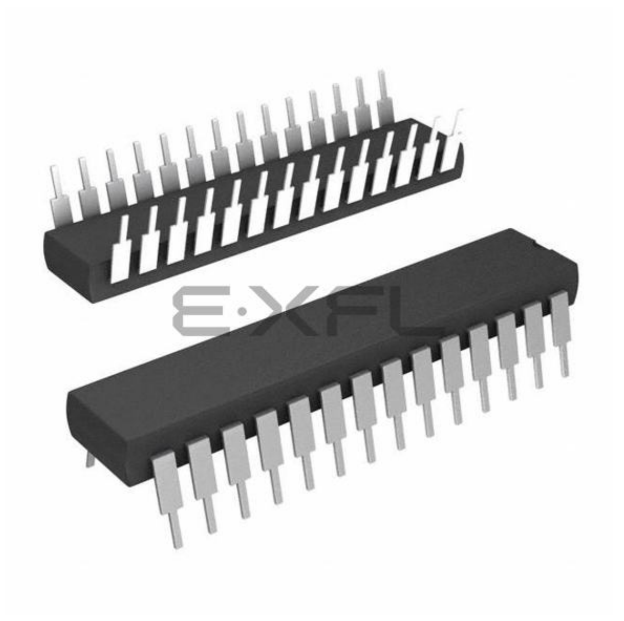
Microchip Technology - PIC18F27K42-E/SP Datasheet



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	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, HLVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	128KB (64K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	8K x 8
Voltage - Supply (Vcc/Vdd)	2.3V ~ 5.5V
Data Converters	A/D 24x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Through Hole
Package / Case	28-DIP (0.300", 7.62mm)
Supplier Device Package	28-SPDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f27k42-e-sp

REGISTER 5-7: CONFIGURATION WORD 4L (30 0006h)

R/W-1	U-1	U-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{WRTAPP}}$ (1)	—	—	$\overline{\text{SAFEN}}$ (1)	$\overline{\text{BBEN}}$ (1)	BBSIZE<2:0> (2)		
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘1’	
-n = Value for blank device	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **$\overline{\text{WRTAPP}}$:** Application Block Write Protection bit(1)
`1` = Application Block is NOT write-protected
`0` = Application Block is write-protected

bit 6-5 **Unimplemented:** Read as ‘`1`’

bit 4 **$\overline{\text{SAFEN}}$:** Storage Area Flash Enable bit(1)
`1` = SAF is disabled
`0` = SAF is enabled

bit 3 **$\overline{\text{BBEN}}$:** Boot Block Enable bit(1)
`1` = Boot Block disabled
`0` = Boot Block enabled

bit 2-0 **BBSIZE<2:0>:** Boot Block Size Selection bits(2)
Refer to Table 5-1.

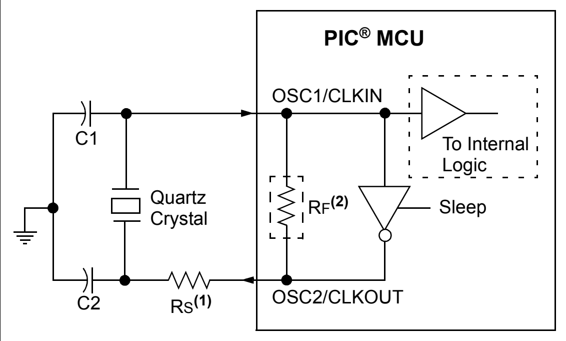
Note 1: Bits are implemented as sticky bits. Once protection is enabled through ICSP™ or a self-write, it can only be reset through a Bulk Erase.
2: BBSIZE<2:0> bits can only be changed when $\overline{\text{BBEN}}$ = `1`. Once $\overline{\text{BBEN}}$ = `0`, BBSIZE<2:0> can only be changed through a Bulk Erase.

TABLE 5-1: BOOT BLOCK SIZE BITS

$\overline{\text{BBEN}}$	BBSIZE<2:0>	Boot Block Size (words)	END_ADDRESS_BOOT	Device Size(1)		
				16k	32k	64k
1	xxx	0	—	X	X	X
0	111	512	00 03FFh	X	X	X
0	110	1024	00 07FFh	X	X	X
0	101	2048	00 0FFFh	X	X	X
0	100	4096	00 1FFFh	X	X	X
0	011	8192	00 3FFFh	X	X	X
0	010	16384	00 7FFFh	—	X	X
0	001	32768	00 FFFFh		Note 2	X
0	000	32768	00 FFFFh	—	—	—

Note 1: For each device, the quoted device size specification is listed in Table 4-1.
2: The maximum boot block size is half the user program memory size. All selections higher than the maximum size default to maximum boot block size of half PFM. For example, all settings of BBSIZE = `000` through BBSIZE = `010`, default to a boot block size of 16 kW on a 32 kW device.

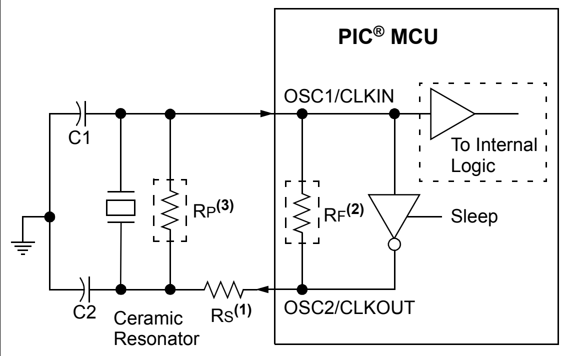
FIGURE 7-3: QUARTZ CRYSTAL OPERATION (LP, XT OR HS MODE)



Note 1: A series resistor (RS) may be required for quartz crystals with low drive level.

2: The value of RF varies with the Oscillator mode selected (typically between 2 MΩ to 10 MΩ).

FIGURE 7-4: CERAMIC RESONATOR OPERATION (XT OR HS MODE)



Note 1: A series resistor (RS) may be required for ceramic resonators with low drive level.

2: The value of RF varies with the Oscillator mode selected (typically between 2 MΩ to 10 MΩ).

3: An additional parallel feedback resistor (RP) may be required for proper ceramic resonator operation.

7.2.1.3 Oscillator Start-up Timer (OST)

If the oscillator module is configured for LP, XT or HS modes, the Oscillator Start-up Timer (OST) counts 1024 oscillations from OSC1. This occurs following a Power-on Reset (POR), or a wake-up from Sleep. The OST ensures that the oscillator circuit, using a quartz crystal resonator or ceramic resonator, has started and is providing a stable system clock to the oscillator module.

