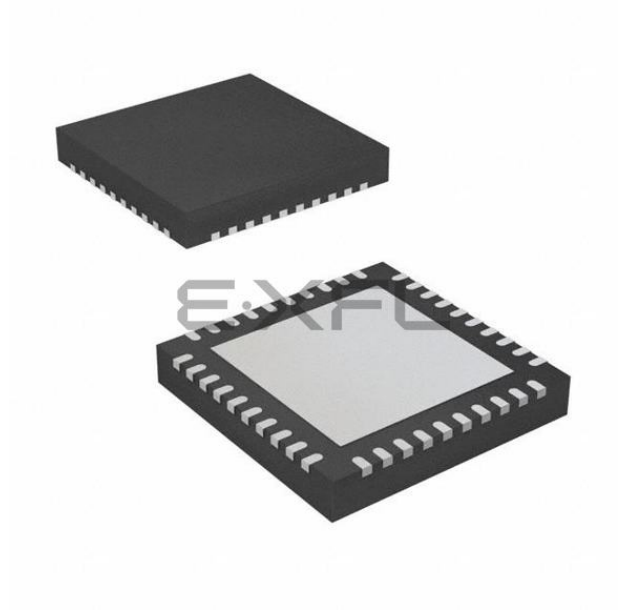


Welcome to E-XFL.COM

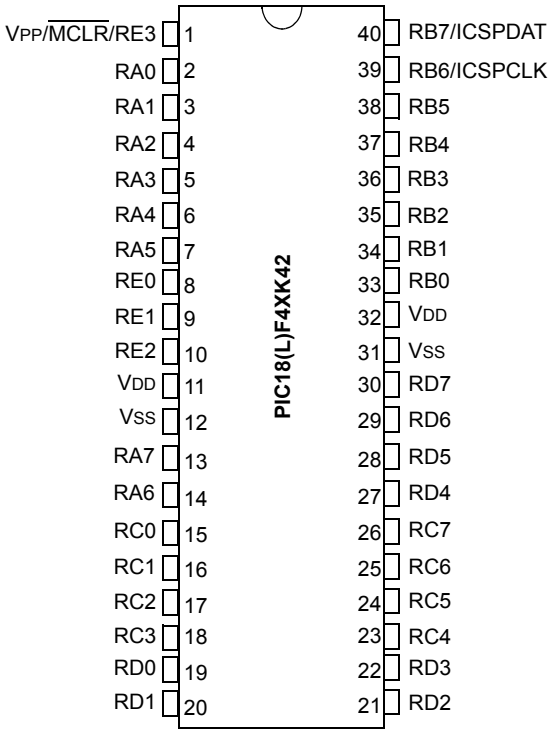
What is "Embedded - Microcontrollers"?

"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"

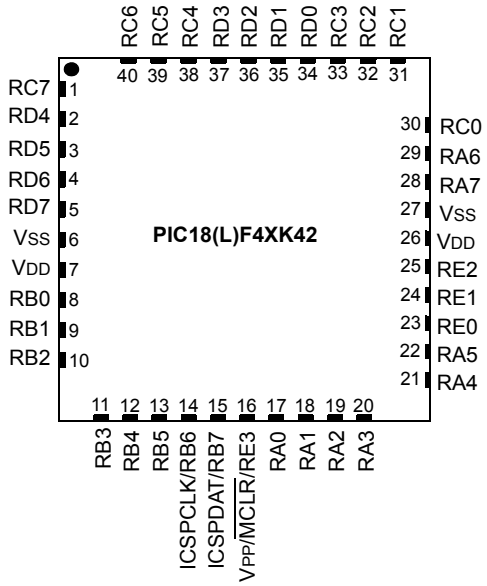
Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, HLVD, POR, PWM, WDT
Number of I/O	36
Program Memory Size	32KB (16K x 16)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 35x12b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	40-UFQFN Exposed Pad
Supplier Device Package	40-UQFN (5x5)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18lf45k42-e-mv

40-pin PDIP



Note: See Table 2 for location of all peripheral functions.

40-pin UQFN (5x5x0.5mm)



Note 1: See Table 2 for location of all peripheral functions.

2: It is recommended that the exposed bottom pad be connected to VSS, however it must not be the only VSS connection to the device.

REGISTER 3-4: DMA2PR: DMA2 PRIORITY REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-0/0	R/W-1/1	R/W-1/1
—	—	—	—	—	DMA2PR<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

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

bit 2-0 **DMA2PR<2:0>**: DMA2 Priority Selection bits

REGISTER 3-5: SCANPR: SCANNER PRIORITY REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-1/1	R/W-0/0	R/W-0/0
—	—	—	—	—	SCANPR<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

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

bit 2-0 **SCANPR<2:0>**: Scanner Priority Selection bits

REGISTER 3-6: PRLOCK: PRIORITY LOCK REGISTER

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0/0
—	—	—	—	—	—	—	PRLOCKED
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

bit 7-1 **Unimplemented:** Read as '0'

bit 0 **PRLOCKED**: PR Register Lock bit(1, 2)
0 = Priority Registers can be modified by write operations; Peripherals do not have access to the memory
1 = Priority Registers are locked and cannot be written; Peripherals have access to the memory

Note 1: The PRLOCKED bit can only be set or cleared after the unlock sequence.

2: If PR1WAY = 1, the PRLOCKED bit cannot be cleared after it has been set. A device Reset will clear the bit and allow one more set.

4.7.2 DIRECT ADDRESSING

Direct addressing specifies all or part of the source and/or destination address of the operation within the opcode itself. The options are specified by the arguments accompanying the instruction.

In the core PIC18 instruction set, bit-oriented and byte-oriented instructions use some version of direct addressing by default. All of these instructions include some 8-bit literal address as their Least Significant Byte. This address specifies either a register address in one of the banks of data RAM (**Section 4.5.2 "General Purpose Register File"**) or a location in the Access Bank (**Section 4.5.4 "Access Bank"**) as the data source for the instruction.

The Access RAM bit 'a' determines how the address is interpreted. When 'a' is '1', the contents of the BSR (**Section 4.5.1 "Bank Select Register (BSR)"**) are used with the address to determine the complete 14-bit address of the register. When 'a' is '0', the address is interpreted as being a register in the Access Bank. Addressing that uses the Access RAM is sometimes also known as Direct Forced Addressing mode.

A few instructions, such as MOVFFL, include the entire 14-bit address (either source or destination) in their opcodes. In these cases, the BSR is ignored entirely.

The destination of the operation's results is determined by the destination bit 'd'. When 'd' is '1', the results are stored back in the source register, overwriting its original contents. When 'd' is '0', the results are stored in the W register. Instructions without the 'd' argument have a destination that is implicit in the instruction; their destination is either the target register being operated on or the W register.

