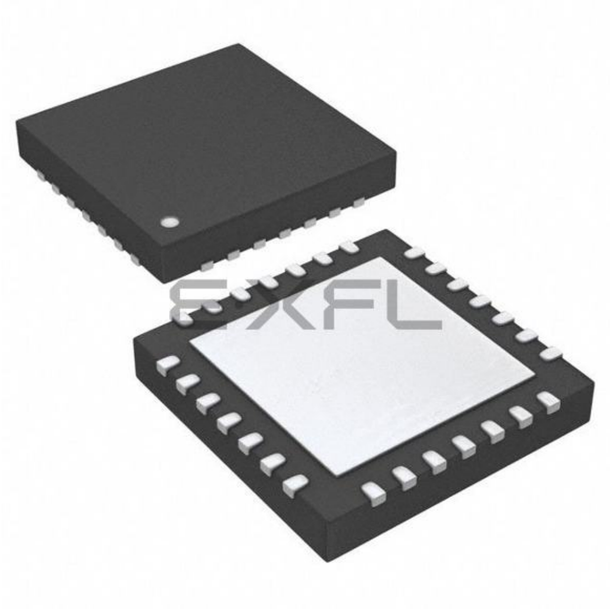
Microchip Technology - PIC18F25K42T-I/MV Datasheet



Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	64MHz
Connectivity	I²C, LINbus, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, HLVD, POR, PWM, WDT
Number of I/O	25
Program Memory Size	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/pic18f25k42t-i-mv

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com**. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Website at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000000A is version A of document DS30000000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Website; **http://www.microchip.com**
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our website at **www.microchip.com** to receive the most current information on all of our products.

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
—	—	—	—	—	DMA2PR2	DMA2PR1	DMA2PR0
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
—	—	—	—	—	SCANPR2	SCANPR1	SCANPR0
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>**: DMA2 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)]

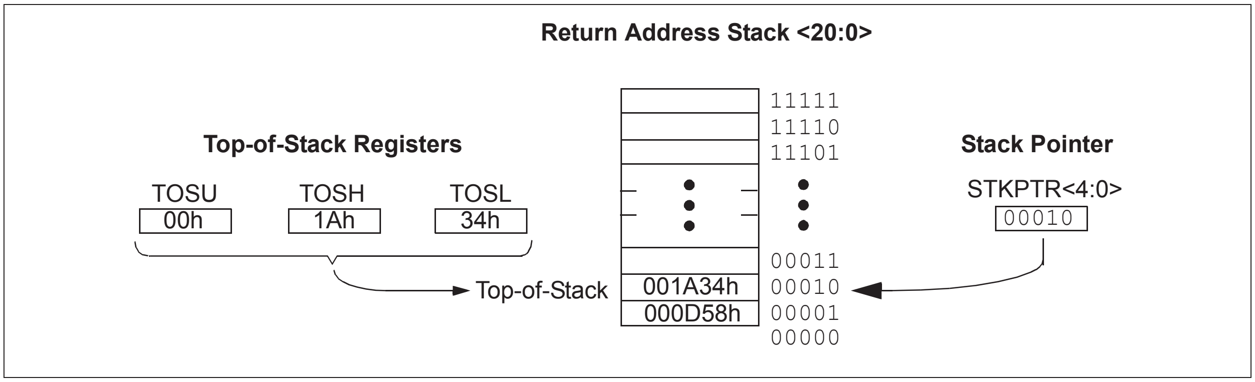
0 = Priority Registers can be modified by write operations; Peripherals do not have access to the memory

1 = Priority Registers are locked and cannot be written; Peripherals do not 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 system Reset will clear the bit and allow one more set.

FIGURE 4-1: RETURN ADDRESS STACK AND ASSOCIATED REGISTERS



4.2.5.2 Return Stack Pointer (STKPTR)

The STKPTR register (Register 4-1) contains the Stack Pointer value. The STKOVF (Stack Overflow) Status bit and the STKUNF (Stack Underflow) Status bit can be accessed using the PCON0 register. The value of the Stack Pointer can be 0 through 31. On Reset, the Stack Pointer value will be zero. The user may read and write the Stack Pointer value. This feature can be used by a Real-Time Operating System (RTOS) for stack maintenance. After the PC is pushed onto the stack 32 times (without popping any values off the stack), the STKOVF bit is set. The STKOVF bit is cleared by software or by a POR. The action that takes place when the stack becomes full depends on the state of the STVREN (Stack Overflow Reset Enable) Configuration bit. (Refer to **Section 5.1 "Configuration Words"** for a description of the device Configuration bits.)

If STVREN is set (default), a Reset will be generated and a Stack Overflow will be indicated by the STKOVF bit when the 32nd push is initiated. This includes CALL and CALLW instructions, as well as stacking the return address during an interrupt response. The STKOVF bit will remain set and the Stack Pointer will be set to zero.

If STVREN is cleared, the STKOVF bit will be set on the 32nd push and the Stack Pointer will remain at 31 but no Reset will occur. Any additional pushes will overwrite the 31st push but the STKPTR will remain at 31.

Setting STKOVF = 1 in software will change the bit, but will not generate a Reset.

The STKUNF bit is set when a stack pop returns a value of zero. The STKUNF bit is cleared by software or by POR. The action that takes place when the stack becomes full depends on the state of the STVREN (Stack Overflow Reset Enable) Configuration bit. (Refer to **Section 5.1 "Configuration Words"** for a description of the device Configuration bits).

If STVREN is set (default) and the stack has been popped enough times to unload the stack, the next pop will return a value of zero to the PC, it will set the STKUNF bit and a Reset will be generated. This condition can be generated by the RETURN, RETLW and RETFIE instructions.

When STVREN = 0, STKUNF will be set but no Reset will occur.

Note: Returning a value of zero to the PC on an underflow has the effect of vectoring the program to the Reset vector, where the stack conditions can be verified and appropriate actions can be taken. This is not the same as a Reset, as the contents of the SFRs are not affected.

4.2.5.3 PUSH and POP Instructions

Since the Top-of-Stack is readable and writable, the ability to push values onto the stack and pull values off the stack without disturbing normal program execution is a desirable feature. The PIC18 instruction set includes two instructions, PUSH and POP, that permit the TOS to be manipulated under software control. TOSU, TOSH and TOSL can be modified to place data or a return address on the stack.

