



Welcome to [E-XFL.COM](https://www.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.

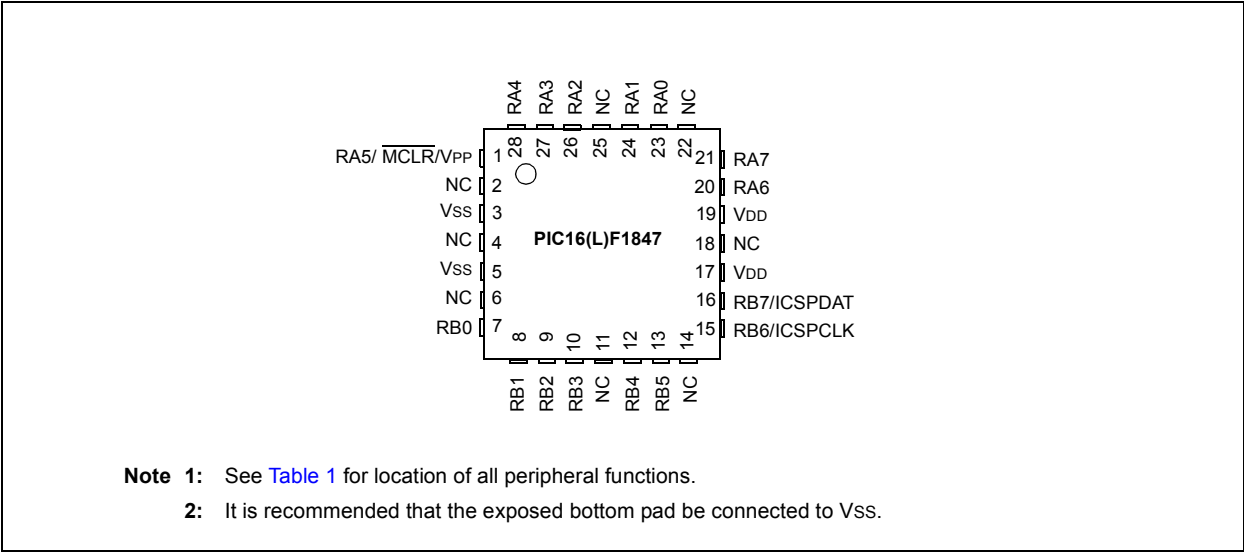
### Applications of "[Embedded - Microcontrollers](#)"

#### Details

Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	32MHz
Connectivity	I <sup>2</sup> C, 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 ~ 5.5V
Data Converters	A/D 12x10b
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	<a href="https://www.e-xfl.com/product-detail/microchip-technology/pic16f1847-i-ml">https://www.e-xfl.com/product-detail/microchip-technology/pic16f1847-i-ml</a>

# PIC16(L)F1847

## Pin Diagram – 28-Pin QFN/UQFN



# PIC16(L)F1847

---

NOTES:

## 3.0 MEMORY ORGANIZATION

There are three types of memory in PIC16(L)F1847: Data Memory, Program Memory and Data EEPROM Memory<sup>(1)</sup>.

- Program Memory
- Data Memory
  - Core Registers
  - Special Function Registers
  - General Purpose RAM
  - Common RAM
  - Device Memory Maps
  - Special Function Registers Summary
- Data EEPROM memory<sup>(1)</sup>

**Note 1:** The Data EEPROM Memory and the method to access Flash memory through the EECON registers is described in [Section 11.0 “Data EEPROM and Flash Program Memory Control”](#).

The following features are associated with access and control of program memory and data memory:

- PCL and PCLATH
- Stack
- Indirect Addressing

### 3.1 Program Memory Organization

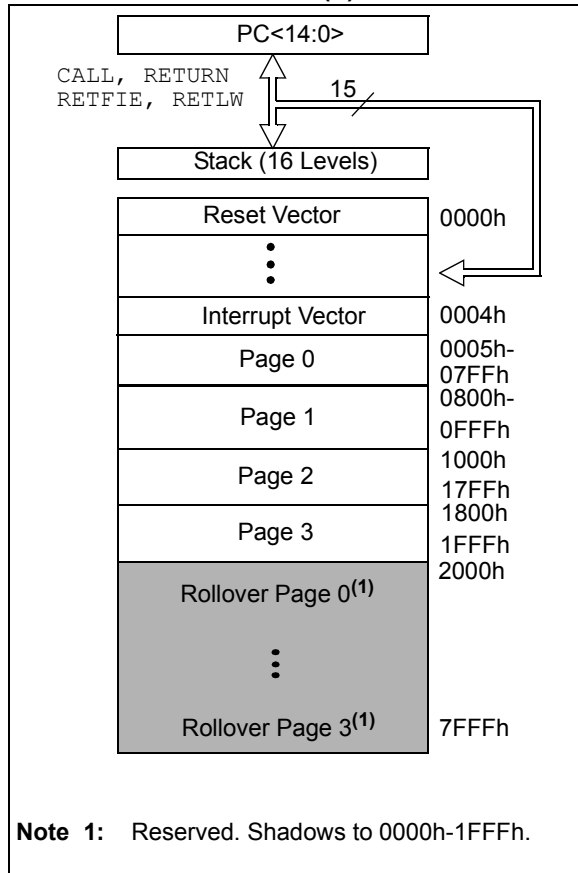
The enhanced mid-range core has a 15-bit program counter capable of addressing a 32K x 14 program memory space. [Table 3-1](#) shows the memory sizes implemented for the PIC16(L)F1847 family. Accessing a location above these boundaries will cause a wrap-around within the implemented memory space. The Reset vector is at 0000h and the interrupt vector is at 0004h (see [Figure 3-1](#)).