7.2.1.4 4x PLL

The oscillator module contains a 4x PLL that can be used with the external clock sources to provide a system clock source. The input frequency for the PLL must fall within specifications. See the PLL Clock Timing Specifications in Table 44-11.

The PLL can be enabled for use by one of two methods:

1. Program the RSTOSC bits in the Configuration Word 1 to `010` (enable EXTOSC with 4x PLL).
2. Write the NOSC bits in the OSCCON1 register to `010` (enable EXTOSC with 4x PLL).

9.3 Interrupt Priority

The final priority level for any pending source of interrupt is determined first by the user-assigned priority of that source in the IPRx register, then by the natural order priority within the IVT. The sections below detail the operation of Interrupt priorities.

9.3.1 USER (SOFTWARE) PRIORITY

User-assigned interrupt priority is enabled by setting the IPEN bit in the INTCON0 register (Register 9-1). Each peripheral interrupt source can be assigned a high or low priority level by the user. The user-assignable interrupt priority control bits for each interrupt are located in the IPRx registers (Registers 9-25 through 9-35).

The interrupts are serviced based on predefined interrupt priority scheme defined below.

1. Interrupts set by the user as high-priority interrupt have higher precedence of execution. High-priority interrupts will override a low-priority request when:
   a) A low priority interrupt has been requested or its request is already pending.
   b) A low- and high-priority interrupt are triggered concurrently, i.e., on the same instruction cycle[(1)].
   c) A low-priority interrupt was requested and the corresponding Interrupt Service Routine is currently executing. In this case, the lower priority interrupt routine will complete executing after the high-priority interrupt has been serviced[(2)].
2. Interrupts set by the user as a low priority have the lower priority of execution and are preempted by any high-priority interrupt.
3. Interrupts defined with the same software priority cannot preempt or interrupt each other. Concurrent pending interrupts with the same user priority are resolved using the natural order priority. (when MVECEN = ON) or in the order the interrupt flag bits are polled in the ISR (when MVECEN = OFF).

Note 1: When a high priority interrupt preempts a concurrent low priority interrupt, the GIEL bit may be cleared in the high priority Interrupt Service Routine. If the GIEL bit is cleared, the low priority interrupt will NOT be serviced even if it was originally requested. The corresponding interrupt flag needs to be cleared in user code.

2: When a high priority interrupt is requested while a low priority Interrupt Service Routine is executing, the GIEL bit may be cleared in the high priority Interrupt Service Routine. The pending low priority interrupt will resume even if the GIEL bit is cleared.

REGISTER 11-2: WDTCON1: WATCHDOG TIMER CONTROL REGISTER 1

U-0	R/W(3)-q/q(1)	R/W(3)-q/q(1)	R/W(3)-q/q(1)	U-0	R/W(4)-q/q(2)	R/W(4)-q/q(2)	R/W(4)-q/q(2)
—	CS<2:0>			—	WINDOW<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	q = Value depends on condition

bit 7 **Unimplemented:** Read as ‘0’

bit 6-4 **CS<2:0>:** Watchdog Timer Clock Select bits
- 111 = Reserved
- •
- •
- •
- 011 = Reserved
- 010 = SOSC
- 001 = MFINTOSC 31.25 kHz
- 000 = LFINTOSC 31 kHz

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **WINDOW<2:0>:** Watchdog Timer Window Select bits

WINDOW<2:0>	Window delay Percent of time	Window opening Percent of time
111	N/A	100
110	12.5	87.5
101	25	75
100	37.5	62.5
011	50	50
010	62.5	37.5
001	75	25
000	87.5	12.5

Note 1: If WDTCCS <2:0> in CONFIG3H = 111, the Reset value of CS<2:0> is 000.

2: The Reset value of WINDOW<2:0> is determined by the value of WDTCWS<2:0> in the CONFIG3H register.

3: If WDTCCS<2:0> in CONFIG3H ≠ 111, these bits are read-only.

4: If WDTCWS<2:0> in CONFIG3H ≠ 111, these bits are read-only.

Example 12-3 shows the sequence to do a 16 x 16 unsigned multiplication. Equation 12-1 shows the algorithm that is used. The 32-bit result is stored in four registers (RES<3:0>).

EQUATION 12-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES3{:}RES0 &= ARG1H{:}ARG1L \bullet ARG2H{:}ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L)
\end{aligned}
$$

EXAMPLE 12-3: 16 x 16 UNSIGNED MULTIPLY ROUTINE

```
    MOVF    ARG1L, W
    MULWF   ARG2L               ; ARG1L * ARG2L->
                                ; PRODH:PRODL
    MOVFF   PRODH, RES1         ;
    MOVFF   PRODL, RES0         ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H               ; ARG1H * ARG2H->
                                ; PRODH:PRODL
    MOVFF   PRODH, RES3         ;
    MOVFF   PRODL, RES2         ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H               ; ARG1L * ARG2H->
                                ; PRODH:PRODL
    MOVF    PRODL, W            ;
    ADDWF   RES1, F             ; Add cross
    MOVF    PRODH, W            ; products
    ADDWFC  RES2, F             ;
    CLRF    WREG                ;
    ADDWFC  RES3, F             ;
;
    MOVF    ARG1H, W            ;
    MULWF   ARG2L               ; ARG1H * ARG2L->
                                ; PRODH:PRODL
    MOVF    PRODL, W            ;
    ADDWF   RES1, F             ; Add cross
    MOVF    PRODH, W            ; products
    ADDWFC  RES2, F             ;
    CLRF    WREG                ;
    ADDWFC  RES3, F             ;
```

Example 12-4 shows the sequence to do a 16 x 16 signed multiply. Equation 12-2 shows the algorithm used. The 32-bit result is stored in four registers (RES<3:0>). To account for the sign bits of the arguments, the MSb for each argument pair is tested and the appropriate subtractions are done.

