

## Using the Analog-to-Digital (A/D) Converter

*Authors: Sumit Mitra,  
Stan D'Souza, and  
Russ Cooper  
Microchip Technology Inc.*

### INTRODUCTION

This application note is intended for PIC16C7X users with some degree of familiarity with analog system design. The various sections discuss the following topics:

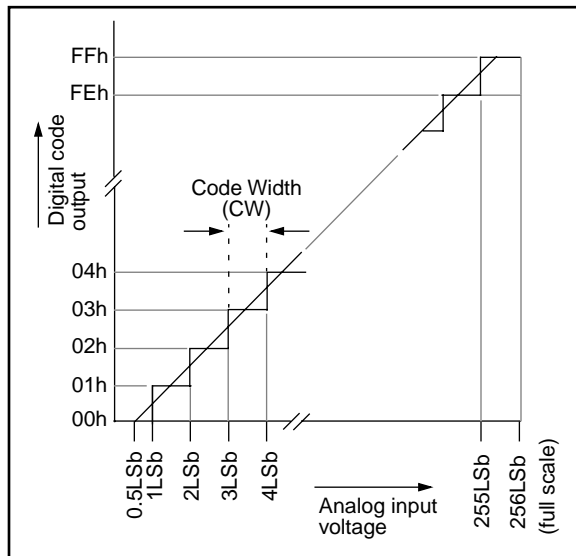
- Commonly used A/D terminology
- How to configure and use the PIC16C71 A/D
- Various ways to generate external reference voltage (VREF)
- Configuring the RA3:RA0 pins

### COMMONLY USED A/D TERMINOLOGY

#### The Ideal Transfer Function

In an A/D converter, an analog voltage is mapped into an N-bit digital value. This mapping function is defined as the transfer function. An ideal transfer is one in which there are no errors or non-linearity. It describes the "ideal" or intended behavior of the A/D. Figure 1 shows the ideal transfer function for the PIC16C7X A/D.

**FIGURE 1: PIC16C7X IDEAL TRANSFER FUNCTION**



Note that the digital output value is 00h for the analog input voltage range of 0 to 1LSb. In some converters, the first transition point is at 0.5LSb and not at 1LSb as shown in Figure 2. Either way, by knowing the transfer function the user can appropriately interpret the data.

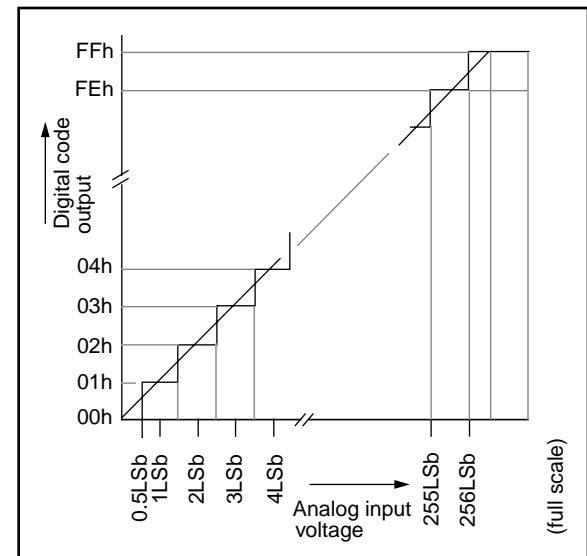
#### Transition Point

The analog input voltage at which the digital output switches from one code to the next is called the "Transition Point." The transition point is typically not a single threshold, but rather a small region of uncertainty (Figure 3). The transition point is therefore defined as the statistical average of many conversions. Stated differently, it is the voltage input at which the uncertainty of the conversion is 50%.

#### Code Width

The distance (voltage differential) between two transition points is called the "Code Width." Ideally the Code Width should be 1LSb (Figure 1).

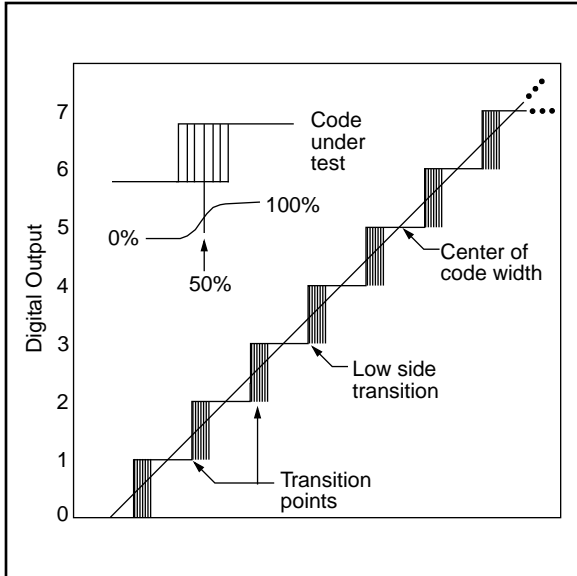
**FIGURE 2: ALTERNATE TRANSFER FUNCTION**



## Center of Code Width

The midpoint between two transition points is called the "Center of Code Width" (Figure 3).

**FIGURE 3: TRANSITION POINTS**



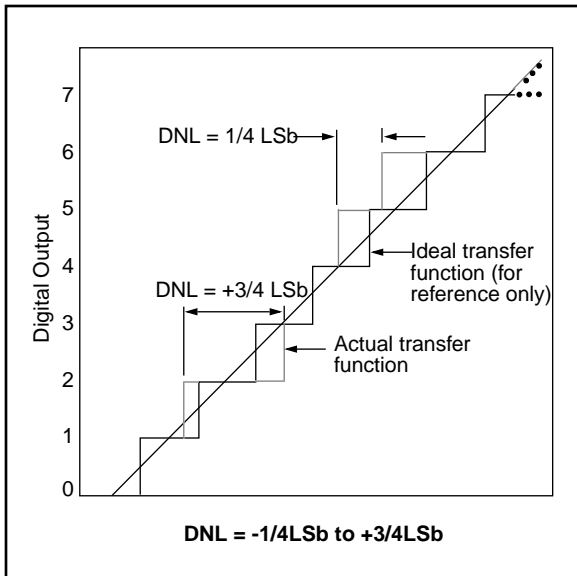
## Differential Non-Linearity (DNL)

It is the deviation in code-width from 1LSb (Figure 4). The difference is calculated for each and every transition. The largest difference is reported as DNL.

It is important to note that the DNL is measured after the transfer function is normalized to match offset error and gain error.

Note that the DNL cannot be any less than  $-1\text{LSb}$ . In the other direction, DNL can be  $>1\text{LSb}$ .

**FIGURE 4: DIFFERENTIAL NON-LINEARITY**



## Absolute Error

The maximum deviation between any transition point from the corresponding ideal transfer function is defined as the absolute error. This is how it is measured and reported in the PIC16C7X (Figure 5). The notable difference between absolute error and integral non-linearity (INL) is that the measured data is not normalized for full scale and offset errors in absolute error.

Absolute Error is probably the first parameter the user will review to evaluate an A/D. Sometimes absolute error is reported as the sum of offset, full-scale and integral non-linearity errors.

## Total Unadjusted Error

Total Unadjusted Error is the same as absolute error. Again, sometimes it is reported as the sum of offset, full-scale and integral non-linearity errors.