**TABLE 3-1: DEVICE SIZES AND ADDRESSES**

Device	Program Memory Space (Words)	Last Program Memory Address
PIC16(L)F1847	8,192	1FFFh

# PIC16(L)F1847

**FIGURE 3-1: PROGRAM MEMORY MAP AND STACK FOR PIC16(L)F1847**



## 3.1.1 READING PROGRAM MEMORY AS DATA

There are two methods of accessing constants in program memory. The first method is to use tables of RETLW instructions. The second method is to set an FSR to point to the program memory.

### 3.1.1.1 RETLW Instruction

The RETLW instruction can be used to provide access to tables of constants. The recommended way to create such a table is shown in [Example 3-1](#).

**EXAMPLE 3-1: RETLW INSTRUCTION**

```
constants
    BRW                ;Add Index in W to
                        ;program counter to
                        ;select data
    RETLW DATA0        ;Index0 data
    RETLW DATA1        ;Index1 data
    RETLW DATA2
    RETLW DATA3

my_function
    ;... LOTS OF CODE...
    MOVLW DATA_INDEX
    CALL constants
    ;... THE CONSTANT IS IN W
```

The BRW instruction makes this type of table very simple to implement. If your code must remain portable with previous generations of microcontrollers, then the BRW instruction is not available so the older table read method must be used.

### 3.1.1.2 Indirect Read with FSR

The program memory can be accessed as data by setting bit 7 of the FSRxH register and reading the matching INDFx register. The MOVLW instruction will place the lower eight bits of the addressed word in the W register. Writes to the program memory cannot be performed via the INDF registers. Instructions that access the program memory via the FSR require one extra instruction cycle to complete. [Example 3-2](#) demonstrates accessing the program memory via an FSR.

The HIGH directive will set bit<7> if a label points to a location in program memory.

**EXAMPLE 3-2: ACCESSING PROGRAM MEMORY VIA FSR**

```
constants
    RETLW DATA0        ;Index0 data
    RETLW DATA1        ;Index1 data
    RETLW DATA2
    RETLW DATA3

my_function
    ;... LOTS OF CODE...
    MOVLW LOW constants
    MOVWF FSR1L
    MOVLW HIGH constants
    MOVWF FSR1H
    MOVLW 0[FSR1]
    ;THE PROGRAM MEMORY IS IN W
```

# PIC16(L)F1847

## 7.3 Register Definitions: BOR Control

### REGISTER 7-1: BORCON: BROWN-OUT RESET CONTROL REGISTER

R/W-1/u	U-0	U-0	U-0	U-0	U-0	U-0	R-q/u
SBOREN	—	—	—	—	—	—	BORRDY
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      **SBOREN:** Software Brown-out Reset Enable bit  
If BOREN <1:0> in Configuration Words  $\neq 01$ :  
SBOREN is read/write, but has no effect on the BOR.  
If BOREN <1:0> in Configuration Words =  $01$ :  
1 = BOR Enabled  
0 = BOR Disabled

bit 6-1      **Unimplemented:** Read as '0'

bit 0      **BORRDY:** Brown-out Reset Circuit Ready Status bit  
1 = The Brown-out Reset circuit is active  
0 = The Brown-out Reset circuit is inactive

## REGISTER 8-6: PIR1: PERIPHERAL INTERRUPT REQUEST REGISTER 1

R/W-0/0	R/W-0/0	R-0/0	R-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
TMR1GIF	ADIF	RCIF	TXIF	SSP1IF	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	<b>TMR1GIF:</b> Timer1 Gate Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending
bit 6	<b>ADIF:</b> ADC Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending
bit 5	<b>RCIF:</b> USART Receive Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending
bit 4	<b>TXIF:</b> USART Transmit Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending
bit 3	<b>SSP1IF:</b> Synchronous Serial Port 1 (MSSP1) Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending
bit 2	<b>CCP1IF:</b> CCP1 Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending
bit 1	<b>TMR2IF:</b> Timer2 to PR2 Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending
bit 0	<b>TMR1IF:</b> Timer1 Overflow Interrupt Flag bit 1 = Interrupt is pending 0 = Interrupt is not pending

**Note:** Interrupt flag bits are set when an interrupt condition occurs, regardless of the state of its corresponding enable bit or the Global Enable bit, GIE of the INTCON register. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

# PIC16(L)F1847

## 11.7 Register Definitions: EEPROM and Flash Control

### REGISTER 11-1: EEDATL: EEPROM DATA 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
EEDAT<7:0>							
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 7-0                      **EEDAT<7:0>**: Read/write value for EEPROM data byte or Least Significant bits of program memory

### REGISTER 11-2: EEDATH: EEPROM DATA HIGH BYTE REGISTER

U-0	U-0	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u
—	—	EEDAT<13:8>					
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 7-6                      **Unimplemented**: Read as '0'

bit 5-0                      **EEDAT<13:8>**: Read/write value for Most Significant bits of program memory

### REGISTER 11-3: EEADRL: EEPROM ADDRESS REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
EEADR<7:0>							
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 7-0                      **EEADR<7:0>**: Specifies the Least Significant bits for program memory address or EEPROM address

### REGISTER 11-4: EEADRH: EEPROM ADDRESS HIGH BYTE REGISTER

U-1	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—(1)	EEADR<14:8>						
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 '1'

bit 6-0                      **EEADR<14:8>**: Specifies the Most Significant bits for program memory address or EEPROM address

**Note 1:** Unimplemented, read as '1'.





## 21.11 Timer1 Control Register

The Timer1 Control register (T1CON), shown in [Register 21-1](#), is used to control Timer1 and select the various features of the Timer1 module.

