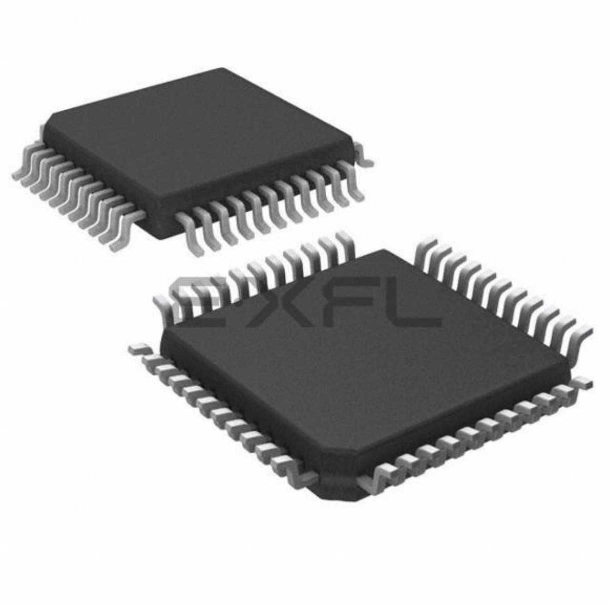


Welcome to E-XFL.COM

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	16MHz
Connectivity	UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	33
Program Memory Size	16KB (8K x 16)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	454 x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 6V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	44-QFP
Supplier Device Package	44-MQFP (10x10)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic17c44-16i-pq

NOTES:

5.4 Interrupt Operation

Global Interrupt Disable bit, GLINTD (CPUSTA<4>), enables all unmasked interrupts (if clear) or disables all interrupts (if set). Individual interrupts can be disabled through their corresponding enable bits in the INTSTA register. Peripheral interrupts need either the global peripheral enable PEIE bit disabled, or the specific peripheral enable bit disabled. Disabling the peripherals via the global peripheral enable bit, disables all peripheral interrupts. GLINTD is set on reset (interrupts disabled).

The `RETFIE` instruction allows returning from interrupt and re-enable interrupts at the same time.

When an interrupt is responded to, the GLINTD bit is automatically set to disable any further interrupt, the return address is pushed onto the stack and the PC is loaded with interrupt vector. There are four interrupt vectors to reduce interrupt latency.

The peripheral interrupt vector has multiple interrupt sources. Once in the peripheral interrupt service routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The peripheral interrupt flag bit(s) must be cleared in software before re-enabling interrupts to avoid continuous interrupts.

The PIC17C4X devices have four interrupt vectors. These vectors and their hardware priority are shown in Table 5-1. If two enabled interrupts occur “at the same time”, the interrupt of the highest priority will be serviced first. This means that the vector address of that interrupt will be loaded into the program counter (PC).

TABLE 5-1: INTERRUPT VECTORS/ PRIORITIES

Address	Vector	Priority
0008h	External Interrupt on RA0/ INT pin (INTF)	1 (Highest)
0010h	TMR0 overflow interrupt (T0IF)	2
0018h	External Interrupt on T0CKI (T0CKIF)	3
0020h	Peripherals (PEIF)	4 (Lowest)

Note 1: Individual interrupt flag bits are set regardless of the status of their corresponding mask bit or the GLINTD bit.

Note 2: When disabling any of the INTSTA enable bits, the GLINTD bit should be set (disabled).

Note 3: For the PIC17C42 only:
If an interrupt occurs while the Global Interrupt Disable (GLINTD) bit is being set, the GLINTD bit may unintentionally be re-enabled by the user’s Interrupt Service Routine (the `RETFIE` instruction). The events that would cause this to occur are:

1. An interrupt occurs simultaneously with an instruction that sets the GLINTD bit.
2. The program branches to the Interrupt vector and executes the Interrupt Service Routine.
3. The Interrupt Service Routine completes with the execution of the `RETFIE` instruction. This causes the GLINTD bit to be cleared (enables interrupts), and the program returns to the instruction after the one which was meant to disable interrupts.

The method to ensure that interrupts are globally disabled is:

1. Ensure that the GLINTD bit was set by the instruction, as shown in the following code:

```
LOOP   BSF    CPUSTA, GLINTD ; Disable Global
                             ; Interrupt
       BTFSS  CPUSTA, GLINTD ; Global Interrupt
                             ; Disabled?
       GOTO   LOOP           ; NO, try again
                             ; YES, continue
                             ; with program
                             ; low
```

6.4.1 INDIRECT ADDRESSING REGISTERS

The PIC17C4X has four registers for indirect addressing. These registers are:

- INDF0 and FSR0
- INDF1 and FSR1

Registers INDF0 and INDF1 are not physically implemented. Reading or writing to these registers activates indirect addressing, with the value in the corresponding FSR register being the address of the data. The FSR is an 8-bit register and allows addressing anywhere in the 256-byte data memory address range. For banked memory, the bank of memory accessed is specified by the value in the BSR.

If file INDF0 (or INDF1) itself is read indirectly via an FSR, all '0's are read (Zero bit is set). Similarly, if INDF0 (or INDF1) is written to indirectly, the operation will be equivalent to a NOP, and the status bits are not affected.

6.4.2 INDIRECT ADDRESSING OPERATION

The indirect addressing capability has been enhanced over that of the PIC16CXX family. There are two control bits associated with each FSR register. These two bits configure the FSR register to:

- Auto-decrement the value (address) in the FSR after an indirect access
- Auto-increment the value (address) in the FSR after an indirect access
- No change to the value (address) in the FSR after an indirect access

These control bits are located in the ALUSTA register. The FSR1 register is controlled by the FS3:FS2 bits and FSR0 is controlled by the FS1:FS0 bits.

When using the auto-increment or auto-decrement features, the effect on the FSR is not reflected in the ALUSTA register. For example, if the indirect address causes the FSR to equal '0', the Z bit will not be set.

If the FSR register contains a value of 0h, an indirect read will read 0h (Zero bit is set) while an indirect write will be equivalent to a NOP (status bits are not affected).

Indirect addressing allows single cycle data transfers within the entire data space. This is possible with the use of the `MOVPF` and `MOVFP` instructions, where either 'p' or 'f' is specified as INDF0 (or INDF1).

If the source or destination of the indirect address is in banked memory, the location accessed will be determined by the value in the BSR.

A simple program to clear RAM from 20h - FFh is shown in Example 6-1.

