the APFCON register(s).
2: Weak pull-up always enabled when MCLR is enabled, otherwise the pull-up is under user control.
3: Default function location.

NOTES:

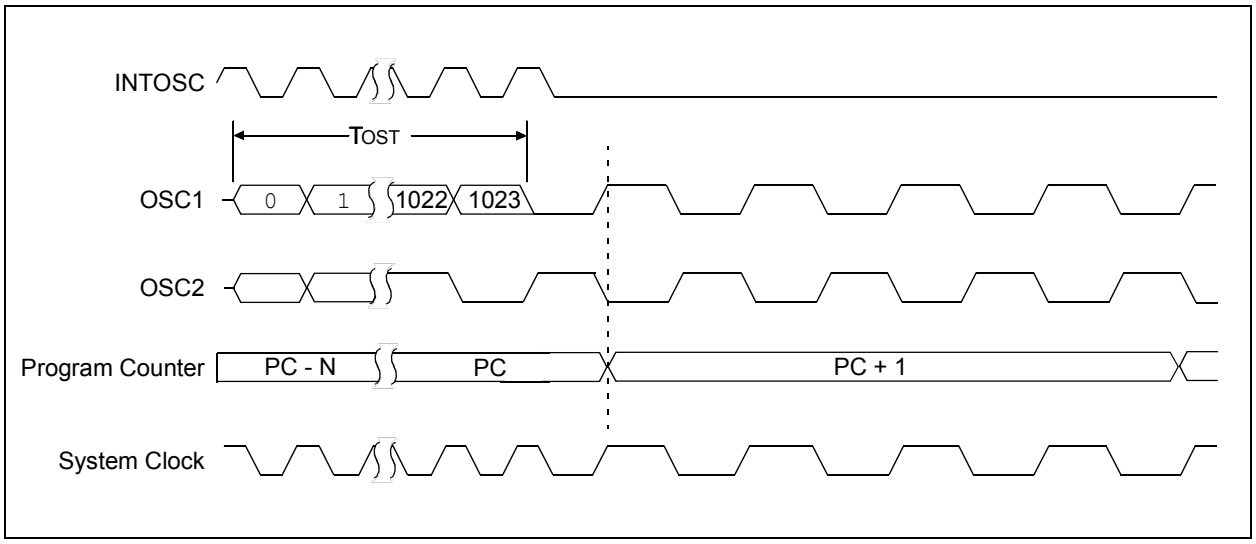
5.4.2 TWO-SPEED START-UP SEQUENCE

1. Wake-up from Power-on Reset or Sleep.
2. Instructions begin execution by the internal oscillator at the frequency set in the IRCF<3:0> bits of the OSCCON register.
3. OST enabled to count 1024 clock cycles.
4. OST timed out, wait for falling edge of the internal oscillator.
5. OSTS is set.
6. System clock held low until the next falling edge of new clock (LP, XT or HS mode).
7. System clock is switched to external clock source.

5.4.3 CHECKING TWO-SPEED CLOCK STATUS

Checking the state of the OSTS bit of the OSCSTAT register will confirm if the microcontroller is running from the external clock source, as defined by the FOSC<2:0> bits in the Configuration Words, or the internal oscillator.

FIGURE 5-8: TWO-SPEED START-UP



6.5 Register Definitions: Reference Clock Control

REGISTER 6-1: CLKRCON: REFERENCE CLOCK CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-1/1	R/W-1/1	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
CLKREN	CLKROE	CLKRSLR	CLKRDC<1:0>		CLKRDIV<2:0>		
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	

bit 7 **CLKREN:** Reference Clock Module Enable bit
- 1 = Reference clock module is enabled
- 0 = Reference clock module is disabled

bit 6 **CLKROE:** Reference Clock Output Enable bit(3)
- 1 = Reference Clock output is enabled on CLKR pin
- 0 = Reference Clock output disabled on CLKR pin

bit 5 **CLKRSLR:** Reference Clock Slew Rate Control Limiting Enable bit
- 1 = Slew Rate limiting is enabled
- 0 = Slew Rate limiting is disabled

bit 4-3 **CLKRDC<1:0>:** Reference Clock Duty Cycle bits
- 11 = Clock outputs duty cycle of 75%
- 10 = Clock outputs duty cycle of 50%
- 01 = Clock outputs duty cycle of 25%
- 00 = Clock outputs duty cycle of 0%

bit 2-0 **CLKRDIV<2:0>** Reference Clock Divider bits
- 111 = Base clock value divided by 128
- 110 = Base clock value divided by 64
- 101 = Base clock value divided by 32
- 100 = Base clock value divided by 16
- 011 = Base clock value divided by 8
- 010 = Base clock value divided by 4
- 001 = Base clock value divided by 2(1)
- 000 = Base clock value(2)

Note 1: In this mode, the 25% and 75% duty cycle accuracy will be dependent on the source clock duty cycle.

2: In this mode, the duty cycle will always be equal to the source clock duty cycle, unless a duty cycle of 0% is selected.

3: To route CLKR to pin, $\overline{\text{CLKOUTEN}}$ of Configuration Words = 1 is required. $\overline{\text{CLKOUTEN}}$ of Configuration Words = 0 will result in FOSC/4. See **Section 6.3 "Conflicts with the CLKR pin"** for details.

