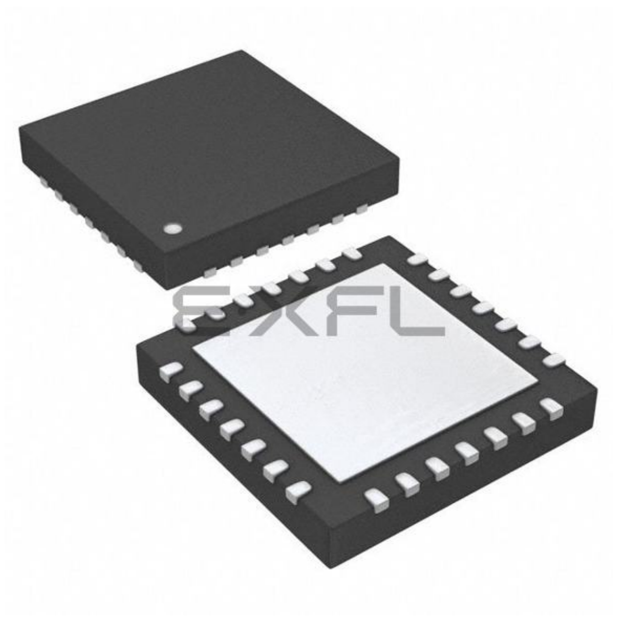


What is "Embedded - Microcontrollers"?

"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, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, POR, PWM, WDT
Number of I/O	25
Program Memory Size	32KB (32K x 8)
Program Memory Type	FLASH
EEPROM Size	256 x 8
RAM Size	2K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 24x10b; D/A 1x5b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-VQFN Exposed Pad
Supplier Device Package	28-QFN (6x6)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic18f25q10t-i-ml

3.9.2 REVISION ID

Name: REVISION ID
Address: 0x3FFFFC

Revision ID Register

Bit	15	14	13	12	11	10	9	8
	1010[3:0]				MJRREV[5:2]			
Access	RO	RO	RO	RO	RO	RO	RO	RO
Reset	1	0	1	0	q	q	q	q

Bit	7	6	5	4	3	2	1	0
	MJRREV[1:0]		MNRREV[5:0]					
Access	RO	RO	RO	RO	RO	RO	RO	RO
Reset	q	q	q	q	q	q	q	q

Bits 15:12 – `1010`[3:0] Read as '1010'
These bits are fixed with value '`1010`' for all devices in this family.

Bits 11:6 – MJRREV[5:0] Major Revision ID bits
These bits are used to identify a major revision. A major revision is indicated by an all-layer revision (A0, B0, C0, etc.).

Revision A = b'00 0000'

Bits 5:0 – MNRREV[5:0] Minor Revision ID bits
These bits are used to identify a minor revision.

readable or writable. Updates to the PCH register are performed through the PCLATH register. The upper byte is called PCU. This register contains the PC<20:16> bits; it is also not directly readable or writable. Updates to the PCU register are performed through the PCLATU register.

The contents of PCLATH and PCLATU are transferred to the Program Counter by any operation that writes PCL. Similarly, the upper two bytes of the Program Counter are transferred to PCLATH and PCLATU by an operation that reads PCL. This is useful for computed offsets to the PC (see 10.1.3.1 Computed GOTO).

The PC addresses bytes in the program memory. To prevent the PC from becoming misaligned with word instructions, the Least Significant bit of PCL is fixed to a value of ‘0’. The PC increments by two to address sequential instructions in the program memory.

The `CALL`, `RCALL`, `GOTO` and program branch instructions write to the Program Counter directly. For these instructions, the contents of PCLATH and PCLATU are not transferred to the Program Counter.

10.1.2 Return Address Stack

The return address stack allows any combination of up to 31 program calls and interrupts to occur. The PC is pushed onto the stack when a `CALL` or `RCALL` instruction is executed or an interrupt is Acknowledged. The PC value is pulled off the stack on a `RETURN`, `RETLW` or a `RETFIE` instruction. PCLATU and PCLATH are not affected by any of the `RETURN` or `CALL` instructions.

The stack operates as a 31-word by 21-bit RAM and a 5-bit Stack Pointer, or as a 35-word by 21-bit RAM with a 6-bit Stack Pointer in ICD mode. The stack space is not part of either program or data space. The Stack Pointer is readable and writable and the address on the top of the stack is readable and writable through the Top-of-Stack (TOS) Special File registers. Data can also be pushed to, or popped from the stack, using these registers.

A `CALL` type instruction causes a push onto the stack; the Stack Pointer is first incremented and the location pointed to by the Stack Pointer is written with the contents of the PC (already pointing to the instruction following the `CALL`). A `RETURN` type instruction causes a pop from the stack; the contents of the location pointed to by the STKPTR are transferred to the PC and then the Stack Pointer is decremented.

The Stack Pointer is initialized to ‘0b00000’ after all Resets. There is no RAM associated with the location corresponding to a Stack Pointer value of ‘0b00000’; this is only a Reset value. Status bits in the PCON0 register indicate if the stack is full or has overflowed or has under-flowed.

