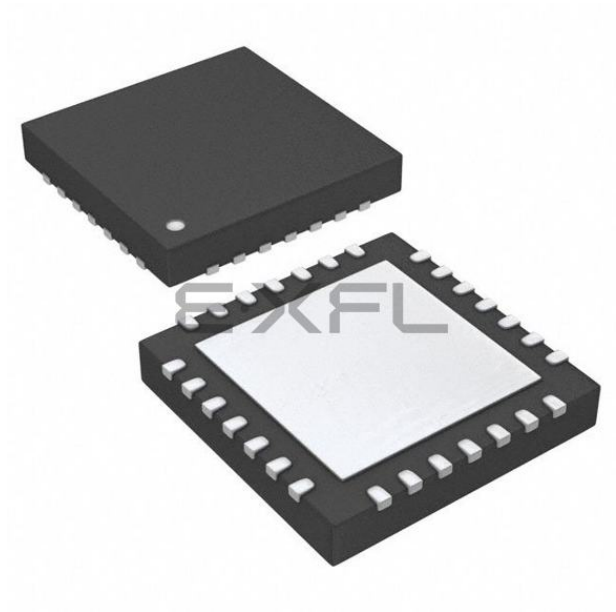


Welcome to E-XFL.COM

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	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K 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 ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-UFQFN Exposed Pad
Supplier Device Package	28-UQFN (4x4)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f25k42-i-mv

3.1 System Arbitration

The System Arbiter resolves memory access between the System Level Selections (i.e., Main, Interrupt Service Routine) and Peripheral Selection (i.e., DMA and Scanner) based on user-assigned priorities. Each of the system level and peripheral selections has its own priority selection registers. Memory access priority is resolved using the number written to the corresponding Priority registers, 0 being the highest priority and 4 being the lowest priority. The Default priorities are listed in Table 3-1.

In case the user wants to change priorities then ensure that each Priority register is written with a unique value from 0 to 4.

TABLE 3-1: DEFAULT PRIORITIES

Selection		Priority register Reset value
System Level	ISR	0
	MAIN	1

TABLE 3-1: DEFAULT PRIORITIES

Selection		Priority register Reset value
Peripheral	DMA1	2
	DMA2	3
	SCANNER	4

FIGURE 3-2: PIC18(L)F24/25K42 SYSTEM ARBITER BLOCK DIAGRAM

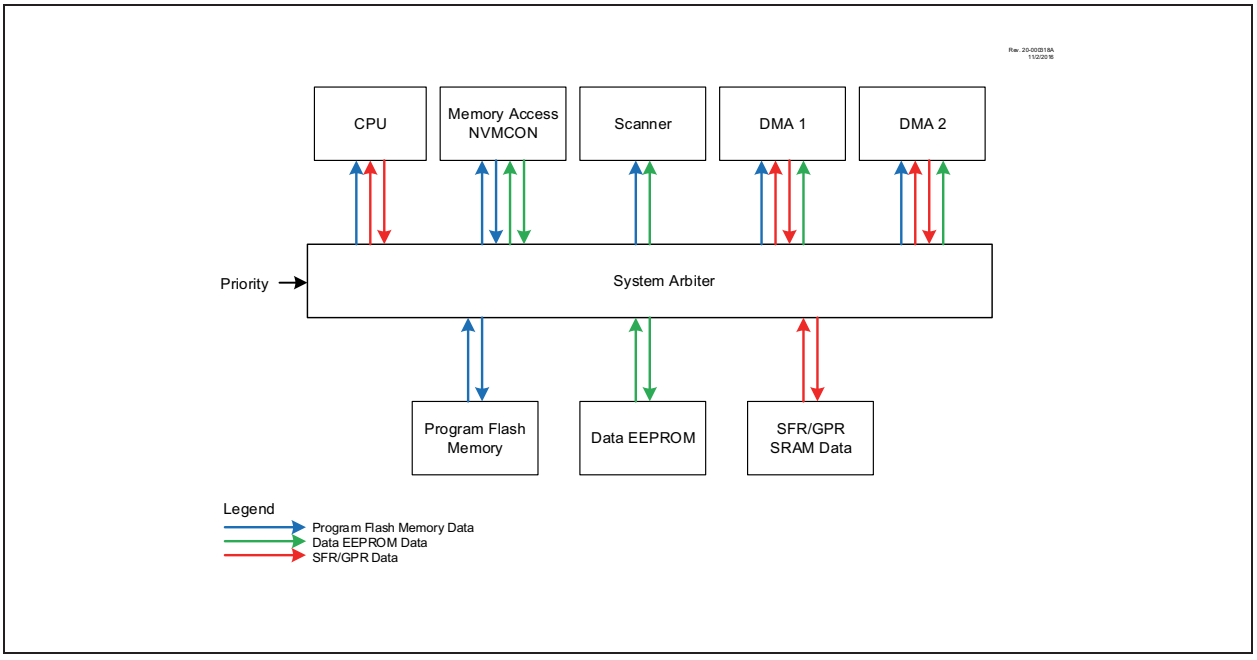


TABLE 4-11: REGISTER FILE SUMMARY FOR PIC18(L)F24/25K42 DEVICES (CONTINUED)

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.

4.7.3.2 FSR Registers and POSTINC, POSTDEC, PREINC and PLUSW

In addition to the INDF operand, each FSR register pair also has four additional indirect operands. Like INDF, these are “virtual” registers which cannot be directly read or written. Accessing these registers actually accesses the location to which the associated FSR register pair points, and also performs a specific action on the FSR value. They are:

- POSTDEC: accesses the location to which the FSR points, then automatically decrements the FSR by 1 afterwards
- POSTINC: accesses the location to which the FSR points, then automatically increments the FSR by 1 afterwards
- PREINC: automatically increments the FSR by one, then uses the location to which the FSR points in the operation
- PLUSW: adds the signed value of the W register (range of -127 to 128) to that of the FSR and uses the location to which the result points in the operation.

In this context, accessing an INDF register uses the value in the associated FSR register without changing it. Similarly, accessing a PLUSW register gives the FSR value an offset by that in the W register; however, neither W nor the FSR is actually changed in the operation. Accessing the other virtual registers changes the value of the FSR register.