EXAMPLE 11-3: FLASH PROGRAM MEMORY READ

```
* This code block will read 1 word of program
* memory at the memory address:
    PROG_ADDR_HI : PROG_ADDR_LO
*   data will be returned in the variables;
*   PROG_DATA_HI, PROG_DATA_LO

    BANKSEL  EEADRL           ; Select Bank for EEPROM registers
    MOVLW    PROG_ADDR_LO     ;
    MOVWF    EEADRL           ; Store LSB of address
    MOVLW    PROG_ADDR_HI     ;
    MOVWL    EEADRH           ; Store MSB of address

    BCF      EECON1,CFGS      ; Do not select Configuration Space
    BSF      EECON1,EEPGD     ; Select Program Memory
    BCF      INTCON,GIE       ; Disable interrupts
    BSF      EECON1,RD        ; Initiate read
    NOP                       ; Executed (Figure 11-1)
    NOP                       ; Ignored (Figure 11-1)
    BSF      INTCON,GIE       ; Restore interrupts

    MOVF     EEDATL,W         ; Get LSB of word
    MOVWF    PROG_DATA_LO     ; Store in user location
    MOVF     EEDATH,W         ; Get MSB of word
    MOVWF    PROG_DATA_HI     ; Store in user location
```

19.3 Comparator Hysteresis

A selectable amount of separation voltage can be added to the input pins of each comparator to provide a hysteresis function to the overall operation. Hysteresis is enabled by setting the CxHYS bit of the CMxCON0 register.

See **Section 30.0 "Electrical Specifications"** for more information.

19.4 Timer1 Gate Operation

The output resulting from a comparator operation can be used as a source for gate control of Timer1. See **Section 21.6 "Timer1 Gate"** for more information. This feature is useful for timing the duration or interval of an analog event.

It is recommended that the comparator output be synchronized to Timer1. This ensures that Timer1 does not increment while a change in the comparator is occurring.

19.4.1 COMPARATOR OUTPUT SYNCHRONIZATION

The output from either comparator, C1 or C2, can be synchronized with Timer1 by setting the CxSYNC bit of the CMxCON0 register.

Once enabled, the comparator output is latched on the falling edge of the Timer1 source clock. If a prescaler is used with Timer1, the comparator output is latched after the prescaling function. To prevent a race condition, the comparator output is latched on the falling edge of the Timer1 clock source and Timer1 increments on the rising edge of its clock source. See the Comparator Block Diagram (Figure 19-2) and the Timer1 Block Diagram (Figure 21-1) for more information.

19.5 Comparator Interrupt

An interrupt can be generated upon a change in the output value of the comparator for each comparator, a rising edge detector and a Falling edge detector are present.

When either edge detector is triggered and its associated enable bit is set (CxINTP and/or CxINTN bits of the CMxCON1 register), the Corresponding Interrupt Flag bit (CxIF bit of the PIR2 register) will be set.

To enable the interrupt, you must set the following bits:

- CxON, CxPOL and CxSP bits of the CMxCON0 register
- CxIE bit of the PIE2 register
- CxINTP bit of the CMxCON1 register (for a rising edge detection)
- CxINTN bit of the CMxCON1 register (for a falling edge detection)
- PEIE and GIE bits of the INTCON register

The associated interrupt flag bit, CxIF bit of the PIR2 register, must be cleared in software. If another edge is detected while this flag is being cleared, the flag will still be set at the end of the sequence.

Note:	Although a comparator is disabled, an interrupt can be generated by changing the output polarity with the CxPOL bit of the CMxCON0 register, or by switching the comparator on or off with the CxON bit of the CMxCON0 register.

19.6 Comparator Positive Input Selection

Configuring the CxPCH<1:0> bits of the CMxCON1 register directs an internal voltage reference or an analog pin to the non-inverting input of the comparator:

- C1IN+ or C12IN+ analog pin
- DAC
- FVR (Fixed Voltage Reference)
- VSS (Ground)

See **Section TABLE 14-1: "Summary of Registers Associated with the Fixed Voltage Reference"** for more information on the Fixed Voltage Reference module.

See **Section 17.0 "Digital-to-Analog Converter (DAC) Module"** for more information on the DAC input signal.

Any time the comparator is disabled (CxON = 0), all comparator inputs are disabled.

REGISTER 24-5: PSTRxCON: PWM STEERING CONTROL REGISTER(1)

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-1/1
—	—	—	STRxSYNC	STRxD	STRxC	STRxB	STRxA
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 7-5 **Unimplemented:** Read as '0'

bit 4 **STRxSYNC:** Steering Sync bit
1 = Output steering update occurs on next PWM period
0 = Output steering update occurs at the beginning of the instruction cycle boundary

bit 3 **STRxD:** Steering Enable bit D
1 = PxD pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxD pin is assigned to port pin

bit 2 **STRxC:** Steering Enable bit C
1 = PxC pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxC pin is assigned to port pin

bit 1 **STRxB:** Steering Enable bit B
1 = PxB pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxB pin is assigned to port pin

bit 0 **STRxA:** Steering Enable bit A
1 = PxA pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxA pin is assigned to port pin

Note 1: The PWM Steering mode is available only when the CCPxCON register bits CCPxM<3:2> = 11 and PxM<1:0> = 00.

