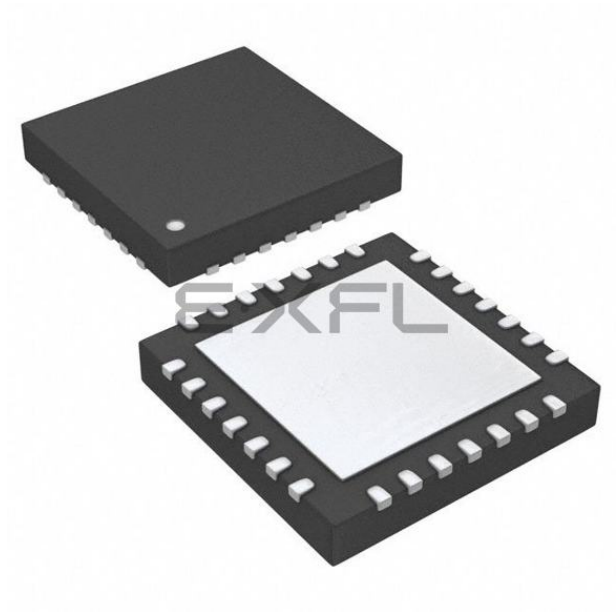


Welcome to E-XFL.COM

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	16KB (8K x 16)
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 24x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°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/pic18lf24k42-e-mv

4.4.3 INSTRUCTIONS IN PROGRAM MEMORY

The program memory is addressed in bytes. Instructions are stored as either two bytes or four bytes in program memory. The Least Significant Byte of an instruction word is always stored in a program memory location with an even address (LSb = 0). To maintain alignment with instruction boundaries, the PC increments in steps of two and the LSb will always read '0' (see **Section 4.2.4 "Program Counter"**).

Figure 4-3 shows an example of how instruction words are stored in the program memory.

The CALL and GOTO instructions have the absolute program memory address embedded into the instruction. Since instructions are always stored on word boundaries, the data contained in the instruction is a word address. The word address is written to PC<20:1>, which accesses the desired byte address in program memory. Instruction #2 in Figure 4-3 shows how the instruction GOTO 0006h is encoded in the program memory. Program branch instructions, which encode a relative address offset, operate in the same manner. The offset value stored in a branch instruction represents the number of single-word instructions that the PC will be offset by. **Section 43.0 "Instruction Set Summary"** provides further details of the instruction set.

4.4.4 MULTI-WORD INSTRUCTIONS

The standard PIC18 instruction set has four two-word instructions: CALL, MOVFF, GOTO and LFSR and two three-word instructions: MOVFL and MOVSFL. In all cases, the second and the third word of the instruction always has '1111' as its four Most Significant bits; the other 12 bits are literal data, usually a data memory address.

The use of '1111' in the 4 MSbs of an instruction specifies a special form of NOP. If the instruction is executed in proper sequence – immediately after the first word – the data in the second word is accessed and used by the instruction sequence. If the first word is skipped for some reason and the second word is executed by itself, a NOP is executed instead. This is necessary for cases when the two-word instruction is preceded by a conditional instruction that changes the PC. Example 4-4 shows how this works.

FIGURE 4-3: INSTRUCTIONS IN PROGRAM MEMORY

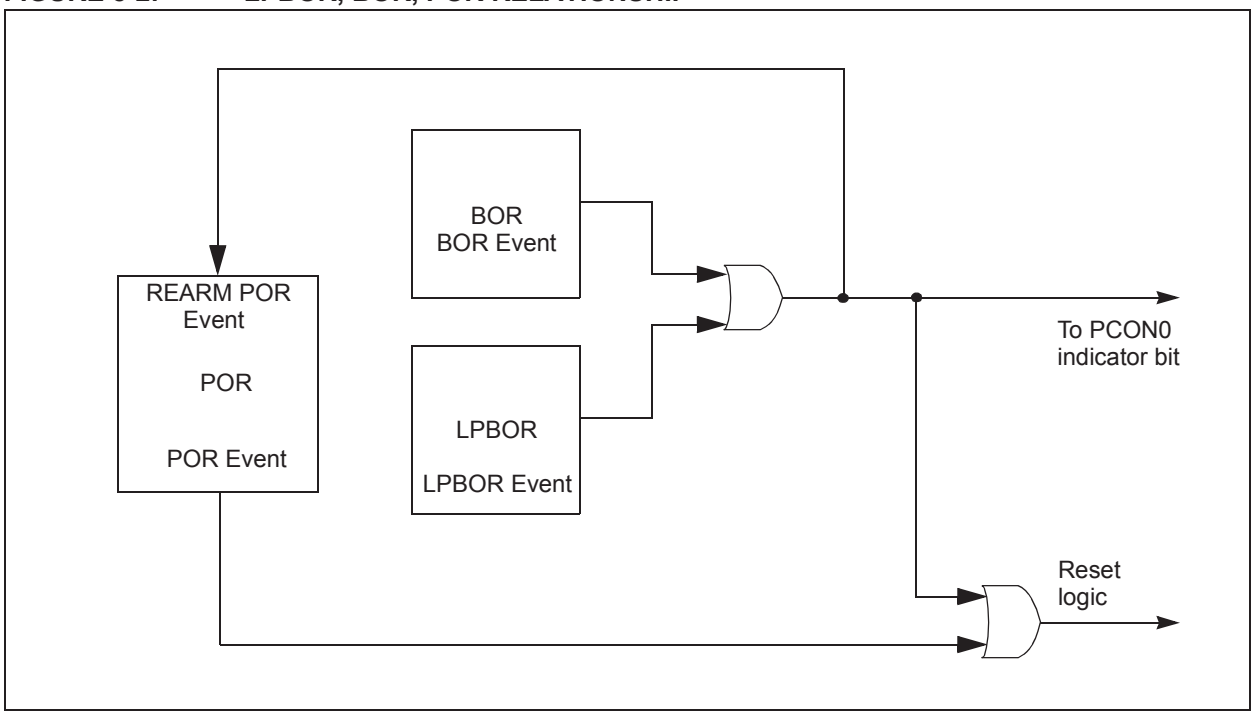
		LSB = 1	LSB = 0	Word Address ↓
	Program Memory Byte Locations →			000000h
				000002h
				000004h
				000006h
Instruction 1:	MOVLW 055h	0Fh	55h	000008h
Instruction 2:	GOTO 0006h	EFh	03h	00000Ah
		F0h	00h	00000Ch
Instruction 3:	MOVFF 123h, 456h	C1h	23h	00000Eh
		F4h	56h	000010h
Instruction 4:	MOVFFL 123h, 456h	00h	60h	000012h
		F4h	8Ch	000014h
		F4h	56h	000016h
				000018h
				00001Ah