EXAMPLE 6-1: INDIRECT ADDRESSING

```
    MOVLW    0x20          ;
    MOVWF    FSR0          ; FSR0 = 20h
    BCF      ALUSTA, FS1   ; Increment FSR
    BSF      ALUSTA, FS0   ; after access
    BCF      ALUSTA, C     ; C = 0
    MOVLW    END_RAM + 1   ;
LP  CLRF     INDF0         ; Addr(FSR) = 0
    CPFSEQ   FSR0          ; FSR0 = END_RAM+1?
    GOTO     LP            ; NO, clear next
    :                      ; YES, All RAM is
    :                      ; cleared
```

6.5 Table Pointer (TBLPTRL and TBLPTRH)

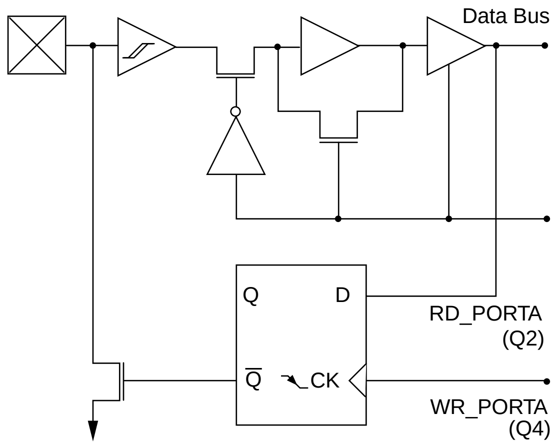
File registers TBLPTRL and TBLPTRH form a 16-bit pointer to address the 64K program memory space. The table pointer is used by instructions `TABLWT` and `TABLRD`.

The `TABLRD` and the `TABLWT` instructions allow transfer of data between program and data space. The table pointer serves as the 16-bit address of the data word within the program memory. For a more complete description of these registers and the operation of Table Reads and Table Writes, see Section 7.0.

6.6 Table Latch (TBLATH, TBLATL)

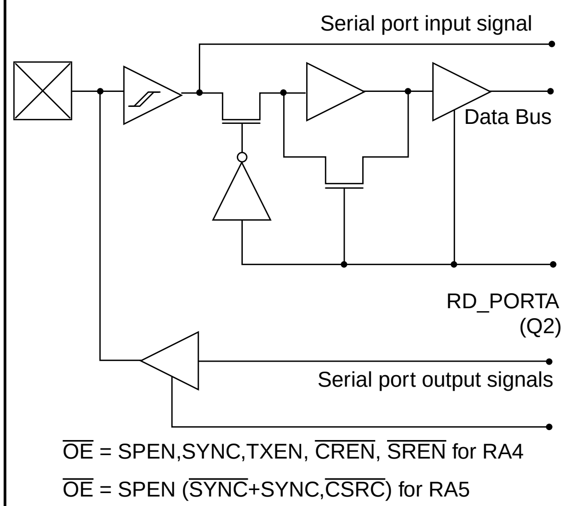
The table latch (TBLAT) is a 16-bit register, with TBLATH and TBLATL referring to the high and low bytes of the register. It is not mapped into data or program memory. The table latch is used as a temporary holding latch during data transfer between program and data memory (see descriptions of instructions `TABLRD`, `TABLWT`, `TLRD` and `TLWT`). For a more complete description of these registers and the operation of Table Reads and Table Writes, see Section 7.0.

FIGURE 9-2: RA2 AND RA3 BLOCK DIAGRAM



Note: I/O pins have protection diodes to VSS.

FIGURE 9-3: RA4 AND RA5 BLOCK DIAGRAM



Note: I/O pins have protection diodes to VDD and VSS.

TABLE 9-1: PORTA FUNCTIONS

Name	Bit0	Buffer Type	Function
RA0/INT	bit0	ST	Input or external interrupt input.
RA1/T0CKI	bit1	ST	Input or clock input to the TMR0 timer/counter, and/or an external interrupt input.
RA2	bit2	ST	Input/Output. Output is open drain type.
RA3	bit3	ST	Input/Output. Output is open drain type.
RA4/RX/DT	bit4	ST	Input or USART Asynchronous Receive or USART Synchronous Data.
RA5/TX/CK	bit5	ST	Input or USART Asynchronous Transmit or USART Synchronous Clock.
$\overline{\text{RBPU}}$	bit7	—	Control bit for PORTB weak pull-ups.

Legend: ST = Schmitt Trigger input.

TABLE 9-2: REGISTERS/BITS ASSOCIATED WITH PORTA

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other resets (Note1)
10h, Bank 0	PORTA	$\overline{\text{RBPU}}$	—	RA5	RA4	RA3	RA2	RA1/T0CKI	RA0/INT	0-xx xxxx	0-uu uuuu
05h, Unbanked	T0STA	INTEDG	T0SE	T0CS	PS3	PS2	PS1	PS0	—	0000 000-	0000 000-
13h, Bank 0	RCSTA	SPEN	RC9	SREN	CREN	—	FERR	OERR	RC9D	0000 -00x	0000 -00u
15h, Bank 0	TXSTA	CSRC	TX9	TXEN	SYNC	—	—	TRMT	TX9D	0000 --1x	0000 --1u

Legend: x = unknown, u = unchanged, - = unimplemented reads as '0'. Shaded cells are not used by PORTA.
Note 1: Other (non power-up) resets include: external reset through $\overline{\text{MCLR}}$ and the Watchdog Timer Reset.

12.2.2 DUAL CAPTURE REGISTER MODE

This mode is selected by setting CA1/$\overline{\text{PR3}}$. A block diagram is shown in Figure 12-8. In this mode, TMR3 runs without a period register and increments from 0000h to FFFFh and rolls over to 0000h. The TMR3 interrupt Flag (TMR3IF) is set on this roll over. The TMR3IF bit must be cleared in software.

Registers PR3H/CA1H and PR3L/CA1L make a 16-bit capture register (Capture1). It captures events on pin RB0/CAP1. Capture mode is configured by the CA1ED1 and CA1ED0 bits. Capture1 Interrupt Flag bit (CA1IF) is set on the capture event. The corresponding interrupt mask bit is CA1IE. The Capture1 Overflow Status bit is CA1OVF.

The Capture2 overflow status flag bit is double buffered. The master bit is set if one captured word is already residing in the Capture2 register and another “event” has occurred on the RB1/CA2 pin. The new event will not transfer the TMR3 value to the capture register which protects the previous unread capture value. When the user reads both the high and the low bytes (in any order) of the Capture2 register, the master overflow bit is transferred to the slave overflow bit (CA2OVF) and then the master bit is reset. The user can then read TCON2 to determine the value of CA2OVF.

The operation of the Capture1 feature is identical to Capture2 (as described in Section 12.2.1).

