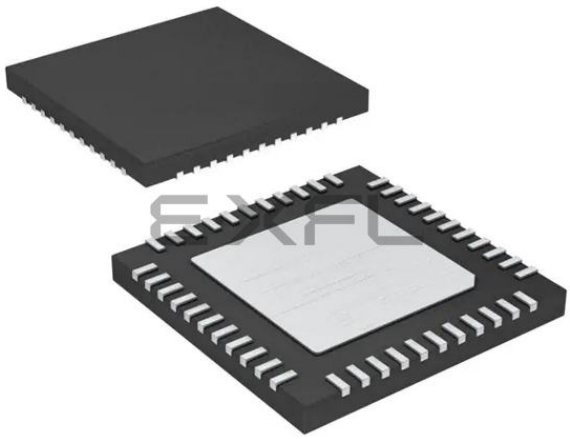


Welcome to E-XFL.COM

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, LVD, POR, PWM, WDT
Number of I/O	36
Program Memory Size	64KB (32K x 16)
Program Memory Type	FLASH
EEPROM Size	1K x 8
RAM Size	3.6K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 35x10b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	44-VQFN Exposed Pad
Supplier Device Package	44-QFN (8x8)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf46k40-e-ml

TABLE 4-3: SUMMARY OF REGISTERS ASSOCIATED WITH CLOCK SOURCES

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
OSCCON1	—	NOSC<2:0>			NDIV<3:0>				36
OSCCON2	—	COSC<2:0>			CDIV<3:0>				37
OSCCON3	CSWHOLD	SOSCPWR	—	ORDY	NOSCR	—	—	—	38
OSCSTAT	EXTOR	HFOR	MFOR	LFOR	SOR	ADOR	—	PLLR	39
OSCTUNE	—	—	HFTUN<5:0>						41
OSCFRQ	—	—	—	—	HFFRQ<3:0>				40
OSCEN	EXTOEN	HFOEN	MFOEN	LFOEN	SOSCEN	ADOEN	—	—	42
PIE1	OSCFIE	CSWIE	—	—	—	—	ADTIE	ADIE	180
PIR1	OSCFIF	CSWIF	—	—	—	—	ADTIF	ADIF	172
IPR1	OSCFIP	CSWIP	—	—	—	—	ADTIP	ADIP	188

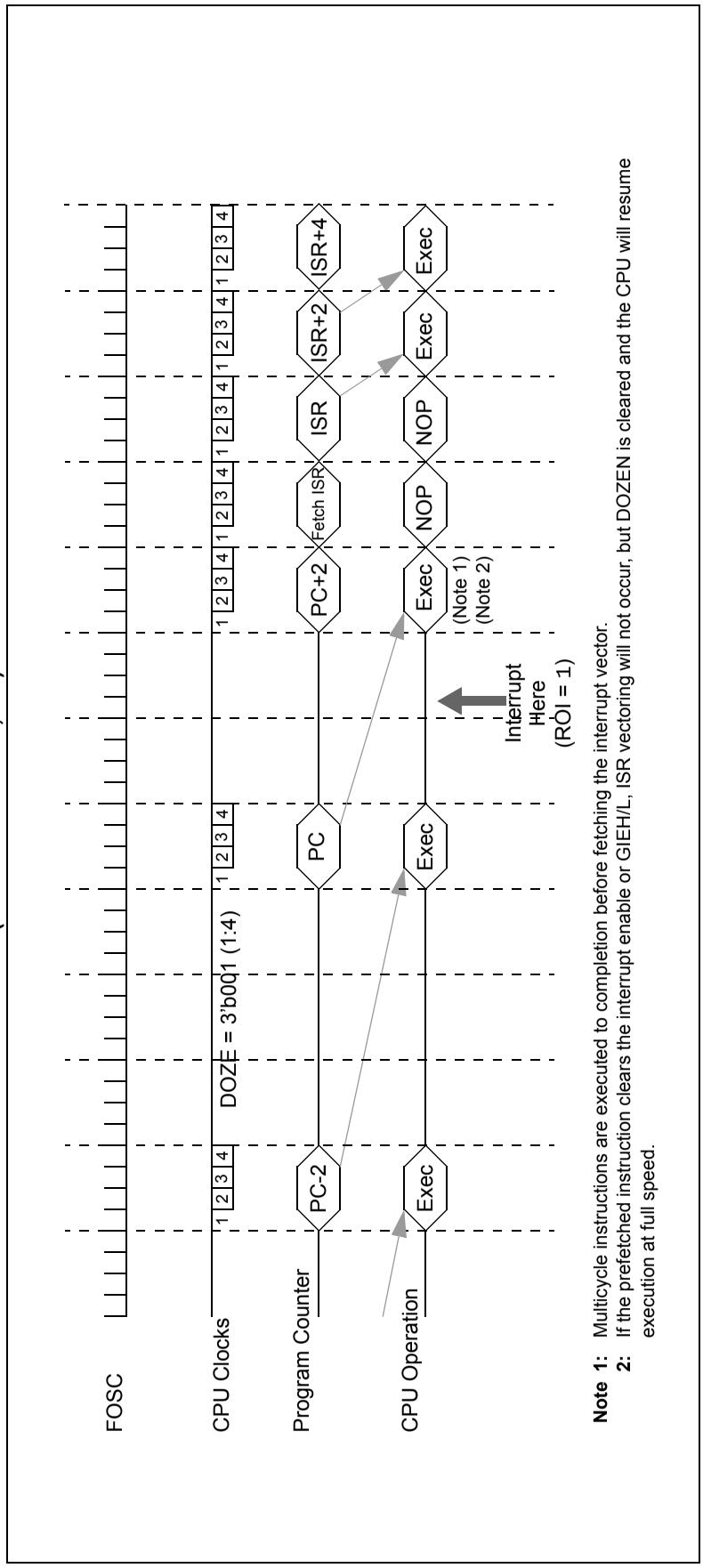
Legend: — = unimplemented location, read as '0'. Shaded cells are not used by clock sources.

TABLE 4-4: SUMMARY OF CONFIGURATION WORD WITH CLOCK SOURCES

Name	Bits	Bit -/7	Bit -/6	Bit 13/5	Bit 12/4	Bit 11/3	Bit 10/2	Bit 9/1	Bit 8/0	Register on Page
CONFIG1	13:8	—	—	FCMEN	—	CSWEN	—	—	$\overline{\text{CLKOUTEN}}$	23
	7:0	—	RSTOSC<2:0>			—	FEXTOSC<2:0>			

Legend: — = unimplemented location, read as '0'. Shaded cells are not used by clock sources.

FIGURE 6-1: DOZE MODE OPERATION EXAMPLE (DOZE<2:0> = 001, 1:4)



Note 1: Multicycle instructions are executed to completion before fetching the interrupt vector.
2: If the prefetched instruction clears the interrupt enable or GIEH/L, ISR vectoring will not occur, but DOZEN is cleared and the CPU will resume execution at full speed.

