Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

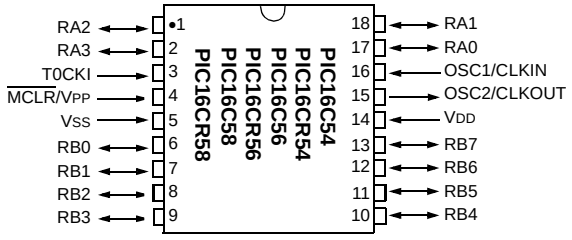
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded"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

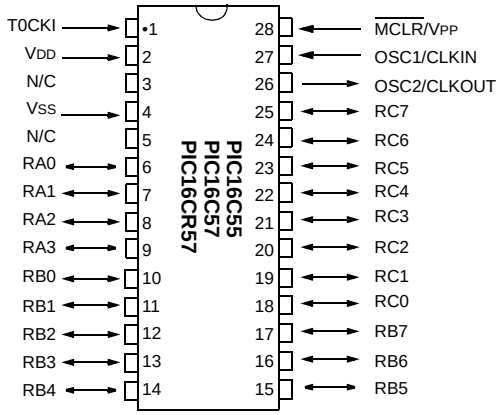
Details	
Product Status	Active
Core Processor	PIC
Core Size	8-Bit
Speed	20MHz
Connectivity	-
Peripherals	POR, WDT
Number of I/O	20
Program Memory Size	768B (512 x 12)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	25 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	-
Oscillator Type	External
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	28-SSOP (0.209", 5.30mm Width)
Supplier Device Package	28-SSOP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic16c55at-20i-ss

Pin Diagrams

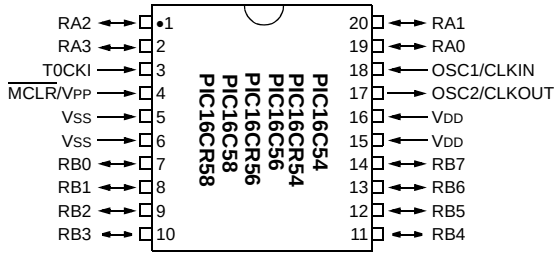
PDIP, SOIC, Windowed CERDIP



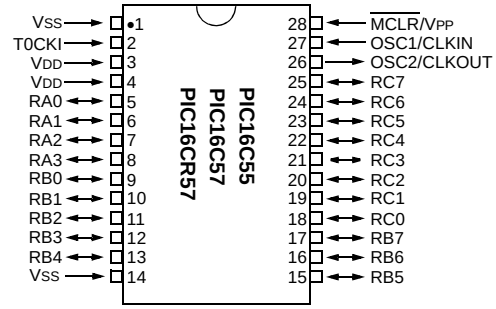
PDIP, SOIC, Windowed CERDIP



SSOP



SSOP



Device Differences

Device	Voltage Range	Oscillator Selection (Program)	Oscillator	Process Technology (Microns)	ROM Equivalent	$\overline{\text{MCLR}}$ Filter
PIC16C54	2.5-6.25	Factory	See **Note 1**	1.2	PIC16CR54A	No
PIC16C54A	2.0-6.25	User	See **Note 1**	0.9	—	No
PIC16C54C	2.5-5.5	User	See **Note 1**	0.7	PIC16CR54C	Yes
PIC16C55	2.5-6.25	Factory	See **Note 1**	1.7	—	No
PIC16C55A	2.5-5.5	User	See **Note 1**	0.7	—	Yes
PIC16C56	2.5-6.25	Factory	See **Note 1**	1.7	—	No
PIC16C56A	2.5-5.5	User	See **Note 1**	0.7	PIC16CR56A	Yes
PIC16C57	2.5-6.25	Factory	See **Note 1**	1.2	—	No
PIC16C57C	2.5-5.5	User	See **Note 1**	0.7	PIC16CR57C	Yes
PIC16C58B	2.5-5.5	User	See **Note 1**	0.7	PIC16CR58B	Yes
PIC16CR54A	2.5-6.25	Factory	See **Note 1**	1.2	N/A	Yes
PIC16CR54C	2.5-5.5	Factory	See **Note 1**	0.7	N/A	Yes
PIC16CR56A	2.5-5.5	Factory	See **Note 1**	0.7	N/A	Yes
PIC16CR57C	2.5-5.5	Factory	See **Note 1**	0.7	N/A	Yes
PIC16CR58B	2.5-5.5	Factory	See **Note 1**	0.7	N/A	Yes

Note 1: If you change from this device to another device, please verify oscillator characteristics in your application.

Note: The table shown above shows the generic names of the PIC16C5X devices. For device varieties, please refer to Section 2.0.

NOTES:

6.5 Program Counter

As a program instruction is executed, the Program Counter (PC) will contain the address of the next program instruction to be executed. The PC value is increased by one, every instruction cycle, unless an instruction changes the PC.

For a GOTO instruction, bits 8:0 of the PC are provided by the GOTO instruction word. The PC Latch (PCL) is mapped to PC<7:0> (Figure 6-7, Figure 6-8 and Figure 6-9).

For the PIC16C56, PIC16CR56, PIC16C57, PIC16CR57, PIC16C58 and PIC16CR58, a page number must be supplied as well. Bit5 and bit6 of the STATUS Register provide page information to bit9 and bit10 of the PC (Figure 6-8 and Figure 6-9).

For a CALL instruction, or any instruction where the PCL is the destination, bits 7:0 of the PC again are provided by the instruction word. However, PC<8> does not come from the instruction word, but is always cleared (Figure 6-7 and Figure 6-8).

Instructions where the PCL is the destination, or modify PCL instructions, include MOVWF PCL, ADDWF PCL, and BSF PCL,5.

For the PIC16C56, PIC16CR56, PIC16C57, PIC16CR57, PIC16C58 and PIC16CR58, a page number again must be supplied. Bit5 and bit6 of the STATUS Register provide page information to bit9 and bit10 of the PC (Figure 6-8 and Figure 6-9).