**REGISTER 21-1: T1CON: TIMER1 CONTROL REGISTER**

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	U-0	R/W-0/u
TMR1CS<1:0>		T1CKPS<1:0>		T1OSCEN	$\overline{\text{T1SYNC}}$	—	TMR1ON
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

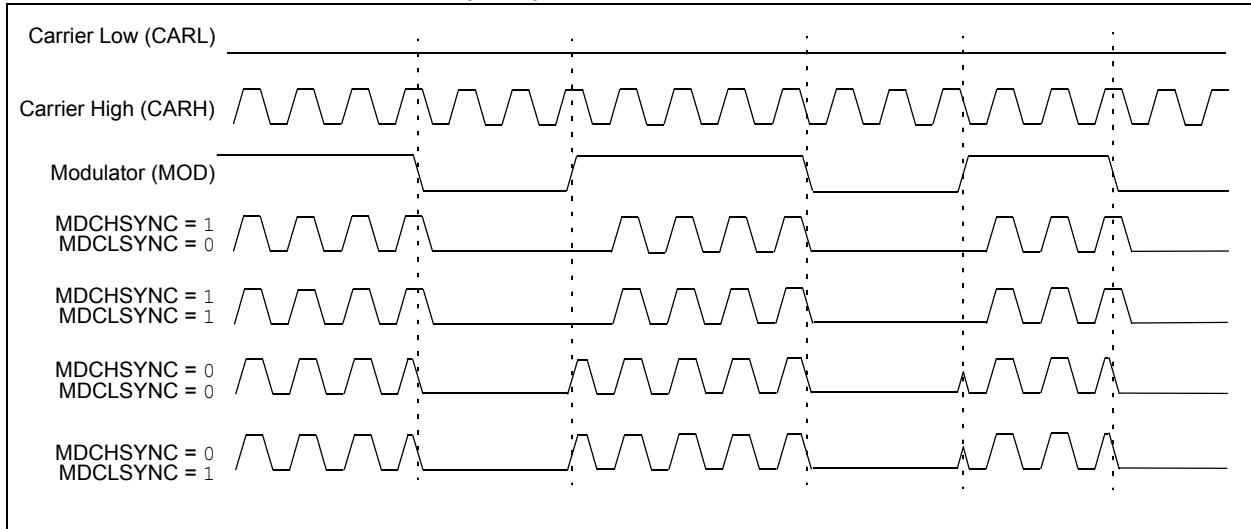
-n/n = Value at POR and BOR/Value at all other Resets

'1' = Bit is set

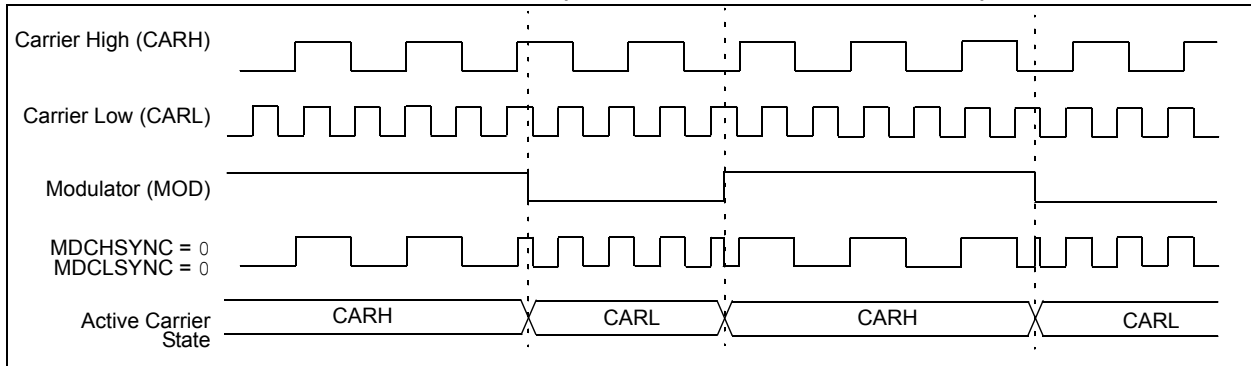
'0' = Bit is cleared

- bit 7-6      **TMR1CS<1:0>**: Timer1 Clock Source Select bits  
               11 = Timer1 clock source is Capacitive Sensing Oscillator (CAPOSC)  
               10 = Timer1 clock source is pin or oscillator:  
                   If T1OSCEN = 0:  
                     External clock from T1CKI pin (on the rising edge)  
                   If T1OSCEN = 1:  
                     Crystal oscillator on T1OSI/T1OSO pins  
               01 = Timer1 clock source is system clock (Fosc)  
               00 = Timer1 clock source is instruction clock (Fosc/4)
- bit 5-4      **T1CKPS<1:0>**: Timer1 Input Clock Prescale Select bits  
               11 = 1:8 Prescale value  
               10 = 1:4 Prescale value  
               01 = 1:2 Prescale value  
               00 = 1:1 Prescale value
- bit 3        **T1OSCEN**: LP Oscillator Enable Control bit  
               1 = Dedicated Timer1 oscillator circuit enabled  
               0 = Dedicated Timer1 oscillator circuit disabled
- bit 2        **T1SYNC**: Timer1 External Clock Input Synchronization Control bit  
               TMR1CS<1:0> = 1X  
               1 = Do not synchronize external clock input  
               0 = Synchronize external clock input with system clock (Fosc)  
  
