TEMPERATURE SENSOR

Revision 04.05.04

_____

Class

_____

Instructor / Professor

© 2003 Axiom Manufacturing
2813 Industrial Lane - Garland, TX 75041

Phone: 972-926-9303
http://www.axman.com
email: sales@axman.com

CONTENTS

1 REQUIREMENTS

1.1 Hardware

To complete this lab, **the following hardware is required**:

- Axiom CML-12C32 Development Kit
- PC running Windows OS
- Project Starter Kit which includes:
  - (1) LM34
  - (3) Jumper wires

1.2 Software

The CML-12C32 board used in this experiment comes with all the software needed to complete this project.

There are many additional utilities included on the boards support CD that can make developing your own projects easier. The CD contains example source code, documentation and experiments for all Axiom development boards. You can also download the latest versions of the software and documentation free from our web site at: www.axman.com.

Also included is an integrated development environment, called AxIDE, for communicating with the board (via the serial port) and for reading and writing its flash memory. To complete this Lab, you should have this program installed on a PC running Microsoft Windows (95/98/2000/XP).

NOTE: This lab does not teach you how to use the AxIDE terminal interface or the MON12 Monitor program to modify memory and upload programs. It assumes you're already familiar with these procedures. Refer to your board manual for details on installing and using this software, including a tutorial for using AxIDE.

CAUTION

Devices used in this lab are static sensitive and easily damaged by mishandling. Use caution when installing wires and devices onto the board to prevent bending the leads.

Experiments should be laid out in an orderly fashion. Start your lab time with the bench clean and free of metal objects and leave the lab area in a clean condition by picking up loose parts, wires and small objects.

2 GETTING STARTED

This lab project will show you how to add a temperature sensor as an input to the microcontroller on your Axiom development board. A temperature sensor with a Fahrenheit scale is used in this example

A temperature sensor is a solid state device that changes its output voltage at a know rate whenever temperature changes. The device used in this lab has a linear output that follows the Fahrenheit scale. An analog port pin on the microcontroller will read this voltage and convert it to digital format. The temperature sensor is powered by +5 on the development board.

The output voltage changes 10mv per degree Fahrenheit. So at room temperature of 72 degrees this sensor should output a voltage of 0.72 volts. Negative Fahrenheit temperature requires a negative supply voltage and is not supported in this lab.

The accuracy of the sensor is normally very high and non-linearity very low. With output impedance low it is able to drive the A/D of the microcontroller. Being a low current device, self-heating is also very low.

Temperature sensors are used in thermostat devices for controlling the temperature of a building. They're also used in appliances, machinery, cars and many other products to monitor temperature inside a product such as monitoring the temperature of the processor inside your computer. They come in Fahrenheit scale or Centigrade scale. These devices may be ordered in several temperature ranges, such as +32 to +212 degrees or –40 to +230 degrees.

3 LAB PROCEDURE

This lab is arranged as a series of simple steps. Each step should be completed before moving on to the next one, which builds on prior ones. Repeat each step as many times as necessary to become familiar with it. You'll find it easier to complete more complex experiments after mastering the simple ones.

As an aid to keeping track of your location it's a good idea to mark each step as it's completed, since the experiment will fail if anything is skipped.

3.1 Description

This example uses the analog port on the hc12 C32 microcontroller. This port can be configured for either analog or digital input. In analog mode the A/D has a 10-bit resolution. With a +5 volt reference this is a resolution of .004888 volts per count.

Channel 6 will be used to convert the analog voltage to digital format. You can see in the C32 reference manual that results are read from location ATDDR6H ($009C) and ATDDR6L ($009D). Reading these results directly reflects the level on the analog pin. The analog channel is configured using analog control registers ATDCTL2 ($0082), ATDCTL3 ($0083), ATDCTL4 ($0084) and ATDCTL5 ($0085). Changing the temperature of the sensor will change the voltage applied to the analog channel which is then converted to digital format.

3.2 Detailed Steps

This section describes how to build the temperature sensor project and test it with the monitor running on the CML-12C32. In the next section, you'll see how to write a simple program to read the temperature sensor.

> **NOTE**: To complete these steps you must be familiar with modifying register contents on your board. For example, to write $F8 to the ATDCTL2 register with the MON12 monitor, type `MM 82` at the monitor prompt and press <enter>. Then type `F8` <enter> then `.` to end modify mode. Type Help for more commands.
>
> You can use a different monitor or debugger if you prefer, such as the GNU GDB.