4.7.3 INDIRECT ADDRESSING

Indirect addressing allows the user to access a location in data memory without giving a fixed address in the instruction. This is done by using File Select Registers (FSRs) as pointers to the locations which are to be read or written. Since the FSRs are themselves located in RAM as Special File Registers, they can also be directly manipulated under program control. This makes FSRs very useful in implementing data structures, such as tables and arrays in data memory.

The registers for indirect addressing are also implemented with Indirect File Operands (INDFs) that permit automatic manipulation of the pointer value with auto-incrementing, auto-decrementing or offsetting with another value. This allows for efficient code, using loops, such as the example of clearing an entire RAM bank in Example 4-6.

EXAMPLE 4-6: HOW TO CLEAR RAM (BANK 1) USING INDIRECT ADDRESSING

```
        LFSR   FSR0, 100h ;
NEXT    CLRF   POSTINC0   ; Clear INDF
                          ; register then
                          ; inc pointer
        BTFSS  FSR0H, 1   ; All done with
                          ; Bank1?
        BRA    NEXT       ; NO, clear next
CONTINUE                  ; YES, continue
```

4.7.3.1 FSR Registers and the INDF Operand

At the core of indirect addressing are three sets of registers: FSR0, FSR1 and FSR2. Each represents a pair of 8-bit registers, FSRnH and FSRnL. Each FSR pair holds a 14-bit value, therefore, the two upper bits of the FSRnH register are not used. The 14-bit FSR value can address the entire range of the data memory in a linear fashion. The FSR register pairs, then, serve as pointers to data memory locations.

Indirect addressing is accomplished with a set of Indirect File Operands, INDF0 through INDF2. These can be thought of as "virtual" registers; they are mapped in the SFR space but are not physically implemented. Reading or writing to a particular INDF register actually accesses the data addressed by its corresponding FSR register pair. A read from INDF1, for example, reads the data at the address indicated by FSR1H:FSR1L. Instructions that use the INDF registers as operands actually use the contents of their corresponding FSR as a pointer to the instruction's target. The INDF operand is just a convenient way of using the pointer.

Because indirect addressing uses a full 14-bit address, data RAM banking is not necessary. Thus, the current contents of the BSR and the Access RAM bit have no effect on determining the target address.

5.7.4 FIXED VOLTAGE REFERENCE DATA

The DIA stores measured FVR voltages for this device in mV for the different buffer settings of 1x, 2x or 4x at Program Memory locations 3F0030h to 3F003Bh. For more information on the FVR, refer to **Section 34.0 "Fixed Voltage Reference (FVR)"**.

- FVRA1X stores the value of ADC FVR1 Output voltage for 1x setting (in mV)
- FVRA2X stores the value of ADC FVR1 Output Voltage for 2x setting (in mV)
- FVRA4X stores the value of ADC FVR1 Output Voltage for 4x setting (in mV)
- FVRC1X stores the value of Comparator FVR2 output voltage for 2x setting (in mV)
- FVRC2X stores the value of Comparator FVR2 output voltage for 2x setting (in mV)
- FVRC4X stores the value of Comparator FVR2 output voltage for 4x setting (in mV)

5.8 Device Configuration Information

The Device Configuration Information (DCI) is a dedicated region in the Program memory space mapped from 3FFF00h to 3FFF09h. The data stored in these locations is read-only and cannot be erased.

Refer to Table 5-4: Device Configuration Information for PIC18(L)F26/27/45/55/46/47/56/57K42 for the complete DCI table address and description. The DCI holds information about the device which is useful for programming and Bootloader applications.

The erase size is the minimum erasable unit in the PFM, expressed as rows. The total device Flash memory capacity is (Row Size * Number of rows)

TABLE 5-4: DEVICE CONFIGURATION INFORMATION FOR PIC18(L)F26/27/45/55/46/47/56/57K42

ADDRESS	Name	DESCRIPTION	VALUE			UNITS
			PIC18(L)F45/55K42	PIC18(L)F26/46/56K42	PIC18(L)F27/47/57K42	
3F FF00h-3F FF01h	ERSIZ	Erase Row Size	64	64	64	Words
3F FF02h-3F FF03h	WLSIZ	Number of write latches per row	128	128	128	Bytes
3F FF04h-3F FF05h	URSIZ	Number of User Rows	256	512	1024	Rows
3F FF06h-3F FF07h	EESIZ	Data EEPROM memory size	256	1024	1024	Bytes
3F FF08h-3F FF09h	PCNT	Pin Count	40(1)/48	28/40(1)/48	28/40(1)/48	Pins

Note 1: Pin count of 40 is also used for 44-pin part.

9.5 Context Saving

The Interrupt controller supports a two-level deep context saving (Main routine context and Low ISR context). Refer to state machine shown in Figure 9-6 for details.

The Program Counter (PC) is saved on the dedicated device PC stack. CPU registers saved include STATUS, WREG, BSR, FSR0/1/2, PRODL/H and PCLATH/U.

After WREG has been saved to the context registers, the resolved vector number of the interrupt source to be serviced is copied into WREG. Context save and restore operation is completed by the interrupt controller based on current state of the interrupts and the order in which they were sent to the CPU.

Context save/restore works the same way in both states of MVECEN. When IPEN = 0, there is only one level interrupt active. Hence, only the main context is saved when an interrupt is received.

9.5.1 ACCESSING SHADOW REGISTERS

The Interrupt controller automatically saves the context information in the shadow registers available in Bank 56. Both the saved context values (i.e., main routine and low ISR) can be accessed using the same set of shadow registers. By clearing the SHADLO bit in the SHADCON register (Register 9-43), the CPU register values saved for main routine context can accessed, and by setting the SHADLO bit of the CPU register, values saved for low ISR context can accessed. Low ISR context is automatically restored to the CPU registers upon exiting the high ISR. Similarly, the main context is automatically restored to the CPU registers upon exiting the low ISR.

The Shadow registers in Bank 56 are readable and writable, so if the user desires to modify the context, then the corresponding shadow register should be modified and the value will be restored when exiting the ISR. Depending on the user's application, other registers may also need to be saved.

REGISTER 9-31: IPR6: PERIPHERAL INTERRUPT PRIORITY REGISTER 6

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
TMR3GIP	TMR3IP	U2IP	U2EIP	U2TXIP	U2RXIP	I2C2EIP	I2C2IP
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 **TMR3GIP:** TMR3 Gate Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **TMR3IP:** TMR3 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **U2IP:** UART2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **U2EIP:** UART2 Framing Error Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **U2TXIP:** UART2 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **U2RXIP:** UART2 Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **I2C2EIP:** I^2C2 Error Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **I2C2IP:** I^2C2 Interrupt Priority bit
1 = High priority
0 = Low priority

12.0 8x8 HARDWARE MULTIPLIER

12.1 Introduction

All PIC18 devices include an 8x8 hardware multiplier as part of the ALU. The multiplier performs an unsigned operation and yields a 16-bit result that is stored in the product register pair, PRODH:PRODL. The multiplier's operation does not affect any flags in the STATUS register.