               TMR1CS<1:0> = 0X  
               This bit is ignored. Timer1 uses the internal clock when TMR1CS<1:0> = 1X.
- bit 1        **Unimplemented**: Read as '0'
- bit 0        **TMR1ON**: Timer1 On bit  
               1 = Enables Timer1  
               0 = Stops Timer1  
                   Clears Timer1 Gate flip-flop

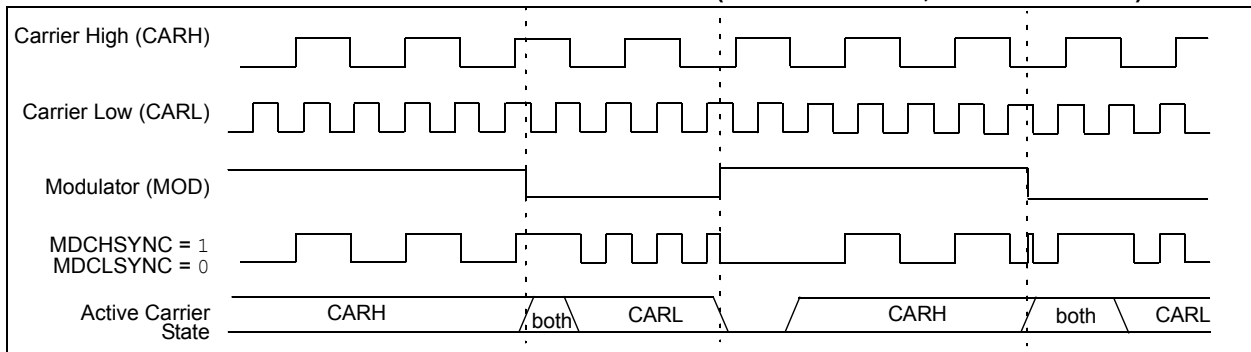
**FIGURE 23-2: ON OFF KEYING (OOK) SYNCHRONIZATION**



**EXAMPLE 23-1: NO SYNCHRONIZATION (MDSHSYNC = 0, MDCLSYNC = 0)**



**FIGURE 23-3: CARRIER HIGH SYNCHRONIZATION (MDSHSYNC = 1, MDCLSYNC = 0)**



# PIC16(L)F1847

---

NOTES:

## 24.2.5 COMPARE DURING SLEEP

The Compare mode is dependent upon the system clock (Fosc) for proper operation. Since Fosc is shut down during Sleep mode, the Compare mode will not function properly during Sleep.

## 24.2.6 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 24-4: SUMMARY OF REGISTERS ASSOCIATED WITH COMPARE**

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
CCP1CON	P1M<1:0>		DC1B<1:0>		CCP1M<3:0>				226
CCPR1L	Capture/Compare/PWM Register Low Byte (LSB)								204*
CCPR1H	Capture/Compare/PWM Register High Byte (MSB)								204*
CM1CON0	C1ON	C1OUT	C1OE	C1POL	—	C1SP	C1HYS	C1SYNC	170
CM1CON1	C1INTP	C1INTN	C1PCH<1:0>		—	—	C1NCH<1:0>		171
CM2CON0	C2ON	C2OUT	C2OE	C2POL	—	C2SP	C2HYS	C2SYNC	170
CM2CON1	C2INTP	C2INTN	C2PCH<1:0>		—	—	C2NCH<1:0>		171
CCP2CON	P2M<1:0>		DC2B<1:0>		CCP2M<3:0>				226
CCPR2L	Capture/Compare/PWM Register Low Byte (LSB)								226
CCPR2H	Capture/Compare/PWM Register High Byte (MSB)								226
CCP3CON	—	—	DC3B<1:0>		CCP3M<3:0>				226
CCPR3L	Capture/Compare/PWM Register Low Byte (LSB)								226
CCPR3H	Capture/Compare/PWM Register High Byte (MSB)								226
CCP4CON	—	—	DC4B<1:0>		CCP4M<3:0>				226
CCPR4L	Capture/Compare/PWM Register Low Byte (LSB)								226
CCPR4H	Capture/Compare/PWM Register High Byte (MSB)								226
INTCON	GIE	PEIE	TMR0IE	INTE	IOCE	TMR0IF	INTF	IOCF	88
PIE1	TMR1GIE	ADIE	RCIE	TXIE	SSP1IE	CCP1IE	TMR2IE	TMR1IE	89
PIE2	OSFIE	C2IE	C1IE	EEIE	BCL1IE	—	—	CCP2IE	90
PIE3	—	—	CCP4IE	CCP3IE	TMR6IE	—	TMR4IE	—	91
PIR1	TMR1GIF	ADIF	RCIF	TXIF	SSP1IF	CCP1IF	TMR2IF	TMR1IF	93
PIR2	OSFIF	C2IF	C1IF	EEIF	BCL1IF	—	—	CCP2IF	94
PIR3	—	—	CCP4IF	CCP3IF	TMR6IF	—	TMR4IF	—	95
T1CON	TMR1CS<1:0>		T1CKPS<1:0>		T1OSCEN	T1SYNC	—	TMR1ON	185
T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/DONE	T1GVAL	T1GSS<1:0>		186
TMR1L	Holding Register for the Least Significant Byte of the 16-bit TMR1 Register								177*
TMR1H	Holding Register for the Most Significant Byte of the 16-bit TMR1 Register								177*
TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	120
TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	126

**Legend:** — = Unimplemented locations, read as '0'. Shaded cells are not used by Compare mode.

\* Page provides register information.