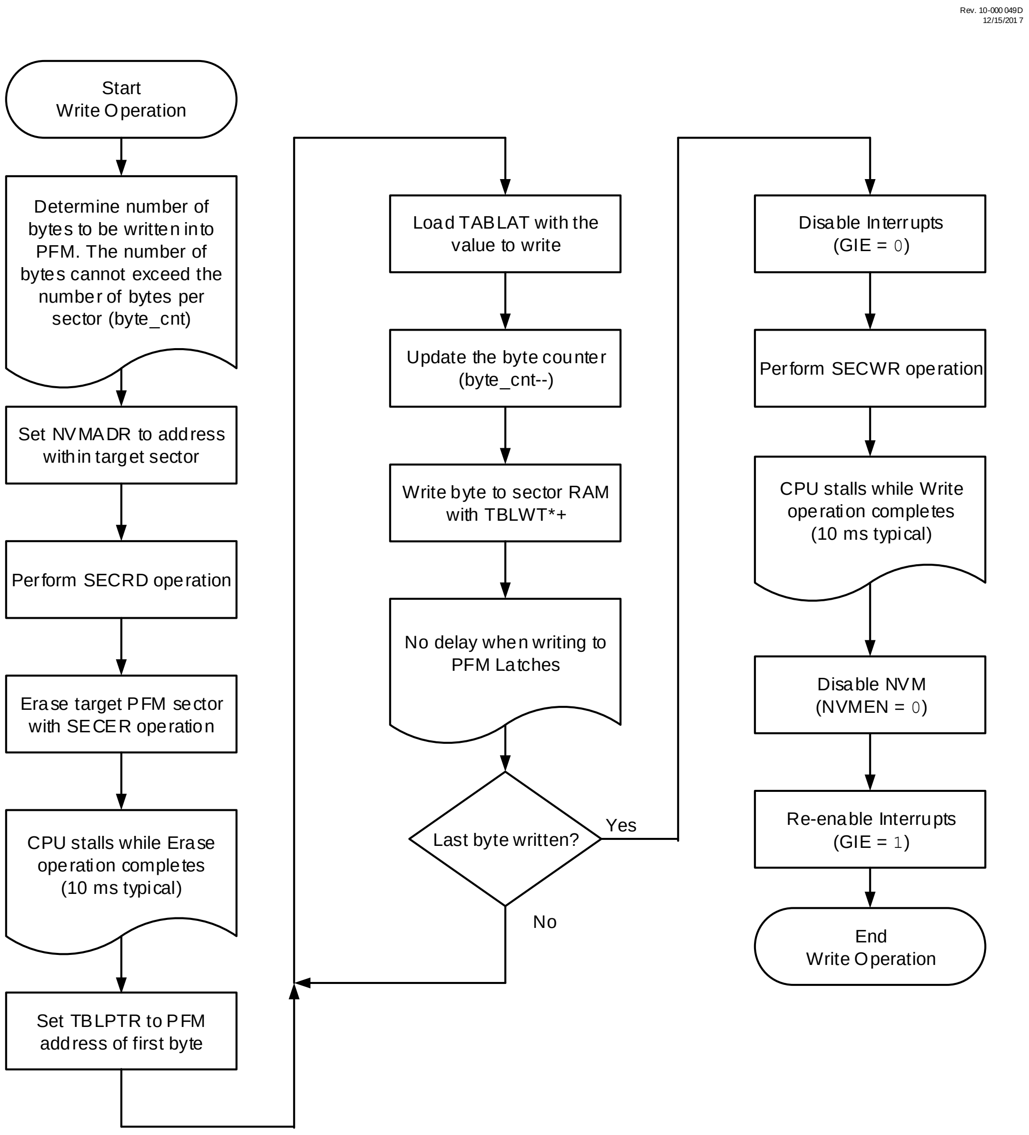
10.1.2.1 Top-of-Stack Access

Only the top of the return address stack (TOS) is readable and writable. A set of three registers, TOSU:TOSH:TOSL, hold the contents of the stack location pointed to by the STKPTR register (see Figure 10-3). This allows users to implement a software stack if necessary. After a `CALL, RCALL` or interrupt, the software can read the pushed value by reading the TOSU:TOSH:TOSL registers. These values can be placed on a user defined software stack. At return time, the software can return these values to TOSU:TOSH:TOSL and do a return.

The user must disable the Global Interrupt Enable (GIE) bits while accessing the stack to prevent inadvertent stack corruption.

Important: Before setting the SECWR bit, the NVMADR value needs to be within the intended address range of the target PFM sector.

Figure 11-9. Program Flash Memory (PFM) Write Flowchart


Rev. 10-000049D
12/15/2017
Start
Write Operation
Determine number of bytes to be written into PFM. The number of bytes cannot exceed the number of bytes per sector (byte_cnt)
Set NVMADR to address within target sector
Perform SECRD operation
Erase target PFM sector with SECER operation
CPU stalls while Erase operation completes (10 ms typical)
Set TBLPTR to PFM address of first byte
Load TABLAT with the value to write
Update the byte counter (byte_cnt--)
Write byte to sector RAM with TBLWT*+
No delay when writing to PFM Latches
Last byte written?
Yes
No
Disable Interrupts (GIE = 0)
Perform SECWR operation
CPU stalls while Write operation completes (10 ms typical)
Disable NVM (NVMEN = 0)
Re-enable Interrupts (GIE = 1)
End
Write Operation

Example 11-4. Writing a Sector of Program Flash Memory

```
; Code sequence to modify one word in a programmed sector of PFM
; Calling routine should check WREG for the following errors:
;
;     00h = Successful modification
;     01h = Read error
;     02h = Erase error
;     03h = Write error
;
READ_BLOCK:
        MOVLW   CODE_ADDR_UPPER     ; load NVMADR with the base
        MOVWF   NVMADRU             ; address of the memory sector
        MOVLW   CODE_ADDR_HIGH
```

14.13.5 PIR3

Name: PIR3
Address: 0xEC8

Peripheral Interrupt Request (Flag) Register 3

Bit	7	6	5	4	3	2	1	0
			RC1IF	TX1IF			BCL1IF	SSP1IF
Access			R	R			R/W	R/W
Reset			0	0			0	0

Bit 5 – RCxIF EUSARTx Receive Interrupt Flag bit

Value	Description
1	The EUSARTx receive buffer, RCxREG, is full (cleared by reading RCxREG)
0	The EUSARTx receive buffer is empty

Bit 4 – TXxIF EUSARTx Transmit Interrupt Flag bit

Value	Description
1	The EUSARTx transmit buffer, TXxREG, is empty (cleared by writing TXxREG)
0	The EUSARTx transmit buffer is full

Bit 1 – BCLxIF MSSPx Bus Collision Interrupt Flag bit

Value	Description
1	A bus collision has occurred while the MSSPx module configured in I^{2}C master was transmitting (must be cleared in software)
0	No bus collision occurred