Making multiplication a hardware operation allows it to be completed in a single instruction cycle. This has the advantages of higher computational throughput and reduced code size for multiplication algorithms and allows the PIC18 devices to be used in many applications previously reserved for digital signal processors. A comparison of various hardware and software multiply operations, along with the savings in memory and execution time, is shown in Table 12-1.

12.2 Operation

Example 12-1 shows the instruction sequence for an 8x8 unsigned multiplication. Only one instruction is required when one of the arguments is already loaded in the WREG register.

Example 12-2 shows the sequence to do an 8x8 signed multiplication. To account for the sign bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EXAMPLE 12-1: 8x8 UNSIGNED MULTIPLY ROUTINE

```
MOVF   ARG1, W   ;
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
```

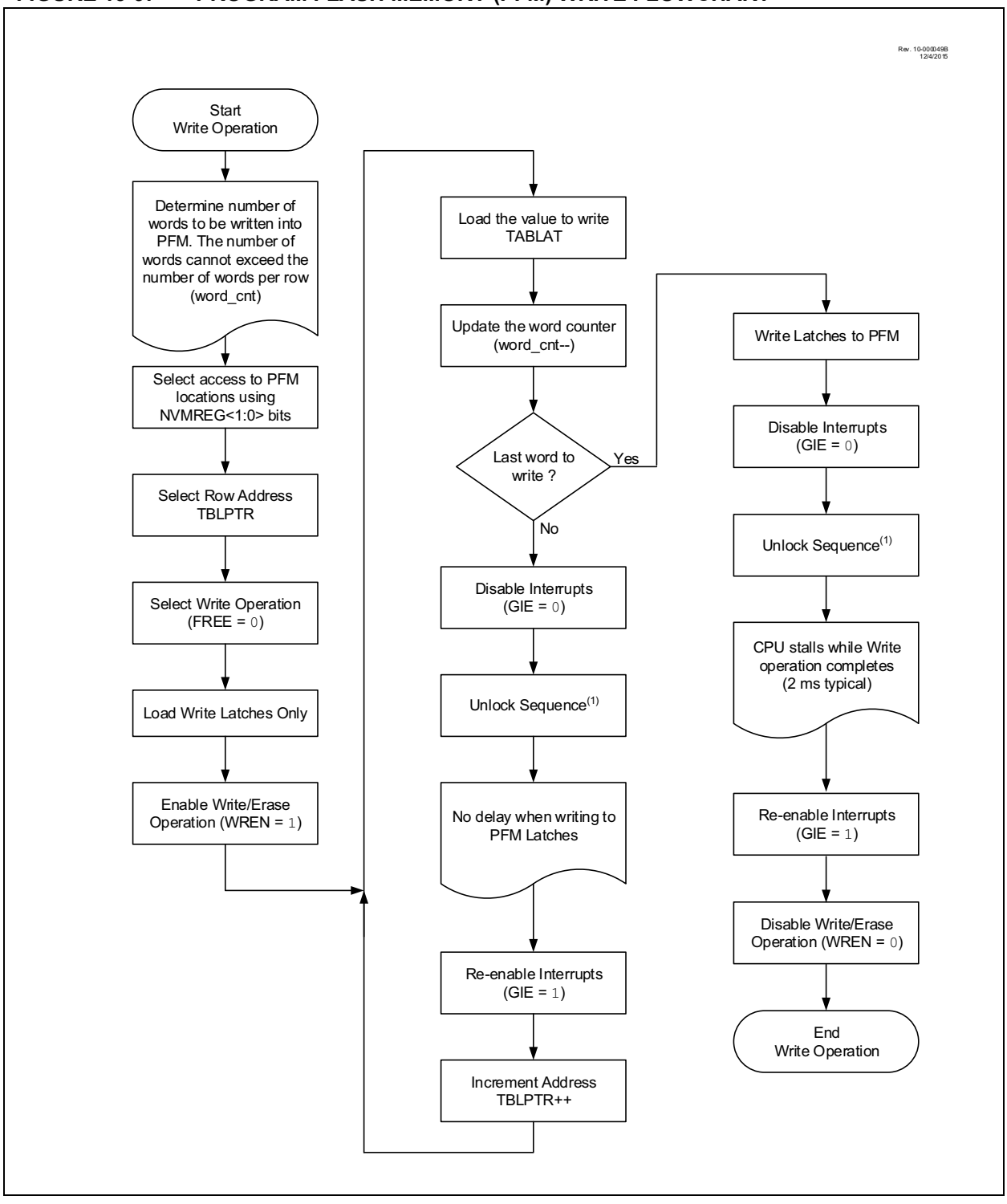
EXAMPLE 12-2: 8x8 SIGNED MULTIPLY ROUTINE

```
MOVF   ARG1, W
MULWF  ARG2      ; ARG1 * ARG2 ->
                 ; PRODH:PRODL
BTFSC  ARG2, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;          - ARG1
MOVF   ARG2, W
BTFSC  ARG1, SB  ; Test Sign Bit
SUBWF  PRODH, F  ; PRODH = PRODH
                 ;          - ARG2
```

TABLE 12-1: PERFORMANCE COMPARISON FOR VARIOUS MULTIPLY OPERATIONS

Routine	Multiply Method	Program Memory (Words)	Cycles (Max)	Time			
				@ 64 MHz	@ 40 MHz	@ 10 MHz	@ 4 MHz
8x8 unsigned	Without hardware multiply	13	69	4.3 µs	6.9 µs	27.6 µs	69 µs
	Hardware multiply	1	1	62.5 ns	100 ns	400 ns	1 µs
8x8 signed	Without hardware multiply	33	91	5.7 µs	9.1 µs	36.4 µs	91 µs
	Hardware multiply	6	6	375 ns	600 ns	2.4 µs	6 µs
16x16 unsigned	Without hardware multiply	21	242	15.1 µs	24.2 µs	96.8 µs	242 µs
	Hardware multiply	28	28	1.8 µs	2.8 µs	11.2 µs	28 µs
16x16 signed	Without hardware multiply	52	254	15.9 µs	25.4 µs	102.6 µs	254 µs
	Hardware multiply	35	40	2.5 µs	4.0 µs	16.0 µs	40 µs

FIGURE 13-9: PROGRAM FLASH MEMORY (PFM) WRITE FLOWCHART



15.5 DMA Message Transfers

Once the Enable bit is set to start DMA message transfers, the Source/Destination pointer and counter registers are initialized to the conditions shown in Table 15-3.