## 25.4.9 ACKNOWLEDGE SEQUENCE

The 9h SCLx pulse for any transferred byte in I<sup>2</sup>C is dedicated as an Acknowledge. It allows receiving devices to respond back to the transmitter by pulling the SDAx line low. The transmitter must release control of the line during this time to shift in the response. The Acknowledge ( $\overline{\text{ACK}}$ ) is an active-low signal, pulling the SDAx line low indicated to the transmitter that the device has received the transmitted data and is ready to receive more.

The result of an  $\overline{\text{ACK}}$  is placed in the ACKSTAT bit of the SSPxCON2 register.

Slave software, when the AHEN and DHEN bits are set, allow the user to set the  $\overline{\text{ACK}}$  value sent back to the transmitter. The ACKDT bit of the SSPxCON2 register is set/cleared to determine the response.

Slave hardware will generate an  $\overline{\text{ACK}}$  response if the AHEN and DHEN bits of the SSPxCON3 register are clear.

There are certain conditions where an  $\overline{\text{ACK}}$  will not be sent by the slave. If the BF bit of the SSPxSTAT register or the SSPOV bit of the SSPxCON1 register are set when a byte is received.

When the module is addressed, after the 8th falling edge of SCLx on the bus, the ACKTIM bit of the SSPxCON3 register is set. The ACKTIM bit indicates the acknowledge time of the active bus. The ACKTIM Status bit is only active when the AHEN bit or DHEN bit is enabled.

## 25.5 I<sup>2</sup>C SLAVE MODE OPERATION

The MSSPx Slave mode operates in one of four modes selected in the SSPM bits of SSPxCON1 register. The modes can be divided into 7-bit and 10-bit Addressing mode. 10-bit Addressing modes operate the same as 7-bit with some additional overhead for handling the larger addresses.

Modes with Start and Stop bit interrupts operate the same as the other modes with SSPxIF additionally getting set upon detection of a Start, Restart, or Stop condition.

### 25.5.1 SLAVE MODE ADDRESSES

The SSPxADD register ([Register 25-6](#)) contains the Slave mode address. The first byte received after a Start or Restart condition is compared against the value stored in this register. If the byte matches, the value is loaded into the SSPxBUF register and an interrupt is generated. If the value does not match, the module goes idle and no indication is given to the software that anything happened.

The SSPx Mask register ([Register 25-5](#)) affects the address matching process. See [Section 25.5.9 “SSPx Mask Register”](#) for more information.

#### 25.5.1.1 I<sup>2</sup>C Slave 7-bit Addressing Mode

In 7-bit Addressing mode, the LSb of the received data byte is ignored when determining if there is an address match.

#### 25.5.1.2 I<sup>2</sup>C Slave 10-bit Addressing Mode

In 10-bit Addressing mode, the first received byte is compared to the binary value of ‘1 1 1 0 A9 A8 0’. A9 and A8 are the two MSb’s of the 10-bit address and stored in bits 2 and 1 of the SSPxADD register.

After the acknowledge of the high byte the UA bit is set and SCLx is held low until the user updates SSPxADD with the low address. The low address byte is clocked in and all eight bits are compared to the low address value in SSPxADD. Even if there is not an address match; SSPxIF and UA are set, and SCLx is held low until SSPxADD is updated to receive a high byte again. When SSPxADD is updated the UA bit is cleared. This ensures the module is ready to receive the high address byte on the next communication.

A high and low address match as a write request is required at the start of all 10-bit addressing communication. A transmission can be initiated by issuing a Restart once the slave is addressed, and clocking in the high address with the R/W bit set. The slave hardware will then acknowledge the read request and prepare to clock out data. This is only valid for a slave after it has received a complete high and low address byte match.

## 25.6.7 I<sup>2</sup>C MASTER MODE RECEPTION

Master mode reception (Figure 25-29) is enabled by programming the Receive Enable bit, RCEN bit of the SSPxCON2 register.

**Note:** The MSSPx module must be in an Idle state before the RCEN bit is set or the RCEN bit will be disregarded.

The Baud Rate Generator begins counting and on each rollover, the state of the SCLx pin changes (high-to-low/low-to-high) and data is shifted into the SSPxSR. After the falling edge of the 8th clock, the receive enable flag is automatically cleared, the contents of the SSPxSR are loaded into the SSPxBUF, the BF flag bit is set, the SSPxIF flag bit is set and the Baud Rate Generator is suspended from counting, holding SCLx low. The MSSPx is now in Idle state awaiting the next command. When the buffer is read by the CPU, the BF flag bit is automatically cleared. The user can then send an Acknowledge bit at the end of reception by setting the Acknowledge Sequence Enable, ACKEN bit of the SSPxCON2 register.

### 25.6.7.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPxBUF from SSPxSR. It is cleared when the SSPxBUF register is read.

### 25.6.7.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when eight bits are received into the SSPxSR and the BF flag bit is already set from a previous reception.

### 25.6.7.3 WCOL Status Flag

If the user writes the SSPxBUF when a receive is already in progress (i.e., SSPxSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write does not occur).