TABLE 10-5: REGISTER FILE SUMMARY FOR PIC18(L)F26/45/46K40 DEVICES (CONTINUED)

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR
EB1h	CWGINPPS	—	—	—	CWGINPPS<4:0>					---01000
EB0h	CCP2PPS	—	—	—	CCP2PPS<4:0>					---10001
EAFh	CCP1PPS	—	—	—	CCP1PPS<4:0>					---10010
EAEh	ADACTPPS	—	—	—	ADACTPPS<4:0>					---01100
EADh	T6INPPS	—	—	—	T6INPPS<4:0>					---01111
EACh	T4INPPS	—	—	—	T4INPPS<4:0>					---10101
EABh	T2INPPS	—	—	—	T2INPPS<4:0>					---10011
EAAh	T5GPPS	—	—	—	T5GPPS<4:0>					---01100
EA9h	T5CKIPPS	—	—	—	T5CKIPPS<4:0>					---10010
EA8h	T3GPPS	—	—	—	T3GPPS<4:0>					---10000
EA7h	T3CKIPPS	—	—	—	T3CKIPPS<4:0>					---10000
EA6h	T1GPPS	—	—	—	T1GPPS<4:0>					---01101
EA5h	T1CKIPPS	—	—	—	T1CKIPPS<4:0>					---10000
EA4h	T0CKIPPS	—	—	—	T0CKIPPS<4:0>					---00100
EA3h	INT2PPS	—	—	—	INT2PPS<4:0>					---01010
EA2h	INT1PPS	—	—	—	INT1PPS<4:0>					---01001
EA1h	INT0PPS	—	—	—	INT0PPS<4:0>					---01000
EA0h	PPSLOCK	—	—	—	—	—	—	—	PPSLOCKED	-------0
E9Fh	BAUD2CON	ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN	01-00-00
E9Eh	TX2STA	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D	00000010
E9Dh	RC2STA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	00000000
E9Ch	SP2BRGH	EUSART2 Baud Rate Generator, High Byte								00000000
E9Bh	SP2BRGL	EUSART2 Baud Rate Generator, Low Byte								00000000
E9Ah	TX2REG	EUSART2 Transmit Register								00000000
E99h	RC2REG	EUSART2 Receive Register								00000000
E98h	SSP2CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN	00000000
E97h	SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN	00000000
E96h	SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM<3:0>				00000000
E95h	SSP2STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	00000000
E94h	SSP2MSK	MSK<7:0>								11111111
E93h	SSP2ADD	ADD<7:0>								00000000
E92h	SSP2BUF	BUF<7:0>								xxxxxxxx
E91h	SSP2SSPPS	—	—	—	SSPSSPPS<4:0>					---00101
E90h	SSP2DATPPS	—	—	—	SSPDATPPS<4:0>					---10100
E8Fh	SSP2CLKPPS	—	—	—	SSPCLKPPS<4:0>					---10011
E8Eh	TX2PPS	—	—	—	TXPPS<4:0>					---10110
E8Dh	RX2PPS	—	—	—	RXPPS<4:0>					---10111
E8Ch – E7Eh	—	Unimplemented								—

Legend: x = unknown, u = unchanged, — = unimplemented, q = value depends on condition

Note 1: Not available on LF devices.

2: Not available on PIC18(L)F26K40 (28-pin variants).

3: Not available on PIC18(L)F45K40 devices.

11.1.1 TABLE READS AND TABLE WRITES

In order to read and write program memory, there are two operations that allow the processor to move bytes between the program memory space and the data RAM:

- Table Read (TBLRD)
- Table Write (TBLWT)

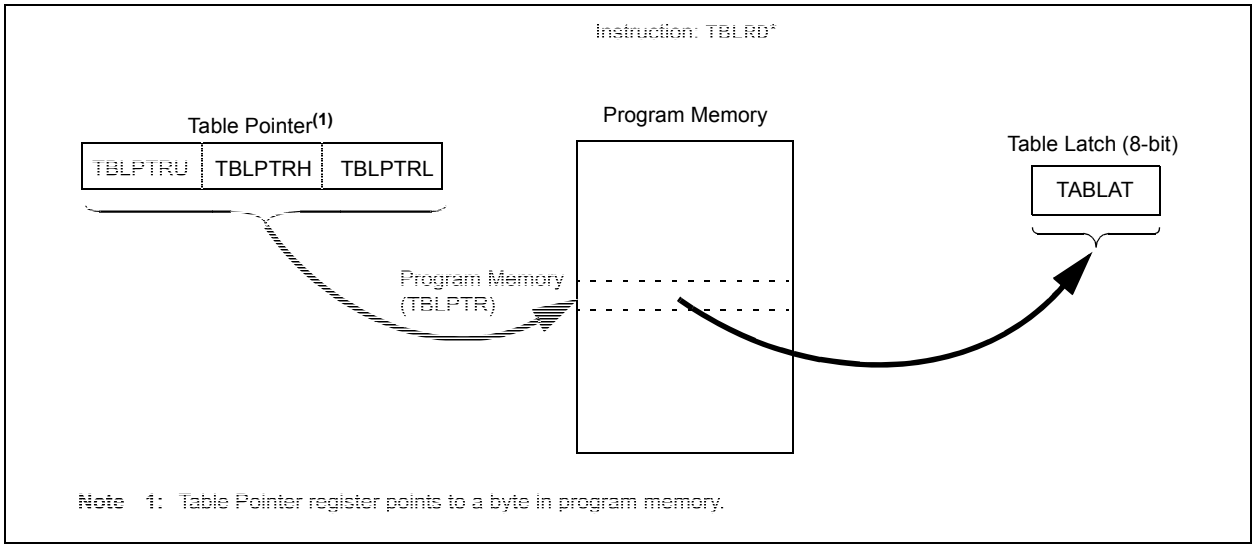
The program memory space is 16 bits wide, while the data RAM space is eight bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

The table read operation retrieves one byte of data directly from program memory and places it into the TABLAT register. Figure 11-1 shows the operation of a table read.

The table write operation stores one byte of data from the TABLAT register into a write block holding register. The procedure to write the contents of the holding registers into program memory is detailed in **Section 11.1.6 "Writing to Program Flash Memory"**. Figure 11-2 shows the operation of a table write with program memory and data RAM.

