

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	4MHz
Connectivity	-
Peripherals	POR, WDT
Number of I/O	5
Program Memory Size	3.5KB (2K x 14)
Program Memory Type	OTP
EEPROM Size	-
RAM Size	128 x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 4x8b
Oscillator Type	Internal
Operating Temperature	0°C ~ 70°C (TA)
Mounting Type	Through Hole
Package / Case	8-DIP (0.300", 7.62mm)
Supplier Device Package	8-PDIP
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/pic12c672-04-p

1.0 GENERAL DESCRIPTION

The PIC12C67X devices are low-cost, high-performance, CMOS, fully-static, 8-bit microcontrollers with integrated analog-to-digital (A/D) converter and EEPROM data memory (EEPROM on PIC12CE67X versions only).

All PIC[®] microcontrollers employ an advanced RISC architecture. The PIC12C67X microcontrollers have enhanced core features, eight-level deep stack, and multiple internal and external interrupt sources. The separate instruction and data buses of the Harvard architecture allow a 14-bit wide instruction word with the 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 35 instructions (reduced instruction set) are available. Additionally, a large register set gives some of the architectural innovations used to achieve a very high performance.

PIC12C67X microcontrollers typically achieve a 2:1 code compression and a 4:1 speed improvement over other 8-bit microcontrollers in their class.

The PIC12C67X devices have 128 bytes of RAM, 16 bytes of EEPROM data memory (PIC12CE67X only), 5 I/O pins and 1 input pin. In addition a timer/counter is available. Also a 4-channel, high-speed, 8-bit A/D is provided. The 8-bit resolution is ideally suited for applications requiring low-cost analog interface, (i.e., thermostat control, pressure sensing, etc.)

The PIC12C67X devices have special features to reduce external components, thus reducing cost, enhancing system reliability and reducing power consumption. The Power-On Reset (POR), Power-up Timer (PWRT), and Oscillator Start-up Timer (OST) eliminate the need for external reset circuitry. There are five oscillator configurations to choose from, including INTRC precision internal oscillator mode and the power-saving LP (Low Power) oscillator mode. Power-saving SLEEP mode, Watchdog Timer and code protection features improve system cost, power and reliability. The SLEEP (power-down) feature provides a power-saving mode. The user can wake-up the chip from SLEEP through several external and internal interrupts and resets.

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

A UV erasable windowed package version is ideal for code development, while the cost-effective One-Time-Programmable (OTP) version is suitable for production in any volume. The customer can take full advantage of Microchip's price leadership in OTP microcontrollers, while benefiting from the OTP's flexibility.

1.1 Applications

The PIC12C67X series fits perfectly in applications ranging from personal care appliances and security systems to low-power remote transmitters/receivers. The EPROM technology makes customizing application programs (transmitter codes, appliance settings, receiver frequencies, etc.) extremely fast and convenient, while the EEPROM data memory (PIC12CE67X only) technology allows for the changing of calibration factors and security codes. The small footprint packages, for through hole or surface mounting, make this microcontroller series perfect for applications with space limitations. Low-cost, low-power, high performance, ease of use and I/O flexibility make the PIC12C67X series very versatile even in areas where no microcontroller use has been considered before (i.e., timer functions, replacement of "glue" logic and PLD's in larger systems, coprocessor applications).

1.2 Family and Upward Compatibility

The PIC12C67X products are compatible with other members of the 14-bit PIC16CXXX families.

1.3 Development Support

The PIC12C67X devices are supported by a full-featured macro assembler, a software simulator, an in-circuit emulator, a low-cost development programmer and a full-featured programmer. A "C" compiler and fuzzy logic support tools are also available.

2.0 PIC12C67X DEVICE VARIETIES

A variety of frequency ranges and packaging options are available. Depending on application and production requirements, the proper device option can be selected using the information in the PIC12C67X Product Identification System section at the end of this data sheet. When placing orders, please use that page of the data sheet to specify the correct part number.

For example, the PIC12C67X device “type” is indicated in the device number:

1. **C**, as in PIC12**C**671. These devices have EPROM type memory and operate over the standard voltage range.
2. **LC**, as in PIC12**LC**671. These devices have EPROM type memory and operate over an extended voltage range.
3. **CE**, as in PIC12**CE**674. These devices have EPROM type memory, EEPROM data memory and operate over the standard voltage range.
4. **LCE**, as in PIC12**LCE**674. These devices have EPROM type memory, EEPROM data memory and operate over an extended voltage range.

2.1 UV Erasable Devices

The UV erasable version, offered in windowed package, is optimal for prototype development and pilot programs.

The UV erasable version can be erased and reprogrammed to any of the configuration modes. Microchip's PICSTART® Plus and PRO MATE® programmers both support the PIC12C67X. Third party programmers also are available; refer to the Microchip Third Party Guide for a list of sources.

Note: Please note that erasing the device will also erase the pre-programmed internal calibration value for the internal oscillator. The calibration value must be saved prior to erasing the part.

2.2 One-Time-Programmable (OTP) Devices

The availability of OTP devices is especially useful for customers who need the flexibility for frequent code updates and small volume applications.

The OTP devices, packaged in plastic packages, permit the user to program them once. In addition to the program memory, the configuration bits must also be programmed.

2.3 Quick-Turn-Programming (QTP) Devices

Microchip offers a QTP Programming Service for factory production orders. This service is made available for users who choose not to program a medium to high quantity of units and whose code patterns have stabilized. The devices are identical to the OTP devices, but with all EPROM locations and configuration options already programmed by the factory. Certain code and prototype verification procedures apply before production shipments are available. Please contact your local Microchip Technology sales office for more details.

2.4 Serialized Quick-Turn Programming (SQTPSM) Devices

Microchip offers a unique programming service where a few user-defined locations in each device are programmed with different serial numbers. The serial numbers may be random, pseudo-random, or sequential.

Serial programming allows each device to have a unique number which can serve as an entry-code, password, or ID number.

3.0 ARCHITECTURAL OVERVIEW

The high performance of the PIC12C67X family can be attributed to a number of architectural features commonly found in RISC microprocessors. To begin with, the PIC12C67X uses a Harvard architecture, in which program and data are accessed from separate memories using separate buses. This improves bandwidth over traditional von Neumann architecture in which program and data are fetched from the same memory using the same bus. Separating program and data buses also allow instructions to be sized differently than the 8-bit wide data word. Instruction opcodes are 14-bits wide making it possible to have all single word instructions. A 14-bit wide program memory access bus fetches a 14-bit instruction in a single instruction cycle. A two-stage pipeline overlaps fetch and execution of instructions (Example 3-1). Consequently, all instructions (35) execute in a single cycle (200 ns @ 20 MHz) except for program branches.

The table below lists program memory (EPROM), data memory (RAM), and non-volatile memory (EEPROM) for each PIC12C67X device.