The PUSH instruction places the current PC value onto the stack. This increments the Stack Pointer and loads the current PC value onto the stack.

The POP instruction discards the current TOS by decrementing the Stack Pointer. The previous value pushed onto the stack then becomes the TOS value.

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

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR
3FFFh	TOSU	—	—	—	Top of Stack Upper byte					---xxxxx
3FFEh	TOSH	Top of Stack High byte								xxxxxxxx
3FFDh	TOSL	Top of Stack Low byte								xxxxxxxx
3FFCh	STKPTR	—	—	—	Stack Pointer					---00000
3FFBh	PCLATU	—	—	—	Holding Register for PC Upper byte					---00000
3FFAh	PCLATH	Holding Register for PC High byte								00000000
3FF9h	PCL	PC Low byte								00000000
3FF8h	TBLPTRU	—	—	Program Memory Table Pointer Upper byte						--000000
3FF7h	TBLPTRH	Program Memory Table Pointer High byte								00000000
3FF6h	TBLPTRL	Program Memory Table Pointer Low byte								00000000
3FF5h	TABLAT	Table Latch								00000000
3FF4h	PRODH	Product Register High byte								xxxxxxxx
3FF3h	PRODL	Product Register Low byte								xxxxxxxx
3FF2h	—	Unimplemented								—
3FF1h	PCON1	—	—	—	—	—	—	$\overline{\text{MEMV}}$	—	------1-
3FF0h	PCON0	STKOVF	STKUNF	$\overline{\text{WDTWV}}$	$\overline{\text{RWDT}}$	$\overline{\text{RMCLR}}$	$\overline{\text{RI}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$	0011110q
3FEFh	INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed								xxxxxxxx
3FEEh	POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented								--------
3FEDh	POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented								--------
3FECh	PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented								--------
3FEBh	PLUSW0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented – value of FSR0 offset by W								--------
3FEAh	FSR0H	—	—	Indirect Data Memory Address Pointer 0 High						--000000
3FE9h	FSR0L	Indirect Data Memory Address Pointer 0 Low								00000000
3FE8h	WREG	Working Register								xxxxxxxx
3FE7h	INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed								xxxxxxxx
3FE6h	POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented								--------
3FE5h	POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented								--------
3FE4h	PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented								--------
3FE3h	PLUSW1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented – value of FSR1 offset by W								--------
3FE2h	FSR1H	—	—	Indirect Data Memory Address Pointer 1 High						--000000
3FE1h	FSR1L	Indirect Data Memory Address Pointer 1 Low								00000000
3FE0h	BSR	—	—	Bank Select Register						--000000
3FDFh	INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed								xxxxxxxx
3FDEh	POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented								--------
3FDDh	POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented								--------
3FDCh	PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented								--------
3FDBh	PLUSW2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented – value of FSR2 offset by W								--------
3FDAh	FSR2H	—	—	Indirect Data Memory Address Pointer 2 High						--000000
3FD9h	FSR2L	Indirect Data Memory Address Pointer 2 Low								00000000
3FD8h	STATUS	—	$\overline{\text{TO}}$	$\overline{\text{PD}}$	N	OV	Z	DC	C	-1100000
3FD7h	IVTBASEU	—	—	—	BASE20	BASE19	BASE18	BASE17	BASE16	---00000
3FD6h	IVTBASEH	BASE15	BASE14	BASE13	BASE12	BASE11	BASE10	BASE9	BASE8	00000000
3FD5h	IVTBASEL	BASE7	BASE6	BASE5	BASE4	BASE3	BASE2	BASE1	BASE0	00001000
3FD4h	IVTLOCK	—	—	—	—	—	—	—	IVTLOCKED	-------0
3FD3h	INTCON1	STAT		—	—	—	—	—	—	00------
3FD2h	INTCON0	GIE	GIEL	IPEN	—	—	INT2EDG	INT1EDG	INT0EDG	000--111
3FD1h - 3FCFh	—	Unimplemented								—

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

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

R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1	R/W-1/1
I2C2TXIP	I2C2RXIP	DMA2AIP	DMA2ORIP	DMA2DCNTIP	DMA2SCNTIP	C2IP	INT1IP
bit 7							bit 0

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

bit 7 **I2C2TXIP:** I^{2}C2 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