EQUATION 12-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES3{:}RES0 &= ARG1H{:}ARG1L \bullet ARG2H{:}ARG2L \\
&= (ARG1H \bullet ARG2H \bullet 2^{16}) + \\
&\quad (ARG1H \bullet ARG2L \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2H \bullet 2^{8}) + \\
&\quad (ARG1L \bullet ARG2L) + \\
&\quad (-1 \bullet ARG2H\langle 7\rangle \bullet ARG1H{:}ARG1L \bullet 2^{16}) + \\
&\quad (-1 \bullet ARG1H\langle 7\rangle \bullet ARG2H{:}ARG2L \bullet 2^{16})
\end{aligned}
$$

EXAMPLE 12-4: 16 x 16 SIGNED MULTIPLY ROUTINE

```
    MOVF    ARG1L, W
    MULWF   ARG2L               ; ARG1L * ARG2L ->
                                ; PRODH:PRODL
    MOVFF   PRODH, RES1         ;
    MOVFF   PRODL, RES0         ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H               ; ARG1H * ARG2H ->
                                ; PRODH:PRODL
    MOVFF   PRODH, RES3         ;
    MOVFF   PRODL, RES2         ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H               ; ARG1L * ARG2H ->
                                ; PRODH:PRODL
    MOVF    PRODL, W            ;
    ADDWF   RES1, F             ; Add cross
    MOVF    PRODH, W            ; products
    ADDWFC  RES2, F             ;
    CLRF    WREG                ;
    ADDWFC  RES3, F             ;
;
    MOVF    ARG1H, W            ;
    MULWF   ARG2L               ; ARG1H * ARG2L ->
                                ; PRODH:PRODL
    MOVF    PRODL, W            ;
    ADDWF   RES1, F             ; Add cross
    MOVF    PRODH, W            ; products
    ADDWFC  RES2, F             ;
    CLRF    WREG                ;
    ADDWFC  RES3, F             ;
;
    BTFSS   ARG2H, 7            ; ARG2H:ARG2L neg?
    BRA     SIGN_ARG1           ; no, check ARG1
    MOVF    ARG1L, W            ;
    SUBWF   RES2                ;
    MOVF    ARG1H, W            ;
    SUBWFB  RES3
;
SIGN_ARG1
    BTFSS   ARG1H, 7            ; ARG1H:ARG1L neg?
    BRA     CONT_CODE           ; no, done
    MOVF    ARG2L, W            ;
    SUBWF   RES2                ;
    MOVF    ARG2H, W            ;
    SUBWFB  RES3
;
CONT_CODE
    :
```

14.2 CRC Functional Overview

The CRC module can be used to detect bit errors in the program memory using the built-in memory scanner or through user input RAM memory. The CRC module can accept up to a 16-bit polynomial with up to a 16-bit seed value. A CRC calculated check value (or checksum) will then be generated into the CRCACC<15:0> registers for user storage. The CRC module uses an XOR shift register implementation to perform the polynomial division required for the CRC calculation.

EXAMPLE 14-1: CRC EXAMPLE

Rev. 10-000206A
1/8/2014

CRC-16-ANSI

$x^{16} + x^{15} + x^2 + 1$ (17 bits)

Standard 16-bit representation = 0x8005

```
CRCXORH = 0b10000000
CRCXORL = 0b0000010- (1)
```

Data Sequence:
0x55, 0x66, 0x77, 0x88

```
DLEN = 0b0111
PLEN = 0b1111
```

Data entered into the CRC:
SHIFTM = 0:

```
01010101 01100110 01110111 10001000
```

SHIFTM = 1:

```
10101010 01100110 11101110 00010001
```

Check Value (ACCM = 1):

SHIFTM = 0: 0x32D6

```
CRCACCH = 0b00110010
CRCACCL = 0b11010110
```

SHIFTM = 1: 0x6BA2

```
CRCACCH = 0b01101011
CRCACCL = 0b10100010
```

Note 1: Bit 0 is unimplemented. The LSb of any CRC polynomial is always ‘1’ and will always be treated as a ‘1’ by the CRC for calculating the CRC check value. This bit will be read in software as a ‘0’.

TABLE 17-1: PPS INPUT REGISTER DETAILS

Peripheral	PPS Input Register	Default Pin Selection at POR	Register Reset Value at POR	Input Available from Selected PORTx													
				PIC18(L)F26/27K42			PIC18(L)F45/46/47K42					PIC18(L)F55/56/57K42					
Interrupt 0	INT0PPS	RB0	0b0 1000	A	B	—	A	B	—	—	—	A	B	—	—	—	—
Interrupt 1	INT1PPS	RB1	0b0 1001	A	B	—	A	B	—	—	—	—	B	—	D	—	—
Interrupt 2	INT2PPS	RB2	0b0 1010	A	B	—	A	B	—	—	—	—	B	—	—	—	F
Timer0 Clock	T0CKIPPS	RA4	0b0 0100	A	B	—	A	B	—	—	—	A	—	—	—	—	F
Timer1 Clock	T1CKIPPS	RC0	0b1 0000	A	—	C	A	—	C	—	—	—	—	C	—	E	—
Timer1 Gate	T1GPPS	RB5	0b0 1101	—	B	C	—	B	C	—	—	—	B	C	—	—	—
Timer3 Clock	T3CKIPPS	RC0	0b1 0000	—	B	C	—	B	C	—	—	-	—	C	—	E	—
Timer3 Gate	T3GPPS	RC0	0b1 0000	A	—	C	A	—	C	—	—	A	—	C	—	—	—
Timer5 Clock	T5CKIPPS	RC2	0b1 0010	A	—	C	A	—	C	—	—	—	—	C	—	E	—
Timer5 Gate	T5GPPS	RB4	0b0 1100	—	B	C	—	B	—	D	—	—	B	—	D	—	—
Timer2 Clock	T2INPPS	RC3	0b1 0011	A	—	C	A	—	C	—	—	A	—	C	—	—	—
Timer4 Clock	T4INPPS	RC5	0b1 0101	—	B	C	—	B	C	—	—	—	B	C	—	—	—
Timer6 Clock	T6INPPS	RB7	0b0 1111	—	B	C	—	B	—	D	—	—	B	—	D	—	—
CCP1	CCP1PPS	RC2	0b1 0010	—	B	C	—	B	C	—	—	—	—	C	—	—	F
CCP2	CCP2PPS	RC1	0b1 0001	—	B	C	—	B	C	—	—	—	—	C	—	—	F
CCP3	CCP3PPS	RB5	0b0 1101	—	B	C	—	B	—	D	—	—	B	—	D	—	—
CCP4	CCP4PPS	RB0	0b0 1000	—	B	C	—	B	—	D	—	—	B	—	D	—	—
SMT1 Window	SMT1WINPPS	RC0	0b1 0000	—	B	C	—	B	C	—	—	—	—	C	—	—	F
SMT1 Signal	SMT1SIGPPS	RC1	0b1 0001	—	B	C	—	B	C	—	—	—	—	C	—	—	F
CWG1	CWG1PPS	RB0	0b0 1000	—	B	C	—	B	—	D	—	—	B	—	D	—	—
CWG2	CWG2PPS	RB1	0b0 1001	—	B	C	—	B	—	D	—	—	B	—	D	—	—
CWG3	CWG3PPS	RB2	0b0 1010	—	B	C	—	B	—	D	—	—	B	—	D	—	—
DSM1 Carrier Low	MD1CARLPPS	RA3	0b0 0011	A	—	C	A	—	—	D	—	A	—	—	D	—	—
DSM1 Carrier High	MD1CARHPPS	RA4	0b0 0100	A	—	C	A	—	—	D	—	A	—	—	D	—	—
DSM1 Source	MD1SRCPPS	RA5	0b0 0101	A	—	C	A	—	—	D	—	A	—	—	D	—	—
CLCx Input 1	CLCIN0PPS	RA0	0b0 0000	A	—	C	A	—	C	—	—	A	—	C	—	—	—
CLCx Input 2	CLCIN1PPS	RA1	0b0 0001	A	—	C	A	—	C	—	—	A	—	C	—	—	—
CLCx Input 3	CLCIN2PPS	RB6	0b0 1110	—	B	C	—	B	—	D	—	—	B	—	D	—	—
CLCx Input 4	CLCIN3PPS	RB7	0b0 1111	—	B	C	—	B	—	D	—	—	B	—	D	—	—