Device	Program Memory	RAM Data Memory	EEPROM Data Memory
PIC12C671	1K x 14	128 x 8	—
PIC12C672	2K x 14	128 x 8	—
PIC12CE673	1K x 14	128 x 8	16x8
PIC12CE674	2K x 14	128 x 8	16x8

The PIC12C67X can directly or indirectly address its register files or data memory. All special function registers, including the program counter, are mapped in the data memory. The PIC12C67X has an orthogonal (symmetrical) instruction set that makes it possible to carry out any operation on any register using any addressing mode. This symmetrical nature and lack of 'special optimal situations' make programming with the PIC12C67X simple yet efficient. In addition, the learning curve is reduced significantly.

PIC12C67X devices contain an 8-bit ALU and working register. The ALU is a general purpose arithmetic unit. It performs arithmetic and Boolean functions between the data in the working register and any register file.

The ALU is 8-bits wide and capable of addition, subtraction, shift and logical operations. Unless otherwise mentioned, arithmetic operations are two's complement in nature. In two-operand instructions, typically one operand is the working register (W register). The other operand is a file register or an immediate constant. In single operand instructions, the operand is either the W register or a file register.

The W register is an 8-bit working register used for ALU operations. It is not an addressable register.

Depending on the instruction executed, the ALU may affect the values of the Carry (C), Digit Carry (DC), and Zero (Z) bits in the STATUS register. The C and DC bits operate as a borrow bit and a digit borrow out bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

PIC12C67X

3.1 Clocking Scheme/Instruction Cycle

The clock input (from OSC1) is internally divided by four to generate four non-overlapping quadrature clocks, namely Q1, Q2, Q3 and Q4. Internally, the program counter (PC) is incremented every Q1, and the instruction is fetched from the program memory and latched into the instruction register in Q4. The instruction is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow is shown in Figure 3-2.

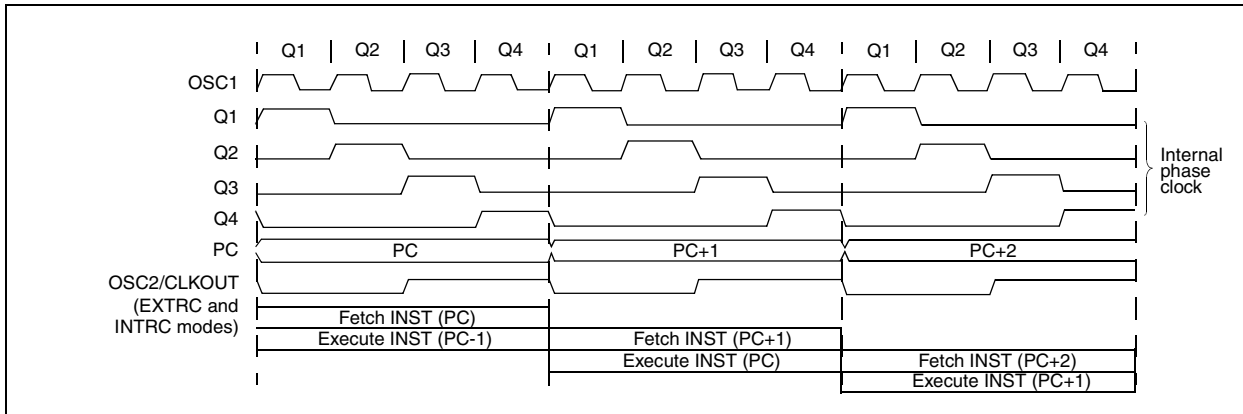
3.2 Instruction Flow/Pipelining

An "Instruction Cycle" consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined such that fetch takes one instruction cycle, while decode and execute takes another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (i.e., GOTO), then two cycles are required to complete the instruction (Example 3-1).

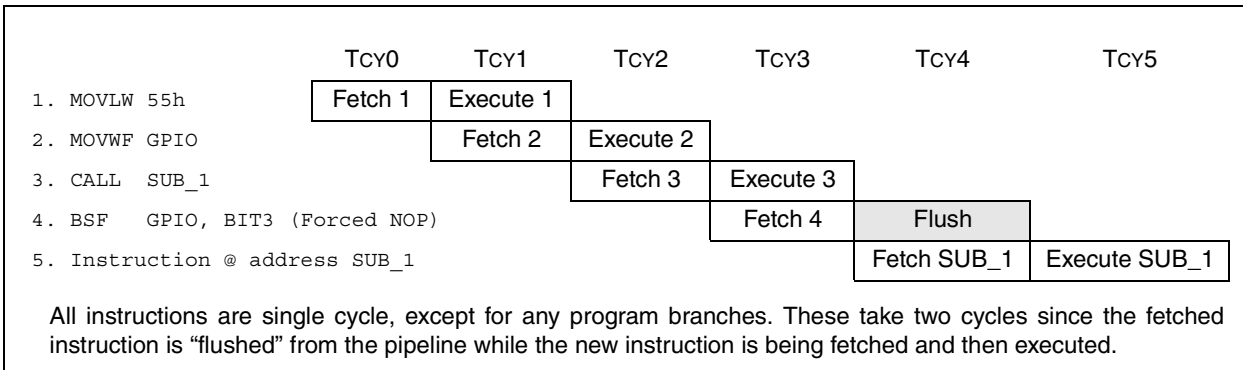
A fetch cycle begins with the program counter (PC) incrementing in Q1.

In the execution cycle, the fetched instruction is latched into the "Instruction Register" (IR) in cycle Q1. This instruction is then decoded and executed during the Q2, Q3, and Q4 cycles. Data memory is read during Q2 (operand read) and written during Q4 (destination write).

FIGURE 3-2: CLOCK/INSTRUCTION CYCLE



EXAMPLE 3-1: INSTRUCTION PIPELINE FLOW



4.2.2.7 OSCCAL REGISTER

The Oscillator Calibration (OSCCAL) Register is used to calibrate the internal 4 MHz oscillator. It contains four bits for fine calibration and two other bits to either increase or decrease frequency.

REGISTER 4-7: OSCCAL REGISTER (ADDRESS 8Fh)

R/W-0	R/W-1	R/W-1	R/W-1	R/W-0	R/W-0	U-0	U-0	
CAL3	CAL2	CAL1	CAL0	CALFST	CALSLW	—	—	
bit7								bit0

R = Readable bit
W = Writable bit
U = Unimplemented bit, read as '0'
- n = Value at POR reset

bit 7-4: **CAL<3:0>**: Fine Calibration

bit 3: **CALFST**: Calibration Fast
1 = Increase frequency
0 = No change

bit 2: **CALSLW**: Calibration Slow
1 = Decrease frequency
0 = No change

bit 1-0: **Unimplemented**: Read as '0'