### 25.6.7.4 Typical Receive Sequence:

1. The user generates a Start condition by setting the SEN bit of the SSPxCON2 register.
2. SSPxIF is set by hardware on completion of the Start.
3. SSPxIF is cleared by software.
4. User writes SSPxBUF with the slave address to transmit and the R/W bit set.
5. Address is shifted out the SDAx pin until all eight bits are transmitted. Transmission begins as soon as SSPxBUF is written to.
6. The MSSPx module shifts in the  $\overline{\text{ACK}}$  bit from the slave device and writes its value into the ACKSTAT bit of the SSPxCON2 register.
7. The MSSPx module generates an interrupt at the end of the 9th clock cycle by setting the SSPxIF bit.
8. User sets the RCEN bit of the SSPxCON2 register and the Master clocks in a byte from the slave.
9. After the 8th falling edge of SCLx, SSPxIF and BF are set.
10. Master clears SSPxIF and reads the received byte from SSPxBUF, clears BF.
11. Master sets  $\overline{\text{ACK}}$  value sent to slave in ACKDT bit of the SSPxCON2 register and initiates the  $\overline{\text{ACK}}$  by setting the ACKEN bit.
12. Masters  $\overline{\text{ACK}}$  is clocked out to the Slave and SSPxIF is set.
13. User clears SSPxIF.
14. Steps 8-13 are repeated for each received byte from the slave.
15. Master sends a not  $\overline{\text{ACK}}$  or Stop to end communication.

# PIC16(L)F1847

## REGISTER 27-2: CPSCON1: CAPACITIVE SENSING CONTROL REGISTER 1

U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	CPSCH<3:0>			
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 7-4 **Unimplemented:** Read as '0'

bit 3-0 **CPSCH<3:0>:** Capacitive Sensing Channel Select bits

If CPSON = 0:

These bits are ignored. No channel is selected.

If CPSON = 1:

1111 = Reserved. Do not use.  
 1110 = Reserved. Do not use.  
 1101 = Reserved. Do not use.  
 1100 = Reserved. Do not use.  
 1011 = channel 11, (CPS11)  
 1010 = channel 10, (CPS10)  
 1001 = channel 9, (CPS9)  
 1000 = channel 8, (CPS8)  
 0111 = channel 7, (CPS7)  
 0110 = channel 6, (CPS6)  
 0101 = channel 5, (CPS5)  
 0100 = channel 4, (CPS4)  
 0011 = channel 3, (CPS3)  
 0010 = channel 2, (CPS2)  
 0001 = channel 1, (CPS1)  
 0000 = channel 0, (CPS0)



## 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 ≤ k ≤ 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

**Example:** 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

**Example:** 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 nRI flag of the PCON register.

**Status Affected:** None

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

# PIC16(L)F1847

**TABLE 30-5: MEMORY PROGRAMMING REQUIREMENTS**

<b>Standard Operating Conditions</b> (unless otherwise stated)							
Param. No.	Sym.	Characteristic	Min.	Typ†	Max.	Units	Conditions
<b>Program Memory Programming Specifications</b>							
D110	VIHH	Voltage on $\overline{\text{MCLR}}$ /VPP pin	8.0	—	9.0	V	(Note 3)
D111	IDDP	Supply Current during Programming	—	—	10	mA	
D112	VBE	VDD for Bulk Erase	2.7	—	VDDMAX	V	
D113	VPEW	VDD for Write or Row Erase	VDDMIN	—	VDDMAX	V	
D114	IPPPGM	Current on $\overline{\text{MCLR}}$ /VPP during Erase/Write	—	1.0	—	mA	
D115	IDDPGM	Current on VDD during Erase/Write	—	5.0	—	mA	
<b>Data EEPROM Memory</b>							
D116	ED	Byte Endurance	100K	—	—	E/W	-40°C to +85°C
D117	VDRW	VDD for Read/Write	VDDMIN	—	VDDMAX	V	
D118	TDEW	Erase/Write Cycle Time	—	4.0	5.0	ms	
D119	TRETD	Characteristic Retention	—	40	—	Year	Provided no other specifications are violated
D120	TREF	Number of Total Erase/Write Cycles before Refresh	1M	10M	—	E/W	-40°C to +85°C (Note 2)
<b>Program Flash Memory</b>							
D121	EP	Cell Endurance	10K	—	—	E/W	-40°C to +85°C (Note 1)
D122	VPRW	VDD for Read/Write	VDDMIN	—	VDDMAX	V	
D123	TIW	Self-timed Write Cycle Time	—	2	2.5	ms	
D124	TRETD	Characteristic Retention	—	40	—	Year	Provided no other specifications are violated

† 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: Self-write and Block Erase.  
 2: Refer to [Section 11.2 "Using the Data EEPROM"](#) for a more detailed discussion on data EEPROM endurance.  
 3: Required only if single-supply programming is disabled.

