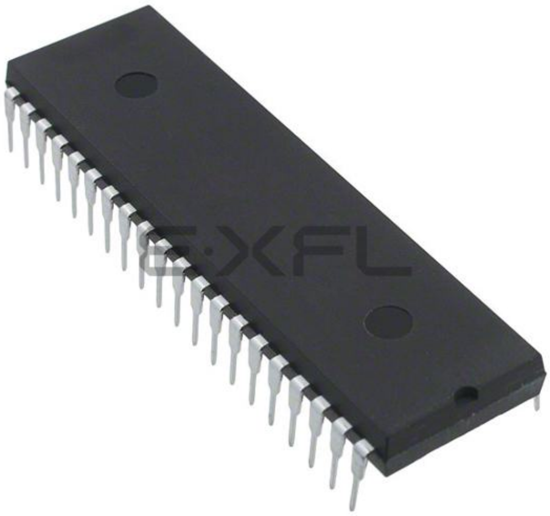


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

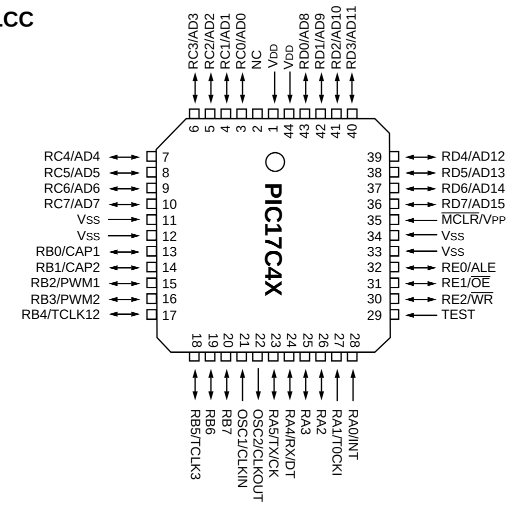
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" 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"

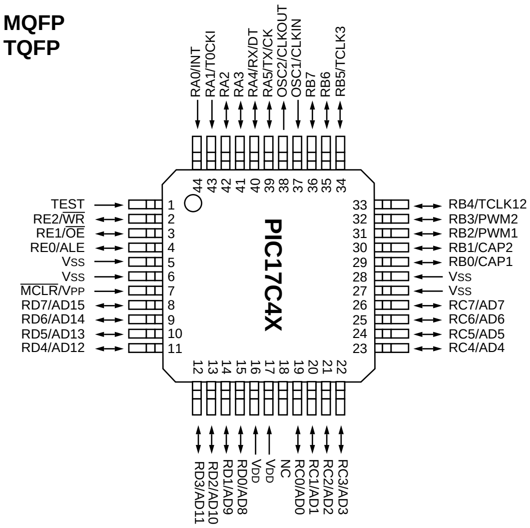
Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	33MHz
Connectivity	UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	33
Program Memory Size	16KB (8K x 16)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	454 x 8
Voltage - Supply (Vcc/Vdd)	4.5V ~ 6V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Through Hole
Package / Case	40-DIP (0.600", 15.24mm)
Supplier Device Package	40-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic17c44-33i-p

Pin Diagrams Cont.'d

PLCC



MQFP
TQFP



All devices are available in all package types, listed in Section 21.0, with the following exceptions:

- ROM devices are not available in Windowed CERDIP Packages
- TQFP is not available for the PIC17C42.

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:

TABLE 3-1: PINOUT DESCRIPTIONS

Name	DIP No.	PLCC No.	QFP No.	I/O/P Type	Buffer Type	Description
						PORTD is a bi-directional I/O Port.
RD0/AD8	40	43	15	I/O	TTL	This is also the upper byte of the 16-bit system bus in microprocessor mode or extended microprocessor mode or extended microcontroller mode. In multiplexed system bus configuration these pins are address output as well as data input or output.
RD1/AD9	39	42	14	I/O	TTL	
RD2/AD10	38	41	13	I/O	TTL	
RD3/AD11	37	40	12	I/O	TTL	
RD4/AD12	36	39	11	I/O	TTL	
RD5/AD13	35	38	10	I/O	TTL	
RD6/AD14	34	37	9	I/O	TTL	
RD7/AD15	33	36	8	I/O	TTL	
						PORTE is a bi-directional I/O Port.
RE0/ALE	30	32	4	I/O	TTL	In microprocessor mode or extended microcontroller mode, it is the Address Latch Enable (ALE) output. Address should be latched on the falling edge of ALE output.
RE1/$\overline{\text{OE}}$	29	31	3	I/O	TTL	In microprocessor or extended microcontroller mode, it is the Output Enable ($\overline{\text{OE}}$) control output (active low).
RE2/$\overline{\text{WR}}$	28	30	2	I/O	TTL	In microprocessor or extended microcontroller mode, it is the Write Enable ($\overline{\text{WR}}$) control output (active low).
TEST	27	29	1	I	ST	Test mode selection control input. Always tie to V_{SS} for normal operation.
V_{SS}	10, 31	11, 12, 33, 34	5, 6, 27, 28	P		Ground reference for logic and I/O pins.
V_{DD}	1	1, 44	16, 17	P		Positive supply for logic and I/O pins.

Legend: I = Input only; O = Output only; I/O = Input/Output; P = Power; — = Not Used; TTL = TTL input; ST = Schmitt Trigger input.

6.2.2.1 ALU STATUS REGISTER (ALUSTA)

The ALUSTA register contains the status bits of the Arithmetic and Logic Unit and the mode control bits for the indirect addressing register.

As with all the other registers, the ALUSTA register can be the destination for any instruction. If the ALUSTA register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Therefore, the result of an instruction with the ALUSTA register as destination may be different than intended.

For example, `CLRF ALUSTA` will clear the upper four bits and set the Z bit. This leaves the ALUSTA register as `0000u1uu` (where u = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions be used to alter the ALUSTA register because these instructions do not affect any status bit. To see how other instructions affect the status bits, see the “Instruction Set Summary.”

> **Note 1:** The C and DC bits operate as a $\overline{\text{borrow}}$ out bit in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.
>
> **Note 2:** The overflow bit will be set if the 2’s complement result exceeds +127 or is less than -128.

Arithmetic and Logic Unit (ALU) is capable of carrying out arithmetic or logical operations on two operands or a single operand. All single operand instructions operate either on the WREG register or a file register. For two operand instructions, one of the operands is the WREG register and the other one is either a file register or an 8-bit immediate constant.