FIGURE 25-3: MSSPx BLOCK DIAGRAM (I^2C™ SLAVE MODE)

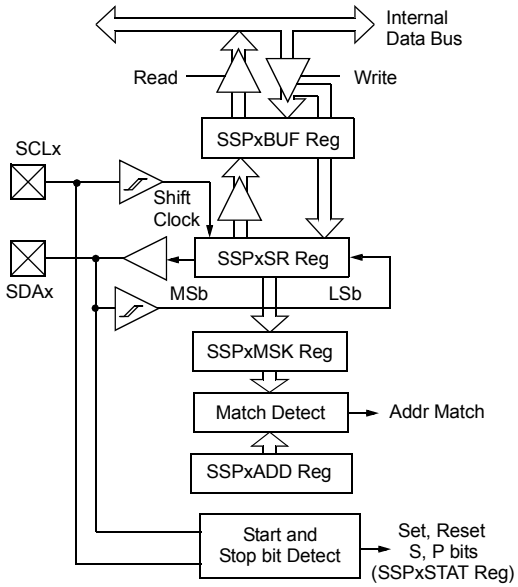


FIGURE 25-18: I²C SLAVE, 7-BIT ADDRESS, TRANSMISSION (AHEN = 0)

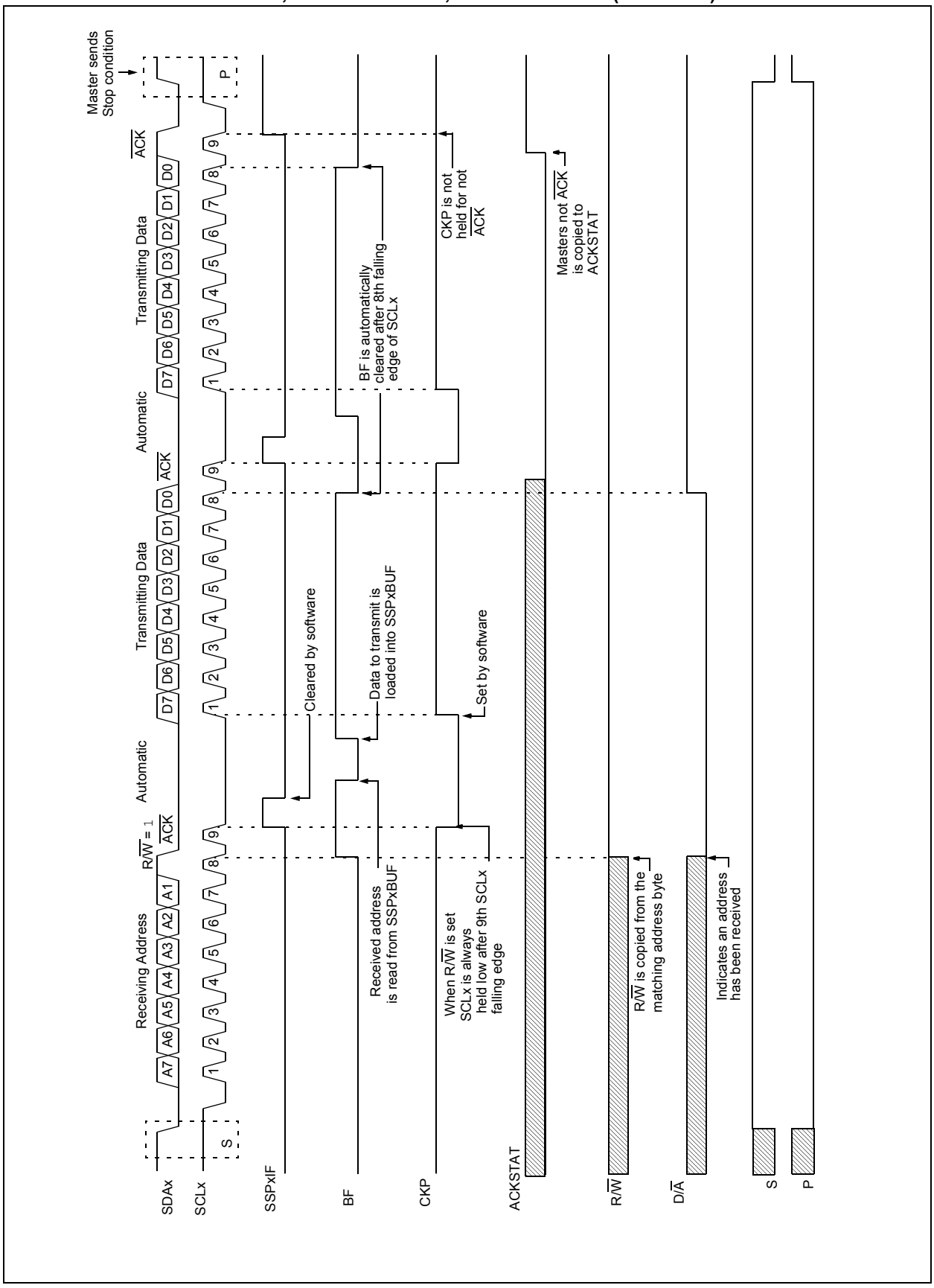


TABLE 26-2: SUMMARY OF REGISTERS ASSOCIATED WITH ASYNCHRONOUS RECEPTION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
APFCON0	RXDTSEL	SDO1SEL	SS1SEL	P2BSEL	CCP2SEL	P1DSEL	P1CSEL	CCP1SEL	118
APFCON1	—	—	—	—	—	—	—	TXCKSEL	118
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	298
INTCON	GIE	PEIE	TMR0IE	INTE	IOCE	TMR0IF	INTF	IOCF	88
PIE1	TMR1GIE	ADIE	RCIE	TXIE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	89
PIR1	TMR1GIF	ADIF	RCIF	TXIF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	93
RCREG	EUSART Receive Data Register								292*
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	297
SPBRGL	BRG<7:0>								299*
SPBRGH	BRG<15:8>								299*
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	126
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	296

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for Asynchronous Reception.

* Page provides register information.

26.3.2 AUTO-BAUD OVERFLOW

During the course of automatic baud detection, the ABDOVF bit of the BAUDCON register will be set if the baud rate counter overflows before the fifth rising edge is detected on the RX pin. The ABDOVF bit indicates that the counter has exceeded the maximum count that can fit in the 16 bits of the SPBRGH:SPBRGL register pair. After the ABDOVF has been set, the counter continues to count until the fifth rising edge is detected on the RX pin. Upon detecting the fifth RX edge, the hardware will set the RCIF interrupt flag and clear the ABDEN bit of the BAUDCON register. The RCIF flag can be subsequently cleared by reading the RCREG register. The ABDOVF flag of the BAUDCON register can be cleared by software directly.

