Report For Lab 7

Connor McCullough

EEN 312

3 April 2013

**Objective:**

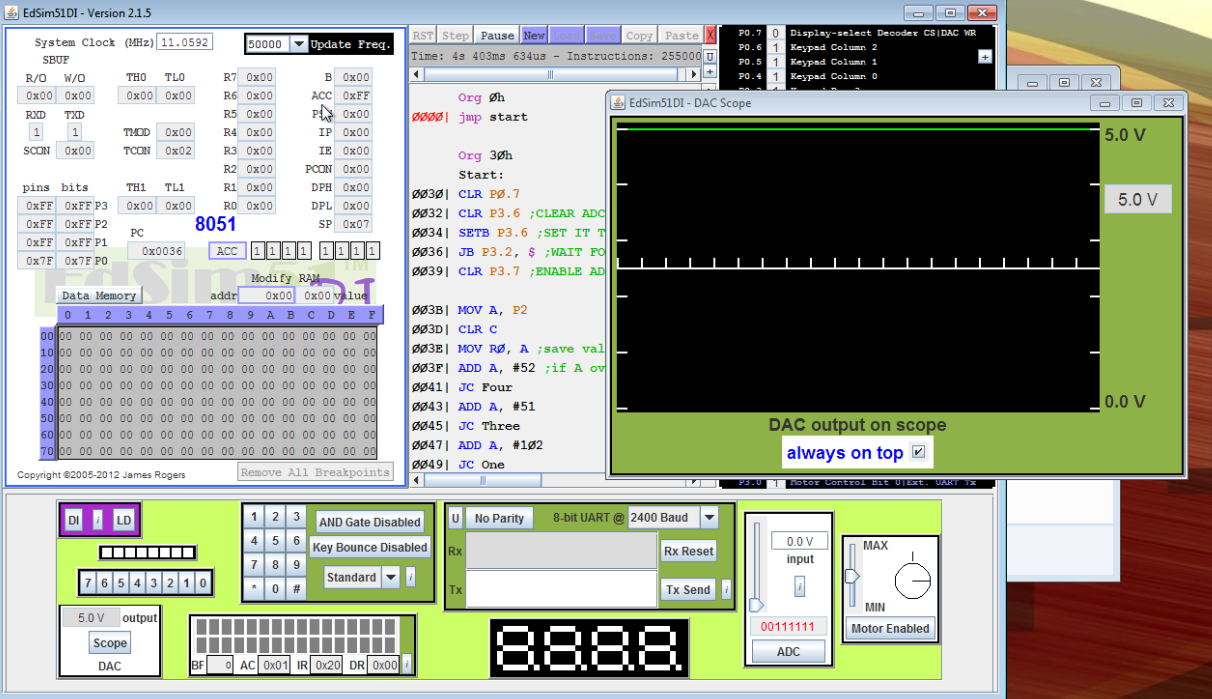
Write a program that converts a signal from Analog to Digital and processes input signal according to several piecewise linear functions. Afterwards, the signal will be converted back from digital to analog.