FIGURE 6-7: ALUSTA REGISTER (ADDRESS: 04h, UNBANKED)

R/W - 1	R/W - 1	R/W - 1	R/W - 1	R/W - x	R/W - x	R/W - x	R/W - x
FS3	FS2	FS1	FS0	OV	Z	DC	C
bit7							bit0

R = Readable bit
W = Writable bit
-n = Value at POR reset
(x = unknown)

bit 7-6: **FS3:FS2**: FSR1 Mode Select bits
00 = Post auto-decrement FSR1 value
01 = Post auto-increment FSR1 value
1x = FSR1 value does not change

bit 5-4: **FS1:FS0**: FSR0 Mode Select bits
00 = Post auto-decrement FSR0 value
01 = Post auto-increment FSR0 value
1x = FSR0 value does not change

bit 3: **OV**: Overflow bit
This bit is used for signed arithmetic (2’s complement). It indicates an overflow of the 7-bit magnitude, which causes the sign bit (bit7) to change state.
1 = Overflow occurred for signed arithmetic, (in this arithmetic operation)
0 = No overflow occurred

bit 2: **Z**: Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The results of an arithmetic or logic operation is not zero

bit 1: **DC**: Digit carry/$\overline{\text{borrow}}$ bit
For `ADDWF` and `ADDLW` instructions.
1 = A carry-out from the 4th low order bit of the result occurred
0 = No carry-out from the 4th low order bit of the result
Note: For borrow the polarity is reversed.

bit 0: **C**: carry/$\overline{\text{borrow}}$ bit
For `ADDWF` and `ADDLW` instructions.
1 = A carry-out from the most significant bit of the result occurred
Note that a subtraction is executed by adding the two’s complement of the second operand. For rotate (`RRCF`, `RLCF`) instructions, this bit is loaded with either the high or low order bit of the source register.
0 = No carry-out from the most significant bit of the result
Note: For borrow the polarity is reversed.

6.7 Program Counter Module

The Program Counter (PC) is a 16-bit register. PCL, the low byte of the PC, is mapped in the data memory. PCL is readable and writable just as is any other register. PCH is the high byte of the PC and is not directly addressable. Since PCH is not mapped in data or program memory, an 8-bit register PCLATH (PC high latch) is used as a holding latch for the high byte of the PC. PCLATH is mapped into data memory. The user can read or write PCH through PCLATH.

The 16-bit wide PC is incremented after each instruction fetch during Q1 unless:

- Modified by `GOTO`, `CALL`, `LCALL`, `RETURN`, `RETLW`, or `RETFIE` instruction
- Modified by an interrupt response
- Due to destination write to PCL by an instruction

"Skips" are equivalent to a forced NOP cycle at the skipped address.

Figure 6-11 and Figure 6-12 show the operation of the program counter for various situations.

FIGURE 6-11: PROGRAM COUNTER OPERATION

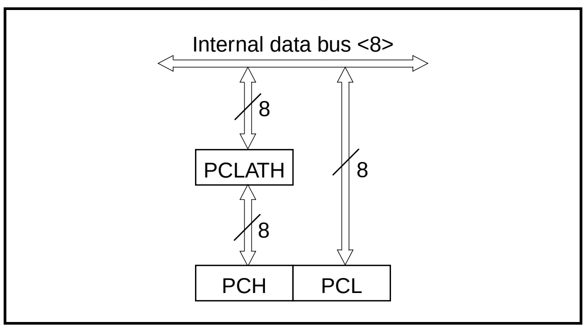
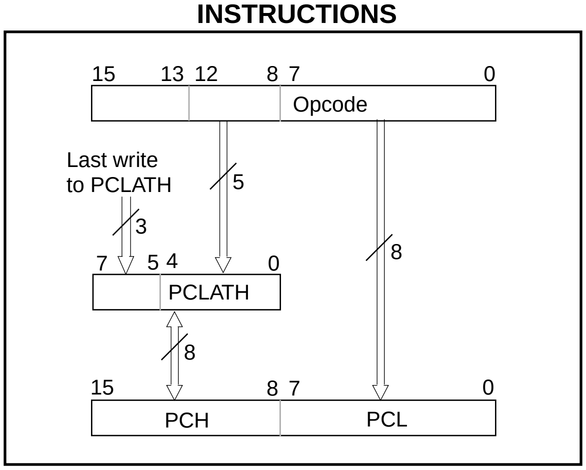


FIGURE 6-12: PROGRAM COUNTER USING THE CALL AND GOTO INSTRUCTIONS



Using Figure 6-11, the operations of the PC and PCLATH for different instructions are as follows:

a) `LCALL` instructions:
   An 8-bit destination address is provided in the instruction (opcode). PCLATH is unchanged.
   PCLATH → PCH
   Opcode<7:0> → PCL

b) Read instructions on PCL:
   Any instruction that reads PCL.
   PCL → data bus → ALU or destination
   PCH → PCLATH

c) Write instructions on PCL:
   Any instruction that writes to PCL.
   8-bit data → data bus → PCL
   PCLATH → PCH

d) Read-Modify-Write instructions on PCL:
   Any instruction that does a read-write-modify operation on PCL, such as `ADDWF PCL`.
   Read: PCL → data bus → ALU
   Write: 8-bit result → data bus → PCL
   PCLATH → PCH

e) `RETURN` instruction:
   PCH → PCLATH
   Stack<MRU> → PC<15:0>

Using Figure 6-12, the operation of the PC and PCLATH for `GOTO` and `CALL` instructions is a follows:

`CALL`, `GOTO` instructions:
A 13-bit destination address is provided in the instruction (opcode).
Opcode<12:0> → PC <12:0>
PC<15:13> → PCLATH<7:5>
Opcode<12:8> → PCLATH <4:0>

The read-modify-write only affects the PCL with the result. PCH is loaded with the value in the PCLATH. For example, `ADDWF PCL` will result in a jump within the current page. If PC = 03F0h, WREG = 30h and PCLATH = 03h before instruction, PC = 0320h after the instruction. To accomplish a true 16-bit computed jump, the user needs to compute the 16-bit destination address, write the high byte to PCLATH and then write the low value to PCL.

The following PC related operations do not change PCLATH:

a) `LCALL`, `RETLW`, and `RETFIE` instructions.
b) Interrupt vector is forced onto the PC.
c) Read-modify-write instructions on PCL (e.g. `BSF PCL`).

Example 8-3 shows the sequence to do a 16 x 16 unsigned multiply. Equation 8-1 shows the algorithm that is used. The 32-bit result is stored in 4 registers RES3:RES0.