Bit 0 – SSPxIF Synchronous Serial Port ‘x’ Interrupt Flag bit

Value	Description
1	The transmission/reception is complete (must be cleared in software)
0	Waiting to transmit/receive

15.5.24 INLVLA

Name: INLVLA
Address: 0xF08

Input Level Control Register

Bit	7	6	5	4	3	2	1	0
	INLVLA7	INLVLA6	INLVLA5	INLVLA4	INLVLA3	INLVLA2	INLVLA1	INLVLA0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	1	1	1	1	1	1	1	1

Bits 0, 1, 2, 3, 4, 5, 6, 7 – INLVLAn Input Level Select on Pins Rx<7:0>, respectively

Value	Description
1	ST input used for port reads and interrupt-on-change
0	TTL input used for port reads and interrupt-on-change

18.1 Timer0 Operation

Timer0 can operate as either an 8-bit or 16-bit timer. The mode is selected with the T016BIT bit.

18.1.1 8-bit Mode

In this mode Timer0 increments on the rising edge of the selected clock source. A prescaler on the clock input gives several prescale options (see prescaler control bits, T0CKPS).

In this mode as shown in Figure 18-1, a buffered version of TMR0H is maintained. This is compared with the value of TMR0L on each cycle of the selected clock source. When the two values match, the following events occur:

- TMR0L is reset
- The contents of TMR0H are copied to the TMR0H buffer for next comparison

18.1.2 16-Bit Mode

In this mode Timer0 increments on the rising edge of the selected clock source. A prescaler on the clock input gives several prescale options (see prescaler control bits, T0CKPS).

In this mode TMR0H:TMR0L form the 16-bit timer value. As shown in Figure 18-1, read and write of the TMR0H register are buffered. TMR0H register is updated with the contents of the high byte of Timer0 during a read of TMR0L register. Similarly, a write to the high byte of Timer0 takes place through the TMR0H buffer register. The high byte is updated with the contents of TMR0H register when a write occurs to TMR0L register. This allows all 16 bits of Timer0 to be read and written at the same time.

Timer0 rolls over to 0x0000 on incrementing past 0xFFFF. This makes the timer free running. TMR0L/H registers cannot be reloaded in this mode once started.

18.2 Clock Selection

Timer0 has several options for clock source selections, option to operate synchronously/asynchronously and a programmable prescaler.

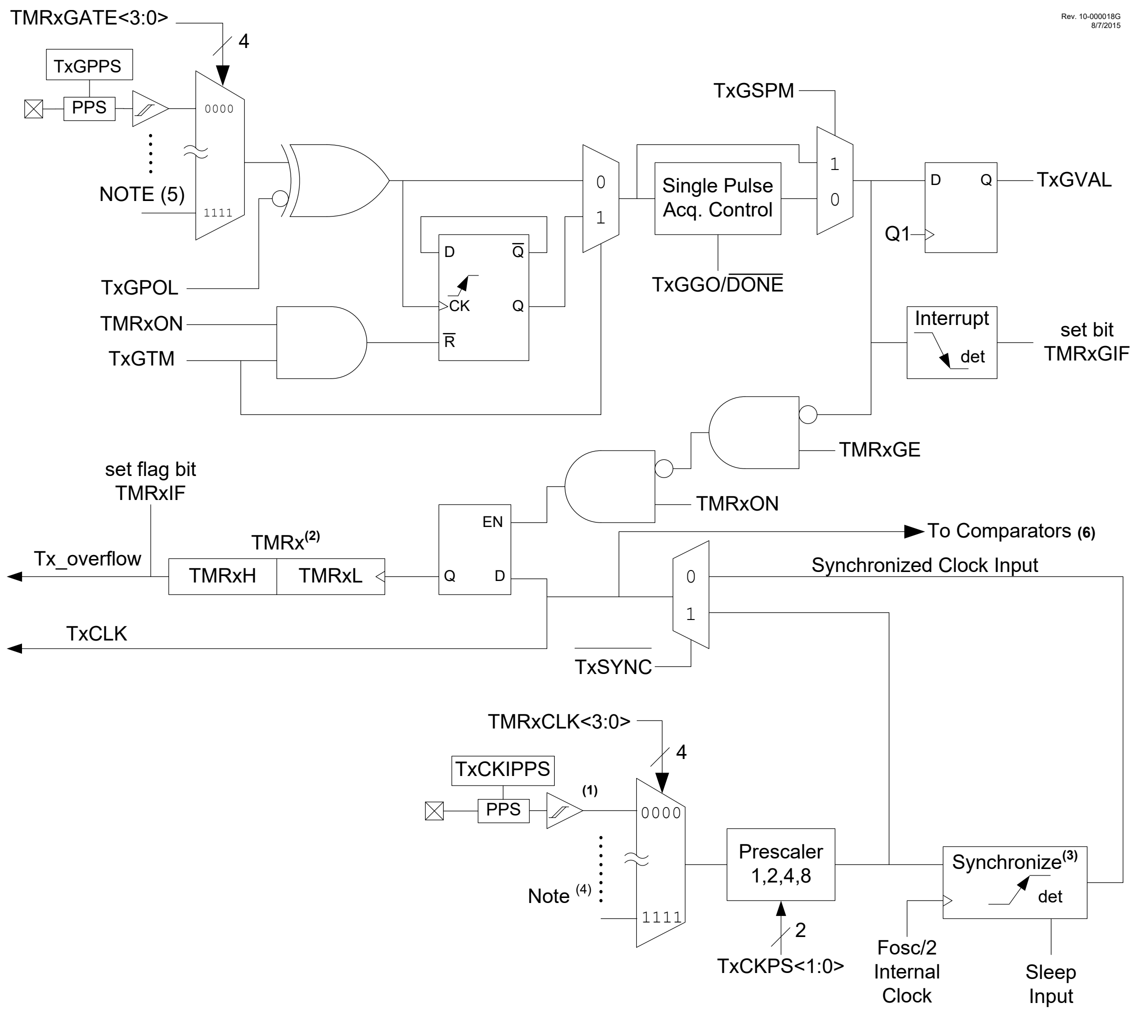
18.2.1 Clock Source Selection

The T0CS bits are used to select the clock source for Timer0. The possible clock sources are listed in the table below.