TABLE 18-1: IOC REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IOCAP	IOCAP7	IOCAP6	IOCAP5	IOCAP4	IOCAP3	IOCAP2	IOCAP1	IOCAP0
IOCAN	IOCAN7	IOCAN6	IOCAN5	IOCAN4	IOCAN3	IOCAN2	IOCAN1	IOCAN0
IOCAF	IOCAF7	IOCAF6	IOCAF5	IOCAF4	IOCAF3	IOCAF2	IOCAF1	IOCAF0
IOCBP	IOCBP7	IOCBP6	IOCBP5	IOCBP4	IOCBP3	IOCBP2	IOCBP1	IOCBP0
IOCBN	IOCBN7	IOCBN6	IOCBN5	IOCBN4	IOCBN3	IOCBN2	IOCBN1	IOCBN0
IOCBF	IOCBF7	IOCBF6	IOCBF5	IOCBF4	IOCBF3	IOCBF2	IOCBF1	IOCBF0
IOCCP	IOCCP7	IOCCP6	IOCCP5	IOCCP4	IOCCP3	IOCCP2	IOCCP1	IOCCP0
IOCCN	IOCCN7	IOCCN6	IOCCN5	IOCCN4	IOCCN3	IOCCN2	IOCCN1	IOCCN0
IOCCF	IOCCF7	IOCCF6	IOCCF5	IOCCF4	IOCCF3	IOCCF2	IOCCF1	IOCCF0
IOCEP	—	—	—	—	IOCEP3(1)	—	—	—
IOCEN	—	—	—	—	IOCEN3(1)	—	—	—
IOCEF	—	—	—	—	IOCEF3(1)	—	—	—

Note 1: If MCLRE = 1 or LVP = 1, RE3 port functionality is disabled and IOC on RE3 is not available.

TABLE 18-2: SUMMARY OF REGISTERS ASSOCIATED WITH INTERRUPT-ON-CHANGE

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
IOCxF	IOCxF7	IOCxF6	IOCxF5	IOCxF4	IOCxF3	IOCxF2	IOCxF1	IOCxF0	287
IOCxN	IOCxN7	IOCxN6	IOCxN5	IOCxN4	IOCxN3	IOCxN2	IOCxN1	IOCxN0	287
IOCxP	IOCxP7	IOCxP6	IOCxP5	IOCxP4	IOCxP3	IOCxP2	IOCxP1	IOCxP0	287

Legend: — = unimplemented location, read as ‘0’. Shaded cells are not used by interrupt-on-change.

20.8 Register Definitions: Timer0 Control

REGISTER 20-1: T0CON0: TIMER0 CONTROL REGISTER 0

R/W-0/0	U-0	R-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
EN	—	OUT	MD16	OUTPS<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 **EN:** TMR0 Enable bit
`1` = The module is enabled and operating
`0` = The module is disabled and in the lowest power mode

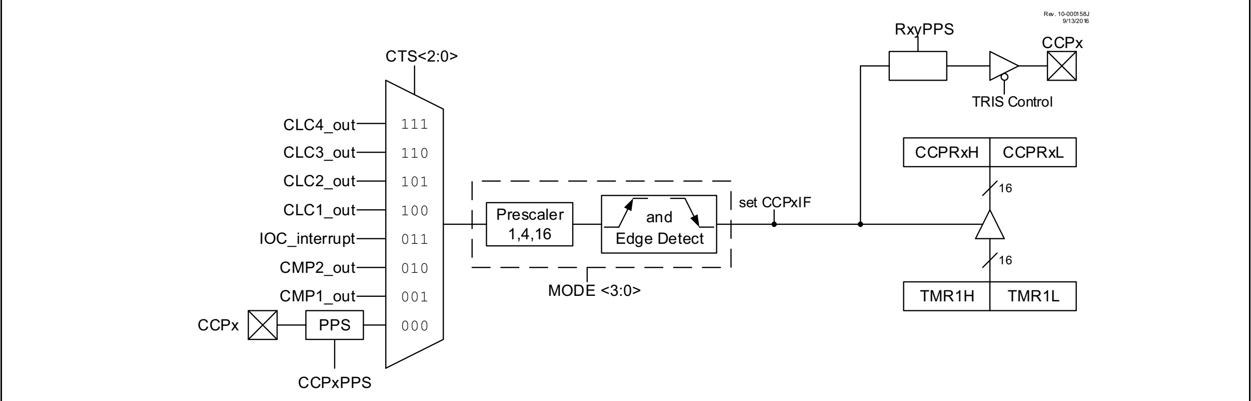
bit 6 **Unimplemented:** Read as ‘`0`’

bit 5 **OUT:** TMR0 Output bit (read-only)
TMR0 output bit

bit 4 **MD16:** TMR0 Operating as 16-Bit Timer Select bit
`1` = TMR0 is a 16-bit timer
`0` = TMR0 is an 8-bit timer

bit 3-0 **OUTPS<3:0>:** TMR0 Output Postscaler (Divider) Select bits
`1111` = 1:16 Postscaler
`1110` = 1:15 Postscaler
`1101` = 1:14 Postscaler
`1100` = 1:13 Postscaler
`1011` = 1:12 Postscaler
`1010` = 1:11 Postscaler
`1001` = 1:10 Postscaler
`1000` = 1:9 Postscaler
`0111` = 1:8 Postscaler
`0110` = 1:7 Postscaler
`0101` = 1:6 Postscaler
`0100` = 1:5 Postscaler
`0011` = 1:4 Postscaler
`0010` = 1:3 Postscaler
`0001` = 1:2 Postscaler
`0000` = 1:1 Postscaler