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

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

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

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

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

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

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

FIGURE 13-1: WINDOWED WATCHDOG TIMER BLOCK DIAGRAM

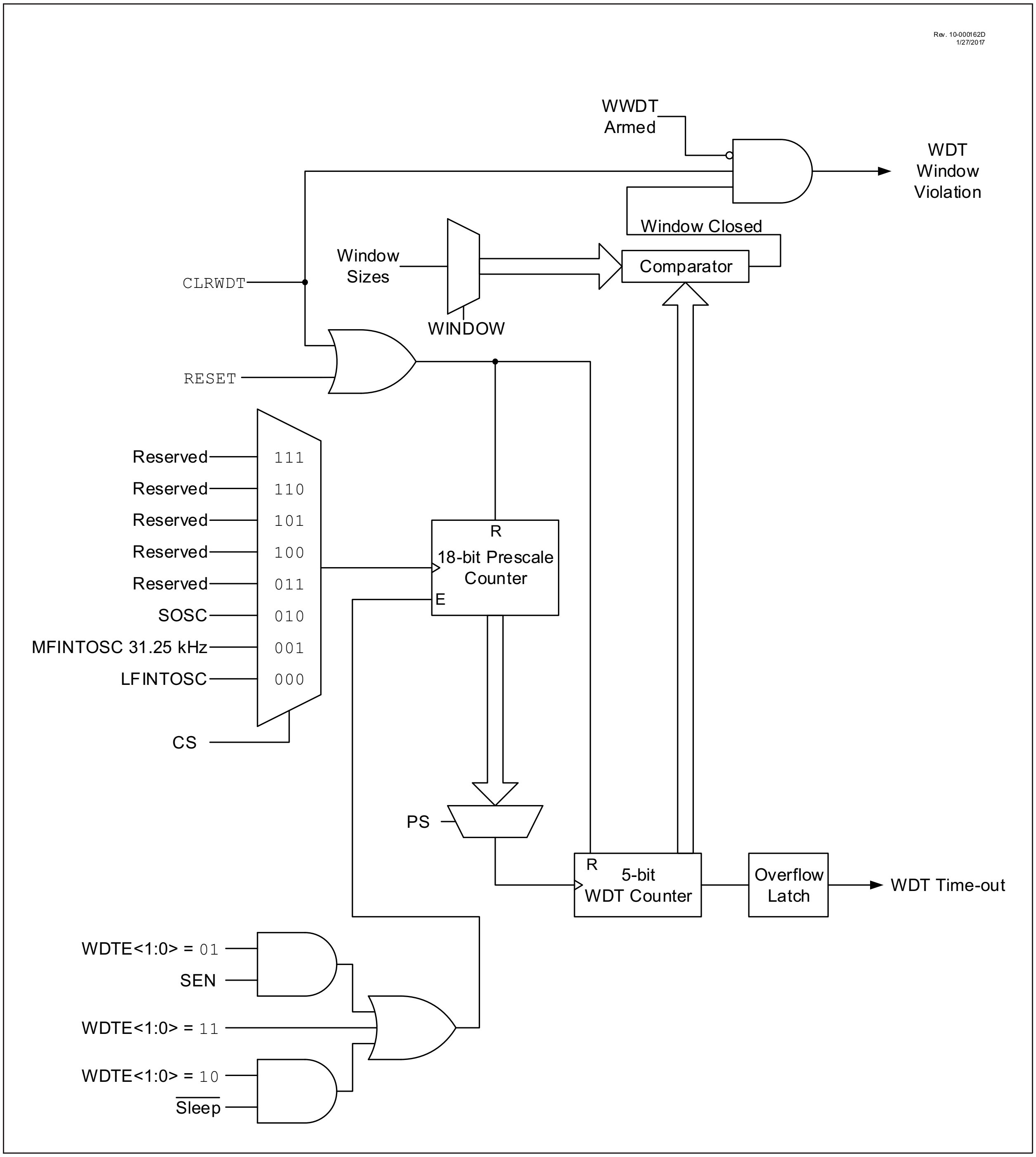
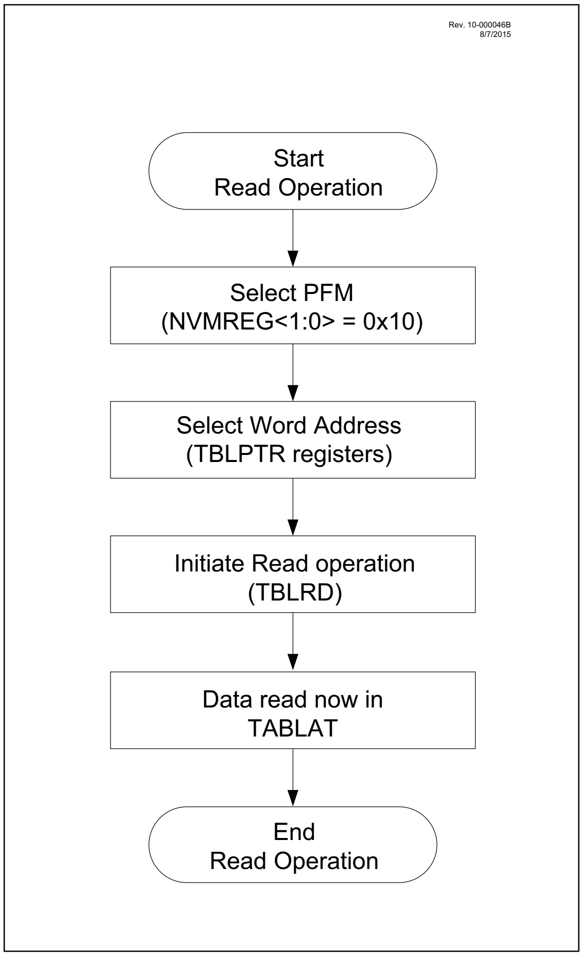


FIGURE 15-5: PROGRAM FLASH MEMORY READ FLOWCHART



20.0 INTERRUPT-ON-CHANGE

PORTA, PORTB, PORTC and pin RE3 of PORTE can be configured to operate as Interrupt-on-Change (IOC) pins on PIC18(L)F2x/4xK42 family devices. An interrupt can be generated by detecting a signal that has either a rising edge or a falling edge. Any individual port pin, or combination of port pins, can be configured to generate an interrupt. The interrupt-on-change module has the following features:

- Interrupt-on-Change enable (Master Switch)
- Individual pin configuration
- Rising and falling edge detection
- Individual pin interrupt flags

Figure 20-1 is a block diagram of the IOC module.

20.1 Enabling the Module

To allow individual port pins to generate an interrupt, the IOCIE bit of the PIE0 register must be set. If the IOCIE bit is disabled, the edge detection on the pin will still occur, but an interrupt will not be generated.

20.2 Individual Pin Configuration

For each port pin, a rising edge detector and a falling edge detector are present. To enable a pin to detect a rising edge, the associated bit of the IOCxP register is set. To enable a pin to detect a falling edge, the associated bit of the IOCxN register is set.

A pin can be configured to detect rising and falling edges simultaneously by setting both associated bits of the IOCxP and IOCxN registers, respectively.

20.3 Interrupt Flags

The IOCAFx, IOCBFx, IOCCFx and IOCEF3 bits located in the IOCAF, IOCBF, IOCCF and IOCEF registers respectively, are status flags that correspond to the interrupt-on-change pins of the associated port. If an expected edge is detected on an appropriately enabled pin, then the status flag for that pin will be set, and an interrupt will be generated if the IOCIE bit is set. The IOCIF bit of the PIR0 register reflects the status of all IOCAFx, IOCBFx, IOCCFx and IOCEF3 bits.

20.4 Clearing Interrupt Flags

The individual status flags, (IOCAFx, IOCBFx, IOCCFx and IOCEF3 bits), can be cleared by resetting them to zero. If another edge is detected during this clearing operation, the associated status flag will be set at the end of the sequence, regardless of the value actually being written.

In order to ensure that no detected edge is lost while clearing flags, only AND operations masking out known changed bits should be performed. The following sequence is an example of what should be performed.