TABLE 15-3: DMA INITIAL CONDITIONS

Register	Value loaded
DMAxSPTR<21:0>	DMAxSSA<21:0>
DMAxSCNT<11:0>	DMAxSSZ<11:0>
DMAxDPTR<15:0>	DMAxDSA<15:0>
DMAxDCNT<11:0>	DMAxDSZ<11:0>

During the DMA Operation after each transaction, Table 15-4 and Table 15-5 indicate how the Source/Destination pointer and counter registers are modified.

TABLE 15-4: DMA SOURCE POINTER/COUNTER DURING OPERATION

Register	Modified Source Counter/Pointer Value
DMAxSCNT<11:0> != 1	DMAxSCNT = DMAxSCNT -1
	SMODE = 00: DMAxSPTR = DMAxSPTR
	SMODE = 01: DMAxSPTR = DMAxSPTR + 1
	SMODE = 10: DMAxSPTR = DMAxSPTR - 1
DMAxSCNT<11:0> == 1	DMAxSCNT = DMAxSSZ
	DMAxSPTR = DMAxSSA

TABLE 15-5: DMA DESTINATION POINTER/COUNTER DURING OPERATION

Register	Modified Destination Counter/Pointer Value
DMAxDCNT<11:0>!= 1	DMAxDCNT = DMAxDCNT -1
	DMODE = 00: DMAxDPTR = DMAxDPTR
	DMODE = 01: DMAxDPTR = DMAxDPTR + 1
	DMODE = 10: DMAxDPTR = DMAxDPTR - 1
DMAxDCNT<11:0> == 1	DMAxDCNT = DMAxDSZ
	DMAxDPTR = DMAxDSA

The following sections discuss how to initiate and terminate DMA transfers.

15.5.1 STARTING DMA MESSAGE TRANSFERS

The DMA can initiate data transactions by either of the following two conditions:

1. User software control
2. Hardware trigger, SIRQ

15.5.1.1 User Software Control

Software starts or stops DMA transaction by setting/clearing the DGO bit. The DGO bit is also used to indicate whether a DMA hardware trigger has been received and a message is in progress.

Note 1: Software start can only occur if the EN bit (DMAxCON1) is set.

2: If the CPU writes to the DGO bit while it is already set, there is no effect on the system, the DMA will continue to operate normally.

15.10 Reset

The DMA registers are set to the default state on any Reset. The registers are also reset to the default state when the enable bit is cleared (DMA1CON1bits.EN=0).

15.11 Power Saving Mode Operation

The DMA utilizes system clocks and it is treated as a peripheral when it comes to power saving operations. Like other peripherals, the DMA also uses Peripheral Module Disable bits to further tailor its operation in low-power states.

15.11.1 SLEEP MODE

When the device enters Sleep mode, the system clock to the module is shut down, therefore no DMA operation is supported in Sleep. Once the system clock is disabled, the requisite read and write clocks are also disabled, without which the DMA cannot perform any of its tasks.

Any transfers that may be in progress are resumed on exiting from Sleep mode. Register contents are not affected by the device entering or leaving Sleep mode. It is recommended that DMA transactions be allowed to finish before entering Sleep mode.

15.11.2 IDLE MODE

In IDLE mode, all of the system clocks (including the read and write clocks) are still operating but the CPU is not using them to save power.

Therefore, every instruction cycle is available to the system arbiter and if the bubble is granted to the DMA, it may be utilized to move data.

15.11.3 DOZE MODE

Similar to the Idle mode, the CPU does not utilize all of the available instruction cycles slots that are available to it in order to save power. It only executes instructions based on its settings from the Doze settings.

Therefore, every instruction not used by the CPU is available for system arbitration and may be utilized by the DMA if granted by the arbiter.

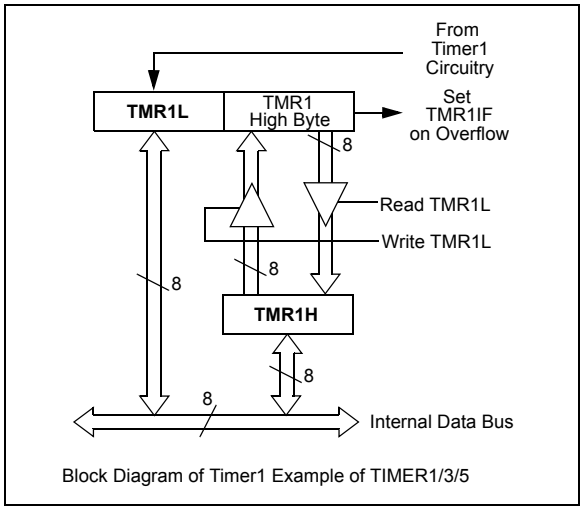
15.11.4 PERIPHERAL MODULE DISABLE

The Peripheral Module Disable (PMD) registers provide a method to disable DMA by gating all clock sources supplied to it. The respective DMAxMD bit needs to be set in order to disable the DMA.

15.12 DMA Register Interfaces

The DMA can transfer data to any GPR or SFR location. For better user accessibility, some of the more commonly used SFR spaces have their Mirror registers placed in Bank 64 (0x4000-0x40FF). These Mirror registers can be only accessed through the DMA Source and Destination Address registers. Refer to Table 4-3 for details about Bank 64 Registers.

FIGURE 21-2: TIMER1/3/5 16-BIT READ/ WRITE MODE BLOCK DIAGRAM



Block Diagram of Timer1 Example of TIMER1/3/5

21.6 Timer1/3/5 Gate

Timer1/3/5 can be configured to count freely or the count can be enabled and disabled using Timer1/3/5 gate circuitry. This is also referred to as Timer1/3/5 gate enable.

Timer1/3/5 gate can also be driven by multiple selectable sources.

21.6.1 TIMER1/3/5 GATE ENABLE

The Timer1/3/5 Gate Enable mode is enabled by setting the TMRxGE bit of the TxGCON register. The polarity of the Timer1/3/5 Gate Enable mode is configured using the TxGPOL bit of the TxGCON register.

When Timer1/3/5 Gate Enable mode is enabled, Timer1/3/5 will increment on the rising edge of the Timer1/3/5 clock source. When Timer1/3/5 Gate signal is inactive, the timer will not increment and hold the current count. See Figure 21-4 for timing details.