EQUATION 8-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L * ARG2H:ARG2L \\
&= (ARG1H * ARG2H * 2^{16}) + \\
&\quad (ARG1H * ARG2L * 2^{8}) + \\
&\quad (ARG1L * ARG2H * 2^{8}) + \\
&\quad (ARG1L * ARG2L)
\end{aligned}
$$

EXAMPLE 8-3: 16 x 16 MULTIPLY ROUTINE

```
   MOVFP    ARG1L, WREG
   MULWF    ARG2L       ; ARG1L * ARG2L ->
                        ;   PRODH:PRODL
   MOVPF    PRODH, RES1 ;
   MOVPF    PRODL, RES0 ;
;
   MOVFP    ARG1H, WREG
   MULWF    ARG2H       ; ARG1H * ARG2H ->
                        ;   PRODH:PRODL
   MOVPF    PRODH, RES3 ;
   MOVPF    PRODL, RES2 ;
;
   MOVFP    ARG1L, WREG
   MULWF    ARG2H       ; ARG1L * ARG2H ->
                        ;   PRODH:PRODL
   MOVFP    PRODL, WREG ;
   ADDWF    RES1, F     ; Add cross
   MOVFP    PRODH, WREG ;    products
   ADDWFC   RES2, F     ;
   CLRF     WREG, F     ;
   ADDWFC   RES3, F     ;
;
   MOVFP    ARG1H, WREG ;
   MULWF    ARG2L       ; ARG1H * ARG2L ->
                        ;   PRODH:PRODL

   MOVFP    PRODL, WREG ;
   ADDWF    RES1, F     ; Add cross
   MOVFP    PRODH, WREG ;    products
   ADDWFC   RES2, F     ;
   CLRF     WREG, F     ;
   ADDWFC   RES3, F     ;
```

9.0 I/O PORTS

The PIC17C4X devices have five I/O ports, PORTA through PORTE. PORTB through PORTE have a corresponding Data Direction Register (DDR), which is used to configure the port pins as inputs or outputs. These five ports are made up of 33 I/O pins. Some of these ports pins are multiplexed with alternate functions.

PORTC, PORTD, and PORTE are multiplexed with the system bus. These pins are configured as the system bus when the device's configuration bits are selected to Microprocessor or Extended Microcontroller modes. In the two other microcontroller modes, these pins are general purpose I/O.

PORTA and PORTB are multiplexed with the peripheral features of the device. These peripheral features are:

- Timer modules
- Capture module
- PWM module
- USART/SCI module
- External Interrupt pin

When some of these peripheral modules are turned on, the port pin will automatically configure to the alternate function. The modules that do this are:

- PWM module
- USART/SCI module

When a pin is automatically configured as an output by a peripheral module, the pins data direction (DDR) bit is unknown. After disabling the peripheral module, the user should re-initialize the DDR bit to the desired configuration.

The other peripheral modules (which require an input) must have their data direction bit configured appropriately.

Note:	A pin that is a peripheral input, can be configured as an output (DDRx<y> is cleared). The peripheral events will be determined by the action output on the port pin.

9.1 PORTA Register

PORTA is a 6-bit wide latch. PORTA does not have a corresponding Data Direction Register (DDR).

Reading PORTA reads the status of the pins.

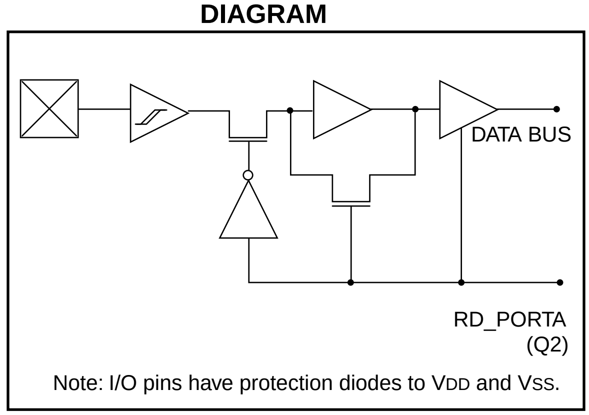
The RA1 pin is multiplexed with TMR0 clock input, and RA4 and RA5 are multiplexed with the USART functions. The control of RA4 and RA5 as outputs is automatically configured by the USART module.

9.1.1 USING RA2, RA3 AS OUTPUTS

The RA2 and RA3 pins are open drain outputs. To use the RA2 or the RA3 pin(s) as output(s), simply write to the PORTA register the desired value. A '0' will cause the pin to drive low, while a '1' will cause the pin to float (hi-impedance). An external pull-up resistor should be used to pull the pin high. Writes to PORTA will not affect the other pins.

Note:	When using the RA2 or RA3 pin(s) as output(s), read-modify-write instructions (such as BCF, BSF, BTG) on PORTA are not recommended. Such operations read the port pins, do the desired operation, and then write this value to the data latch. This may inadvertently cause the RA2 or RA3 pins to switch from input to output (or vice-versa). It is recommended to use a shadow register for PORTA. Do the bit operations on this shadow register and then move it to PORTA.

FIGURE 9-1: RA0 AND RA1 BLOCK DIAGRAM



12.2.1 ONE CAPTURE AND ONE PERIOD REGISTER MODE

In this mode registers PR3H/CA1H and PR3L/CA1L constitute a 16-bit period register. A block diagram is shown in Figure 12-7. The timer increments until it equals the period register and then resets to 0000h. TMR3 Interrupt Flag bit (TMR3IF) is set at this point. This interrupt can be disabled by clearing the TMR3 Interrupt Enable bit (TMR3IE). TMR3IF must be cleared in software.

This mode is selected if control bit CA1/$\overline{\text{PR3}}$ is clear. In this mode, the Capture1 register, consisting of high byte (PR3H/CA1H) and low byte (PR3L/CA1L), is configured as the period control register for TMR3. Capture1 is disabled in this mode, and the corresponding Interrupt bit CA1IF is never set. TMR3 increments until it equals the value in the period register and then resets to 0000h.

Capture2 is active in this mode. The CA2ED1 and CA2ED0 bits determine the event on which capture will occur. The possible events are:

- Capture on every falling edge
- Capture on every rising edge
- Capture every 4th rising edge
- Capture every 16th rising edge

When a capture takes place, an interrupt flag is latched into the CA2IF bit. This interrupt can be enabled by setting the corresponding mask bit CA2IE. The Peripheral Interrupt Enable bit (PEIE) must be set and the Global Interrupt Disable bit (GLINTD) must be cleared for the interrupt to be acknowledged. The CA2IF interrupt flag bit must be cleared in software.

When the capture prescale select is changed, the prescaler is not reset and an event may be generated. Therefore, the first capture after such a change will be ambiguous. However, it sets the time-base for the next capture. The prescaler is reset upon chip reset.

Capture pin RB1/CAP2 is a multiplexed pin. When used as a port pin, Capture2 is not disabled. However, the user can simply disable the Capture2 interrupt by clearing CA2IE. If RB1/CAP2 is used as an output pin, the user can activate a capture by writing to the port pin. This may be useful during development phase to emulate a capture interrupt.

The input on capture pin RB1/CAP2 is synchronized internally to internal phase clocks. This imposes certain restrictions on the input waveform (see the Electrical Specification section for timing).

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 Timer3 value to the capture register, protecting 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 recommended sequence to read capture registers and capture overflow flag bits is shown in Example 12-1.