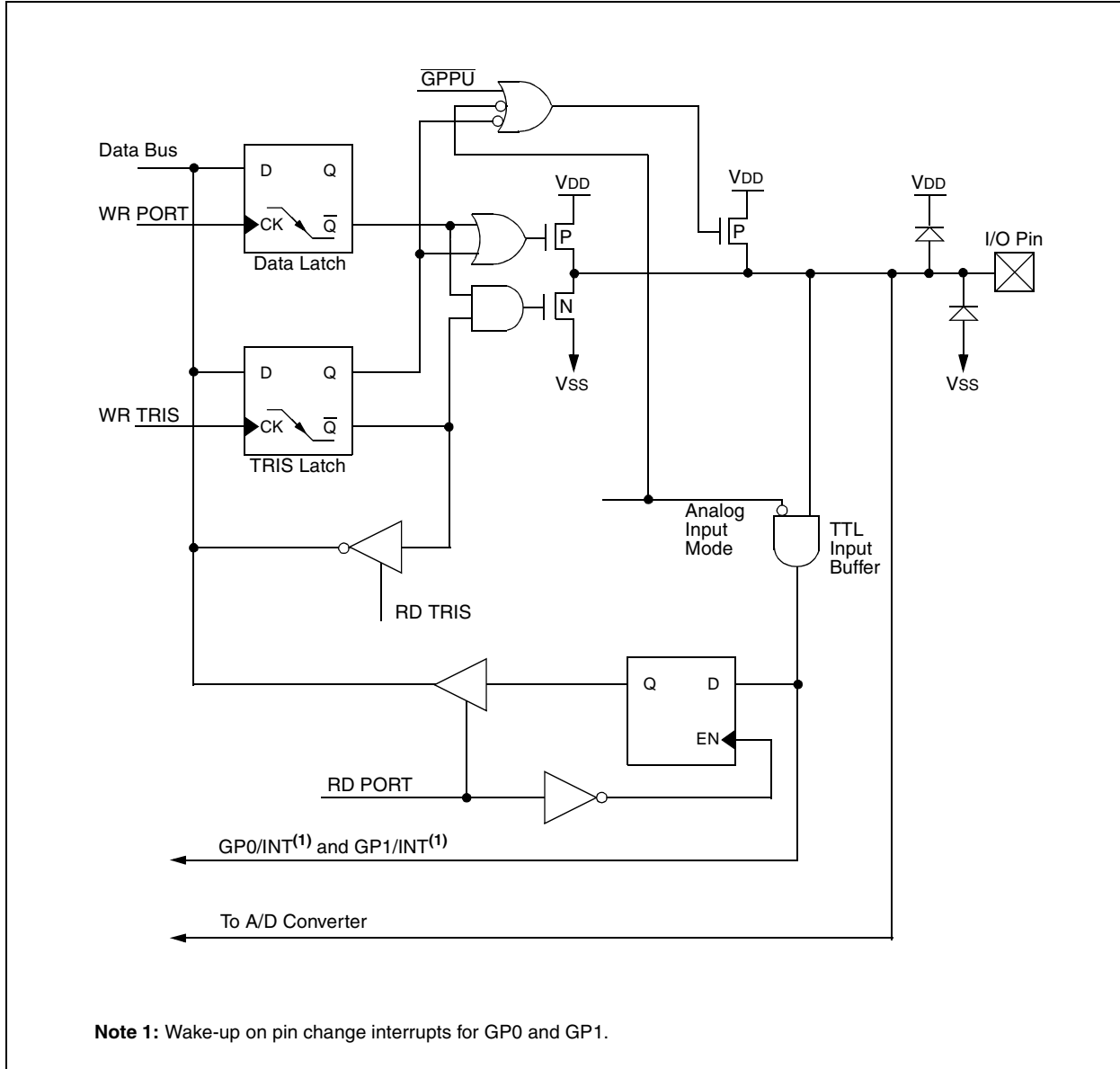
Note: If CALFST = 1 and CALSLW = 1, CALFST has precedence.

PIC12C67X

NOTES:

PIC12C67X

FIGURE 5-1: BLOCK DIAGRAM OF GP0/AN0 AND GP1/AN1/VREF PIN



PIC12C67X

6.3 Write Operations

6.3.1 BYTE WRITE

Following the start signal from the processor, the device code (4 bits), the don't care bits (3 bits), and the R/W bit (which is a logic low) are placed onto the bus by the processor. This indicates to the addressed EEPROM that a byte with a word address will follow after it has generated an acknowledge bit during the ninth clock cycle. Therefore, the next byte transmitted by the processor is the word address and will be written into the address pointer. Only the lower four address bits are used by the device, and the upper four bits are don't cares. If the address byte is acknowledged, the processor will then transmit the data word to be written into the addressed memory location. The memory acknowledges again and the processor generates a stop condition. This initiates the internal write cycle, and during this time will not generate acknowledge signals. After a byte write command, the internal address counter will not be incremented and will point to the same address location that was just written. If a stop bit sequence is transmitted to the device at any point in the write command sequence before the entire sequence is complete, then the command will abort and no data will be written. If more than 8 data bits are transmitted before the stop bit sequence is sent, then the device will clear the previously loaded byte and begin loading the data buffer again. If more than one data byte is transmitted to the device and a stop bit is sent before a full eight data bits have been transmitted, then the write command will abort and no data will be written. The EEPROM memory employs a VCC threshold detector circuit, which disables the internal erase/write logic if the VCC is below minimum VDD. Byte write operations must be preceded and immediately followed by a bus not busy bus cycle where both SDA and SCL are held high. (See Figure 6-7 for Byte Write operation.)

6.4 Acknowledge Polling

Since the EEPROM will not acknowledge during a write cycle, this can be used to determine when the cycle is complete (this feature can be used to maximize bus throughput). Once the stop condition for a write command has been issued from the processor, the device initiates the internally timed write cycle. ACK polling can be initiated immediately. This involves the processor sending a start condition followed by the control byte for a write command ($R/\bar{W} = 0$). If the device is still busy with the write cycle, then no ACK will be returned. If no ACK is returned, then the start bit and control byte must be re-sent. If the cycle is complete, then the device will return the ACK and the processor can then proceed with the next read or write command. (See Figure 6-6 for flow diagram.)

FIGURE 6-6: ACKNOWLEDGE POLLING FLOW

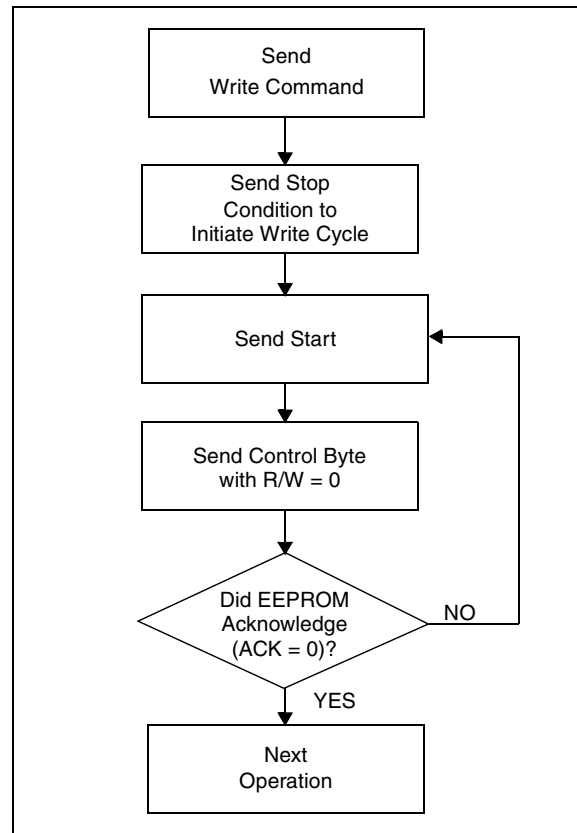
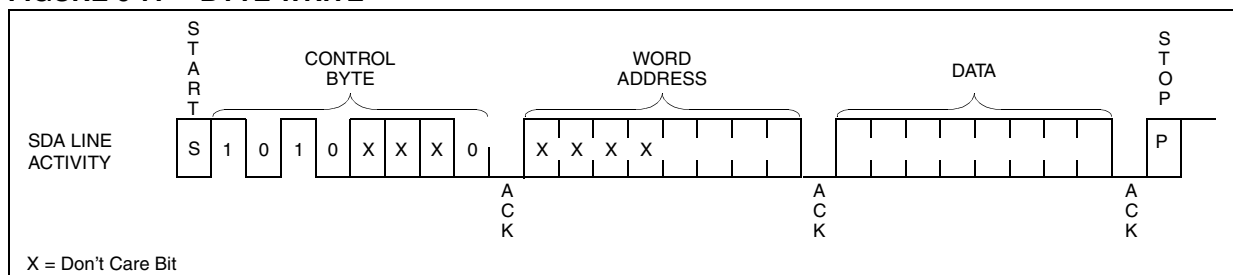