## No Missing Code

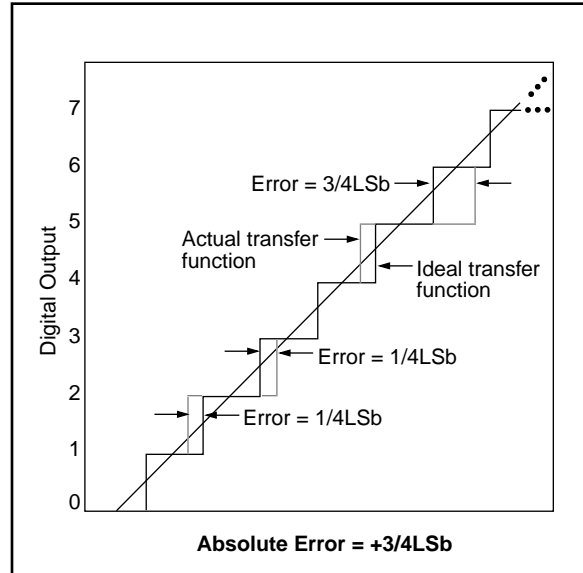
No missing code implies that as the analog input voltage is gradually increased from zero to full scale (or vice versa), all digital codes are produced. Stated otherwise, changing analog input voltage from one quantum of the analog range to the next adjacent range will not produce a change in the digital output by more than one code count.

## Monotonic

Monotonicity guarantees that an increase (or decrease) in the analog input value will result in an equal or greater digital code (or less). Monotonicity does not guarantee that there are no missing codes. However, it is an important criterion for feedback control systems. Non-monotonicity may cause oscillations in such systems.

The first derivative of a monotonic function always has the same sign.

**FIGURE 5: ABSOLUTE ERROR**



## Ratiometric Conversion

Ratiometric Conversion is the A/D conversion process in which the binary result is a ratio of the supply voltage or reference voltage, the latter being equal to full-scale value by default. The PIC16C7X is a ratiometric A/D converter where the result depends on VDD or VREF.

In some A/Ds, an absolute reference is provided resulting in “absolute conversion”.

## Sample and Hold

In sample and hold type A/D converters, the analog input has a switch (typically a FET switch in CMOS) which is opened for a short duration to capture the analog input voltage onto an on-chip capacitor. Conversion is typically started after the sampling switch is closed.

## Track and Hold

Track and Hold is basically the same as sample and hold, except the sampling switch is typically left on. Therefore the voltage on the on-chip holding capacitor “tracks” the analog input voltage. To begin a conversion, the sampling switch is closed.

The PIC16C7X A/D falls in this category.

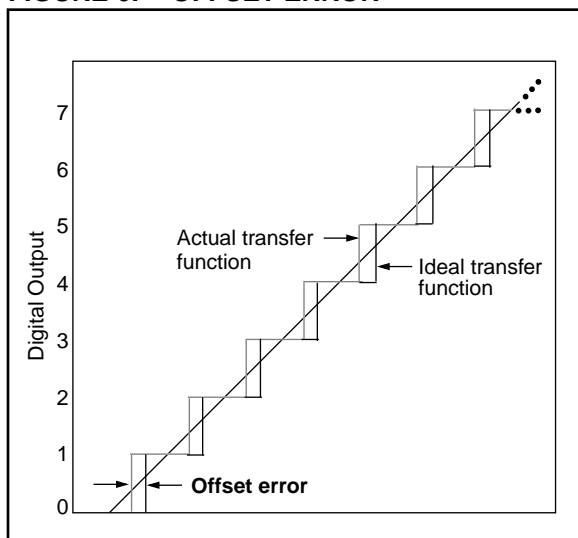
## Sampling Time

Sampling Time is the time required to charge the on-chip holding capacitor to the same value as is on the analog input pin. The sampling time depends on the magnitude of the holding capacitor and the source impedance of the analog voltage input.

## Offset Error (or Zero Error)

Offset Error is the difference between the first actual (measured) transition point and the first ideal transition point as shown in Figure 6. It can be corrected (by the user) by subtracting the offset error from each conversion result.

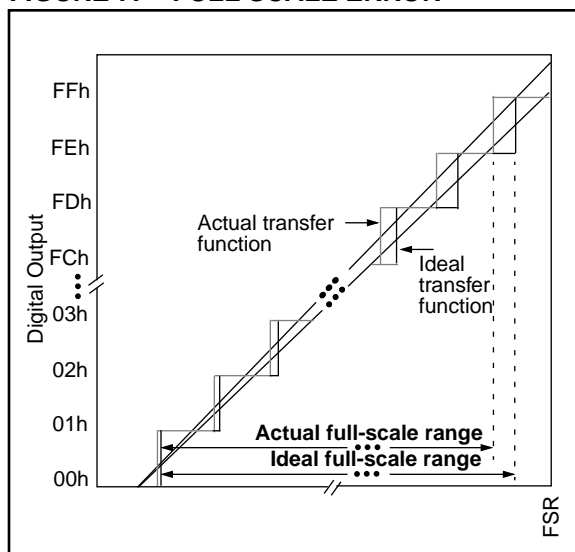
**FIGURE 6: OFFSET ERROR**



## Full Scale Error (or Gain Error)

Full Scale Error is the difference between the ideal full scale and the actual (measured) full scale range (Figure 7). It is also called gain error, because the error changes the slope of the ideal transfer function creating a gain factor. It can be corrected (by the user) by multiplying each conversion result by the inverse of the gain.

**FIGURE 7: FULL SCALE ERROR**

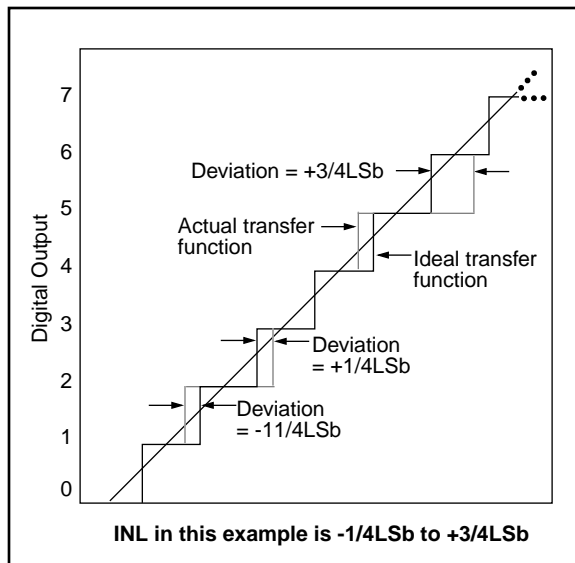


## Integral Non-Linearity (INL), or Relative Error

The deviation of a transition point from its corresponding point on the ideal transfer curve is called “Integral Non-Linearity” (Figure 8). The maximum difference is reported as the INL of the converter.

It is important to note that Full Scale Error and the Offset Error are normalized to match end transition points before measuring the INL.

**FIGURE 8: INTEGRAL NON-LINEARITY**



## HOW TO USE THE PIC16C71 A/D

The A/D in the PIC16C71 is easy to set up and use. There are a few considerations:

1. Select either VDD or VREF as reference voltage. (More on using VREF input later)  
Select A/D conversion clock (TAD): 2TOSC, 8TOSC, TOSC or TRC (internal RC clock). For the first three options, make sure that  $TAD \geq 2.0 \mu s$ . If deterministic conversion time is required, select TOSC time-base. If conversion during SLEEP is required, select TRC.
2. Channel Selection: If only one A/D channel is required, program the ADCON1 register to 03h. This configures the A/D pins as digital I/O. If multiple channels are required, prior to each conversion the new channel must be selected.
3. Sampling and Conversion: After a new channel is selected, a minimum amount of sampling time must be allowed before the GO/DONE bit in ADCON0 is set to begin conversion. Once conversion begins, it is OK to select the next channel, **but sampling does not begin until current conversion is complete**. Therefore, it is always necessary to ensure the minimum sampling time is provided for:
  - i) after a conversion
  - ii) after a new channel is selected
  - iii) after A/D is turned on (bit ADON = 1)
4. Reading Result: Completion of a conversion can be determined by polling the GO/DONE bit (cleared), or polling flag bit ADIF (set), or waiting for an ADIF interrupt.

### Additional tips:

- a) Do not set bits GO/DONE and ADON in the same instruction. First, turn the A/D is on by setting bit ADON. Then allow at least  $5 \mu s$  before conversion begins (setting the GO/DONE bit), longer if sampling time requirement is not met within  $5 \mu s$ .
- b) Aborting a conversion: A conversion can be aborted by clearing bit GO/DONE. The A/D converter will stop conversion and revert back to sampling state.
- c) Using the ADRES register as a normal register: The A/D only writes to the ADRES register at the end of a conversion. Therefore, it is possible to use the ADRES register as a normal file register between conversions and when A/D is off.

The following four examples provide sample code on using the A/D module.

### EXAMPLE 1: HOW TO DO A SAMPLE A/D CONVERSION

```
;
; InitializeAD, initializes and sets up the A/D hardware.
; Always ch2, internal RC OSC.
InitializeAD
    bsf     STATUS, 5      ; select Bank1
    movlw  b'00000000'    ; select RA3-RA0
    movwf  ADCON1         ; as analog inputs
    bcf    STATUS, 5      ; select Bank0
    movlw  b'11010001'    ; select: RC osc, ch2...
    movwf  ADCON0         ; turn on A/D
Convert  call  sample-delay ; provide necessary sampling time
;
    bsf    ADCON0, 2      ; start new A/D conversion
loop
    btfsc  ADCON0, 2      ; A/D over?
    goto   loop           ; no then loop
;
    movf   adres, w       ; yes then get A/D value
;
```

A detailed code listing is provided in Appendix A.

**EXAMPLE 2: SEQUENTIAL CHANNEL CONVERSIONS**

```

;
; InitializeAD, initializes and sets up the A/D hardware.
; Select ch0 to ch3 in a round robin fashion, internal RC OSC.
; Load results in 4 consecutive addresses starting at ADTABLE (10h)
;
InitializeAD
    bsf     STATUS, RP0      ; select Bank1
    movlw  b'00000000'      ; select RA3-RA0
    movwf  ADCON1           ; as analog inputs
    bcf     STATUS, RP0      ; select Bank0
    movlw  b'11000001'      ; select: RC osc, ch0...
    movwf  ADCON0           ; turn on A/D
    movlw  ADTABLE          ; point fsr to top of...
    movwf  FSR              ; table
;
new_ad    call    sample_delay ; provide necessary sampling time
          bsf     ADCON0, GO    ; start new A/D conversion
loop
          btfsc  ADCON0, GO    ; A/D over?
          goto   loop          ; no then loop
;
          movf   adres, w      ; yes then get A/D value
          movwf  0             ; load indirectly
          movlw  4             ; select next channel
          addwf  ADCON0        ; /
          bcf    ADCON0, ADIF  ; reset interrupt flag bit.
; increment pointer to correct table offset.
          clrf   temp          ; clear temp register
          btfsc  ADCON0, CH50  ; test lsb of channel select
          bsf    temp, 0       ; set if ch1 selected
          btfsc  ADCON0, CH51  ; test msb of channel select
          bsf    temp, 1       ; /
          movlw  ADTABLE       ; get table address
          addwf  temp, w        ; add with temp
          movwf  FSR           ; move into indirect
          goto   new_ad
;

```

A detailed code listing is provided in Appendix B.

## EXAMPLE 3: SAMPLE INTERRUPT HANDLER FOR THE A/D

```
    org     0x00
    goto   start
    org     0x04
    goto   service_ad      ; interrupt vector
;
;
    org     0x10
start
    movlw  b'00000000'    ;init I/O ports
    movwf  PORT_B
    tris   PORT_B
;
    call   InitializeAD
update
    bcf    flag, adover   ; reset software A/D flag
    call   SetupDelay    ; setup delay >= 10uS.
    bcf    ADCON0, adif   ; reset A/D int flag (ADIF)
    bsf    ADCON0, go     ; start new A/D conversion
    bsf    INTCON, gie    ; enable global interrupt
loop
    btfsc  flag, adover   ; A/D over?
    goto   update        ; yes start new conv.
    goto   loop          ; no then keep checking
; InitializeAD, initializes and sets up the A/D hardware.
; select ch0 to ch3, RC OSC., a/d interrupt.
InitializeAD
    bsf    STATUS, RP0    ; select Bank1
    movlw  b'00000000'    ; select RA0-RA3...
    movwf  ADCON1        ; as analog inputs
    bcf    STATUS, RP0    ; select Bank0
    clrf   INTCON        ; clr all interrupts
    bsf    INTCON, ADIE   ; enable A/D int.
    movlw  b'11010001'    ; select: RC osc, ch2...
    movwf  ADCON0        ; turn on A/D
    return
;
service_ad
    btfss  ADCON0, ADIF   ; A/D interrupt?
    retfie ; no then ignore
    movf   ADRES, W      ; get A/D value
    return               ; do not enable int
;
```

A detailed code listing is provided in Appendix C.

**EXAMPLE 4: CONVERSIONS DURING SLEEP MODE**

```
;
; InitializeAD, initializes and sets up the A/D hardware.
; Select ch0 to ch3, internal RC OSC.
; While doing the conversion put unit to sleep. This will
; minimize digital noise interference.
; Note that A/D's RC osc. has to be selected in this instance.
;
InitializeAD
    bsf        STATUS, RP0        ; select Bank1
    movlw     b'00000000'        ; select RA0-RA3...
    movwf    ADCON1              ; as analog inputs
    bcf        STATUS, RP0        ; select Bank0
    movlw     b'11000001'        ; select: RC osc, ch0...
    movwf    ADCON0              ; turn on A/D & ADIE
    movlw     ADTABLE            ; point fsr to top of...
    movwf    FSR                 ; table
;
new_ad
    bsf        ADCON0, GO        ; start new A/D conversion
    sleep     ; goto sleep
; when A/D is over program will continue from here
;
    movf     ADRES, w           ; get A/D value
;
```

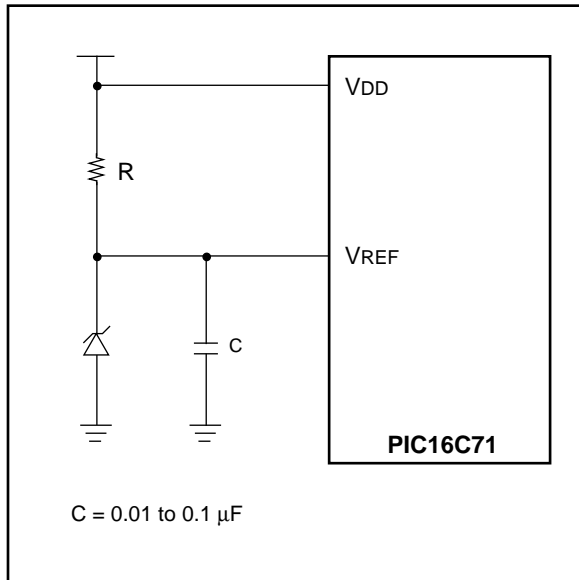
A detailed code listing is provided in Appendix D.

## USING EXTERNAL REFERENCE VOLTAGE

When using the external reference voltage, keep in mind that any analog input voltage must not exceed  $V_{REF}$ .

An inexpensive way to generate  $V_{REF}$  is by employing a zener diode (Figure 9). Most common zener diodes offer 5% accuracy. Reverse bias current may be as low as  $10\ \mu\text{A}$ . However, larger currents (1 mA - 20 mA) are recommended for stability, as well as lower impedance of the  $V_{REF}$  source.

**FIGURE 9: LOW COST VOLTAGE REFERENCE**



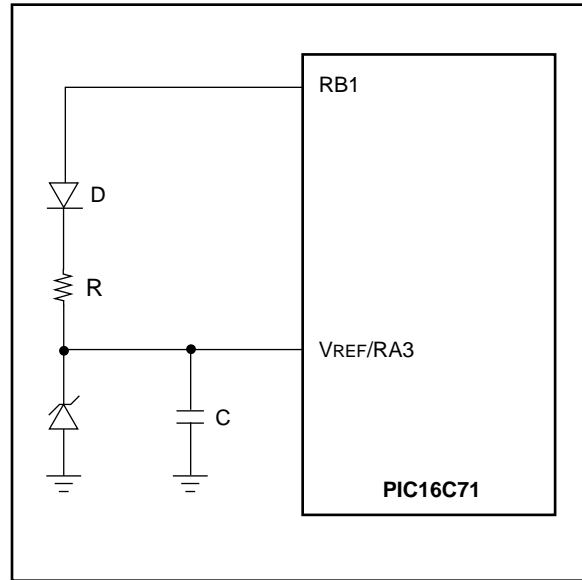
### POWER MANAGEMENT IN USING $V_{REF}$

In power sensitive applications, the user may turn on a  $V_{REF}$  generator using another I/O pin (Figure 10). Drive a '1' on pin RB1, in this example, when using the A/D. Drive a '0' on pin RB1 when not using the A/D converter.

Note that this way RB1 is not floating. Even if  $V_{REF}$  decays to some intermediate voltage, it will not cause the input buffer on RB1 to draw current.

Alternately, use RA0, RA1 or RA2 pin to supply the current instead of RB1. Configure the RA pin as analog (this will turn off its input buffer). Then use it as a digital output (Figure 11).

**FIGURE 10: POWER-SENSITIVE APPLICATIONS #1**



### ZENERS AND REFERENCE GENERATORS

Finally, various reference voltage generator chips (typically using on-chip band-gap reference) are available. They are more accurate.

**TABLE 1: ZENERS AND REFERENCE GENERATORS**

Zeners	$V_z$	Tolerance
1N746	3.3V	$\pm 5\%$
1N747	3.6V	$\pm 5\%$
1N748	3.9V	$\pm 5\%$
1N749	4.3V	$\pm 5\%$
1N750	4.7V	$\pm 5\%$
1N751	5.1V	$\pm 5\%$
1N752	5.6V	$\pm 5\%$
Voltage Reference	$V_{REF}$	Tolerance
AD580 (Maxim)	2.5V	$\pm 3\%$ to $\pm 0.4\%$
LM385	2.5V	$\pm 1.5\%$
LM1004	2.5V	$\pm 1.2\%$
LT1009 (LIN. Tech.)	2.5V	$\pm 0.2\%$
LT1019 (LIN. Tech.)	5.0V	$\pm 0.2\%$
LT1021 (LIN. Tech.)	5.0V	$\pm 0.05\%$ to $\pm 1\%$
LT1029 (LIN. Tech.)	5.0V	$\pm 0.2\%$ to $\pm 1\%$



## VREF IMPEDANCE AND CURRENT SUPPLY REQUIREMENTS

Ideally, VREF should have as low a source impedance as possible. Referring to Figure 9, VREF source impedance  $\approx R$ . However, smaller R increases current consumption. Since VREF is used to charge capacitor arrays inside the A/D converter and the holding capacitor,  $Chold \approx 51 \text{ pF}$ , the following guideline should be met:

$$T_{AD} = 6(1k + R)51.2pF + 1.677\mu s$$

$T_{AD}$  = conversion clock. For  $T_{AD} = 2 \mu s$  and for  $CHOLD = 50 \text{ pF}$ ,  $V_{REF} \approx 50\Omega$ .

For VREF impedance higher than this, the conversion clock ( $T_{AD}$ ) should be increased appropriately.

**FIGURE 11: POWER-SENSITIVE APPLICATIONS #2**

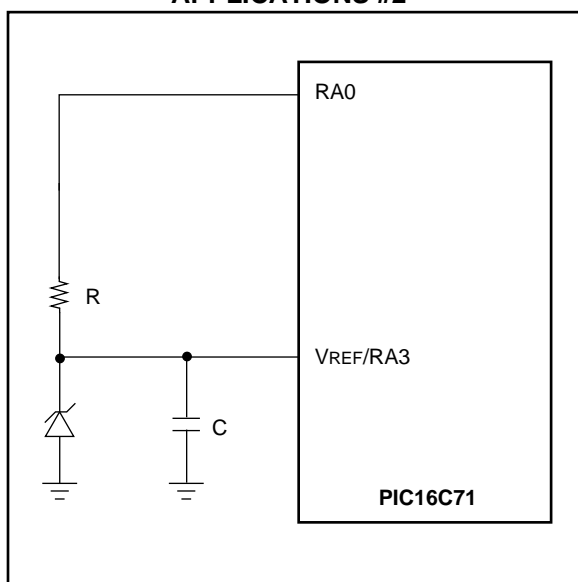


Table 2 gives examples of the maximum rate of conversion per bit, relating to the voltage reference impedance.

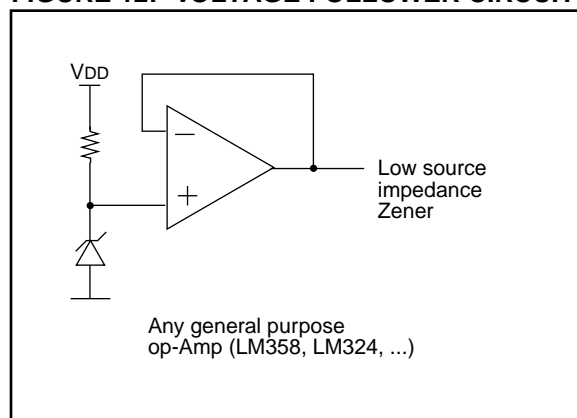
**TABLE 2: MAXIMUM RATE OF CONVERSION / BIT**

RVREF	TAD (Max)
1k	2.29 $\mu s$
5k	3.52 $\mu s$
10k	5.056 $\mu s$
50k	16.66 $\mu s$
100k	32.70 $\mu s$

Assumes no external capacitors

To achieve a low source impedance when using a Zener diode, a voltage follower circuit is recommended. This is shown in Figure 12.

**FIGURE 12: VOLTAGE FOLLOWER CIRCUIT**



## CONFIGURING PORTA INPUTS AS ANALOG OR DIGITAL

Two bits in the ADCON1 register, PCFG1 and PCFG0, control how pins RA3:RA0 are configured.

When any of these pins are selected as analog:

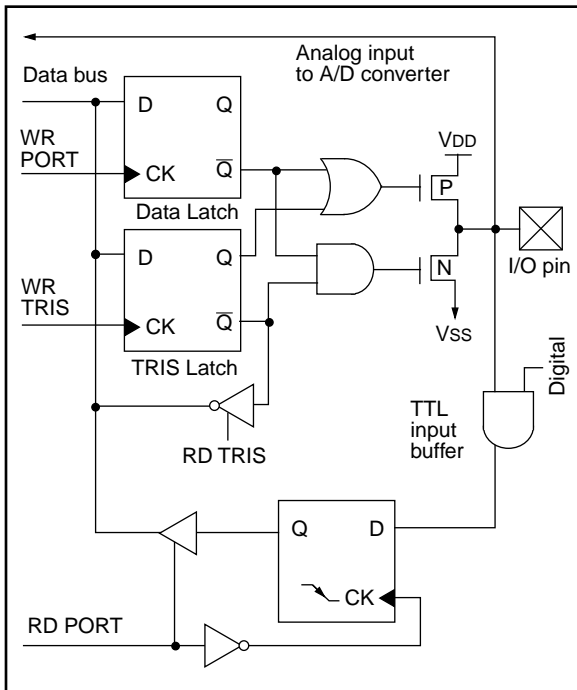
- The digital input buffer is turned off to save current (Figure 13). Reading the port will read this pin as '0'.
- The TRIS bit still controls the output buffer on this pin. So, normally the TRIS bit will be set (input).
- However, if the TRIS bit is cleared, then the pin will output whatever is in the data latch.

When any of these pins are selected as digital:

- The analog input still directly connects to the A/D and therefore the pin can be used as analog input.
- The digital input buffer is not disabled.

The user has, therefore, great flexibility in configuring these pins.

**FIGURE 13: BLOCK DIAGRAM OF RA3:RA0 PINS**



## CURRENT CONSUMPTION THROUGH INPUT BUFFER

A CMOS input buffer will draw current when the input voltage is near its threshold (Figure 14).

In power-sensitive applications, the RA pins, when used as analog inputs, should be configured as "analog" to avoid unintended power drain.

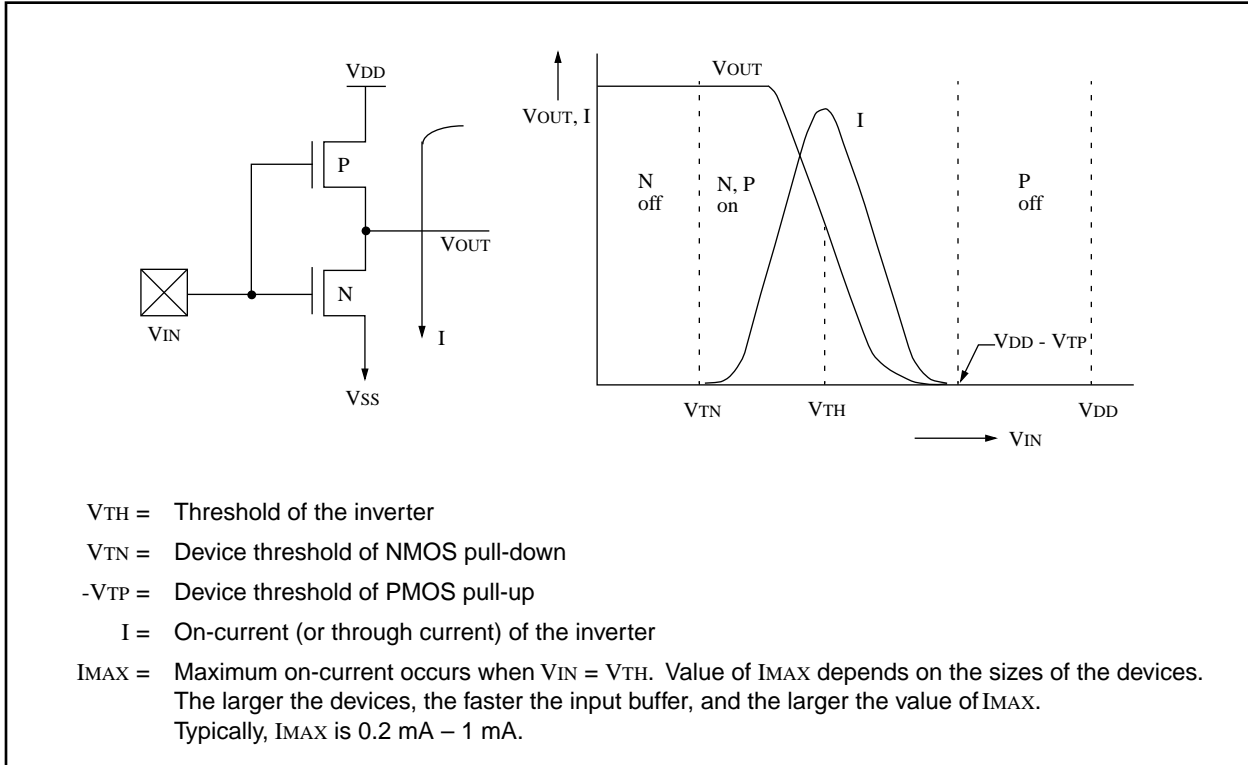
Other considerations and tips:

1. If possible, avoid any digital output next to analog inputs.
2. Avoid digital inputs that switch frequently (e.g., clocks) next to analog inputs.
3. If  $V_{REF}$  is used, then ensure that no analog pin being sampled exceeds  $V_{REF}$ .

## SUMMARY

The PIC16C71 A/D converter is simple to use. It is versatile and has low power consumption.

**FIGURE 14: A SIMPLE CMOS INPUT BUFFER**



Please check the Microchip BBS for the latest version of the source code. Microchip's Worldwide Web Address: [www.microchip.com](http://www.microchip.com); Bulletin Board Support: MCHIPBBS using CompuServe® (CompuServe membership not required).

## APPENDIX A: SINGLE CHANNEL A/D (SAD)

MPASM 01.40 Released

SAD.ASM 1-16-1997 15:22:04

PAGE 1

```

LOC  OBJECT CODE      LINE SOURCE TEXT
VALUE
                                00001 ;TITLE      "Single channel A/D (SAD)"
                                00002 ;This program is a simple implementation of the PIC16C71's
                                00003 ;A/D. 1 Channel is selected (CH0).
                                00004 ;The A/D is configured as follows:
                                00005 ;      Vref = +5V internal.
                                00006 ;      A/D Osc. = internal RC
                                00007 ;      A/D Channel = CH0
                                00008 ;Hardware for this program is the PICDEM1 board.
                                00009 ;
                                00010 ;
                                00011 ;      Program:          SAD.ASM
                                00012 ;      Revision Date:
                                00013 ;                  1-14-97      Compatibility with MPASMWIN 1.40
                                00014 ;
                                00015 ;
                                00016      LIST P=16C71
                                00017      ERRORLEVEL -302
                                00018 ;
                                00019      include "pl6c71.inc"
                                00001      LIST
                                00002 ;P16C71.INC Standard Header File, Version 1.00 Microchip Technology
                                00142      LIST
                                00020 ;
00000010      00021 TEMP EQU 10h
00000001      00022 adif equ 1
00000002      00023 adgo equ 2
                                00024 ;
0000      00025      ORG 0x00
                                00026 ;
                                00027 ;
0000 2810      00028      goto start
                                00029 ;
0004      00030      org 0x04
0004 281E      00031      goto service_int ;interrupt vector
                                00032 ;
                                00033 ;
0010      00034      org 0x10
0010      00035 start
0010 3000      00036      movlw B'00000000' ;set port b as
0011 0086      00037      movwf PORTB ;all outputs
                                00038 ;      tris PORTB ; /
0012 1683      00039      BSF STATUS, RP0 ; Bank1
0013 0086      00040      MOVWF TRISB ; PortB as outputs
0014 1283      00041      BCF STATUS, RP0 ; Bank0
                                00042 ;
0015 201F      00043      call InitializeAD
0016      00044 update
0016 0809      00045      movf ADRES,W ;get a/d value
0017 0086      00046      movwf PORTB ;output to port b
0018 2027      00047      call SetupDelay ;setup time >= 10uS.
0019 1088      00048      bcf ADCON0,adif ;clear int flag
001A 1508      00049      bsf ADCON0,adgo ;start new conversion
001B      00050 loop
001B 1888      00051      btfsc ADCON0,adif ;a/d done?
001C 2816      00052      goto update ;yes then update new value.

```

# AN546

```
001D 281B      00053      goto    loop           ;no then keep checking
                00054 ;
                00055 ;no interrupts are enabled, so if the program ever reaches here,
                00056 ;it should be returned with the global interrupts disabled.
001E          00057 service_int
001E 0008      00058      return           ;do not enable global.
                00059 ;
                00060 ;
                00061 ;
                00062 ;InitializeAD, initializes and sets up the A/D hardware.
                00063 ;Select ch0 to ch3 as analog inputs, fosc/2 and read ch3.
                00064 ;
001F          00065 InitializeAD
001F 1683      00066      bsf     STATUS,5       ;select Bank1
0020 3000      00067      movlw   B'00000000'     ;select ch0-ch3...
0021 0088      00068      movwf  ADCON1         ;as analog inputs
0022 1283      00069      bcf     STATUS,5       ;select Bank0
0023 30C1      00070      movlw   B'11000001'     ;select:RC,ch0..
0024 0088      00071      movwf  ADCON0         ;turn on A/D.
0025 0189      00072      clrf   ADRES          ;clr result reg.
0026 0008      00073      return
                00074 ;
                00075 ;This routine is a software delay of 10uS for the a/d setup.
                00076 ;At 4Mhz clock, the loop takes 3uS, so initialize TEMP with
                00077 ;a value of 3 to give 9uS, plus the move etc should result in
                00078 ;a total time of > 10uS.
0027          00079 SetupDelay
0027 3003      00080      movlw   .3
0028 0090      00081      movwf  TEMP
0029          00082 SD
0029 0B90      00083      decfsz TEMP, F
002A 2829      00084      goto   SD
002B 0008      00085      return
                00086
                00087
                00088      END
```

MEMORY USAGE MAP ('X' = Used, '-' = Unused)

```
0000 : X---X----- XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXX-----
```

All other memory blocks unused.

```
Program Memory Words Used:    30
Program Memory Words Free:   994
```

```
Errors   :    0
Warnings :    0 reported,    0 suppressed
Messages :    0 reported,    2 suppressed
```

Please check the Microchip BBS for the latest version of the source code. Microchip's Worldwide Web Address: [www.microchip.com](http://www.microchip.com); Bulletin Board Support: MCHIPBBS using CompuServe® (CompuServe membership not required).

## APPENDIX B: SLPAD.ASM

MPASM 01.40 Released

SLPAD.ASM 1-16-1997 15:22:32

PAGE 1

```

LOC  OBJECT CODE      LINE SOURCE TEXT
VALUE

00001
00002 ;TITLE      "A/D in Sleep Mode"
00003 ;This program is a simple implementation of the PIC16C71's
00004 ;A/D feature. This program demonstrates
00005 ;how to do a a/d in sleep mode on the PIC16C71.
00006 ;The A/D is configured as follows:
00007 ;      Vref = +5V internal.
00008 ;      A/D Osc. = internal RC
00009 ;      A/D Interrupt = OFF
00010 ;      A/D Channels = ch 0
00011 ;
00012 ;The ch0 A/D result is displayed as a 8 bit binary value
00013 ;on 8 leds connected to port b. Hardware used is that of
00014 ;the PICDEMO board.
00015 ;
00016 ;
00017 ;      Program:          SLPAD.ASM
00018 ;      Revision Date:
00019 ;                      1-14-97      Compatibility with MPASMWIN 1.40
00020 ;
00021 ;
00022      LIST P=16C71
00023      ERRORLEVEL -302
00024 ;
00025      include "p16c71.inc"
00001      LIST
00002 ;P16C71.INC Standard Header File, Version 1.00 Microchip Technology
00142      LIST
00026 ;
00000010      00027 TEMP      EQU      10h
00000001      00028 adif      equ      1
00000002      00029 adgo      equ      2
00030 ;
00031 ;
0000      00032      ORG      0x00
00033 ;
00034 ;
0000 2810      00035      goto      start
00036 ;
0004      00037      org      0x04
0004 281D      00038      goto      service_int      ;interrupt vector
00039 ;
00040 ;
0010      00041      org      0x10
0010      00042 start
0010 3000      00043      movlw   B'00000000'      ;make port b all
0011 0086      00044      movwf   PORTB      ;outputs.
00045 ;
0012 1683      00046      BSF     STATUS, RP0      ; Bank1
0013 0086      00047      MOVWF  TRISB      ; PortB as outputs
0014 1283      00048      BCF     STATUS, RP0      ; Bank0
00049 ;
0015 201E      00050      call   InitializeAD
0016      00051 update

```

# AN546

```
0016 0809      00052      movf    ADRES,W
0017 0086      00053      movwf   PORTB      ;save in table
0018 2027      00054      call   SetupDelay  ;
0019 1088      00055      bcf    ADCON0,adif ;clr a/d flag
001A 1508      00056      bsf    ADCON0,adgo ;start new a/d conversion
                   00057 ;
001B 0063      00058      sleep
001C 2816      00059      goto   update      ;wake up and update
                   00060 ;
001D           00061      service_int
001D 0008      00062      return           ;do not enable int
                   00063 ;
                   00064 ;InitializeAD, initializes and sets up the A/D hardware.
001E           00065      InitializeAD
001E 1683      00066      bsf    STATUS,5    ;select Bank1
001F 3000      00067      movlw  B'00000000' ;select ch0-ch3...
0020 0088      00068      movwf  ADCON1      ;as analog inputs
0021 1283      00069      bcf    STATUS,5    ;select Bank0
0022 30C1      00070      movlw  B'11000001' ;select:internal RC, ch0.
0023 0088      00071      movwf  ADCON0      ;turn on a/d
0024 018B      00072      clrf   INTCON      ;clear all interrupts
0025 170B      00073      bsf    INTCON,ADIE ;enable a/d
0026 0008      00074      return
                   00075 ;
                   00076 ;This routine is a software delay of 10uS for the a/d setup.
                   00077 ;At 4Mhz clock, the loop takes 3uS, so initialize TEMP with
                   00078 ;a value of 3 to give 9uS, plus the move should result in
                   00079 ;a total time of > 10uS.
0027           00080      SetupDelay
0027 3003      00081      movlw  .3
0028 0090      00082      movwf  TEMP
0029           00083      SD
0029 0B90      00084      decfsz TEMP, F
002A 2829      00085      goto   SD
002B 0008      00086      return
                   00087
                   00088 ;
                   00089
002B 0008      00090      END
```

MEMORY USAGE MAP ('X' = Used, '-' = Unused)

```
0000 : X---X----- XXXXXXXXXXXXXXXX XXXXXXXXXXXXX-----
```

All other memory blocks unused.

```
Program Memory Words Used:    30
Program Memory Words Free:   994
```

```
Errors   :    0
Warnings :    0 reported,    0 suppressed
Messages :    0 reported,    2 suppressed
```

Please check the Microchip BBS for the latest version of the source code. Microchip's Worldwide Web Address: [www.microchip.com](http://www.microchip.com); Bulletin Board Support: MCHIPBBS using CompuServe® (CompuServe membership not required).

## APPENDIX C: INTAD.ASM

MPASM 01.40 Released

INTAD.ASM 1-16-1997 15:21:10

PAGE 1

```

LOC  OBJECT CODE      LINE SOURCE TEXT
VALUE

00001
00002 ;TITLE      "Single channel A/D with interrupts"
00003 ;This program is a simple implementation of the PIC16C71's
00004 ;A/D. 1 Channel is selected (CH0). A/D interrupt is turned on,
00005 ;hence on completion of a/d conversion, an interrupt is generated.
00006 ;The A/D is configured as follows:
00007 ;      Vref = +5V internal.
00008 ;      A/D Osc. = internal RC Osc.
00009 ;      A/D Interrupt = On
00010 ;      A/D Channel = CH0
00011 ;
00012 ;The A/D result is displayed as a 8 bit value on 8 leds connected
00013 ;to port b. Hardware setup is the PICDEMO board.
00014 ;
00015 ;
00016 ;      Program:          INTAD.ASM
00017 ;      Revision Date:
00018 ;                      1-14-97      Compatibility with MPASMWIN 1.40
00019 ;
00020 ;
00021      LIST P=16C71
00022      ERRORLEVEL  -302
00023 ;
00024      include "p16c71.inc"
00001      LIST
00002 ; P16C71.INC Standard Header File, Version 1.00 Microchip Technology
00142      LIST
00025 ;
00000010      00026 flag      equ      10
00000011      00027 TEMP      equ      11
00000000      00028 adover     equ      0
00000001      00029 adif      equ      1
00000002      00030 adgo      equ      2
00000006      00031 adie      equ      6
00000007      00032 gie      equ      7
00000005      00033 rp0      equ      5
00034 ;
0000      00035      ORG      0x00
00036 ;
00037 ;
0000 2810      00038      goto     start
00039 ;
0004      00040      org      0x04
0004 281E      00041      goto     service_ad      ;interrupt vector
00042 ;
00043 ;
0010      00044      org      0x10
0010      00045 start
0010 3000      00046      movlw   B'00000000'      ;init i/o ports
0011 0086      00047      movwf   PORTB
00048 ;      tris   PORTB
0012 1683      00049      BSF    STATUS, RP0      ; Bank1
0013 0086      00050      MOVWF  TRISB      ; PortB as outputs
0014 1283      00051      BCF    STATUS, RP0      ; Bank0
00052 ;

```

# AN546

```
0015 2024      00053      call    InitializeAD
0016           00054 update
0016 1010      00055      bcf     flag,adover      ;reset software a/d flag
0017 202D      00056      call    SetupDelay      ;setup delay >= 10uS.
0018 1088      00057      bcf     ADCON0,adif     ;reset a/d int flag (ADIF)
0019 1508      00058      bsf     ADCON0,adgo     ;start new a/d conversion
001A 178B      00059      bsf     INTCON,gie      ;enable global interrupt
001B           00060 loop
001B 1810      00061      btfsc  flag,adover     ;a/d over?
001C 2816      00062      goto   update          ;yes start new conv.
001D 281B      00063      goto   loop            ;no then keep checking
001E           00064 ;
001E           00065 service_ad
001E 1C88      00066      btfss  ADCON0,adif     ;ad interrupt?
001F 0009      00067      retfie                ;no then ignore
0020 0809      00068      movf   ADRES,W         ;get a/d value
0021 0086      00069      movwf  PORTB           ;output to port b
0022 1410      00070      bsf     flag,adover     ;a/d done set
0023 0008      00071      return                ;do not enable int
0024           00072 ;
0024           00073 ;
0024           00074 ;InitializeAD, initializes and sets up the A/D hardware.
0024           00075 ;select ch0 to ch3, RC OSC., a/d interrupt.
0024           00076 InitializeAD
0024 1683      00077      bsf     STATUS,rp0     ;select Bank1
0025 3000      00078      movlw  B'00000000'     ;select ch0-ch3...
0026 0088      00079      movwf  ADCON1          ;as analog inputs
0027 1283      00080      bcf     STATUS,rp0     ;select Bank0
0028 018B      00081      clrf   INTCON          ;clr all interrupts
0029 170B      00082      bsf     INTCON,adie    ;enable a/d int.
002A 30C1      00083      movlw  B'11000001'     ;select:RC osc,ch0...
002B 0088      00084      movwf  ADCON0          ;turn on a/d
002C 0008      00085      return
002C           00086 ;
002C           00087 ;This routine is a software delay of 10uS for the a/d setup.
002C           00088 ;At 4Mhz clock, the loop takes 3uS, so initialize TEMP with
002C           00089 ;a value of 3 to give 9uS, plus the move should result in
002C           00090 ;a total time of > 10uS.
002D           00091 SetupDelay
002D 3003      00092      movlw  .3
002E 0091      00093      movwf  TEMP
002F           00094 SD
002F 0B91      00095      decfsz TEMP, F
0030 282F      00096      goto   SD
0031 0008      00097      return
0031           00098 ;
0031           00099 ;
0031           00100      END
```

MEMORY USAGE MAP ('X' = Used, '-' = Unused)

```
0000 : X---X----- XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XX-----
```

All other memory blocks unused.

Program Memory Words Used: 36  
Program Memory Words Free: 988

Errors : 0  
Warnings : 0 reported, 0 suppressed  
Messages : 0 reported, 2 suppressed



Please check the Microchip BBS for the latest version of the source code. Microchip's Worldwide Web Address: [www.microchip.com](http://www.microchip.com); Bulletin Board Support: MCHIPBBS using CompuServe® (CompuServe membership not required).

## APPENDIX D: MULTAD.ASM

MPASM 01.40 Released

MULTAD.ASM 1-16-1997 15:21:41

PAGE 1

```

LOC  OBJECT CODE      LINE SOURCE TEXT
VALUE

00001 ;TITLE      "A/D using Multiple Channels"
00002 ;This program is a simple implementation of the PIC16C71's
00003 ;A/D feature. This program demonstrates
00004 ;how to select multiple channels on the PIC16C71.
00005 ;The A/D is configured as follows:
00006 ;      Vref = +5V internal.
00007 ;      A/D Osc. = internal RC osc.
00008 ;      A/D Interrupt = Off
00009 ;      A/D Channels = all in a "Round Robin" format.
00010 ;      A/D results are stored in ram locations as follows:
00011 ;      ch0 --> ADTABLE + 0
00012 ;      ch1 --> ADTABLE + 1
00013 ;      ch2 --> ADTABLE + 2
00014 ;      ch3 --> ADTABLE + 3
00015 ;
00016 ;The ch0 A/D result is displayed as a 8 bit value on 8 leds
00017 ;connected to port b.
00018 ;Hardware: PICDEMO board.
00019 ;      Stan D'Souza 7/6/93.
00020 ;
00021 ;      Program:          MULTAD.ASM
00022 ;      Revision Date:
00023 ;                      1-14-97      Compatibility with MPASMWIN 1.40
00024 ;
00025 ;
00026      LIST P=16C71
00027      ERRORLEVEL  -302
00028 ;
00029      include "p16c71.inc"
00001      LIST
00002 ;P16C71.INC Standard Header File, Version 1.00 Microchip Technology
00142      LIST
00030 ;
00000010      00031 TEMP      EQU      10h
00000001      00032 adif      equ      1
00000002      00033 adgo      equ      2
00034 ;
00000006      00035 ch2      equ      6
00000007      00036 ch3      equ      7
0000000C      00037 flag      equ      0C
00000020      00038 ADTABLE equ      20
00039 ;
0000      00040      ORG      0x00
00041 ;
00042 ;
0000 2810      00043      goto      start
00044 ;
0004      00045      org      0x04
0004 2825      00046      goto      service_int      ;interrupt vector
00047 ;
00048 ;
0010      00049      org      0x10
0010      00050 start
0010 3000      00051      movlw   B'00000000'      ;make port b

```

# AN546

```
0011 0086      00052      movwf    PORTB      ;as all outputs
                00053      ;      tris    PORTB      ;      /
0012 1683      00054      BSF     STATUS, RP0 ; Bank1
0013 0086      00055      MOVWF   TRISB      ; PortB as outputs
0014 1283      00056      BCF     STATUS, RP0 ; Bank0
                00057      ;
0015 2026      00058      call    InitializeAD
0016           00059      update
0016 0809      00060      movf    ADRES,W
0017 0080      00061      movwf   0           ;save in table
0018 3020      00062      movlw   ADTABLE    ;chk if ch0
0019 0204      00063      subwf   FSR,W      ;      /
001A 1D03      00064      btfss  STATUS,Z    ;yes then skip
001B 281E      00065      goto   NextAd      ;else do next channel
001C 0809      00066      movf    ADRES,W    ;get a/d value
001D 0086      00067      movwf   PORTB      ;output to port b
001E           00068      NextAd
001E 2030      00069      call    NextChannel ;select next channel
001F 203C      00070      call    SetupDelay  ;set up > = 10uS
0020 1088      00071      bcf    ADCON0,adif ;clear flag
0021 1508      00072      bsf    ADCON0,adgo ;start new a/d conversion
0022           00073      loop
0022 1888      00074      btfsc  ADCON0,adif ;a/d done?
0023 2816      00075      goto   update      ;yes then update
0024 2822      00076      goto   loop        ;wait till done
                00077      ;
0025           00078      service_int
0025 0008      00079      return           ;do not enable int
                00080      ;
                00081      ;
                00082      ;InitializeAD, initializes and sets up the A/D hardware.
0026           00083      InitializeAD
0026 1683      00084      bsf    STATUS,5    ;select pg1
0027 3000      00085      movlw  B'00000000' ;select ch0-ch3...
0028 0088      00086      movwf  ADCON1      ;as analog inputs
0029 1283      00087      bcf    STATUS,5    ;select pg0
002A 30C1      00088      movlw  B'11000001' ;select:fosc/2, ch0.
002B 0088      00089      movwf  ADCON0      ;turn on a/d
002C 3020      00090      movlw  ADTABLE    ;get top of table address
002D 0084      00091      movwf  FSR         ;load into indirect reg
002E 0189      00092      clrf   ADRES      ;clr result reg.
002F 0008      00093      return
                00094      ;
                00095      ;NextChannel, selects the next channel to be sampled in a
                00096      ;"round-robin" format.
0030           00097      NextChannel
0030 3008      00098      movlw  0x08       ;get channel offset
0031 0788      00099      addwf  ADCON0, F   ;add to conf. reg.
0032 1288      00100      bcf    ADCON0,5    ;clear any carry over
                00101      ;increment pointer to correct a/d result register
0033 0190      00102      clrf   TEMP
0034 1988      00103      btfsc  ADCON0,3    ;test lsb of chnl select
0035 1410      00104      bsf    TEMP,0      ;set if ch1 or ch3
0036 1A08      00105      btfsc  ADCON0,4    ;test msb of chnl select
0037 1490      00106      bsf    TEMP,1      ;set if ch0 or ch2
0038 3020      00107      movlw  ADTABLE    ;get top of table
0039 0710      00108      addwf  TEMP,W      ;add with temp
003A 0084      00109      movwf  FSR         ;allocate new address
003B 0008      00110      return
                00111      ;
                00112      ;This routine is a software delay of 10uS for the a/d setup.
                00113      ;At 4Mhz clock, the loop takes 3uS, so initialize TEMP with
                00114      ;a value of 3 to give 9uS, plus the move etc should result in
                00115      ;a total time of > 10uS.
003C           00116      SetupDelay
003C 3003      00117      movlw  .3
```

```
003D 0090          00118      movwf  TEMP
003E          00119 SD
003E 0B90          00120      decfsz TEMP, F
003F 283E          00121      goto   SD
0040 0008          00122      return
                   00123
                   00124 ;
                   00125
                   00126      END
```

MEMORY USAGE MAP ('X' = Used, '-' = Unused)

```
0000 : X---X----- XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXX
0040 : X-----
```

All other memory blocks unused.

```
Program Memory Words Used:    51
Program Memory Words Free:   973
```

```
Errors   :    0
Warnings :    0 reported,    0 suppressed
Messages :    0 reported,    2 suppressed
```

---

---

# WORLDWIDE SALES & SERVICE

---

---

## AMERICAS

### Corporate Office

Microchip Technology Inc.  
2355 West Chandler Blvd.  
Chandler, AZ 85224-6199  
Tel: 602-786-7200 Fax: 602-786-7277  
Technical Support: 602 786-7627  
Web: <http://www.microchip.com>

### Atlanta

Microchip Technology Inc.  
500 Sugar Mill Road, Suite 200B  
Atlanta, GA 30350  
Tel: 770-640-0034 Fax: 770-640-0307

### Boston

Microchip Technology Inc.  
5 Mount Royal Avenue  
Marlborough, MA 01752  
Tel: 508-480-9990 Fax: 508-480-8575

### Chicago

Microchip Technology Inc.  
333 Pierce Road, Suite 180  
Itasca, IL 60143  
Tel: 630-285-0071 Fax: 630-285-0075

### Dallas

Microchip Technology Inc.  
14651 Dallas Parkway, Suite 816  
Dallas, TX 75240-8809  
Tel: 972-991-7177 Fax: 972-991-8588

### Dayton

Microchip Technology Inc.  
Two Prestige Place, Suite 150  
Miamisburg, OH 45342  
Tel: 937-291-1654 Fax: 937-291-9175

### Los Angeles

Microchip Technology Inc.  
18201 Von Karman, Suite 1090  
Irvine, CA 92612  
Tel: 714-263-1888 Fax: 714-263-1338

### New York

Microchip Technology Inc.  
150 Motor Parkway, Suite 416  
Hauppauge, NY 11788  
Tel: 516-273-5305 Fax: 516-273-5335

### San Jose

Microchip Technology Inc.  
2107 North First Street, Suite 590  
San Jose, CA 95131  
Tel: 408-436-7950 Fax: 408-436-7955

### Toronto

Microchip Technology Inc.  
5925 Airport Road, Suite 200  
Mississauga, Ontario L4V 1W1, Canada  
Tel: 905-405-6279 Fax: 905-405-6253

## ASIA/PACIFIC

### Hong Kong

Microchip Asia Pacific  
RM 3801B, Tower Two  
Metroplaza  
223 Hing Fong Road  
Kwai Fong, N.T., Hong Kong  
Tel: 852-2-401-1200 Fax: 852-2-401-3431

### India

Microchip Technology India  
No. 6, Legacy, Convent Road  
Bangalore 560 025, India  
Tel: 91-80-229-0061 Fax: 91-80-229-0062

### Korea

Microchip Technology Korea  
168-1, Youngbo Bldg. 3 Floor  
Samsung-Dong, Kangnam-Ku  
Seoul, Korea  
Tel: 82-2-554-7200 Fax: 82-2-558-5934

### Shanghai

Microchip Technology  
RM 406 Shanghai Golden Bridge Bldg.  
2077 Yan'an Road West, Hongjiao District  
Shanghai, PRC 200335  
Tel: 86-21-6275-5700  
Fax: 86 21-6275-5060

### Singapore

Microchip Technology Taiwan  
Singapore Branch  
200 Middle Road  
#10-03 Prime Centre  
Singapore 188980  
Tel: 65-334-8870 Fax: 65-334-8850

### Taiwan, R.O.C

Microchip Technology Taiwan  
10F-1C 207  
Tung Hua North Road  
Taipei, Taiwan, ROC  
Tel: 886 2-717-7175 Fax: 886-2-545-0139

## EUROPE

### United Kingdom

Arizona Microchip Technology Ltd.  
Unit 6, The Courtyard  
Meadow Bank, Furlong Road  
Bourne End, Buckinghamshire SL8 5AJ  
Tel: 44-1628-851077 Fax: 44-1628-850259

### France

Arizona Microchip Technology SARL  
Zone Industrielle de la Bonde  
2 Rue du Buisson aux Fraises  
91300 Massy, France  
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

### Germany

Arizona Microchip Technology GmbH  
Gustav-Heinemann-Ring 125  
D-81739 München, Germany  
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

### Italy

Arizona Microchip Technology SRL  
Centro Direzionale Colleone  
Palazzo Taurus 1 V. Le Colleoni 1  
20041 Agrate Brianza  
Milan, Italy  
Tel: 39-39-6899939 Fax: 39-39-6899883

## JAPAN

Microchip Technology Intl. Inc.  
Benex S-1 6F  
3-18-20, Shin Yokohama  
Kohoku-Ku, Yokohama  
Kanagawa 222 Japan  
Tel: 81-4-5471- 6166 Fax: 81-4-5471-6122

5/8/97



# MICROCHIP

All rights reserved. © 1997, Microchip Technology Incorporated, USA. 6/97

---

Information contained in this publication regarding device applications and the like is intended for suggestion only and may be superseded by updates. No representation or warranty is given and no liability is assumed by Microchip Technology Incorporated with respect to the accuracy or use of such information, or infringement of patents or other intellectual property rights arising from such use or otherwise. Use of Microchip's products as critical components in life support systems is not authorized except with express written approval by Microchip. No licenses are conveyed, implicitly or otherwise, under any intellectual property rights. The Microchip logo and name are registered trademarks of Microchip Technology Inc. in the U.S.A. and other countries. All rights reserved. All other trademarks mentioned herein are the property of their respective companies.