EXAMPLE 12-1: SEQUENCE TO READ CAPTURE REGISTERS

```
MOVLB 3                 ;Select Bank 3
MOVPF CA2L,LO_BYTE      ;Read Capture2 low
                        ;byte, store in LO_BYTE
MOVPF CA2H,HI_BYTE      ;Read Capture2 high
                        ;byte, store in HI_BYTE
MOVPF TCON2,STAT_VAL    ;Read TCON2 into file
                        ;STAT_VAL
```

FIGURE 12-7: TIMER3 WITH ONE CAPTURE AND ONE PERIOD REGISTER BLOCK DIAGRAM

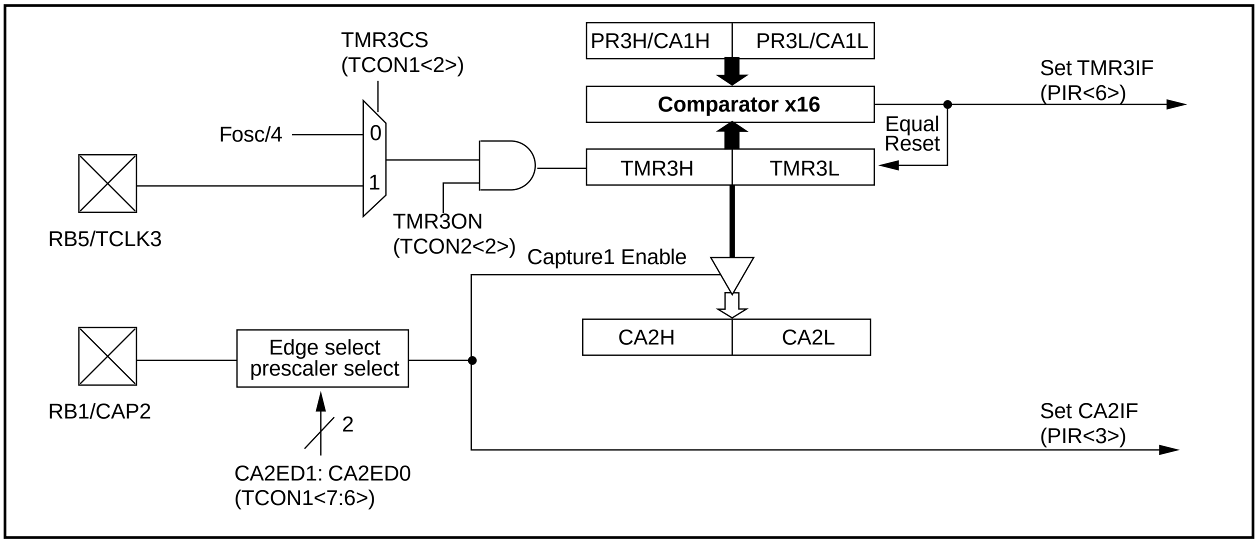


TABLE 13-4: BAUD RATES FOR ASYNCHRONOUS MODE

BAUD RATE (K)	FOSC = 33 MHz KBAUD	%ERROR	SPBRG value (decimal)	FOSC = 25 MHz KBAUD	%ERROR	SPBRG value (decimal)	FOSC = 20 MHz KBAUD	%ERROR	SPBRG value (decimal)	FOSC = 16 MHz KBAUD	%ERROR	SPBRG value (decimal)
0.3	NA	—	—	NA	—	—	NA	—	—	NA	—	—
1.2	NA	—	—	NA	—	—	1.221	+1.73	255	1.202	+0.16	207
2.4	2.398	-0.07	214	2.396	0.14	162	2.404	+0.16	129	2.404	+0.16	103
9.6	9.548	-0.54	53	9.53	-0.76	40	9.469	-1.36	32	9.615	+0.16	25
19.2	19.09	-0.54	26	19.53	+1.73	19	19.53	+1.73	15	19.23	+0.16	12
76.8	73.66	-4.09	6	78.13	+1.73	4	78.13	+1.73	3	83.33	+8.51	2
96	103.12	+7.42	4	97.65	+1.73	3	104.2	+8.51	2	NA	—	—
300	257.81	-14.06	1	390.63	+30.21	0	312.5	+4.17	0	NA	—	—
500	515.62	+3.13	0	NA	—	—	NA	—	—	NA	—	—
HIGH	515.62	—	0	—	—	0	312.5	—	0	250	—	0
LOW	2.014	—	255	1.53	—	255	1.221	—	255	0.977	—	255

BAUD RATE (K)	FOSC = 10 MHz KBAUD	%ERROR	SPBRG value (decimal)	FOSC = 7.159 MHz KBAUD	%ERROR	SPBRG value (decimal)	FOSC = 5.068 MHz KBAUD	%ERROR	SPBRG value (decimal)
0.3	NA	—	—	NA	—	—	0.31	+3.13	255
1.2	1.202	+0.16	129	1.203	_0.23	92	1.2	0	65
2.4	2.404	+0.16	64	2.380	-0.83	46	2.4	0	32
9.6	9.766	+1.73	15	9.322	-2.90	11	9.9	-3.13	7
19.2	19.53	+1.73	7	18.64	-2.90	5	19.8	+3.13	3
76.8	78.13	+1.73	1	NA	—	—	79.2	+3.13	0
96	NA	—	—	NA	—	—	NA	—	—
300	NA	—	—	NA	—	—	NA	—	—
500	NA	—	—	NA	—	—	NA	—	—
HIGH	156.3	—	0	111.9	—	0	79.2	—	0
LOW	0.610	—	255	0.437	—	255	0.309	—	255

BAUD RATE (K)	FOSC = 3.579 MHz KBAUD	%ERROR	SPBRG value (decimal)	FOSC = 1 MHz KBAUD	%ERROR	SPBRG value (decimal)	FOSC = 32.768 kHz KBAUD	%ERROR	SPBRG value (decimal)
0.3	0.301	+0.23	185	0.300	+0.16	51	0.256	-14.67	1
1.2	1.190	-0.83	46	1.202	+0.16	12	NA	—	—
2.4	2.432	+1.32	22	2.232	-6.99	6	NA	—	—
9.6	9.322	-2.90	5	NA	—	—	NA	—	—
19.2	18.64	-2.90	2	NA	—	—	NA	—	—
76.8	NA	—	—	NA	—	—	NA	—	—
96	NA	—	—	NA	—	—	NA	—	—
300	NA	—	—	NA	—	—	NA	—	—
500	NA	—	—	NA	—	—	NA	—	—
HIGH	55.93	—	0	15.63	—	0	0.512	—	0
LOW	0.218	—	255	0.061	—	255	0.002	—	255