To terminate the auto-baud process before the RCIF flag is set, clear the ABDEN bit then clear the ABDOVF bit of the BAUDCON register. The ABDOVF bit will remain set if the ABDEN bit is not cleared first.

26.3.3 AUTO-WAKE-UP ON BREAK

During Sleep mode, all clocks to the EUSART are suspended. Because of this, the Baud Rate Generator is inactive and a proper character reception cannot be performed. The Auto-Wake-up feature allows the controller to wake-up due to activity on the RX/DT line. This feature is available only in Asynchronous mode.

The Auto-Wake-up feature is enabled by setting the WUE bit of the BAUDCON register. Once set, the normal receive sequence on RX/DT is disabled, and the EUSART remains in an Idle state, monitoring for a wake-up event independent of the CPU mode. A wake-up event consists of a high-to-low transition on the RX/DT line. (This coincides with the start of a Sync Break or a wake-up signal character for the LIN protocol.)

The EUSART module generates an RCIF interrupt coincident with the wake-up event. The interrupt is generated synchronously to the Q clocks in normal CPU operating modes (Figure 26-7), and asynchronously if the device is in Sleep mode (Figure 26-8). The interrupt condition is cleared by reading the RCREG register.

The WUE bit is automatically cleared by the low-to-high transition on the RX line at the end of the Break. This signals to the user that the Break event is over. At this point, the EUSART module is in Idle mode waiting to receive the next character.

26.3.3.1 Special Considerations

Break Character

To avoid character errors or character fragments during a wake-up event, the wake-up character must be all zeros.

When the wake-up is enabled the function works independent of the low time on the data stream. If the WUE bit is set and a valid non-zero character is received, the low time from the Start bit to the first rising edge will be interpreted as the wake-up event. The remaining bits in the character will be received as a fragmented character and subsequent characters can result in framing or overrun errors.

Therefore, the initial character in the transmission must be all '0's. This must be ten or more bit times, 13-bit times recommended for LIN bus, or any number of bit times for standard RS-232 devices.

Oscillator Start-up Time

Oscillator start-up time must be considered, especially in applications using oscillators with longer start-up intervals (i.e., LP, XT or HS/PLL mode). The Sync Break (or wake-up signal) character must be of sufficient length, and be followed by a sufficient interval, to allow enough time for the selected oscillator to start and provide proper initialization of the EUSART.

WUE Bit

The wake-up event causes a receive interrupt by setting the RCIF bit. The WUE bit is cleared in hardware by a rising edge on RX/DT. The interrupt condition is then cleared in software by reading the RCREG register and discarding its contents.

To ensure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process before setting the WUE bit. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

26.4.2 SYNCHRONOUS SLAVE MODE

The following bits are used to configure the EUSART for Synchronous slave operation:

- SYNC = 1
- CSRC = 0
- SREN = 0 (for transmit); SREN = 1 (for receive)
- CREN = 0 (for transmit); CREN = 1 (for receive)
- SPEN = 1

Setting the SYNC bit of the TXSTA register configures the device for synchronous operation. Clearing the CSRC bit of the TXSTA register configures the device as a slave. Clearing the SREN and CREN bits of the RCSTA register ensures that the device is in the Transmit mode, otherwise the device will be configured to receive. Setting the SPEN bit of the RCSTA register enables the EUSART.

26.4.2.1 EUSART Synchronous Slave Transmit

The operation of the Synchronous Master and Slave modes are identical (see **Section 26.4.1.3 "Synchronous Master Transmission"**), except in the case of the Sleep mode.

If two words are written to the TXREG and then the SLEEP instruction is executed, the following will occur:

1. The first character will immediately transfer to the TSR register and transmit.
2. The second word will remain in TXREG register.
3. The TXIF bit will not be set.
4. After the first character has been shifted out of TSR, the TXREG register will transfer the second character to the TSR and the TXIF bit will now be set.
5. If the PEIE and TXIE bits are set, the interrupt will wake the device from Sleep and execute the next instruction. If the GIE bit is also set, the program will call the Interrupt Service Routine.

26.4.2.2 Synchronous Slave Transmission Setup:

1. Set the SYNC and SPEN bits and clear the CSRC bit.
2. Clear the ANSEL bit for the CK pin (if applicable).
3. Clear the CREN and SREN bits.
4. If interrupts are desired, set the TXIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
5. If 9-bit transmission is desired, set the TX9 bit.
6. Enable transmission by setting the TXEN bit.
7. If 9-bit transmission is selected, insert the Most Significant bit into the TX9D bit.
8. Start transmission by writing the Least Significant eight bits to the TXREG register.

TABLE 26-9: SUMMARY OF REGISTERS ASSOCIATED WITH SYNCHRONOUS SLAVE TRANSMISSION

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
APFCON0	RXDTSEL	SDO1SEL	SS1SEL	P2BSEL	CCP2SEL	P1DSEL	P1CSEL	CCP1SEL	118
APFCON1	—	—	—	—	—	—	—	TXCKSEL	118
BAUDCON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	298
INTCON	GIE	PEIE	TMR0IE	INTE	IOCE	TMR0IF	INTF	IOCF	88
PIE1	TMR1GIE	ADIE	RCIE	TXIE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	89
PIR1	TMR1GIF	ADIF	RCIF	TXIF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	93
RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	297
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	126
TXREG	EUSART Transmit Data Register								289*
TXSTA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	296

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for Synchronous Slave Transmission.