Note:	Because PC<8> is cleared in the CALL instruction, or any modify PCL instruction, all subroutine calls or computed jumps are limited to the first 256 locations of any program memory page (512 words long).

FIGURE 6-7: LOADING OF PC BRANCH INSTRUCTIONS - PIC16C54, PIC16CR54, PIC16CR55

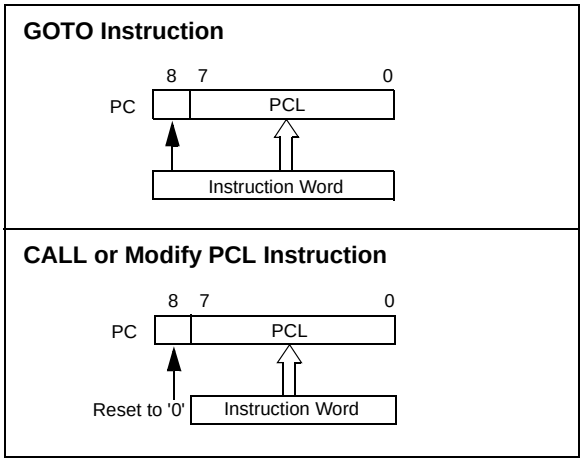


FIGURE 6-8: LOADING OF PC BRANCH INSTRUCTIONS - PIC16C56/PIC16CR56

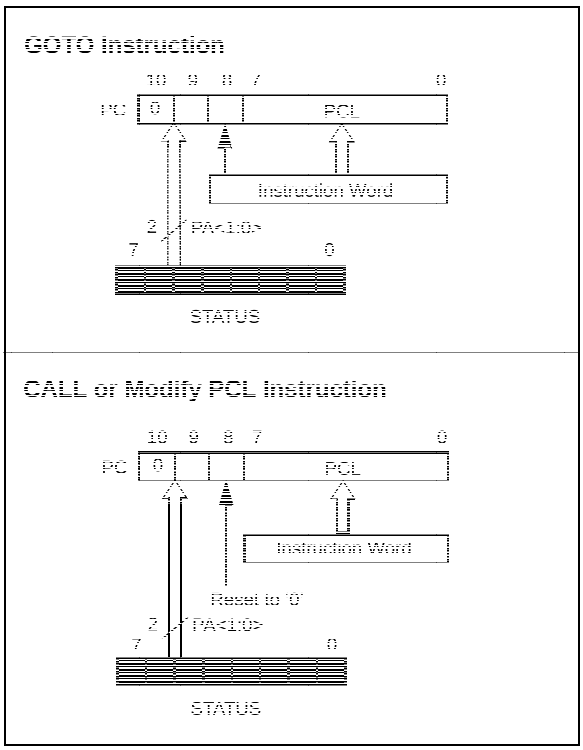
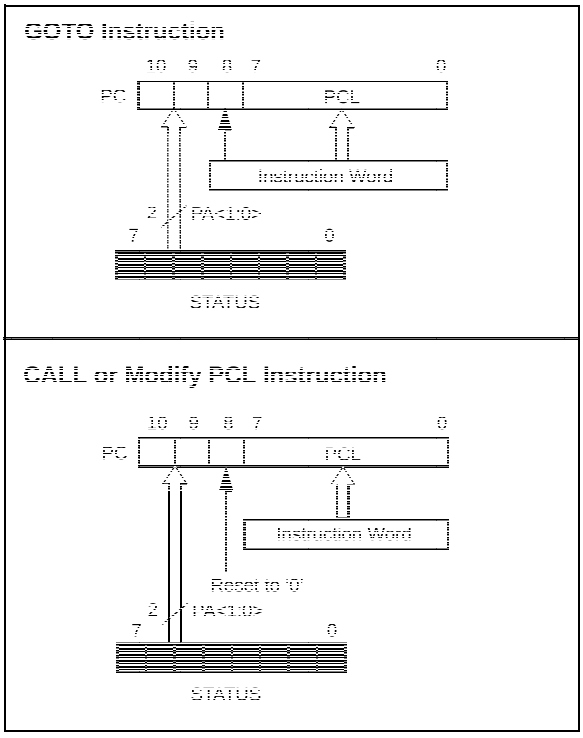


FIGURE 6-9: LOADING OF PC BRANCH INSTRUCTIONS - PIC16C57/PIC16CR57, AND PIC16C58/ PIC16CR58



6.7 Indirect Data Addressing; INDF and FSR Registers

The INDF Register is not a physical register. Addressing INDF actually addresses the register whose address is contained in the FSR Register (FSR is a *pointer*). This is indirect addressing.

EXAMPLE 6-1: INDIRECT ADDRESSING

- Register file 08 contains the value 10h
- Register file 09 contains the value 0Ah
- Load the value 08 into the FSR Register
- A read of the INDF Register will return the value of 10h
- Increment the value of the FSR Register by one (FSR = 09h)
- A read of the INDF register now will return the value of 0Ah.

Reading INDF itself indirectly (FSR = 0) will produce 00h. Writing to the INDF Register indirectly results in a no-operation (although STATUS bits may be affected).

A simple program to clear RAM locations 10h-1Fh using indirect addressing is shown in Example 6-2.

