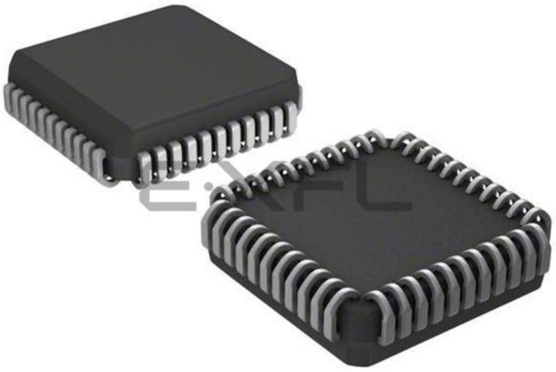


Welcome to E-XFL.COM

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	Obsolete
Core Processor	PIC
Core Size	8-Bit
Speed	33MHz
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 ~ 125°C (TA)
Mounting Type	Surface Mount
Package / Case	44-LCC (J-Lead)
Supplier Device Package	44-PLCC (16.59x16.59)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic17c44-33e-l

Table of Contents

For register and module descriptions in this data sheet, device legends show which devices apply to those sections. For example, the legend below shows that some features of only the PIC17C43, PIC17CR43, PIC17C44 are described in this section.

Applicable Devices					
42	R42	42A	43	R43	44

To Our Valued Customers

We constantly strive to improve the quality of all our products and documentation. We have spent an exceptional amount of time to ensure that these documents are correct. However, we realize that we may have missed a few things. If you find any information that is missing or appears in error from the previous version of the PIC17C4X Data Sheet (Literature Number DS30412B), please use the reader response form in the back of this data sheet to inform us. We appreciate your assistance in making this a better document.

To assist you in the use of this document, Appendix C contains a list of new information in this data sheet, while Appendix D contains information that has changed

1.0 OVERVIEW

This data sheet covers the PIC17C4X group of the PIC17CXX family of microcontrollers. The following devices are discussed in this data sheet:

- PIC17C42
- PIC17CR42
- PIC17C42A
- PIC17C43
- PIC17CR43
- PIC17C44

The PIC17CR42, PIC17C42A, PIC17C43, PIC17CR43, and PIC17C44 devices include architectural enhancements over the PIC17C42. These enhancements will be discussed throughout this data sheet.

The PIC17C4X devices are 40/44-Pin, EPROM/ROM-based members of the versatile PIC17CXX family of low-cost, high-performance, CMOS, fully-static, 8-bit microcontrollers.

All PIC16/17 microcontrollers employ an advanced RISC architecture. The PIC17CXX has enhanced core features, 16-level deep stack, and multiple internal and external interrupt sources. The separate instruction and data buses of the Harvard architecture allow a 16-bit wide instruction word with a separate 8-bit wide data. The two stage instruction pipeline allows all instructions to execute in a single cycle, except for program branches (which require two cycles). A total of 55 instructions (reduced instruction set) are available in the PIC17C42 and 58 instructions in all the other devices. Additionally, a large register set gives some of the architectural innovations used to achieve a very high performance. For mathematical intensive applications all devices, except the PIC17C42, have a single cycle 8 x 8 Hardware Multiplier.

PIC17CXX microcontrollers typically achieve a 2:1 code compression and a 4:1 speed improvement over other 8-bit microcontrollers in their class.

PIC17C4X devices have up to 454 bytes of RAM and 33 I/O pins. In addition, the PIC17C4X adds several peripheral features useful in many high performance applications including:

- Four timer/counters
- Two capture inputs
- Two PWM outputs
- A Universal Synchronous Asynchronous Receiver Transmitter (USART)

These special features reduce external components, thus reducing cost, enhancing system reliability and reducing power consumption. There are four oscillator options, of which the single pin RC oscillator provides a low-cost solution, the LF oscillator is for low frequency crystals and minimizes power consumption, XT is a standard crystal, and the EC is for external clock input. The SLEEP (power-down) mode offers additional power saving. The user can wake-up the chip from SLEEP through several external and internal interrupts and device resets.

There are four configuration options for the device operational modes:

- Microprocessor
- Microcontroller
- Extended microcontroller
- Protected microcontroller

The microprocessor and extended microcontroller modes allow up to 64K-words of external program memory.

A highly reliable Watchdog Timer with its own on-chip RC oscillator provides protection against software malfunction.

Table 1-1 lists the features of the PIC17C4X devices.

A UV-erasable CERDIP-packaged version is ideal for code development while the cost-effective One-Time Programmable (OTP) version is suitable for production in any volume.

The PIC17C4X fits perfectly in applications ranging from precise motor control and industrial process control to automotive, instrumentation, and telecom applications. Other applications that require extremely fast execution of complex software programs or the flexibility of programming the software code as one of the last steps of the manufacturing process would also be well suited. The EPROM technology makes customization of application programs (with unique security codes, combinations, model numbers, parameter storage, etc.) fast and convenient. Small footprint package options make the PIC17C4X ideal for applications with space limitations that require high performance. High speed execution, powerful peripheral features, flexible I/O, and low power consumption all at low cost make the PIC17C4X ideal for a wide range of embedded control applications.

1.1 Family and Upward Compatibility

Those users familiar with the PIC16C5X and PIC16CXX families of microcontrollers will see the architectural enhancements that have been implemented. These enhancements allow the device to be more efficient in software and hardware requirements. Please refer to Appendix A for a detailed list of enhancements and modifications. Code written for PIC16C5X or PIC16CXX can be easily ported to PIC17CXX family of devices (Appendix B).

1.2 Development Support

The PIC17CXX family is supported by a full-featured macro assembler, a software simulator, an in-circuit emulator, a universal programmer, a “C” compiler, and fuzzy logic support tools.