FIGURE 12-8: TIMER3 WITH TWO CAPTURE REGISTERS BLOCK DIAGRAM

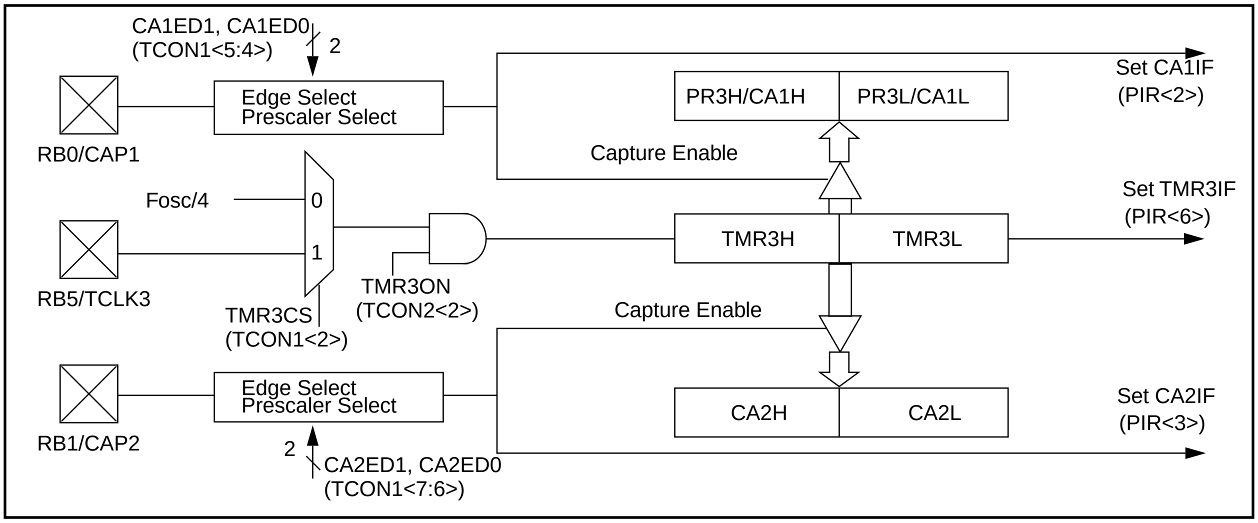


TABLE 12-5: REGISTERS ASSOCIATED WITH CAPTURE

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other resets (Note1)
16h, Bank 3	TCON1	CA2ED1	CA2ED0	CA1ED1	CA1ED0	T16	TMR3CS	TMR2CS	TMR1CS	0000 0000	0000 0000
17h, Bank 3	TCON2	CA2OVF	CA1OVF	PWM2ON	PWM1ON	CA1/$\overline{\text{PR3}}$	TMR3ON	TMR2ON	TMR1ON	0000 0000	0000 0000
12h, Bank 2	TMR3L	TMR3 register; low byte								xxxx xxxx	uuuu uuuu
13h, Bank 2	TMR3H	TMR3 register; high byte								xxxx xxxx	uuuu uuuu
16h, Bank 1	PIR	RBIF	TMR3IF	TMR2IF	TMR1IF	CA2IF	CA1IF	TXIF	RCIF	0000 0010	0000 0010
17h, Bank 1	PIE	RBIE	TMR3IE	TMR2IE	TMR1IE	CA2IE	CA1IE	TXIE	RCIE	0000 0000	0000 0000
07h, Unbanked	INTSTA	PEIF	T0CKIF	T0IF	INTF	PEIE	T0CKIE	T0IE	INTE	0000 0000	0000 0000
06h, Unbanked	CPUSTA	—	—	STKAV	GLINTD	$\overline{\text{TO}}$	$\overline{\text{PD}}$	—	—	--11 11--	--11 qq--
16h, Bank 2	PR3L/CA1L	Timer3 period register, low byte/capture1 register, low byte								xxxx xxxx	uuuu uuuu
17h, Bank 2	PR3H/CA1H	Timer3 period register, high byte/capture1 register, high byte								xxxx xxxx	uuuu uuuu
14h, Bank 3	CA2L	Capture2 low byte								xxxx xxxx	uuuu uuuu
15h, Bank 3	CA2H	Capture2 high byte								xxxx xxxx	uuuu uuuu

Legend: x = unknown, u = unchanged, - = unimplemented read as '0', q - value depends on condition, shaded cells are not used by Capture.

Note 1: Other (non power-up) resets include: external reset through $\overline{\text{MCLR}}$ and WDT Timer Reset.

13.3 USART Synchronous Master Mode

In Master Synchronous mode, the data is transmitted in a half-duplex manner; i.e. transmission and reception do not occur at the same time: when transmitting data, the reception is inhibited and vice versa. The synchronous mode is entered by setting the SYNC (TXSTA<4>) bit. In addition, the SPEN (RCSTA<7>) bit is set in order to configure the RA5 and RA4 I/O ports to CK (clock) and DT (data) lines respectively. The Master mode indicates that the processor transmits the master clock on the CK line. The Master mode is entered by setting the CSRC (TXSTA<7>) bit.

13.3.1 USART SYNCHRONOUS MASTER TRANSMISSION

The USART transmitter block diagram is shown in Figure 13-3. The heart of the transmitter is the transmit (serial) shift register (TSR). The shift register obtains its data from the read/write transmit buffer TXREG. TXREG is loaded with data in software. The TSR is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from TXREG (if available). Once TXREG transfers the data to the TSR (occurs in one TCY at the end of the current BRG cycle), TXREG is empty and the TXIF (PIR<1>) bit is set. This interrupt can be enabled/disabled by setting/clearing the TXIE bit (PIE<1>). TXIF will be set regardless of the state of bit TXIE and cannot be cleared in software. It will reset only when new data is loaded into TXREG. While TXIF indicates the status of TXREG, TRMT (TXSTA<1>) shows the status of the TSR. TRMT is a read only bit which is set when the TSR is empty. No interrupt logic is tied to this bit, so the user has to poll this bit in order to determine if the TSR is empty. The TSR is not mapped in data memory, so it is not available to the user.

Transmission is enabled by setting the TXEN (TXSTA<5>) bit. The actual transmission will not occur until TXREG has been loaded with data. The first data bit will be shifted out on the next available rising edge of the clock on the RA5/TX/CK pin. Data out is stable around the falling edge of the synchronous clock (Figure 13-10). The transmission can also be started by first loading TXREG and then setting TXEN. This is advantageous when slow baud rates are selected, since BRG is kept in RESET when the TXEN, CREN, and SREN bits are clear. Setting the TXEN bit will start the BRG, creating a shift clock immediately. Normally when transmission is first started, the TSR is empty, so a transfer to TXREG will result in an immediate transfer to the TSR, resulting in an empty TXREG. Back-to-back transfers are possible.