* Page provides register information.

26.5 EUSART Operation During Sleep

The EUSART will remain active during Sleep only in the Synchronous Slave mode. All other modes require the system clock and therefore cannot generate the necessary signals to run the Transmit or Receive Shift registers during Sleep.

Synchronous Slave mode uses an externally generated clock to run the Transmit and Receive Shift registers.

26.5.1 SYNCHRONOUS RECEIVE DURING SLEEP

To receive during Sleep, all the following conditions must be met before entering Sleep mode:

- RCSTA and TXSTA Control registers must be configured for Synchronous Slave Reception (see **Section 26.4.2.4 "Synchronous Slave Reception Setup:"**).
- If interrupts are desired, set the RCIE bit of the PIE1 register and the GIE and PEIE bits of the INTCON register.
- The RCIF interrupt flag must be cleared by reading RCREG to unload any pending characters in the receive buffer.

Upon entering Sleep mode, the device will be ready to accept data and clocks on the RX/DT and TX/CK pins, respectively. When the data word has been completely clocked in by the external device, the RCIF interrupt flag bit of the PIR1 register will be set. Thereby, waking the processor from Sleep.

Upon waking from Sleep, the instruction following the SLEEP instruction will be executed. If the Global Interrupt Enable (GIE) bit of the INTCON register is also set, then the Interrupt Service Routine at address 004h will be called.

26.5.2 SYNCHRONOUS TRANSMIT DURING SLEEP

To transmit during Sleep, all the following conditions must be met before entering Sleep mode:

- RCSTA and TXSTA Control registers must be configured for Synchronous Slave Transmission (see **Section 26.4.2.2 "Synchronous Slave Transmission Setup:"**).
- The TXIF interrupt flag must be cleared by writing the output data to the TXREG, thereby filling the TSR and transmit buffer.
- If interrupts are desired, set the TXIE bit of the PIE1 register and the PEIE bit of the INTCON register.
- Interrupt enable bits TXIE of the PIE1 register and PEIE of the INTCON register must set.

Upon entering Sleep mode, the device will be ready to accept clocks on TX/CK pin and transmit data on the RX/DT pin. When the data word in the TSR has been completely clocked out by the external device, the pending byte in the TXREG will transfer to the TSR and the TXIF flag will be set. Thereby, waking the processor from Sleep. At this point, the TXREG is available to accept another character for transmission, which will clear the TXIF flag.

Upon waking from Sleep, the instruction following the SLEEP instruction will be executed. If the Global Interrupt Enable (GIE) bit is also set then the Interrupt Service Routine at address 0004h will be called.

26.5.3 ALTERNATE PIN LOCATIONS

This module incorporates I/O pins that can be moved to other locations with the use of the alternate pin function registers, APFCON0 and APFCON1. To determine which pins can be moved and what their default locations are upon a Reset, see **Section 12.1 "Alternate Pin Function"** for more information.

TABLE 29-3: *DEVICE(S)* ENHANCED INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb	LSb	Status Affected	Notes
CONTROL OPERATIONS							
BRA	k	Relative Branch	2	11 001k kkkk kkkk			
BRW	–	Relative Branch with W	2	00 0000 0000 1011			
CALL	k	Call Subroutine	2	10 0kkk kkkk kkkk			
CALLW	–	Call Subroutine with W	2	00 0000 0000 1010			
GOTO	k	Go to address	2	10 1kkk kkkk kkkk			
RETFIE	k	Return from interrupt	2	00 0000 0000 1001			
RETLW	k	Return with literal in W	2	11 0100 kkkk kkkk			
RETURN	–	Return from Subroutine	2	00 0000 0000 1000			
INHERENT OPERATIONS							
CLRWDT	–	Clear Watchdog Timer	1	00 0000 0110 0100		$\overline{TO}$, $\overline{PD}$	
NOP	–	No Operation	1	00 0000 0000 0000			
OPTION	–	Load OPTION_REG register with W	1	00 0000 0110 0010			
RESET	–	Software device Reset	1	00 0000 0000 0001			
SLEEP	–	Go into Standby mode	1	00 0000 0110 0011		$\overline{TO}$, $\overline{PD}$	
TRIS	f	Load TRIS register with W	1	00 0000 0110 0fff			
C-COMPILER OPTIMIZED							
ADDFSR	n, k	Add Literal k to FSRn	1	11 0001 0nkk kkkk			
MOVIW	n mm	Move Indirect FSRn to W with pre/post inc/dec modifier, mm	1	00 0000 0001 0nmm		Z	**2, 3**
	k[n]	Move INDFn to W, Indexed Indirect.	1	11 1111 0nkk kkkk		Z	**2**
MOVWI	n mm	Move W to Indirect FSRn with pre/post inc/dec modifier, mm	1	00 0000 0001 1nmm			**2, 3**
	k[n]	Move W to INDFn, Indexed Indirect.	1	11 1111 1nkk kkkk			**2**

Note 1: If the Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

2: If this instruction addresses an INDF register and the MSb of the corresponding FSR is set, this instruction will require one additional instruction cycle.

3: See Table in the MOVIW and MOVWI instruction descriptions.

CALL **Call Subroutine**

Syntax:	[*label*] CALL k
Operands:	$0 \le k \le 2047$
Operation:	(PC)+ 1→ TOS, k → PC<10:0>, (PCLATH<6:3>) → PC<14:11>
Status Affected:	None
Description:	Call Subroutine. First, return address (PC + 1) is pushed onto the stack. The 11-bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a 2-cycle instruction.