TABLE 4-11: REGISTER FILE SUMMARY FOR PIC18(L)F24/25K42 DEVICES (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR
3FCEh	PORTE	—	—	—	—	RE3	—	—	—	----x---
3FCDh	—	Unimplemented								—
3FCCh	PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0	xxxxxxxx
3FCBh	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	xxxxxxxx
3FCAh	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	xxxxxxxx
3FC9h - 3FC5h	—	Unimplemented								—
3FC4h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	TRISC1	TRISC0	11111111
3FC3h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0	11111111
3FC2h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	11111111
3FC1h - 3FBDh	—	Unimplemented								—
3FBCh	LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	LATC1	LATC0	xxxxxxxx
3FBBh	LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0	xxxxxxxx
3FBAh	LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0	xxxxxxxx
3FB9h	T0CON1	CS<2:0>			ASYNC	CKPS<3:0>				00000000
3FB8h	T0CON0	EN	—	OUT	MD16	OUTPS				0-000000
3FB7h	TMR0H	TMR0H								11111111
3FB6h	TMR0L	TMR0L								00000000
3FB5h	T1CLK	CS								---00000
3FB4h	T1GATE	GSS								---00000
3FB3h	T1GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	00000x--
3FB2h	T1CON	—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON	--00-000
3FB1h	TMR1H	TMR1H								00000000
3FB0h	TMR1L	TMR1L								00000000
3FAFh	T2RST	—	—	—	RSEL					---00000
3FAEh	T2CLK	—	—	—	—	CS				----0000
3FADh	T2HLT	PSYNC	CKPOL	CKSYNC	MODE					00000000
3FACh	T2CON	ON	CKPS			OUTPS				00000000
3FABh	T2PR	PR2								11111111
3FAAh	T2TMR	TMR2								00000000
3FA9h	T3CLK	CS								---00000
3FA8h	T3GATE	GSS								---00000
3FA7h	T3GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	00000x--
3FA6h	T3CON	—	—	CKPS		—	NOT_SYNC	RD16	ON	--00-000
3FA5h	TMR3H	TMR3H								00000000
3FA4h	TMR3L	TMR3L								00000000
3FA3h	T4RST	—	—	—	RSEL					---00000
3FA2h	T4CLK	—	—	—	—	CS				----0000
3FA1h	T4HLT	PSYNC	CKPOL	CKSYNC	MODE					00000000
3FA0h	T4CON	ON	CKPS			OUTPS				00000000
3F9Fh	T4PR	PR4								11111111
3F9Eh	T4TMR	TMR4								00000000
3F9Dh	T5CLK	CS								---00000
3F9Ch	T5GATE	GSS								---00000
3F9Bh	T5GCON	GE	GPOL	GTM	GSPM	GGO	GVAL	—	—	00000x--
3F9Ah	T5CON	—	—	CKPS		—	NOT_SYNC	RD16	ON	--00-000
3F99h	TMR5H	TMR5H								00000000
3F98h	TMR5L	TMR5L								00000000

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition
Note 1: Not present in LF devices.

FIGURE 8-2: LPBOR, BOR, POR RELATIONSHIP



11.4.3 PREEMPTING LOW PRIORITY INTERRUPTS

Low-priority interrupts can be preempted by high priority interrupts. While in the low priority ISR, if a high-priority interrupt arrives, the high priority interrupt request is generated and the low priority ISR is suspended, while the high priority ISR is executed, see Figure 11-4.

After the high priority ISR is complete and if any other high priority interrupt requests are not active, the execution returns to the preempted low priority ISR.

Note 1: The high priority interrupt flag must be cleared to avoid recursive interrupts.

2: If a low-priority ISR was already serviced halfway before moving on to a high priority ISR, then the low priority ISR is completely serviced even if user code clears GIEL.