FIGURE 6-7: BYTE WRITE



6.5 Read Operations

Read operations are initiated in the same way as write operations with the exception that the R/W bit of the EEPROM address is set to one. There are three basic types of read operations; current address read, random read and sequential read.

6.5.1 CURRENT ADDRESS READ

The EEPROM contains an address counter that maintains the address of the last word accessed, internally incremented by one. Therefore, if the previous read access was to address n , the next current address read operation would access data from address $n + 1$. Upon receipt of the EEPROM address with the R/W bit set to one, the EEPROM issues an acknowledge and transmits the 8-bit data word. The processor will not acknowledge the transfer, but does generate a stop condition and the EEPROM discontinues transmission (Figure 6-8).

6.5.2 RANDOM READ

Random read operations allow the processor to access any memory location in a random manner. To perform this type of read operation, first the word address must be set. This is done by sending the word address to the EEPROM as part of a write operation. After the word

address is sent, the processor generates a start condition following the acknowledge. This terminates the write operation, but not before the internal address pointer is set. Then the processor issues the control byte again, but with the R/W bit set to a one. The EEPROM will then issue an acknowledge and transmits the 8-bit data word. The processor will not acknowledge the transfer, but does generate a stop condition and the EEPROM discontinues transmission (Figure 6-9). After this command, the internal address counter will point to the address location following the one that was just read.

6.5.3 SEQUENTIAL READ

Sequential reads are initiated in the same way as a random read, except that after the device transmits the first data byte, the processor issues an acknowledge as opposed to a stop condition in a random read. This directs the EEPROM to transmit the next sequentially addressed 8-bit word (Figure 6-10).

To provide sequential reads, the EEPROM contains an internal address pointer, which is incremented by one at the completion of each read operation. This address pointer allows the entire memory contents to be serially read during one operation.

FIGURE 6-8: CURRENT ADDRESS READ

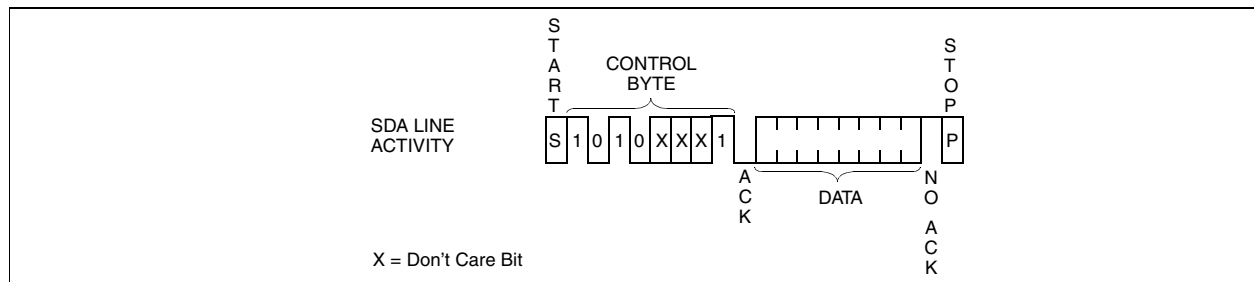


FIGURE 6-9: RANDOM READ

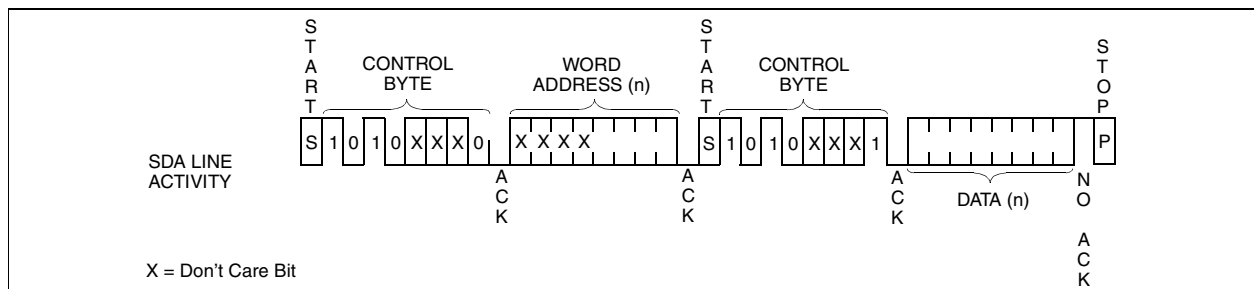
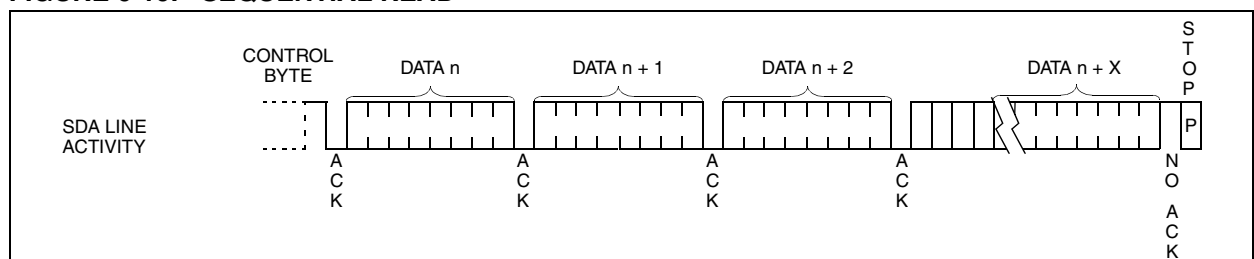


FIGURE 6-10: SEQUENTIAL READ



8.2 Selecting the A/D Conversion Clock

The A/D conversion time per bit is defined as TAD. The A/D conversion requires 9.5 TAD per 8-bit conversion. The source of the A/D conversion clock is software selected. The four possible options for TAD are:

- 2TOSC
- 8TOSC
- 32TOSC
- Internal ADC RC oscillator

For correct A/D conversions, the A/D conversion clock (TAD) must be selected to ensure a minimum TAD time of 1.6 μ s. If the minimum TAD time of 1.6 μ s can not be obtained, TAD should be ≤ 8 μ s for preferred operation.