EXAMPLE 20-1: CLEARING INTERRUPT FLAGS (PORTA EXAMPLE)

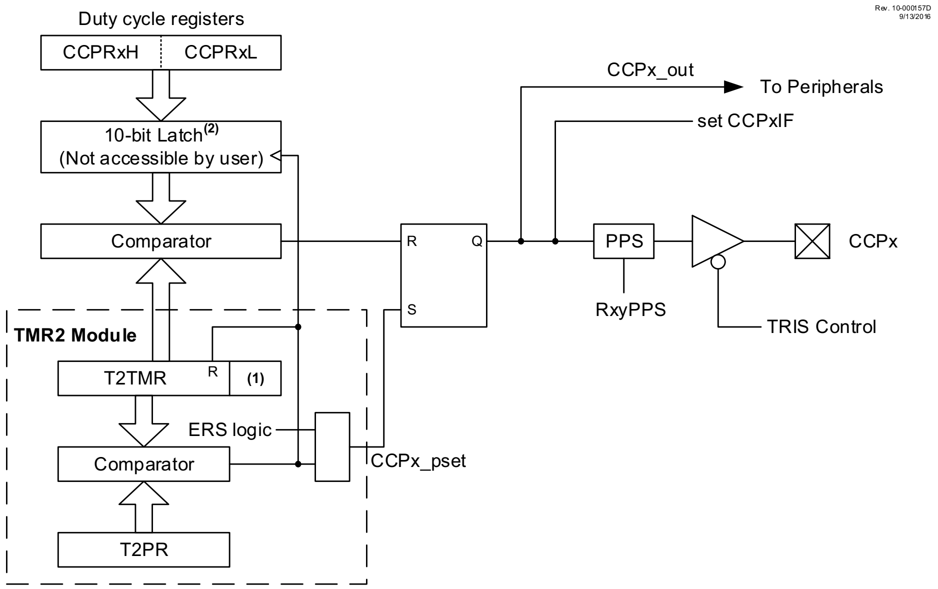
```
MOVLW   0xff
XORWF   IOCAF, W
ANDWF   IOCAF, F
```

20.5 Operation in Sleep

The interrupt-on-change interrupt sequence will wake the device from Sleep mode, if the IOCIE bit is set.

If an edge is detected while in Sleep mode, the IOCxF register will be updated prior to the first instruction executed out of Sleep.

FIGURE 25-4: SIMPLIFIED PWM BLOCK DIAGRAM



Notes: 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.
2. The alignment of the 10 bits from the CCPR register is determined by the CCPxFMT bit.

REGISTER 25-3: CCPxCAP: CAPTURE INPUT SELECTION MULTIPLEXER REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-0/x	R/W-0/x	R/W-0/x
—	—	—	—	—	CTS<2:0>		
bit 7							bit 0

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

bit 7-3 **Unimplemented:** Read as ‘0’

bit 2-0 **CTS<2:0>:** Capture Trigger Input Selection bits

CTS<1:0>	Connection			
	CCP1	CCP2	CCP3	CCP4
111	CLC4_out			
110	CLC3_out			
101	CLC2_out			
100	CLC1_out			
011	IOC_Interrupt			
010	CMP2_output			
001	CMP1_output			
000	Pin selected by CCP1PPS	Pin selected by CCP2PPS	Pin selected by CCP3PPS	Pin selected by CCP4PPS

REGISTER 25-4: CCPRxL: CCPx REGISTER LOW BYTE

R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x	R/W-x/x
RL<7:0>							
bit 7							bit 0

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

bit 7-0 MODE = Capture Mode:
RL<7:0>: LSB of captured TMR1 value
MODE = Compare Mode:
RL<7:0>: LSB compared to TMR1 value
MODE = PWM Mode && FMT = 0:
RL<7:0>: CCPW<7:0> – Pulse-Width LS 8 bits
MODE = PWM Mode && FMT = 1:
RL<7:6>: CCPW<1:0> – Pulse-Width LS 2 bits
RL<5:0>: Not used

REGISTER 26-2: CCPTMRS1: CCP TIMERS CONTROL REGISTER 1

R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1	R/W-0/0	R/W-1/1
P8TSEL<1:0>		P7TSEL<1:0>		P6TSEL<1:0>		P5TSEL<1:0>	
bit 7							bit 0

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

bit 7-6 **P8TSEL<1:0>:** PWM8 Timer Selection bits
11 = PWM8 based on TMR6
10 = PWM8 based on TMR4
01 = PWM8 based on TMR2
00 = Reserved

bit 5-4 **P7TSEL<1:0>:** PWM7 Timer Selection bits
11 = PWM7 based on TMR6
10 = PWM7 based on TMR4
01 = PWM7 based on TMR2
00 = Reserved

bit 3-2 **P6TSEL<1:0>:** PWM6 Timer Selection bits
11 = PWM6 based on TMR6
10 = PWM6 based on TMR4
01 = PWM6 based on TMR2
00 = Reserved

bit 1-0 **P5TSEL<1:0>:** PWM5 Timer Selection bits
11 = PWM5 based on TMR6
10 = PWM5 based on TMR4
01 = PWM5 based on TMR2
00 = Reserved

28.0 COMPLEMENTARY WAVEFORM GENERATOR (CWG) MODULE

The Complementary Waveform Generator (CWG) produces half-bridge, full-bridge, and steering of PWM waveforms. It is backwards compatible with previous CCP functions. The PIC18(L)F2x/4xK42 family has three instances of the CWG module.

Each of the CWG modules has the following features:

- Six operating modes:
  - Synchronous Steering mode
  - Asynchronous Steering mode
  - Full-Bridge mode, Forward
  - Full-Bridge mode, Reverse
  - Half-Bridge mode
  - Push-Pull mode
- Output polarity control
- Output steering
- Independent 6-bit rising and falling event dead-band timers
  - Clocked dead band
  - Independent rising and falling dead-band enables
- Auto-shutdown control with:
  - Selectable shutdown sources
  - Auto-restart option
  - Auto-shutdown pin override control

28.1 Fundamental Operation

The CWG generates two output waveforms from the selected input source.

The off-to-on transition of each output can be delayed from the on-to-off transition of the other output, thereby, creating a time delay immediately where neither output is driven. This is referred to as dead time and is covered in **Section 28.6 "Dead-Band Control"**.