FIGURE 11-4: INTERRUPT EXECUTION: HIGH PRIORITY INTERRUPT PREEMPTING LOW PRIORITY INTERRUPTS

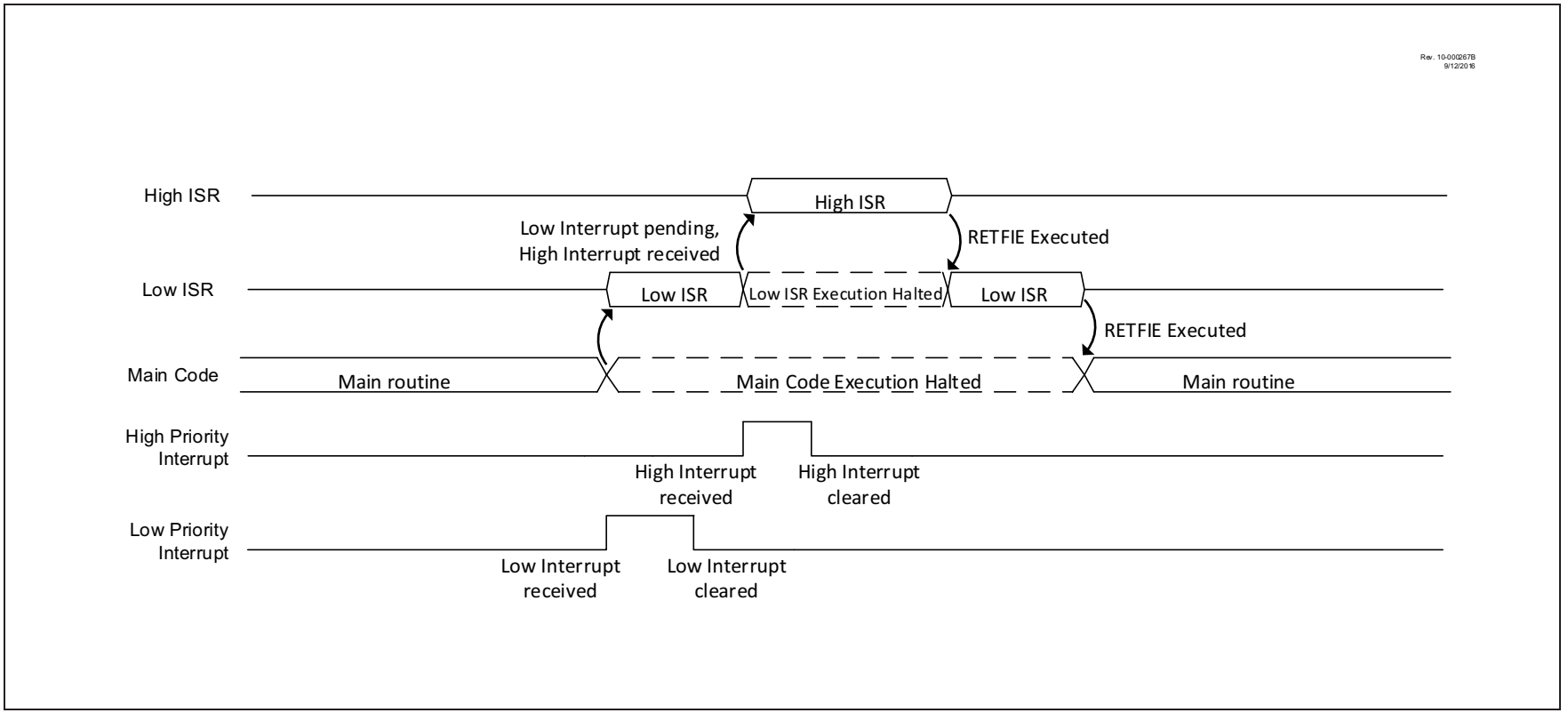
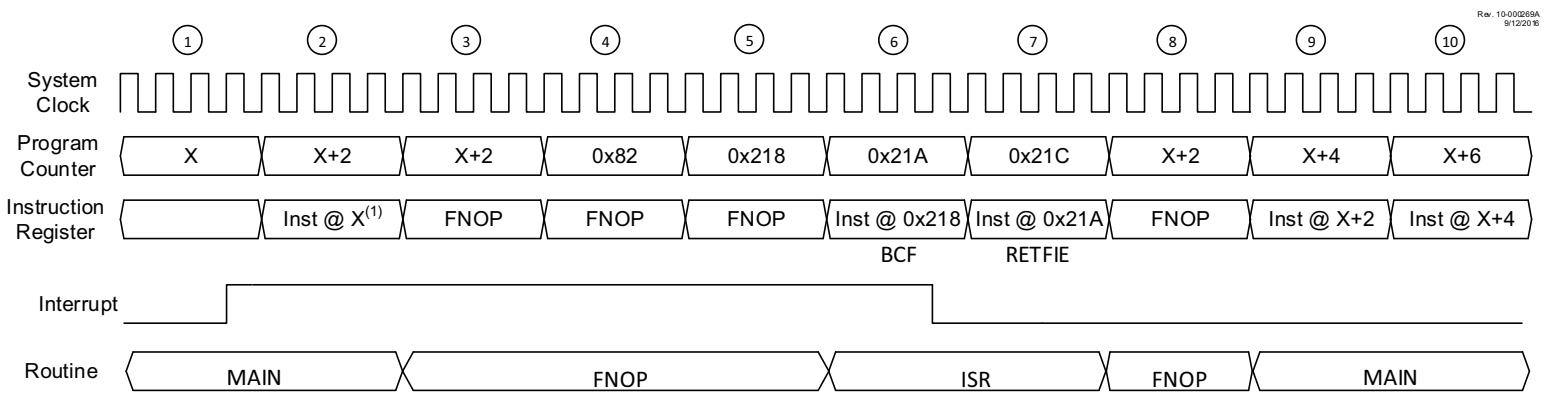
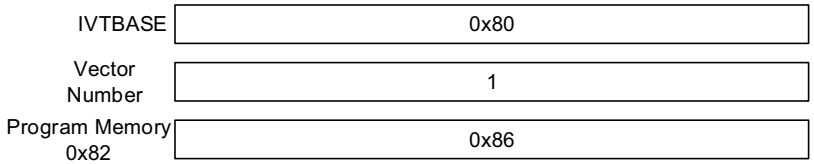


FIGURE 11-7: INTERRUPT TIMING DIAGRAM - ONE CYCLE INSTRUCTION





Interrupt Location = Interrupt vector table entry << 2
= 0x86 << 2 = 0x218

Note 1: Instruction @ X is a One-cycle Instruction

REGISTER 11-3: PIR0: PERIPHERAL INTERRUPT Request Register 0

R-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0	R/W/HS-0/0
IOCIF	CRCIF	SCANIF	NVMIF	CSWIF	OSFIF	HLVDIF	SWIF
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	HS = Bit is set in hardware

bit 7 **IOCIF:** Interrupt-on-Change Interrupt Flag bit
- 1 = Interrupt has occurred
- 0 = Interrupt event has not occurred

bit 6 **CRCIF:** CRC Interrupt Flag bit
- 1 = Interrupt has occurred (must be cleared by software)
- 0 = Interrupt event has not occurred

bit 5 **SCANIF:** Memory Scanner Interrupt Flag bit
- 1 = Interrupt has occurred (must be cleared by software)
- 0 = Interrupt event has not occurred

bit 4 **NVMIF:** NVM Interrupt Flag bit
- 1 = Interrupt has occurred (must be cleared by software)
- 0 = Interrupt event has not occurred

bit 3 **CSWIF:** Clock Switch Interrupt Flag bit
- 1 = Interrupt has occurred (must be cleared by software)
- 0 = Interrupt event has not occurred

bit 2 **OSFIF:** Oscillator Fail Interrupt Flag bit
- 1 = Interrupt has occurred (must be cleared by software)
- 0 = Interrupt event has not occurred

bit 1 **HLVDIF:** HLVD Interrupt Flag bit
- 1 = Interrupt has occurred (must be cleared by software)
- 0 = Interrupt event has not occurred

bit 0 **SWIF:** Software Interrupt Flag bit
- 1 = Software Interrupt Flag Enable
- 0 = Software Interrupt Flag Disable