CALLW **Subroutine Call With W**

Syntax:	[*label*] CALLW
Operands:	None
Operation:	(PC) +1 → TOS, (W) → PC<7:0>, (PCLATH<6:0>) → PC<14:8>
Status Affected:	None
Description:	Subroutine call with W. First, the return address (PC + 1) is pushed onto the return stack. Then, the contents of W is loaded into PC<7:0>, and the contents of PCLATH into PC<14:8>. CALLW is a 2-cycle instruction.

CLRF **Clear f**

Syntax:	[*label*] CLRF f
Operands:	$0 \le f \le 127$
Operation:	00h → (f) 1 → Z
Status Affected:	Z
Description:	The contents of register ‘f’ are cleared and the Z bit is set.

CLRW **Clear W**

Syntax:	[*label*] CLRW
Operands:	None
Operation:	00h → (W) 1 → Z
Status Affected:	Z
Description:	W register is cleared. Zero bit (Z) is set.

CLRWDT **Clear Watchdog Timer**

Syntax:	[*label*] CLRWDT
Operands:	None
Operation:	00h → WDT 0 → WDT prescaler, 1 → $\overline{TO}$ 1 → $\overline{PD}$
Status Affected:	$\overline{TO}$, $\overline{PD}$
Description:	CLRWDT instruction resets the Watchdog Timer. It also resets the prescaler of the WDT. Status bits $\overline{TO}$ and $\overline{PD}$ are set.

COMF **Complement f**

Syntax:	[*label*] COMF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	$(\bar{f}) \rightarrow$ (destination)
Status Affected:	Z
Description:	The contents of register ‘f’ are complemented. If ‘d’ is ‘0’, the result is stored in W. If ‘d’ is ‘1’, the result is stored back in register ‘f’.

DECF **Decrement f**

Syntax:	[*label*] DECF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f) - 1 → (destination)
Status Affected:	Z
Description:	Decrement register ‘f’. If ‘d’ is ‘0’, the result is stored in the W register. If ‘d’ is ‘1’, the result is stored back in register ‘f’.

RETFIE **Return from Interrupt**

Syntax: [*label*] RETFIE k

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**

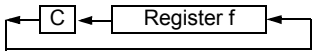
Syntax: [*label*] RLF f,d

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

SWAPF **Swap Nibbles in f**

Syntax:	[*label*] SWAPF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(f<3:0>) → (destination<7:4>), (f<7:4>) → (destination<3:0>)
Status Affected:	None
Description:	The upper and lower nibbles of register 'f' are exchanged. If 'd' is '0', the result is placed in the W register. If 'd' is '1', the result is placed in register 'f'.

TRIS **Load TRIS Register with W**

Syntax:	[*label*] TRIS f
Operands:	$5 \le f \le 7$
Operation:	(W) → TRIS register 'f'
Status Affected:	None
Description:	Move data from W register to TRIS register. When 'f' = 5, TRISA is loaded. When 'f' = 6, TRISB is loaded. When 'f' = 7, TRISC is loaded.

XORLW **Exclusive OR literal with W**

Syntax:	[*label*] XORLW k
Operands:	$0 \le k \le 255$
Operation:	(W) .XOR. k → (W)
Status Affected:	Z
Description:	The contents of the W register are XOR'ed with the 8-bit literal 'k'. The result is placed in the W register.

XORWF **Exclusive OR W with f**

Syntax:	[*label*] XORWF f,d
Operands:	$0 \le f \le 127$ $d \in [0,1]$
Operation:	(W) .XOR. (f) → (destination)
Status Affected:	Z
Description:	Exclusive OR the contents of the W register with register 'f'. If 'd' is '0', the result is stored in the W register. If 'd' is '1', the result is stored back in register 'f'.