DCFSNZ — Decrement f, skip if not 0

Syntax:	[*label*] DCFSNZ f,d
Operands:	$0 \le f \le 255$ $d \in [0,1]$
Operation:	(f) – 1 → (dest); skip if not 0
Status Affected:	None
Encoding:	`0010` `011d` `ffff` `ffff`
Description:	The contents of register 'f' are decremented. 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     DCFSNZ  TEMP, 1
ZERO     :
NZERO    :
```

Before Instruction
TEMP_VALUE = ?

After Instruction
TEMP_VALUE = TEMP_VALUE - 1,
If TEMP_VALUE = 0;
PC = Address (`ZERO`)
If TEMP_VALUE ≠ 0;
PC = Address (`NZERO`)

GOTO — Unconditional Branch

Syntax:	[*label*] GOTO k
Operands:	$0 \le k \le 8191$
Operation:	k → PC<12:0>; k<12:8> → PCLATH<4:0>, PC<15:13> → PCLATH<7:5>
Status Affected:	None
Encoding:	`110k` `kkkk` `kkkk` `kkkk`
Description:	`GOTO` allows an unconditional branch anywhere within an 8K page boundary. The thirteen bit immediate value is loaded into PC bits <12:0>. Then the upper eight bits of PC are loaded into PCLATH. `GOTO` is always a two-cycle instruction.
Words:	1
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>	Execute	NOP
Forced NOP	NOP	Execute	NOP

Example: `GOTO THERE`

After Instruction
PC = Address (`THERE`)

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

FIGURE 17-5: TIMER0 CLOCK TIMINGS

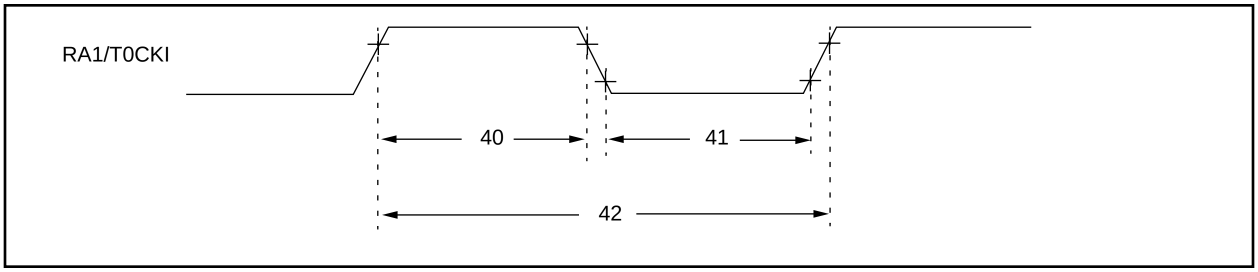


TABLE 17-5: TIMER0 CLOCK REQUIREMENTS

Parameter No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
40	Tt0H	T0CKI High Pulse Width	No Prescaler	$0.5T_{CY} + 20$ §	—	—	ns	
			With Prescaler	10*	—	—	ns	
41	Tt0L	T0CKI Low Pulse Width	No Prescaler	$0.5T_{CY} + 20$ §	—	—	ns	
			With Prescaler	10*	—	—	ns	
42	Tt0P	T0CKI Period		$\frac{T_{CY} + 40}{N}$ §	—	—	ns	N = prescale value (1, 2, 4, ..., 256)

* 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 17-6: TIMER1, TIMER2, AND TIMER3 CLOCK TIMINGS