FIGURE 23-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



REGISTER 23-2: CCPTMRS0: CCP TIMERS CONTROL REGISTER 0

R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1
C4TSEL<1:0>		C3TSEL<1:0>		C2TSEL<1:0>		C1TSEL<1:0>	
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-6 **C4TSEL<1:0>:** CCP4 Timer Selection bits
`11` = CCP4 is based off Timer5 in Capture/Compare mode and Timer6 in PWM mode
`10` = CCP4 is based off Timer3 in Capture/Compare mode and Timer4 in PWM mode
`01` = CCP4 is based off Timer1 in Capture/Compare mode and Timer2 in PWM mode
`00` = Reserved

bit 5-4 **C3TSEL<1:0>:** CCP3 Timer Selection bits
`11` = CCP3 is based off Timer5 in Capture/Compare mode and Timer6 in PWM mode
`10` = CCP3 is based off Timer3 in Capture/Compare mode and Timer4 in PWM mode
`01` = CCP3 is based off Timer1 in Capture/Compare mode and Timer2 in PWM mode
`00` = Reserved

bit 3-2 **C2TSEL<1:0>:** CCP2 Timer Selection bits
`11` = CCP2 is based off Timer5 in Capture/Compare mode and Timer6 in PWM mode
`10` = CCP2 is based off Timer3 in Capture/Compare mode and Timer4 in PWM mode
`01` = CCP2 is based off Timer1 in Capture/Compare mode and Timer2 in PWM mode
`00` = Reserved

bit 1-0 **C1TSEL<1:0>:** CCP1 Timer Selection bits
`11` = CCP1 is based off Timer5 in Capture/Compare mode and Timer6 in PWM mode
`10` = CCP1 is based off Timer3 in Capture/Compare mode and Timer4 in PWM mode
`01` = CCP1 is based off Timer1 in Capture/Compare mode and Timer2 in PWM mode
`00` = Reserved