TABLE 21-2: TIMER1/3/5 GATE ENABLE SELECTIONS

TMRxCLK	TxGPOL	TxG	Timer1/3/5 Operation
↑	1	1	Counts
↑	1	0	Holds Count
↑	0	1	Holds Count
↑	0	0	Counts

FIGURE 21-5: TIMER1/3/5 GATE TOGGLE MODE

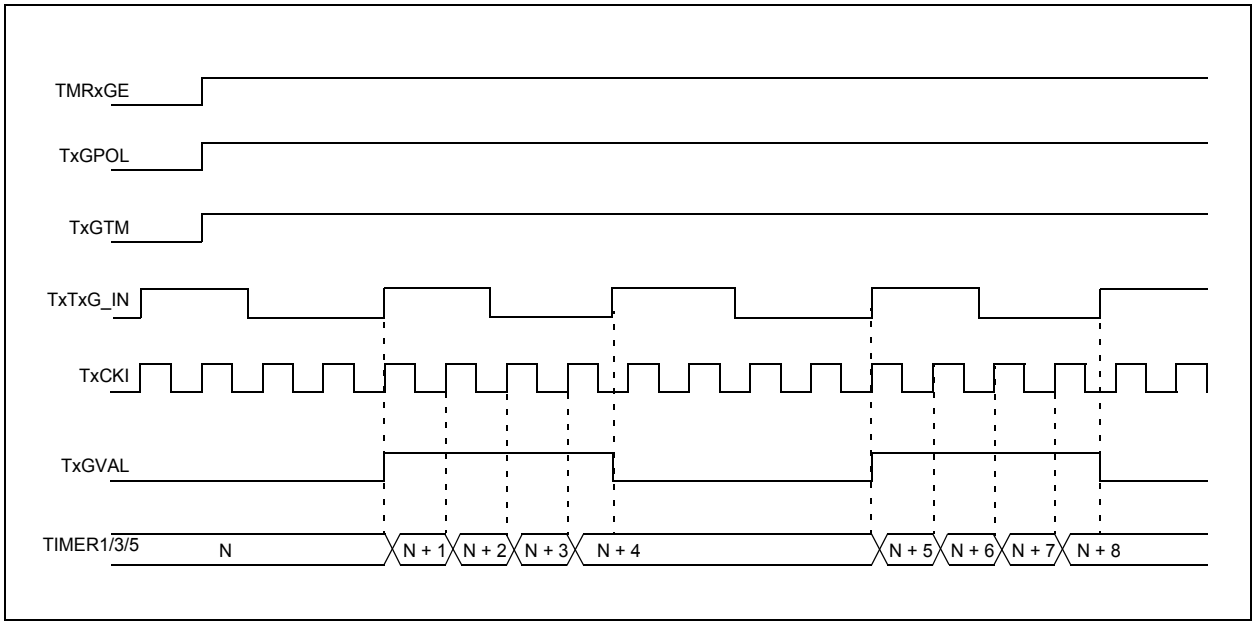
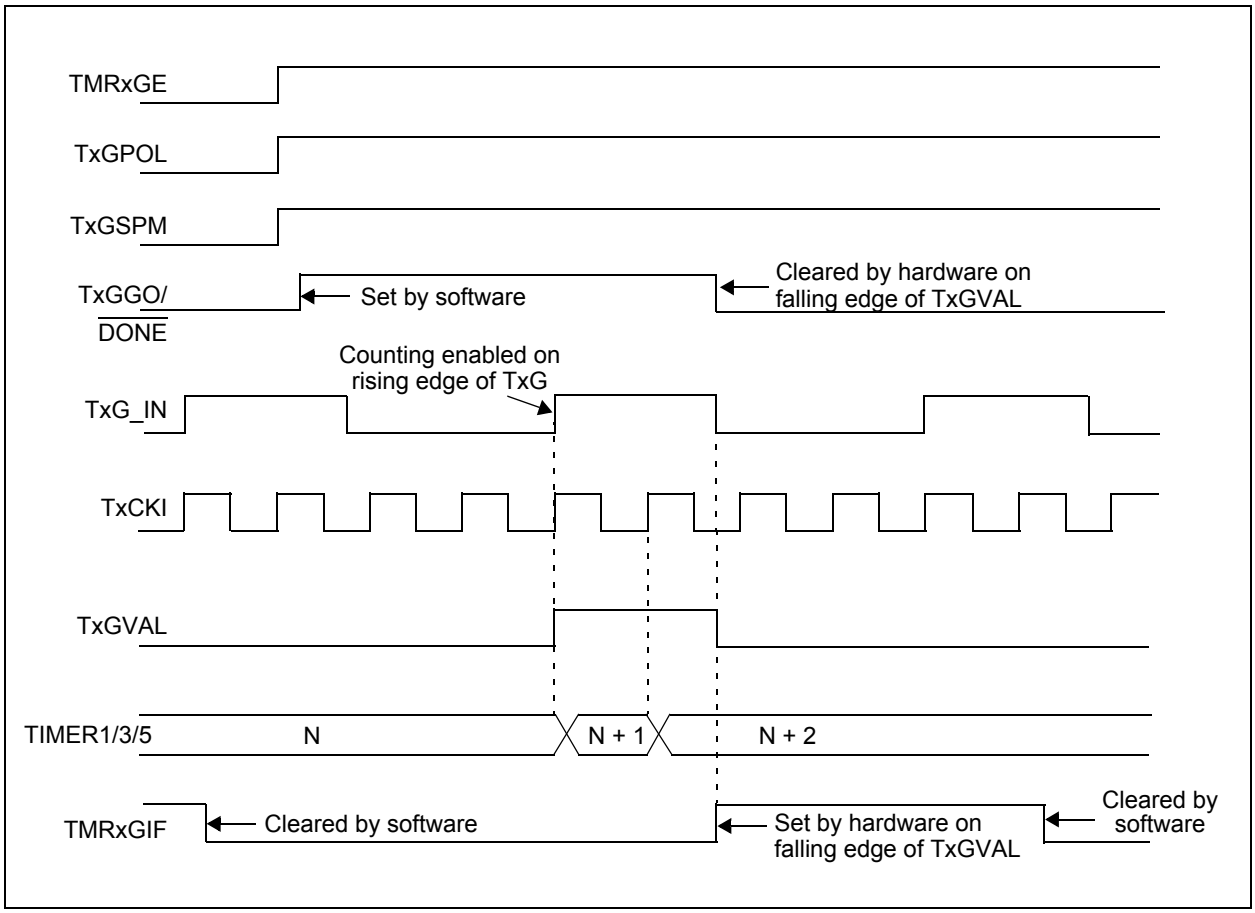


FIGURE 21-6: TIMER1/3/5 GATE SINGLE-PULSE MODE



21.12 Register Definitions: Timer1/3/5

Long bit name prefixes for the Timer1/3/5 are shown below. Refer to **Section 1.3.2.2 "Long Bit Names"** for more information.