FIGURE 4-6: INDIRECT ADDRESSING

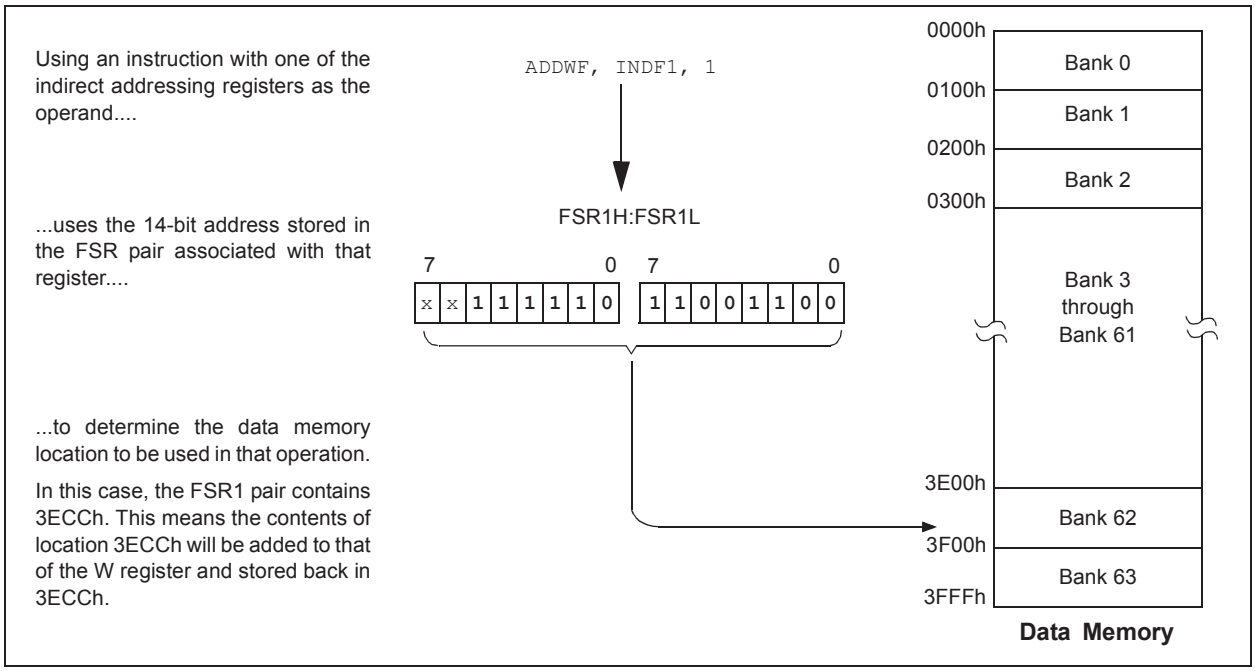


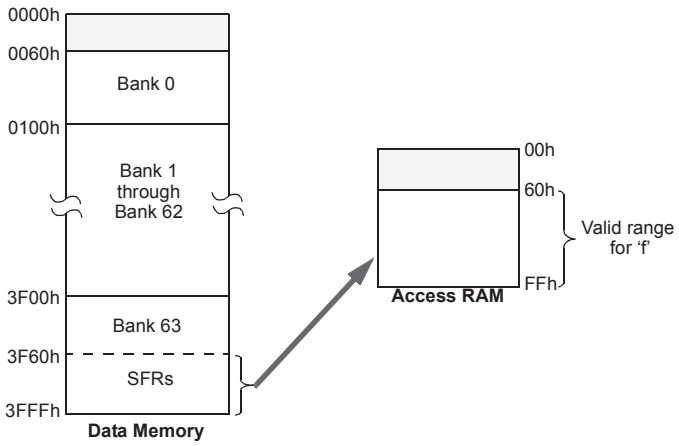
FIGURE 4-7: COMPARING ADDRESSING OPTIONS FOR BIT-ORIENTED AND BYTE-ORIENTED INSTRUCTIONS (EXTENDED INSTRUCTION SET ENABLED)

EXAMPLE INSTRUCTION: `ADDWF, f, d, a` (Opcode: `0010 01da ffff ffff`)

When 'a' = 0 and f ≥ 60h:

The instruction executes in Direct Forced mode. 'f' is interpreted as a location in the Access RAM between 060h and 0FFh. This is the same as locations 3F60h to 3FFFh (Bank 63) of data memory.

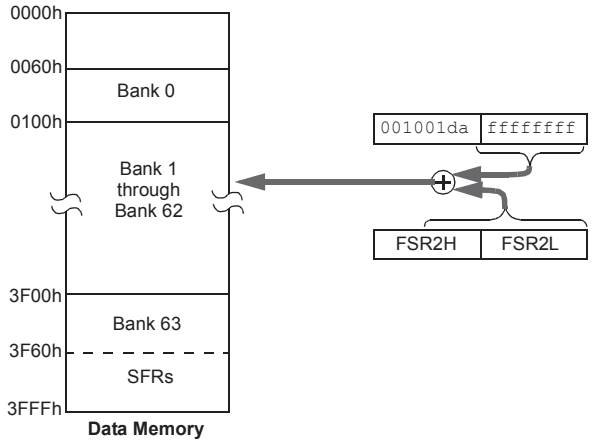
Locations below 60h are not available in this addressing mode.



When 'a' = 0 and f ≤ 5Fh:

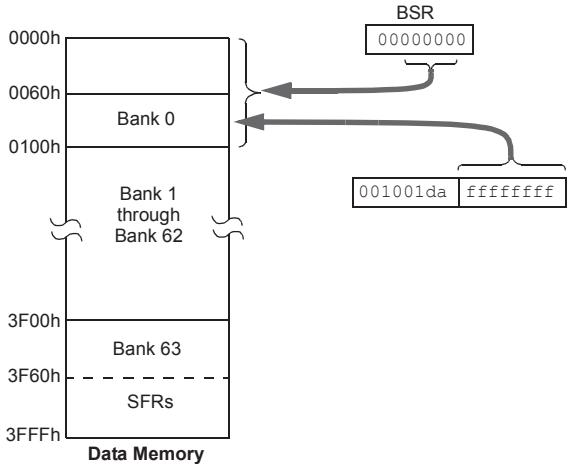
The instruction executes in Indexed Literal Offset mode. 'f' is interpreted as an offset to the address value in FSR2. The two are added together to obtain the address of the target register for the instruction. The address can be anywhere in the data memory space.

Note that in this mode, the correct syntax is now:
`ADDWF [k], d`
where 'k' is the same as 'f'.



When 'a' = 1 (all values of f):

The instruction executes in Direct mode (also known as Direct Long mode). 'f' is interpreted as a location in one of the 63 banks of the data memory space. The bank is designated by the Bank Select Register (BSR). The address can be in any implemented bank in the data memory space.



REGISTER 16-14: SCANLADRL: SCAN LOW ADDRESS LOW BYTE 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
LADR<7:0>(1, 2)							
bit 7							bit 0

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

bit 7-0 **LADR<7:0>**: Scan Start/Current Address bits(1, 2)
Least Significant bits of the current address to be fetched from, value increments on each fetch of memory

Note 1: Registers SCANLADRU/H/L form a 22-bit value, but are not guarded for atomic or asynchronous access; registers should only be read or written while SGO = 0 (SCANCON0 register).
2: While SGO = 1 (SCANCON0 register), writing to this register is ignored.

REGISTER 16-15: SCANHADRU: SCAN HIGH ADDRESS UPPER BYTE REGISTER

U-0	U-0	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
—	—	HADR<21:16>					
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 **HADR<21:16>:** Scan End Address bits(1, 2)
Upper bits of the address at the end of the designated scan

Note 1: Registers SCANHADRU/H/L form a 22-bit value but are not guarded for atomic or asynchronous access; registers should only be read or written while SGO = 0 (SCANCON0 register).
2: While SGO = 1 (SCANCON0 register), writing to this register is ignored.