NOTES:

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)

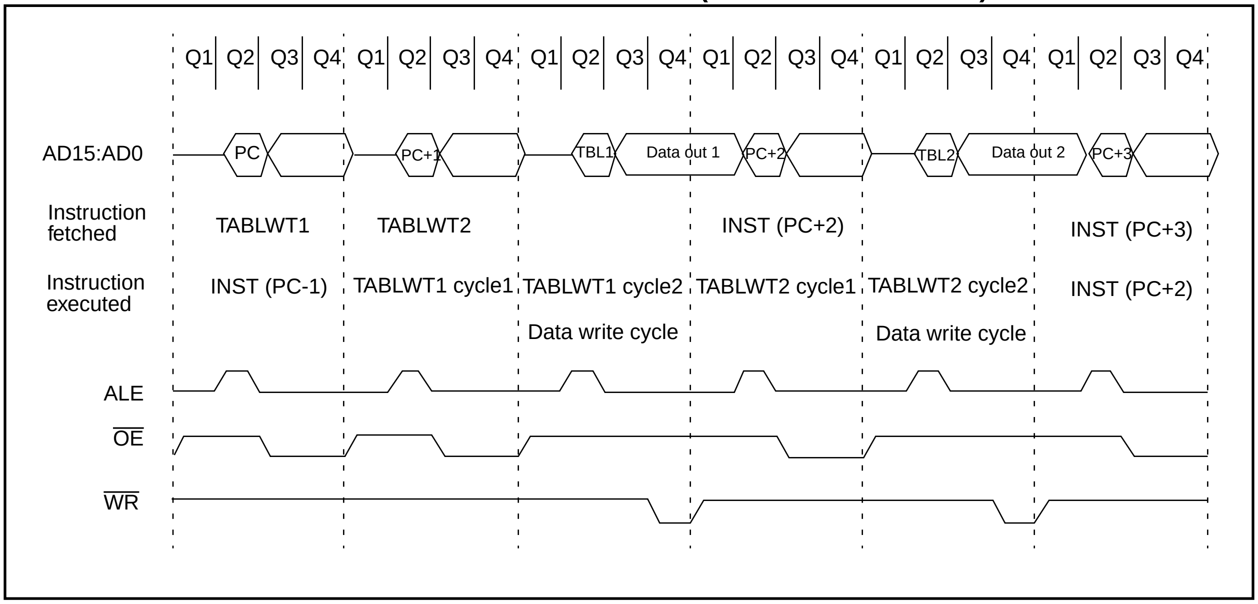
File registers TBLPTRL and TBLPTRH form a 16-bit pointer to address the 64K program memory space. The table pointer is used by instructions `TABLWT` and `TABLRD`.

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 7-6: CONSECUTIVE TABLWT WRITE TIMING (EXTERNAL MEMORY)



12.0 TIMER1, TIMER2, TIMER3, PWMS AND CAPTURES

The PIC17C4X has a wealth of timers and time-based functions to ease the implementation of control applications. These time-base functions include two PWM outputs and two Capture inputs.

Timer1 and Timer2 are two 8-bit incrementing timers, each with a period register (PR1 and PR2 respectively) and separate overflow interrupt flags. Timer1 and Timer2 can operate either as timers (increment on internal Fosc/4 clock) or as counters (increment on falling edge of external clock on pin RB4/TCLK12). They are also software configurable to operate as a single 16-bit timer. These timers are also used as the time-base for the PWM (pulse width modulation) module.

Timer3 is a 16-bit timer/counter consisting of the TMR3H and TMR3L registers. This timer has four other associated registers. Two registers are used as a 16-bit period register or a 16-bit Capture1 register (PR3H/CA1H:PR3L/CA1L). The other two registers are strictly the Capture2 registers (CA2H:CA2L). Timer3 is the time-base for the two 16-bit captures.

TMR3 can be software configured to increment from the internal system clock or from an external signal on the RB5/TCLK3 pin.

Figure 12-1 and Figure 12-2 are the control registers for the operation of Timer1, Timer2, and Timer3, as well as PWM1, PWM2, Capture1, and Capture2.

FIGURE 12-1: TCON1 REGISTER (ADDRESS: 16h, BANK 3)

R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
CA2ED1	CA2ED0	CA1ED1	CA1ED0	T16	TMR3CS	TMR2CS	TMR1CS
bit7							bit0

R = Readable bit
W = Writable bit
-n = Value at POR reset

bit 7-6: **CA2ED1:CA2ED0**: Capture2 Mode Select bits
00 = Capture on every falling edge
01 = Capture on every rising edge
10 = Capture on every 4th rising edge
11 = Capture on every 16th rising edge

bit 5-4: **CA1ED1:CA1ED0**: Capture1 Mode Select bits
00 = Capture on every falling edge
01 = Capture on every rising edge
10 = Capture on every 4th rising edge
11 = Capture on every 16th rising edge

bit 3: **T16**: Timer1:Timer2 Mode Select bit
1 = Timer1 and Timer2 form a 16-bit timer
0 = Timer1 and Timer2 are two 8-bit timers

bit 2: **TMR3CS**: Timer3 Clock Source Select bit
1 = TMR3 increments off the falling edge of the RB5/TCLK3 pin
0 = TMR3 increments off the internal clock

bit 1: **TMR2CS**: Timer2 Clock Source Select bit
1 = TMR2 increments off the falling edge of the RB4/TCLK12 pin
0 = TMR2 increments off the internal clock

bit 0: **TMR1CS**: Timer1 Clock Source Select bit
1 = TMR1 increments off the falling edge of the RB4/TCLK12 pin
0 = TMR1 increments off the internal clock

FIGURE 12-2: TCON2 REGISTER (ADDRESS: 17h, BANK 3)

R - 0	R - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0	R/W - 0
CA2OVF	CA1OVF	PWM2ON	PWM1ON	CA1/$\overline{PR3}$	TMR3ON	TMR2ON	TMR1ON
bit7							bit0

R = Readable bit
W = Writable bit
-n = Value at POR reset

bit 7: **CA2OVF**: Capture2 Overflow Status bit
This bit indicates that the capture value had not been read from the capture register pair (CA2H:CA2L) before the next capture event occurred. The capture register retains the oldest unread capture value (last capture before overflow). Subsequent capture events will not update the capture register with the Timer3 value until the capture register has been read (both bytes).
1 = Overflow occurred on Capture2 register
0 = No overflow occurred on Capture2 register

bit 6: **CA1OVF**: Capture1 Overflow Status bit
This bit indicates that the capture value had not been read from the capture register pair (PR3H/CA2H:PR3L/CA2L) before the next capture event occurred. The capture register retains the oldest unread capture value (last capture before overflow). Subsequent capture events will not update the capture register with the TMR3 value until the capture register has been read (both bytes).
1 = Overflow occurred on Capture1 register
0 = No overflow occurred on Capture1 register

bit 5: **PWM2ON**: PWM2 On bit
1 = PWM2 is enabled (The RB3/PWM2 pin ignores the state of the DDRB<3> bit)
0 = PWM2 is disabled (The RB3/PWM2 pin uses the state of the DDRB<3> bit for data direction)