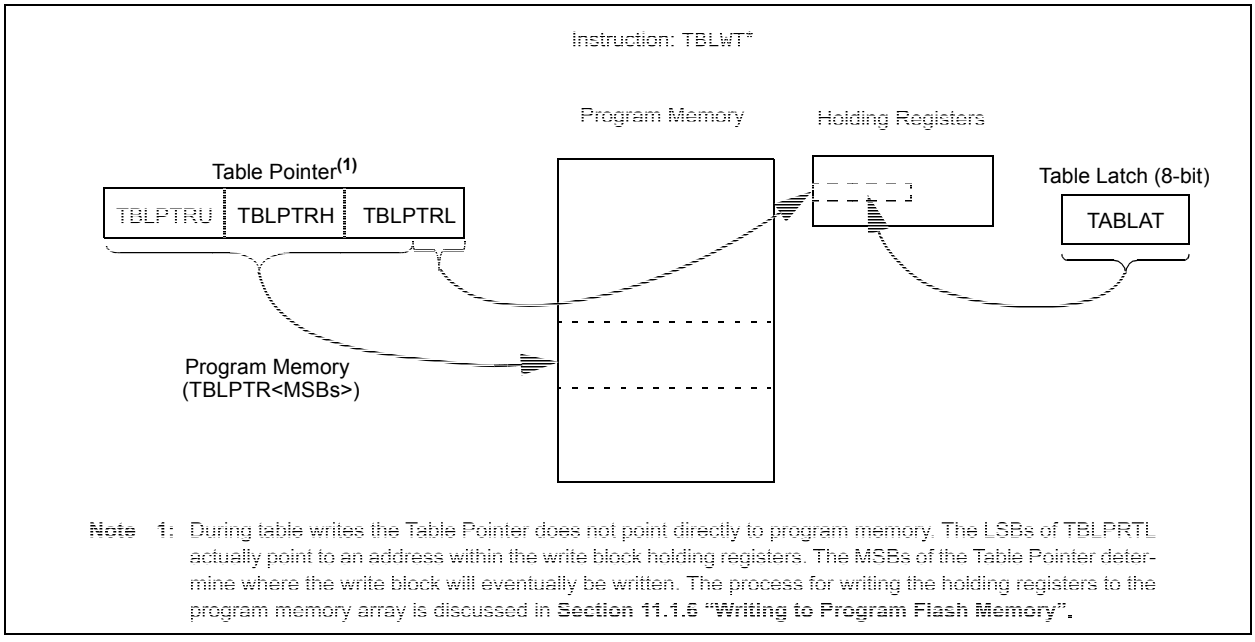
Table operations work with byte entities. Tables containing data, rather than program instructions, are not required to be word aligned. Therefore, a table can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions will need to be word aligned.

FIGURE 11-1: TABLE READ OPERATION



Note 1: Table Pointer register points to a byte in program memory.

FIGURE 11-2: TABLE WRITE OPERATION



Note 1: During table writes the Table Pointer does not point directly to program memory. The LSBs of TBLPRTL actually point to an address within the write block holding registers. The MSBs of the Table Pointer determine where the write block will eventually be written. The process for writing the holding registers to the program memory array is discussed in **Section 11.1.6 "Writing to Program Flash Memory"**.

11.3 Data EEPROM Memory

The data EEPROM is a nonvolatile memory array, separate from the data RAM and program memory, which is used for long-term storage of program data. It is not directly mapped in either the register file or program memory space but is indirectly addressed through the Special Function Registers (SFRs). The EEPROM is readable and writable during normal operation over the entire VDD range.

Four SFRs are used to read and write to the data EEPROM as well as the program memory. They are:

- NVMCON1
- NVMCON2
- NVMDAT
- NVMADRL
- NVMADRH(1)

Note 1: NVMADRH register is not implemented on PIC18(L)F45K40.

The data EEPROM allows byte read and write. When interfacing to the data memory block, NVMDAT holds the 8-bit data for read/write and the NVMADRH:NVMADRL register pair hold the address of the EEPROM location being accessed.

The EEPROM data memory is rated for high erase/write cycle endurance. A byte write automatically erases the location and writes the new data (erase-before-write). The write time is controlled by an internal programming timer; it will vary with voltage and temperature as well as from chip-to-chip. Please refer to the Data EEPROM Memory parameters in **Section 37.0 "Electrical Specifications"** for limits.

11.3.1 NVMADRL AND NVMADRH REGISTERS

The NVMADRH:NVMADRL registers are used to address the data EEPROM for read and write operations.

11.3.2 NVMCON1 AND NVMCON2 REGISTERS

Access to the data EEPROM is controlled by two registers: NVMCON1 and NVMCON2. These are the same registers which control access to the program memory and are used in a similar manner for the data EEPROM.

The NVMCON1 register (Register 11-1) is the control register for data and program memory access. Control bits NVMREG<1:0> determine if the access will be to program, Data EEPROM Memory or the User IDs, Configuration bits, Revision ID and Device ID.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear.

The WRERR bit is set by hardware when the WR bit is set and cleared when the internal programming timer expires and the write operation is complete.

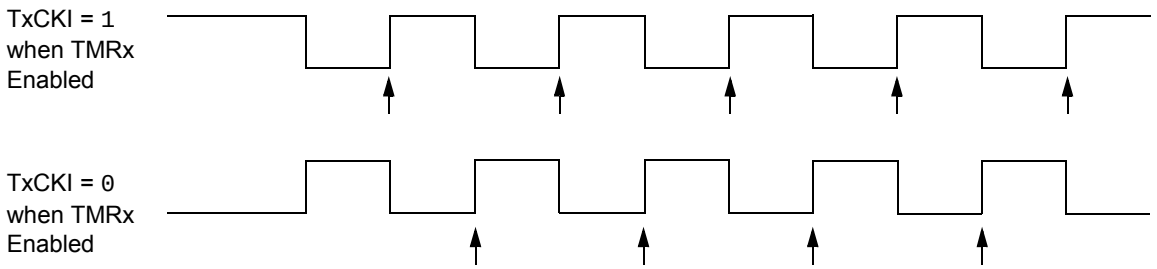
The WR control bit initiates write operations. The bit can be set but not cleared by software. It is cleared only by hardware at the completion of the write operation.

The NVMIF interrupt flag bit of the PIR7 register is set when the write is complete. It must be cleared by software.

Control bits, RD and WR, start read and erase/write operations, respectively. These bits are set by firmware and cleared by hardware at the completion of the operation.

The RD bit cannot be set when accessing program memory (NVMREG<1:0> = 0x10). Program memory is read using table read instructions. See **Section 11.1.1 "Table Reads and Table Writes"** regarding table reads.

FIGURE 19-3: TIMER1/3/5 INCREMENTING EDGE



Note 1: Arrows indicate counter increments.

2: In Counter mode, a falling edge must be registered by the counter prior to the first incrementing rising edge of the clock.