24.6 Timer2 Operation During Sleep

When PSYNC = 1, Timer2 cannot be operated while the processor is in Sleep mode. The contents of the T2TMR and T2PR registers will remain unchanged while processor is in Sleep mode.

When PSYNC = 0, Timer2 will operate in Sleep as long as the clock source selected is also still running. Selecting the LFINTOSC, MFINTOSC, or HFINTOSC oscillator as the timer clock source will keep the selected oscillator running during Sleep.

25.4.2 SETUP FOR PWM OPERATION

The following steps should be taken when configuring the CCP module for standard PWM operation:

1. Use the desired output pin RxyPPS control to select CCPx as the source and disable the CCPx pin output driver by setting the associated TRIS bit.
2. Load the T2PR register with the PWM period value.
3. Configure the CCP module for the PWM mode by loading the CCPxCON register with the appropriate values.
4. Load the CCPRxL register, and the CCPRxH register with the PWM duty cycle value and configure the FMT bit of the CCPxCON register to set the proper register alignment.
5. Configure and start Timer2:
   - Clear the TMR2IF interrupt flag bit of the respective PIR register. See Note below.
   - Select the timer clock source to be as FOSC/4 using the T2CLK register. This is required for correct operation of the PWM module.
   - Configure the CKPS bits of the T2CON register with the Timer prescale value.
   - Enable the Timer by setting the ON bit of the T2CON register.
6. Enable PWM output pin:
   - Wait until the Timer overflows and the TMR2IF bit of the PIR4 register is set. See Note below.
   - Enable the CCPx pin output driver by clearing the associated TRIS bit.

Note:	In order to send a complete duty cycle and period on the first PWM output, the above steps must be included in the setup sequence. If it is not critical to start with a complete PWM signal on the first output, then step 6 may be ignored.

25.4.3 TIMER2 TIMER RESOURCE

The PWM standard mode makes use of the 8-bit Timer2 timer resources to specify the PWM period.

25.4.4 PWM PERIOD

The PWM period is specified by the T2PR register of Timer2. The PWM period can be calculated using the formula of Equation 25-1.

EQUATION 25-1: PWM PERIOD

$$PWM\ Period = [(T2PR) + 1] \bullet 4 \bullet TOSC \bullet (TMR2\ Prescale\ Value)$$

Note 1: TOSC = 1/FOSC

When T2TMR is equal to T2PR, the following three events occur on the next increment cycle:

- T2TMR is cleared
- The CCPx pin is set. (Exception: If the PWM duty cycle = 0%, the pin will not be set.)
- The PWM duty cycle is transferred from the CCPRxL/H register pair into a 10-bit buffer.

Note:	The Timer postscaler (see **Section 24.3 "External Reset Sources"**) is not used in the determination of the PWM frequency.

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

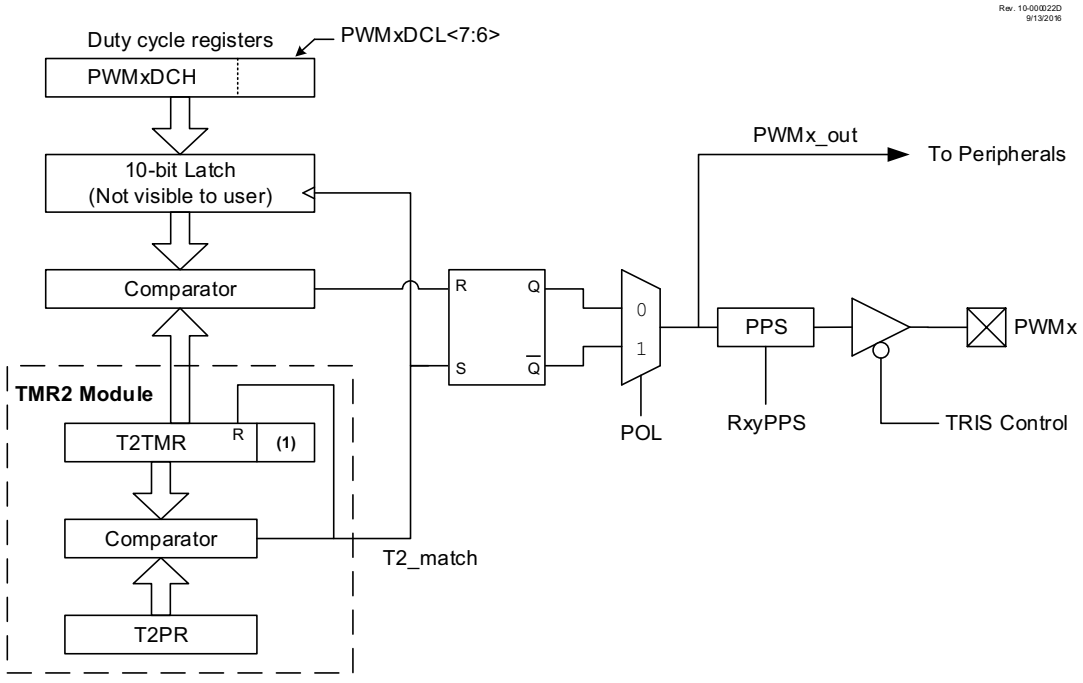
Note: The corresponding TRIS bit must be cleared to enable the PWM output on the PWMx pin.

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.

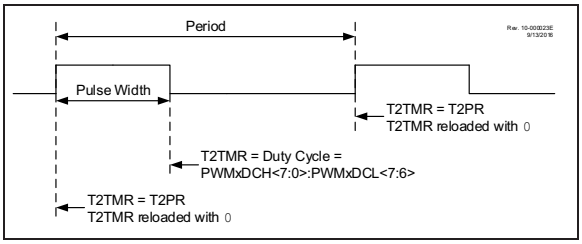
Figure 26-2 shows a typical waveform of the PWM signal.