Peripheral	Bit Name Prefix
Timer1	T1
Timer3	T3
Timer5	T5

REGISTER 21-1: 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

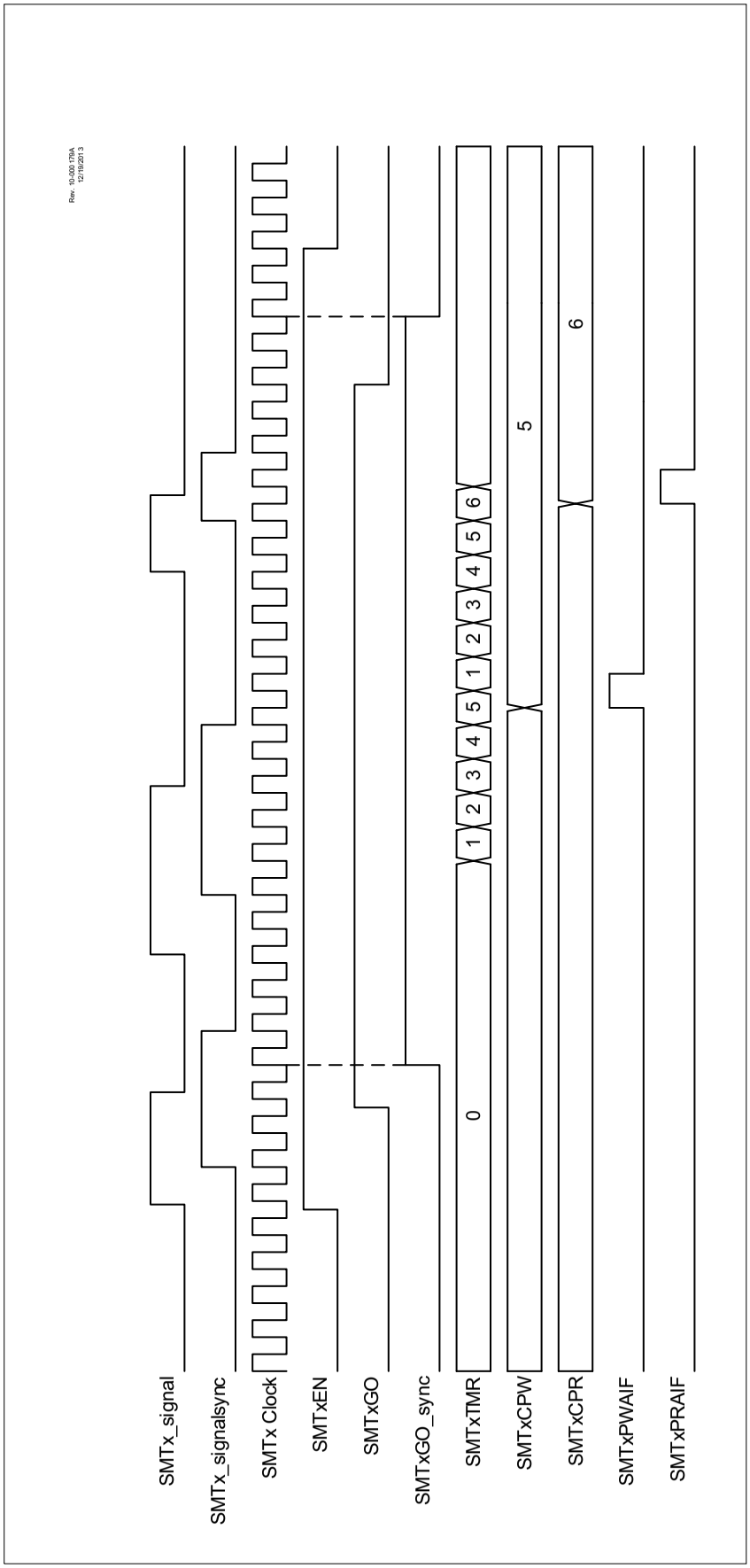
bit 3 **Unimplemented:** Read as '0'

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

FIGURE 25-9: HIGH AND LOW MEASURE MODE SINGLE ACQUISITION TIMING DIAGRAM



REGISTER 28-2: NCO1CLK: NCO1 INPUT CLOCK CONTROL REGISTER

R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0
PWS<2:0>(1,2)			—	CKS<3:0>			
bit 7							bit 0

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

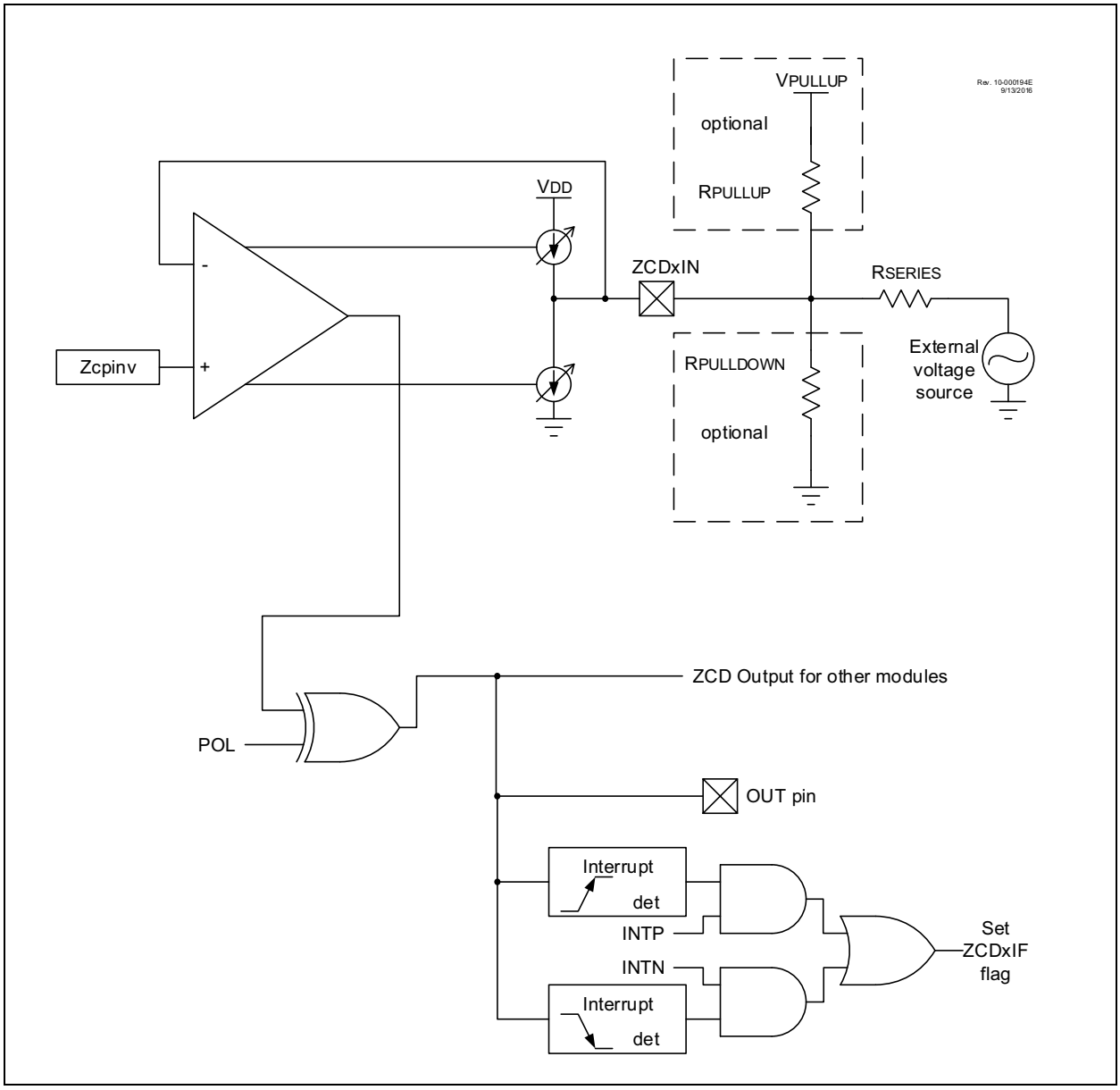
bit 7-5 **PWS<2:0>:** NCO1 Output Pulse Width Select bits(1,2)
- 111 = NCO1 output is active for 128 input clock periods
- 110 = NCO1 output is active for 64 input clock periods
- 101 = NCO1 output is active for 32 input clock periods
- 100 = NCO1 output is active for 16 input clock periods
- 011 = NCO1 output is active for 8 input clock periods
- 010 = NCO1 output is active for 4 input clock periods
- 001 = NCO1 output is active for 2 input clock periods
- 000 = NCO1 output is active for 1 input clock period