# PIC16(L)F1847

## 30.4 AC Characteristics

Timing Parameter Symbolology has been created with one of the following formats:

- 1. TppS2ppS
- 2. TppS

<b>T</b>			
F	Frequency	T	Time

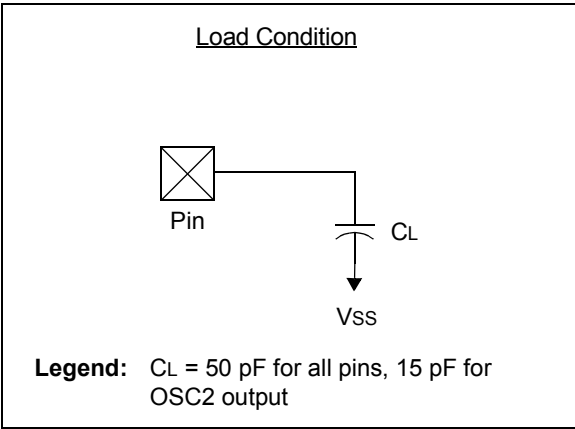
Lowercase letters (pp) and their meanings:

<b>pp</b>			
cc	CCP1	osc	OSC1
ck	CLKOUT	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ or $\overline{WR}$
di	SDIx	sc	$\overline{SCKx}$
do	SDO	ss	$\overline{SS}$
dt	Data in	t0	T0CKI
io	I/O PORT	t1	T1CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

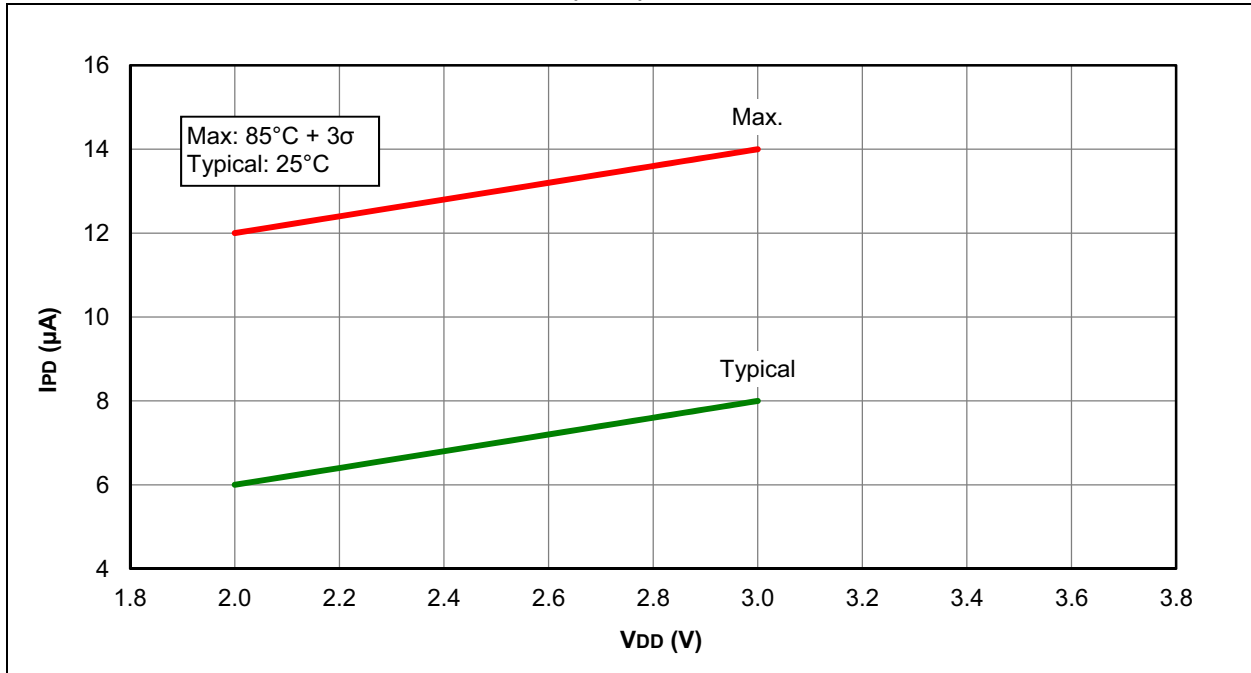
Uppercase letters and their meanings:

<b>S</b>			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (High-impedance)	V	Valid
L	Low	Z	High-impedance