FIGURE 26-1: SIMPLIFIED PWM BLOCK DIAGRAM



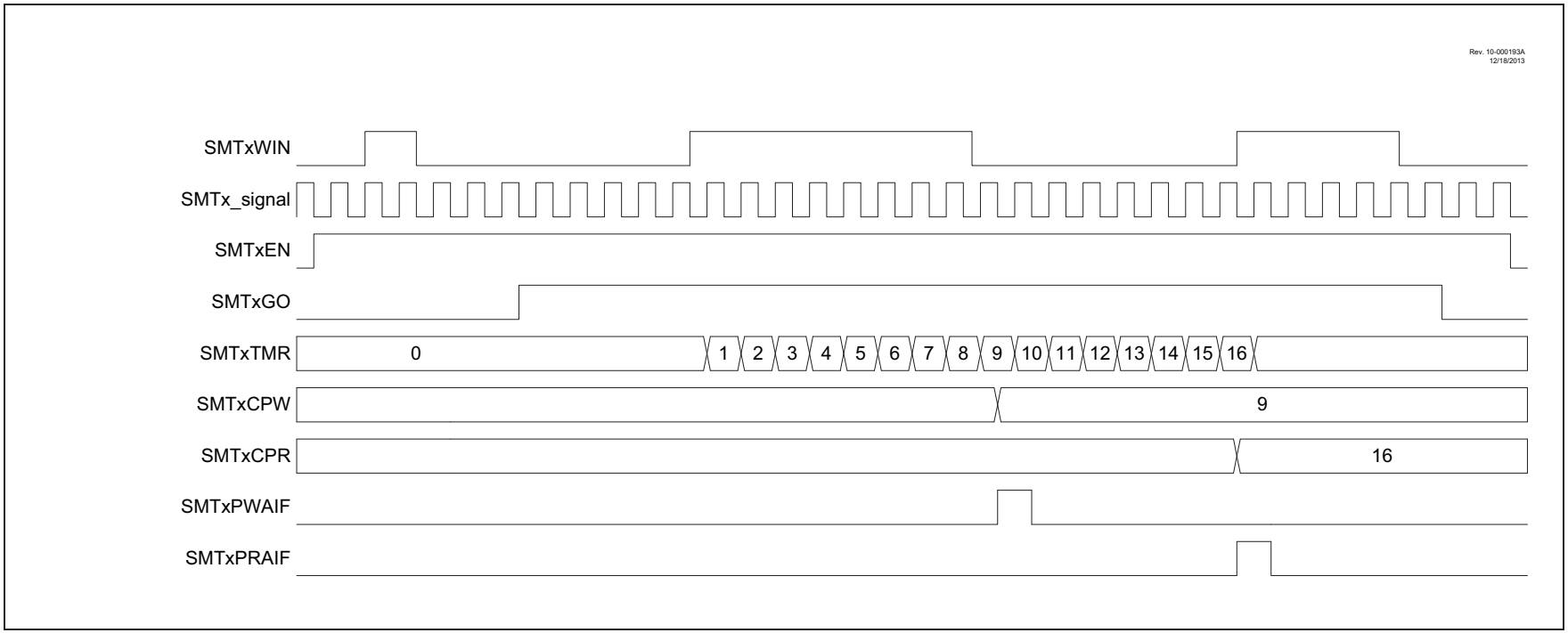
Note 1: 8-bit timer is concatenated with two bits generated by Fosc or two bits of the internal prescaler to create 10-bit time-base.

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 27-22: WINDOWED COUNTER MODE SINGLE ACQUISITION TIMING DIAGRAM



27.7 Interrupts

The SMT can trigger an interrupt under three different conditions:

- PW Acquisition Complete
- PR Acquisition Complete
- Counter Period Match

The interrupts are controlled by the PIR and PIE registers of the device.

27.7.1 PW AND PR ACQUISITION INTERRUPTS

The SMT can trigger interrupts whenever it updates the SMT1CPW and SMT1CPR registers, the circumstances for which are dependent on the SMT mode, and are discussed in each mode's specific section. The SMT1CPW interrupt is controlled by SMT1PWAIF and SMT1PWAIE bits in the respective PIR and PIE registers. The SMT1CPR interrupt is controlled by the SMT1PRAIF and SMT1PRAIE bits, also located in the respective PIR and PIE registers.

In synchronous SMT modes, the interrupt trigger is synchronized to the SMT1CLK. In Asynchronous modes, the interrupt trigger is asynchronous. In either mode, once triggered, the interrupt will be synchronized to the CPU clock.

27.7.2 COUNTER PERIOD MATCH INTERRUPT

As described in **Section 27.1.2 "Period Match interrupt"**, the SMT will also interrupt upon SMT1TMR, matching SMT1PR with its period match limit functionality described in **Section 27.3 "Halt Operation"**. The period match interrupt is controlled by SMT1IF and SMT1IE, located in the respective PIR and PIE registers.

REGISTER 33-6: UxUIR: UART GENERAL INTERRUPT REGISTER

R/S/W-0/0	R/S/W-0/0	U-0	U-0	U-0	R/W-0/0	U-0	U-0
WUIF	ABDIF	—	—	—	ABDIE	—	—
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

bit 7 **WUIF:** Wake-up Interrupt bit
- 1 = Idle to non-idle transition on RX line detected when WUE is set. Also sets UxIF. (WUIF must be cleared by software to clear UxIF)
- 0 = WUE not enabled by software or no transition detected

bit 6 **ABDIF**: Auto-baud detect interrupt bit
- 1 = Auto-baud detection complete. Status shown in UxIF when ABDIE is set. (Must be cleared by software)
- 0 = Auto-baud not enabled or auto-baud enabled and auto-baud detection not complete

bit 5-3 **Unimplemented:** Read as '0'

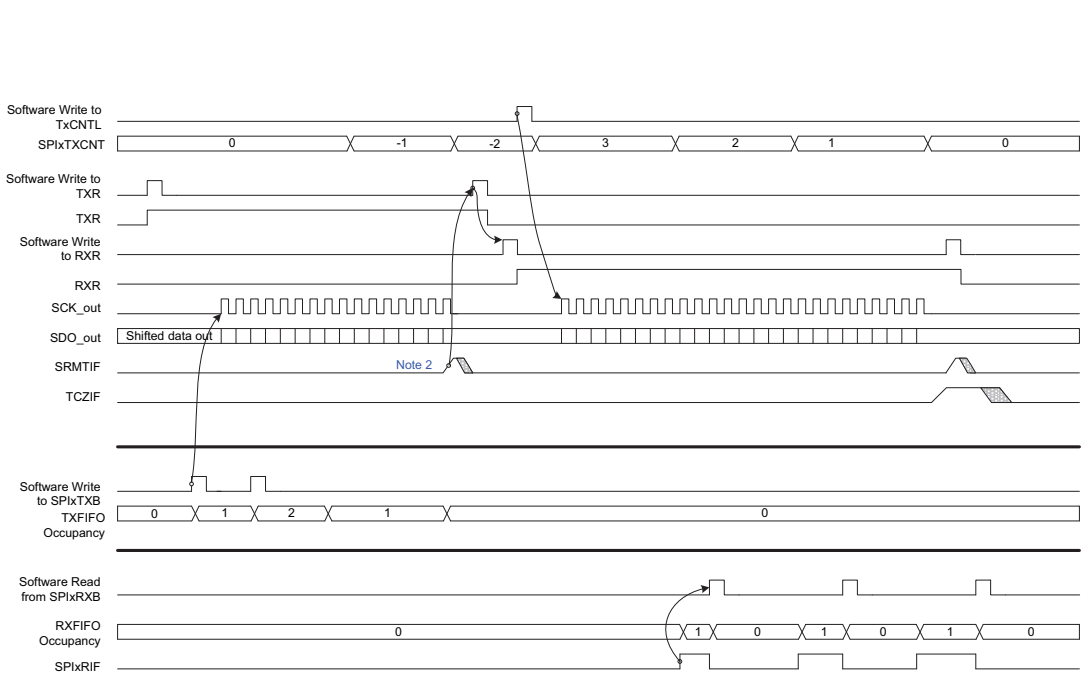
bit 2 **ABDIE:** Auto-baud Detect Interrupt Enable bit
- 1 = ABDIF will set UxIF bit in PIRx register
- 0 = ABDIF will not set UxIF