It may be necessary to guard against the possibility of circuit faults or a feedback event arriving too late or not at all. In this case, the active drive must be terminated before the Fault condition causes damage. This is referred to as auto-shutdown and is covered in **Section 28.10 "Auto-Shutdown"**.

28.2 Operating Modes

The CWG module can operate in six different modes, as specified by the MODE<2:0> bits of the CWGxCON0 register:

- Half-Bridge mode
- Push-Pull mode
- Asynchronous Steering mode
- Synchronous Steering mode
- Full-Bridge mode, Forward
- Full-Bridge mode, Reverse

All modes accept a single pulse data input, and provide up to four outputs as described in the following sections.

All modes include auto-shutdown control as described in **Section 28.10 "Auto-Shutdown"**

Note:	Except as noted for Full-bridge mode (**Section 28.2.3 "Full-Bridge Modes"**), mode changes should only be performed while EN = 0 (Register 28-1).

28.2.1 HALF-BRIDGE MODE

In Half-Bridge mode, two output signals are generated as true and inverted versions of the input as illustrated in Figure 28-2. A non-overlap (dead-band) time is inserted between the two outputs as described in **Section 28.6 "Dead-Band Control"**. The output steering feature cannot be used in this mode. A basic block diagram of this mode is shown in Figure 28-1.

The unused outputs CWGxC and CWGxD drive similar signals as CWGxA and CWGxB, with polarity independently controlled by the POLC and POLD bits of the CWGxCON1 register, respectively.

FIGURE 29-3: PROGRAMMABLE LOGIC FUNCTIONS

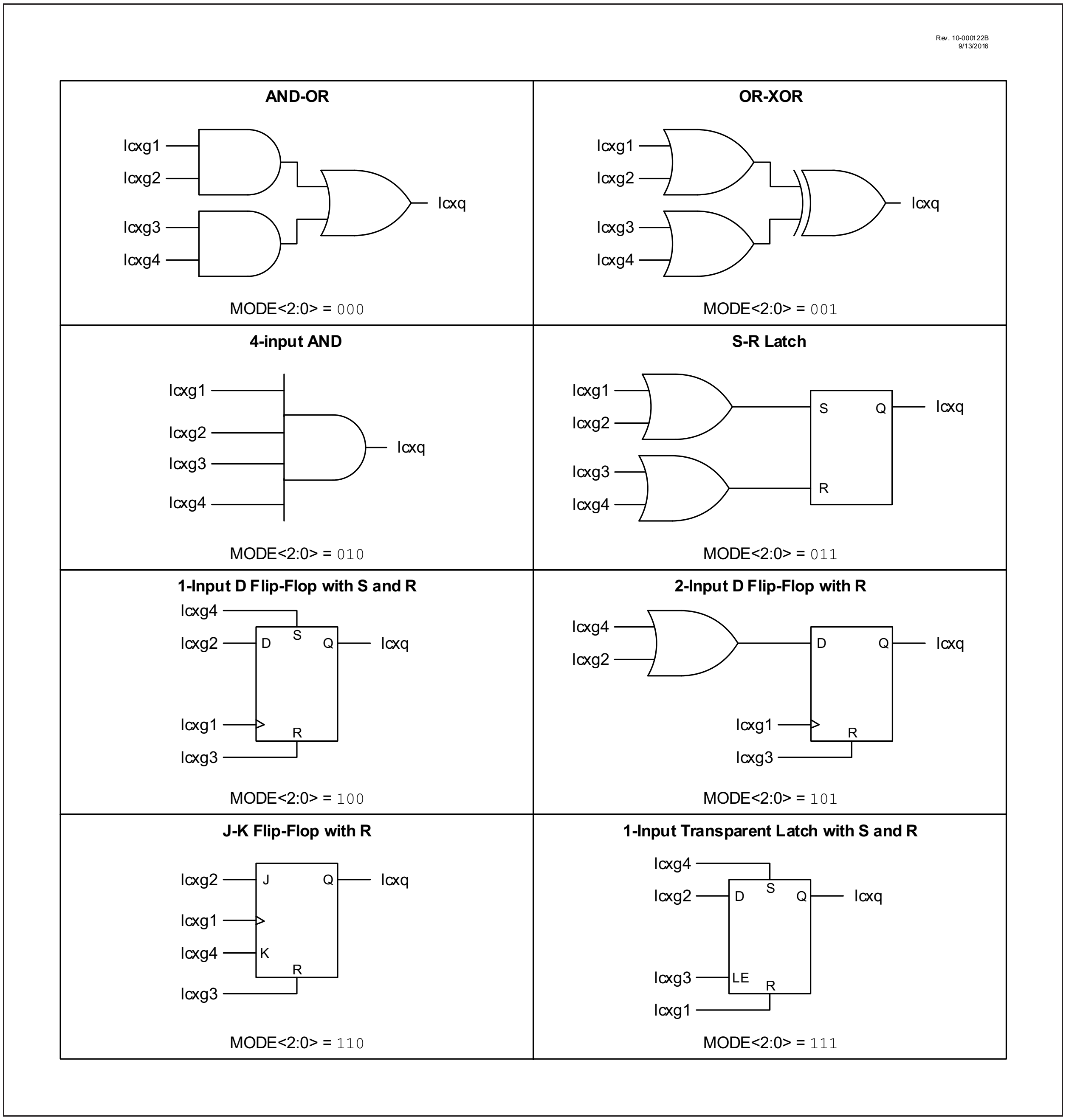
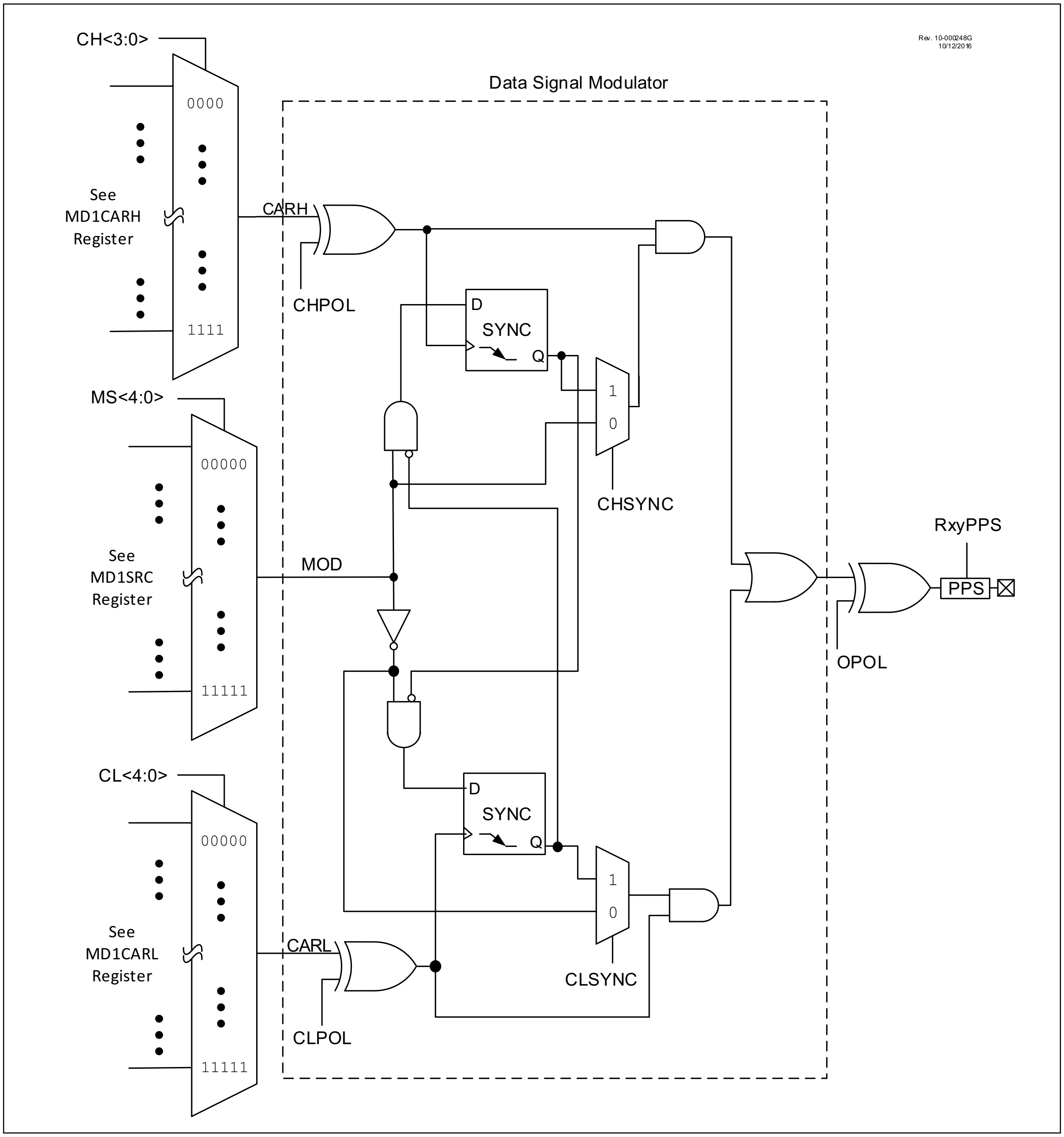