FIGURE 25-3: TIMER MODE TIMING DIAGRAM

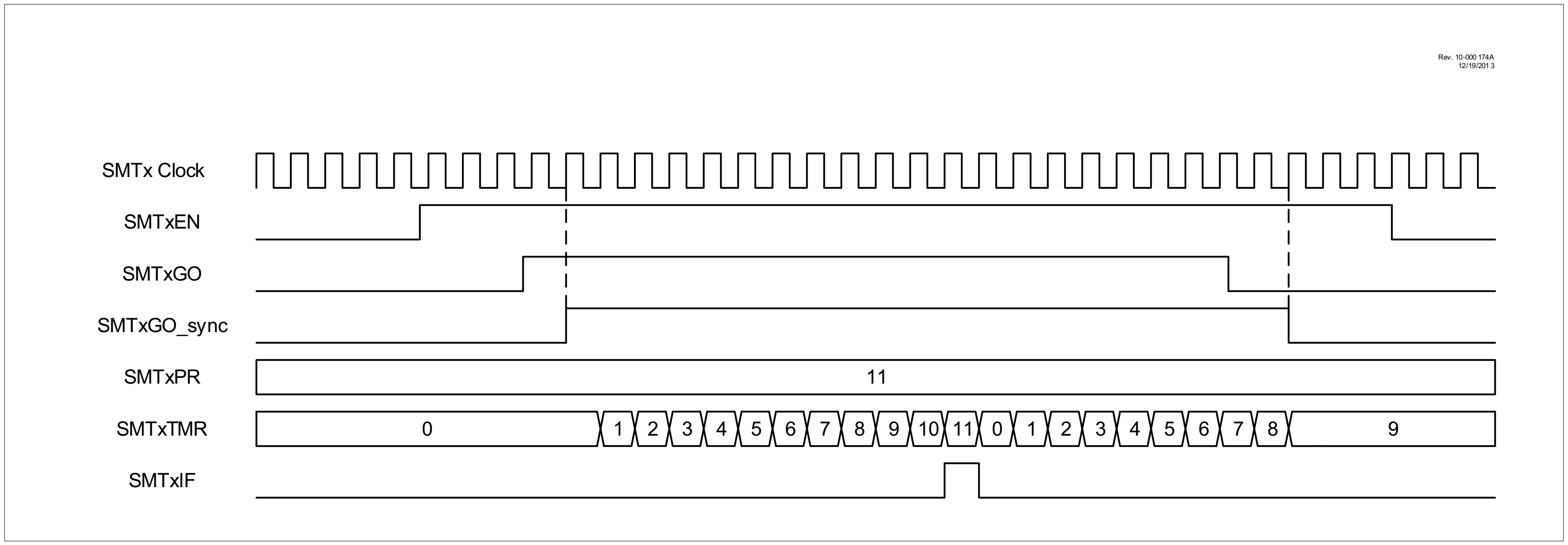


FIGURE 25-19: GATED COUNTER MODE REPEAT ACQUISITION TIMING DIAGRAM

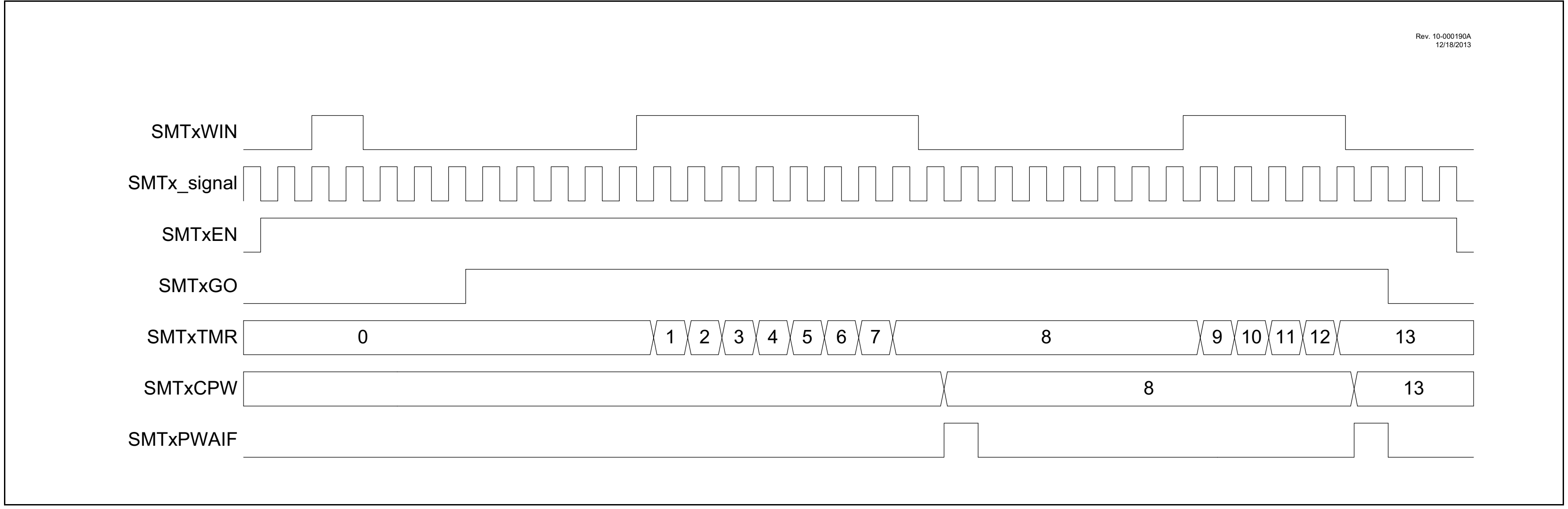
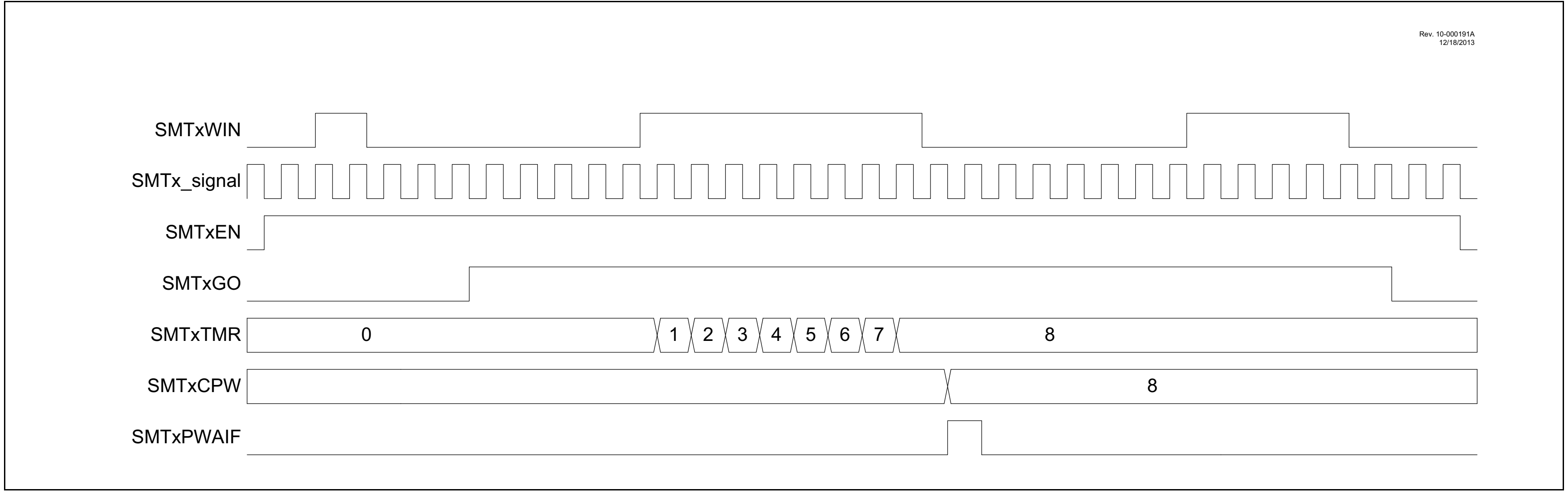


FIGURE 25-20: GATED COUNTER MODE SINGLE ACQUISITION TIMING DIAGRAM



29.0 ZERO-CROSS DETECTION (ZCD) MODULE

The ZCD module detects when an A/C signal crosses through the ground potential. The actual zero-crossing threshold is the zero-crossing reference voltage, VCPINV, which is typically 0.75V above ground.

The connection to the signal to be detected is through a series current-limiting resistor. The module applies a current source or sink to the ZCD pin to maintain a constant voltage on the pin, thereby preventing the pin voltage from forward biasing the ESD protection diodes. When the applied voltage is greater than the reference voltage, the module sinks current. When the applied voltage is less than the reference voltage, the module sources current. The current source and sink action keeps the pin voltage constant over the full range of the applied voltage. The ZCD module is shown in the simplified block diagram Figure 29-2.

The ZCD module is useful when monitoring an A/C waveform for, but not limited to, the following purposes:

- A/C period measurement
- Accurate long term time measurement
- Dimmer phase delayed drive
- Low EMI cycle switching

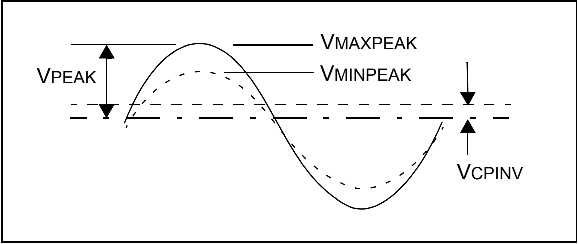
29.1 External Resistor Selection

The ZCD module requires a current-limiting resistor in series with the external voltage source. The impedance and rating of this resistor depends on the external source peak voltage. Select a resistor value that will drop all of the peak voltage when the current through the resistor is nominally 300 µA. Refer to Equation 29-1 and Figure 29-1. Make sure that the ZCD I/O pin internal weak pull-up is disabled so it does not interfere with the current source and sink.