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

REGISTER 11-9: PIR6: PERIPHERAL INTERRUPT REGISTER 6

R/W/HS-0/0	R/W/HS-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0	R-0/0
TMR3GIF	TMR3IF	U2IF	U2EIF	U2TXIF	U2RXIF	I2C2EIF	I2C2IF
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	HS = Bit is set in hardware

bit 7 **TMR3GIF:** TMR3 Gate Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 6 **TMR3IF:** TMR3 Interrupt Flag bit
1 = Interrupt has occurred (must be cleared by software)
0 = Interrupt event has not occurred

bit 5 **U2IF:** UART2 Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 4 **U2EIF:** UART2 Framing Error Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 3 **U2TXIF:** UART2 Transmit Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 2 **U2RXIF:** UART2 Receive Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 1 **I2C2EIF:** I^2C2 Error Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

bit 0 **I2C2IF:** I^2C2 Interrupt Flag bit
1 = Interrupt has occurred
0 = Interrupt event has not occurred

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

REGISTER 11-30: IPR5: PERIPHERAL INTERRUPT Priority REGISTER 5

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
I2C2TXIP	I2C2RXIP	DMA2AIP	DMA2ORIP	DMA2DCNTIP	DMA2SCNTIP	C2IP	INT1IP
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 **I2C2TXIP:** I^2C2 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **I2C2RXIP:** I^2C2 Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **DMA2AIP:** DMA2 Abort Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **DMA2ORIP:** DMA2 Overrun Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **DMA2DCNTIP:** DMA2 Destination Count Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **DMA2SCNTIP:** DMA2 Source Count Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **C2IP:** C2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **INT1IP:** External Interrupt 1 Interrupt Priority bit
1 = High priority
0 = Low priority

REGISTER 17-15: DMAxDSAH – DMAx DESTINATION START ADDRESS HIGH 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
DSA<15:8>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n/n = Value at POR and BOR/Value at all other Resets	1 = bit is set	0 = bit is cleared	x = bit is unknown u = bit is unchanged

bit 7-0 **DSA<15:8>:** Destination Start Address bits

REGISTER 17-16: DMAxDPTRL – DMAx DESTINATION POINTER LOW REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
DPTR<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n/n = Value at POR and BOR/Value at all other Resets	1 = bit is set	0 = bit is cleared	x = bit is unknown u = bit is unchanged

bit 7-0 **DPTR<7:0>:** Current Destination Address Pointer

REGISTER 17-17: DMAxDPTRH – DMAx DESTINATION POINTER HIGH REGISTER

R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
DPTR<15:8>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n/n = Value at POR and BOR/Value at all other Resets	1 = bit is set	0 = bit is cleared	x = bit is unknown u = bit is unchanged

bit 7-0 **DPTR<15:8>:** Current Destination Address Pointer

REGISTER 19-2: RxyPPS: PIN Rxy OUTPUT SOURCE SELECTION REGISTER (CONTINUED)

bit 5-0

6'b01 0000	PWM8	A	—	C
6'b00 1111	PWM7	A	—	C
6'b00 1110	PWM6	A	—	C
6'b00 1101	PWM5	A	—	C
6'b00 1100	CCP4	—	B	C
6'b00 1100	CCP3	—	B	C
6'b00 1010	CCP2	—	B	C
6'b00 1001	CCP1	—	B	C
6'b00 1000	CWG1D	—	B	C
6'b00 0111	CWG1C	—	B	C
6'b00 0110	CWG1B	—	B	C
6'b00 0101	CWG1A	—	B	C
6'b00 0100	CLC4OUT	—	B	C
6'b00 0011	CLC3OUT	—	B	C
6'b00 0010	CLC2OUT	A	—	C
6'b00 0001	CLC1OUT	A	—	C
6'b00 0000	LATxy	A	B	C

23.12 Register Definitions: Timer1/3/5

Long bit name prefixes for the Timer1/3/5 are shown in Table 24-2. Refer to **Section 1.4.2.2 "Long Bit Names"** for more information.

TABLE 23-3:

Peripheral	Bit Name Prefix
Timer1	T1
Timer3	T3
Timer5	T5

REGISTER 23-1: TXCON: TIMERx CONTROL REGISTER

U-0	U-0	R/W-0/u	R/W-0/u	U-0	R/W-0/u	R/W-0/0	R/W-0/u
—	—	CKPS<1:0>		—	$\overline{\text{SYNC}}$	RD16	ON
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	u = unchanged

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **CKPS<1:0>:** Timerx 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 **Unimplemented:** Read as '0'

bit 2 **$\overline{\text{SYNC}}$:** Timerx External Clock Input Synchronization Control bit
TMRxCLK = FOSC/4 or FOSC:
This bit is ignored. Timer1 uses the incoming clock as is.
Else:
1 = Do not synchronize external clock input
0 = Synchronize external clock input with system clock

bit 1 **RD16:** 16-Bit Read/Write Mode Enable bit
1 = Enables register read/write of Timerx in one 16-bit operation
0 = Enables register read/write of Timerx in two 8-bit operation

bit 0 **ON:** Timerx On bit
1 = Enables Timerx
0 = Disables Timerx

26.0 PULSE-WIDTH MODULATION (PWM)

The PWM module generates a Pulse-Width Modulated signal determined by the duty cycle, period, and resolution that are configured by the following registers:

- TxPR
- TxCON
- PWMxDCH
- PWMxDCL
- PWMxCON

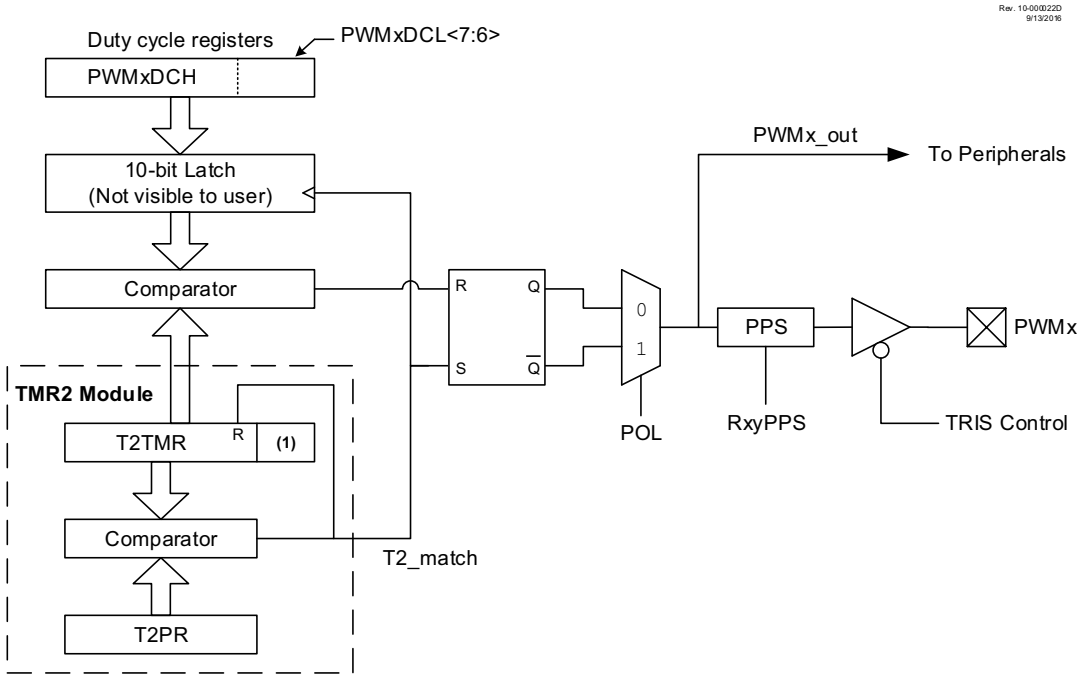
Note: The corresponding TRIS bit must be cleared to enable the PWM output on the PWMx pin.

Each PWM module can select the timer source that controls the module. Each module has an independent timer selection which can be accessed using the CCPTMRS1 register (Register 25-2). Please note that the PWM mode operation is described with respect to T2TMR in the following sections.

Figure 26-1 shows a simplified block diagram of PWM operation.

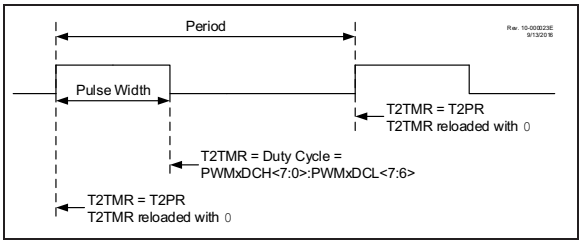
Figure 26-2 shows a typical waveform of the PWM signal.

FIGURE 26-1: SIMPLIFIED PWM BLOCK DIAGRAM



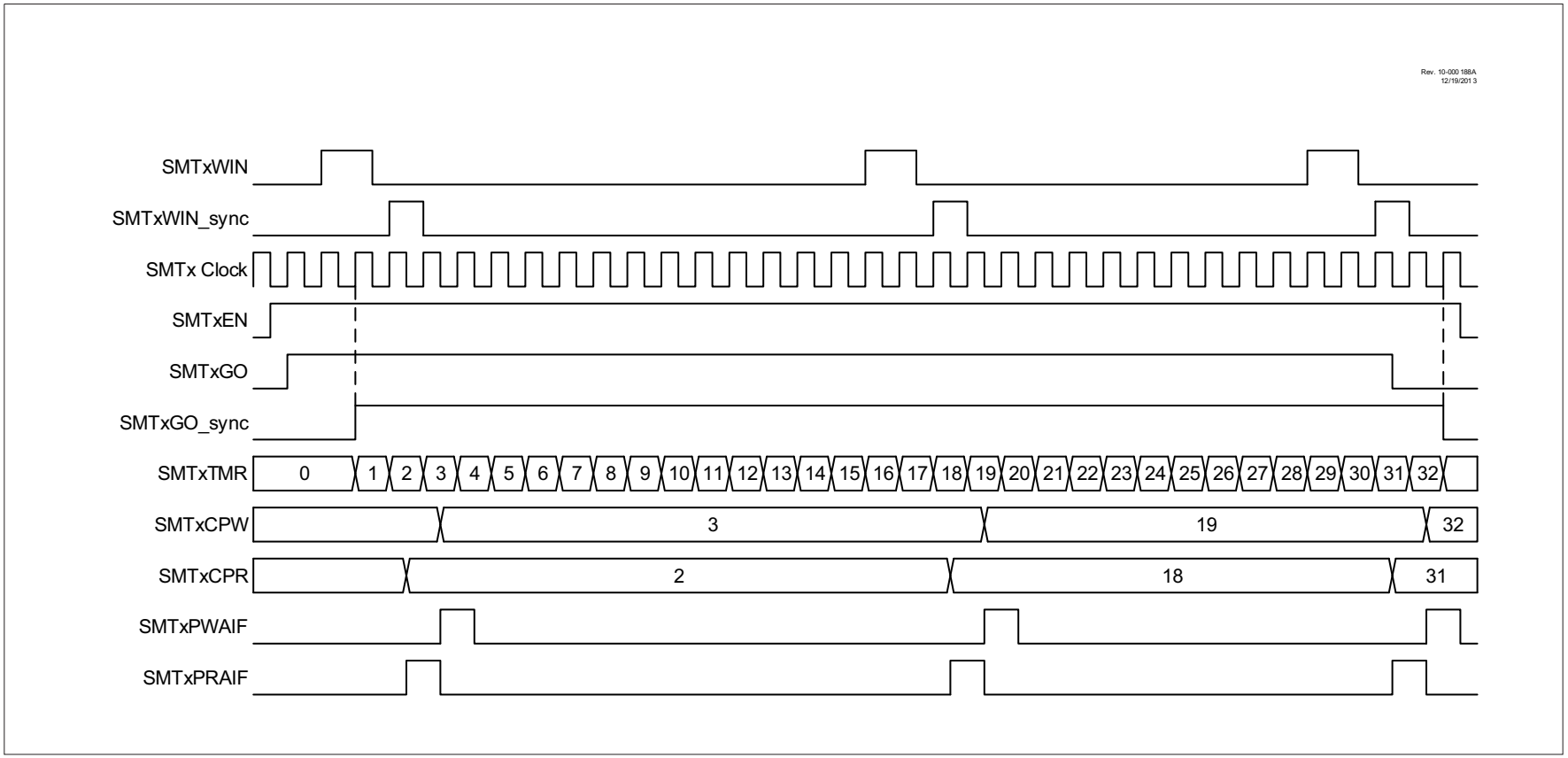
Note 1: 8-bit timer is concatenated with two bits generated by Fosc or two bits of the internal prescaler to create 10-bit time-base.