bit 4: **PWM1ON**: PWM1 On bit
1 = PWM1 is enabled (The RB2/PWM1 pin ignores the state of the DDRB<2> bit)
0 = PWM1 is disabled (The RB2/PWM1 pin uses the state of the DDRB<2> bit for data direction)

bit 3: **CA1/$\overline{PR3}$**: CA1/$\overline{PR3}$ Register Mode Select bit
1 = Enables Capture1 (PR3H/CA1H:PR3L/CA1L is the Capture1 register. Timer3 runs without a period register)
0 = Enables the Period register (PR3H/CA1H:PR3L/CA1L is the Period register for Timer3)

bit 2: **TMR3ON**: Timer3 On bit
1 = Starts Timer3
0 = Stops Timer3

bit 1: **TMR2ON**: Timer2 On bit
This bit controls the incrementing of the Timer2 register. When Timer2:Timer1 form the 16-bit timer (T16 is set), TMR2ON must be set. This allows the MSB of the timer to increment.
1 = Starts Timer2 (Must be enabled if the T16 bit (TCON1<3>) is set)
0 = Stops Timer2

bit 0: **TMR1ON**: Timer1 On bit
<u>When T16 is set (in 16-bit Timer Mode)</u>
1 = Starts 16-bit Timer2:Timer1
0 = Stops 16-bit Timer2:Timer1

<u>When T16 is clear (in 8-bit Timer Mode)</u>
1 = Starts 8-bit Timer1
0 = Stops 8-bit Timer1

14.1 Configuration Bits

The PIC17CXX has up to seven configuration locations (Table 14-1). These locations can be programmed (read as '0') or left unprogrammed (read as '1') to select various device configurations. Any write to a configuration location, regardless of the data, will program that configuration bit. A `TABLWT` instruction is required to write to program memory locations. The configuration bits can be read by using the `TABLRD` instructions. Reading any configuration location between FE00h and FE07h will read the low byte of the configuration word (Figure 14-1) into the TABLATL register. The TABLATH register will be FFh. Reading a configuration location between FE08h and FE0Fh will read the high byte of the configuration word into the TABLATL register. The TABLATH register will be FFh.

Addresses FE00h thorough FE0Fh are only in the program memory space for microcontroller and code protected microcontroller modes. A device programmer will be able to read the configuration word in any processor mode. See programming specifications for more detail.

TABLE 14-1: CONFIGURATION LOCATIONS

Bit	Address
FOSC0	FE00h
FOSC1	FE01h
WDTPS0	FE02h
WDTPS1	FE03h
PM0	FE04h
PM1	FE06h
PM2 (1)	FE0Fh (1)

Note 1: This location does not exist on the PIC17C42.

Note: When programming the desired configuration locations, they must be programmed in ascending order. Starting with address FE00h.

14.2 Oscillator Configurations

14.2.1 OSCILLATOR TYPES

The PIC17CXX can be operated in four different oscillator modes. The user can program two configuration bits (FOSC1:FOSC0) to select one of these four modes:

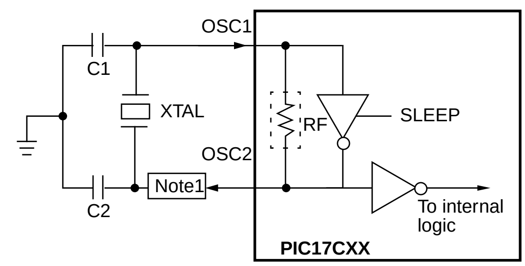
- LF: Low Power Crystal
- XT: Crystal/Resonator
- EC: External Clock Input
- RC: Resistor/Capacitor

14.2.2 CRYSTAL OSCILLATOR / CERAMIC RESONATORS

In XT or LF modes, a crystal or ceramic resonator is connected to the OSC1/CLKIN and OSC2/CLKOUT pins to establish oscillation (Figure 14-2). The PIC17CXX Oscillator design requires the use of a parallel cut crystal. Use of a series cut crystal may give a frequency out of the crystal manufacturers specifications.

For frequencies above 20 MHz, it is common for the crystal to be an overtone mode crystal. Use of overtone mode crystals require a tank circuit to attenuate the gain at the fundamental frequency. Figure 14-3 shows an example of this.

FIGURE 14-2: CRYSTAL OR CERAMIC RESONATOR OPERATION (XT OR LF OSC CONFIGURATION)



See Table 14-2 and Table 14-3 for recommended values of C1 and C2.

Note 1: A series resistor may be required for AT strip cut crystals.

14.3 Watchdog Timer (WDT)

The Watchdog Timer's function is to recover from software malfunction. The WDT uses an internal free running on-chip RC oscillator for its clock source. This does not require any external components. This RC oscillator is separate from the RC oscillator of the OSC1/CLKIN pin. That means that the WDT will run, even if the clock on the OSC1/CLKIN and OSC2/CLKOUT pins of the device has been stopped, for example, by execution of a `SLEEP` instruction. During normal operation and SLEEP mode, a WDT time-out generates a device RESET. The WDT can be permanently disabled by programming the configuration bits WDTPS1:WDTPS0 as '`00`' (Section 14.1).

Under normal operation, the WDT must be cleared on a regular interval. This time is less the minimum WDT overflow time. Not clearing the WDT in this time frame will cause the WDT to overflow and reset the device.

14.3.1 WDT PERIOD

The WDT has a nominal time-out period of 12 ms, (with postscaler = 1). The time-out periods vary with temperature, V_{DD} and process variations from part to part (see DC specs). If longer time-out periods are desired, a postscaler with a division ratio of up to 1:256 can be assigned to the WDT. Thus, typical time-out periods up to 3.0 seconds can be realized.

The `CLRWDT` and `SLEEP` instructions clear the WDT and the postscaler (if assigned to the WDT) and prevent it from timing out thus generating a device RESET condition.

The $\overline{TO}$ bit in the CPUSTA register will be cleared upon a WDT time-out.

14.3.2 CLEARING THE WDT AND POSTSCALER

The WDT and postscaler are cleared when:

- The device is in the reset state
- A `SLEEP` instruction is executed
- A `CLRWDT` instruction is executed
- Wake-up from SLEEP by an interrupt

The WDT counter/postscaler will start counting on the first edge after the device exits the reset state.

14.3.3 WDT PROGRAMMING CONSIDERATIONS

It should also be taken in account that under worst case conditions (V_{DD} = Min., Temperature = Max., max. WDT postscaler) it may take several seconds before a WDT time-out occurs.