FIGURE 19-4: TIMER1/3/5 GATE ENABLE MODE

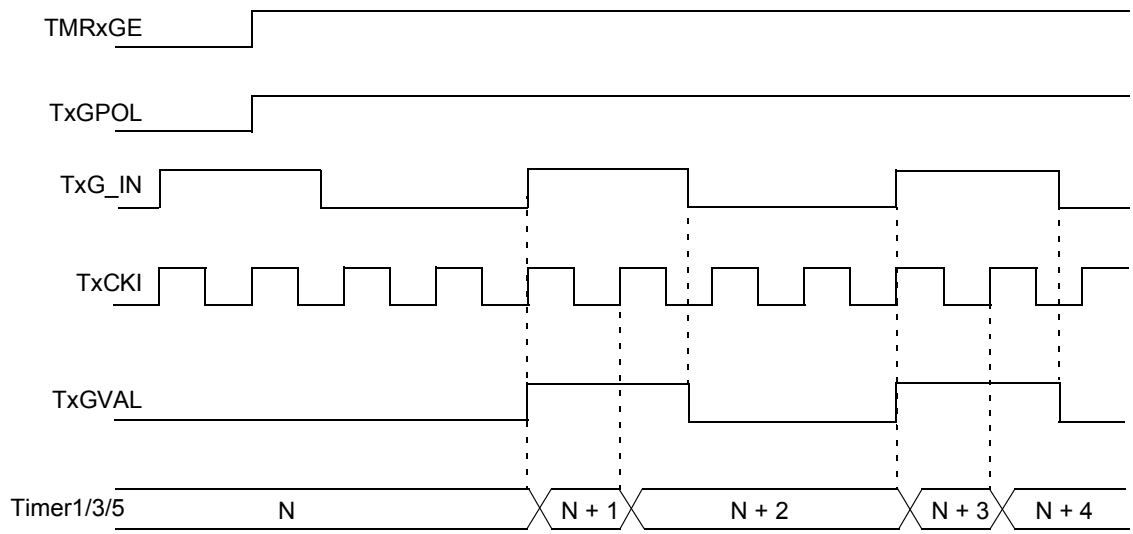


TABLE 20-3: SUMMARY OF REGISTERS ASSOCIATED WITH TIMER2

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
INTCON	GIE/GIEH	PEIE/GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG	170
PIE4	—	—	TMR6IE	TMR5IE	TMR4IE	TMR3IE	TMR2IE	TMR1IE	183
PIR4	—	—	TMR6IF	TMR5IF	TMR4IF	TMR3IF	TMR2IF	TMR1IF	175
IPR4	—	—	TMR6IP	TMR5IP	TMR4IP	TMR3IP	TMR2IP	TMR1IP	191
PMD1	—	TMR6MD	TMR5MD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD	69
PR2	Timer2 Module Period Register								244*
TMR2	Holding Register for the 8-bit TMR2 Register								244*
T2CON	ON	CKPS<2:0>			OUTPS<3:0>				262
T2CLKCON	—	—	—	—	—	CS<2:0>			264
T2RST	—	—	—	—	RSEL<3:0>				265
T2HLT	PSYNC	CPOL	CSYNC	MODE<4:0>					263
PR4	Timer4 Module Period Register								244*
TMR4	Holding Register for the 8-bit TMR4 Register								244*
T4CON	ON	CKPS<2:0>			OUTPS<3:0>				262
T4CLKCON	—	—	—	—	—	CS<2:0>			264
T4RST	—	—	—	—	RSEL<3:0>				265
T4HLT	PSYNC	CPOL	CSYNC	MODE<4:0>					263
PR6	Timer6 Module Period Register								244*
TMR6	Holding Register for the 8-bit TMR6 Register								244*
T6CON	ON	CKPS<2:0>			OUTPS<3:0>				262
T6CLKCON	—	—	—	—	—	CS<2:0>			264
T6RST	—	—	—	—	RSEL<3:0>				265
T6HLT	PSYNC	CPOL	CSYNC	MODE<4:0>					263
T2INPPS	—	—	—	T2INPPS<4:0>					216
T4INPPS	—	—	—	T4INPPS<4:0>					216
T6INPPS	—	—	—	T6INPPS<4:0>					216

Legend: — = unimplemented location, read as '0'. Shaded cells are not used for Timer2 module.

* Page provides register information.

REGISTER 22-3: PWMxDCH: PWM DUTY CYCLE HIGH BITS

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
DCH<7:0>							
bit 7							bit 0

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

bit 7-0 **DCh<7:0>:** PWM Duty Cycle Most Significant bits
These bits are the MSbs of the PWM duty cycle. The two LSbs are found in PWMxDCL Register.

REGISTER 22-4: PWMxDCL: PWM DUTY CYCLE LOW BITS

R/W-x/u	R/W-x/u	U-0	U-0	U-0	U-0	U-0	U-0
DCL<7:6>		—	—	—	—	—	—
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 **DC<8:9>:** PWM Duty Cycle Least Significant bits
These bits are the LSbs of the PWM duty cycle. The MSbs are found in PWMxDCH Register.

bit 5-0 **Unimplemented:** Read as ‘0’

23.10 Register Definitions: ZCD Control

REGISTER 23-1: ZCDCON: ZERO-CROSS DETECT CONTROL REGISTER

R/W-0/0	U-0	R-x	R/W-0/0	U-0	U-0	R/W-0/0	R/W-0/0
ZCDSEN	—	ZCDOUT	ZCDPOL	—	—	ZCDINTP	ZCDINTN
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 **ZCDSEN:** Zero-Cross Detect Software Enable bit
This bit is ignored when ZCDSEN fuse is set.
1= Zero-cross detect is enabled. ZCD pin is forced to output to source and sink current.
0= Zero-cross detect is disabled. ZCD pin operates according to PPS and TRIS controls.

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

bit 5 **ZCDOUT:** Zero-Cross Detect Data Output bit
ZCDPOL bit = 0:
1 = ZCD pin is sinking current
0 = ZCD pin is sourcing current
ZCDPOL bit = 1:
1 = ZCD pin is sourcing current
0 = ZCD pin is sinking current

bit 4 **ZCDPOL:** Zero-Cross Detect Polarity bit
1 = ZCD logic output is inverted
0 = ZCD logic output is not inverted

bit 3-2 **Unimplemented:** Read as '0'

bit 1 **ZCDINTP:** Zero-Cross Detect Positive-Going Edge Interrupt Enable bit
1 = ZCDIF bit is set on low-to-high ZCD_output transition
0 = ZCDIF bit is unaffected by low-to-high ZCD_output transition

bit 0 **ZCDINTN:** Zero-Cross Detect Negative-Going Edge Interrupt Enable bit
1 = ZCDIF bit is set on high-to-low ZCD_output transition
0 = ZCDIF bit is unaffected by high-to-low ZCD_output transition

24.2.3.1 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 CWG1CON0 register. The sequence is illustrated in Figure 24-8.

- The associated active output CWG1A and the inactive output CWG1C are switched to drive in the opposite direction.
- The previously modulated output CWG1D is switched to the inactive state, and the previously inactive output CWG1B begins to modulate.
- CWG modulation resumes after the direction-switch dead band has elapsed.

24.2.3.2 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 (CWG1A and CWG1C) are not afforded dead band, and switch essentially simultaneously.

Figure 24-8 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 CWG1A and CWG1D become inactive, while output CWG1C 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 24-8: EXAMPLE OF PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE

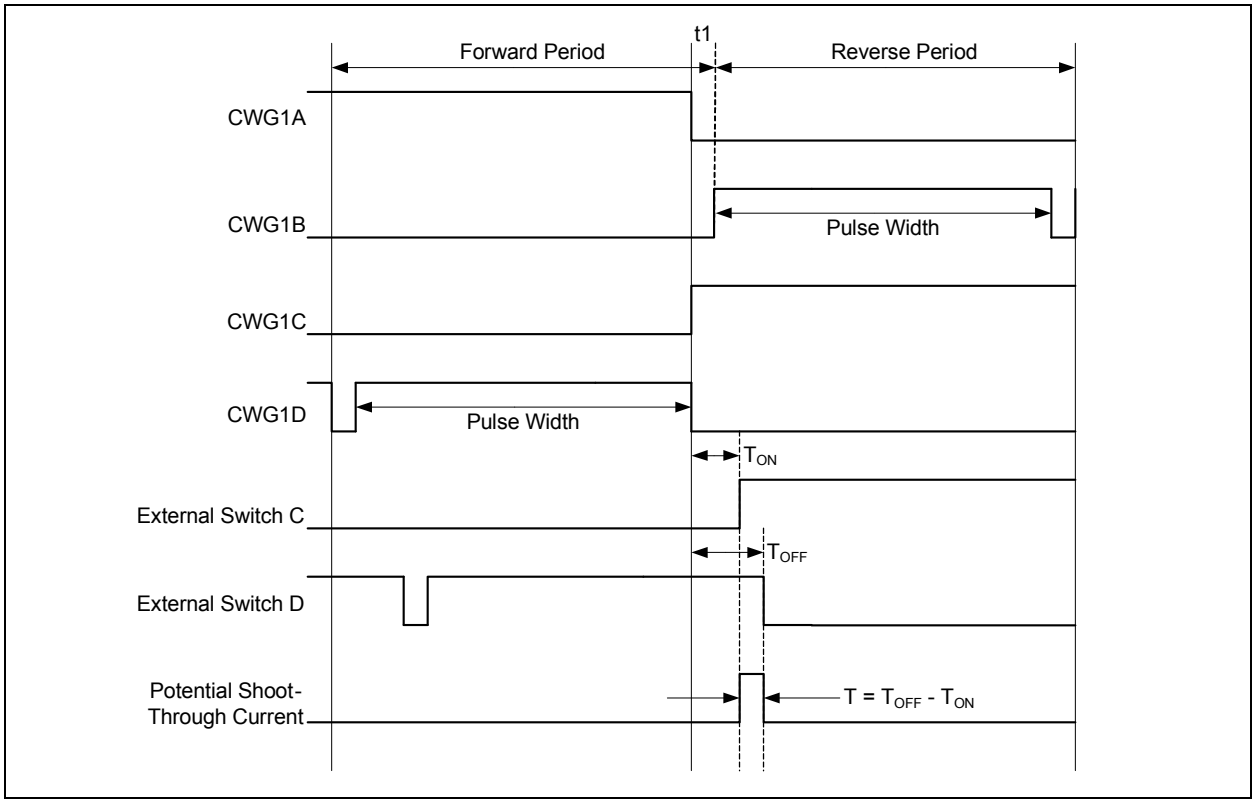


TABLE 24-3: SUMMARY OF REGISTERS ASSOCIATED WITH CWG

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Register on Page
CWG1CON0	EN	LD	—	—	—	MODE<2:0>			315
CWG1CON1	—	—	IN	—	POLD	POLC	POLB	POLA	316
CWG1CLKCON	—	—	—	—	—	—	—	CS	317
CWG1ISM	—	—	—	—	—	ISM<2:0>			317
CWG1STR	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA	318
CWG1AS0	SHUTDOWN	REN	LSBD<1:0>		LSAC<1:0>		—	—	319
CWG1AS1	—	—	AS5E	AS4E	AS3E	AS2E	AS1E	AS0E	320
CWG1DBR	—	—	DBR<5:0>						321
CWG1DBF	—	—	DBF<5:0>						321
PIE7	SCANIE	CRCIE	NVMIE	—	—	—	—	CWG1IE	186
PIR7	SCANIF	CRCIF	NVMIF	—	—	—	—	CWG1IF	178
IPR7	SCANIP	CRCIP	NVMIP	—	—	—	—	CWG1IP	194
PMD4	UART2MD	UART1MD	MSSP2MD	MSSP1MD	—	—	—	CWG1MD	72

Legend: – = unimplemented locations read as '0'. Shaded cells are not used by CWG.

26.5.1 SPI MASTER MODE

The master can initiate the data transfer at any time because it controls the SCK line. The master determines when the slave (Processor 2, Figure 26-3) is to broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPxBUF register is written to. If the SPI is only going to receive, the SDO output could be disabled (programmed as an input). The SSPSR register will continue to shift in the signal present on the SDI pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPxBUF register as if a normal received byte (interrupts and Status bits appropriately set).

The clock polarity is selected by appropriately programming the CKP bit of the SSPxCON1 register and the CKE bit of the SSPxSTAT register. This then, would give waveforms for SPI communication as shown in Figure 26-4, Figure 26-6, Figure 26-7 and Figure 26-8, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user programmable to be one of the following:

- FOSC/4 (or TCY)
- FOSC/16 (or 4 * TCY)
- FOSC/64 (or 16 * TCY)
- Timer2 output/2
- FOSC/(4 * (SSPxADD + 1))

Figure 26-4 shows the waveforms for Master mode.

When the CKE bit is set, the SDO data is valid before there is a clock edge on SCK. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPxBUF is loaded with the received data is shown.

Note:	In Master mode the clock signal output to the SCK pin is also the clock signal input to the peripheral. The pin selected for output with the RxyPPS register must also be selected as the peripheral input with the SSPxCLKPPS register. The pin that is selected using the SSPxCLKPPS register should also be made a digital I/O. This is done by clearing the corresponding ANSEL bit.

FIGURE 26-7: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

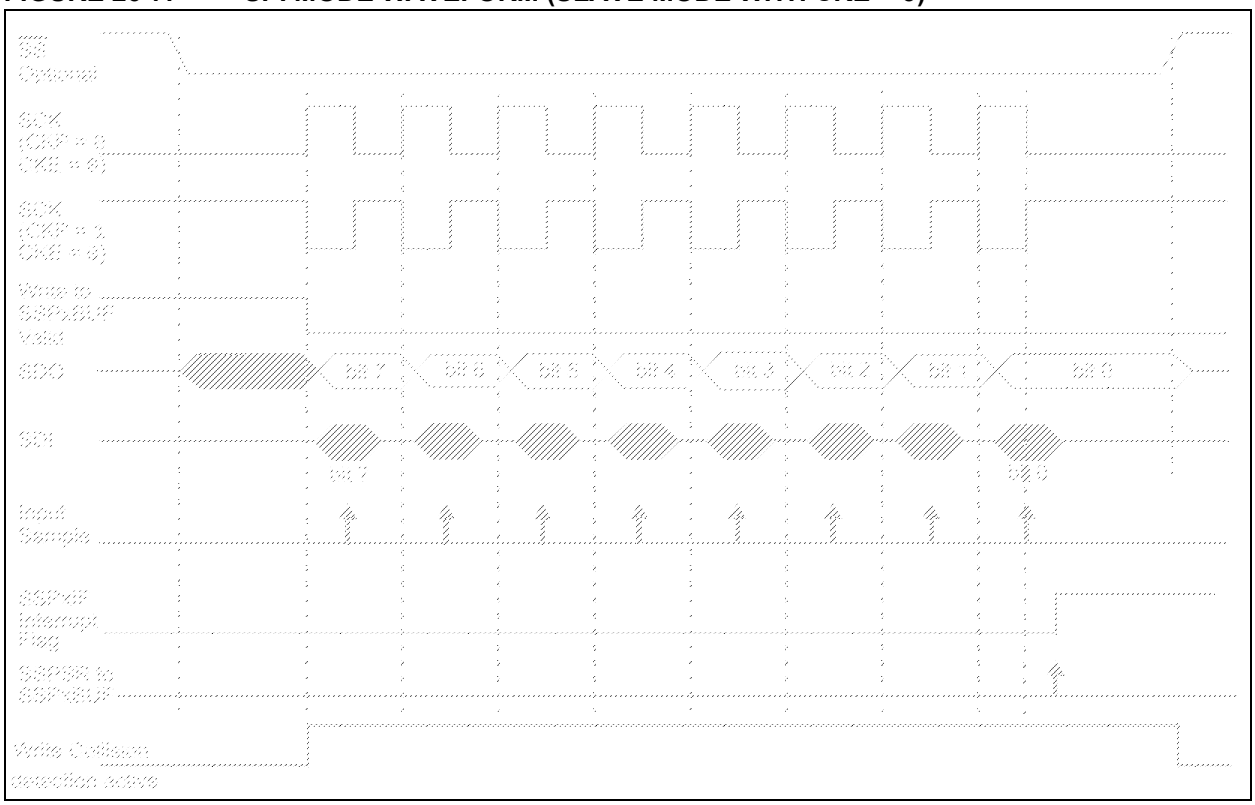


FIGURE 26-8: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)