bit 4 **Unimplemented:** Read as '0'

bit 3-0 **CKS<3:0>:** NCO1 Clock Source Select bits
- 1111 = Reserved
- •
- •
- •
- 1011 = Reserved
- 1010 = CLC4_out
- 1001 = CLC3_out
- 1000 = CLC2_out
- 0111 = CLC1_out
- 0110 = CLKREF_out
- 0101 = SOSC
- 0100 = MFINTOSC/4 (32 kHz)
- 0011 = MFINTOSC (500 kHz)
- 0010 = LFINTOSC
- 0001 = HFINTOSC
- 0000 = FOSC

Note 1: N1PWS applies only when operating in Pulse Frequency mode.
2: If NCO1 pulse width is greater than NCO1 overflow period, operation is undefined.

FIGURE 29-2: SIMPLIFIED ZCD BLOCK DIAGRAM



29.2 ZCD Logic Output

The ZCD module includes a Status bit, which can be read to determine whether the current source or sink is active. The OUT bit of the ZCDCON register is set when the current sink is active, and cleared when the current source is active. The OUT bit is affected by the polarity bit, even if the module is disabled.

The OUT signal can also be used as input to other modules. This is controlled by the registers of the corresponding module. OUT can be used as follows:

- Gate source for TMR1/3/5
- Clock source for TMR2/4/6
- Reset source for TMR2/4/6

29.3 ZCD Logic Polarity

The POL bit of the ZCDCON register inverts the OUT bit relative to the current source and sink output. When the POL bit is set, a OUT high indicates that the current source is active, and a low output indicates that the current sink is active.

The POL bit affects the ZCD interrupts.

30.1 DSM Operation

The DSM module can be enabled by setting the EN bit in the MD1CON0 register. Clearing the EN bit in the MD1CON0 register, disables the DSM module output and switches the carrier high and carrier low signals to the default option of MD1CARHPPS and MD1CARLPPS, respectively. The modulator signal source is also switched to the BIT in the MD1CON0 register.

The values used to select the carrier high, carrier low, and modulator sources held by the Modulation Source, Modulation High Carrier, and Modulation Low Carrier control registers are not affected when the EN bit is cleared and the DSM module is disabled. The values inside these registers remain unchanged while the DSM is inactive. The sources for the carrier high, carrier low and modulator signals will once again be selected when the EN bit is set and the DSM module is again enabled and active.

30.2 Modulator Signal Sources

The modulator signal can be supplied from the sources specified in Table 30-3.

The modulator signal is selected by configuring the MS<4:0> bits in the MD1SRC register.

30.3 Carrier Signal Sources

The carrier high signal and carrier low signal can be supplied from the sources specified in Table 30-1.

The carrier high signal is selected by configuring the CH<4:0> bits in the MD1CARH register. The carrier low signal is selected by configuring the CL<4:0> bits in the MD1CARL register.

30.4 Carrier Synchronization

During the time when the DSM switches between carrier high and carrier low signal sources, the carrier data in the modulated output signal can become truncated. To prevent this, the carrier signal can be synchronized to the modulator signal. When synchronization is enabled, the carrier pulse that is being mixed at the time of the transition is allowed to transition low before the DSM switches over to the next carrier source.

Synchronization is enabled separately for the carrier high and carrier low signal sources. Synchronization for the carrier high signal is enabled by setting the CHSYNC bit in the MD1CON1 register. Synchronization for the carrier low signal is enabled by setting the CLSYNC bit in the MD1CON1 register.

Figure 30-2 through Figure 30-6 show timing diagrams of using various synchronization methods.

FIGURE 30-5: Carrier Low Synchronization (CHSYNC = 0, CLSYNC = 1)