The WDT and postscaler is the Power-up Timer during the Power-on Reset sequence.

14.3.4 WDT AS NORMAL TIMER

When the WDT is selected as a normal timer, the clock source is the device clock. Neither the WDT nor the postscaler are directly readable or writable. The overflow time is 65536 T_{OSC} cycles. On overflow, the $\overline{TO}$ bit is cleared (device is not reset). The `CLRWDT` instruction can be used to set the $\overline{TO}$ bit. This allows the WDT to be a simple overflow timer. When in sleep, the WDT does not increment.

Table 15-2 lists the instructions recognized by the MPASM assembler.

> **Note 1:** Any unused opcode is Reserved. Use of any reserved opcode may cause unexpected operation.
>
> **Note 2:** The shaded instructions are not available in the PIC17C42

All instruction examples use the following format to represent a hexadecimal number:

0xhh

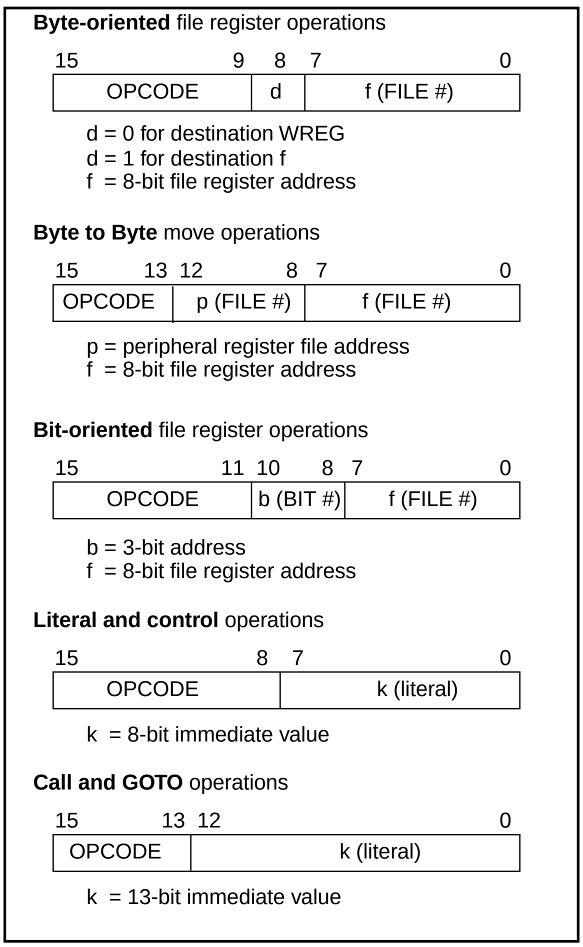
where h signifies a hexadecimal digit.

To represent a binary number:

`0000 0100b`

where b signifies a binary string.

FIGURE 15-1: GENERAL FORMAT FOR INSTRUCTIONS



15.1 Special Function Registers as Source/Destination

The PIC17C4X's orthogonal instruction set allows read and write of all file registers, including special function registers. There are some special situations the user should be aware of:

15.1.1 ALUSTA AS DESTINATION

If an instruction writes to ALUSTA, the Z, C, DC and OV bits may be set or cleared as a result of the instruction and overwrite the original data bits written. For example, executing `CLRF ALUSTA` will clear register ALUSTA, and then set the Z bit leaving `0000 0100b` in the register.

15.1.2 PCL AS SOURCE OR DESTINATION

Read, write or read-modify-write on PCL may have the following results:

Read PC:	PCH → PCLATH; PCL → dest
Write PCL:	PCLATH → PCH; 8-bit destination value → PCL
Read-Modify-Write:	PCL→ ALU operand PCLATH → PCH; 8-bit result → PCL

Where PCH = program counter high byte (not an addressable register), PCLATH = Program counter high holding latch, dest = destination, WREG or f.

15.1.3 BIT MANIPULATION

All bit manipulation instructions are done by first reading the entire register, operating on the selected bit and writing the result back (read-modify-write). The user should keep this in mind when operating on special function registers, such as ports.

15.2 Q Cycle Activity

Each instruction cycle (Tcy) is comprised of four Q cycles (Q1-Q4). The Q cycles provide the timing/designation for the Decode, Read, Execute, Write etc., of each instruction cycle. The following diagram shows the relationship of the Q cycles to the instruction cycle.

The 4 Q cycles that make up an instruction cycle (Tcy) can be generalized as:

Q1: Instruction Decode Cycle or forced NOP

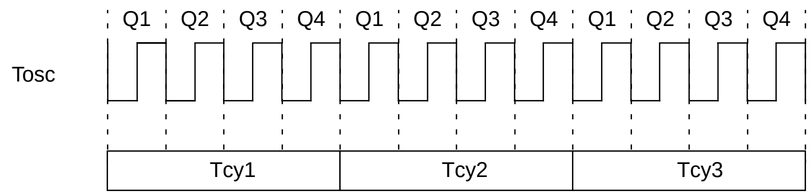
Q2: Instruction Read Cycle or NOP

Q3: Instruction Execute

Q4: Instruction Write Cycle or NOP

Each instruction will show the detailed Q cycle operation for the instruction.

FIGURE 15-2: Q CYCLE ACTIVITY



SWAPF — Swap f

Syntax: [*label*] SWAPF f,d

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]

Operation: f<3:0> → dest<7:4>;
f<7:4> → dest<3:0>

Status Affected: None

Encoding:

`0001`	`110d`	`ffff`	`ffff`

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is 0 the result is placed in WREG. If 'd' is 1 the result is placed in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

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

Example: `SWAPF REG, 0`

Before Instruction
REG = 0x53

After Instruction
REG = 0x35

TABLRD — Table Read

Syntax: [*label*] TABLRD t,i,f

Operands: 0 ≤ f ≤ 255
i ∈ [0,1]
t ∈ [0,1]

Operation: If t = 1,
TBLATH → f;
If t = 0,
TBLATL → f;
Prog Mem (TBLPTR) → TBLAT;
If i = 1,
TBLPTR + 1 → TBLPTR

Status Affected: None

Encoding:

`1010`	`10ti`	`ffff`	`ffff`

Description:
1. A byte of the table latch (TBLAT) is moved to register file 'f'.
If t = 0: the high byte is moved;
If t = 1: the low byte is moved
2. Then the contents of the program memory location pointed to by the 16-bit Table Pointer (TBLPTR) is loaded into the 16-bit Table Latch (TBLAT).
3. If i = 1: TBLPTR is incremented;
If i = 0: TBLPTR is not incremented