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

34.5.3 RECEIVE ONLY MODE

When RXR is set and TXR is clear, the SPI master is in Receive Only mode. In this mode, data transfers when the RXFIFO is not full and the Transfer Counter is non-zero. In this mode, writing a value to SPIxTCNTL will start the clocks for transfer. The clocks will suspend while the RXFIFO is full and cease when the SPIxTCNT reaches zero (see **Section 34.4 "Transfer Counter"**). If there is any data in the TXFIFO, the first data written to the TXFIFO will be transmitted on each data exchange, although the TXFIFO occupancy will not change, meaning that the same message will be sent on each transmission. If there is no data in the TXFIFO, the most recently received data will instead be transmitted. Figure 34-5 shows an example of sending a command using **Section 34.5.2 "Transmit Only Mode"** and then receiving a byte of data using this mode.

FIGURE 34-5: SPI MASTER OPERATION, COMMAND+READ DATA, TXR/RXR=0/1



Note: 1. SS(out) is not shown
2. Software must wait for shift-register empty (SRMTIF) before changing TXR, RXR, BYTES and BITS controls. This is not considered an imposition in this case, because the slave probably needs time to load output data (see also Figure 4-14).

34.5.4 TRANSFER OFF MODE

When both TXR and RXR are cleared, the SPI master is in Transfer Off mode. In this mode, SCK will not toggle and no data is exchanged. However, writes to SPIxTXB will be transferred to the TXFIFO which will be transmitted if the TXR bit is set.

If a BTO event occurs when the module is configured as a master and is active, (i.e., MMA bit is set), and the module immediately tries to assert a Stop condition and also sets the BTOIF bit. The actual generation of the Stop condition may be delayed if the bus is been clock stretched by some slave device. The MMA bit will be cleared only after the Stop condition is generated.

35.3.10 ADDRESS BUFFERS

The I^2C module has two address buffer registers, I2CxADB0 and I2CxADB1. Depending on the mode, these registers are used as either receive or transmit address buffers. See Table 35-2 for data flow directions in these registers. In Slave modes, these registers are only updated when there is an address match. The ADB bit in the I2CxCON2 register is used to enable/disable the address buffer functionality. When disabled, the address data is sourced from the transmit buffer and is stored in the receive buffer.

TABLE 35-2: ADDRESS BUFFER DIRECTION AS PER I^2C MODE

Modes	MODE<2:0>	I2CxADB0	I2CxADB1
Slave (7-bit)	000	RX	—
	001	RX	—
Slave (10-bit)	010	RX	RX
	011	RX	RX
Master (7-bit)	100	—	TX
Master (10-bit)	101	TX	TX
Multi-Master (7-bit)	110	RX	TX
	111	RX	TX

35.3.10.1 Slave Mode (7-bit)

In 7-bit Slave mode, I2CxADB0 is loaded with the received matching address and R/W data. The I2CxADB1 register is ignored in this mode.

35.3.10.2 Slave Mode (10-bit)

In 10-bit Slave mode, I2CxADB0 is loaded with the lower eight bits of the matching received address. I2CxADB1 is loaded with full eight bits of the high address byte, including the R/W bit.

35.3.10.3 Master Mode (7-bit)

The I2CxADB0 register is ignored in this mode. In 7-bit Master mode, the I2CxADB1 register is used to copy address data byte, including the R/W value, to the shift register.

35.3.10.4 Master Mode (10-bit)

In 10-bit Master mode, the I2CxADB0 register stores the low address data byte value that will be copied to the shift register after the high address byte is shifted out. The I2CxADB1 register stores the high address byte value that will be copied to the shift register. It is up to the user to specify all eight of these bits, even though the I^2C specification defines the upper five bits as a constant.

35.3.10.5 Multi-Master Mode (7-bit only)

In Multi-Master mode, the device can be both master and slave depending on the sequence of events on the bus. If being addressed as a slave, the I2CxADB0 register stores the received matching slave address byte. If the device is trying to communicate as a master on the bus, the contents of the I2CxADB1 register are copied to the shift register for addressing a slave device.

35.3.11 RECEIVE AND TRANSMIT BUFFER

The receive buffer holds one byte of data while another is shifted into the SDA pin. The user can access the buffer by software (or DMA) through the I2CxRXB register. When new data is loaded into the I2CxRXB register, the receive buffer full Status bit (RXBF) is set and reading the I2CxRXB register clears this bit.

If the user tries to read I2CxRXB when it is empty (i.e., RXBF = 0), receive read error bit (RXRE) is set and a NACK will be generated. The user must clear the error bit to resume normal operation.

The transmit buffer holds one byte of data while another can be shifted out through the SDA pin. The user can access the buffer by software (or DMA) through the I2CxTXB register. When the I2CxTXB does not contain any transmit data, the transmit buffer empty status bit (TXBE) is set. At this point, the user can load another byte into the buffer.

If the user tries to write I2CxTXB when it is NOT empty (i.e. TXBE = 0), transmit write error flag bit (TXRE) is set and the new data is discarded. When TXRE is set, the user must clear this error condition to resume normal operation.

By setting the CLRBF bit in the I2CxSTAT1 register, the user can clear both receive and transmit buffers. CLRBF will also clear the I2CxRXIF and I2CxTXIF bits.

35.3.12 CLOCK STRETCHING

When a slave device has not completed processing data, it can delay the transfer of more data through the process of clock stretching. An addressed slave device may hold the SCL clock line low after receiving or sending a bit, indicating that it is not yet ready to continue. The master will attempt to raise the SCL line in order to transfer the next bit, but will detect that the clock line has not yet been released. Since the SCL connection is open-drain, the slave has the ability to hold the line low until it is ready to continue communicating. Clock stretching allows receivers that cannot keep up with a transmitter to control the flow of incoming data.

Clock stretching can be enabled or disabled by the clearing or setting of CSTRDIS (clock stretching disable) bit in the I2CxCON2 register. This bit is valid only in the Multi-Master and Slave modes of operation.

35.3.12.1 Clock Stretching for Buffer Operations

If enabled, clock stretching is forced during buffer read/write operations. For example, in Slave mode if RXBF = 1 (receive buffer full), the clock will be stretched after the seventh falling edge of SCL. The SCL line is released only after the user reads data from the receive buffer. This ensures that there is never a receive data overflow. In this situation, if clock stretching is disabled, the RXO bit in I2CxCON1 is set indicating a receive overflow. When set, the module will always respond with a NACK.