EQUATION 29-1: EXTERNAL RESISTOR

$$R_{SERIES} = \frac{V_{PEAK}}{3 \times 10^{-4}}$$

FIGURE 29-1: EXTERNAL VOLTAGE



REGISTER 31-4: UxERRIR: UART ERROR INTERRUPT FLAG REGISTER

R/S/C-1/1	R/S/C-0/0	R/W/S-0/0	R/W/S-0/0	R/S/C-0/0	R/W/S-0/0	R/W/S-0/0	R/W/S-0/0
TXMTIF	PERIF	ABDOVF	CERIF	FERIF	RXBKIF	RXFOIF	TXCIF
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	S = Hardware set	C = Hardware clear

bit 7 **TXMTIF:** Transmit Shift Register Empty Interrupt Flag bit
1 = Transmit shift register is empty (Set at end of Stop bits)
0 = Transmit shift register is actively shifting data

bit 6 **PERIF**: Parity Error Interrupt Flag bit
LIN and Parity modes:
1 = Unread byte at top of input FIFO has parity error
0 = Unread byte at top of input FIFO does not have parity error
DALI Device mode:
1 = Unread byte at top of input FIFO received as Forward Frame
0 = Unread byte at top of input FIFO received as Back Frame
Address mode:
1 = Unread byte at top of input FIFO received as address
0 = Unread byte at top of input FIFO received as data
Other modes:
Not used

bit 5 **ABDOVF:** Auto-baud Detect Overflow Interrupt Flag bit
DALI mode:
1 = Start bit measurement overflowed counter
0 = No overflow during Start bit measurement
Other modes:
1 = Baud rate generator overflowed during the auto detection sequence
0 = Baud rate generator has not overflowed

bit 4 **CERIF**: Checksum Error Interrupt Flag bit (LIN mode only)
1 = Checksum error
0 = No checksum error

bit 3 **FERIF:** Framing Error Interrupt Flag bit
1 = Unread byte at top of input FIFO has framing error
0 = Unread byte at top of input FIFO does not have framing error

bit 2 **RXBKIF:** Break Reception Interrupt Flag bit
1 = Break detected
0 = No Break detected

bit 1 **RXFOIF:** Receive FIFO Overflow Interrupt Flag bit
1 = Receive FIFO has overflowed
0 = Receive FIFO has not overflowed

bit 0 **TXCIF:** Transmit Collision Interrupt Flag bit
1 = Transmitted word is not equal to the word received during transmission
0 = Transmitted word equals the word received during transmission

REGISTER 36-18: ADRESH: ADC RESULT REGISTER HIGH, FM = 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	R/W-x/u	R/W-x/u
ADRES<11:4>							
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	**ADRES<11:4>**: ADC Result Register bits Upper eight bits of 12-bit conversion result.

REGISTER 36-19: ADRESL: ADC RESULT REGISTER LOW, FM = 0

R/W-x/u	R/W-x/u	R/W-x/u	R/W-x/u	U-0	U-0	U-0	U-0
ADRES<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	**ADRES<3:0>**: ADC Result Register bits. Lower four bits of 12-bit conversion result.
bit 3-0	**Reserved**

41.0 INSTRUCTION SET SUMMARY

PIC18(L)F26/27/45/46/47/55/56/57K42 devices incorporate the standard set of PIC18 core instructions, as well as an extended set of instructions, for the optimization of code that is recursive or that utilizes a software stack. The extended set is discussed later in this section.

41.1 Standard Instruction Set

The standard PIC18 instruction set adds many enhancements to the previous PIC® MCU instruction sets, while maintaining an easy migration from these PIC® MCU instruction sets. Most instructions are a single program memory word (16 bits), but there are four instructions that require two-program memory locations and two that require three-program memory locations.

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

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

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

The PIC18 instruction set summary in Table 41-3 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 41-1 shows the opcode field descriptions.

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

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

The file register designator ‘f’ specifies which file register is to be used by the instruction. The destination designator ‘d’ specifies where the result of the operation is to be placed. If ‘d’ is zero, the result is placed in the WREG register. If ‘d’ is one, the result is placed in the file register specified in the instruction.

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

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

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

The literal instructions may use some of the following operands:

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

The control instructions may use some of the following operands:

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

All instructions are a single word, except for four double-word instructions. These instructions were made double-word to contain the required information in 32 bits. In the second word, the four MSbs are ‘`1`’s. If this second word is executed as an instruction (by itself), it will execute as a `NOP`.

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

The double-word instructions execute in two instruction cycles.

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

Figure 41-1 shows the general formats that the instructions can have. All examples use the convention ‘nnh’ to represent a hexadecimal number.

The Instruction Set Summary, shown in Table 41-3, lists the standard instructions recognized by the Microchip Assembler (MPASM™).

Section 41.1.1 “Standard Instruction Set” provides a description of each instruction.

TABLE 41-2: INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected	Notes
				MSb			LSb		
BYTE-ORIENTED FILE REGISTER INSTRUCTIONS									
ADDWF	f, d ,a	Add WREG and f	1	0010	01da	ffff	ffff	C, DC, Z, OV, N	
ADDWFC	f, d, a	Add WREG and Carry bit to f	1	0010	00da	ffff	ffff	C, DC, Z, OV, N	
ANDWF	f, d, a	AND WREG with f	1	0001	01da	ffff	ffff	Z, N	
CLRF	f, a	Clear f	1	0110	101a	ffff	ffff	Z	
COMF	f, d, a	Complement f	1	0001	11da	ffff	ffff	Z, N	
DECF	f, d, a	Decrement f	1	0000	01da	ffff	ffff	C, DC, Z, OV, N	
INCF	f, d, a	Increment f	1	0010	10da	ffff	ffff	C, DC, Z, OV, N	
IORWF	f, d, a	Inclusive OR WREG with f	1	0001	00da	ffff	ffff	Z, N	
MOVF	f, d, a	Move f to WREG or f	1	0101	00da	ffff	ffff	Z, N	
MOVFF	f_s, f_d	Move f_s (source) to 1st word	2	1100	ffff	ffff	ffff	None	2, 3
		f_d (destination) 2nd word		1111	ffff	ffff	ffff		
MOVFFL	f_s, f_d	Move f_s (source) to	3	0000	0000	0110	ffff	None	2
		g (full destination)		1111	ffff	ffff	ffgg		
		f_d (full destination)3rd word		1111	gggg	gggg	gggg		
MOVWF	f, a	Move WREG to f	1	0110	111a	ffff	ffff	None	
MULWF	f, a	Multiply WREG with f	1	0000	001a	ffff	ffff	None	
NEGF	f, a	Negate f	1	0110	110a	ffff	ffff	C, DC, Z, OV, N	
RLCF	f, d, a	Rotate Left f through Carry	1	0011	01da	ffff	ffff	C, Z, N	
RLNCF	f, d, a	Rotate Left f (No Carry)	1	0100	01da	ffff	ffff	Z, N	
RRCF	f, d, a	Rotate Right f through Carry	1	0011	00da	ffff	ffff	C, Z, N	
RRNCF	f, d, a	Rotate Right f (No Carry)	1	0100	00da	ffff	ffff	Z, N	
SETF	f, a	Set f	1	0110	100a	ffff	ffff	None	
SUBFWB	f, d, a	Subtract f from WREG with borrow	1	0101	01da	ffff	ffff	C, DC, Z, OV, N	
SUBWF	f, d, a	Subtract WREG from f	1	0101	11da	ffff	ffff	C, DC, Z, OV, N	
SUBWFB	f, d, a	Subtract WREG from f with borrow	1	0101	10da	ffff	ffff	C, DC, Z, OV, N	
SWAPF	f, d, a	Swap nibbles in f	1	0011	10da	ffff	ffff	None	
XORWF	f, d, a	Exclusive OR WREG with f	1	0001	10da	ffff	ffff	Z, N	
BYTE-ORIENTED SKIP INSTRUCTIONS									
CPFSEQ	f, a	Compare f with WREG, skip =	1 (2 or 3)	0110	001a	ffff	ffff	None	1
CPFSGT	f, a	Compare f with WREG, skip >	1 (2 or 3)	0110	010a	ffff	ffff	None	1
CPFSLT	f, a	Compare f with WREG, skip <	1 (2 or 3)	0110	000a	ffff	ffff	None	1
DECFSZ	f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010	11da	ffff	ffff	None	1
DCFSNZ	f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100	11da	ffff	ffff	None	1
INCFSZ	f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011	11da	ffff	ffff	None	1
INFSNZ	f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100	10da	ffff	ffff	None	1
TSTFSZ	f, a	Test f, skip if 0	1 (2 or 3)	0110	011a	ffff	ffff	None	1
BIT-ORIENTED FILE REGISTER INSTRUCTIONS									
BCF	f, b, a	Bit Clear f	1	1001	bbba	ffff	ffff	None	
BSF	f, b, a	Bit Set f	1	1000	bbba	ffff	ffff	None	
BTG	f, d, a	Bit Toggle f	1	0111	bbba	ffff	ffff	None	
BIT-ORIENTED SKIP INSTRUCTIONS									
BTFSC	f, b, a	Bit Test f, Skip if Clear	1 (2 or 3)	1011	bbba	ffff	ffff	None	1
BTFSS	f, b, a	Bit Test f, Skip if Set	1 (2 or 3)	1010	bbba	ffff	ffff	None	1