Table 8-1 shows the resultant TAD times derived from the device operating frequencies and the A/D clock source selected.

8.3 Configuring Analog Port Pins

The ADCON1 and TRIS Registers control the operation of the A/D port pins. The port pins that are desired as analog inputs must have their corresponding TRIS bits set (input). If the TRIS bit is cleared (output), the digital output level (VOH or VOL) will be converted.

The A/D operation is independent of the state of the CHS<2:0> bits and the TRIS bits.

Note 1: When reading the port register, all pins configured as analog input channel will read as cleared (a low level). Pins configured as digital inputs, will convert an analog input. Analog levels on a digitally configured input will not affect the conversion accuracy.

2: Analog levels on any pin that is defined as a digital input (including the AN<3:0> pins) may cause the input buffer to consume current that is out of the devices specification.

TABLE 8-1: TAD vs. DEVICE OPERATING FREQUENCIES

AD Clock Source (TAD)		Device Frequency		
Operation	ADCS<1:0>	4 MHz	1.25 MHz	333.33 kHz
2TOSC	00	500 ns ⁽²⁾	1.6 μ s	6 μ s
8TOSC	01	2.0 μ s	6.4 μ s	24 μ s ⁽³⁾
32TOSC	10	8.0 μ s	25.6 μ s ⁽³⁾	96 μ s ⁽³⁾
Internal ADC RC Oscillator ⁽⁵⁾	11	2 - 6 μ s ^(1,4)	2 - 6 μ s ^(1,4)	2 - 6 μ s ⁽¹⁾

Note 1: The RC source has a typical TAD time of 4 μ s.

2: These values violate the minimum required TAD time.

3: For faster conversion times, the selection of another clock source is recommended.

4: While in RC mode, with device frequency above 1 MHz, conversion accuracy is out of specification.

5: For extended voltage devices (LC), please refer to Electrical Specifications section.

9.0 SPECIAL FEATURES OF THE CPU

What sets a microcontroller apart from other processors are special circuits to deal with the needs of real-time applications. The PIC12C67X family has a host of such features intended to maximize system reliability, minimize cost through elimination of external components, provide power saving operating modes and offer code protection. These are:

- Oscillator selection
- Reset
 - Power-on Reset (POR)
 - Power-up Timer (PWRT)
 - Oscillator Start-up Timer (OST)
- Interrupts
- Watchdog Timer (WDT)
- SLEEP
- Code protection
- ID locations
- In-circuit serial programming

The PIC12C67X has a Watchdog Timer, which can be shut off only through configuration bits. It runs off its own RC oscillator for added reliability. There are two timers that offer necessary delays on power-up. One is the Oscillator Start-up Timer (OST), intended to keep

the chip in reset until the crystal oscillator is stable. The other is the Power-up Timer (PWRT), which provides a fixed delay of 72 ms (nominal) on power-up only, designed to keep the part in reset while the power supply stabilizes. With these two timers on-chip, most applications need no external reset circuitry.

SLEEP mode is designed to offer a very low current power-down mode. The user can wake-up from SLEEP through external reset, Watchdog Timer Wake-up, or through an interrupt. Several oscillator options are also made available to allow the part to fit the application. The INTRC/EXTRC oscillator option saves system cost, while the LP crystal option saves power. A set of configuration bits are used to select various options.

9.1 Configuration Bits

The configuration bits can be programmed (read as '0') or left unprogrammed (read as '1') to select various device configurations. These bits are mapped in program memory location 2007h.

The user will note that address 2007h is beyond the user program memory space. In fact, it belongs to the special test/configuration memory space (2000h-3FFFh), which can be accessed only during programming.

REGISTER 9-1: CONFIGURATION WORD

	CP1	CP0	CP1	CP0	CP1	CP0	MCLRE	CP1	CP0	PWRTE	WDTE	FOSC2	FOSC1	FOSC0	
bit13														bit0	Register: CONFIG Address: 2007h
<p>bit 13-8, CP<1:0>: Code Protection bit pairs⁽¹⁾</p> <p>6-5: 11 = Code protection off 10 = Locations 400h through 7FEh code protected (do not use for PIC12C671 and PIC12CE673) 01 = Locations 200h through 7FEh code protected 00 = All memory is code protected</p> <p>bit 7: MCLRE: Master Clear Reset Enable bit 1 = Master Clear Enabled 0 = Master Clear Disabled</p> <p>bit 4: PWRTE: Power-up Timer Enable bit 1 = PWRT disabled 0 = PWRT enabled</p> <p>bit 3: WDTE: Watchdog Timer Enable bit 1 = WDT enabled 0 = WDT disabled</p> <p>bit 2-0: FOSC<2:0>: Oscillator Selection bits 111 = EXTRC, Clockout on OSC2 110 = EXTRC, OSC2 is I/O 101 = INTRC, Clockout on OSC2 100 = INTRC, OSC2 is I/O 011 = Invalid Selection 010 = HS Oscillator 001 = XT Oscillator 000 = LP Oscillator</p> <p>Note 1: All of the CP<1:0> pairs have to be given the same value to enable the code protection scheme listed.</p>															

TABLE 9-6: RESET CONDITION FOR SPECIAL REGISTERS

Condition	Program Counter	STATUS Register	PCON Register
Power-on Reset	000h	0001 1xxx	---- --0-
MCLR Reset during normal operation	000h	000u uuuu	---- --u-
MCLR Reset during SLEEP	000h	0001 0uuu	---- --u-
WDT Reset during normal operation	000h	0000 uuuu	---- --u-
WDT Wake-up from SLEEP	PC + 1	uuu0 0uuu	---- --u-
Interrupt wake-up from SLEEP	PC + 1 ⁽¹⁾	uuu1 0uuu	---- --u-

Legend: u = unchanged, x = unknown, - = unimplemented bit read as '0'.

Note 1: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

TABLE 9-7: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Power-on Reset	MCLR Resets WDT Reset	Wake-up via WDT or Interrupt
W	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF	0000 0000	0000 0000	0000 0000
TMR0	xxxx xxxx	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	PC + 1 ⁽²⁾
STATUS	0001 1xxx	000q quuu ⁽³⁾	uuuq quuu ⁽³⁾
FSR	xxxx xxxx	uuuu uuuu	uuuu uuuu
GPIO PIC12CE67X	11xx xxxx	11uu uuuu	11uu uuuu
GPIO PIC12C67X	--xx xxxx	--uu uuuu	--uu uuuu
PCLATH	---0 0000	---0 0000	---u uuuu
INTCON	0000 000x	0000 000u	uuuu uqqq ⁽¹⁾
PIR1	-0-- ----	-0-- ----	-q-- ---- ⁽⁴⁾
ADCON0	0000 0000	0000 0000	uuuu uquu ⁽⁵⁾
OPTION	1111 1111	1111 1111	uuuu uuuu
TRIS	--11 1111	--11 1111	--uu uuuu
PIE1	-0-- ----	-0-- ----	-u-- ----
PCON	---- --0-	---- --u-	---- --u-
OSCCAL	0111 00--	uuuu uu--	uuuu uu--
ADCON1	---- -000	---- -000	---- -uuu

Legend: u = unchanged, x = unknown, - = unimplemented bit, read as '0', q = value depends on condition.