Similarly, when TXBE = 1 (transmit buffer empty) and I2CCNT! = 0, the clock is stretched after the 8th falling edge of SCL. The SCL line is released only after the user loads new data into the transmit buffer. This ensures that there is never a transmit underflow. In this situation, if clock stretching is disabled, the TXUF bit in I2CxCON1 is set indicating a transmit underflow. When set, the module will always respond with a NACK.

35.3.12.2 Clock Stretching for Other Slave Operations

There are three Interrupt and Hold bits that provide clock stretching in Slave mode. These bits can also be used in conjunction with the I2xIE bit in PIR3/6 register to generate system level interrupts.

- Incoming address match interrupt
  - Clock stretching after an incoming matching address byte is enabled by the Address Interrupt and Hold (ADRIE) bit of the I2CxPIE register. When ADRIE = 1, the CSTR bit is set and the SCL line is stretched following the 8th falling edge of SCL of a received matching address. This allows the user to read the received address from the I2CADB0/1 registers and selectively ACK/NACK based on the received address. Clock stretching from ADRIE is released by software clearing the CSTR bit.
- Data Write Interrupt
  - The data write interrupt and hold enable (WRIE) bit is used to enable clock stretching after a received data byte. When WRIE = 1, the CSTR bit is set, and the SCL line is stretched, following the 8th falling SCL edge for incoming slave data. This bit allows user software to selectively ACK/NACK each received data byte. Clock stretching from WRIE is released by software clearing the CSTR bit.
- Acknowledge status
  - The acknowledge status time interrupt and hold enable (ACKTIE) bit is used to enable clock stretching after the ACK phase of a transmission. This bit enables clock stretching for all address/data transactions; address, write, or read. Following the ACK, the slave hardware will set CSTR. Clock stretching from ACKTIE is released by software clearing the CSTR bit.

35.3.13 DATA BYTE COUNT

The I2CxCNT register is used to specify the number of bytes in a complete I^2C packet. The value in this register will decrement every time a data byte is received or transmitted from the I^2C module. The I2CxCNT register will not decrement past zero.

If a byte transfer causes the I2CxCNT register to decrement to zero, the Count Interrupt Flag bit (CNTIF) in I2CxPIR is set. This flag bit is set on the 9^{th} falling edge of SCL for transmit operations and can be more informational if the edge is specified.

The I2CxCNT register can be auto-loaded if the ACNT bit in the I2CxCON2 register is set. When ACNT bit is set, the data byte following the address byte is loaded into the I2CxCNT register.

Note 1:	I2CxCNT decrements on the eighth (receive) or ninth (transmit) falling edge of SCL; writes during this bit time can corrupt the value.
2:	If the block size of the message is greater than 255, the I2CxCNT register can be updated mid-message to prevent decrement to zero.

35.4 I^2C Slave Mode

The I^2C Slave mode operates in one of four modes selected in the Mode bits of I2CxCON0. The modes can be divided into 7- and 10-bit Addressing modes. 10-bit Addressing modes operate the same as 7-bit with some additional overhead for handling the larger addresses.

35.4.1 SLAVE ADDRESSING MODES

The I2CxADR/1/2/3 registers contain the Slave mode addresses. The first byte received after a Start or Restart condition is compared against the values stored in these registers. If the byte matches a value, it is loaded into the I2CxADB0/1 registers. If the value does not match, there is no response from the module. The I^2C module can be configured in the following Slave configurations.

FIGURE 35-8: I²C SLAVE, 7-BIT ADDRESS, RECEPTION NO I2CxCNT (ACKTIE = 0, ADRIE = 1, WRIE = 1)

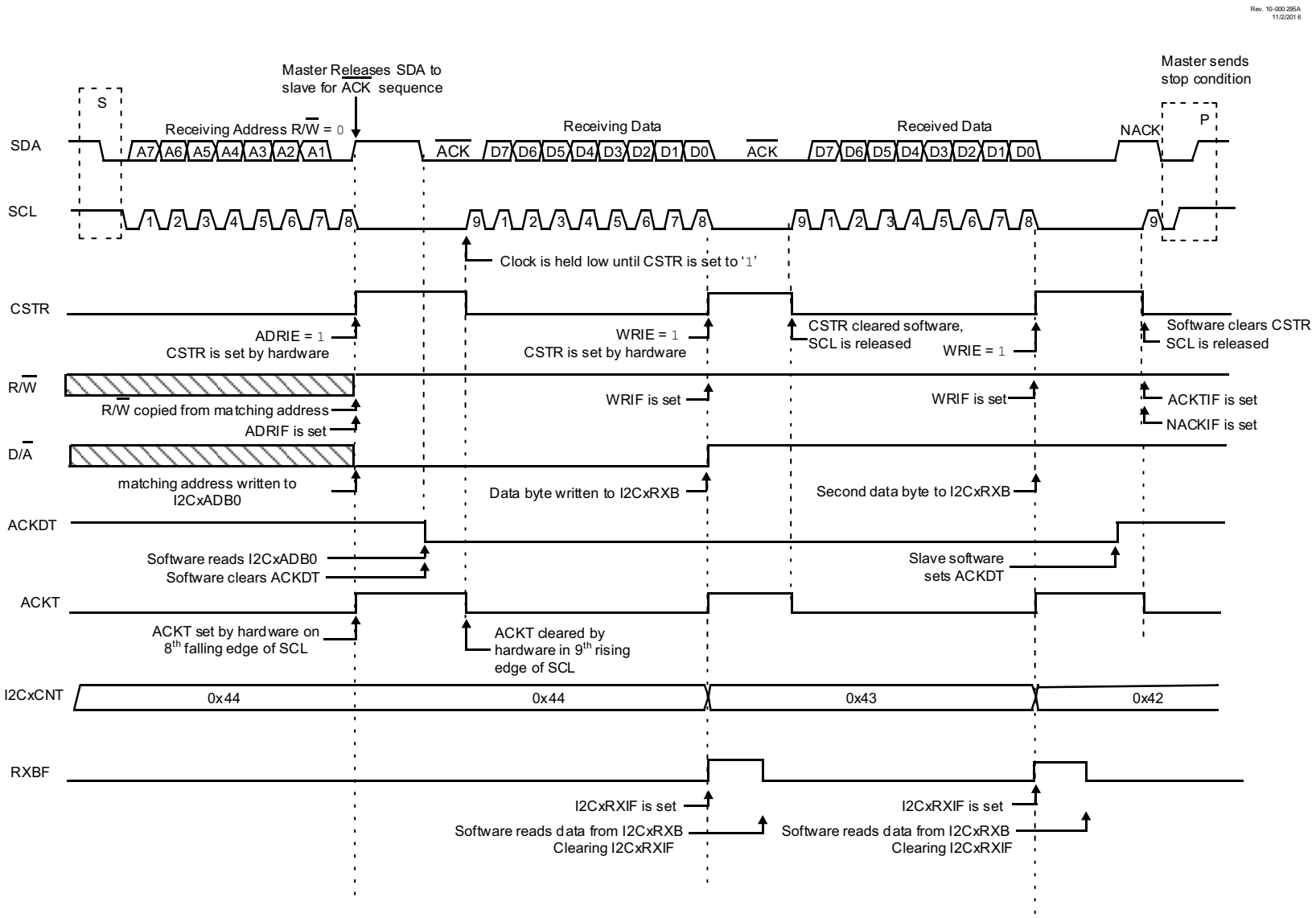
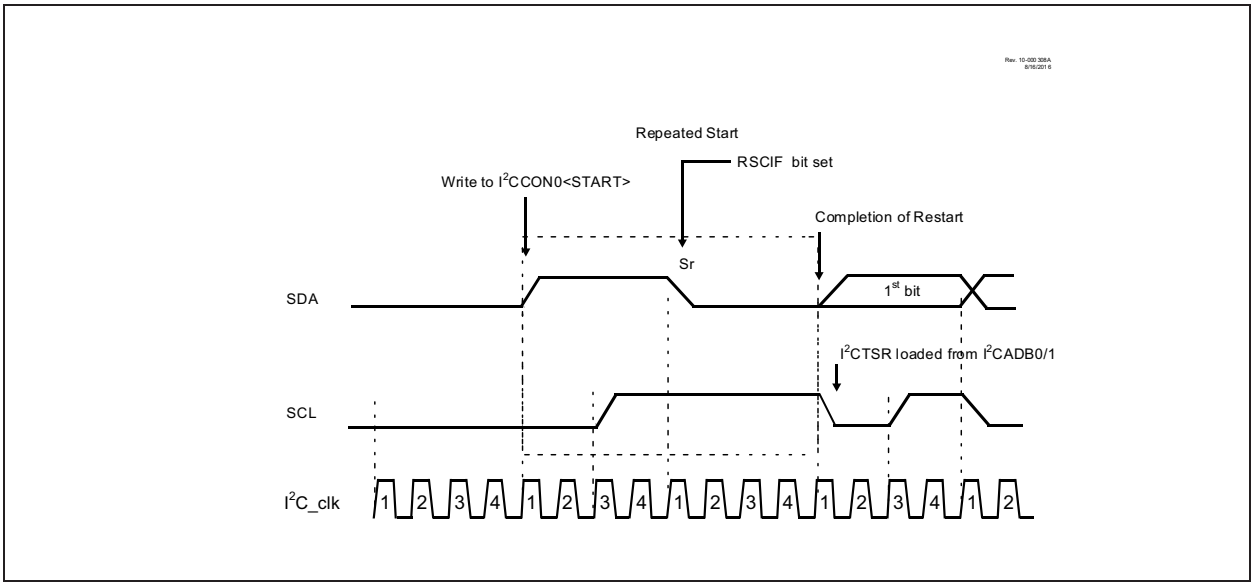