Words: 1

Cycles: 2 (3 cycle if f = PCL)

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register TBLATH or TBLATL	Execute	Write register 'f'

Applicable Devices	42	R42	42A	43	R43	44

FIGURE 17-7: CAPTURE TIMINGS

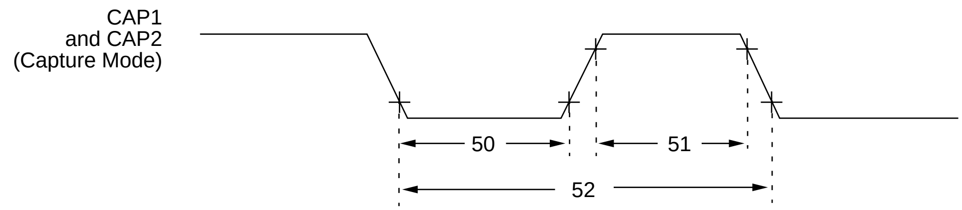


TABLE 17-7: CAPTURE REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
50	TccL	Capture1 and Capture2 input low time	10 *	—	—	ns	
51	TccH	Capture1 and Capture2 input high time	10 *	—	—	ns	
52	TccP	Capture1 and Capture2 input period	$\frac{2\ T_{CY}}{N}$ §	—	—	ns	N = prescale value (4 or 16)

* 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.
§ This specification ensured by design.

FIGURE 17-8: PWM TIMINGS

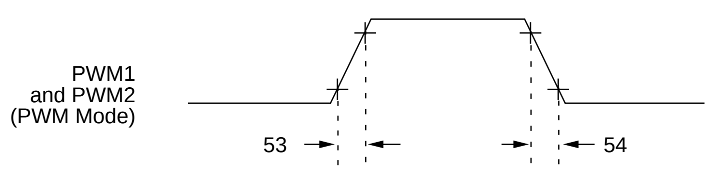


TABLE 17-8: PWM REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
53	TccR	PWM1 and PWM2 output rise time	—	10 *	35 *§	ns	
54	TccF	PWM1 and PWM2 output fall time	—	10 *	35 *§	ns	

* 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.
§ This specification ensured by design.

Applicable Devices	42	R42	42A	43	R43	44

FIGURE 19-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, AND POWER-UP TIMER TIMING

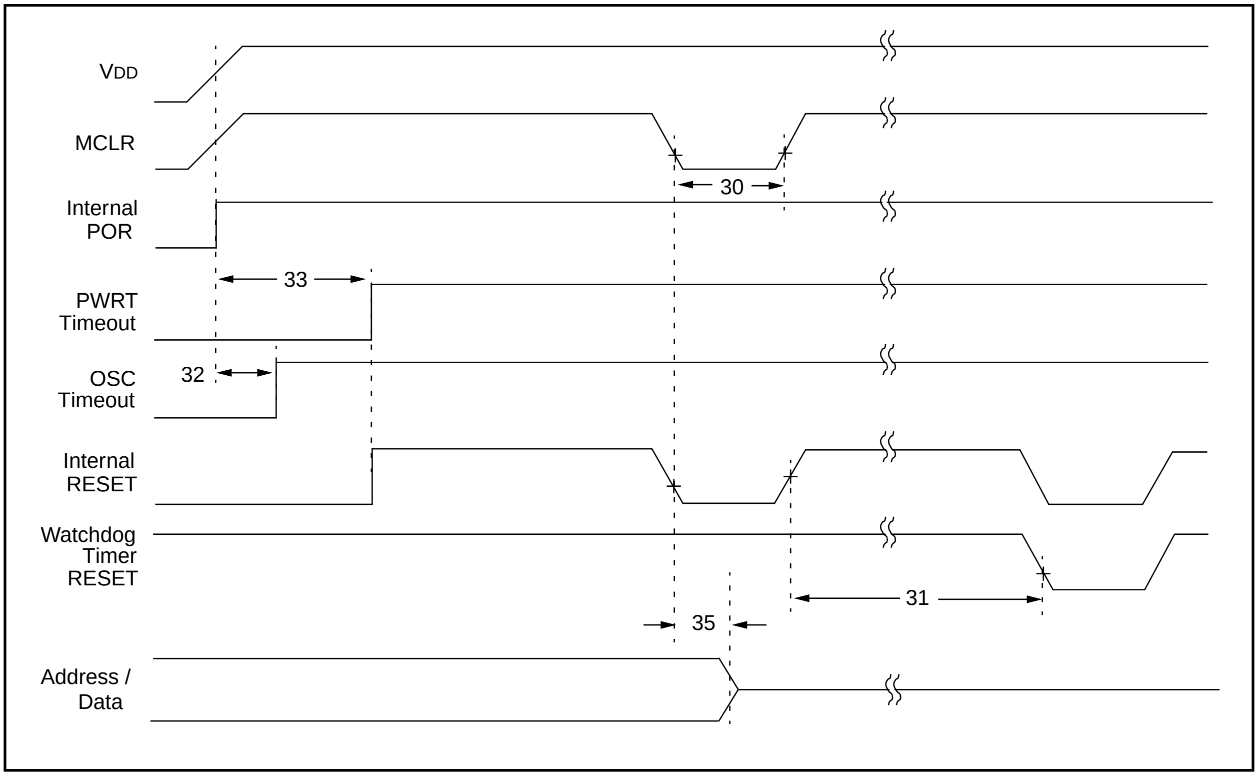


TABLE 19-4: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER REQUIREMENTS

Parameter No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
30	TmcL	$\overline{\text{MCLR}}$ Pulse Width (low)		100 *	—	—	ns	VDD = 5V
31	Twdt	Watchdog Timer Time-out Period (Prescale = 1)		5 *	12	25 *	ms	VDD = 5V
32	Tost	Oscillation Start-up Timer Period		—	1024TOSC§	—	ms	TOSC = OSC1 period
33	Tpwrt	Power-up Timer Period		40 *	96	200 *	ms	VDD = 5V
35	TmcL2adl	$\overline{\text{MCLR}}$ to System Interface bus (AD15:AD0>) invalid	PIC17CR42/42A/43/R43/44	—	—	100 *	ns	
			PIC17LCR42/42A/43/R43/44	—	—	120 *	ns	

* 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.
§ This specification ensured by design.