FIGURE 32-1: SIMPLIFIED BLOCK DIAGRAM OF THE DATA SIGNAL MODULATOR



33.2.1.8 Asynchronous Transmission Setup

1. Initialize the UxBRGH, UxBRGL register pair and the BRGS bit to achieve the desired baud rate (see **Section 33.17 "UART Baud Rate Generator (BRG)"**).
2. Set the MODE<3:0> bits to the desired asynchronous mode.
3. Set TXPOL bit if inverted TX output is desired.
4. Enable the asynchronous serial port by setting the ON bit.
5. Enable the transmitter by setting the TXEN control bit. This will cause the UxTXIF interrupt flag to be set.
6. If the device has PPS, configure the desired I/O pin RxyPPS register with the code for TX output.
7. If interrupts are desired, set the UxTXIE interrupt enable bit in the respective PIE register. An interrupt will occur immediately provided that the GIE bits in the INTCON0 register are also set.
8. Write one byte of data into the UxTXB register. This will start the transmission.
9. Subsequent bytes may be written when the UxTXIF bit is '1'.

FIGURE 33-3: ASYNCHRONOUS TRANSMISSION

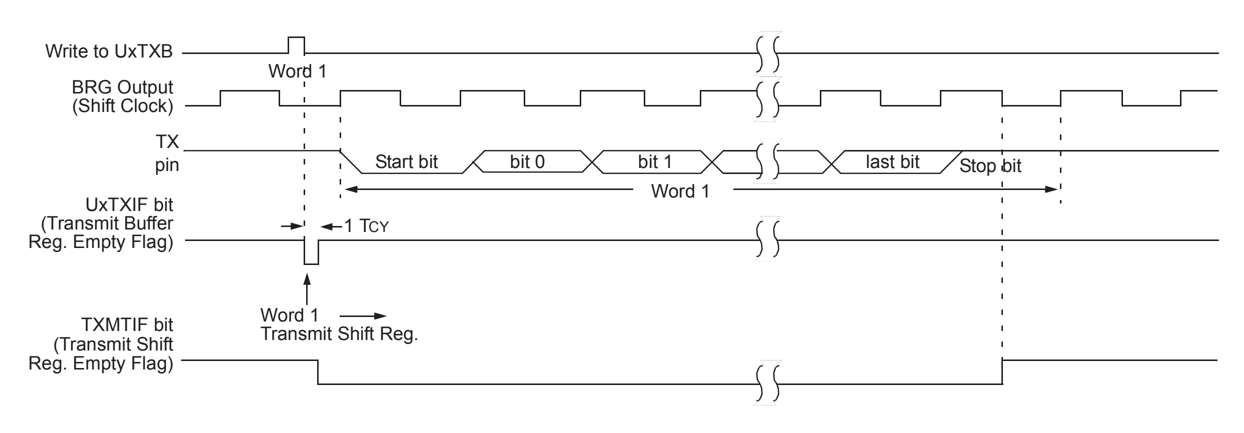
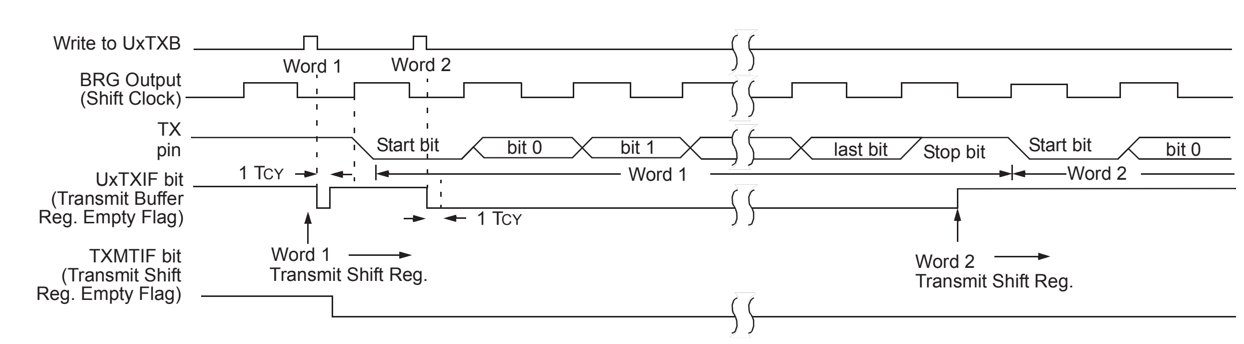