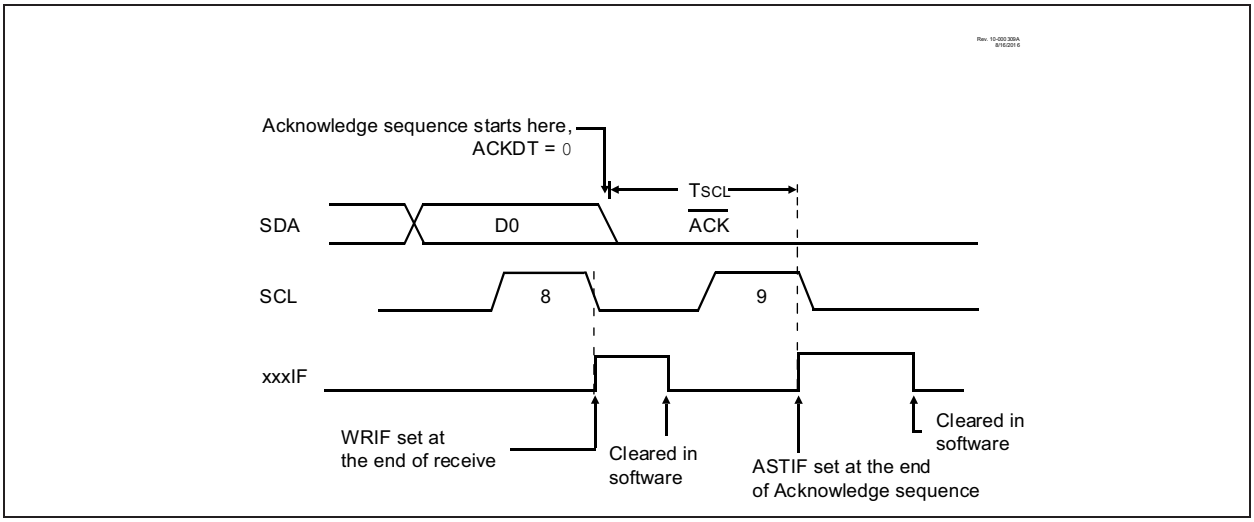
FIGURE 35-16: REPEATED START CONDITION TIMING



35.5.7 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled automatically following an address/data byte transmission. The SCL pin is pulled low and the contents of the Acknowledge Data bits (ACKDT/ACKCNT) are presented on the SDA pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The master then waits one clock period (TSCL) and the SCL pin is released high. When the SCL pin is sampled high (clock arbitration), the master counts another TSCL. The SCL pin is then pulled low. Figure 35-17 shows the timings for Acknowledge sequence.

FIGURE 35-17: ACKNOWLEDGE SEQUENCE TIMING



35.5.8 STOP CONDITION TIMING

A Stop bit is asserted on the SDA pin at the end of receive/transmit when I2CxCNT = 0. After the last byte of a receive/transmit sequence, the SCL line is held low. The master asserts the SDA line low. The SCL pin is then released high TSCL/2 later and is detected high. The SDA pin is then released. When the SDA pin transitions high while SCL is high, the PCIF bit of the I2CxIF register is set. Figure 35-18 shows the timings for a Stop condition.