Note 1: One or more bits in INTCON and PIR1 will be affected (to cause wake-up).

2: When the wake-up is due to an interrupt and the GIE bit is set, the PC is loaded with the interrupt vector (0004h).

3: See Table 9-5 for reset value for specific condition.

4: If wake-up was due to A/D completing then bit 6 = 1, all other interrupts generating a wake-up will cause bit 6 = u.

5: If wake-up was due to A/D completing then bit 3 = 0, all other interrupts generating a wake-up will cause bit 3 = u.

PIC12C67X

9.5.1 TMR0 INTERRUPT

An overflow (FFh → 00h) in the TMR0 register will set flag bit T0IF (INTCON<2>). The interrupt can be enabled/disabled by setting/clearing enable bit T0IE (INTCON<5>) (Section 7.0). The flag bit T0IF (INTCON<2>) will be set, regardless of the state of the enable bits. If used, this flag must be cleared in software.

9.5.2 INT INTERRUPT

External interrupt on GP2/INT pin is edge triggered; either rising if bit INTEDG (OPTION<6>) is set, or falling, if the INTEDG bit is clear. When a valid edge appears on the GP2/INT pin, flag bit INTF (INTCON<1>) is set. This interrupt can be disabled by clearing enable bit INTE (INTCON<4>). Flag bit INTF must be cleared in software in the interrupt service routine before re-enabling this interrupt. The INT interrupt can wake-up the processor from SLEEP, if bit INTE was set prior to going into SLEEP. The status of global interrupt enable bit GIE decides whether or not the processor branches to the interrupt vector following wake-up. See Section 9.8 for details on SLEEP mode.

9.5.3 GPIO INTCON CHANGE

An input change on GP3, GP1 or GP0 sets flag bit GPIF (INTCON<0>). The interrupt can be enabled/disabled by setting/clearing enable bit GPIE (INTCON<3>) (Section 5.1). This flag bit GPIF (INTCON<0>) will be set, regardless of the state of the enable bits. If used, this flag must be cleared in software.

9.6 Context Saving During Interrupts

During an interrupt, only the return PC value is saved on the stack. Typically, users may wish to save key registers during an interrupt (i.e., W register and STATUS register). This will have to be implemented in software.

Example 9-1 shows the storing and restoring of the STATUS and W registers. The register, W_TEMP, must be defined in both banks and must be defined at the same offset from the bank base address (i.e., if W_TEMP is defined at 0x20 in bank 0, it must also be defined at 0xA0 in bank 1).

Example 9-2 shows the saving and restoring of STATUS and W using RAM locations 0x70 - 0x7F. W_TEMP is defined at 0x70 and STATUS_TEMP is defined at 0x71.

The example:

- Stores the W register.
- Stores the STATUS register in bank 0.
- Executes the ISR code.
- Restores the STATUS register (and bank select bit).
- Restores the W register.
- Returns from interrupt.

EXAMPLE 9-1: SAVING STATUS AND W REGISTERS USING GENERAL PURPOSE RAM (0x20 - 0x6F)

```
MOVWF    W_TEMP          ;Copy W to TEMP register, could be bank one or zero
SWAPF    STATUS,W        ;Swap status to be saved into W
BCF      STATUS,RP0      ;Change to bank zero, regardless of current bank
MOVWF    STATUS_TEMP     ;Save status to bank zero STATUS_TEMP register
:
:(ISR)
:
SWAPF    STATUS_TEMP,W   ;Swap STATUS_TEMP register into W
                        ;(sets bank to original state)
MOVWF    STATUS          ;Move W into STATUS register
SWAPF    W_TEMP,F        ;Swap W_TEMP
SWAPF    W_TEMP,W        ;Swap W_TEMP into W
RETFIE   ;Return from interrupt
```

EXAMPLE 9-2: SAVING STATUS AND W REGISTERS USING SHARED RAM (0x70 - 0x7F)

```
MOVWF    W_TEMP          ;Copy W to TEMP register (bank independent)
MOVF     STATUS,W        ;Move STATUS register into W
MOVWF    STATUS_TEMP     ;Save contents of STATUS register
:
:(ISR)
:
MOVF     STATUS_TEMP,W   ;Retrieve copy of STATUS register
MOVWF    STATUS          ;Restore pre-isr STATUS register contents
SWAPF    W_TEMP,F        ;
SWAPF    W_TEMP,W        ;Restore pre-isr W register contents
RETFIE   ;Return from interrupt
```

PIC12C67X

BTFS **Bit Test f, Skip if Set**

Syntax: [*label*] BTFS *f*,*b*

Operands: $0 \leq f \leq 127$
 $0 \leq b < 7$

Operation: skip if (*f*<*b*>) = 1

Status Affected: None

Encoding:

01	11bb	bfff	ffff
----	------	------	------

Description: If bit '*b*' in register '*f*' is '1', then the next instruction is skipped. If bit '*b*' is '1', then the next instruction fetched during the current instruction execution, is discarded and a NOP is executed instead, making this a 2 cycle instruction.

Words: 1

Cycles: 1(2)

Example

```

HERE    BTFS    FLAG,1
FALSE   GOTO   PROCESS_CO
TRUE    •       DE
        •
        •
  
```

Before Instruction
PC = address HERE

After Instruction
if FLAG<1> = 0,
PC = address FALSE
if FLAG<1> = 1,
PC = address TRUE

CALL **Call Subroutine**

Syntax: [*label*] CALL *k*

Operands: $0 \leq k \leq 2047$

Operation: (PC)+ 1 → TOS,
k → PC<10:0>,
(PCLATH<4:3>) → PC<12:11>

Status Affected: None

Encoding:

10	0kkk	kkkk	kkkk
----	------	------	------

Description: Call Subroutine. First, return address (PC+1) is pushed onto the stack. The eleven bit immediate address is loaded into PC bits <10:0>. The upper bits of the PC are loaded from PCLATH. CALL is a two cycle instruction.

Words: 1

Cycles: 2

Example

```

HERE    CALL
        THER
        E
  
```

Before Instruction
PC = Address HERE

After Instruction
PC = Address THERE
TOS = Address HERE+1

CLRF **Clear f**

Syntax: [*label*] CLRF *f*

Operands: $0 \leq f \leq 127$

Operation: 00h → (*f*)
1 → Z

Status Affected: Z

Encoding:

00	0001	1fff	ffff
----	------	------	------

Description: The contents of register '*f*' are cleared and the Z bit is set.

Words: 1

Cycles: 1

Example

```

CLRF    FLAG_REG
  
```

Before Instruction
FLAG_REG = 0x5A

After Instruction
FLAG_REG = 0x00
Z = 1

CLRW **Clear W**

Syntax: [*label*] CLRW

Operands: None

Operation: 00h → (W)
1 → Z

Status Affected: Z

Encoding:

00	0001	0000	0011
----	------	------	------

Description: W register is cleared. Zero bit (Z) is set.