FIGURE 30-10: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

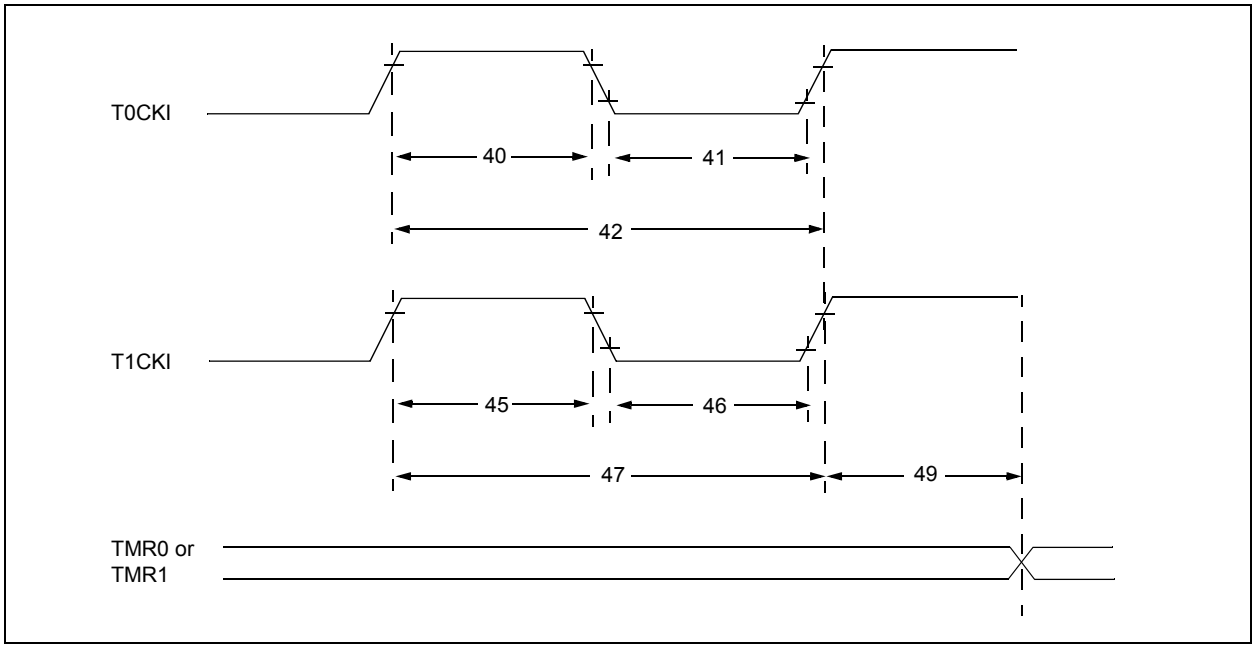


TABLE 30-12: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Standard Operating Conditions (unless otherwise stated)								
Param. No.	**Sym.**	**Characteristic**		**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
40*	TT0H	T0CKI High Pulse Width	No Prescaler	0.5 TCY + 20	—	—	ns	
			With Prescaler	10	—	—	ns	
41*	TT0L	T0CKI Low Pulse Width	No Prescaler	0.5 TCY + 20	—	—	ns	
			With Prescaler	10	—	—	ns	
42*	TT0P	T0CKI Period		Greater of: 20 or TCY + 40 / N	—	—	ns	N = prescale value
45*	TT1H	T1CKI High Time	Synchronous, No Prescaler	0.5 TCY + 20	—	—	ns	
			Synchronous, with Prescaler	15	—	—	ns	
			Asynchronous	30	—	—	ns	
46*	TT1L	T1CKI Low Time	Synchronous, No Prescaler	0.5 TCY + 20	—	—	ns	
			Synchronous, with Prescaler	15	—	—	ns	
			Asynchronous	30	—	—	ns	
47*	TT1P	T1CKI Input Period	Synchronous	Greater of: 30 or TCY + 40 / N	—	—	ns	N = prescale value
			Asynchronous	60	—	—	ns	
48	FT1	Secondary Oscillator Input Frequency Range (Oscillator enabled by setting bit T1OSCEN)		32.4	32.768	33.1	kHz	
49*	TCKEZTMR1	Delay from External Clock Edge to Timer Increment		2 TOSC	—	7 TOSC	—	Timers in Sync mode

* 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.

