INTRODUCTION

Comparators are used for many things, but many people want to build analog-to-digital converters using a comparator. Comparators are one basic building block of all ADC architectures. A PICmicro® microcontroller with internal comparators can become an ADC with the application of some software and a minimum of external hardware. In this application note you will be shown how to build a Slope ADC. This ADC has been used with the digital pins on PICmicro microcontrollers for many years. The addition of a comparator to the circuit improves the results and reduces the power consumption.

A characteristic to measure with a Slope ADC is resistance. By using a thermistor, temperature can be directly converted to a resistance. Measuring a thermistor is a good way to demonstrate the functions of a Slope ADC. Temperature measurement, or control, is one of the more common applications of microprocessor systems. One of the first signs of an illness in a person is a fever. Overheating fuels cause disasters. Food that has not been stored at the proper temperature spoils. Nearly every system or process requires an accurate understanding of the temperature. This application note describes another way to measure the temperature, and a way to build an analog-to-digital converter.

An example of some of the PICmicro microcontrollers that have comparators are:

- PIC12F629
- PIC12F675
- PIC16C620
- PIC16C621
- PIC16C22
- PIC16CE623

Thermistors

Thermistors are devices that change in resistance with changes in temperature. Thermistors are more properly called RTDs (Resistive Temperature Detectors). An RTD can be made from a variety of substances depending on the desired operation temperature range, the desired operation environment, the required accuracy and the cost. Platinum RTDs can achieve an accuracy of much less than 1°C. Other low cost RTDs achieve an accuracy of just a few degrees.

Slope ADC

Analog-to-digital converters can be created in a variety of ways. For this application note, we will build a slope converter. The slope converter works by timing the rate that a capacitor charges to a specified voltage. This is done twice, once with a reference source and once with a sensor. By performing the operation twice, variations in capacitor values or timing accuracy can be minimized. The reference source for this application note was simply a 10 kΩ resistor. The sensor was a 10 kΩ thermistor. See Figure 1.
To make a measurement, the following steps are taken:

1. Activate the comparator & voltage reference.
2. Clear the timer.
3. Activate the thermistor.
4. Wait for the comparator to trip.
5. Multiply the timer value by the value of the reference resistor.
6. Discharge the timing capacitor.
7. Clear the timer.
8. Activate the reference.
9. Wait for the comparator to trip.
10. Divide the result of step 5 by the new timer value.
11. Shutdown the comparator and voltage reference.
12. Return with the result of step 10 stored as the thermistor resistance.

Scaling the Data

One advantage of using RTD is that the response is nearly linear. Most RTDs are specified with an R25/R50 value. This is the ratio of the 25°C, with the 25°C value. The ratio is a linear relationship that can be coded to convert the ADC values to temperature. The Multiple and Divide code is located in Appendix C.

Using the Data

How an application will use the data from a sensor will depend on the function. For this application note, the data is simply displayed on a 3-digit, 7-segment LED display. The segments are driven by PORTB on an 18-pin PICMicro microcontroller. The digits are selected by a second 8-pin PICmicro microcontroller (See Figure 2). The digit selection was performed by the second PICmicro microcontroller to demonstrate a simple I/O expander.

The software for the PIC12F629 is located in Appendix B. The master sends the pulse train to select the correct digit while driving the segment lines. See Figure 4 for a scope capture of the pulse train.

FIGURE 2: DISPLAY SCHEMATIC

![Diagram of display schematic]
Conclusion

Using a comparator to read analog’s voltage is a straight-forward process combining a small amount of discrete circuitry with a small piece of firmware. Other ADCs can also be built in this manner. See AN700 for directions on how to build a delta sigma ADC using comparators and a few inexpensive discrete components.
Software License Agreement

The software supplied herewith by Microchip Technology Incorporated (the "Company") is intended and supplied to you, the Company's customer, for use solely and exclusively on Microchip products manufactured by the company.

The software is owned by the Company and/or its supplier, and is protected under applicable copyright laws. All rights are reserved. Any use in violation of the foregoing restrictions may subject the user to criminal sanctions under applicable laws, as well as to civil liability for the breach of the terms and conditions of this license.

THIS SOFTWARE IS PROVIDED IN AN "AS IS" CONDITION. NO WARRANTIES, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE APPLY TO THIS SOFTWARE. THE COMPANY SHALL NOT, IN ANY CIRCUMSTANCES, BE LIABLE FOR SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, FOR ANY REASON WHATSOEVER.