EXAMPLE 6-2: HOW TO CLEAR RAM USING INDIRECT ADDRESSING

```
            MOVLW   H'10'   ;initialize pointer
            MOVWF   FSR     ; to RAM
NEXT        CLRF    INDF    ;clear INDF Register
            INCF    FSR,F   ;inc pointer
            BTFSC   FSR,4   ;all done?
            GOTO    NEXT    ;NO, clear next
CONTINUE
            :               ;YES, continue
```

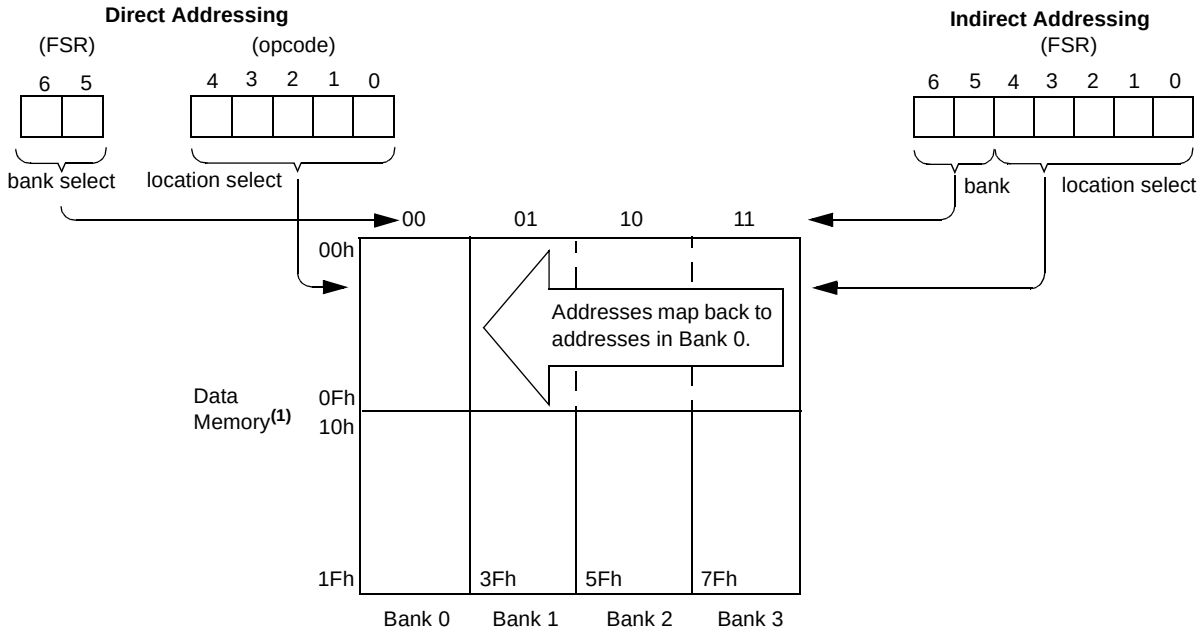
The FSR is either a 5-bit (PIC16C54, PIC16CR54, PIC16C55, PIC16C56, PIC16CR56) or 7-bit (PIC16C57, PIC16CR57, PIC16C58, PIC16CR58) wide register. It is used in conjunction with the INDF Register to indirectly address the data memory area.

The FSR<4:0> bits are used to select data memory addresses 00h to 1Fh.

PIC16C54, PIC16CR54, PIC16C55, PIC16C56, PIC16CR56: These do not use banking. FSR<6:5> bits are unimplemented and read as '1's.

PIC16C57, PIC16CR57, PIC16C58, PIC16CR58: FSR<6:5> are the bank select bits and are used to select the bank to be addressed (`00` = bank 0, `01` = bank 1, `10` = bank 2, `11` = bank 3).

FIGURE 6-10: DIRECT/INDIRECT ADDRESSING



Note 1: For register map detail see Section 6.2.

8.1 Using Timer0 with an External Clock

When an external clock input is used for Timer0, it must meet certain requirements. The external clock requirement is due to internal phase clock (TOSC) synchronization. Also, there is a delay in the actual incrementing of Timer0 after synchronization.

8.1.1 EXTERNAL CLOCK SYNCHRONIZATION

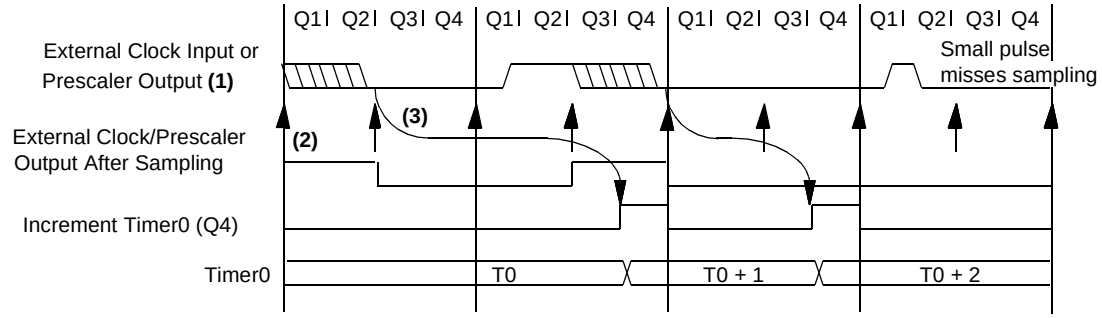
When no prescaler is used, the external clock input is the same as the prescaler output. The synchronization of T0CKI with the internal phase clocks is accomplished by sampling the prescaler output on the Q2 and Q4 cycles of the internal phase clocks (Figure 8-5). Therefore, it is necessary for T0CKI to be high for at least 2TOSC (and a small RC delay of 20 ns) and low for at least 2TOSC (and a small RC delay of 20 ns). Refer to the electrical specification of the desired device.

When a prescaler is used, the external clock input is divided by the asynchronous ripple counter-type prescaler so that the prescaler output is symmetrical. For the external clock to meet the sampling requirement, the ripple counter must be taken into account. Therefore, it is necessary for T0CKI to have a period of at least 4TOSC (and a small RC delay of 40 ns) divided by the prescaler value. The only requirement on T0CKI high and low time is that they do not violate the minimum pulse width requirement of 10 ns. Refer to parameters 40, 41 and 42 in the electrical specification of the desired device.

8.1.2 TIMER0 INCREMENT DELAY

Since the prescaler output is synchronized with the internal clocks, there is a small delay from the time the external clock edge occurs to the time the Timer0 module is actually incremented. Figure 8-5 shows the delay from the external clock edge to the timer incrementing.

FIGURE 8-5: TIMER0 TIMING WITH EXTERNAL CLOCK



Note 1: External clock if no prescaler selected, prescaler output otherwise.
2: The arrows indicate the points in time where sampling occurs.
3: Delay from clock input change to Timer0 increment is 3Tosc to 7Tosc (duration of Q = Tosc). Therefore, the error in measuring the interval between two edges on Timer0 input = ± 4Tosc max.