FIGURE 26-2: PWM OUTPUT



For a step-by-step procedure on how to set up this module for PWM operation, refer to **Section 26.1.9 "Setup for PWM Operation using PWMx Pins"**.

FIGURE 27-16: CAPTURE MODE REPEAT ACQUISITION TIMING DIAGRAM



28.2.3.1 Direction Change in Full-Bridge Mode

In Full-Bridge mode, changing MODE<2:0> controls the forward/reverse direction. Changes to MODE<2:0> change to the new direction on the next rising edge of the modulated input.

A direction change is initiated in software by changing the MODE<2:0> bits of the CWGxCON0 register. The sequence is illustrated in Figure 28-8.

- The associated active output CWGxA and the inactive output CWGxC are switched to drive in the opposite direction.
- The previously modulated output CWGxD is switched to the inactive state, and the previously inactive output CWGxB begins to modulate.
- CWG modulation resumes after the direction-switch dead band has elapsed.

28.2.3.2 Dead-Band Delay in Full-Bridge Mode

Dead-band delay is important when either of the following conditions is true:

1. The direction of the CWG output changes when the duty cycle of the data input is at or near 100%, or
2. The turn-off time of the power switch, including the power device and driver circuit, is greater than the turn-on time.

The dead-band delay is inserted only when changing directions, and only the modulated output is affected. The statically-configured outputs (CWGxA and CWGxC) are not afforded dead band, and switch essentially simultaneously.

Figure 28-8 shows an example of the CWG outputs changing directions from forward to reverse, at near 100% duty cycle. In this example, at time t1, the output of CWGxA and CWGxD become inactive, while output CWGxC becomes active. Since the turn-off time of the power devices is longer than the turn-on time, a shoot-through current will flow through power devices QC and QD for the duration of 't'. The same phenomenon will occur to power devices QA and QB for the CWG direction change from reverse to forward.

When changing the CWG direction at high duty cycle is required for an application, two possible solutions for eliminating the shoot-through current are:

1. Reduce the CWG duty cycle for one period before changing directions.
2. Use switch drivers that can drive the switches off faster than they can drive them on.