Write to UxTXB
Word 1
BRG Output (Shift Clock)
TX pin
Start bit
bit 0
bit 1
last bit
Stop bit
Word 1
UxTXIF bit (Transmit Buffer Reg. Empty Flag)
1 TCY
TXMTIF bit (Transmit Shift Reg. Empty Flag)
Word 1 Transmit Shift Reg.

FIGURE 33-4: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)


Write to UxTXB
Word 1
Word 2
BRG Output (Shift Clock)
TX pin
Start bit
bit 0
bit 1
last bit
Stop bit
Start bit
bit 0
UxTXIF bit (Transmit Buffer Reg. Empty Flag)
1 TCY
Word 1
Word 2
1 TCY
TXMTIF bit (Transmit Shift Reg. Empty Flag)
Word 1 Transmit Shift Reg.
Word 2 Transmit Shift Reg.

Note: This timing diagram shows the first transmission and the start of the second consecutive transmission.

33.16 Clock Accuracy with Asynchronous Operation

The factory calibrates the internal oscillator block output (INTOSC). However, the INTOSC frequency may drift as VDD or temperature changes, and this directly affects the asynchronous baud rate. Two methods may be used to adjust the baud rate clock, but both require a reference clock source of some kind.

The first (preferred) method uses the OSCTUNE register to adjust the INTOSC output. Adjusting the value of the OSCTUNE register allows for fine resolution changes to the system clock source. See **Section 6.2.2.3 "Internal Oscillator Frequency Adjustment"** for more information.

The other method adjusts the value of the Baud Rate Generator. This can be done automatically with the Auto-Baud Detect feature (see **Section 33.17.1 "Auto-Baud Detect"**). There may not be fine enough resolution when adjusting the Baud Rate Generator to compensate for a gradual change of the peripheral clock frequency.

33.17 UART Baud Rate Generator (BRG)

The Baud Rate Generator (BRG) is a 16-bit timer that is dedicated to the support of the UART operation.

The UxBRGH, UxBRGL register pair determines the period of the free running baud rate timer. The multiplier of the baud rate period is determined by the BRGS bit in the UxCON0 register.

Table 33-1 contains the formulas for determining the baud rate. Example 33-1 provides a sample calculation for determining the baud rate and baud rate error.

The high baud rate range (BRGS = `1`) is intended to extend the baud rate range up to a faster rate when the desired baud rate is not possible otherwise. Using the normal baud rate range (BRGS = `0`) is recommended when the desired baud rate is achievable with either range.

Writing a new value to the UxBRGH, UxBRGL register pair causes the BRG timer to be reset (or cleared). This ensures that the BRG does not wait for a timer overflow before outputting the new baud rate.

If the system clock is changed during an active receive operation, a receive error or data loss may result. To avoid this problem, check the status of the RXIDL bit to make sure that the receive operation is idle before changing the system clock.

EXAMPLE 33-1: CALCULATING BAUD RATE ERROR

For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, BRGS = `0`:

$$\text{Desired Baud Rate} = \frac{F_{OSC}}{16([UxBRG] + 1)}$$

$$X = \frac{\dfrac{F_{OSC}}{\text{Desired Baud Rate}}}{16} - 1$$

$$= \frac{\dfrac{16000000}{9600}}{16} - 1$$

$$= [103.17] = 103$$

$$\text{Calculated Baud Rate} = \frac{16000000}{16(103 + 1)}$$

$$= 9615$$

$$\text{Error} = \frac{\text{Calc. Baud Rate} - \text{Desired Baud Rate}}{\text{Desired Baud Rate}}$$

$$= \frac{(9615 - 9600)}{9600} = 0.16\%$$

TABLE 33-1: BAUD RATE FORMULAS

BRGS	BRG/UART Mode	Baud Rate Formula
1	High Rate	FOSC/[4 (n+1)]
0	Normal Rate	FOSC/[16(n+1)]

Legend: n = value of UxBRGH, UxBRGL register pair.

39.6 Register Definitions: DAC Control

Long bit name prefixes for the DAC peripheral is shown in Table 39-1. Refer to **Section 1.4.2.2 "Long Bit Names"** for more information.

TABLE 39-1:

Peripheral	Bit Name Prefix
DAC1	DAC1

REGISTER 39-1: DAC1CON0: DAC CONTROL REGISTER

R/W-0/0	U-0	R/W-0/0	R/W-0/0	R/W-0/0	R/W-0/0	U-0	R/W-0/0
EN	—	OE1	OE2	PSS<1:0>		—	NSS
bit 7							bit 0

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

bit 7 **EN:** DAC Enable bit
1 = DAC is enabled
0 = DAC is disabled[(1)]

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

bit 5 **OE1:** DAC Voltage Output Enable bit
1 = DAC voltage level is output on the DAC1OUT1 pin
0 = DAC voltage level is disconnected from the DAC1OUT1 pin

bit 4 **OE2:** DAC Voltage Output Enable bit
1 = DAC voltage level is output on the DAC1OUT2 pin
0 = DAC voltage level is disconnected from the DAC1OUT2 pin

bit 3-2 **PSS<1:0>:** DAC Positive Source Select bit
11 = Reserved
10 = FVR buffer 2
01 = VREF+
00 = VDD

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

bit 0 **NSS:** DAC Negative Source Select bit
1 = VREF-
0 = VSS

Note 1: DAC1OUTx output pins are still active.