9.1 Configuration Bits

Configuration bits can be programmed to select various device configurations. Two bits are for the selection of the oscillator type and one bit is the Watchdog Timer enable bit. Nine bits are code protection bits for the PIC16C54A, PIC16CR54A, PIC16C54C, PIC16CR54C, PIC16C55A, PIC16C56A, PIC16CR56A, PIC16C57C, PIC16CR57C, PIC16C58B, and PIC16CR58B devices (Register 9-1). One bit is for code protection for the PIC16C54, PIC16C55, PIC16C56 and PIC16C57 devices (Register 9-2).

QTP or ROM devices have the oscillator configuration programmed at the factory and these parts are tested accordingly (see "Product Identification System" diagrams in the back of this data sheet).

REGISTER 9-1: CONFIGURATION WORD FOR PIC16C54A/CR54A/C54C/CR54C/C55A/C56A/ CR56A/C57C/CR57C/C58B/CR58B

CP	CP	CP	CP	CP	CP	CP	CP	CP	WDTE	FOSC1	FOSC0
bit 11											bit 0

bit 11-3: **CP**: Code Protection Bit
1 = Code protection off
0 = Code protection on

bit 2: **WDTE**: Watchdog timer enable bit
1 = WDT enabled
0 = WDT disabled

bit 1-0: **FOSC1:FOSC0**: Oscillator Selection Bit
00 = LP oscillator
01 = XT oscillator
10 = HS oscillator
11 = RC oscillator

Note 1: Refer to the PIC16C5X Programming Specification (Literature Number DS30190) to determine how to access the configuration word.

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

ADDWF **Add W and f**

Syntax: [*label*] ADDWF f,d

Operands: $0 \le f \le 31$
$d \in [0,1]$

Operation: (W) + (f) → (dest)

Status Affected: C, DC, Z

Encoding:

0001	11df	ffff

Description: Add the contents of the W register and register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is '1' the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example:
```
ADDWF  TEMP_REG, 0
```
Before Instruction
W = 0x17
TEMP_REG = 0xC2
After Instruction
W = 0xD9
TEMP_REG = 0xC2

ANDWF **AND W with f**

Syntax: [*label*] ANDWF f,d

Operands: $0 \le f \le 31$
$d \in [0,1]$

Operation: (W) .AND. (f) → (dest)

Status Affected: Z

Encoding:

0001	01df	ffff

Description: The contents of the W register are AND'ed with register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is '1' the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example:
```
ANDWF  TEMP_REG, 1
```
Before Instruction
W = 0x17
TEMP_REG = 0xC2
After Instruction
W = 0x17
TEMP_REG = 0x02

ANDLW **AND literal with W**

Syntax: [*label*] ANDLW k

Operands: $0 \le k \le 255$

Operation: (W).AND. (k) → (W)

Status Affected: Z

Encoding:

1110	kkkk	kkkk

Description: The contents of the W register are AND'ed with the eight-bit literal 'k'. The result is placed in the W register.

Words: 1

Cycles: 1

Example:
```
ANDLW  H'5F'
```
Before Instruction
W = 0xA3
After Instruction
W = 0x03

BCF **Bit Clear f**

Syntax: [*label*] BCF f,b

Operands: $0 \le f \le 31$
$0 \le b \le 7$

Operation: 0 → (f<b>)

Status Affected: None

Encoding:

0100	bbbf	ffff

Description: Bit 'b' in register 'f' is cleared.

Words: 1

Cycles: 1

Example:
```
BCF    FLAG_REG,  7
```
Before Instruction
FLAG_REG = 0xC7
After Instruction
FLAG_REG = 0x47

XORLW **Exclusive OR literal with W**

Syntax: [*label*] XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → (W)

Status Affected: Z

Encoding:

1111	kkkk	kkkk

Description: The contents of the W register are XOR'ed with the eight bit literal 'k'. The result is placed in the W register.

Words: 1

Cycles: 1

Example:
```
XORLW  0xAF
```
Before Instruction
W = 0xB5
After Instruction
W = 0x1A

XORWF **Exclusive OR W with f**

Syntax: [*label*] XORWF f,d

Operands: $0 \le f \le 31$
$d \in [0,1]$

Operation: (W) .XOR. (f) → (dest)

Status Affected: Z

Encoding:

0001	10df	ffff

Description: Exclusive OR the contents of the W register with register 'f'. If 'd' is 0 the result is stored in the W register. If 'd' is 1 the result is stored back in register 'f'.

Words: 1

Cycles: 1

Example
```
XORWF  REG,1
```
Before Instruction
REG = 0xAF
W = 0xB5
After Instruction
REG = 0x1A
W = 0xB5

11.8 MPLAB ICD In-Circuit Debugger

Microchip's In-Circuit Debugger, MPLAB ICD, is a powerful, low cost, run-time development tool. This tool is based on the FLASH PIC MCUs and can be used to develop for this and other PIC microcontrollers. The MPLAB ICD utilizes the in-circuit debugging capability built into the FLASH devices. This feature, along with Microchip's In-Circuit Serial Programming™ protocol, offers cost-effective in-circuit FLASH debugging from the graphical user interface of the MPLAB Integrated Development Environment. This enables a designer to develop and debug source code by watching variables, single-stepping and setting break points. Running at full speed enables testing hardware in real-time.

11.9 PRO MATE II Universal Device Programmer

The PRO MATE II universal device programmer is a full-featured programmer, capable of operating in Stand-alone mode, as well as PC-hosted mode. The PRO MATE II device programmer is CE compliant.

The PRO MATE II device programmer has programmable V_{DD} and V_{PP} supplies, which allow it to verify programmed memory at V_{DD} min and V_{DD} max for maximum reliability. It has an LCD display for instructions and error messages, keys to enter commands and a modular detachable socket assembly to support various package types. In Stand-alone mode, the PRO MATE II device programmer can read, verify, or program PIC devices. It can also set code protection in this mode.