REGISTER 35-5: I2CxBTO: I²C BUS TIMEOUT SELECTION REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-0/0	R/W-0/0
—	—	—	—	—	BTO<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	HS = Hardware set HC = Hardware clear

bit 7-3 **Unimplemented:** Read as '0'

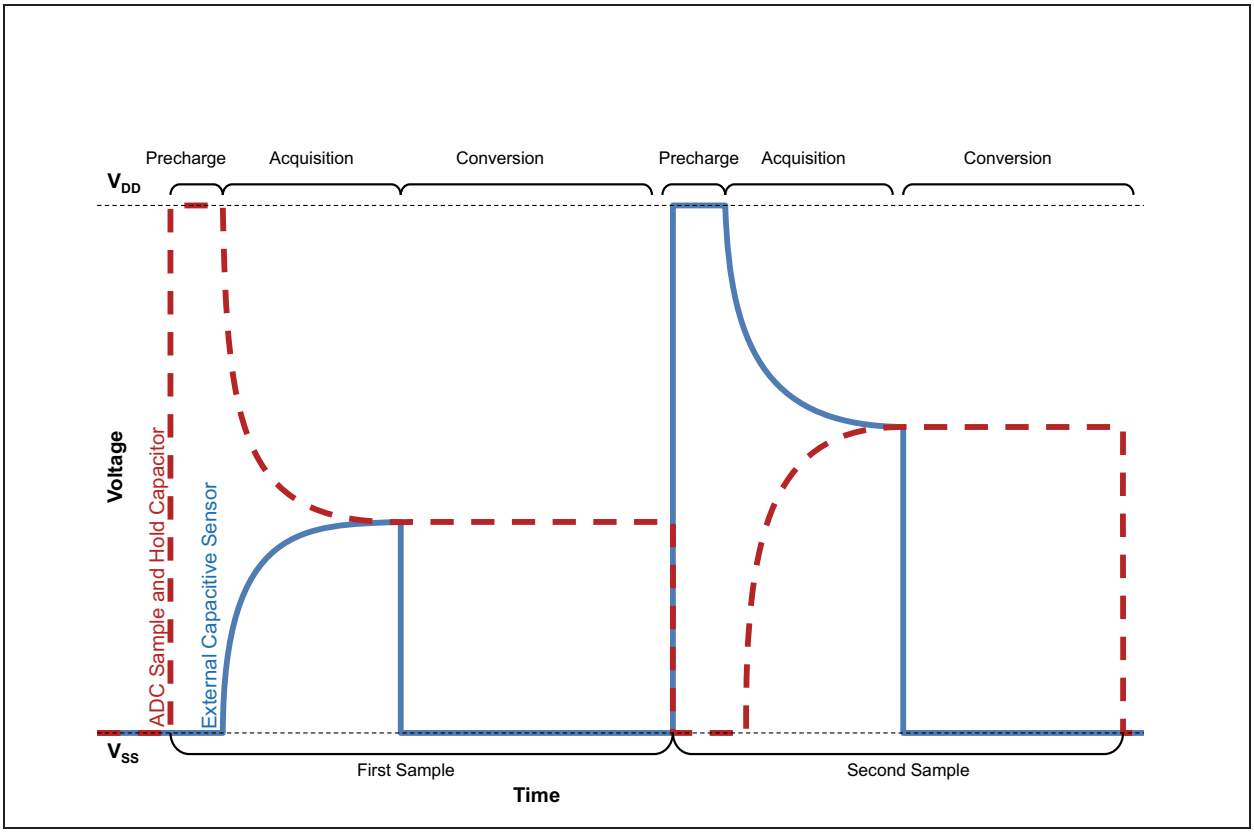
bit 2-0 **BTO<2:0>:** I2C Bus Timeout Selection bits

BTO<2:0>	I²Cx Bus Timeout Selection
111	CLC4OUT
110	CLC3OUT
101	CLC2OUT
100	CLC1OUT
011	TMR6 post scaled output
010	TMR4 post scaled output
001	TMR2 post scaled output
000	Reserved

38.5.1 CVD OPERATION

A CVD operation begins with the ADC's internal sample and hold capacitor (C_{HOLD}) being disconnected from the path which connects it to the external capacitive sensor node. While disconnected, C_{HOLD} is precharged to VDD or VSS, while the path to the sensor node is precharged to the level opposite that of CHOLD. When the precharge phase is complete, the VDD/VSS precharge paths for the two nodes are shut off and CHOLD and the path to the external sensor node are re-connected, at which time the acquisition phase of the CVD operation begins. During acquisition, a capacitive voltage divider is formed between the precharged CHOLD and sensor nodes, which results in a final voltage level setting on CHOLD, which is determined by the capacitances and precharge levels of the two nodes. After acquisition, the ADC converts the voltage level on CHOLD. This process is then repeated with inverted precharge levels for both the CHOLD and external sensor nodes. Figure 38-7 shows the waveform for two inverted CVD measurements, which is known as differential CVD measurement.

FIGURE 38-7: DIFFERENTIAL CVD MEASUREMENT WAVEFORM



REGISTER 38-8: ADPCH: ADC POSITIVE CHANNEL SELECTION REGISTER

U-0	U-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
—	—	ADPCH<5:0>					
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 7-6 **Unimplemented**: Read as '0'

bit 5-0 **ADPCH<5:0>**: ADC Positive Input Channel Selection bits
111111 = FVR Buffer 2(2)
111110 = FVR Buffer 1(2)
111101 = DAC1 output(1)
111100 = Temperature Indicator(3)
111011 = Vss (Analog Ground)
111010 = Reserved. No channel connected.
•
•
•
011000 = Reserved. No channel connected.
010111 = ANC7
010110 = ANC6
010101 = ANC5
010100 = ANC4
010011 = ANC3
010010 = ANC2
010001 = ANC1
010000 = ANC0
001111 = ANB7
001110 = ANB6
001101 = ANB5
001100 = ANB4
001011 = ANB3
001010 = ANB2
001001 = ANB1
001000 = ANB0
000111 = ANA7
000110 = ANA6
000101 = ANA5
000100 = ANA4
000011 = ANA3
000010 = ANA2
000001 = ANA1
000000 = ANA0

Note 1: See **Section 39.0 "5-Bit Digital-to-Analog Converter (DAC) Module"** for more information.
2: See **Section 36.0 "Fixed Voltage Reference (FVR)"** for more information.
3: See **Section 37.0 "Temperature Indicator Module"** for more information.

41.0 HIGH/LOW-VOLTAGE DETECT (HLVD)

The PIC18(L)F2x/4xK42 family of devices has a High/Low-Voltage Detect module (HLVD). This is a programmable circuit that sets both a device voltage trip point and the direction of change from that point (positive going, negative going or both). If the device experiences an excursion past the trip point in that direction, an interrupt flag is set. If the interrupt is enabled, the program execution branches to the interrupt vector address and the software responds to the interrupt.

Complete control of the HLVD module is provided through the HLVDCON0 and HLVDCON1 register. This allows the circuitry to be “turned off” by the user under software control, which minimizes the current consumption for the device.

The module’s block diagram is shown in Figure 41-1.

Since the HLVD can be software enabled through the EN bit, setting and clearing the enable bit does not produce a false HLVD event glitch. Each time the HLVD module is enabled, the circuitry requires some time to stabilize. The RDY bit (HLVDCON0<4>) is a read-only bit used to indicate when the band gap reference voltages are stable.

The module can only generate an interrupt after the module is turned ON and the band gap reference voltages are ready.

The INTH and INTL bits determine the overall operation of the module. When INTH is set, the module monitors for rises in V_{DD} above the trip point set by the HLVDCON1 register. When INTL is set, the module monitors for drops in V_{DD} below the trip point set by the HLVDCON1 register. When both the INTH and INTL bits are set, any changes above or below the trip point set by the HLVDCON1 register can be monitored.

The OUT bit can be read to determine if the voltage is greater than or less than the voltage level selected by the HLVDCON1 register.