Clearing TXEN during a transmission will cause the transmission to be aborted and will reset the transmitter. The RA4/RX/DT and RA5/TX/CK pins will revert to hi-impedance. If either CREN or SREN are set during a transmission, the transmission is aborted and the RA4/RX/DT pin reverts to a hi-impedance state (for a reception). The RA5/TX/CK pin will remain an output if the CSRC bit is set (internal clock). The transmitter logic is not reset, although it is disconnected from the pins. In order to reset the transmitter, the user has to clear the TXEN bit. If the SREN bit is set (to interrupt an ongoing transmission and receive a single word), then after the single word is received, SREN will be cleared and the serial port will revert back to transmitting, since the TXEN bit is still set. The DT line will immediately switch from hi-impedance receive mode to transmit and start driving. To avoid this, TXEN should be cleared.

In order to select 9-bit transmission, the TX9 (TXSTA<6>) bit should be set and the ninth bit should be written to TX9D (TXSTA<0>). The ninth bit must be written before writing the 8-bit data to TXREG. This is because a data write to TXREG can result in an immediate transfer of the data to the TSR (if the TSR is empty). If the TSR was empty and TXREG was written before writing the “new” TX9D, the “present” value of TX9D is loaded.

Steps to follow when setting up a Synchronous Master Transmission:

1. Initialize the SPBRG register for the appropriate baud rate (see Baud Rate Generator Section for details).
2. Enable the synchronous master serial port by setting the SYNC, SPEN, and CSRC bits.
3. Ensure that the CREN and SREN bits are clear (these bits override transmission when set).
4. If interrupts are desired, then set the TXIE bit (the GLINTD bit must be clear and the PEIE bit must be set).
5. If 9-bit transmission is desired, then set the TX9 bit.
6. Start transmission by loading data to the TXREG register.
7. If 9-bit transmission is selected, the ninth bit should be loaded in TX9D.
8. Enable the transmission by setting TXEN.

Writing the transmit data to the TXREG, then enabling the transmit (setting TXEN) allows transmission to start sooner then doing these two events in the reverse order.

Note:	To terminate a transmission, either clear the SPEN bit, or the TXEN bit. This will reset the transmit logic, so that it will be in the proper state when transmit is re-enabled.

TABLE 13-8: REGISTERS ASSOCIATED WITH SYNCHRONOUS MASTER RECEPTION

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-on Reset	Value on all other resets (Note1)
16h, Bank 1	PIR	RBIF	TMR3IF	TMR2IF	TMR1IF	CA2IF	CA1IF	TXIF	RCIF	0000 0010	0000 0010
13h, Bank 0	RCSTA	SPEN	RX9	SREN	CREN	—	FERR	OERR	RX9D	0000 -00x	0000 -00u
14h, Bank 0	RCREG	RX7	RX6	RX5	RX4	RX3	RX2	RX1	RX0	xxxx xxxx	uuuu uuuu
17h, Bank 1	PIE	RBIE	TMR3IE	TMR2IE	TMR1IE	CA2IE	CA1IE	TXIE	RCIE	0000 0000	0000 0000
15h, Bank 0	TXSTA	CSRC	TX9	TXEN	SYNC	—	—	TRMT	TX9D	0000 --1x	0000 --1u
17h, Bank 0	SPBRG	Baud rate generator register								xxxx xxxx	uuuu uuuu

Legend: x = unknown, u = unchanged, - = unimplemented read as a '0', shaded cells are not used for synchronous master reception.

Note 1: Other (non power-up) resets include: external reset through $\overline{MCLR}$ and Watchdog Timer Reset.

14.4.2 MINIMIZING CURRENT CONSUMPTION

To minimize current consumption, all I/O pins should be either at V_{DD}, or V_{SS}, with no external circuitry drawing current from the I/O pin. I/O pins that are hi-impedance inputs should be pulled high or low externally to avoid switching currents caused by floating inputs. The T0CKI input should be at V_{DD} or V_{SS}. The contributions from on-chip pull-ups on PORTB should also be considered, and disabled when possible.

14.5 Code Protection

The code in the program memory can be protected by selecting the microcontroller in code protected mode (PM2:PM0 = '000').

Note:	PM2 does not exist on the PIC17C42. To select code protected microcontroller mode, PM1:PM0 = '00'.

In this mode, instructions that are in the on-chip program memory space, can continue to read or write the program memory. An instruction that is executed outside of the internal program memory range will be inhibited from writing to or reading from program memory.

Note:	Microchip does not recommend code protecting windowed devices.

If the code protection bit(s) have not been programmed, the on-chip program memory can be read out for verification purposes.

TABLE 15-2: PIC17CXX INSTRUCTION SET

Mnemonic,	Operands	Description	Cycles	16-bit Opcode MSb			LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS									
ADDWF	**f,d**	ADD WREG to f	1	0000	111d	ffff	ffff	OV,C,DC,Z	
ADDWFC	**f,d**	ADD WREG and Carry bit to f	1	0001	000d	ffff	ffff	OV,C,DC,Z	
ANDWF	**f,d**	AND WREG with f	1	0000	101d	ffff	ffff	Z	
CLRF	**f,s**	Clear f, or Clear f and Clear WREG	1	0010	100s	ffff	ffff	None	3
COMF	**f,d**	Complement f	1	0001	001d	ffff	ffff	Z	
CPFSEQ	**f**	Compare f with WREG, skip if f = WREG	1 (2)	0011	0001	ffff	ffff	None	6,8
CPFSGT	**f**	Compare f with WREG, skip if f > WREG	1 (2)	0011	0010	ffff	ffff	None	2,6,8
CPFSLT	**f**	Compare f with WREG, skip if f < WREG	1 (2)	0011	0000	ffff	ffff	None	2,6,8
DAW	**f,s**	Decimal Adjust WREG Register	1	0010	111s	ffff	ffff	C	3
DECF	**f,d**	Decrement f	1	0000	011d	ffff	ffff	OV,C,DC,Z	
DECFSZ	**f,d**	Decrement f, skip if 0	1 (2)	0001	011d	ffff	ffff	None	6,8
DCFSNZ	**f,d**	Decrement f, skip if not 0	1 (2)	0010	011d	ffff	ffff	None	6,8
INCF	**f,d**	Increment f	1	0001	010d	ffff	ffff	OV,C,DC,Z	
INCFSZ	**f,d**	Increment f, skip if 0	1 (2)	0001	111d	ffff	ffff	None	6,8
INFSNZ	**f,d**	Increment f, skip if not 0	1 (2)	0010	010d	ffff	ffff	None	6,8
IORWF	**f,d**	Inclusive OR WREG with f	1	0000	100d	ffff	ffff	Z	
MOVFP	**f,p**	Move f to p	1	011p	pppp	ffff	ffff	None	
MOVPF	**p,f**	Move p to f	1	010p	pppp	ffff	ffff	Z	
MOVWF	**f**	Move WREG to f	1	0000	0001	ffff	ffff	None	
MULWF	**f**	Multiply WREG with f	1	0011	0100	ffff	ffff	None	9
NEGW	**f,s**	Negate WREG	1	0010	110s	ffff	ffff	OV,C,DC,Z	1,3
NOP	**—**	No Operation	1	0000	0000	0000	0000	None	
RLCF	**f,d**	Rotate left f through Carry	1	0001	101d	ffff	ffff	C	
RLNCF	**f,d**	Rotate left f (no carry)	1	0010	001d	ffff	ffff	None	
RRCF	**f,d**	Rotate right f through Carry	1	0001	100d	ffff	ffff	C	
RRNCF	**f,d**	Rotate right f (no carry)	1	0010	000d	ffff	ffff	None	
SETF	**f,s**	Set f	1	0010	101s	ffff	ffff	None	3
SUBWF	**f,d**	Subtract WREG from f	1	0000	010d	ffff	ffff	OV,C,DC,Z	1
SUBWFB	**f,d**	Subtract WREG from f with Borrow	1	0000	001d	ffff	ffff	OV,C,DC,Z	1
SWAPF	**f,d**	Swap f	1	0001	110d	ffff	ffff	None	
TABLRD	**t,i,f**	Table Read	2 (3)	1010	10ti	ffff	ffff	None	7