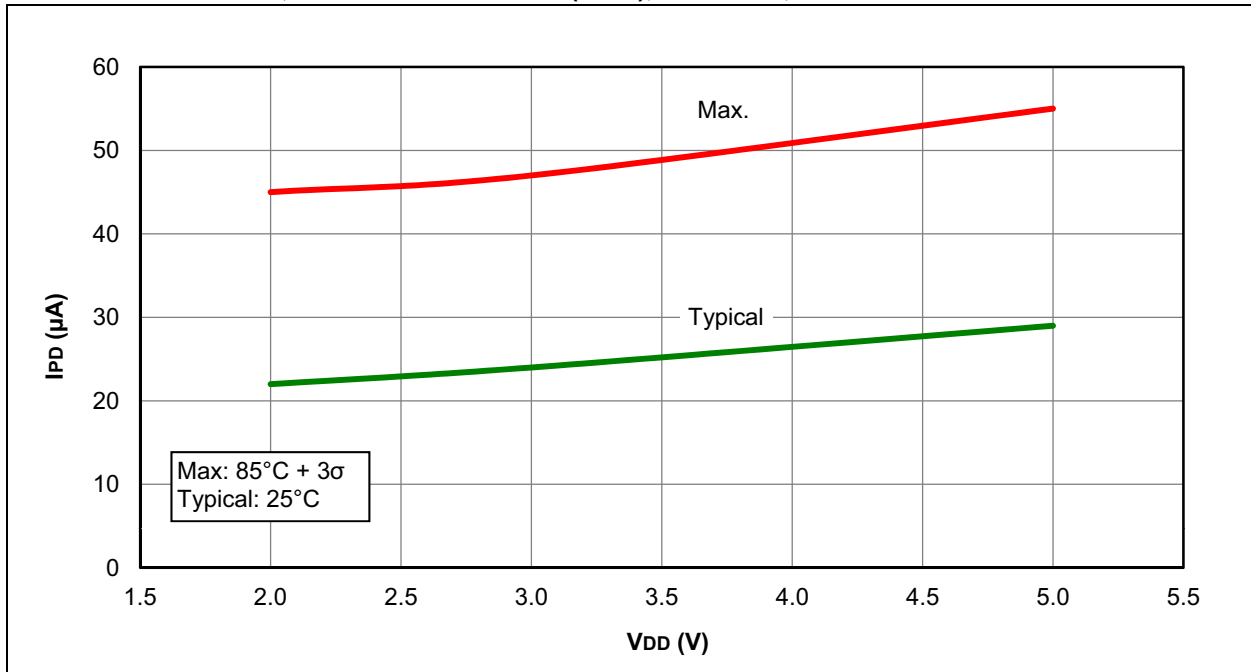
FIGURE 30-4: LOAD CONDITIONS



**FIGURE 31-27:  $I_{PD}$ , BROWN-OUT RESET (BOR), BORV = 1, PIC16LF1847 ONLY**



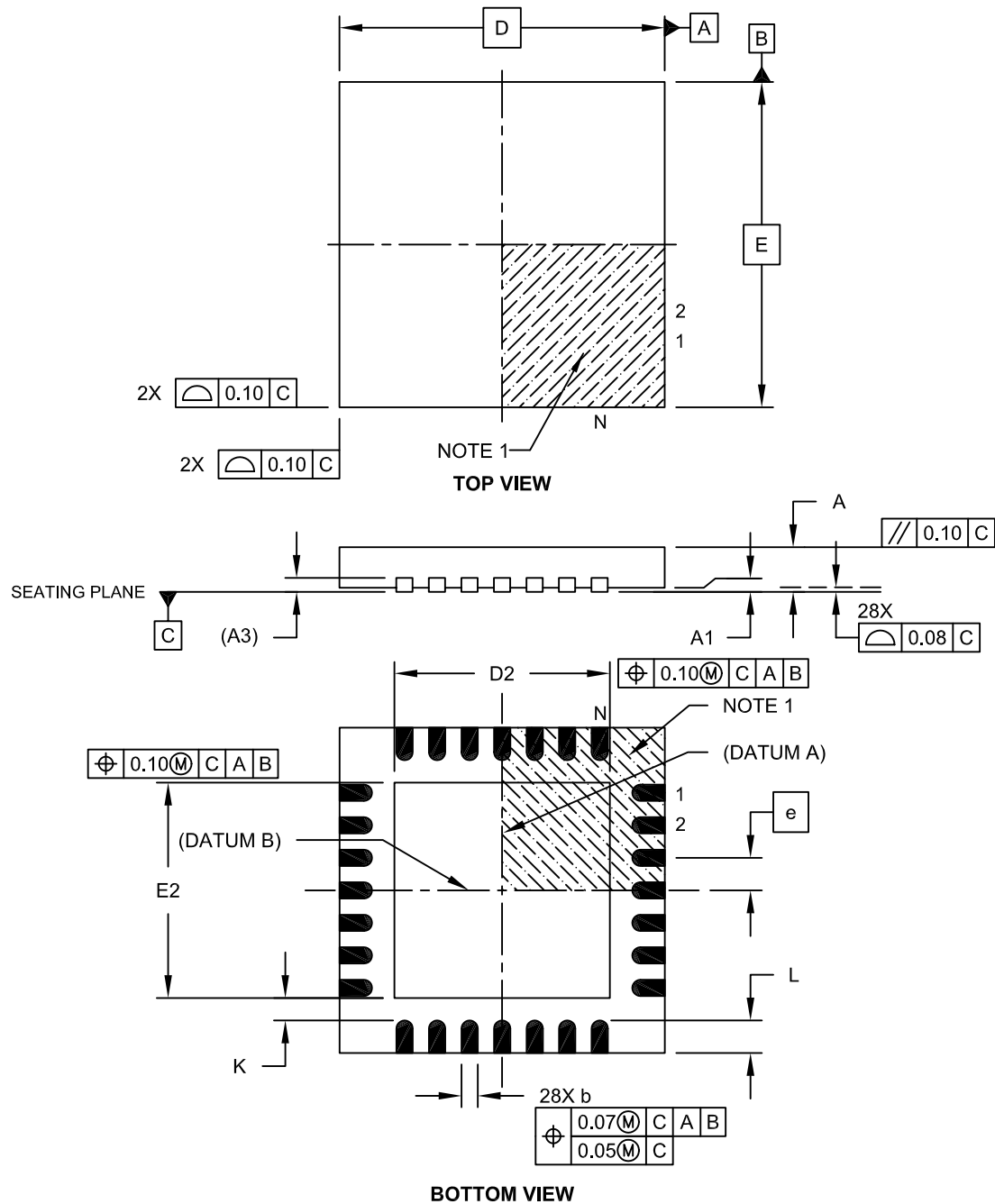
**FIGURE 31-28:  $I_{PD}$ , BROWN-OUT RESET (BOR), BORV = 1, PIC16F1847 ONLY**



# PIC16(L)F1847

## 28-Lead Plastic Ultra Thin Quad Flat, No Lead Package (MV) – 4x4x0.5 mm Body [UQFN]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at <http://www.microchip.com/packaging>



Microchip Technology Drawing C04-152A Sheet 1 of 2