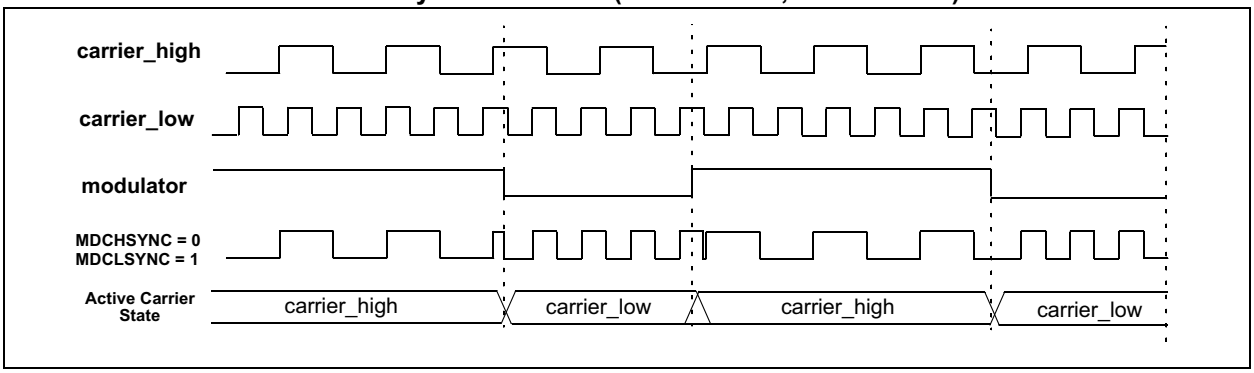
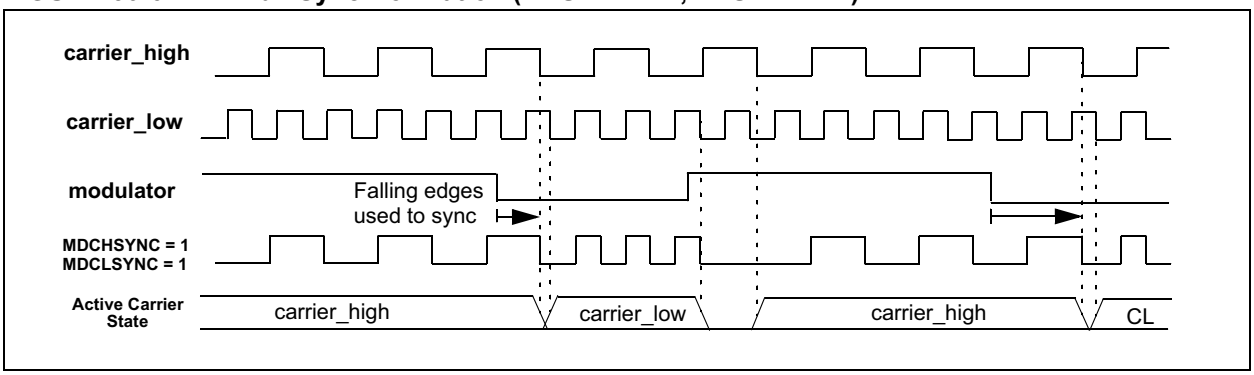


FIGURE 30-6: Full Synchronization (CHSYNC = 1, CLSYNC = 1)



33.5.11 MASTER TRANSMISSION IN 10-BIT ADDRESSING MODE

This section describes the sequence of events for the I^2C module configured as an I^2C master in 10-bit Addressing mode and is transmitting data. Figure 33-21 is used as a visual reference for this description

1. If ABD = 0; i.e., Address buffers are enabled

Master software loads number of bytes to be transmitted in one sequence in I2CxCNT, high address byte of slave address in I2CxADB1 with R/W = 0, low address byte in I2CxADB0 and the first byte of data in I2CxTXB. Master software has to set the Start (S) bit to initiate communication.

If ABD = 1; i.e., Address buffers are disabled

Master software loads the number of bytes to be transmitted in one sequence in I2CxCNT and the high address byte of the slave address with R/W = 0 into the I2CxTXB register. Writing to the I2CxTXB will assert the start condition on the bus and sets the S bit. Software writes to the S bit are ignored in this case.

2. Master hardware waits for BFRE bit to be set; then shifts out the start and high address and waits for acknowledge.
3. If NACK, master hardware sends Stop.
4. If ABD = 0; i.e., Address buffer are enabled

If ACK, master hardware sends the low address byte from I2CxADB0.

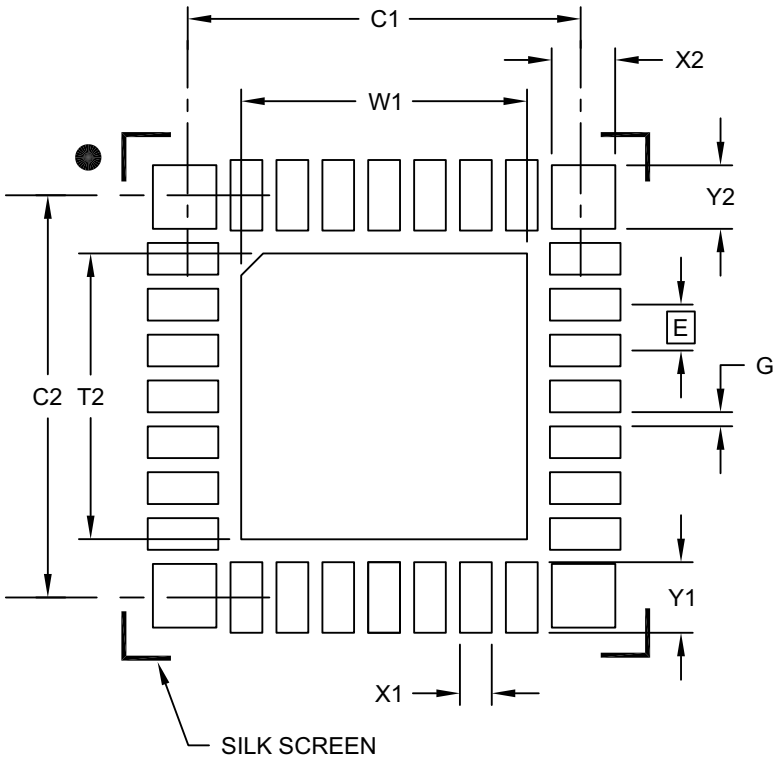
If ABD = 1; i.e., Address buffer are disabled

If ACK, master hardware sets TXIF and MDR bits and the software has to write the low address byte into I2CxTXB. Writing to I2CxTXB sends the low address on the bus.

5. If TXBE = 1 and I2CxCNT! = 0, I2CxTXIF and MDR bits are set. Clock is stretched on 8th falling SCL edge until master software writes next data byte to I2CxTXB.
6. Master hardware sends ninth SCL pulse for ACK from slave and loads the shift register from I2CxTXB. I2CxCNT is decremented.
7. If slave sends a NACK, master hardware sends Stop and ends transmission.
8. If slave sends an ACK, master hardware outputs data in the shift register on SDA. I2CxCNT value is checked on the 8th falling SCL edge. If I2CxCNT = 0; master hardware sends 9th SCL pulse for ACK and CNTIF is set.
9. If I2CxCNT! = 0; go to step 5.

28-Lead Plastic Quad Flat, No Lead Package (MX) - 6x6 mm Body [UQFN]
With 0.60mm Contact Length And Corner Anchors

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



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.65 BSC	
Optional Center Pad Width	W1			4.05
Optional Center Pad Length	T2			4.05
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.45
Contact Pad Length (X28)	Y1			1.00
Corner Pad Width (X4)	X2			0.90
Corner Pad Length (X4)	Y2			0.90
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2209B