Legend: Refer to Table 15-1 for opcode field descriptions.
Note 1: 2’s Complement method.
2: Unsigned arithmetic.
3: If s = '1', only the file is affected: If s = '0', both the WREG register and the file are affected; If only the Working register (WREG) is required to be affected, then f = WREG must be specified.
4: During an LCALL, the contents of PCLATH are loaded into the MSB of the PC and kkkk kkkk is loaded into the LSB of the PC (PCL)
5: Multiple cycle instruction for EPROM programming when table pointer selects internal EPROM. The instruction is terminated by an interrupt event. When writing to external program memory, it is a two-cycle instruction.
6: Two-cycle instruction when condition is true, else single cycle instruction.
7: Two-cycle instruction except for TABLRD to PCL (program counter low byte) in which case it takes 3 cycles.
8: A “skip” means that instruction fetched during execution of current instruction is not executed, instead an NOP is executed.
9: These instructions are not available on the PIC17C42.

ADDLW — ADD Literal to WREG

Syntax: [*label*] ADDLW k

Operands: $0 \le k \le 255$

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

Status Affected: OV, C, DC, Z

Encoding:

`1011`	`0001`	`kkkk`	`kkkk`

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

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Execute	Write to WREG

Example: `ADDLW 0x15`

Before Instruction
WREG = 0x10

After Instruction
WREG = 0x25

ADDWF — ADD WREG to f

Syntax: [*label*] ADDWF f,d

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

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

Status Affected: OV, C, DC, Z

Encoding:

`0000`	`111d`	`ffff`	`ffff`

Description: Add WREG to register 'f'. If 'd' is 0 the result is stored in WREG. If 'd' is 1 the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Execute	Write to destination

Example: `ADDWF REG, 0`

Before Instruction
WREG = 0x17
REG = 0xC2

After Instruction
WREG = 0xD9
REG = 0xC2

INFSNZ	Increment f, skip if not 0
Syntax:	[*label*] INFSNZ f,d
Operands:	$0 \le f \le 255$ $d \in [0,1]$
Operation:	(f) + 1 $\rightarrow$ (dest), skip if not 0
Status Affected:	None
Encoding:	`0010` `010d` `ffff` `ffff`
Description:	The contents of register 'f' are incremented. If 'd' is 0 the result is placed in WREG. If 'd' is 1 the result is placed back in register 'f'. If the result is not 0, the next instruction, which is already fetched, is discarded, and an NOP is executed instead making it a two-cycle instruction.
Words:	1
Cycles:	1(2)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Execute	Write to destination

If skip:

Q1	Q2	Q3	Q4
Forced NOP	NOP	Execute	NOP

Example:

```
HERE    INFSNZ  REG, 1
ZERO
NZERO
```

Before Instruction
REG = REG

After Instruction
REG = REG + 1
If REG = 1;
PC = Address (ZERO)
If REG = 0;
PC = Address (NZERO)

IORLW	Inclusive OR Literal with WREG
Syntax:	[*label*] IORLW k
Operands:	$0 \le k \le 255$
Operation:	(WREG) .OR. (k) $\rightarrow$ (WREG)
Status Affected:	Z
Encoding:	`1011` `0011` `kkkk` `kkkk`
Description:	The contents of WREG are OR'ed with the eight bit literal 'k'. The result is placed in WREG.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Execute	Write to WREG

Example:

```
IORLW   0x35
```

Before Instruction
WREG = 0x9A

After Instruction
WREG = 0xBF

SUBWF — Subtract WREG from f

Syntax:	[*label*] SUBWF f,d
Operands:	$0 \le f \le 255$ $d \in [0,1]$
Operation:	(f) – (W) → (dest)
Status Affected:	OV, C, DC, Z
Encoding:	`0000` `010d` `ffff` `ffff`
Description:	Subtract WREG from register 'f' (2's complement method). If 'd' is 0 the result is stored in WREG. If 'd' is 1 the result is stored back in register 'f'.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Execute	Write to destination

Example 1: `SUBWF REG1, 1`

Before Instruction
```
REG1  =  3
WREG  =  2
C     =  ?
```
After Instruction
```
REG1  =  1
WREG  =  2
C     =  1      ; result is positive
Z     =  0
```

Example 2:

Before Instruction
```
REG1  =  2
WREG  =  2
C     =  ?
```
After Instruction
```
REG1  =  0
WREG  =  2
C     =  1      ; result is zero
Z     =  1
```

Example 3:

Before Instruction
```
REG1  =  1
WREG  =  2
C     =  ?
```
After Instruction
```
REG1  =  FF
WREG  =  2
C     =  0      ; result is negative
Z     =  0
```

SUBWFB — Subtract WREG from f with Borrow

Syntax:	[*label*] SUBWFB f,d
Operands:	$0 \le f \le 255$ $d \in [0,1]$
Operation:	(f) – (W) – $\overline{C}$ → (dest)
Status Affected:	OV, C, DC, Z
Encoding:	`0000` `001d` `ffff` `ffff`
Description:	Subtract WREG and the carry flag (borrow) from register 'f' (2's complement method). If 'd' is 0 the result is stored in WREG. If 'd' is 1 the result is stored back in register 'f'.
Words:	1
Cycles:	1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Execute	Write to destination