Table 18-1. Timer 0 Clock Source Selections

T0CS	Clock Source
111	Reserved
110	Reserved
101	SOSC
100	LFINTOSC
011	HFINTOSC
010	Fosc/4
001	Pin selected by T0CKIPPS (Inverted)
000	Pin selected by T0CKIPPS (Non-inverted)

Figure 19-1. Timer1 Block Diagram



Note:

1. This signal comes from the pin seleted by TxCKIPPS.
2. TMRx register increments on rising edge.
3. Synchronize does not operate while in Sleep.
4. See TMRxCLK for clock source selections.
5. See TMRxGATE for gate source selection.
6. Synchronized comparator output should not be used in conjunction with synchronized input clock.

19.1 Timer1 Operation

The Timer1 module is a 16-bit incrementing counter that is accessed through the TMRxH:TMRxL register pair. Writes to TMRxH or TMRxL directly update the counter.

When used with an internal clock source, the module is a timer and increments on every instruction cycle. When used with an external clock source, the module can be used as either a timer or counter and increments on every selected edge of the external source.

Timer1 is enabled by configuring the ON and GE bits in the TxCON and TxGCON registers, respectively. The table below displays the Timer1 enable selections.

  - Configure the T2CKPS bits of the T2CON register with the Timer2 prescale value.
  - Enable Timer2 by setting the T2ON bit of the T2CON register.
6. Enable PWM output pin and wait until Timer2 overflows, TMR2IF bit of the PIRx register is set.(2)
7. Enable the PWMx pin output driver(s) by clearing the associated TRIS bit(s) and setting the desired pin PPS control bits.
8. Configure the PWM module by loading the PWMxCON register with the appropriate values.

Note:

1. In order to send a complete duty cycle and period on the first PWM output, the above steps must be followed in the order given. If it is not critical to start with a complete PWM signal, then move Step 8 to replace Step 4.
2. For operation with other peripherals only, disable PWMx pin outputs.

22.9.1 PWMx Pin Configuration

All PWM outputs are multiplexed with the PORT data latch. The user must configure the pins as outputs by clearing the associated TRIS bits.

22.10 Setup for PWM Operation to Other Device Peripherals

The following steps should be taken when configuring the module for PWM operation to be used by other device peripherals:

1. Disable the PWMx pin output driver(s) by setting the associated TRIS bit(s).
2. Clear the PWMxCON register.
3. Load the T2PR register with the PWM period value.
4. Load the PWMxDCH register and bits <7:6> of the PWMxDCL register with the PWM duty cycle value.
5. Configure and start Timer2:
   - Clear the TMR2IF interrupt flag bit of the PIRx register.(1)
   - Select the timer clock source to be as $F_{OSC}/4$ using the TxCLKCON register. This is required for correct operation of the PWM module.
   - Configure the T2CKPS bits of the T2CON register with the Timer2 prescale value.
   - Enable Timer2 by setting the T2ON bit of the T2CON register.
6. Wait until Timer2 overflows, TMR2IF bit of the PIRx register is set.(1)
7. Configure the PWM module by loading the PWMxCON register with the appropriate values.

Note:

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

22.12.3 PWMxDC

Name: PWMxDC
Address: 0xFA2,0xF9F

PWM Duty Cycle Register

Bit	15	14	13	12	11	10	9	8
	DCH[7:0]							
Access								
Reset	x	x	x	x	x	x	x	x

Bit	7	6	5	4	3	2	1	0
	DCL[1:0]							
Access								
Reset	x	x						

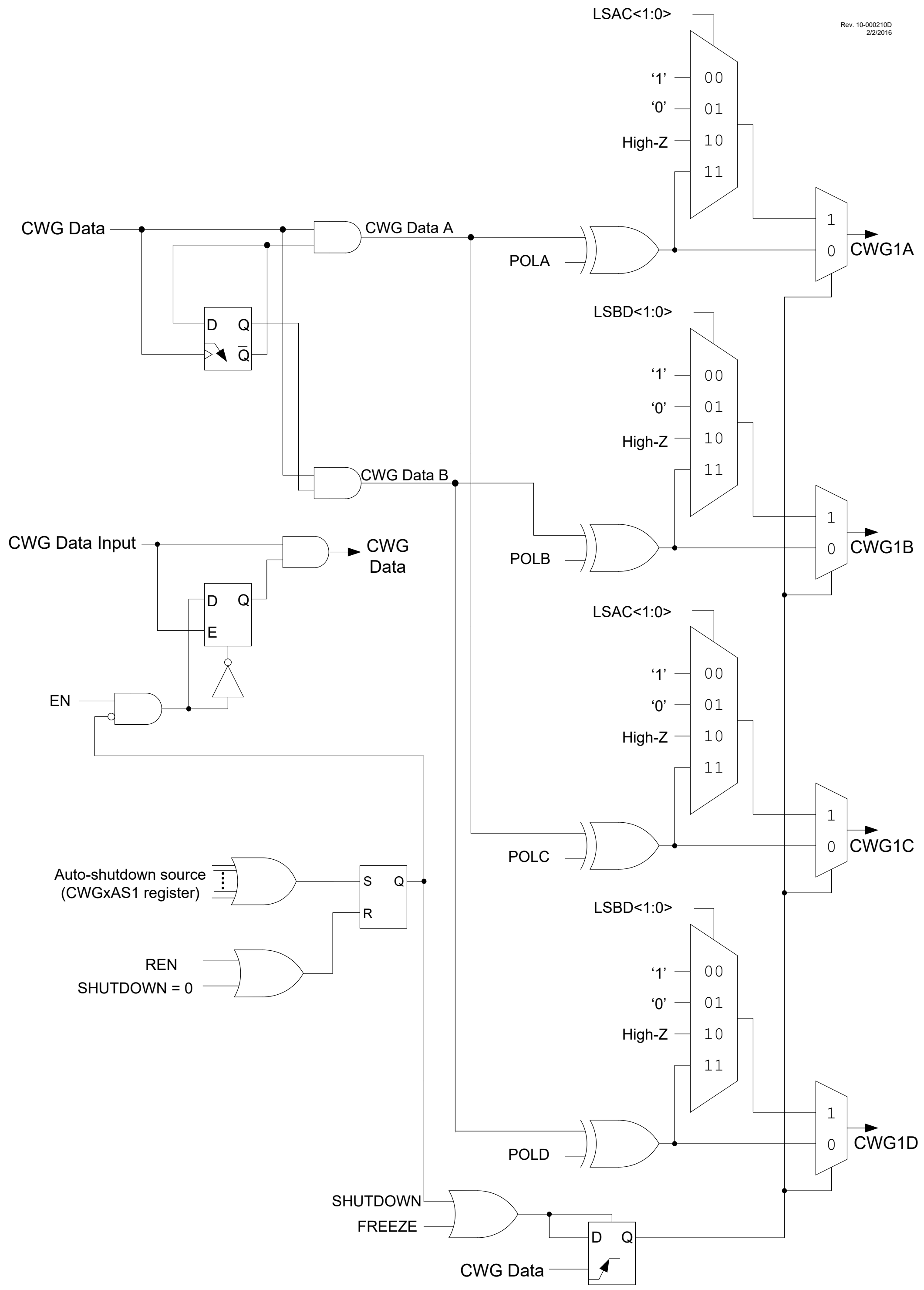
Bits 15:8 – DCH[7:0] PWM Duty Cycle Most Significant bits
These bits are the MSbs of the PWM duty cycle.