11.10 PICSTART Plus Entry Level Development Programmer

The PICSTART Plus development programmer is an easy-to-use, low cost, prototype programmer. It connects to the PC via a COM (RS-232) port. MPLAB Integrated Development Environment software makes using the programmer simple and efficient.

The PICSTART Plus development programmer supports all PIC devices with up to 40 pins. Larger pin count devices, such as the PIC16C92X and PIC17C76X, may be supported with an adapter socket. The PICSTART Plus development programmer is CE compliant.

11.11 PICDEM 1 Low Cost PIC MCU Demonstration Board

The PICDEM 1 demonstration board is a simple board which demonstrates the capabilities of several of Microchip's microcontrollers. The microcontrollers supported are: PIC16C5X (PIC16C54 to PIC16C58A), PIC16C61, PIC16C62X, PIC16C71, PIC16C8X, PIC17C42, PIC17C43 and PIC17C44. All necessary hardware and software is included to run basic demo programs. The user can program the sample microcontrollers provided with the PICDEM 1 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer, and easily test firmware. The user can also connect the PICDEM 1 demonstration board to the MPLAB ICE in-circuit emulator and download the firmware to the emulator for testing. A prototype area is available for the user to build some additional hardware and connect it to the microcontroller socket(s). Some of the features include an RS-232 interface, a potentiometer for simulated analog input, push button switches and eight LEDs connected to PORTB.

11.12 PICDEM 2 Low Cost PIC16CXX Demonstration Board

The PICDEM 2 demonstration board is a simple demonstration board that supports the PIC16C62, PIC16C64, PIC16C65, PIC16C73 and PIC16C74 microcontrollers. All the necessary hardware and software is included to run the basic demonstration programs. The user can program the sample microcontrollers provided with the PICDEM 2 demonstration board on a PRO MATE II device programmer, or a PICSTART Plus development programmer, and easily test firmware. The MPLAB ICE in-circuit emulator may also be used with the PICDEM 2 demonstration board to test firmware. A prototype area has been provided to the user for adding additional hardware and connecting it to the microcontroller socket(s). Some of the features include a RS-232 interface, push button switches, a potentiometer for simulated analog input, a serial EEPROM to demonstrate usage of the I^2C™ bus and separate headers for connection to an LCD module and a keypad.

13.3 DC Characteristics: PIC16CR54A-04, 10, 20, PIC16LCR54A-04 (Commercial) PIC16CR54A-04I, 10I, 20I, PIC16LCR54A-04I (Industrial)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise specified) Operating Temperature 0°C ≤ TA ≤ +70°C for commercial –40°C ≤ TA ≤ +85°C for industrial				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
D030	VIL	**Input Low Voltage**					
		I/O ports	VSS	—	0.2 VDD	V	Pin at hi-impedance
		MCLR (Schmitt Trigger)	VSS	—	0.15 VDD	V	
		T0CKI (Schmitt Trigger)	VSS	—	0.15 VDD	V	
		OSC1 (Schmitt Trigger)	VSS	—	0.15 VDD	V	RC mode only(3)
		OSC1	VSS	—	0.15 VDD	V	XT, HS and LP modes
D040	VIH	**Input High Voltage**					
		I/O ports	2.0	—	VDD	V	VDD = 3.0V to 5.5V(4)
		I/O ports	0.6 VDD	—	VDD	V	Full VDD range(4)
		MCLR (Schmitt Trigger)	0.85 VDD	—	VDD	V	
		T0CKI (Schmitt Trigger)	0.85 VDD	—	VDD	V	
		OSC1 (Schmitt Trigger)	0.85 VDD	—	VDD	V	RC mode only(3)
		OSC1	0.85 VDD	—	VDD	V	XT, HS and LP modes
D050	VHYS	**Hysteresis of Schmitt Trigger inputs**	0.15 VDD*	—	—	V	
D060	IIL	**Input Leakage Current(1,2)**					**For VDD ≤ 5.5V:**
		I/O ports	–1.0	—	+1.0	µA	VSS ≤ VPIN ≤ VDD, pin at hi-impedance
		MCLR	–5.0	—	—	µA	VPIN = VSS + 0.25V
		MCLR	—	0.5	+5.0	µA	VPIN = VDD
		T0CKI	–3.0	0.5	+3.0	µA	VSS ≤ VPIN ≤ VDD
		OSC1	–3.0	0.5	+3.0	µA	VSS ≤ VPIN ≤ VDD, XT, HS and LP modes
D080	VOL	**Output Low Voltage**					
		I/O ports	—	—	0.5	V	IOL = 10 mA, VDD = 6.0V
		OSC2/CLKOUT	—	—	0.5	V	IOL = 1.9 mA, VDD = 6.0V, RC mode only
D090	VOH	**Output High Voltage(2)**					
		I/O ports	VDD – 0.5	—	—	V	IOH = –4.0 mA, VDD = 6.0V
		OSC2/CLKOUT	VDD – 0.5	—	—	V	IOH = –0.8 mA, VDD = 6.0V, RC mode only

* These parameters are characterized but not tested.

† Data in the Typical ("Typ") column is based on characterization results at 25°C. This data is for design guidance only and is not tested.