Example 1: `SUBWFB REG1, 1`

Before Instruction
```
REG1  =  0x19    (0001 1001)
WREG  =  0x0D    (0000 1101)
C     =  1
```
After Instruction
```
REG1  =  0x0C    (0000 1011)
WREG  =  0x0D    (0000 1101)
C     =  1       ; result is positive
Z     =  0
```

Example2: `SUBWFB REG1,0`

Before Instruction
```
REG1  =  0x1B    (0001 1011)
WREG  =  0x1A    (0001 1010)
C     =  0
```
After Instruction
```
REG1  =  0x1B    (0001 1011)
WREG  =  0x00
C     =  1       ; result is zero
Z     =  1
```

Example3: `SUBWFB REG1,1`

Before Instruction
```
REG1  =  0x03    (0000 0011)
WREG  =  0x0E    (0000 1101)
C     =  1
```
After Instruction
```
REG1  =  0xF5    (1111 0100) [2's comp]
WREG  =  0x0E    (0000 1101)
C     =  0       ; result is negative
Z     =  0
```

Applicable Devices	42	R42	42A	43	R43	44

DC CHARACTERISTICS			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40˚C ≤ TA ≤ +40˚C Operating voltage VDD range as described in Section 17.1				
Parameter No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
		Internal Program Memory Programming Specs (Note 4)					
D110	VPP	Voltage on $\overline{MCLR}$/VPP pin	12.75	–	13.25	V	Note 5
D111	VDDP	Supply voltage during programming	4.75	5.0	5.25	V	
D112	IPP	Current into $\overline{MCLR}$/VPP pin	–	25 ‡	50 ‡	mA	
D113	IDDP	Supply current during programming	–	–	30 ‡	mA	
D114	TPROG	Programming pulse width	10	100	1000	µs	Terminated via internal/external interrupt or a reset

* These parameters are characterized but not tested.

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

‡ These parameters are for design guidance only and are not tested, nor characterized.

Note 1: In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC17CXX devices be driven with external clock in RC mode.

2: The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as coming out of the pin.

4: These specifications are for the programming of the on-chip program memory EPROM through the use of the table write instructions. The complete programming specifications can be found in: PIC17CXX Programming Specifications (Literature number DS30139).

5: The $\overline{MCLR}$/VPP pin may be kept in this range at times other than programming, but is not recommended.

6: For TTL buffers, the better of the two specifications may be used.

Note: When using the Table Write for internal programming, the device temperature must be less than 40˚C.

Applicable Devices	42	R42	42A	43	R43	44

17.4 Timing Diagrams and Specifications

FIGURE 17-2: EXTERNAL CLOCK TIMING

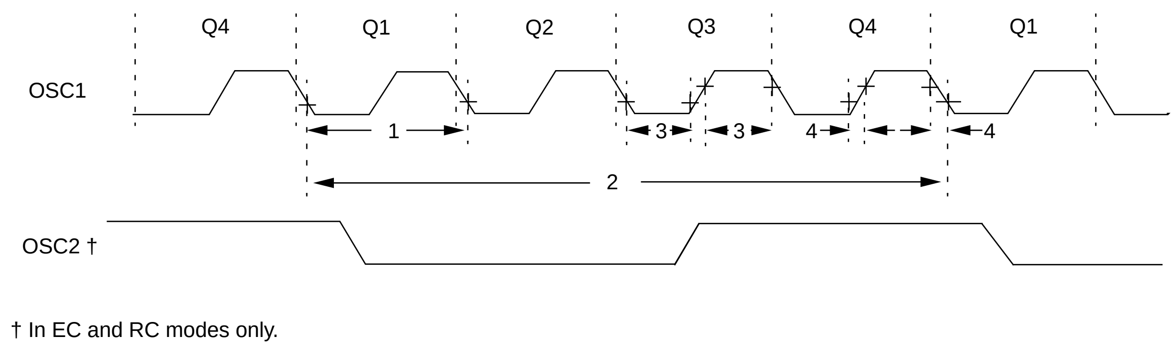


TABLE 17-2: EXTERNAL CLOCK TIMING REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
	Fosc	**External CLKIN Frequency (Note 1)**	DC DC	— —	16 25	MHz MHz	EC osc mode - PIC17C42-16 - PIC17C42-25
		Oscillator Frequency (Note 1)	DC 1 1 DC	— — — —	4 16 25 2	MHz MHz MHz MHz	RC osc mode XT osc mode - PIC17C42-16 - PIC17C42-25 LF osc mode
1	Tosc	**External CLKIN Period (Note 1)**	62.5 40	— —	— —	ns ns	EC osc mode - PIC17C42-16 - PIC17C42-25
		Oscillator Period (Note 1)	250 62.5 40 500	— — — —	— 1,000 1,000 —	ns ns ns ns	RC osc mode XT osc mode - PIC17C42-16 - PIC17C42-25 LF osc mode
2	TCY	**Instruction Cycle Time (Note 1)**	160	4/Fosc	DC	ns	
3	TosL, TosH	**Clock in (OSC1) High or Low Time**	10 ‡	—	—	ns	EC oscillator
4	TosR, TosF	**Clock in (OSC1) Rise or Fall Time**	—	—	5 ‡	ns	EC oscillator

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

‡ These parameters are for design guidance only and are not tested, nor characterized.

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time-base period. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at “min.” values with an external clock applied to the OSC1 pin.
When an external clock input is used, the “Max.” cycle time limit is “DC” (no clock) for all devices.