Words: 1

Cycles: 1

Example

```

CLRW
  
```

Before Instruction
W = 0x5A

After Instruction
W = 0x00
Z = 1

MPLIB is a librarian for pre-compiled code to be used with MPLINK. When a routine from a library is called from another source file, only the modules that contains that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications. MPLIB manages the creation and modification of library files.

MPLINK features include:

- MPLINK works with MPASM and MPLAB-C17 and MPLAB-C18.
- MPLINK allows all memory areas to be defined as sections to provide link-time flexibility.

MPLIB features include:

- MPLIB makes linking easier because single libraries can be included instead of many smaller files.
- MPLIB helps keep code maintainable by grouping related modules together.
- MPLIB commands allow libraries to be created and modules to be added, listed, replaced, deleted, or extracted.

11.5 MPLAB-SIM Software Simulator

The MPLAB-SIM Software Simulator allows code development in a PC host environment by simulating the PIC series microcontrollers on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a file or user-defined key press to any of the pins. The execution can be performed in single step, execute until break, or trace mode.

MPLAB-SIM fully supports symbolic debugging using MPLAB-C17 and MPLAB-C18 and MPASM. The Software Simulator offers the flexibility to develop and debug code outside of the laboratory environment making it an excellent multi-project software development tool.

11.6 MPLAB-ICE High Performance Universal In-Circuit Emulator with MPLAB IDE

The MPLAB-ICE Universal In-Circuit Emulator is intended to provide the product development engineer with a complete microcontroller design tool set for PIC microcontrollers (MCUs). Software control of MPLAB-ICE is provided by the MPLAB Integrated Development Environment (IDE), which allows editing, “make” and download, and source debugging from a single environment.

Interchangeable processor modules allow the system to be easily reconfigured for emulation of different processors. The universal architecture of the MPLAB-ICE allows expansion to support new PIC microcontrollers.

The MPLAB-ICE Emulator System has been designed as a real-time emulation system with advanced features that are generally found on more expensive devel-

opment tools. The PC platform and Microsoft® Windows 3.x/95/98 environment were chosen to best make these features available to you, the end user.

MPLAB-ICE 2000 is a full-featured emulator system with enhanced trace, trigger, and data monitoring features. Both systems use the same processor modules and will operate across the full operating speed range of the PIC MCU.

11.7 PICMASTER/PICMASTER CE

The PICMASTER system from Microchip Technology is a full-featured, professional quality emulator system. This flexible in-circuit emulator provides a high-quality, universal platform for emulating Microchip 8-bit PIC microcontrollers (MCUs). PICMASTER systems are sold worldwide, with a CE compliant model available for European Union (EU) countries.

11.8 ICEPIC

ICEPIC is a low-cost in-circuit emulation solution for the Microchip Technology PIC16C5X, PIC16C6X, PIC16C7X, and PIC16CXXX families of 8-bit one-time-programmable (OTP) microcontrollers. The modular system can support different subsets of PIC16C5X or PIC16CXXX products through the use of interchangeable personality modules or daughter boards. The emulator is capable of emulating without target application circuitry being present.

11.9 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 PIC16F877 and can be used to develop for this and other PIC microcontrollers from the PIC16CXXX family. MPLAB-ICD utilizes the In-Circuit Debugging capability built into the PIC16F87X. This feature, along with Microchip's In-Circuit Serial Programming protocol, offers cost-effective in-circuit flash programming and 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. The MPLAB-ICD is also a programmer for the flash PIC16F87X family.

11.10 PRO MATE II Universal Programmer

The PRO MATE II Universal Programmer is a full-featured programmer capable of operating in stand-alone mode as well as PC-hosted mode. PRO MATE II is CE compliant.

The PRO MATE II has programmable VDD and VPP supplies which allows it to verify programmed memory at VDD min and VDD 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

TABLE 12-7: A/D CONVERTER CHARACTERISTICS:
PIC12C671/672-04/PIC12CE673/674-04 (COMMERCIAL, INDUSTRIAL, EXTENDED)
PIC12C671/672-10/PIC12CE673/674-10 (COMMERCIAL, INDUSTRIAL, EXTENDED)
PIC12LC671/672-04/PIC12LCE673/674-04 (COMMERCIAL, INDUSTRIAL)

Param No.	Sym	Characteristic	Min	Typ†	Max	Units	Conditions	
A01	NR	Resolution	—	—	8-bits	bit	$V_{REF} = V_{DD} = 5.12V$, $V_{SS} \leq V_{AIN} \leq V_{REF}$	
A02	EABS	Total absolute error	—	—	$< \pm 1$	LSb	$V_{REF} = V_{DD} = 5.12V$, $V_{SS} \leq V_{AIN} \leq V_{REF}$	
A03	EIL	Integral linearity error	—	—	$< \pm 1$	LSb	$V_{REF} = V_{DD} = 5.12V$, $V_{SS} \leq V_{AIN} \leq V_{REF}$	
A04	EDL	Differential linearity error	—	—	$< \pm 1$	LSb	$V_{REF} = V_{DD} = 5.12V$, $V_{SS} \leq V_{AIN} \leq V_{REF}$	
A05	EFS	Full scale error	—	—	$< \pm 1$	LSb	$V_{REF} = V_{DD} = 5.12V$, $V_{SS} \leq V_{AIN} \leq V_{REF}$	
A06	E0FF	Offset error	—	—	$< \pm 1$	LSb	$V_{REF} = V_{DD} = 5.12V$, $V_{SS} \leq V_{AIN} \leq V_{REF}$	
A10	—	Monotonicity	—	guaranteed (Note 3)	—	—	$V_{SS} \leq V_{AIN} \leq V_{REF}$	
A20	VREF	Reference voltage	2.5V	—	$V_{DD} + 0.3$	V		
A25	VAIN	Analog input voltage	$V_{SS} - 0.3$	—	$V_{REF} + 0.3$	V		
A30	ZAIN	Recommended impedance of analog voltage source	—	—	10.0	k Ω		
A40	IAD	A/D conversion current (VDD)	PIC12C67X	—	180	—	μA	Average current consumption when A/D is on. (Note 1)
			PIC12LC67X	—	90	—	μA	
A50	IREF	VREF input current (Note 2)	10	—	1000	μA	During VAIN acquisition. Based on differential of VHOLD to VAIN to charge CHOLD, see Section 8.1.	
			—	—	10	μA	During A/D Conversion cycle	

* These parameters are characterized but not tested.

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

Note 1: When A/D is off, it will not consume any current other than minor leakage current. The power-down current spec includes any such leakage from the A/D module.

2: VREF current is from GP1 pin or VDD pin, whichever is selected as reference input.

3: The A/D conversion result never decreases with an increase in the Input Voltage, and has no missing codes.

13.0 DC AND AC CHARACTERISTICS - PIC12C671/PIC12C672/PIC12LC671/ PIC12LC672/PIC12CE673/PIC12CE674/PIC12LCE673/PIC12LCE674

The graphs and tables provided in this section are for design guidance and are not tested. In some graphs or tables the data presented are outside specified operating range (i.e., outside specified V_{DD} range). This is for information only and devices will operate properly only within the specified range.