Note 1: If Program Counter (PC) is modified or a conditional test is true, the instruction requires an additional cycle. The extra cycle is executed as a `NOP`.

2: Some instructions are multi word instructions. The second/third words of these instructions will be decoded as a `NOP`, unless the first word of the instruction retrieves the information embedded in these 16-bits. This ensures that all program memory locations have a valid instruction.

3: f_s and f_d do not cover the full memory range. 2 MSBs of bank selection are forced to ‘b00 to limit the range of these instructions to lower 4k addressing space.

TABLE 42-1: REGISTER FILE SUMMARY FOR PIC18(L)F26/27/45/46/47/55/56/57K42 DEVICES

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
3EE8h	ADACCL	ACC								631
3EE7h	ADFLTRH	FLTR								627
3EE6h	ADFLTRL	FLTR								627
3EE5h	ADSTPTH	STPT								632
3EE4h	ADSTPTL	STPT								632
3EE3h	ADERRH	ERR								633
3EE2h	ADERRL	ERR								633
3EE1h	ADUTHH	UTH								634
3EE0h	ADUTHL	UTH								634
3EDFh	ADLTHH	LTH								633
3EDEh	ADLTHL	LTH								634
3EDDh - 3ED8h	—	Unimplemented								
3ED7h	ADCP	ON	—	—	—	—	—	—	CPRDY	636
3ED6h - 3ECBh	—	Unimplemented								
3ECAh	HLVDCON1	—	—	—	—	SEL				658
3EC9h	HLVDCON0	EN	—	OUT	RDY	—	—	INTH	INTL	657
3EC8h - 3EC4h	—	Unimplemented								
3EC3h	ZCDCON	SEN	—	OUT	POL	—	—	INTP	INTN	462
3EC2h	—	Unimplemented								
3EC1h	FVRCON	EN	RDY	TSEN	TSRNG	CDAFVR		ADFVR		597
3EC0h	CMOUT	—	—	—	—	—	—	C2OUT	C1OUT	650
3EBFh	CM1PCH	—	—	—	—	—	PCH			650
3EBEh	CM1NCH	—	—	—	—	—	NCH			649
3EBDh	CM1CON1	—	—	—	—	—	—	INTP	INTN	649
3EBCh	CM1CON0	EN	OUT	—	POL	—	—	HYS	SYNC	648
3EBBh	CM2PCH	—	—	—	—	—	PCH			650
3EBAh	CM2NCH	—	—	—	—	—	NCH			649
3EB9h	CM2CON1	—	—	—	—	—	—	INTP	INTN	649
3EB8h	CM2CON0	EN	OUT	—	POL	—	—	HYS	SYNC	648
3EB7h - 3E9Fh	—	Unimplemented								
3E9Eh	DAC1CON0	EN	—	OE1	OE2	PSS		—	NSS	640
3E9Dh	—	Unimplemented								
3E9Ch	DAC1CON1	—	—	—	DATA					641
3E9Bh - 3DFBh	—	Unimplemented								
3DFAh	U1ERRIE	TXMTIE	PERIE	ABDOVE	CERIE	FERIE	RXBKIE	RXFOIE	TXCIE	502
3DF9h	U1ERRIR	TXMTIF	PERIF	ABDOVF	CERIF	FERIF	RXBKIF	RXFOIF	TXCIF	501
3DF8h	U1UIR	WUIF	ABDIF	—	—	—	ABDIE	—	—	503
3DF7h	U1FIFO	TXWRE	STPMD	TXBE	TXBF	RXIDL	XON	RXBE	RXBF	504
3DF6h	U1BRGH	BRGH								505
3DF5h	U1BRGL	BRGL								505
3DF4h	U1CON2	RUNOVF	RXPOL	STP		C0EN	TXPOL	FLO		500
3DF3h	U1CON1	ON	—	—	WUE	RXBIMD	—	BRKOVR	SENDB	499
3DF2h	U1CON0	BRGS	ABDEN	TXEN	RXEN	MODE				498
3DF1h	U1P3H	—	—	—	—	—	—	—	P3H	509

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition
Note 1: Unimplemented in LF devices.
2: Unimplemented in PIC18(L)F26/27K42.
3: Unimplemented on PIC18(L)F26/27/45/46/47K42 devices.
4: Unimplemented in PIC18(L)F45/55K42.

APPENDIX A: REVISION HISTORY

Revision A (6/2017)

Initial release of the document.

Revision B (12/2017)

Standard operating conditions updated in Section 44.0, Electrical Specifications. Other minor corrections.