Applicable Devices	42	R42	42A	43	R43	44

19.1 DC CHARACTERISTICS: PIC17CR42/42A/43/R43/44-16 (Commercial, Industrial) PIC17CR42/42A/43/R43/44-25 (Commercial, Industrial) PIC17CR42/42A/43/R43/44-33 (Commercial, Industrial)

DC CHARACTERISTICS			**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40˚C ≤ T_A ≤ +85˚C for industrial and 0˚C ≤ T_A ≤ +70˚C for commercial				
Parameter No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
D001	V_{DD}	Supply Voltage	4.5	–	6.0	V	
D002	V_{DR}	RAM Data Retention Voltage (Note 1)	1.5 *	–	–	V	Device in SLEEP mode
D003	V_{POR}	V_{DD} start voltage to ensure internal Power-on Reset signal	–	V_{SS}	–	V	See section on Power-on Reset for details
D004	S_{VDD}	V_{DD} rise rate to ensure internal Power-on Reset signal	0.060 *	–	–	mV/ms	See section on Power-on Reset for details
D010 D011 D012 D013 D015 D014	I_{DD}	Supply Current (Note 2)	– – – – – –	3 6 11 19 25 95	6 12 * 24 * 38 50 150	mA mA mA mA mA µA	F_{OSC} = 4 MHz (Note 4) F_{OSC} = 8 MHz F_{OSC} = 16 MHz F_{OSC} = 25 MHz F_{OSC} = 33 MHz F_{OSC} = 32 kHz, WDT enabled (EC osc configuration)
D020 D021	I_{PD}	Power-down Current (Note 3)	– –	10 < 1	40 5	µA µA	V_{DD} = 5.5V, WDT enabled V_{DD} = 5.5V, WDT disabled

* These parameters are characterized but not tested.

† Data in "Typ" column is at 5V, 25˚C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: This is the limit to which V_{DD} can be lowered in SLEEP mode without losing RAM data.

2: The supply current is mainly a function of the operating voltage and frequency. Other factors such as I/O pin loading and switching rate, oscillator type, internal code execution pattern, and temperature also have an impact on the current consumption.
The test conditions for all I_{DD} measurements in active operation mode are:
OSC1 = external square wave, from rail to rail; all I/O pins tristated, pulled to V_{DD} or V_{SS}, T0CKI = V_{DD}, $\overline{MCLR}$ = V_{DD}; WDT enabled/disabled as specified.
Current consumed from the oscillator and I/O's driving external capacitive or resistive loads needs to be considered.
For the RC oscillator, the current through the external pull-up resistor (R) can be estimated as: V_{DD} / (2 • R).
For capacitive loads, the current can be estimated (for an individual I/O pin) as (C_L • V_{DD}) • f
C_L = Total capacitive load on the I/O pin; f = average frequency the I/O pin switches.
The capacitive currents are most significant when the device is configured for external execution (includes extended microcontroller mode).

3: The power down current in SLEEP mode does not depend on the oscillator type. Power-down current is measured with the part in SLEEP mode, with all I/O pins in hi-impedance state and tied to V_{DD} and V_{SS}.

4: For RC osc configuration, current through Rext is not included. The current through the resistor can be estimated by the formula I_R = V_{DD}/2Rext (mA) with Rext in kOhm.

Applicable Devices	42	R42	42A	43	R43	44

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature $-40^{\circ}C \leq T_A \leq +40^{\circ}C$ Operating voltage V_{DD} range as described in Section 19.1				
Parameter No.	**Sym**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
		Internal Program Memory Programming Specs (Note 4)					
D110	V_{PP}	Voltage on $\overline{MCLR}/V_{PP}$ pin	12.75	–	13.25	V	Note 5
D111	V_{DDP}	Supply voltage during programming	4.75	5.0	5.25	V	
D112	I_{PP}	Current into $\overline{MCLR}/V_{PP}$ pin	–	25 ‡	50 ‡	mA	
D113	I_{DDP}	Supply current during programming	–	–	30 ‡	mA	
D114	T_{PROG}	Programming pulse width	10	100	1000	µs	Terminated via internal/external interrupt or a reset

* These parameters are characterized but not tested.

† Data in “Typ” column is at 5V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

‡ These parameters are for design guidance only and are not tested, nor characterized.

Note 1: In RC oscillator configuration, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC17CXX devices be driven with external clock in RC mode.

2: The leakage current on the $\overline{MCLR}$ pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltages.

3: Negative current is defined as coming out of the pin.

4: These specifications are for the programming of the on-chip program memory EPROM through the use of the table write instructions. The complete programming specifications can be found in: PIC17CXX Programming Specifications (Literature number DS30139).

5: The $\overline{MCLR}/V_{PP}$ pin may be kept in this range at times other than programming, but is not recommended.

6: For TTL buffers, the better of the two specifications may be used.

Note: When using the Table Write for internal programming, the device temperature must be less than 40°C.

