Laboratory Exercise 1: Build Your Own DC Voltmeter ENGR 221, Winter 2010

This exercise covers basic concepts such as potentiometers, voltage divider circuits, analog-to-digital converters, serial communication, binary-to-decimal conversions and voltage and current measurement.

In this experiment a potentiometer will be used to generate an analog voltage signal. The ADC0831, an 8-bit successive approximation analog-to-digital converter, will convert the analog signal into a digital signal and transmit that information serially to the Basic Stamp. The digital value will be displayed on the laptop computer via the debug terminal and compared against the actual signal as measured by a digital voltmeter. Since the voltage level of Vdd is different for each Boe-Bot, the program for this lab has been written in such a way as to allow the user to enter the necessary information required to make the Boe-Bot display the same value as the voltmeter.

Once the Boe-Bot has been configured to display the correct voltage from the potentiometer, then that voltage signal will be applied across a resistor of known value. The potentiometer will be adjusted from 0 V to the full scale value of Vdd in approximately 0.5 V increments and the current flowing through the resistor will be measured by a digital ammeter. These current values will be recorded and then graphed using Microsoft Excel.

Construct the circuit shown below in Figure 1 using the following parts:

Parts list:	(1) ADC0831 analog-to-digital converter
	(1) 25 kΩ potentiometer
	(1) digital multimeter

Schematic:

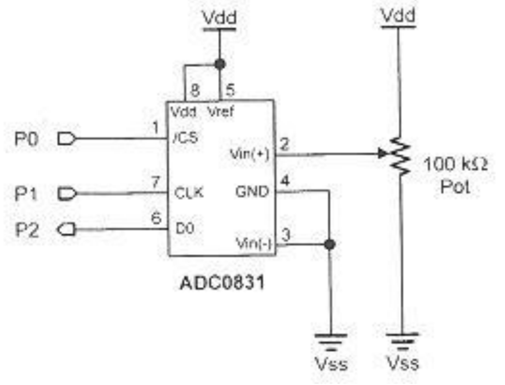

Figure 1 – Schematic for Laboratory Exercise 1

Since the ADC0831 is an 8-bit analog to digital converter, it takes the input voltage signal present on pin 2 and converts it to a digital value between 0 and 255, with a value of 0 for an input voltage of 0 V and 255 for an input voltage equal to Vdd. Thus, the equation for the voltage conversion is:

$$Input\ Voltage = \frac{Vdd}{255} * Digital\ Value$$

So in order for the Boe-Bot to display the value of the input voltage, the program needs to perform this calculation, but as you may recall from ENGR 120-122, the Boe-Bot does not include a decimal point. Instead, it can only work with integers (as does any fixed-point microprocessor or microcontroller). So as we have done before, we will multiply any value with a decimal point by the multiple of 10 needed to eliminate the decimal point to while performing any calculations with the Boe-Bot. The decimal point will be reinserted when the values are displayed on the debug terminal.

For example, suppose we measure Vdd on our Boe-Bot and find it to be 4.903 V. Then for a digital value of 128 from the ADC0831 (representing approximately half of Vdd), the equation above yields the following:

$$Input\ Voltage = \frac{4.903}{255} * 128 = 0.019227 * 128 = 2.461\ V$$

Inside the Boe-Bot program we will compute the following equation:

$$Input\ Voltage = \frac{4903}{255} * 127 = 19.227 * 128 = 2461$$

and then when this value is displayed, the 2 will be displayed, followed by a decimal point, and then 461 V.

Now, this may still look a bit strange to you since we are saying that the Boe-Bot needs to multiply the digital value from the ADC0831 by 19.227and that number includes a decimal point, yet we have established that there is no decimal point inside the Boe-Bot. Not to worry, there is an instruction in the Boe-Bot library that allows us to multiply a value by a fraction if we will construct the multiplier properly.

The command is “multiply middle” and the syntax is as follows:

$$Input\ Voltage = mult */decimal\ value$$

where “mult” is the multiplier value to be constructed. The value “mult” is constructed in two parts: first the part of the multiplier that is to the left of the decimal point; and second the part of the multiplier that is to the right of the decimal point, expressed as a numerator over 256.

So, in order to get our multiplier of 19.227 in the example above, we would construct a value with 19 as the first part and 58 as the second part since

$$0.227 = \frac{58}{256} \quad \text{or} \quad 58 = 0.227 * 256$$

Lab Exercise #1

Setup:

1. Measure Vdd for your Boe-Bot. Determine the multiplier value you need by dividing Vdd*1000 by 255. Record the first part of your multiplier value as the part of this value to the left of the decimal point, and compute the second part of your multiplier value by multiplying the part to the right of the decimal point by 256.

2. Run the BS2 program you downloaded from Blackboard. When asked, enter the two parts of your multiplier value.

3. Verify that the Boe-Bot displays the same value as your digital voltmeter for when the input value is Vdd volts.

Data gathering:

4. Disconnect the voltmeter and connect the output of the potentiometer to the 1.5 kΩ resistor. Configure the digital multimeter as an ammeter to measure the current flowing through this resistor.

5. Record the current for 10 values of voltage, ranging from 0 V to Vdd V in approximately 0.5 V increments.

6. Repeat steps 4 and 5 for the 7.5 kΩ and 30 kΩ resistors.

Analysis:

7. For each resistor value, plot the measured current as a function of supplied voltage (hence, three separate graphs).

8. For each graph, determine the slope of the line. Explain how this slope value relates to the resistor in the circuit.

9. Document your results in a lab report formatted as per the instructions on the course syllabus. The lab report is due one week from the date of this lab exercise.

Code listing for lab exercise 1:

```
' {$STAMP BS2}
' {$PBASIC 2.5}

adcBits     VAR Byte
v           VAR Word
v2          VAR Word
v3          VAR Word

mult        VAR Word
whole       VAR mult.HIGHBYTE
frac        VAR mult.LOWBYTE

CS          PIN 0
CLK         PIN 1
DataOutput  PIN 2

DEBUG CLS

DEBUG "What is the whole part of your multiplier?"
DEBUGIN DEC2 whole

DEBUG HOME
DEBUG CLS
DEBUG "What is the fractional part of your multiplier?", CR
DEBUG "Enter 3 digits or 2 digits and the Enter key"
DEBUGIN DEC3 frac

DEBUG CLS

DO
  GOSUB ADC_Data
  GOSUB Calc_Volts
  GOSUB Display
LOOP
```

```
ADC_Data:
  HIGH    CS
  LOW     CS
  LOW     CLK
  PULSOUT CLK, 210
  SHIFTIN DataOutput, CLK, MSBPOST, [adcBits\8]
RETURN

Calc_Volts:
  v = adcBits */ mult
  v2 = v / 1000
  v3 = v - (v2 * 1000)
RETURN

Display:
  DEBUG HOME
  DEBUG "8-bit binary value:  ", BIN8 adcBits
  DEBUG CR, "Decimal value:  ", DEC3 adcBits
  DEBUG CR, "Integer value:  ", DEC5 v
  DEBUG CR, "DVM Reading:  ", DEC1 v2, ".", DEC3 v3, " volts"
RETURN
```