Reset States: POR/BOR = xxxxxxxx
All Other Resets = uuuuuuuu

Bits 7:6 – DCL[1:0] PWM Duty Cycle Least Significant bits
These bits are the LSbs of the PWM duty cycle.

Reset States: POR/BOR = xx
All Other Resets = uu

Figure 24-4. Simplified CWG Block Diagram (Push-Pull Mode, MODE<2:0> = `101`)



24.2.3 Full-Bridge Modes

In Forward and Reverse Full-Bridge modes, three outputs drive static values while the fourth is modulated by the input data signal. The mode selection may be toggled between forward and reverse by toggling the MODE<0> bit of the CWGxCON0 while keeping MODE<2:1> static, without disabling the

24.2.4 Steering Modes

In both Synchronous and Asynchronous Steering modes, the modulated input signal can be steered to any combination of four CWG outputs. A fixed-value will be presented on all the outputs not used for the PWM output. Each output has independent polarity, steering, and shutdown options. Dead-band control is not used in either steering mode.

24.14 Register Summary - CWG Control

Address	Name	Bit Pos.								
0x0F3B	CWG1CLK	7:0								CS
0x0F3C	CWG1ISM	7:0						ISM[2:0]		
0x0F3D	CWG1DBR	7:0			DBR[5:0]					
0x0F3E	CWG1DBF	7:0			DBF[5:0]					
0x0F3F	CWG1CON0	7:0	EN	LD				MODE[2:0]		
0x0F40	CWG1CON1	7:0			IN		POLD	POLC	POLB	POLA
0x0F41	CWG1AS0	7:0	SHUTDOWN	REN	LSBD[1:0]		LSAC[1:0]			
0x0F42	CWG1AS1	7:0			AS5E	AS4E	AS3E	AS2E	AS1E	AS0E
0x0F43	CWG1STR	7:0	OVRD	OVRC	OVRB	OVRA	STRD	STRC	STRB	STRA

24.15 Register Definitions: CWG Control

Long bit name prefixes for the CWG peripherals are shown in the table below. Refer to the *"Long Bit Names Section"* for more information.

Table 24-3. CWG Bit Name Prefixes

Peripheral	Bit Name Prefix
CWG1	CWG1

Related Links

1.4.2.2 Long Bit Names

26.6.1 I²C Master Mode Operation

The master device generates all of the serial clock pulses and the Start and Stop conditions. A transfer is ended with a Stop condition or with a Repeated Start condition. Since the Repeated Start condition is also the beginning of the next serial transfer, the I²C bus will not be released.

In Master Transmitter mode, serial data is output through SDA, while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the 26.9.1.6 R/W bit. In this case, the 26.9.1.6 R/W bit will be logic '0'. Serial data is transmitted eight bits at a time. After each byte is transmitted, an Acknowledge bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

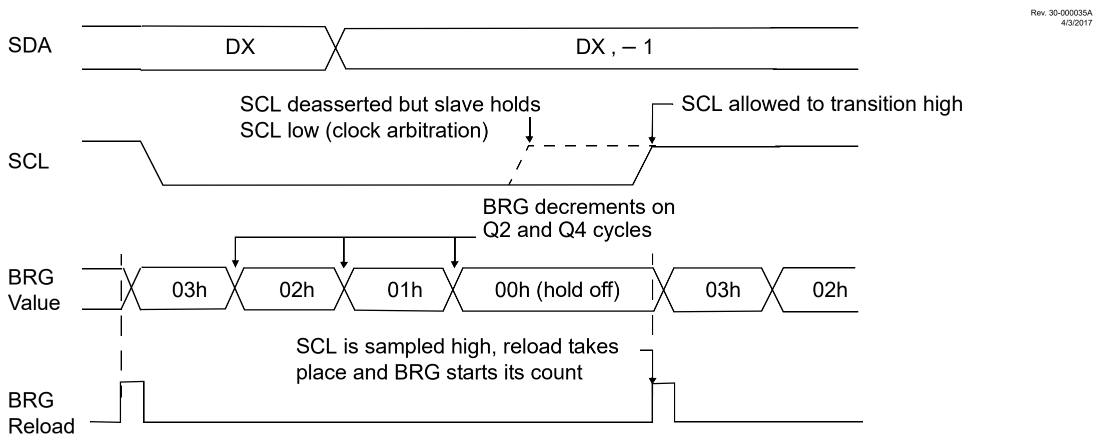
In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/W bit. In this case, the R/W bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address followed by a '1' to indicate the receive bit. Serial data is received via SDA, while SCL outputs the serial clock. Serial data is received eight bits at a time. After each byte is received, an Acknowledge bit is transmitted. Start and Stop conditions indicate the beginning and end of transmission.

A Baud Rate Generator is used to set the clock frequency output on SCL. See 26.7 Baud Rate Generator for more detail.

26.6.2 Clock Arbitration

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, releases the SCL pin (SCL allowed to float high). When the SCL pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCL pin is actually sampled high. When the SCL pin is sampled high, the Baud Rate Generator is reloaded with the contents of 26.9.6 SSPxADD and begins counting. This ensures that the SCL high time will always be at least one BRG rollover count in the event that the clock is held low by an external device as shown in the following figure.

Figure 26-25. Baud Rate Generator Timing with Clock Arbitration


Rev. 30-000035A
4/3/2017
SDA
DX
DX , – 1
SCL deasserted but slave holds SCL low (clock arbitration)
SCL allowed to transition high
SCL
BRG decrements on Q2 and Q4 cycles
BRG Value
03h
02h
01h
00h (hold off)
03h
02h
SCL is sampled high, reload takes place and BRG starts its count
BRG Reload

26.6.3 WCOL Status Flag

If the user writes the SSPxBUF when a Start, Restart, Stop, Receive or Transmit sequence is in progress, the 26.9.2.1 WCOL bit is set and the contents of the buffer are unchanged (the write does not occur). Any time the WCOL bit is set it indicates that an action on SSPxBUF was attempted while the module was not idle.

Value	Condition	Description
1	27.6.2.4 SYNC= 0	High speed, if BRG16 = 1, baud rate is baudclk/4; else baudclk/16
0	27.6.2.4 SYNC= 0	Low speed
X	27.6.2.4 SYNC= 1	Don't care

Bit 1 – TRMT Transmit Shift Register (TSR) Status bit

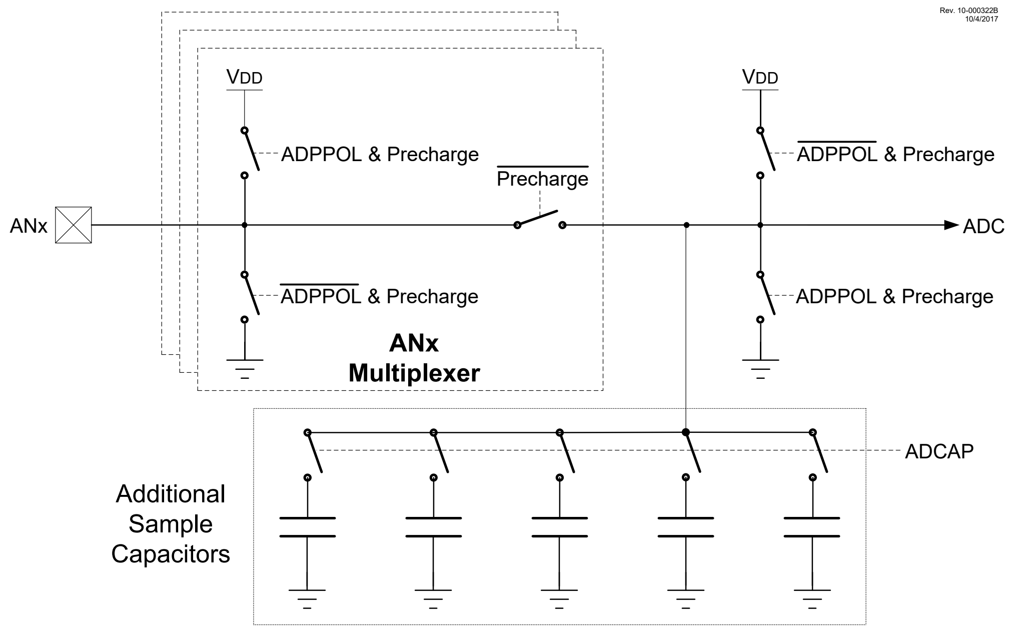
Value	Description
1	TSR is empty
0	TSR is not empty

Bit 0 – TX9D Ninth bit of Transmit Data
Can be address/data bit or a parity bit.

Note:

1. 27.6.1.3 SREN and 27.6.1.4 CREN bits override TXEN in Sync mode.

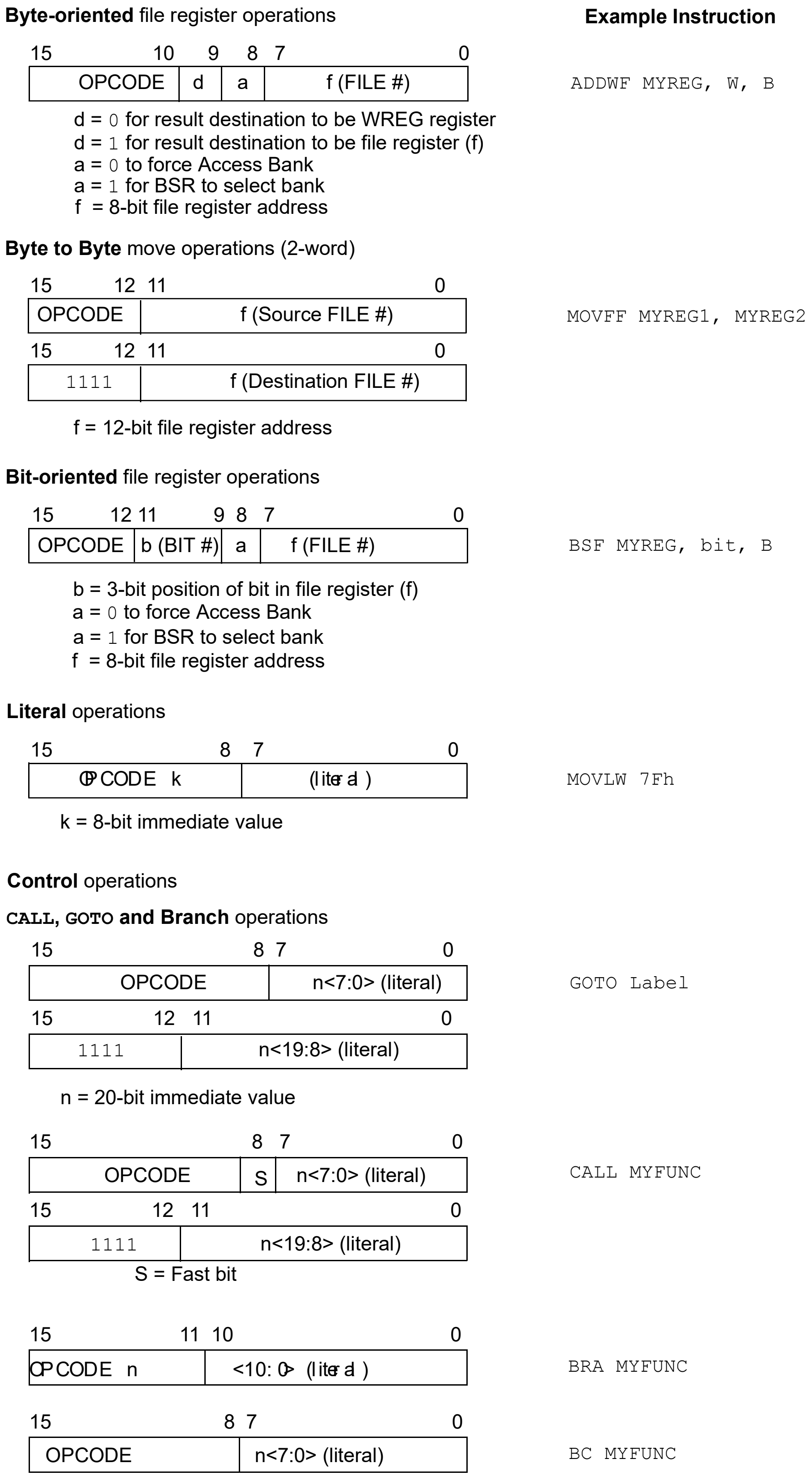
Figure 31-6. Hardware Capacitive Voltage Divider Block Diagram



31.4.1 CVD Operation

A CVD operation begins with the ADC's internal sample and hold capacitor (C_{HOLD}) being disconnected from the path which connects it to the external capacitive sensor node. While disconnected, C_{HOLD} is precharged to V_{DD} or V_{SS} the sensor node is also charged to V_{SS} or V_{DD}, respectively to the level opposite that of C_{HOLD}. When the precharge phase is complete, the V_{DD}/V_{SS} bias paths for the two nodes are shut off and the paths between C_{HOLD} and the external sensor node is reconnected, at which time the acquisition phase of the CVD operation begins. During acquisition, a capacitive voltage divider is formed between the precharged C_{HOLD} and sensor nodes, which results in a final voltage level setting on C_{HOLD} which is determined by the capacitances and precharge levels of the two nodes. After acquisition, the ADC converts the voltage level on C_{HOLD}. This process is then repeated with the selected precharge levels inverted for both the C_{HOLD} and the sensor nodes. The waveform for two CVD measurements, which is known as differential CVD measurement, is shown in the following figure.

Figure 36-1. General Format for Instructions



d = 0 for result destination to be WREG register
d = 1 for result destination to be file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

f = 12-bit file register address

b = 3-bit position of bit in file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

k = 8-bit immediate value

n = 20-bit immediate value

S = Fast bit

MOVWF	Move W to f			
Syntax:	MOVWF f {,a}			
Operands:	0 ≤ f ≤ 255 a ∈ [0,1]			
Operation:	(W) → f			
Status Affected:	None			
Encoding:	0110	111a	ffff	ffff
Description:	Move data from W to register 'f'. Location 'f' can be anywhere in the 256-byte bank. If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank. If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See 36.2.3 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:	`MOVWF`	`REG, 0`

Before Instruction
W = 4Fh
REG = FFh
After Instruction
W = 4Fh
REG = 4Fh		

MULLW	Multiply literal with W
Syntax:	MULLW k
Operands:	0 ≤ k ≤ 255
Operation:	(W) x k → PRODH:PRODL

MULLW	Multiply literal with W			
Status Affected:	None			
Encoding:	0000	1101	kkkk	kkkk
Description:	An unsigned multiplication is carried out between the contents of W and the 8-bit literal ‘k’. The 16-bit result is placed in the PRODH:PRODL register pair. PRODH contains the high byte. W is unchanged. None of the Status flags are affected. Note that neither overflow nor carry is possible in this operation. A zero result is possible but not detected.			
Words:	1			
Cycles:	1			

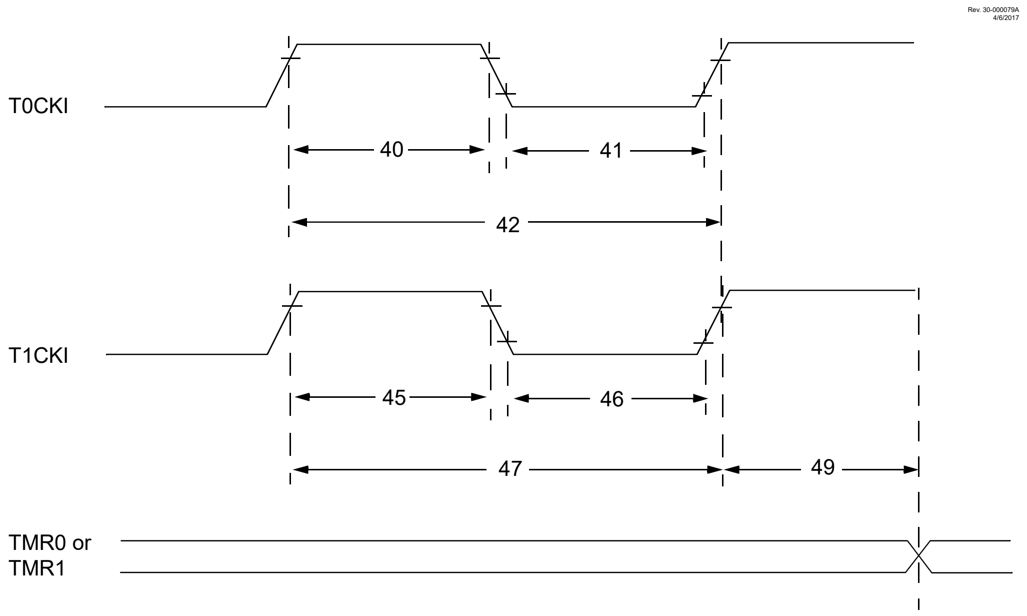
Q Cycle Activity:			
Q1	Q2	Q3	Q4
Decode	Read literal ‘k’	Process Data	Write registers PRODH: PRODL

Example:	`MULLW   0C4h`

Before Instruction
W = E2h
PRODH = ?
PRODL = ?
After Instruction
W = E2h
PRODH = ADh
PRODL = 08h	

MULWF	Multiply W with f
Syntax:	MULWF f {,a}
Operands:	$0 \leq f \leq 255$ $a \in [0,1]$
Operation:	(W) x (f) → PRODH:PRODL
Status Affected:	None

Figure 38-11. Timer0 and Timing1 External Clock Timings



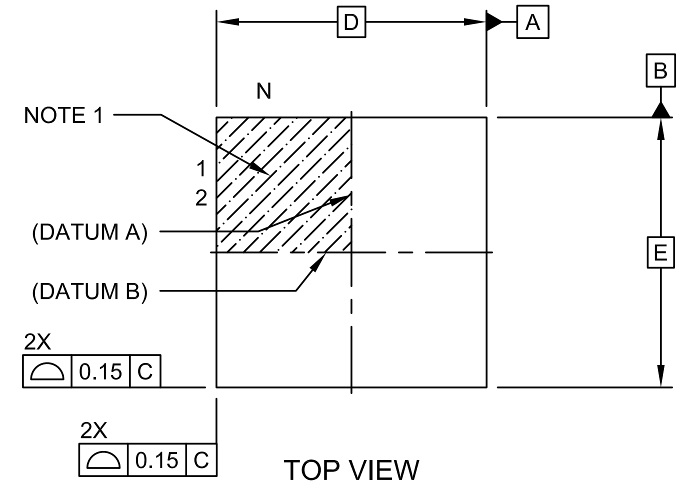
38.4.14 Capture/Compare/PWM Requirements (CCP)

Table 38-20.

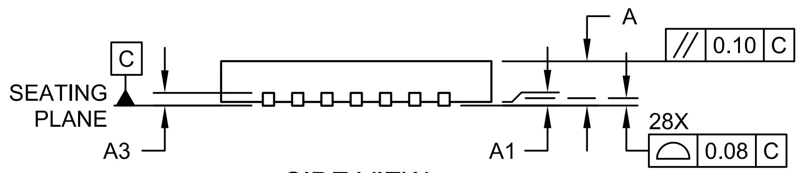
Standard Operating Conditions (unless otherwise stated) Operating Temperature: -40°C≤T_A≤+125°C								
Param No.	**Sym.**	**Characteristic**		**Min.**	**Typ. †**	**Max.**	**Units**	**Conditions**
CC01*	$T_{CC}L$	CCPx Input Low Time	No Prescaler	$0.5T_{CY}$+20	—	—	ns	
			With Prescaler	20	—	—	ns	
CC02*	$T_{CC}H$	CCPx Input High Time	No Prescaler	$0.5T_{CY}$+20	—	—	ns	
			With Prescaler	20	—	—	ns	
CC03*	$T_{CC}P$	CCPx Input Period		$(3T_{CY}$ +40)/N	—	—	ns	N = Prescale value
* - These parameters are characterized but not tested. † Data in "Typ" column is at 3.0V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.								

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

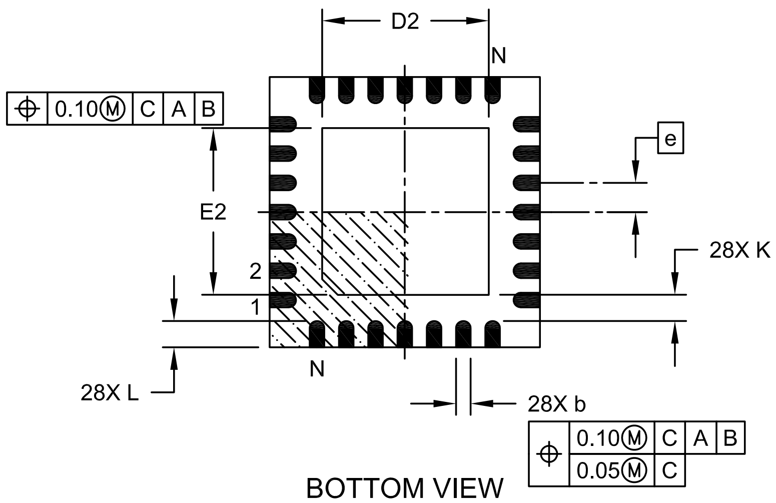
Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW



SIDE VIEW



BOTTOM VIEW

Microchip Technology Drawing C04-105C Sheet 1 of 2