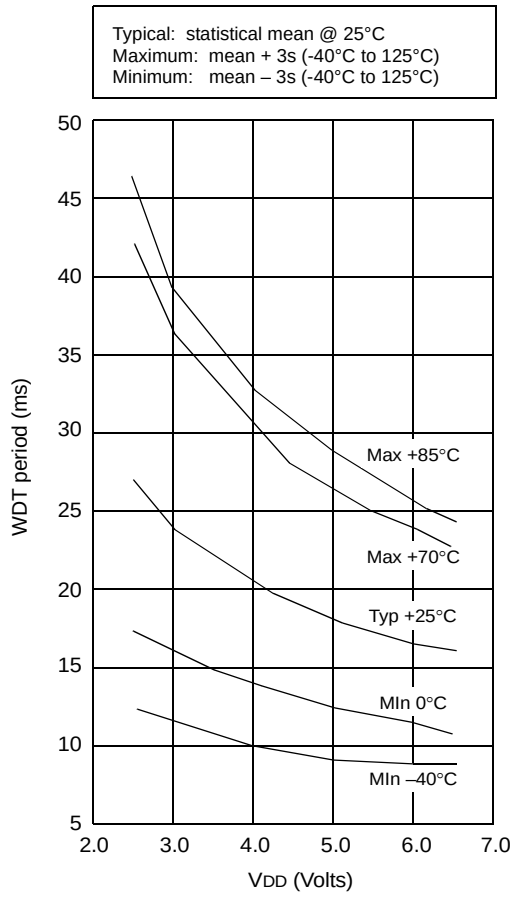
Note 1: The leakage current on the MCLR/VPP pin is strongly dependent on the applied voltage level. The specified levels represent normal operating conditions. Higher leakage current may be measured at different input voltage.

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

3: For the RC mode, the OSC1/CLKIN pin is a Schmitt Trigger input. It is not recommended that the PIC16C5X be driven with external clock in RC mode.

4: The user may use the better of the two specifications.

FIGURE 14-15: WDT TIMER TIME-OUT PERIOD vs. V_{DD} (1)



Note 1: Prescaler set to 1:1.