FIGURE 28-8: EXAMPLE OF PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE

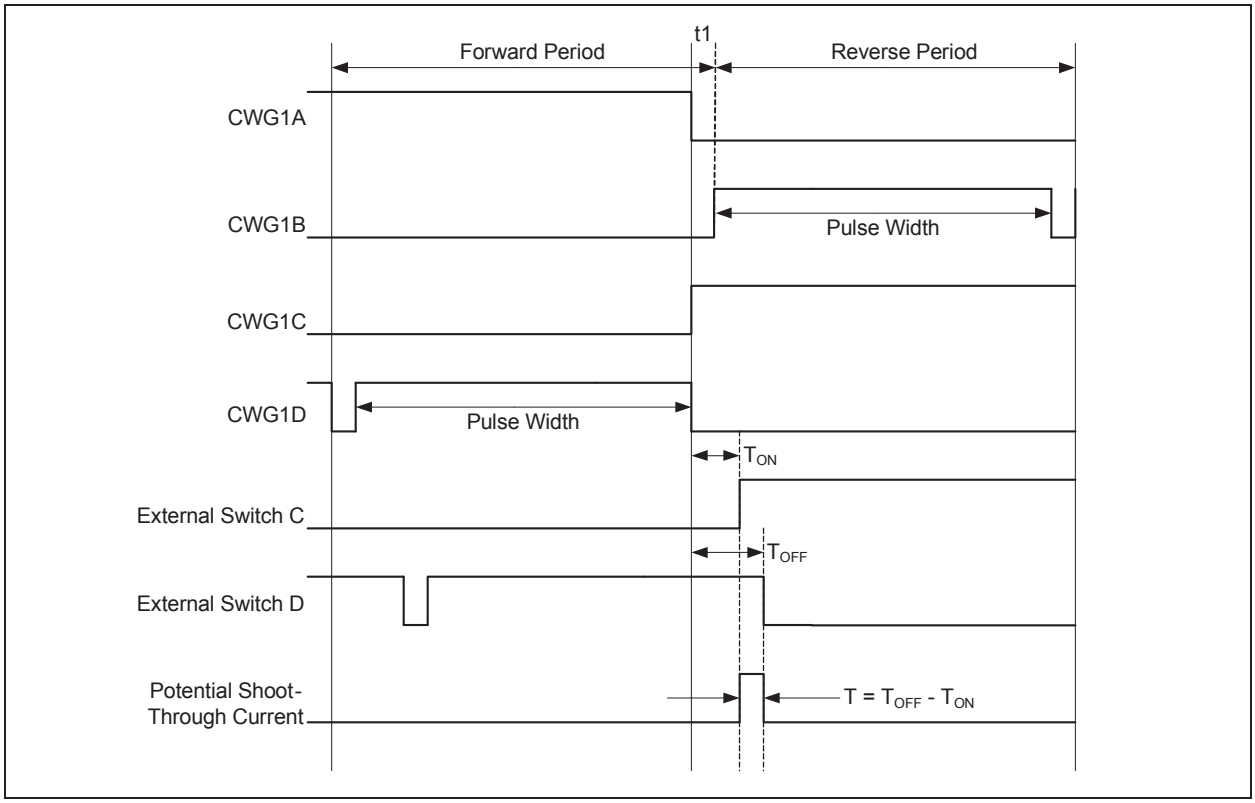
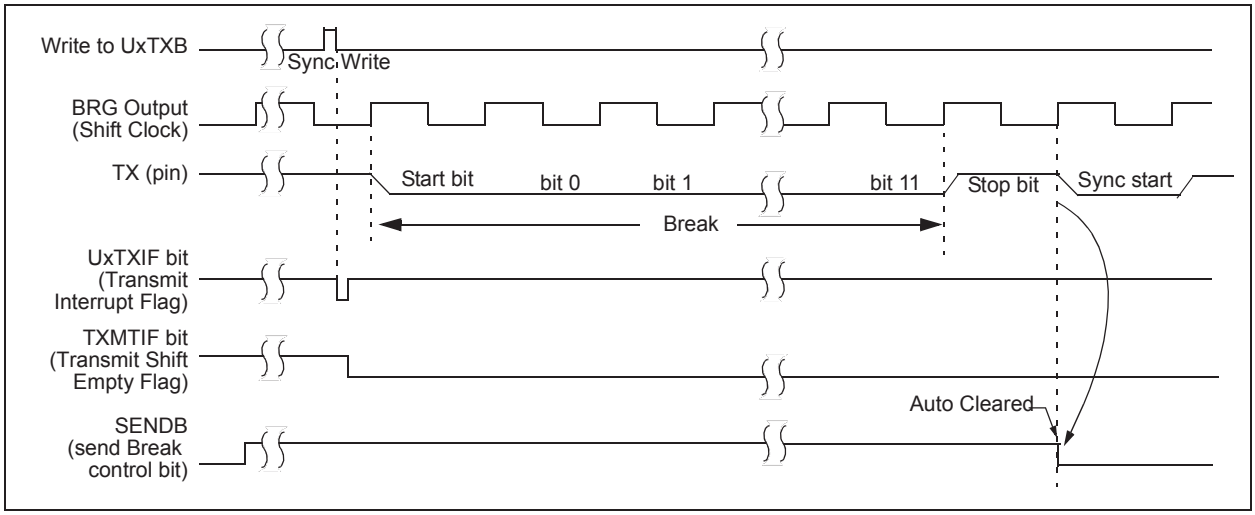


FIGURE 33-15: SEND BREAK CHARACTER SEQUENCE



REGISTER 35-10: I2CxPIR – I2CxIF INTERRUPT FLAG REGISTER

R/W/HS-0	R/W/HS-0	U-0	R/W/HS-0	R/W/HS-0	R/W/HS-0	R/W/HS-0	R/W/HS-0
CNTIF	ACKTIF	—	WRIF	ADRIF	PCIF	RSCIF	SCIF
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	HS = Hardware set HC = Hardware clear

bit 7 **CNTIF:** Byte Count Interrupt Flag bit
1 = When I2CCNT = 0, set by the 9th falling edge of SCL.
0 = I2CCNT condition has not occurred.

bit 6 **ACKTIF:** Acknowledge Status Time Interrupt Flag bit **(2)** (MODE<2:0> = 0xx OR 11x)
1 = Set by the 9th falling edge of SCL for any byte when addressed as a Slave
0 = Acknowledge condition not detected.

bit 5 **Unimplemented:** Read as '0'

bit 4 **WRIF:** Data Write Interrupt Flag bit (MODE<2:0> = 0xx OR 11x)
1 = Set the 8th falling edge of SCL for a received data byte.
0 = Data Write condition not detected

bit 3 **ADRIF:** Address Interrupt Flag bit (MODE<2:0> = 0xx OR 11x)
1 = Set the 8th falling edge of SCL for a matching received (high/low) address byte
0 = Address condition not detected

bit 2 **PCIF:** Stop Condition Interrupt Flag
1 = Set on detection of Stop condition
0 = No Stop condition detected

bit 1 **RSCIF:** Restart Condition Interrupt Flag
1 = Set on detection of Restart condition
0 = No Restart condition detected

bit 0 **SCIF:** Start Condition Interrupt Flag
1 = Set on detection of Start condition
0 = No Start condition detected

Note 1: Enabled interrupt flags are OR'd to produce the PIRx<I2CxIF> bit.
2: ACKTIF is not set by a matching, 10-bit, high address byte with the R/W bit clear. It is only set after the matching low address byte is shifted in.

REGISTER 38-35: ADACT: ADC AUTO CONVERSION TRIGGER CONTROL REGISTER

U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	ACT<4:0>				
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 7-5 **Unimplemented**: Read as '0'

bit 4-0 **ADACT<4:0>**: Auto-Conversion Trigger Select Bits

11111 = Reserved, do not use
•
•
•
11110 = Reserved, do not use
11101 = Software write to ADPCH
11100 = Reserved, do not use
11011 = Software read of ADRESH
11010 = Software read of ADERRH
11001 = CLC4_out
11000 = CLC3_out
10111 = CLC2_out
10110 = CLC1_out
10101 = Logical OR of all Interrupt-on-change Interrupt Flags
10100 = CMP2_out
10011 = CMP1_out
10010 = NCO1_out
10001 = PWM8_out
10000 = PWM7_out
01111 = PWM6_out
01110 = PWM5_out
01101 = CCP4_trigger
01100 = CCP3_trigger
01011 = CCP2_trigger
01010 = CCP1_trigger
01001 = SMT1_trigger
01000 = TMR6_postscaled
00111 = TMR5_overflow
00110 = TMR4_postscaled
00101 = TMR3_overflow
00100 = TMR2_postscaled
00011 = TMR1_overflow
00010 = TMR0_overflow
00001 = Pin selected by ADACTPPS
00000 = External Trigger Disabled