APPENDIX A: SLOPE CODE

;***************************************************************;
; Perform Slope Conversion
; Uses a0,a1,b0,b1
; result is the unknown resistance in Ohms stored at c0,c1,c2,c3
;***************************************************************
reference_value equ .10000 ; reference resistance
Measure_Resistance

banksel TRISA
movlw 0x27 ;          76543210
movwf TRISA ; PortA is OOIOOIII

banksel PORTA
movf CMCON,W
bsf PORTA,3 ; start current through the reference

call T0Timing

movwf a0
movf T0_HIGH,W
movwf a1 ; store timer value

clrf PORTA ; clearing the Capacitor

banksel TRISA
movlw 0x2D ;          76543210
movwf TRISA ; PortA is OOIOIIOI

banksel b0
movlw high reference_value
movwf b1

movlw low reference_value
movwf b0

call multiply

banksel TRISA
movlw 0x2E ;          76543210
movwf TRISA ; PortA is OOIOIII0

banksel CMCON
movf CMCON,W

bsf PORTA,0 ; start current through the sensor

call T0Timing

movwf a0
movf T0_HIGH,W
movwf a1

clrf PORTA
banksel TRISA
movlw 0x2D ; 76543210
movwf TRISA ; PortA is 00101101
banksel PORTA

call divide

return
APPENDIX B: DIGIT MULTIPLEXING CODE

main
  call  0x3FF ; retrieve factory calibration value
  bsf   STATUS,RP0 ; set file register bank to 1
  movwf OSCCAL ; update register with factory cal value
  bcf   STATUS,RP0 ; set file register bank to 0

; time the input
  movlw reset_time

  time_loop
  addlw 0xFF
  btfss STATUS,Z ; countdown the reset time
  goto time_loop

  btfsc GPIO,3
  goto reset_return

  incf state,w
  andlw 0x03
  addwf PCL,F
  goto state0
  goto state1
  goto state2
  goto state3

state0
state1
  movlw .1
  movwf state
  movlw .4
  goto update

state2
  movlw .2
  movwf state
  movlw .2
  goto update

state3
  movlw .3
  movwf state
  movlw .1

update
  movwf GPIO
  goto normal_return
APPENDIX C: MULTIPLE AND DIVIDE Routines

;*******************************************************
; Unsigned Divide Routine 32/16=32
; c3:c2:c1:c0 / a1:a0 = c3:c2c1:c0
;*******************************************************

divide
; Test for zero division
movf a0,W
iorwf a1,W
btfsc STATUS,Z
retlw 0x00
; prepare used variables
clrft tmp
clrft tmp1
clrft tmp2
movlw D'32' ; initialize bit counter
movwf bcnt

DIVIDE_LOOP_32by16
rlf c0,F
rlf c1,F
rlf c2,F
rlf c3,F
; shift in highest bit from dividend through carry in temp
rlf tmp1,F
rlf tmp,F
rlf tmp2,F
movf a0,W ; get LSB of divisor
btfsc tmp2,7
goto Div32by16_add

; subtract 16 bit divisor from 16 bit temp
subwf tmp1,F ; subtract
movf a1,W ; get top byte
btfss STATUS,C ; if overflow ( from prev. subtraction )
incfsz a1,W ; increase source
subwf tmp,F ; and subtract from dest.

movlw 1
btfss STATUS,C
subwf tmp2,F
goto DIVIDE_SKIP_32by16 ; carry was set, subtraction ok, continue with next bit
Div32by16_add

; result of subtraction was negative restore temp
addwf tmp1,F ; add it to the lsb of temp

movf a1,W ; MSB byte
btfsc STATUS,C ; check carry for overflow from previous addition
incfsz a1,W
addwf tmp,F ; handle overflow

movlw 1
btfsc STATUS,C
addwf tmp2,F

DIVIDE_SKIP_32by16

decfsz bcnt,F ; decrement loop counter
goto DIVIDE_LOOP_32by16 ; another run

; finally shift in the last carry
rlf c0,F
rlf c1,F
rlf c2,F
rlf c3,F

retlw 0x01 ; done

;*******************************************************************************

; Unsigned Multiply 16x16
; a1:a0 * b1:b0 = c3:c2:c1:c0
;*******************************************************************************
multiply

clr c3
clr c2
clr c1
clr c0
bsf c1, 7

multiply_lp1

rrf a1, f
rrf a0, f
btfss STATUS,C
goto multiply_lp2

movf b0, w
addwf c2, f
movf b1, w
btfsc STATUS,C
incfsz b1, w
addwf c3, f

multiply_lp2

rrf c3, f
rrf c2, f
rrf c1, f
rrf c0, f
btfss STATUS,C
goto multiply_lp1

return
Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Trademark

The Microchip name and logo, KEELOQ, MPLAB, PIC, PICmicro, PICSTART, PRO MATE and PowerSmart are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

FilterLab, microID, MXDEV, MXLAB, PICMASTER, SEEVAL and The Embedded Control Solutions Company are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Accuron, Application Maestro, dsPIC, dsPICDEM, dsPICDEM.net, ECONOMONITOR, FanSense, FlexROM, fuzzyLAB, In-Circuit Serial Programming, ICSP, ICEPIC, microPort, Migratable Memory, MPASM, MPLIB, MPLINK, MPSIM, PICC, PICkit, PICDEM, PICDEM.net, PowerCal, PowerInfo, PowerMate, PowerTool, rFLAB, rF PIC, Select Mode, SmartSensor, SmartShunt, SmartTel and Total Endurance are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

Serialized Quick Turn Programming (SQTP) is a service mark of Microchip Technology Incorporated in the U.S.A.

All other trademarks mentioned herein are property of their respective companies.

© 2003, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

Printed on recycled paper.