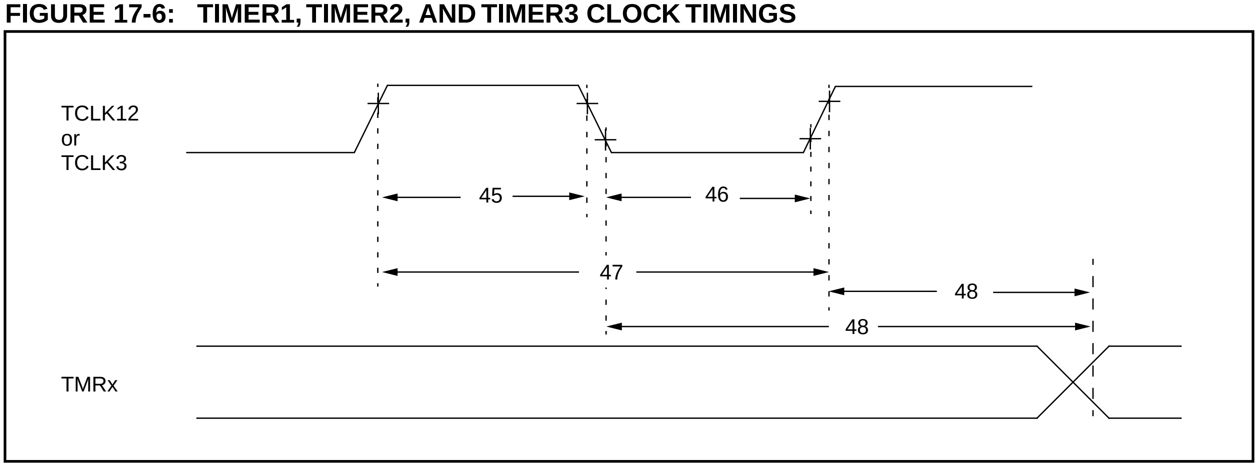


TABLE 17-6: TIMER1, TIMER2, AND TIMER3 CLOCK REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ †	Max	Units	Conditions
45	Tt123H	TCLK12 and TCLK3 high time	$0.5\ T_{CY} + 20$ §	—	—	ns	
46	Tt123L	TCLK12 and TCLK3 low time	$0.5\ T_{CY} + 20$ §	—	—	ns	
47	Tt123P	TCLK12 and TCLK3 input period	$\frac{T_{CY} + 40}{N}$ §	—	—	ns	N = prescale value (1, 2, 4, 8)
48	TckE2tmrI	Delay from selected External Clock Edge to Timer increment	$2T_{OSC}$ §	—	6 Tosc §	—	

* 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 19-9: USART MODULE: SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING

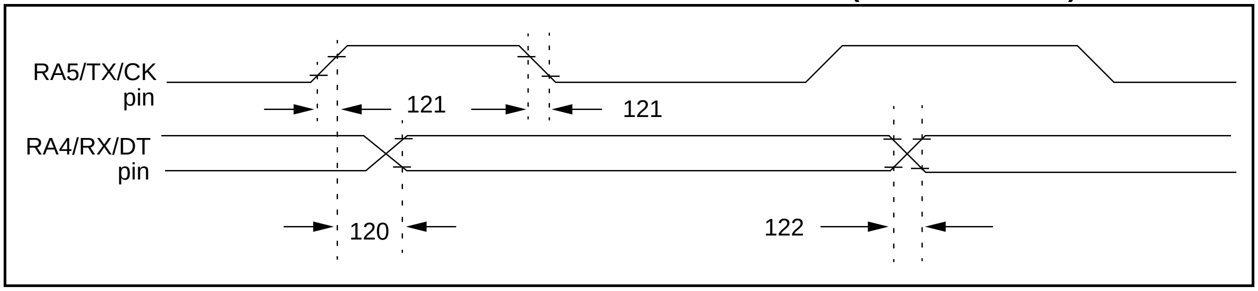


TABLE 19-9: SYNCHRONOUS TRANSMISSION REQUIREMENTS

Param No.	Sym	Characteristic		Min	Typ†	Max	Units	Conditions
120	TckH2dtV	SYNC XMIT (MASTER & SLAVE) Clock high to data out valid	PIC17CR42/42A/43/R43/44	—	—	50	ns	
			PIC17LCR42/42A/43/R43/44	—	—	75	ns	
121	TckRF	Clock out rise time and fall time (Master Mode)	PIC17CR42/42A/43/R43/44	—	—	25	ns	
			PIC17LCR42/42A/43/R43/44	—	—	40	ns	
122	TdtRF	Data out rise time and fall time	PIC17CR42/42A/43/R43/44	—	—	25	ns	
			PIC17LCR42/42A/43/R43/44	—	—	40	ns	

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

FIGURE 19-10: USART MODULE: SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING

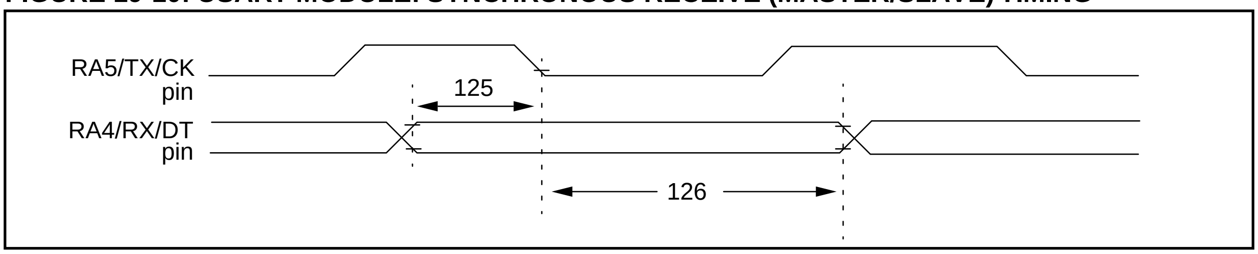


TABLE 19-10: SYNCHRONOUS RECEIVE REQUIREMENTS

Parameter No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions
125	TdtV2ckL	SYNC RCV (MASTER & SLAVE) Data hold before CK↓ (DT hold time)	15	—	—	ns	
126	TckL2dtl	Data hold after CK↓ (DT hold time)	15	—	—	ns	

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

Applicable Devices	42	R42	42A	43	R43	44

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