FIGURE 14-16: TRANSCONDUCTANCE (gm) OF HS OSCILLATOR vs. V_{DD}

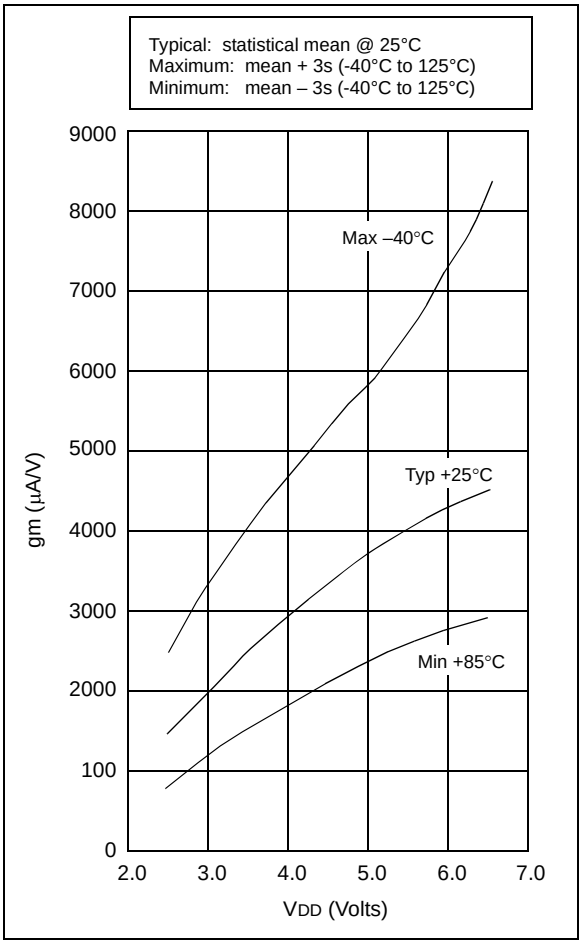


FIGURE 14-19: PORTA, B AND C I_{OH} vs. V_{OH}, V_{DD} = 3 V

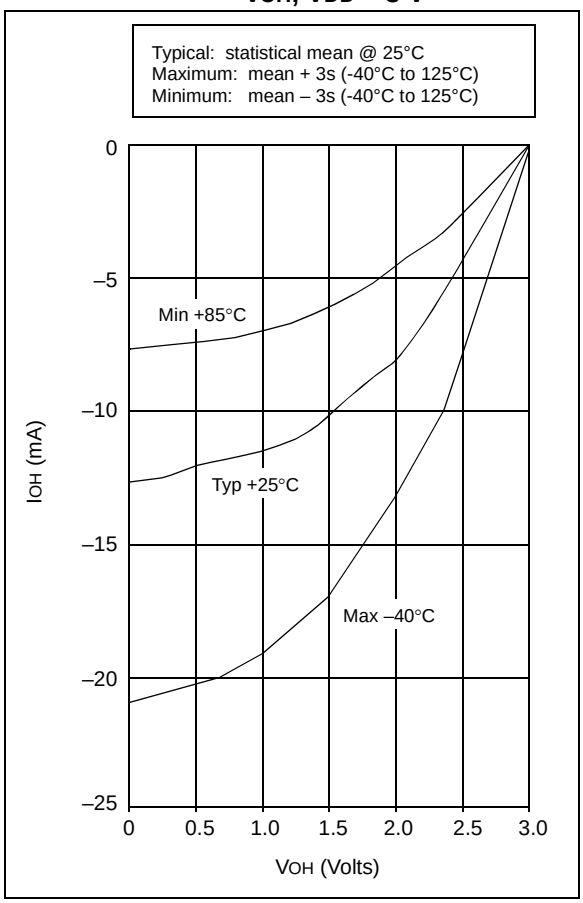


FIGURE 14-20: PORTA, B AND C I_{OH} vs. V_{OH}, V_{DD} = 5 V

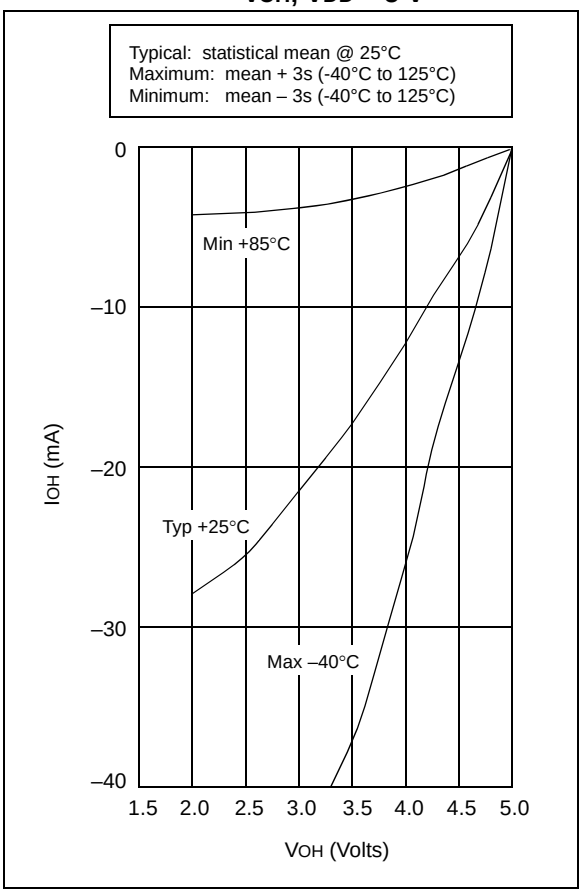
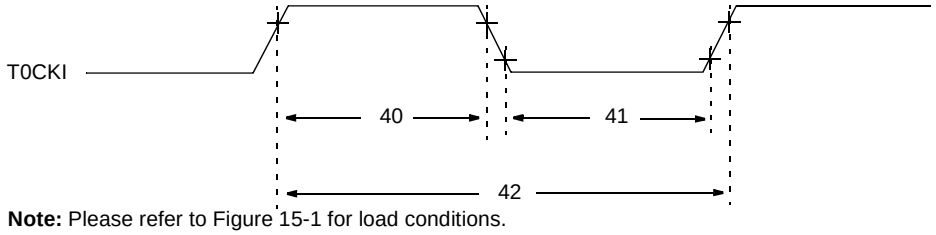


FIGURE 15-5: TIMER0 CLOCK TIMINGS - PIC16C54A



Note: Please refer to Figure 15-1 for load conditions.

TABLE 15-4: TIMER0 CLOCK REQUIREMENTS - PIC16C54A

AC Characteristics			**Standard Operating Conditions (unless otherwise specified)** Operating Temperature 0°C ≤ TA ≤ +70°C for commercial –40°C ≤ TA ≤ +85°C for industrial –20°C ≤ TA ≤ +85°C for industrial - PIC16LV54A-02I –40°C ≤ TA ≤ +125°C for extended				
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
40	Tt0H	T0CKI High Pulse Width - No Prescaler	0.5 TCY + 20*	—	—	ns	
		- With Prescaler	10*	—	—	ns	
41	Tt0L	T0CKI Low Pulse Width - No Prescaler	0.5 TCY + 20*	—	—	ns	
		- With Prescaler	10*	—	—	ns	
42	Tt0P	T0CKI Period	20 or $\frac{T_{CY} + 40}{N}$*	—	—	ns	Whichever is greater. N = Prescale Value (1, 2, 4,..., 256)

* These parameters are characterized but not tested.

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

FIGURE 16-20: PORTA, B AND C I_{OH} vs. V_{OH}, V_{DD} = 3V

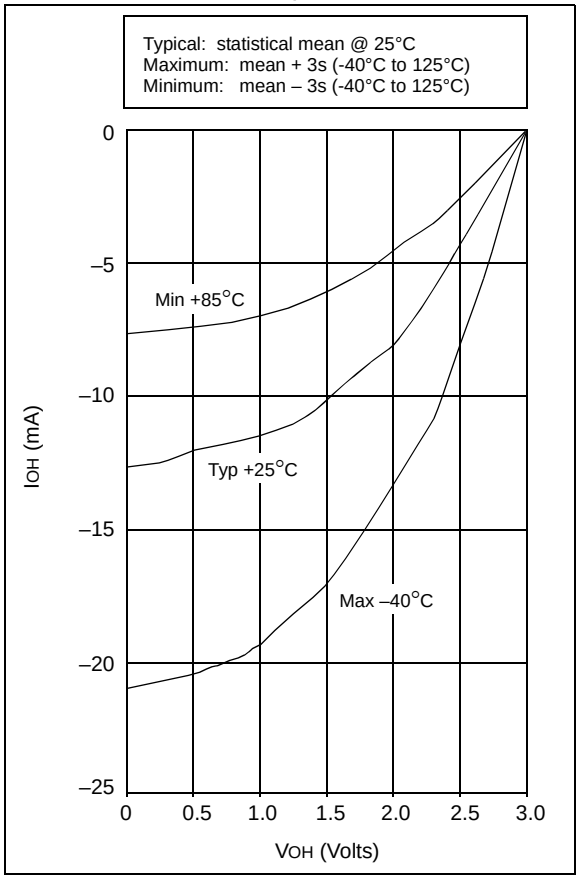
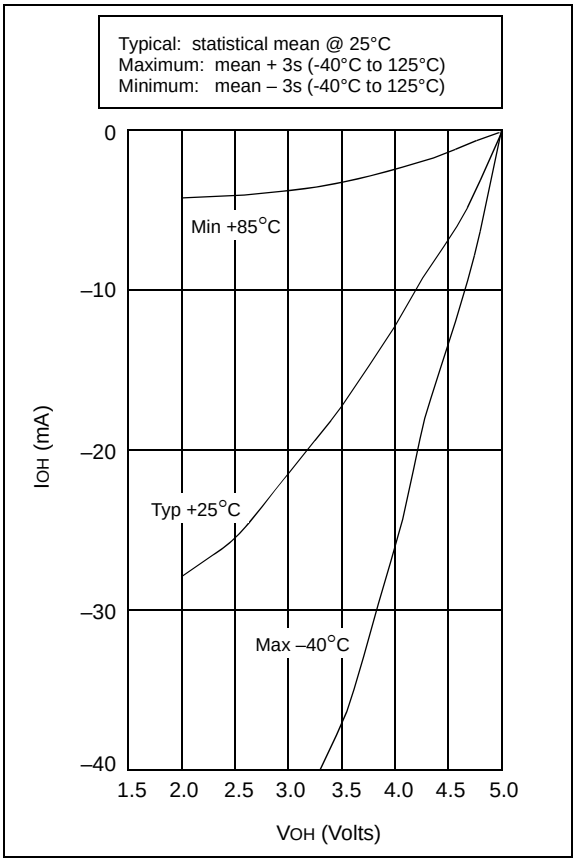


FIGURE 16-21: PORTA, B AND C I_{OH} vs. V_{OH}, V_{DD} = 5V



19.4 Timing Diagrams and Specifications

FIGURE 19-3: EXTERNAL CLOCK TIMING - PIC16C5X-40

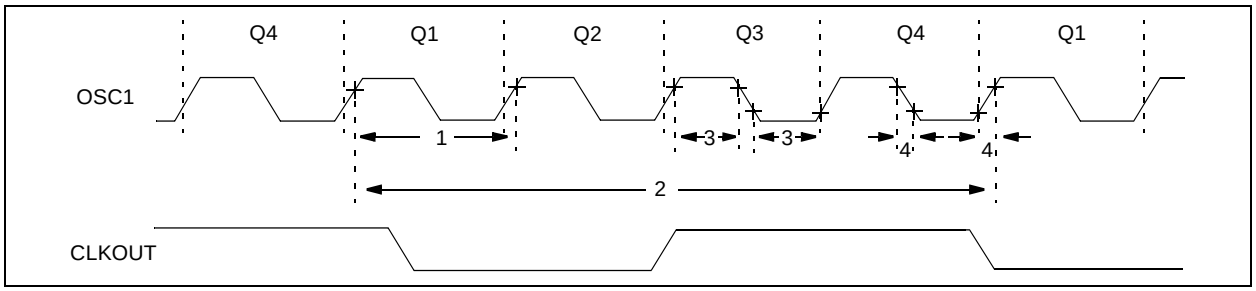


TABLE 19-1: EXTERNAL CLOCK TIMING REQUIREMENTS - PIC16C5X-40

AC Characteristics		Standard Operating Conditions (unless otherwise specified) Operating Temperature 0°C ≤ TA ≤ +70°C for commercial					
Param No.	**Symbol**	**Characteristic**	**Min**	**Typ†**	**Max**	**Units**	**Conditions**
	FOSC	External CLKIN Frequency(1)	20	—	40	MHz	HS OSC mode
1	TOSC	External CLKIN Period(1)	25	—	—	ns	HS OSC mode
2	Tcy	Instruction Cycle Time(2)	—	4/FOSC	—	—	
3	TosL, TosH	Clock in (OSC1) Low or High Time	6.0*	—	—	ns	HS oscillator
4	TosR, TosF	Clock in (OSC1) Rise or Fall Time	—	—	6.5*	ns	HS oscillator

* These parameters are characterized but not tested.

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

Note 1: 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 an unstable oscillator operation and/or higher than expected current consumption.
When an external clock input is used, the "max" cycle time limit is "DC" (no clock) for all devices.

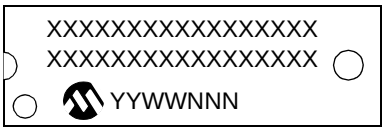
2: Instruction cycle period (TCY) equals four times the input oscillator time base period.

NOTES:

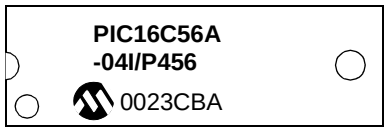
21.0 PACKAGING INFORMATION

21.1 Package Marketing Information

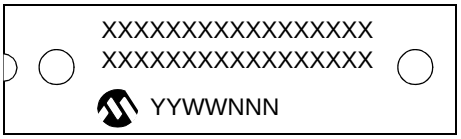
18-Lead PDIP



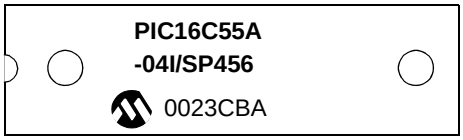
Example



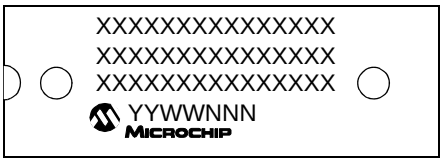
28-Lead Skinny PDIP (.300")



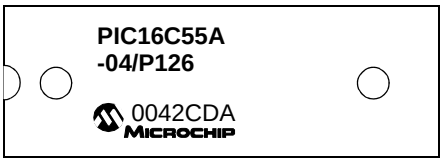
Example



28-Lead PDIP (.600")



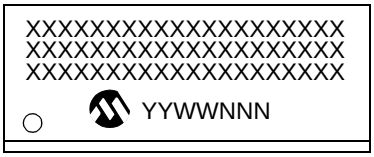
Example



18-Lead SOIC

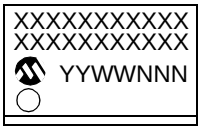
Example

28-Lead SOIC



Example

20-Lead SSOP



Example

28-Lead SSOP

Example

W

X

Z

READER RESPONSE

It is our intention to provide you with the best documentation possible to ensure successful use of your Microchip product. If you wish to provide your comments on organization, clarity, subject matter, and ways in which our documentation can better serve you, please FAX your comments to the Technical Publications Manager at (480) 792-4150.

Please list the following information, and use this outline to provide us with your comments about this Data Sheet.

To: Technical Publications Manager Total Pages Sent

RE: Reader Response

From: Name _____

Company _____

Address _____

City / State / ZIP / Country _____

Telephone: (_____) _____ - _____ FAX: (_____) _____ - _____

Application (optional):

Would you like a reply? ___Y ___N

Device: **PIC16C5X** Literature Number: **DS30453E**

Questions:

1. What are the best features of this document?

2. How does this document meet your hardware and software development needs?

3. Do you find the organization of this data sheet easy to follow? If not, why?

4. What additions to the data sheet do you think would enhance the structure and subject?

5. What deletions from the data sheet could be made without affecting the overall usefulness?

6. Is there any incorrect or misleading information (what and where)?

7. How would you improve this document?

8. How would you improve our software, systems, and silicon products?