FIGURE 31-37: IPD, COMPARATOR, LOW-POWER MODE, CxSP = 0, PIC16LF1847 ONLY

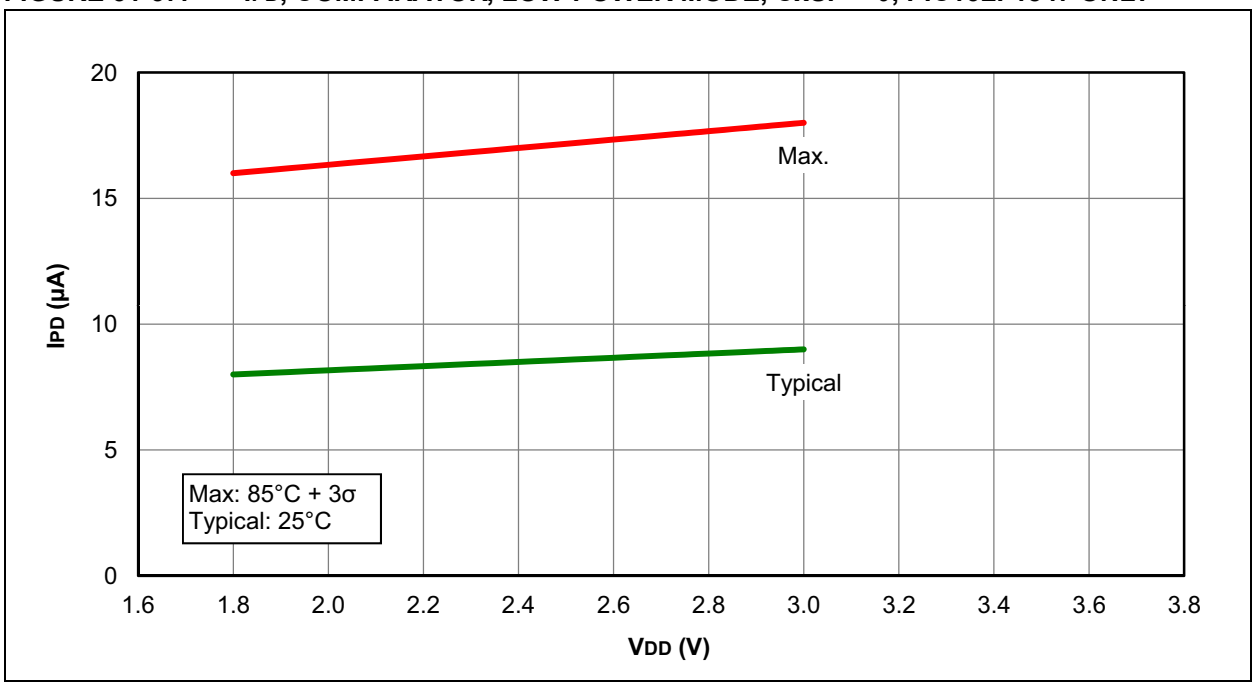


FIGURE 31-38: IPD, COMPARATOR, LOW-POWER MODE, CxSP = 0, PIC16F1847 ONLY

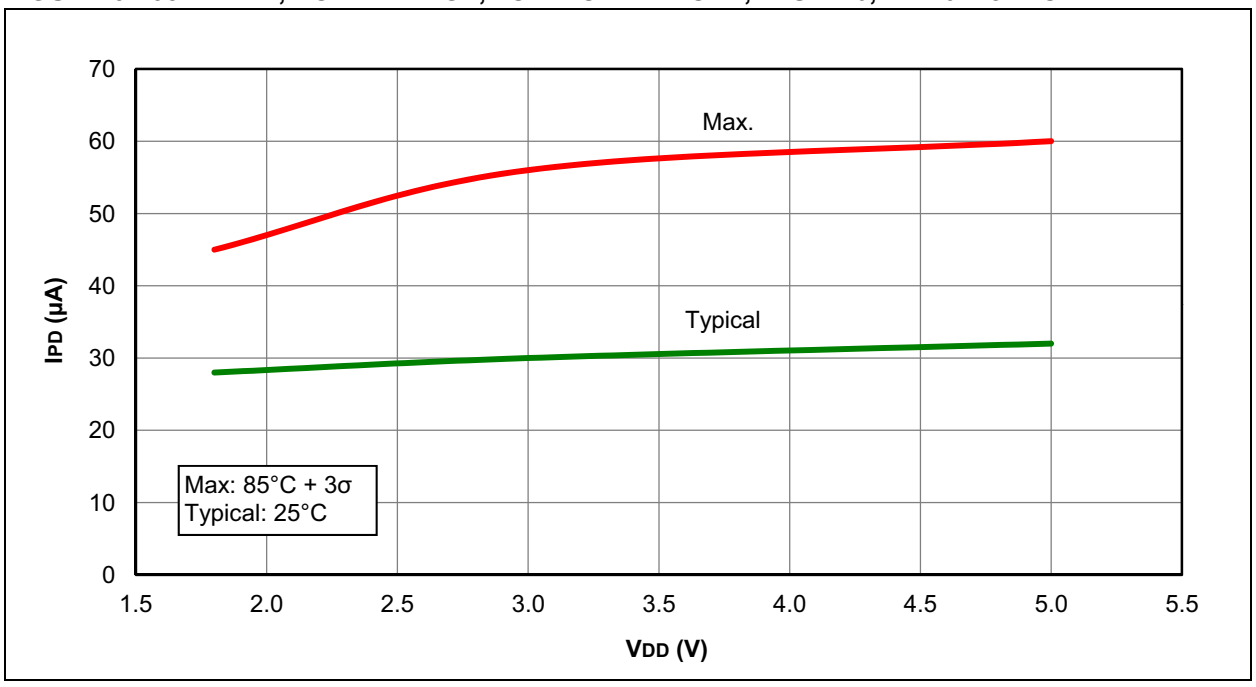


FIGURE 31-62: CAP SENSE CURRENT SINK/SOURCE CHARACTERISTICS FIXED VOLTAGE REFERENCE (CPSRM = 0), HIGH CURRENT RANGE (CPSRNG = 11)

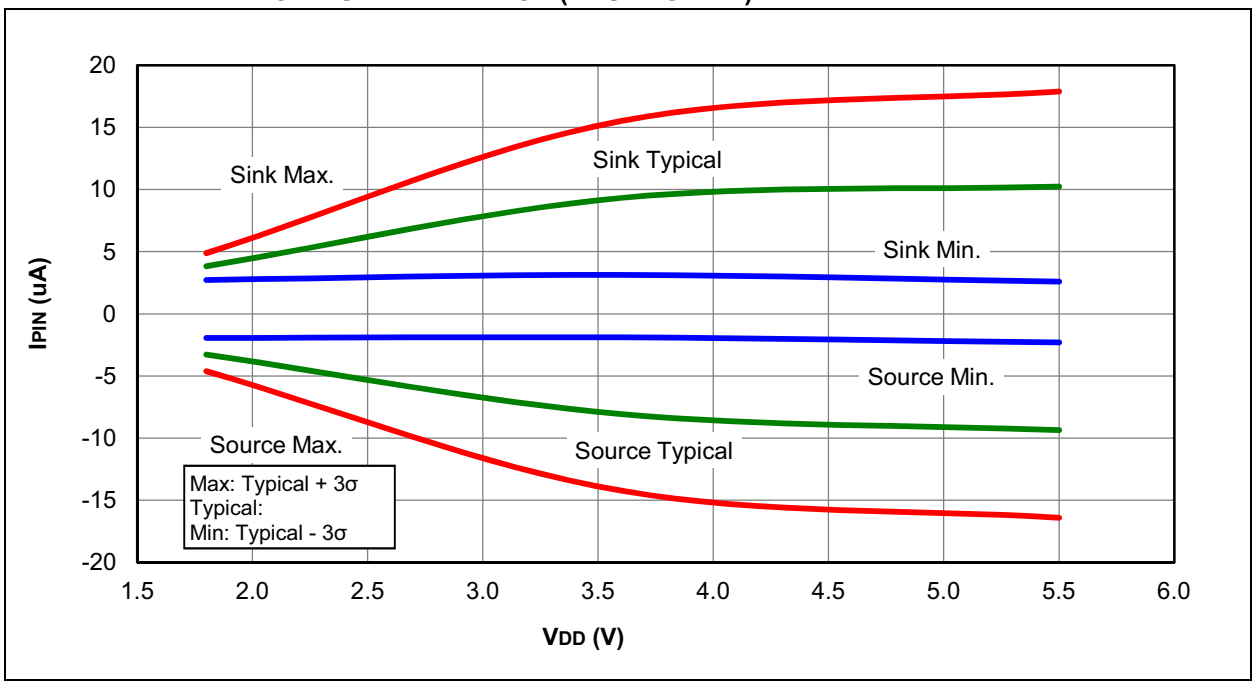


FIGURE 31-63: CAP SENSE CURRENT SINK/SOURCE CHARACTERISTICS FIXED VOLTAGE REFERENCE (CPSRM = 0), MEDIUM CURRENT RANGE (CPSRNG = 10)