INCF **Increment f**

Syntax: INCF f {,d {,a}}

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (f) + 1 → dest

Status Affected: C, DC, N, OV, Z

Encoding:

0010	10da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 43.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `INCF CNT, 1, 0`

Before Instruction
CNT = FFh
Z = 0
C = ?
DC = ?
After Instruction
CNT = 00h
Z = 1
C = 1
DC = 1

INCFSZ **Increment f, skip if 0**

Syntax: INCFSZ f {,d {,a}}

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

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

Status Affected: None

Encoding:

0011	11da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f' (default).
If the result is '0', the next instruction, which is already fetched, is discarded and a NOP is executed instead, making it a 2-cycle instruction.
If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.
If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 43.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    INCFSZ  CNT, 1, 0
NZERO   :
ZERO    :
```

Before Instruction
PC = Address (HERE)
After Instruction
CNT = CNT + 1
If CNT = 0;
PC = Address (ZERO)
If CNT ≠ 0;
PC = Address (NZERO)

SLEEP **Enter Sleep mode**

Syntax: SLEEP

Operands: None

Operation: 00h → WDT,
0 → WDT postscaler,
1 → $\overline{TO}$,
0 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

0000	0000	0000	0011

Description: The Power-down Status bit ($\overline{PD}$) is cleared. The Time-out Status bit ($\overline{TO}$) is set. Watchdog Timer and its postscaler are cleared.
The processor is put into Sleep mode with the oscillator stopped.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	Go to Sleep

Example: SLEEP

Before Instruction
$\overline{TO}$ = ?
$\overline{PD}$ = ?
After Instruction
$\overline{TO}$ = 1 †
$\overline{PD}$ = 0

† If WDT causes wake-up, this bit is cleared.

SUBFSR **Subtract Literal from FSR**

Syntax: SUBFSR f, k

Operands: $0 \le k \le 63$
$f \in [0, 1, 2]$

Operation: FSR(f) – k → FSRf

Status Affected: None

Encoding:

1110	1001	ffkk	kkkk

Description: The 6-bit literal ‘k’ is subtracted from the contents of the FSR specified by ‘f’.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write to destination

Example: SUBFSR 2, 23h

Before Instruction
FSR2 = 03FFh
After Instruction
FSR2 = 03DCh

TABLE 44-1: REGISTER FILE SUMMARY FOR PIC18(L)F24/25K42 DEVICES (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on page
3F5Eh	CCPTMRS0	C4TSEL		C3TSEL		C2TSEL		C1TSEL		362
3F5Dh - 3F5Bh	—	Unimplemented								
3F5Ah	CWG1STR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	429
3F59h	CWG1AS1	—	AS6E	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E	433
3F58h	CWG1AS0	SHUTDOWN	REN	LSBD		LSAC		—	—	430
3F57h	CWG1CON1	—	—	IN	—	POLD	POLC	POLB	POLA	428
3F56h	CWG1CON0	EN	LD	—	—	—	MODE			427
3F55h	CWG1DBF	—	—	DBF						434
3F54h	CWG1DBR	—	—	DBR						434
3F53h	CWG1ISM	—	—	—	—	IS				430
3F52h	CWG1CLK	—	—	—	—	—	—	—	CS	429
3F51h	CWG2STR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	429
3F50h	CWG2AS1	—	AS6E	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E	433
3F4Fh	CWG2AS0	SHUTDOWN	REN	LSBD		LSAC		—	—	430
3F4Eh	CWG2CON1	—	—	IN	—	POLD	POLC	POLB	POLA	428
3F4Dh	CWG2CON0	EN	LD	—	—	—	MODE			427
3F4Ch	CWG2DBF	—	—	DBF						434
3F4Bh	CWG2DBR	—	—	DBR						434
3F4Ah	CWG2ISM	—	—	—	—	IS				430
3F49h	CWG2CLK	—	—	—	—	—	—	—	CS	429
3F48h	CWG3STR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	429
3F47h	CWG3AS1	—	AS6E	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E	433
3F46h	CWG3AS0	SHUTDOWN	REN	LSBD		LSAC		—	—	430
3F45h	CWG3CON1	—	—	IN	—	POLD	POLC	POLB	POLA	428
3F44h	CWG3CON0	EN	LD	—	—	—	MODE			427
3F43h	CWG3DBF	—	—	DBF						434
3F42h	CWG3DBR	—	—	DBR						434
3F41h	CWG3ISM	—	—	—	—	IS				430
3F40h	CWG3CLK	—	—	—	—	—	—	—	CS	429
3F3Fh	NCO1CLK	PWS			—	CKS				457
3F3Eh	NCO1CON	EN	—	OUT	POL	—	—	—	PFM	456
3F3Dh	NCO1INCU	INC								460
3F3Ch	NCO1INCH	INC								459
3F3Bh	NCO1INCL	INC								459
3F3Ah	NCO1ACCU	ACC								459
3F39h	NCO1ACCH	ACC								458
3F38h	NCO1ACCL	ACC								458
3F37h - 3F24h	—	Unimplemented								
3F23h	SMT1WIN	—	—	—	WSEL					401
3F22h	SMT1SIG	—	—	—	SSEL					402
3F21h	SMT1CLK	—	—	—	—	—	CSEL			400
3F20h	SMT1STAT	CPRUP	CPWUP	RST	—	—	TS	WS	AS	399
3F1Fh	SMT1CON1	GO	REPEAT	—	—	MODE				398
3F1Eh	SMT1CON0	EN	—	STP	WPOL	SPOL	CPOL	PS		397
3F1Dh	SMT1PRU	PR								406
3F1Ch	SMT1PRH	PR								406
3F1Bh	SMT1PRL	PR								406

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition
Note 1: Not present in LF devices.