Applicable Devices	42	R42	42A	43	R43	44

FIGURE 19-12: MEMORY INTERFACE READ TIMING (NOT SUPPORTED IN PIC17LC4X DEVICES)

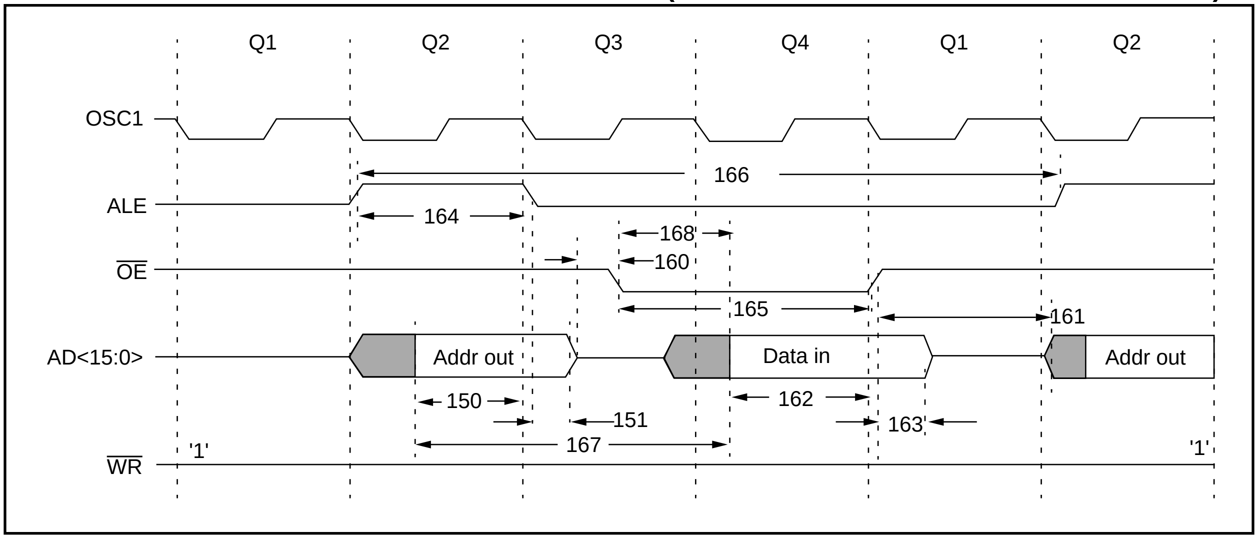


TABLE 19-12: MEMORY INTERFACE READ REQUIREMENTS (NOT SUPPORTED IN PIC17LC4X DEVICES)

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
150	TadV2alL	AD15:AD0 (address) valid to ALE↓ (address setup time)	0.25Tcy - 10	—	—	ns	
151	TalL2adI	ALE↓ to address out invalid (address hold time)	5*	—	—	ns	
160	TadZ2oeL	AD15:AD0 hi-impedance to $\overline{OE}$↓	0*	—	—	ns	
161	ToeH2adD	$\overline{OE}$↑ to AD15:AD0 driven	0.25Tcy - 15	—	—	ns	
162	TadV2oeH	Data in valid before $\overline{OE}$↑ (data setup time)	35	—	—	ns	
163	ToeH2adI	$\overline{OE}$↑to data in invalid (data hold time)	0	—	—	ns	
164	TalH	ALE pulse width	—	0.25TCY §	—	ns	
165	ToeL	$\overline{OE}$ pulse width	0.5Tcy - 35 §	—	—	ns	
166	TalH2alH	ALE↑ to ALE↑(cycle time)	—	TCY §	—	ns	
167	Tacc	Address access time	—	—	0.75TCY - 30	ns	
168	Toe	Output enable access time ($\overline{OE}$ low to Data Valid)	—	—	0.5TCY - 45	ns	

* 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.

§ This specification ensured by design.

NOTES:

Applicable Devices	42	R42	42A	43	R43	44

FIGURE 20-5: TRANSCONDUCTANCE (gm) OF LF OSCILLATOR vs. VDD

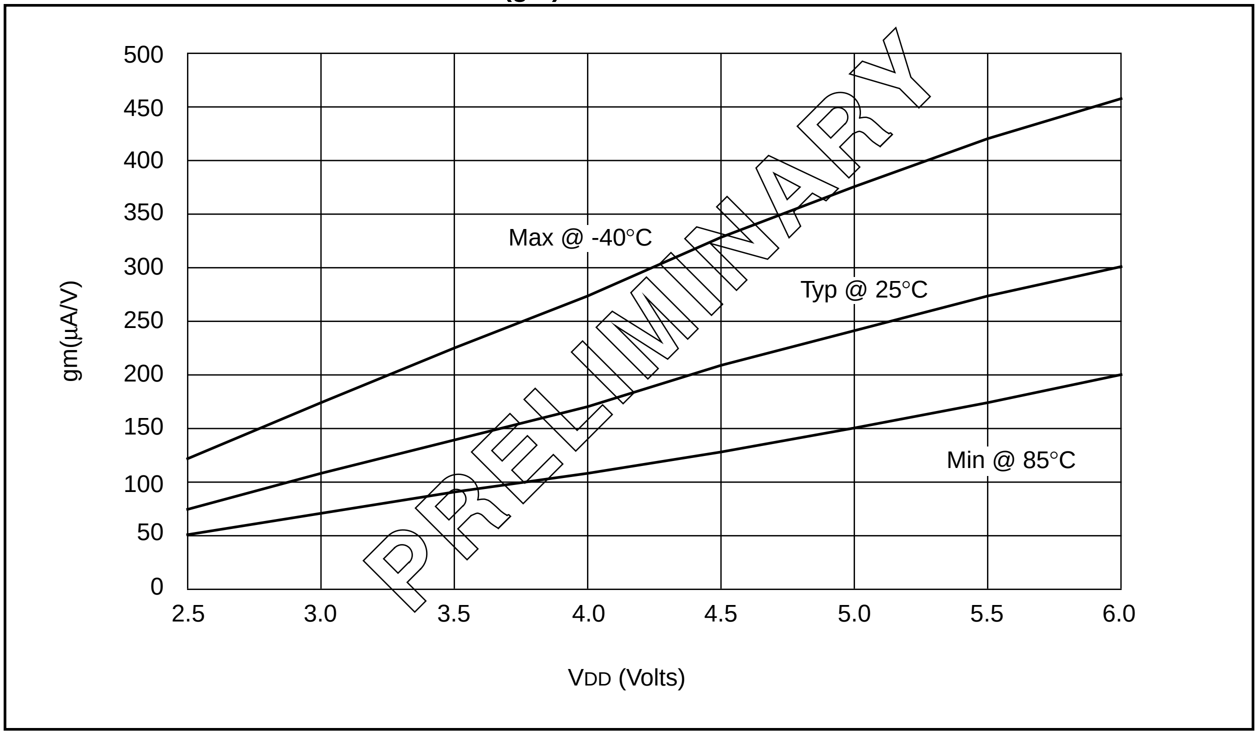
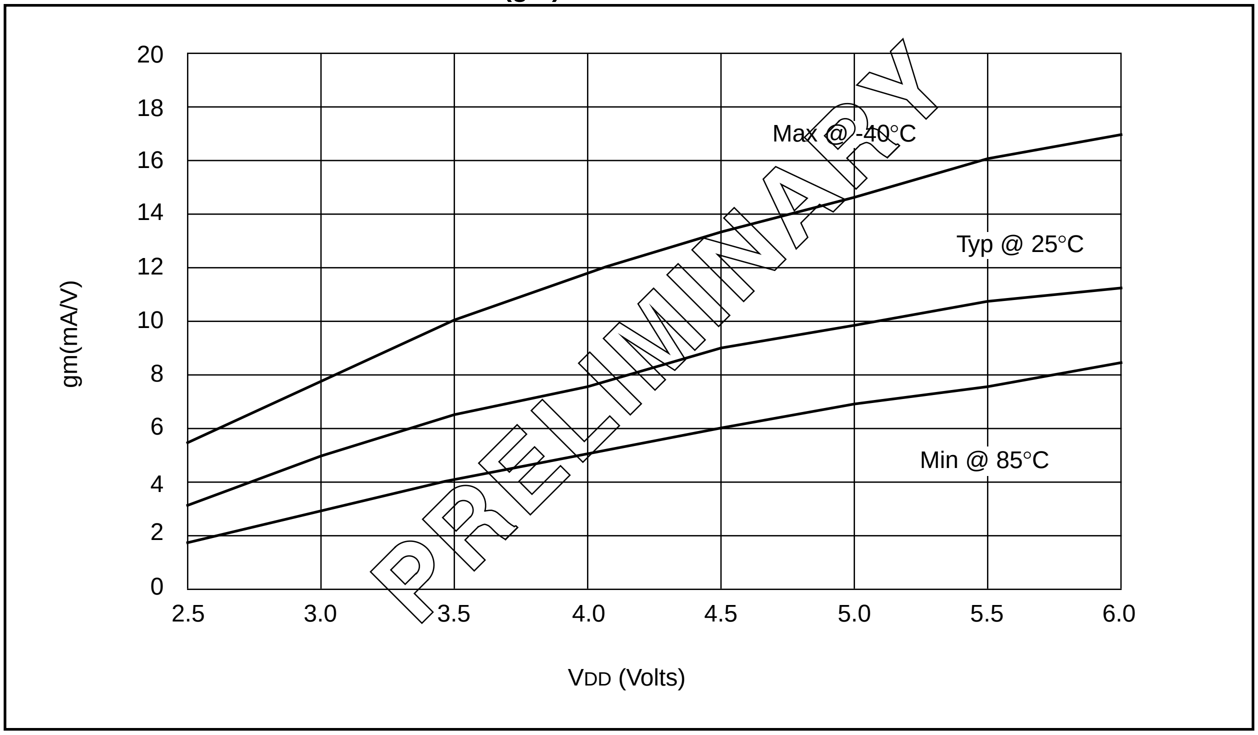