The data presented in this section is a statistical summary of data collected on units from different lots over a period of time. "Typical" represents the mean of the distribution while "max" or "min" represents (mean + 3 σ) and (mean - 3 σ) respectively, where σ is standard deviation.

FIGURE 13-1: CALIBRATED INTERNAL RC FREQUENCY RANGE VS. TEMPERATURE (V_{DD} = 5.0V) (INTERNAL RC IS CALIBRATED TO 25°C, 5.0V)

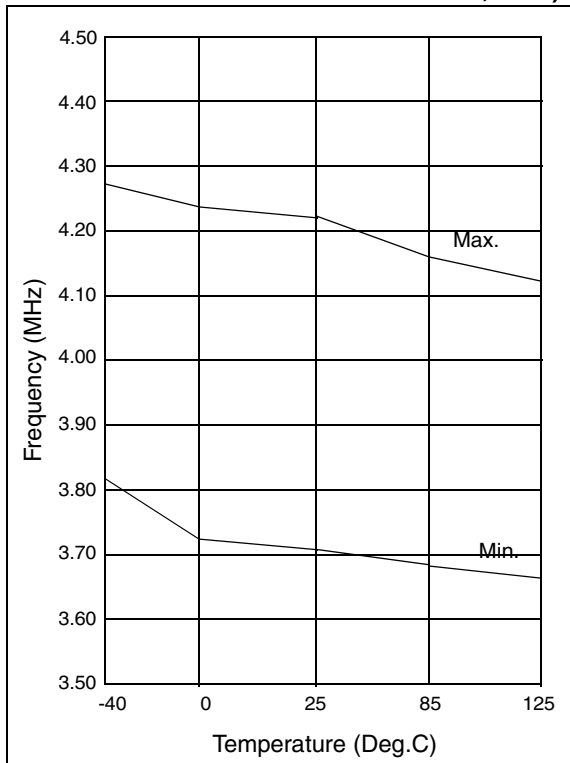
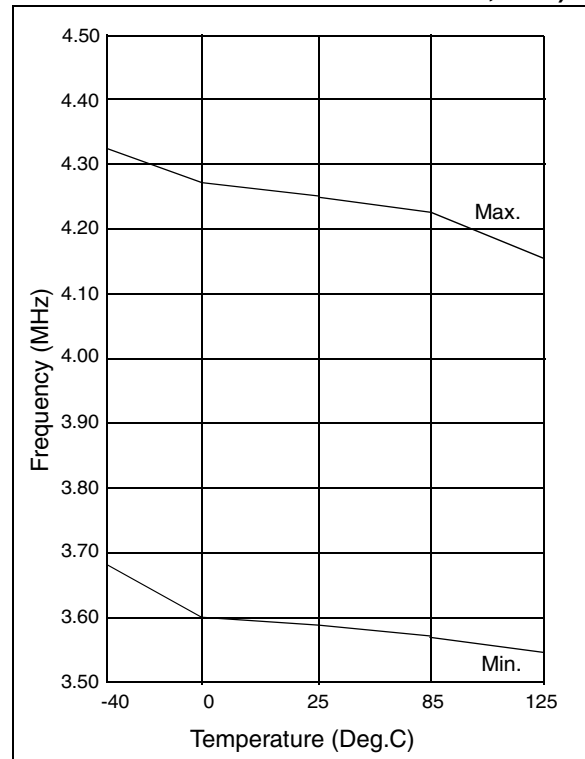


FIGURE 13-2: CALIBRATED INTERNAL RC FREQUENCY RANGE VS. TEMPERATURE (V_{DD} = 2.5V) (INTERNAL RC IS CALIBRATED TO 25°C, 5.0V)



PIC12C67X

FIGURE 13-9: I_{OL} vs. V_{OL} , $V_{DD} = 5.5\text{ V}$

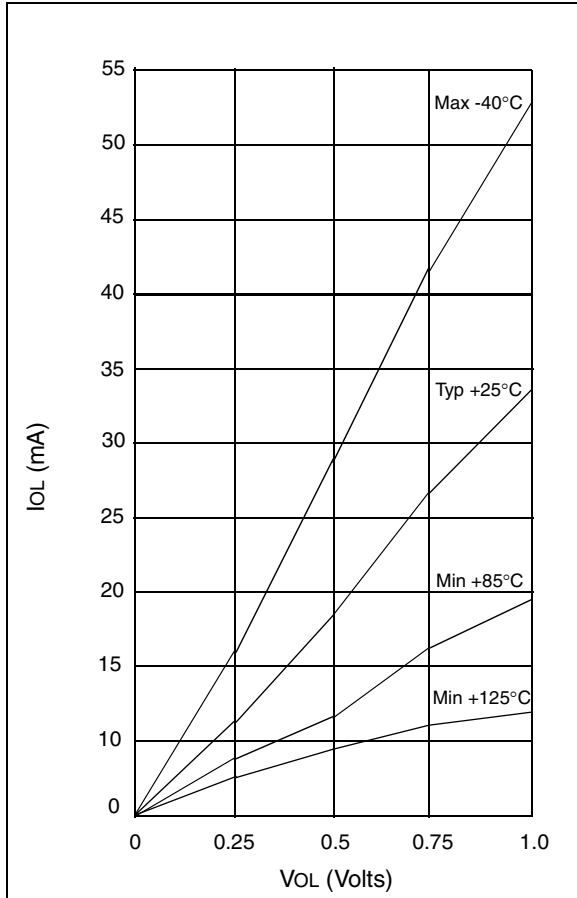


FIGURE 13-10: V_{TH} (INPUT THRESHOLD VOLTAGE) OF GPIO PINS vs. V_{DD}

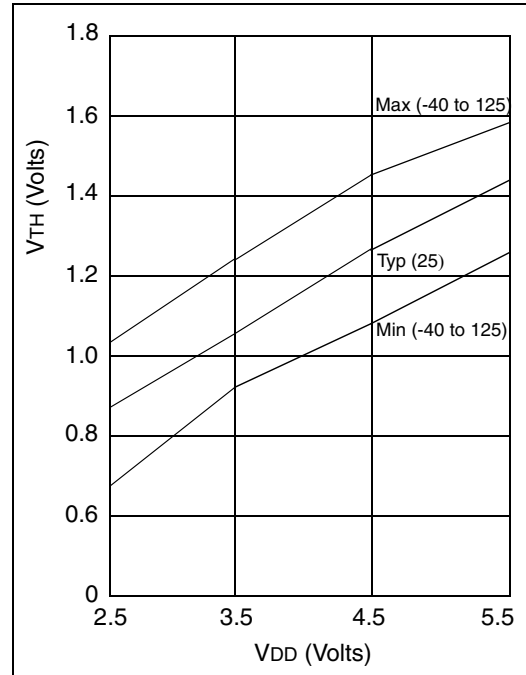


FIGURE 13-11: V_{IL} , V_{IH} OF NMCLR AND T0CKI vs. V_{DD}