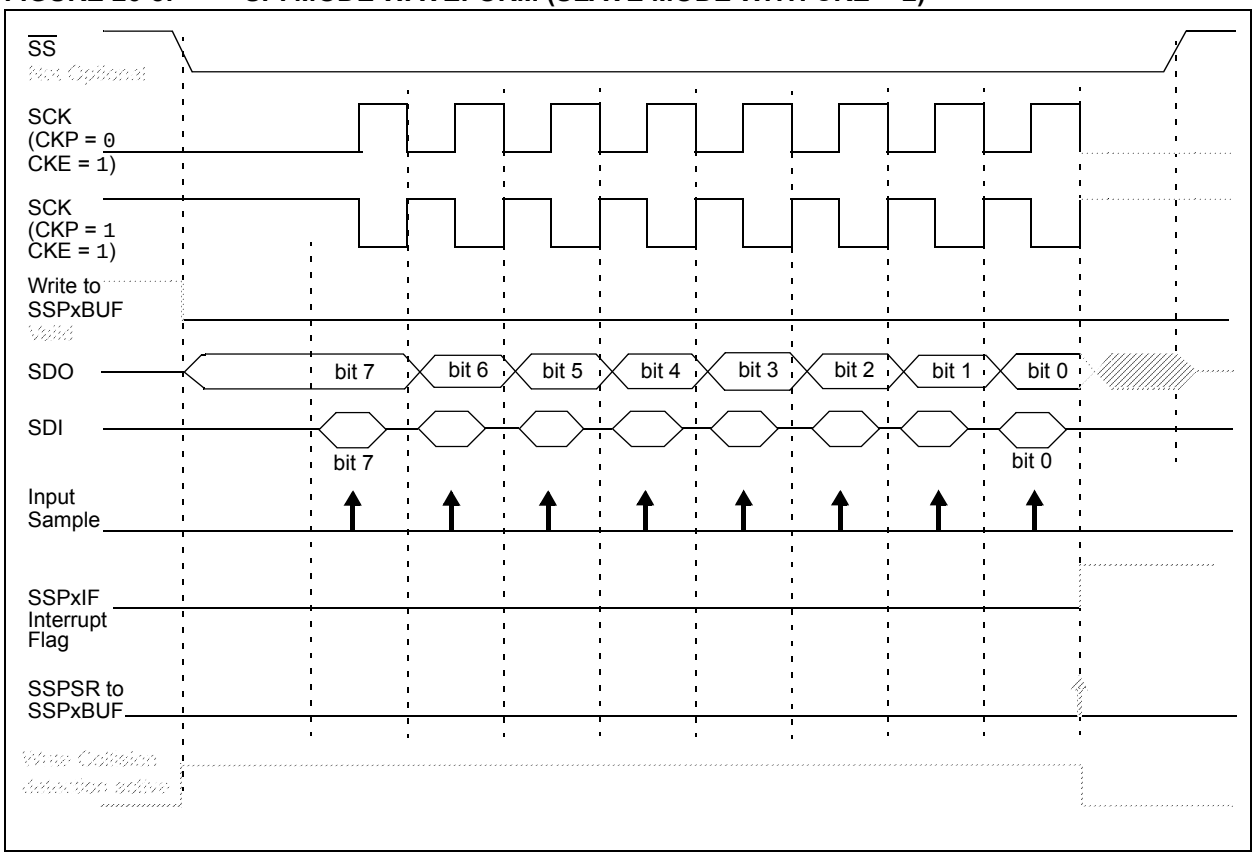
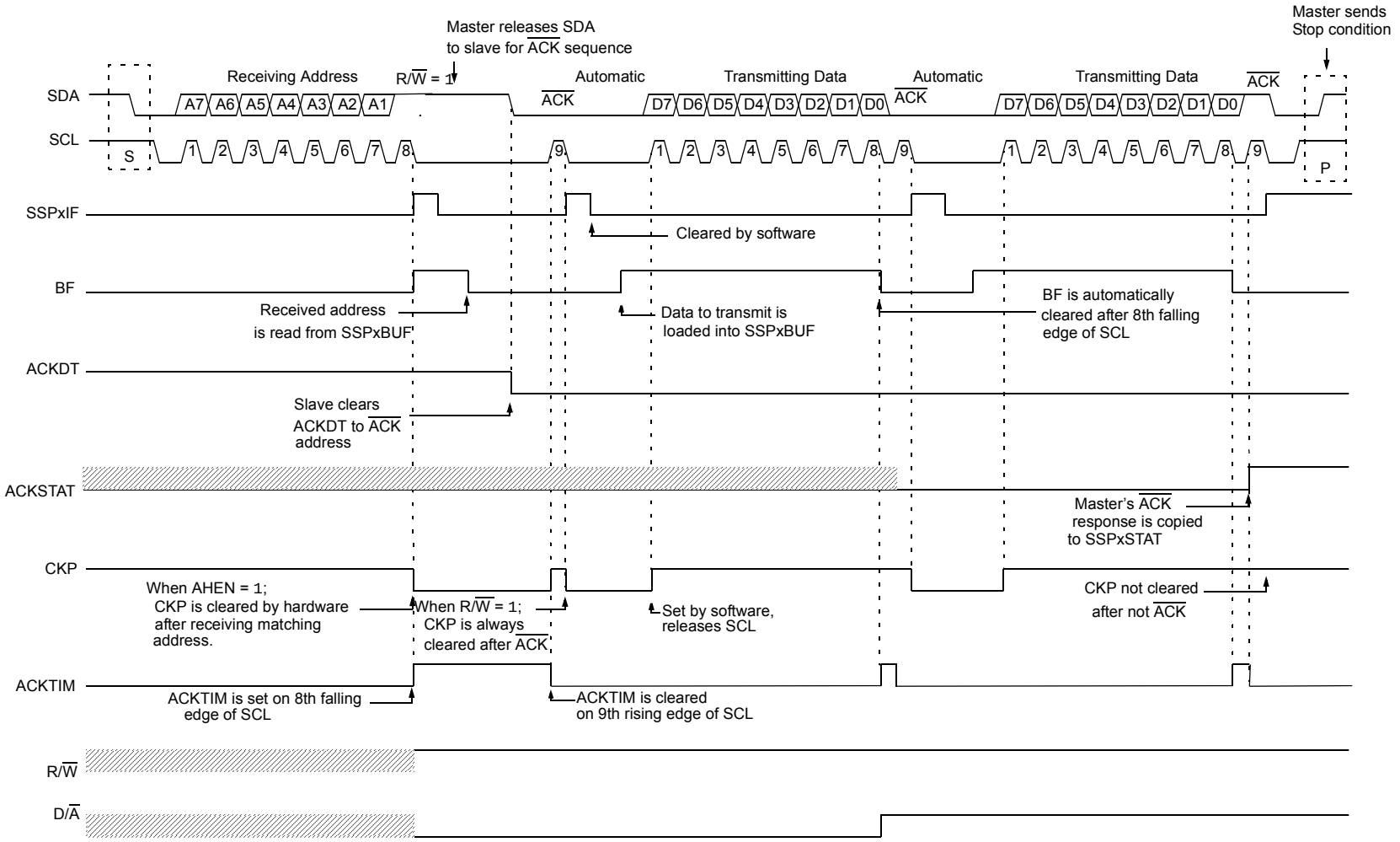