FIGURE 20-6: TRANSCONDUCTANCE (gm) OF XT OSCILLATOR vs. VDD



Applicable Devices	42	R42	42A	43	R43	44

FIGURE 20-17: IOL vs. VOL, VDD = 5V

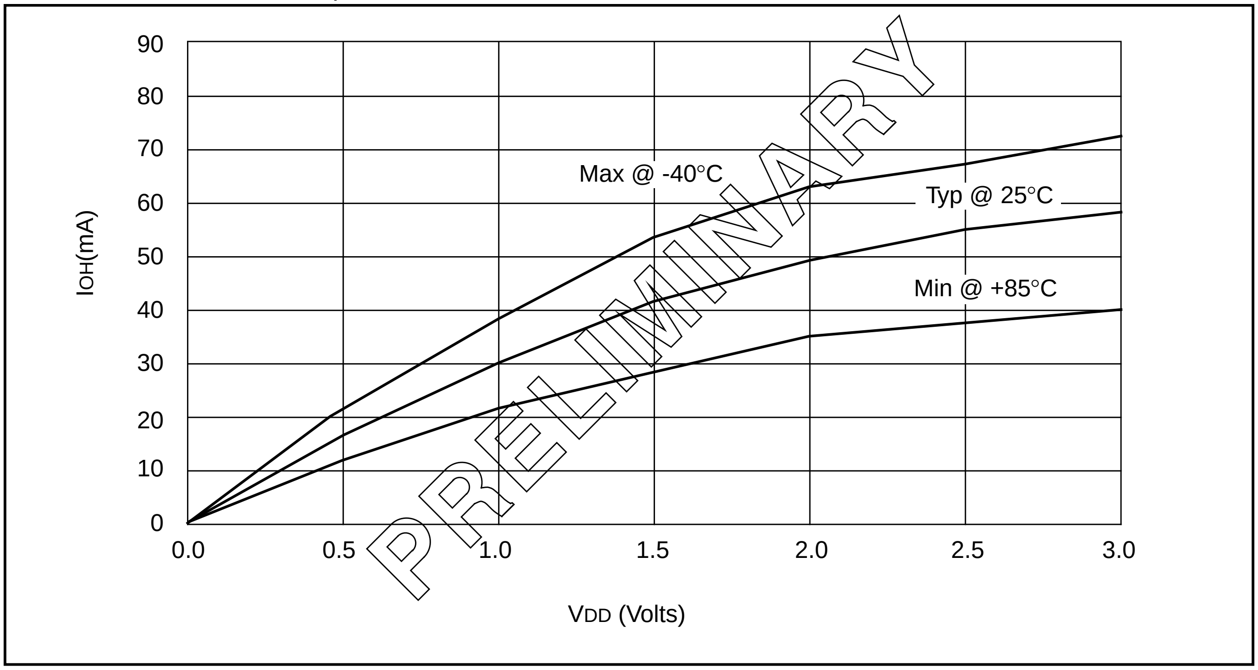
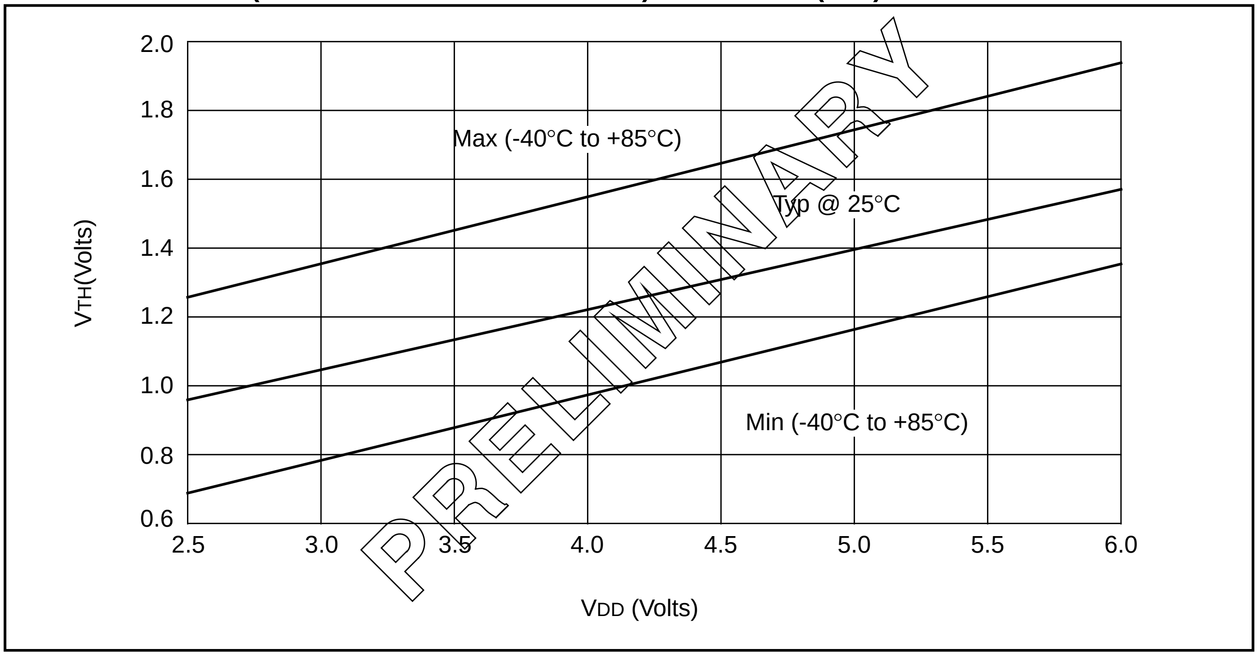


FIGURE 20-18: VTH (INPUT THRESHOLD VOLTAGE) OF I/O PINS (TTL) vs. VDD



APPENDIX A: MODIFICATIONS

The following is the list of modifications over the PIC16CXX microcontroller family:

1. Instruction word length is increased to 16-bit. This allows larger page sizes both in program memory (8 Kwords verses 2 Kwords) and register file (256 bytes versus 128 bytes).
2. Four modes of operation: microcontroller, protected microcontroller, extended microcontroller, and microprocessor.
3. 22 new instructions.
   The `MOVF`, `TRIS` and `OPTION` instructions have been removed.
4. 4 new instructions for transferring data between data memory and program memory. This can be used to “self program” the EPROM program memory.
5. Single cycle data memory to data memory transfers possible (`MOVPF` and `MOVFP` instructions). These instructions do not affect the Working register (WREG).
6. W register (WREG) is now directly addressable.
7. A PC high latch register (PCLATH) is extended to 8-bits. The PCLATCH register is now both readable and writable.
8. Data memory paging is redefined slightly.
9. DDR registers replaces function of TRIS registers.
10. Multiple Interrupt vectors added. This can decrease the latency for servicing the interrupt.
11. Stack size is increased to 16 deep.
12. BSR register for data memory paging.
13. Wake up from SLEEP operates slightly differently.
14. The Oscillator Start-Up Timer (OST) and Power-Up Timer (PWRT) operate in parallel and not in series.
15. PORTB interrupt on change feature works on all eight port pins.
16. TMR0 is 16-bit plus 8-bit prescaler.
17. Second indirect addressing register added (FSR1 and FSR2). Configuration bits can select the FSR registers to auto-increment, auto-decrement, remain unchanged after an indirect address.
18. Hardware multiplier added (8 x 8 $\rightarrow$ 16-bit) (PIC17C43 and PIC17C44 only).
19. Peripheral modules operate slightly differently.
20. Oscillator modes slightly redefined.
21. Control/Status bits and registers have been placed in different registers and the control bit for globally enabling interrupts has inverse polarity.
22. Addition of a test mode pin.
23. In-circuit serial programming is not implemented.

APPENDIX B: COMPATIBILITY

To convert code written for PIC16CXX to PIC17CXX, the user should take the following steps:

1. Remove any `TRIS` and `OPTION` instructions, and implement the equivalent code.
2. Separate the interrupt service routine into its four vectors.
3. Replace:
   ```
   MOVF    REG1, W
   ```
   with:
   ```
   MOVFP   REG1, WREG
   ```
4. Replace:
   ```
   MOVF    REG1, W
   MOVWF   REG2
   ```
   with:
   ```
   MOVPF   REG1, REG2 ; Addr(REG1)<20h
   ```
   or
   ```
   MOVFP   REG1, REG2 ; Addr(REG2)<20h
   ```

Note:	If REG1 and REG2 are both at addresses greater then 20h, two instructions are required.
	`MOVFP   REG1, WREG ;`
	`MOVPF   WREG, REG2 ;`

5. Ensure that all bit names and register names are updated to new data memory map location.
6. Verify data memory banking.
7. Verify mode of operation for indirect addressing.
8. Verify peripheral routines for compatibility.
9. Weak pull-ups are enabled on reset.

To convert code from the PIC17C42 to all the other PIC17C4X devices, the user should take the following steps:

1. If the hardware multiply is to be used, ensure that any variables at address 18h and 19h are moved to another address.
2. Ensure that the upper nibble of the BSR was not written with a non-zero value. This may cause unexpected operation since the RAM bank is no longer 0.
3. The disabling of global interrupts has been enhanced so there is no additional testing of the GLINTD bit after a `BSF CPUSTA, GLINTD` instruction.

LIST OF TABLES