Applicable Devices	42	R42	42A	43	R43	44

FIGURE 20-9: TYPICAL IPD vs. VDD WATCHDOG DISABLED 25°C

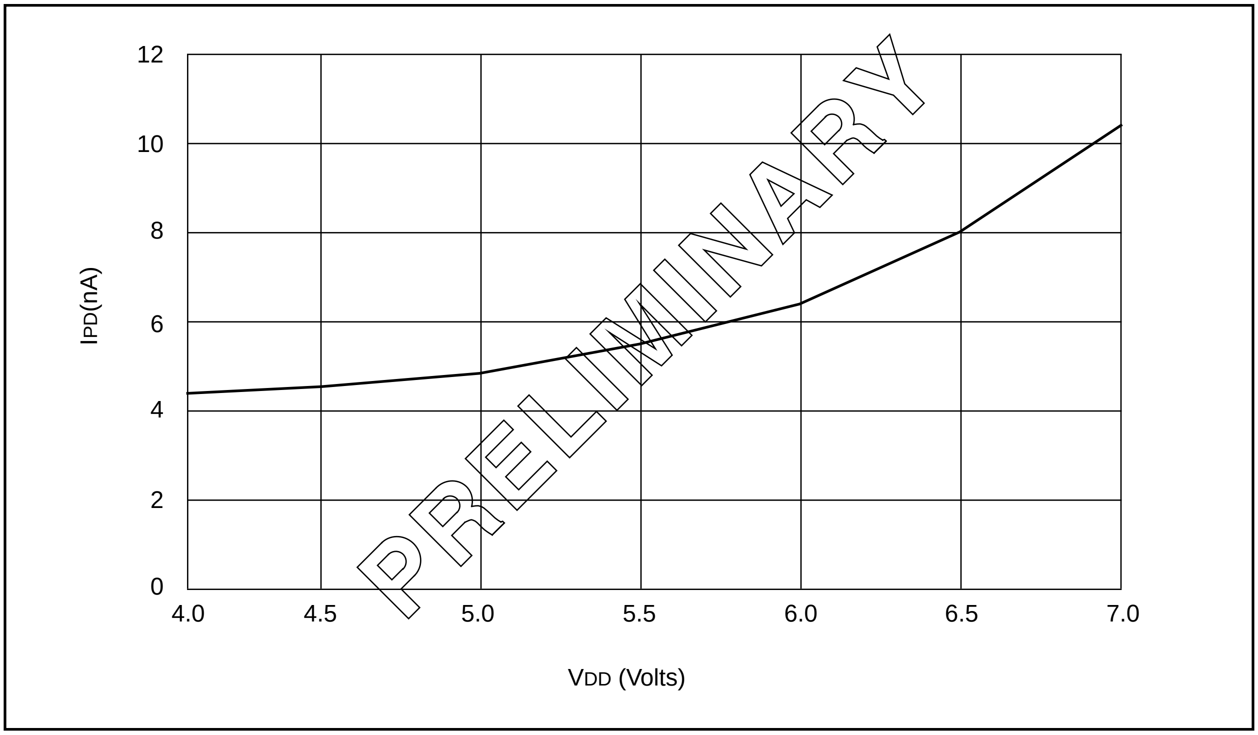
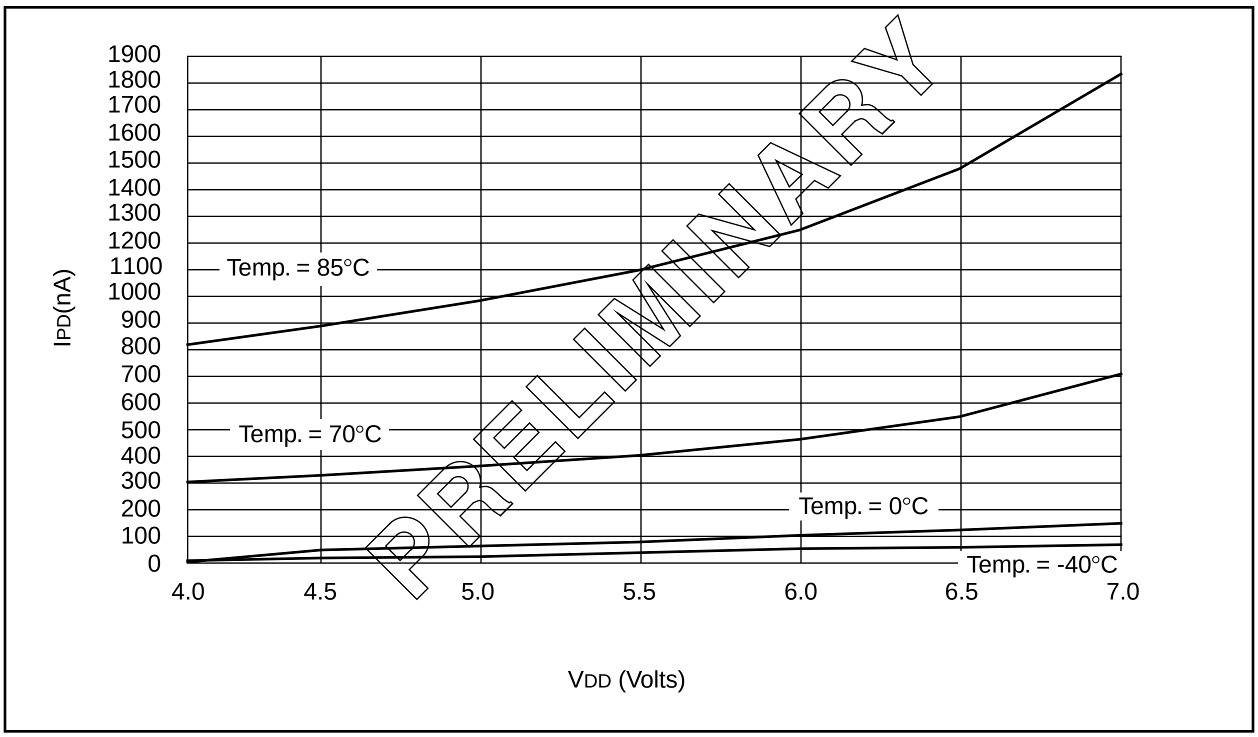


FIGURE 20-10: MAXIMUM IPD vs. VDD WATCHDOG DISABLED



E.4 PIC16C6X Family of Devices

	Clock	Memory			Peripherals						Features			
	Maximum Frequency of Operation (MHz)	Program Memory (x14 words) EPROM	Program Memory (x14 words) ROM	Data Memory (bytes)	Timer Module(s)	Capture/Compare/PWM Module(s)	Serial Port(s) (SPI/I²C, USART)	Parallel Slave Port	Interrupt Sources	I/O Pins	Voltage Range (Volts)	In-Circuit Serial Programming	Brown-out Reset	Packages
PIC16C62	20	2K	—	128	TMR0, TMR1, TMR2	1	SPI/I²C	—	7	22	3.0-6.0	Yes	—	28-pin SDIP, SOIC, SSOP
PIC16C62A(1)	20	2K	—	128	TMR0, TMR1, TMR2	1	SPI/I²C	—	7	22	2.5-6.0	Yes	Yes	28-pin SDIP, SOIC, SSOP
PIC16CR62(1)	20	—	2K	128	TMR0, TMR1, TMR2	1	SPI/I²C	—	7	22	2.5-6.0	Yes	Yes	28-pin SDIP, SOIC, SSOP
PIC16C63	20	4K	—	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	—	10	22	2.5-6.0	Yes	Yes	28-pin SDIP, SOIC
PIC16CR63(1)	20	—	4K	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	—	10	22	2.5-6.0	Yes	Yes	28-pin SDIP, SOIC
PIC16C64	20	2K	—	128	TMR0, TMR1, TMR2	1	SPI/I²C	Yes	8	33	3.0-6.0	Yes	—	40-pin DIP; 44-pin PLCC, MQFP
PIC16C64A(1)	20	2K	—	128	TMR0, TMR1, TMR2	1	SPI/I²C	Yes	8	33	2.5-6.0	Yes	Yes	40-pin DIP; 44-pin PLCC, MQFP, TQFP
PIC16CR64(1)	20	—	2K	128	TMR0, TMR1, TMR2	1	SPI/I²C	Yes	8	33	2.5-6.0	Yes	Yes	40-pin DIP; 44-pin PLCC, MQFP, TQFP
PIC16C65	20	4K	—	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	Yes	11	33	3.0-6.0	Yes	—	40-pin DIP; 44-pin PLCC, MQFP
PIC16C65A(1)	20	4K	—	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	Yes	11	33	2.5-6.0	Yes	Yes	40-pin DIP; 44-pin PLCC, MQFP, TQFP
PIC16CR65(1)	20	—	4K	192	TMR0, TMR1, TMR2	2	SPI/I²C, USART	Yes	11	33	2.5-6.0	Yes	Yes	40-pin DIP; 44-pin PLCC, MQFP, TQFP

All PIC16/17 family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect, and high I/O current capability.
All PIC16C6X family devices use serial programming with clock pin RB6 and data pin RB7.
Note 1: Please contact your local sales office for availability of these devices.

NOTES:

NOTES: