TEMPERATURE SENSOR

Revision 04.02.11

_____

Class

_____

Instructor / Professor

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 CME-11E9 EVBU Development Kit
- PC running Windows OS
- Temperature Sensor Lab Kit which includes:
  - (1) LM34
  - (3) Jumper wires

1.2 Software

The CME11E9 EVBU 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 Buffalo 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 device to the microcontroller on your Axiom development board. In this example, a temperature sensor with a Fahrenheit scale is used.

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 in 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 will find it easier to complete more complex experiments after mastering the simple ones.

As an aid to keeping track of 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 PORTE, the analog port on the HC11 microcontroller. This port must be configured to continuously scan for analog input. With the analog channels 8-bit resolution and a +5 volt reference, the A/D has a resolution of .0194 volts per count.

Analog channel 6 will be used to convert the analog voltage to digital format. You can see in the HC11 reference manual that A/D conversion results are read from location ADR3 (address $1033). Reading this result directly reflects the level on the analog pin. The analog channel is configured using the analog control registers OPTION ($1039) and ADCTL ($1030).

Changing the temperature of the sensor will change the voltage applied to the analog channel. This change is converted to a Fahrenheit temperature. The temperature resolution is two degrees Fahrenheit.

3.2 Detailed Steps

This section describes how to build the temperature sensor project and test it with the monitor running on the CME-11E9 EVBU. 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 read PORTE with the Buffalo monitor, type: `MM 1033` into the terminal at the buffalo prompt and press `<enter>`. The value will display after the address. 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
   VCC----------VCC
   PE6- ----------TEMP

4. Install the MODA jumper on the EVBU board and remove jumpers MODB and MEM_EN, to configure the board for single chip operation, then apply power to the board.

> Register address used (see M68HC11E Technical Data manual for more information).
> `OPTION = $1039  ADCTL = $1030  ADR3 = $1033`

5. Using the debugger or monitor, write $80 to the OPTION register. This sets the ADPU bit high which turns the analog to digital converter on. Note: the buffalo monitor turns this on for you because it must be written in the first 64 cycles after reset.

6. Write $34 to ADCTRL register. This sets the A/D converter to continuous conversion on multiple channels. It also selects port E channels 4,5,6,7. The conversion results will be placed in ADR1 – ADR4.

7. Read address ADR3. This is the result of the channel 6 A/D conversion of the voltage output from the temperature sensor. Record this 8-bit hex value as the ROOM Temperature.

8. Press the temperature sensor between your fingers for one minute then repeat step 7, but record this value as the HOT temperature.

9. Convert both ROOM and HOT readings to decimal values then multiply them by .0194. This will give you the voltage output of the temperature sensor.

10. By multiplying the voltage output by 100, one can convert the voltage to a Fahrenheit temperature reading.

11. Compare the ROOM temperature reading with the HOT reading. You should notice the HOT reading is higher than the ROOM temperature reading.