FIGURE 26-19: I²C SLAVE, 7-BIT ADDRESS, TRANSMISSION (AHEN = 1)



27.2.2 EUSART ASYNCHRONOUS RECEIVER

The Asynchronous mode is typically used in RS-232 systems. The receiver block diagram is shown in Figure 27-2. The data is received on the RXx/DTx pin and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at 16 times the baud rate, whereas the serial Receive Shift Register (RSR) operates at the bit rate. When all eight or nine bits of the character have been shifted in, they are immediately transferred to a two character First-In-First-Out (FIFO) memory. The FIFO buffering allows reception of two complete characters and the start of a third character before software must start servicing the EUSART receiver. The FIFO and RSR registers are not directly accessible by software. Access to the received data is via the RCxREG register.

27.2.2.1 Enabling the Receiver

The EUSART receiver is enabled for asynchronous operation by configuring the following three control bits:

- CREN = 1
- SYNC = 0
- SPEN = 1

All other EUSART control bits are assumed to be in their default state.

Setting the CREN bit of the RCxSTA register enables the receiver circuitry of the EUSART. Clearing the SYNC bit of the TXxSTA register configures the EUSART for asynchronous operation. Setting the SPEN bit of the RCxSTA register enables the EUSART. The programmer must set the corresponding TRIS bit to configure the RXx/DTx I/O pin as an input.

Note:	If the RX/DT function is on an analog pin, the corresponding ANSEL bit must be cleared for the receiver to function.

27.2.2.2 Receiving Data

The receiver data recovery circuit initiates character reception on the falling edge of the first bit. The first bit, also known as the Start bit, is always a zero. The data recovery circuit counts one-half bit time to the center of the Start bit and verifies that the bit is still a zero. If it is not a zero then the data recovery circuit aborts character reception, without generating an error, and resumes looking for the falling edge of the Start bit. If the Start bit zero verification succeeds then the data recovery circuit counts a full bit time to the center of the next bit. The bit is then sampled by a majority detect circuit and the resulting '0' or '1' is shifted into the RSR. This repeats until all data bits have been sampled and shifted into the RSR. One final bit time is measured and the level sampled. This is the Stop bit, which is always a '1'. If the data recovery circuit samples a '0' in the Stop bit position then a framing error is set for this character, otherwise the framing error is cleared for this character. See **Section 27.2.2.4 "Receive Framing Error"** for more information on framing errors.

Immediately after all data bits and the Stop bit have been received, the character in the RSR is transferred to the EUSART receive FIFO and the RCxIF interrupt flag bit of the PIR3 register is set. The top character in the FIFO is transferred out of the FIFO by reading the RCxREG register.

Note:	If the receive FIFO is overrun, no additional characters will be received until the overrun condition is cleared. See **Section 27.2.2.5 "Receive Overrun Error"** for more information on overrun errors.

27.2.2.3 Receive Interrupts

The RCxIF interrupt flag bit of the PIR3 register is set whenever the EUSART receiver is enabled and there is an unread character in the receive FIFO. The RCxIF interrupt flag bit is read-only, it cannot be set or cleared by software.

RCxIF interrupts are enabled by setting all of the following bits:

- RCxIE, Interrupt Enable bit of the PIE3 register
- PEIE, Peripheral Interrupt Enable bit of the INTCON register
- GIE, Global Interrupt Enable bit of the INTCON register

The RCxIF interrupt flag bit will be set when there is an unread character in the FIFO, regardless of the state of interrupt enable bits.

30.0 5-BIT DIGITAL-TO-ANALOG CONVERTER (DAC) MODULE

The Digital-to-Analog Converter supplies a variable voltage reference, ratiometric with the input source, with 32 selectable output levels.

The positive input source (VSOURCE+) of the DAC can be connected to:

- FVR Buffer
- External VREF+ pin
- VDD supply voltage

The negative input source (VSOURCE-) of the DAC can be connected to:

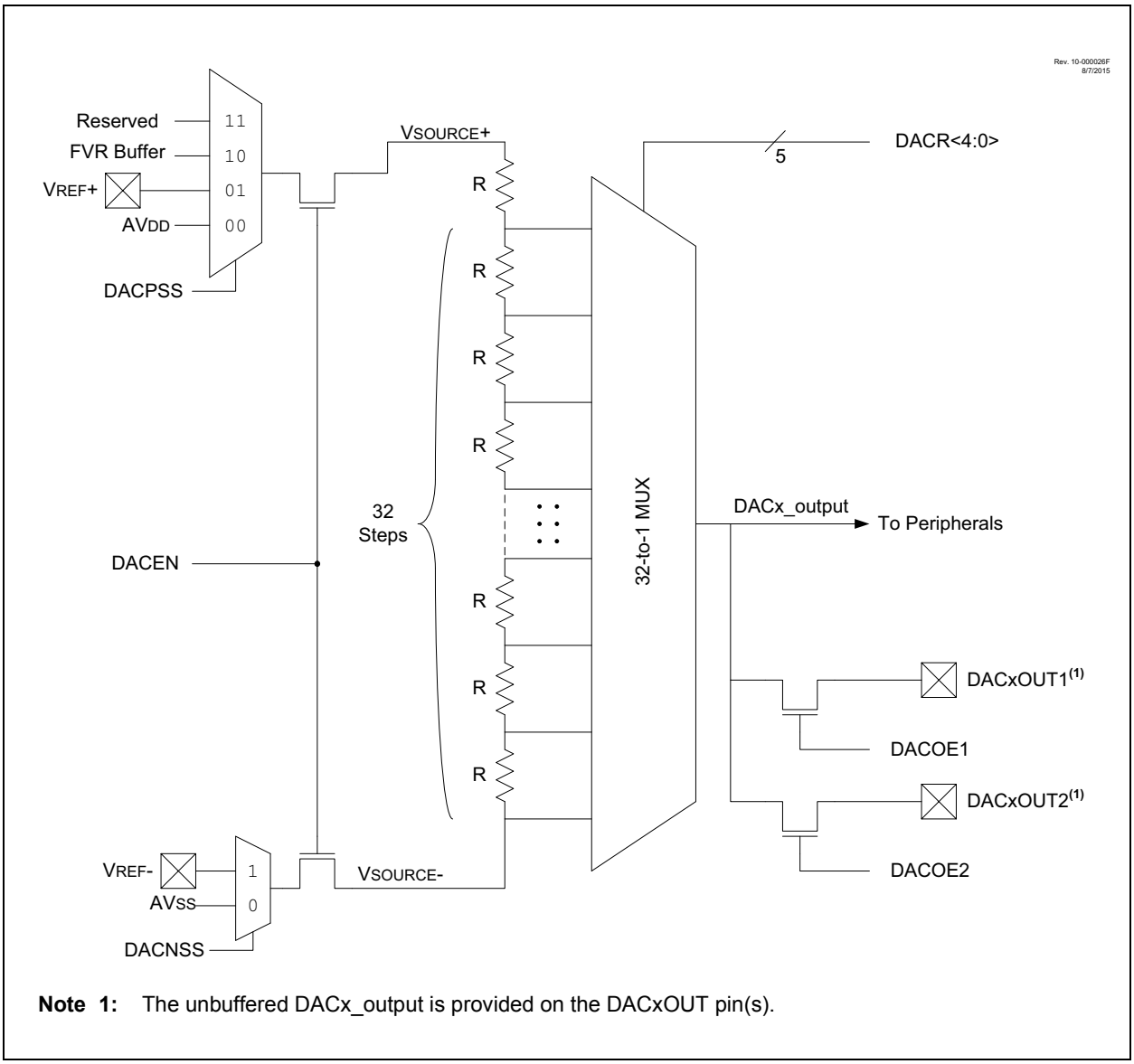
- External VREF- pin
- Vss

The output of the DAC (DACx_output) can be selected as a reference voltage to the following:

- Comparator positive input
- ADC input channel
- DACxOUT1 pin
- DACxOUT2 pin

The Digital-to-Analog Converter (DAC) can be enabled by setting the DACEN bit of the DAC1CON0 register.

FIGURE 30-1: DIGITAL-TO-ANALOG CONVERTER BLOCK DIAGRAM



Note 1: The unbuffered DACx_output is provided on the DACxOUT pin(s).

35.1.1 STANDARD INSTRUCTION SET

ADDLW **ADD literal to W**

Syntax: ADDLW k

Operands: $0 \le k \le 255$

Operation: $(W) + k \rightarrow W$

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

Encoding:

0000	1111	kkkk	kkkk

Description: The contents of W are added to the 8-bit literal 'k' and the result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: `ADDLW 15h`

Before Instruction
W = 10h
After Instruction
W = 25h

ADDWF **ADD W to f**

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

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $(W) + (f) \rightarrow dest$

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

Encoding:

0010	01da	ffff	ffff

Description: Add W to register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored 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 $f \le 95$ (5Fh). See **Section 35.2.3 "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: `ADDWF REG, 0, 0`

Before Instruction
W = 17h
REG = 0C2h
After Instruction
W = 0D9h
REG = 0C2h

Note: All PIC18 instructions may take an optional label argument preceding the instruction mnemonic for use in symbolic addressing. If a label is used, the instruction format then becomes: {label} instruction argument(s).

CLRF **Clear f**

Syntax: CLRF f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: 000h → f
1 → Z

Status Affected: Z

Encoding:

0110	101a	ffff	ffff

Description: Clears the contents of the specified register.
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 $f \le 95$ (5Fh). See **Section 35.2.3 "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 register 'f'

Example: CLRF FLAG_REG, 1

Before Instruction
FLAG_REG = 5Ah
After Instruction
FLAG_REG = 00h

CLRWDT **Clear Watchdog Timer**

Syntax: CLRWDT

Operands: None

Operation: 000h → WDT,
000h → WDT postscaler,
1 → $\overline{TO}$,
1 → $\overline{PD}$

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

Encoding:

0000	0000	0000	0100

Description: CLRWDT instruction resets the Watchdog Timer. It also resets the postscaler of the WDT. Status bits, $\overline{TO}$ and $\overline{PD}$, are set.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	No operation

Example: CLRWDT

Before Instruction
WDT Counter = ?
After Instruction
WDT Counter = 00h
WDT Postscaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

SUBLW **Subtract W from literal**

Syntax: SUBLW k

Operands: $0 \le k \le 255$

Operation: $k - (W) \rightarrow W$

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

Encoding:

0000	1000	kkkk	kkkk

Description: W is subtracted from the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example 1:
```
SUBLW   02h
Before Instruction
    W   =   01h
    C   =   ?
After Instruction
    W   =   01h
    C   =   1    ; result is positive
    Z   =   0
    N   =   0
```

Example 2:
```
SUBLW   02h
Before Instruction
    W   =   02h
    C   =   ?
After Instruction
    W   =   00h
    C   =   1    ; result is zero
    Z   =   1
    N   =   0
```

Example 3:
```
SUBLW   02h
Before Instruction
    W   =   03h
    C   =   ?
After Instruction
    W   =   FFh  ; (2's complement)
    C   =   0    ; result is negative
    Z   =   0
    N   =   1
```

SUBWF **Subtract W from f**

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

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $(f) - (W) \rightarrow$ dest

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

Encoding:

0101	11da	ffff	ffff

Description: Subtract W from register 'f' (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored 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 $f \le 95$ (5Fh). See **Section 35.2.3 "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

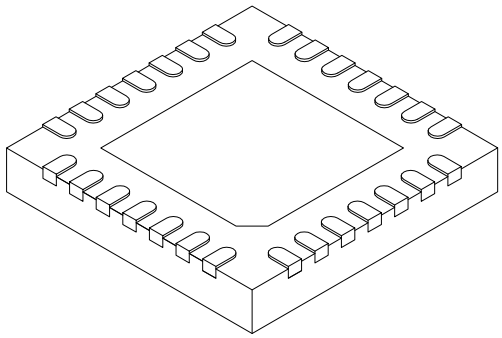
Example 1:
```
SUBWF   REG, 1, 0
Before Instruction
    REG   =   3
    W     =   2
    C     =   ?
After Instruction
    REG   =   1
    W     =   2
    C     =   1    ; result is positive
    Z     =   0
    N     =   0
```

Example 2:
```
SUBWF   REG, 0, 0
Before Instruction
    REG   =   2
    W     =   2
    C     =   ?
After Instruction
    REG   =   2
    W     =   0
    C     =   1    ; result is zero
    Z     =   1
    N     =   0
```

Example 3:
```
SUBWF   REG, 1, 0
Before Instruction
    REG   =   1
    W     =   2
    C     =   ?
After Instruction
    REG   =   FFh  ;(2's complement)
    W     =   2
    C     =   0    ; result is negative
    Z     =   0
    N     =   1
```

28-Lead Plastic Quad Flat, No Lead Package (ML) - 6x6 mm Body [QFN]
With 0.55 mm Terminal Length

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



	Units	MILLIMETERS		
Dimension	Limits	MIN	NOM	MAX
Number of Pins	N		28	
Pitch	e		0.65 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Terminal Thickness	A3		0.20 REF	
Overall Width	E		6.00 BSC	
Exposed Pad Width	E2	3.65	3.70	4.20
Overall Length	D		6.00 BSC	
Exposed Pad Length	D2	3.65	3.70	4.20
Terminal Width	b	0.23	0.30	0.35
Terminal Length	L	0.50	0.55	0.70
Terminal-to-Exposed Pad	K	0.20	-	-

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated
3. Dimensioning and tolerancing per ASME Y14.5M.
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-105C Sheet 2 of 2