1. Verify power is NOT applied to the development board.

2. Install the temperature sensor on the breadboard area as shown in **Figure 1**.

3. Install the jumper wires on the board as follows:

   MCU PORT -----Breadboard
   GND----------GND
   +5V----------+5V
   PAD6----------TEMP

4. Configure the CML-12C32 board to start the MON12 debugger by setting the jumpers to their default positions (NO_AUTO and MEM_EN are ON).

5. Write $C0 to the ATDCTL2 register. This sets the ADPU bit high which turns the analog to digital converter on with fast clear.

6. Write $00 to ATDCTL3. This sets A/D converter for 8 channels.

7. Write $75 to ATDCTL4. This sets A/D converter for 10-bit resolution.

8. Write $90 to ATDCTL5. This sets A/D converter for multi channel, single conversion. It also starts a conversion.

9. Read address ATDDR6H. This is the high byte of the A/D conversion. Only bits 0 and 1 are valid.

10. Read address ATDDR6L. This is the low byte of the A/D conversion. These 8 bits combined with the two bits in step 9 create a 10-bit result. Save this value for later as the Room Temp.

11. Press the temperature sensor between your fingers for one minute and repeat steps 8, 9 and 10. Save this value for later as the Warm Temp.

12. Convert both Room and Warm 10-bit values from Hex to a Decimal then multiple the result by .004888. This will give you the voltage outputs of the temperature sensor.

13. By multiplying these voltages by 100 they can be converted to a Fahrenheit temperature reading.

14. Compare the Room Temperature reading with the Warm Temperature reading. You should notice the Warm reading is higher than the Room reading.

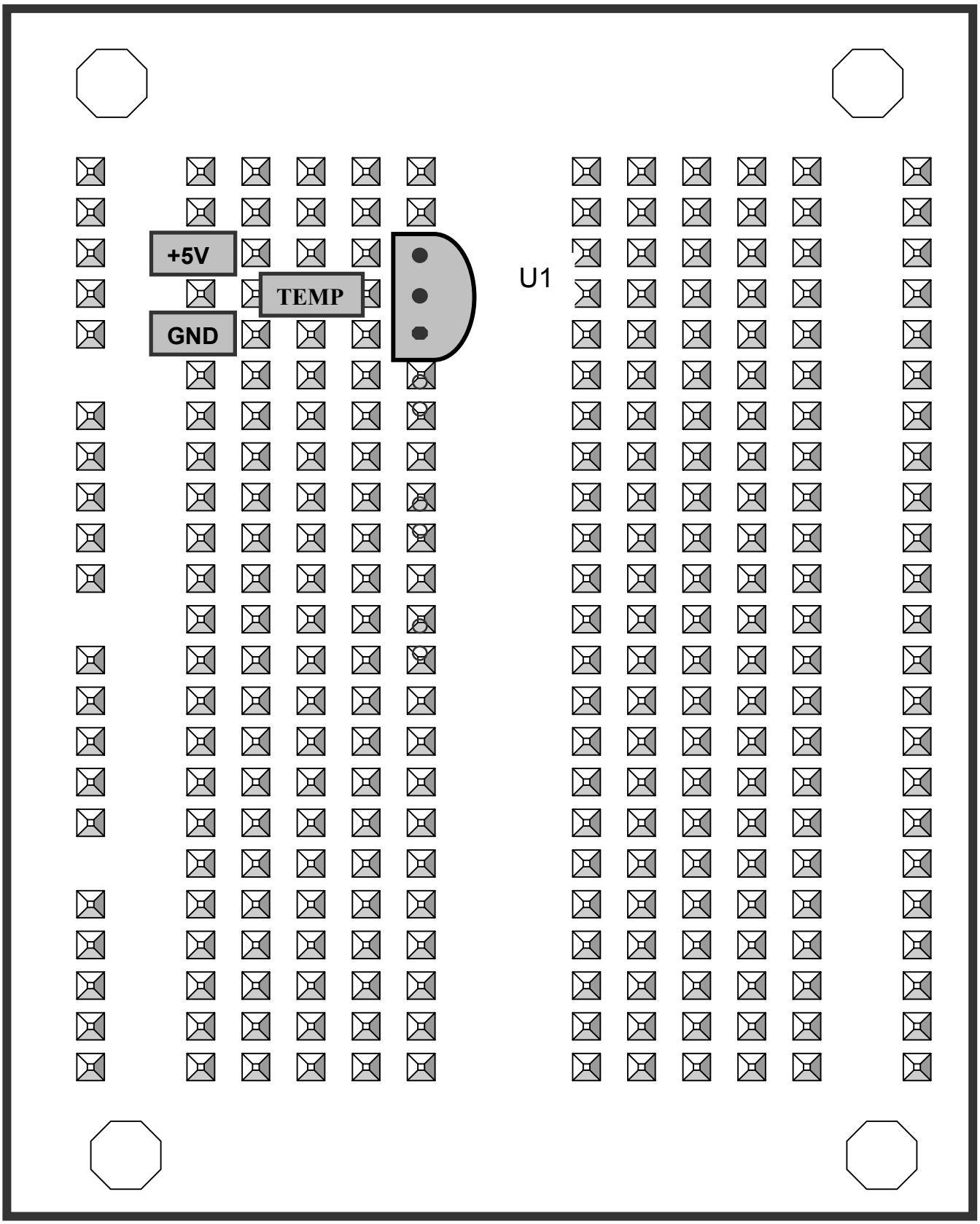
This simple experiment demonstrates how easy a temperature sensor can be added to and processed by a microcontroller. By using one of the analog port pins the temperature is read and converted into a register. Any number of sensors can be added, the hc12 C32 has 8 A/D channels for this purpose.