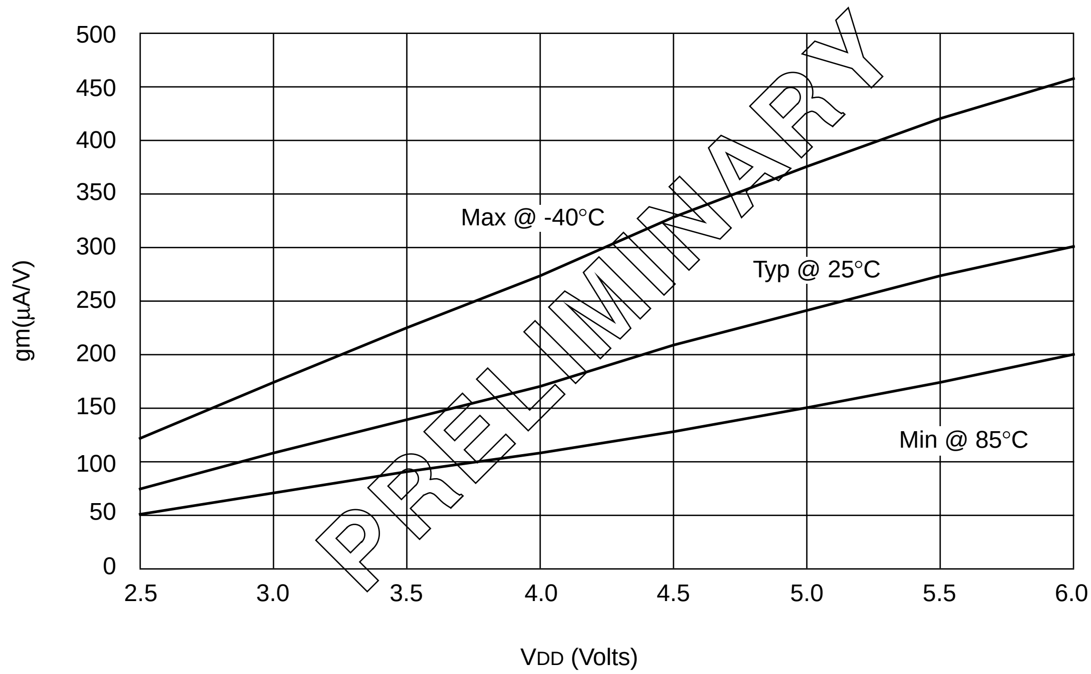
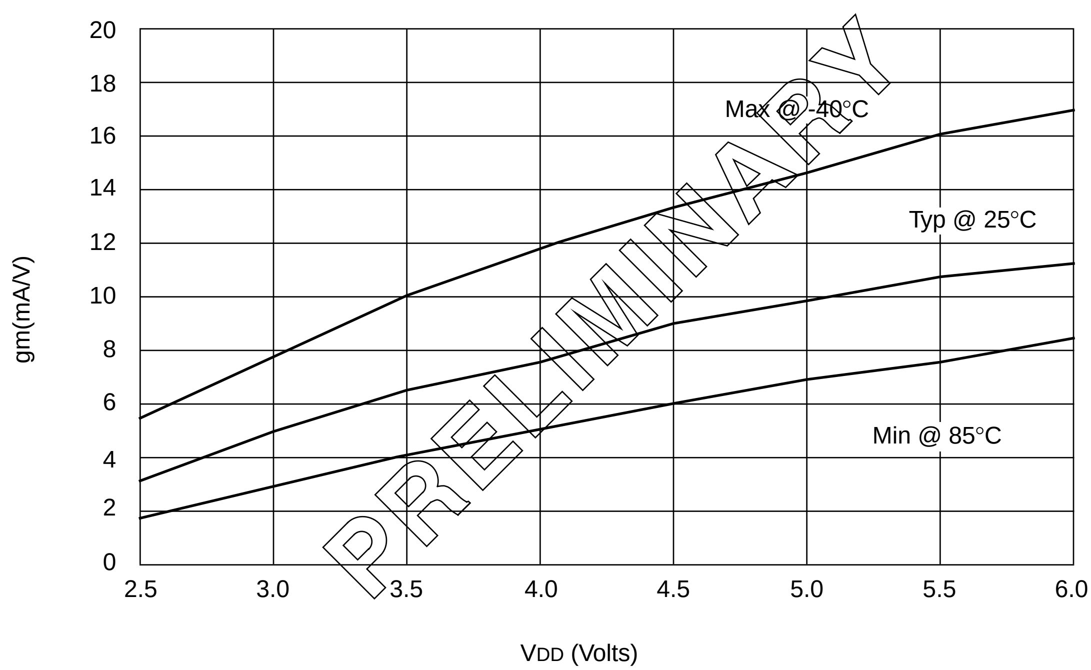


FIGURE 20-6: TRANSCONDUCTANCE (gm) OF XT OSCILLATOR vs. VDD



Applicable Devices	42	R42	42A	43	R43	44

FIGURE 20-19: VTH, VIL of I/O PINS (SCHMITT TRIGGER) VS. VDD

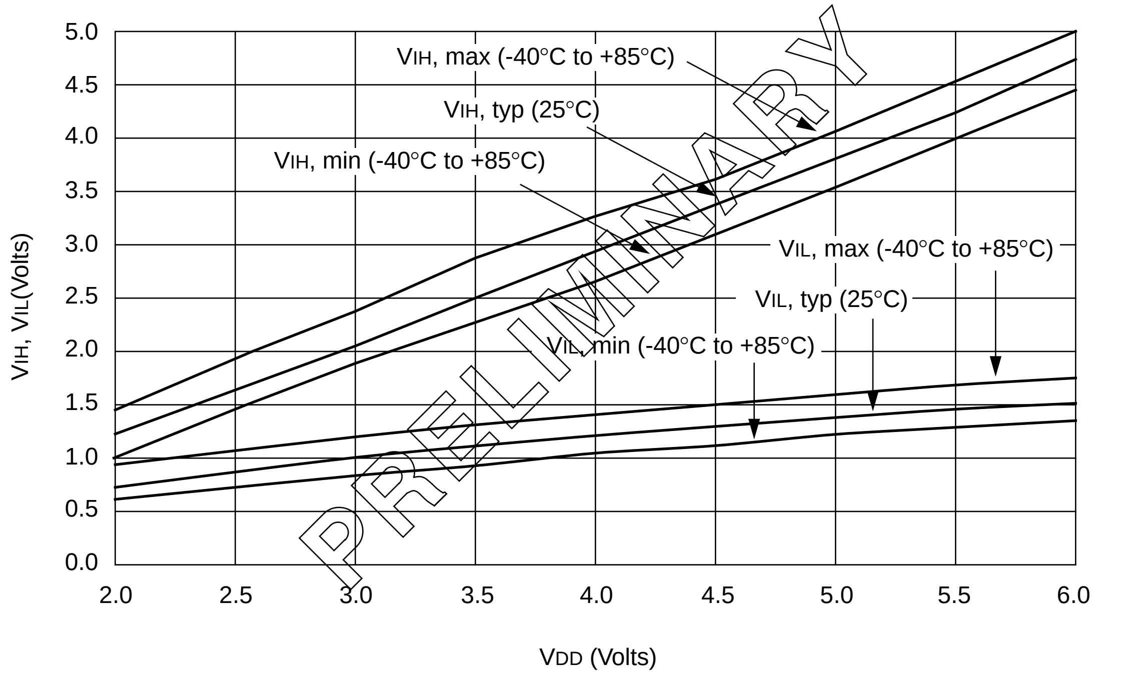
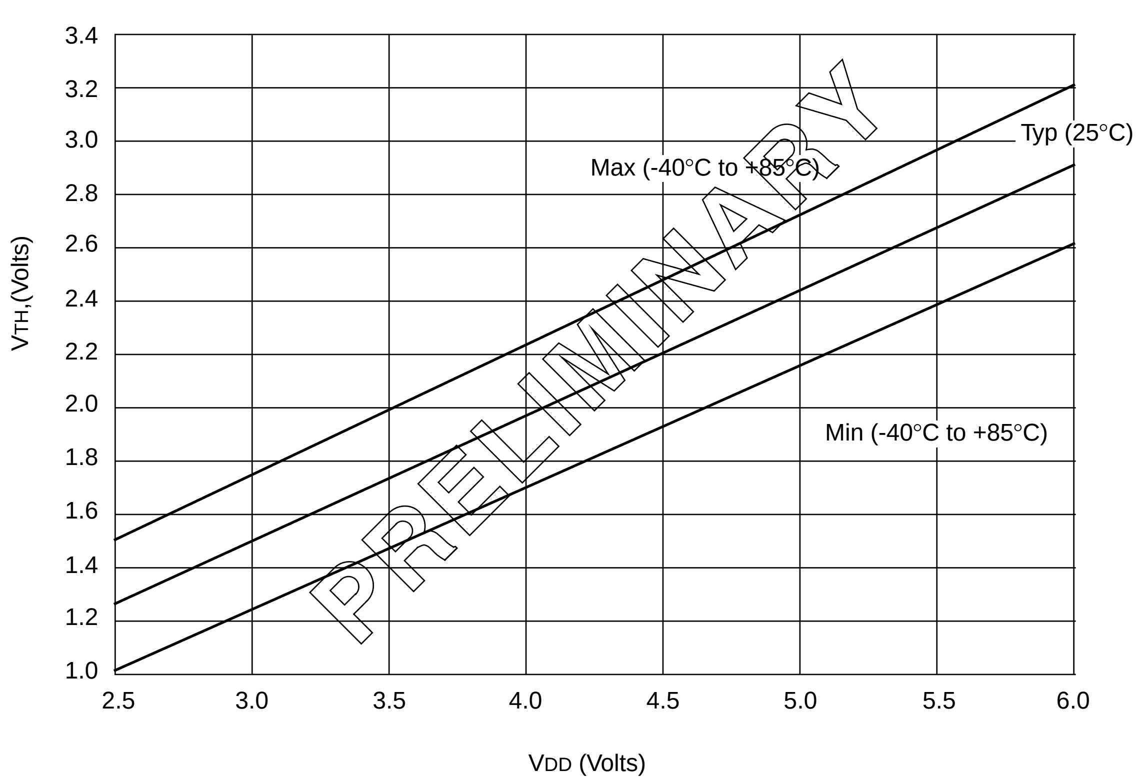
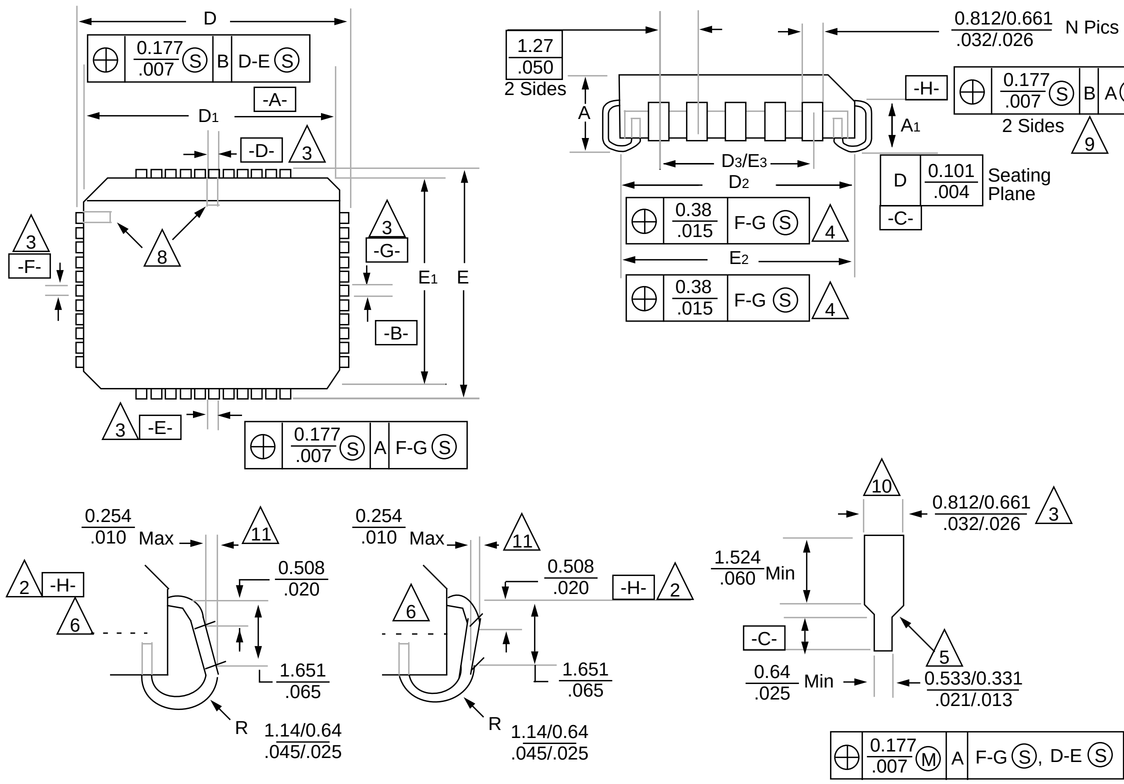


FIGURE 20-20: VTH (INPUT THRESHOLD VOLTAGE) OF OSC1 INPUT (IN XT AND LF MODES) vs. VDD



21.3 44-Lead Plastic Leaded Chip Carrier (Square)



Package Group: Plastic Leaded Chip Carrier (PLCC)						
	Millimeters			Inches		
Symbol	**Min**	**Max**	**Notes**	**Min**	**Max**	**Notes**
A	4.191	4.572		0.165	0.180	
A1	2.413	2.921		0.095	0.115	
D	17.399	17.653		0.685	0.695	
D1	16.510	16.663		0.650	0.656	
D2	15.494	16.002		0.610	0.630	
D3	12.700	12.700	**Reference**	0.500	0.500	**Reference**
E	17.399	17.653		0.685	0.695	
E1	16.510	16.663		0.650	0.656	
E2	15.494	16.002		0.610	0.630	
E3	12.700	12.700	**Reference**	0.500	0.500	**Reference**
N	44	44		44	44	
CP	–	0.102		–	0.004	
LT	0.203	0.381		0.008	0.015	

E.8 PIC17CXX Family of Devices

	Clock	Memory			Peripherals					Features					
	Maximum Frequency of Operation (MHz)	Program Memory (Words): EPROM	Program Memory (Words): ROM	RAM Data Memory (bytes)	Timer Module(s)	Captures	PWMs	Serial Port(s) (USART)	Hardware Multiply	External Interrupts	Interrupt Sources	I/O Pins	Voltage Range (Volts)	Number of Instructions	Packages
PIC17C42	25	2K	—	232	TMR0,TMR1, TMR2,TMR3	2	2	Yes	—	Yes	11	33	4.5-5.5	55	40-pin DIP; 44-pin PLCC, MQFP
PIC17C42A	25	2K	—	232	TMR0,TMR1, TMR2,TMR3	2	2	Yes	Yes	Yes	11	33	2.5-6.0	58	40-pin DIP; 44-pin PLCC, TQFP, MQFP
PIC17CR42	25	—	2K	232	TMR0,TMR1, TMR2,TMR3	2	2	Yes	Yes	Yes	11	33	2.5-6.0	58	40-pin DIP; 44-pin PLCC, TQFP, MQFP