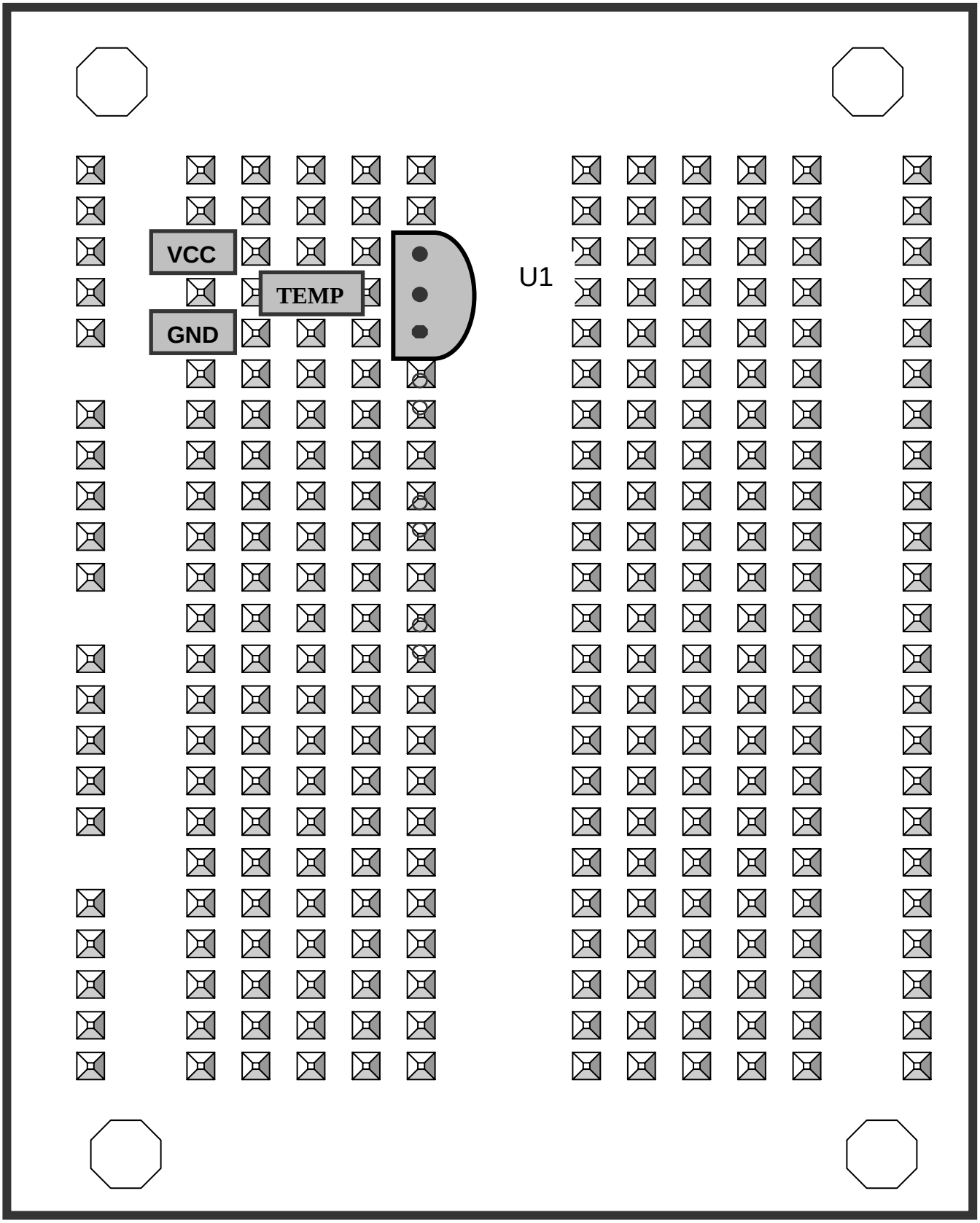
This simple experiment demonstrates how easy a temperature sensor can be added to and processed by a microcontroller. By using one of the analog port pins on the HC11 the temperature is read and converted into a register. Any number of sensors can be added. The HC11 has 8 A/D channels for this purpose.

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 8-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 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 software to automatically read 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 source code and assembled executable for this example can also 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 continuous conversion and multi channels. A delay routine is called before each conversion so the values can be read on the terminal.

The conversion result is read from an 8-bit result register. This value is multiplied by 194, which is the .0194 mentioned earlier times 10,000. The result is divided by 100.

The final value is the temperature in Fahrenheit in hex form. The value goes though a hex to decimal conversion routine that sends three decimal digits to 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 TEMP1A.S19 into the RAM on your board. This program starts at address $0100, which is internal memory on the CME-11E9 EVBU. The source code for this program is shown below.

2. Execute the program by typing `call 100` 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 - LM34
*  Results: Displays the temperature in Fahrenheit
*  Note: Only 2 degrees resolution
*  Math: A = value read in hex
*        B = value of each step (.0194 * 10000 = 194)
*        Temperature = (A * 194)/100)

* Analog Register Equates
ADCTL:    equ     $1030        * Analog Control Register
ADR3:     equ     $1033        * CH6 Analog Result
OPTION:   equ     $1039        * Analog Enable Register
OUTPUT:   equ     $FFAF        * Buffalo output routine
OUTRHLF:  equ     $FFB5        * Buffalo routine right nibble to AscII

* Buffalo Routines
OUTCRLF:  equ     $FFC4        * car/line feed
OUT1BYT:  equ     $FFBB        * Display Byte as hex
TEMP:     equ     $01fe        * temp storage

* Setup the A/D converter
          org     $0100        * program starts here
          ldaa    #$80         * Turn A/D On
          staa    OPTION
          ldaa    #$36         * Select PE6, multi conversions
          staa    ADCTL        * Setup A/D

* Display Results
Loop2
          jsr     OUTCRLF      * new line
          jsr     Delay

* read value & calculate temperature
          ldab    #194         * Multi factor
          ldaa    ADR3         * read 8-bit conversion
          mul                  * Multiple Unsigned D=Result
          ldx     #100         * divide factor
          idiv
          pshx                 * save value
          puly                 * move to y
          bsr     HexDec       * convert hex to decimal
          ldx     #TEMP        * point to data
          std     0,x          * save result
          std     0,x
          jsr     OUT1BYT      * send high byte
          jsr     OUT1BYT      * send low byte
          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

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 example temperature program.
A. $1000 **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. Continuous 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 the answer that affects 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*194)/100 **C**. F = (A*256)+4888 **D**.F = A-33

6 SCHEMATIC