Some circuit designs place an op-amp between the temperature sensor and microcontroller. By setting the op-amp gain to a value of two, the resolution of the microcontroller can be increased by two. This increases the resolution but reduces the temperature range.

The temperature sensor is a precision integrated circuit but conversion errors add up. Errors can involve sensor precision, the 10-bit resolution or reference for the A/D on the microcontroller. The part comes in several packages. This lab used a thru-hole device but most production units use a surface mount device. They can be mounted directly onto a board, a motor or other device for monitoring temperature at its source.

You can use a temperature sensor to control the temperature of a room. For example, using values produced by the A/D channel, a software program could toggle an I/O pin when the temperature moves out of a desired range. This I/O pin change could then turn a heater On or Off. Another application might be to control the speed of a fan based on the temperature reading.

Figure 1



4 Temperature Program

The previous section described how to read the temperature sensor manually using a debugger. While this method is useful for testing and experimenting, once the hardware is working you'll want to write a software program that automatically reads the sensor.

This section describes how to write such a program in assembly language. The source code is listed at the end of this section. Both code and assembled executable is on the support CD or can be downloaded from the Axiom web site: **www.axman.com**.

If viewing this on your PC, you can copy and paste the source code below into a text editor (such as notepad) then save and assemble it using AxIDE. Refer to the owner's manual of your board for instructions on creating software and running programs for your development board.

4.1 Program Description

The program first enables the analog converter for single conversion and multi channels. A delay routine is called before each conversion so the values can be read on the terminal.

Next, a conversion is started and the analog result register is read. This 10-bit value is multiplied by 4888, which is the .004888 mentioned earlier times 1,000,000. The result is divided by 10,000. The final value is the temperature in Fahrenheit but is still in hex form.

The hex value is process though a hex to decimal conversion routine that displays three decimal values on the terminal. Finally, the program jumps back to the beginning and repeats until you press reset or remove power.

4.2 Running the Program

1. Upload the assembled program TEMPCA.S19 into the RAM on your board. This program starts at address $0800, which is internal memory on the CML-12C32. The source code for this program is shown below.

2. Execute the program by typing `call 0800` in the terminal and pressing <enter>.

3. Verify a temperature reading is displayed on the terminal.

4. Change the temperature of the sensor and verify the temperature reading is also changing.

4.3 Temperature Program Source Code

```
;  Example: Fahrenheit Temperature Sensor
;  Results: Displays the temperature in Fahrenheit
;
;     Math: A = Value read in hex
;           B = value of each step (.004888 * 1000000 = $1315)
;            Temperature = (A * B) / 10000
;         or Temperature = $1315A / $2710
;
; Register Equates
ATDDR6H:     equ     $009C        ; Analog Result high byte
ATDDR6L:     equ     $009D        ; Analog Result low byte
ATDCTL2:     equ     $0082        ; Analog Control
ATDCTL3:     equ     $0083        ; Analog Control
ATDCTL4:     equ     $0084        ; Analog Control
ATDCTL5:     equ     $0085        ; Analog Control
SC1SR1:      equ     $00CC        ; Sci status register
SC1DRL:      equ     $00CF        ; Sci data register
;
; Setup the A/D converter
      org     $0800          ; program starts here
      movb    #$C0,ATDCTL2  ; Turn A/D On
      clr     ATDCTL3       ; Eight Conversions
      movb    #$75,ATDCTL4  ; Ten bit Resolution
      movb    #$90,ATDCTL5  ; Multi Channel, Single Conversion
; Display Results
Loop2
      jsr      CrLf           ; new line
      movb     #$90,ATDCTL5  ; start a conversion
      jsr      Delay          ; delay for conversion
; read value & calculate temperature
      ldx      ATDDR6H         ; read 10 bit conversion
      xgdx                    ; move to Acc D
      ldy      #4888          ; Conversion factor
      emul
      ldx      #10000         ; load divisor
      ediv                    ; y contains result
      bsr      HexDec         ; convert hex to decimal
      pshb                    ; save high byte
      jsr      HEX2           ; send hex
      pula                    ; get low byte
      jsr      HEX2           ; send hex
      bra      Loop2          ; display next digit
;
; calculate Decimal Temperature
; Input: Y should contain hex
; Output: D should contain Decimal
HexDec:
      clra
      clrb
HexDecL:
      cmpy     #$0000
      beq      HexDecE
      dey                     ; decrement y
      adda     #$01           ; add one to A
      daa                     ; Decimal adjust
      bcc      HexDecK        ; skip over if carry not set
      addb     #$01           ; add one to B
HexDecK:
      bra      HexDecL        ; repeat until y = 0
HexDecE:
; exchange A & B
```

```
      pshb
      tab                   ; transfer a to b
      pula                  ; get A
      rts

; Output A to SCI0
Output:
      brclr   SC1SR1,#$80,Output  ; wait for Xtrans to empty
      staa    SC1DRL        ; send character
      rts

; output right four bits of acc A as a hex ascii value, send to com 1
HEXOUT:
      anda    #$0f          ; mask off bits
      cmpa    #$09          ; compare to number
      bhi     HEX01         ; branch if a thru f
      adda    #$30          ; add standard offset
      jsr     Output
      rts
HEX01:
      adda    #$37          ; change a-f to ascii
      jsr     Output
      rts
;
; send two hex values in acc A
; send to com 1
HEX2:
      psha                  ; save a
      lsra                  ; shift right four places
      lsra
      lsra
      lsra
      bsr     HEXOUT        ; send high hex char
      pula                            ; restore a
      anda    #$0f          ; only send lower 4 bits
      bsr     HEXOUT        ; send low hex char
      rts
;
; send car & line feed
CrLf:
      psha                  ; save A
      ldaa    #$0a          ; send car
      bsr     Output
      bsr     Delay
      ldaa    #$0d          ; line feed
      bsr     Output
      bsr     Delay
      pula                  ; restore A
      rts
;
; Delay Loop
Delay:
      ldab    #$10
DelayB:
      ldy     #$FFFF
DelayA:
      dey
      bne     DelayA
      decb
      bne     DelayB
      rts
```

5 QUIZ

Answer the following questions based on the example presented in this lab.

1. Select the start address of the temperature program.
A. $0800 **B**. $0100 **C**.$8000 **D**. $FFFE

2. What type temperature sensor is used in this lab?
A. Centigrade **B**. Fahrenheit **C**. Thermal Couple **D**. Thermistor

3. How much does the sensor change per degree Fahrenheit?
A. 10 MV **B**. 4888 MV **C**. .01 MV **D**. 5 V

4. What type port is used in this lab?
A. Digital **B**. Output **C**. Bi-Directional **D**. Analog

5. Which channel was chosen?
A. 6 **B**. 5 **C**. 4 **D**. 3

6. What are the terms that best describe the temperature sensor?
A. Linear **B**. Low Impedance **C**. Low Current **D**. A,B,C

7. The A/D is setup for?
A. Single Conversion **B**. Stop Mode **C**. BDM Mode **D**. A, B, C

8. Which temperature below is out of range of this lab?
A. 10 °F **B**. 120 °F **C**. –10 °F **D**. 75 °F

9. Select an answer that effects the reading?
A. Hex/Decimal Conversion **B**. Stack Data **C**. Interrupts **D**. Reference

10. Choose the number of bytes in the temperature reading.
A. 1 **B**. 2 **C**. 4 **D**. 8

BONUS QUESTION: What is the formula for conversion A/D value to degrees? Note: A = is the value read from the A/D converted to decimal. F = Fahrenheit
A. F = (A*72)/5 **B**.F = A-33 **C**. F = (A*256)+4888 **D**.F = (A*4888)/10000

6 SCHEMATIC