PIC17C43	25	4K	—	454	TMR0,TMR1, TMR2,TMR3	2	2	Yes	Yes	Yes	11	33	2.5-6.0	58	40-pin DIP; 44-pin PLCC, TQFP, MQFP
PIC17CR43	25	—	4K	454	TMR0,TMR1, TMR2,TMR3	2	2	Yes	Yes	Yes	11	33	2.5-6.0	58	40-pin DIP; 44-pin PLCC, TQFP, MQFP
PIC17C44	25	8K		454	TMR0,TMR1, TMR2,TMR3	2	2	Yes	Yes	Yes	11	33	2.5-6.0	58	40-pin DIP; 44-pin PLCC, TQFP, MQFP

All PIC16/17 Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability.

PIN COMPATIBILITY

Devices that have the same package type and VDD, VSS and $\overline{MCLR}$ pin locations are said to be pin compatible. This allows these different devices to operate in the same socket. Compatible devices may only requires minor software modification to allow proper operation in the application socket (ex., PIC16C56 and PIC16C61 devices). Not all devices in the same package size are pin compatible; for example, the PIC16C62 is compatible with the PIC16C63, but not the PIC16C55.

Pin compatibility does not mean that the devices offer the same features. As an example, the PIC16C54 is pin compatible with the PIC16C71, but does not have an A/D converter, weak pull-ups on PORTB, or interrupts.

TABLE E-1: PIN COMPATIBLE DEVICES

Pin Compatible Devices	Package
PIC12C508, PIC12C509	8-pin
PIC16C54, PIC16C54A, PIC16CR54A, PIC16C56, PIC16C58A, PIC16CR58A, PIC16C61, PIC16C554, PIC16C556, PIC16C558 PIC16C620, PIC16C621, PIC16C622, PIC16C710, PIC16C71, PIC16C711, PIC16F83, PIC16CR83, PIC16C84, PIC16F84A, PIC16CR84	18-pin 20-pin
PIC16C55, PIC16C57, PIC16CR57B	28-pin
PIC16C62, PIC16CR62, PIC16C62A, PIC16C63, PIC16C72, PIC16C73, PIC16C73A	28-pin
PIC16C64, PIC16CR64, PIC16C64A, PIC16C65, PIC16C65A, PIC16C74, PIC16C74A	40-pin
PIC17C42, PIC17CR42, PIC17C42A, PIC17C43, PIC17CR43, PIC17C44	40-pin
PIC16C923, PIC16C924	64/68-pin

U

W