NEGF — Negate f

Syntax: NEGF f {,a}

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

Operation: $(\overline{f}) + 1 \rightarrow f$

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

Encoding:

`0110`	`110a`	`ffff`	`ffff`

Description: Location ‘f’ is negated using two’s complement. The result is placed in the data memory location ‘f’.
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 43.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `NEGF    REG, 1`

Before Instruction
REG = `0011 1010` [3Ah]
After Instruction
REG = `1100 0110` [C6h]

NOP — No Operation

Syntax: NOP

Operands: None

Operation: No operation

Status Affected: None

Encoding:

`0000`
`1111`
	`0000`
	`xxxx`
		`0000`
		`xxxx`
			`0000`
			`xxxx`

Description: No operation.

Words: 1

Cycles: 1

Q Cycle Activity:

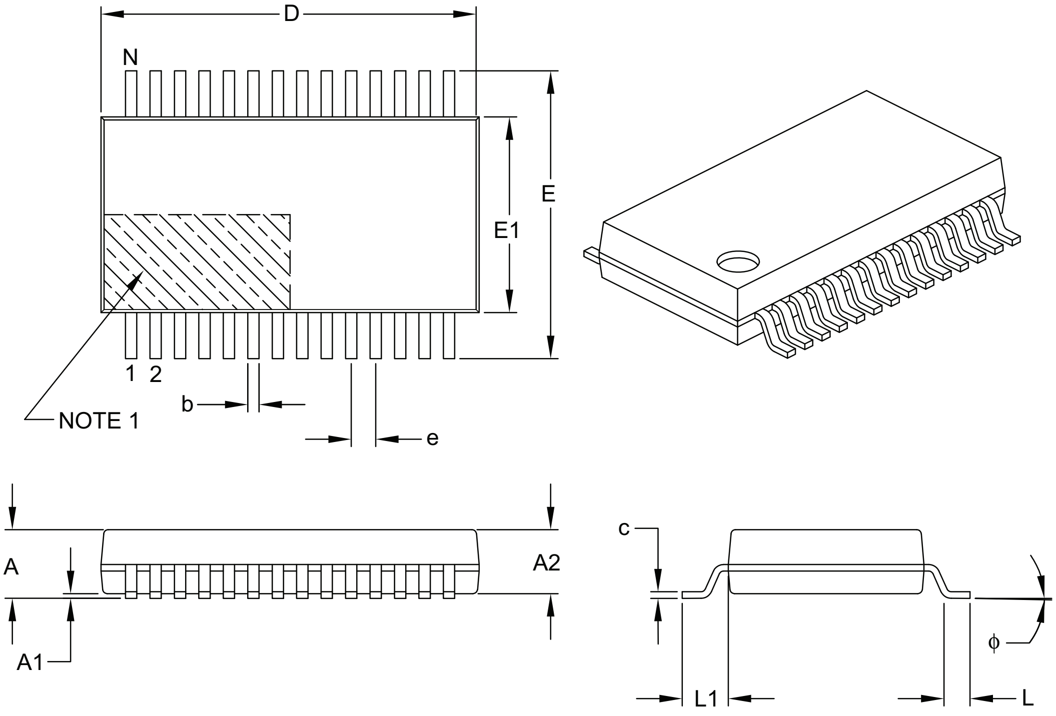
Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation

Example:

None.

28-Lead Plastic Shrink Small Outline (SS) – 5.30 mm Body [SSOP]

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	–	–	2.00
Molded Package Thickness	A2	1.65	1.75	1.85
Standoff	A1	0.05	–	–
Overall Width	E	7.40	7.80	8.20
Molded Package Width	E1	5.00	5.30	5.60
Overall Length	D	9.90	10.20	10.50
Foot Length	L	0.55	0.75	0.95
Footprint	L1	1.25 REF		
Lead Thickness	c	0.09	–	0.25
Foot Angle	ϕ	0°	4°	8°
Lead Width	b	0.22	–	0.38

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.20 mm per side.
3. Dimensioning and tolerancing per ASME Y14.5M.

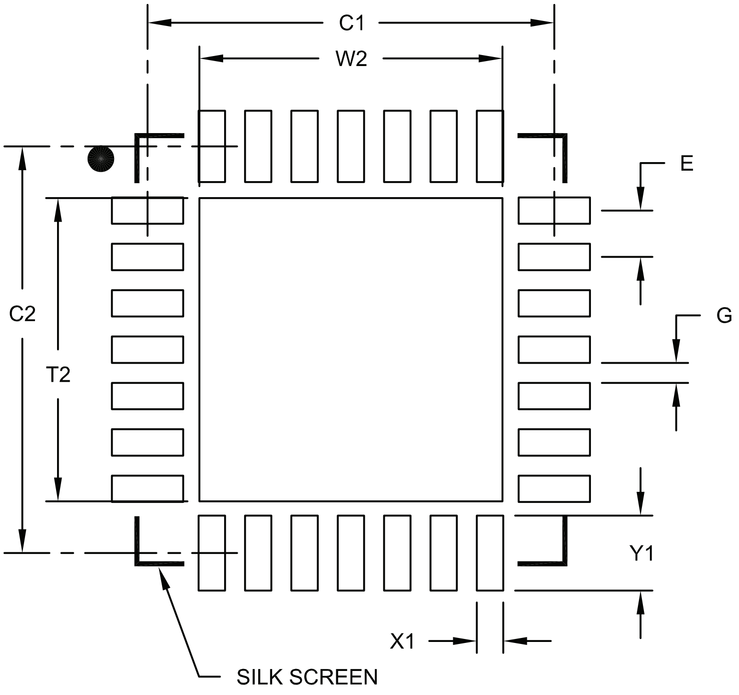
BSC: Basic Dimension. Theoretically exact value shown without tolerances.

REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-073B

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

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	W2			4.25
Optional Center Pad Length	T2			4.25
Contact Pad Spacing	C1		5.70	
Contact Pad Spacing	C2		5.70	
Contact Pad Width (X28)	X1			0.37
Contact Pad Length (X28)	Y1			1.00
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. C04-2105A