**Equipment:**

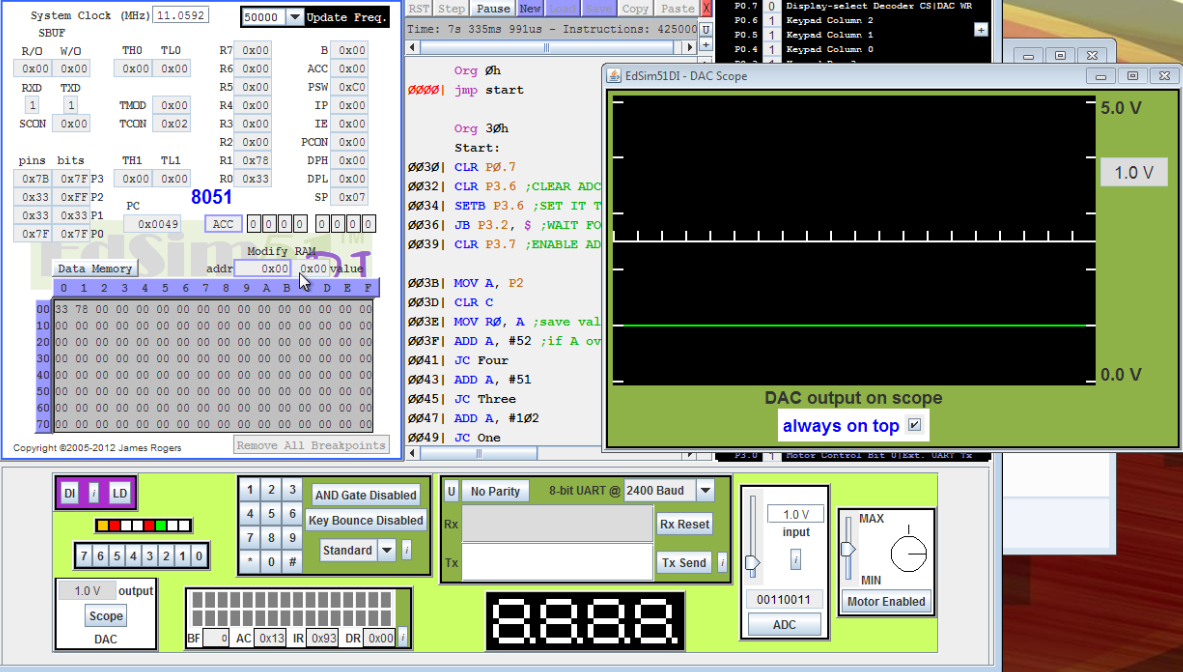
* Text editor and 8051 ASM assembler
* Step debugger to execute program one instruction at a time
* Data memory, accumulator, code memory
* Keyboard panel, Text monitor panel

**Results:**

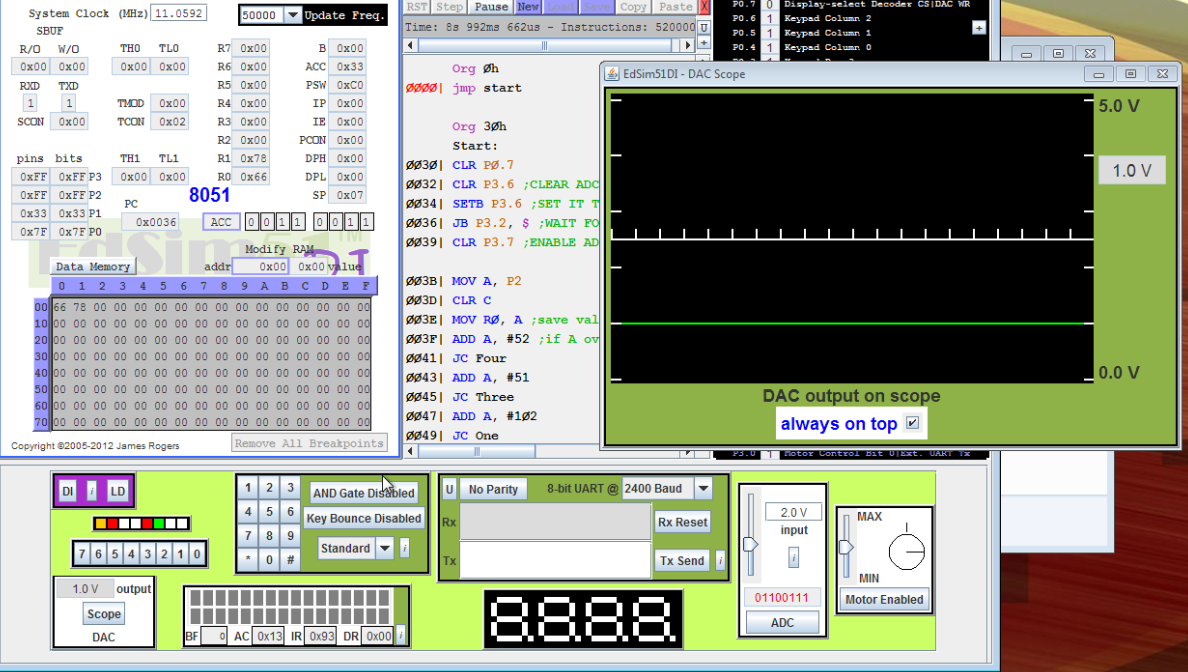
Output at 0V is 5V:



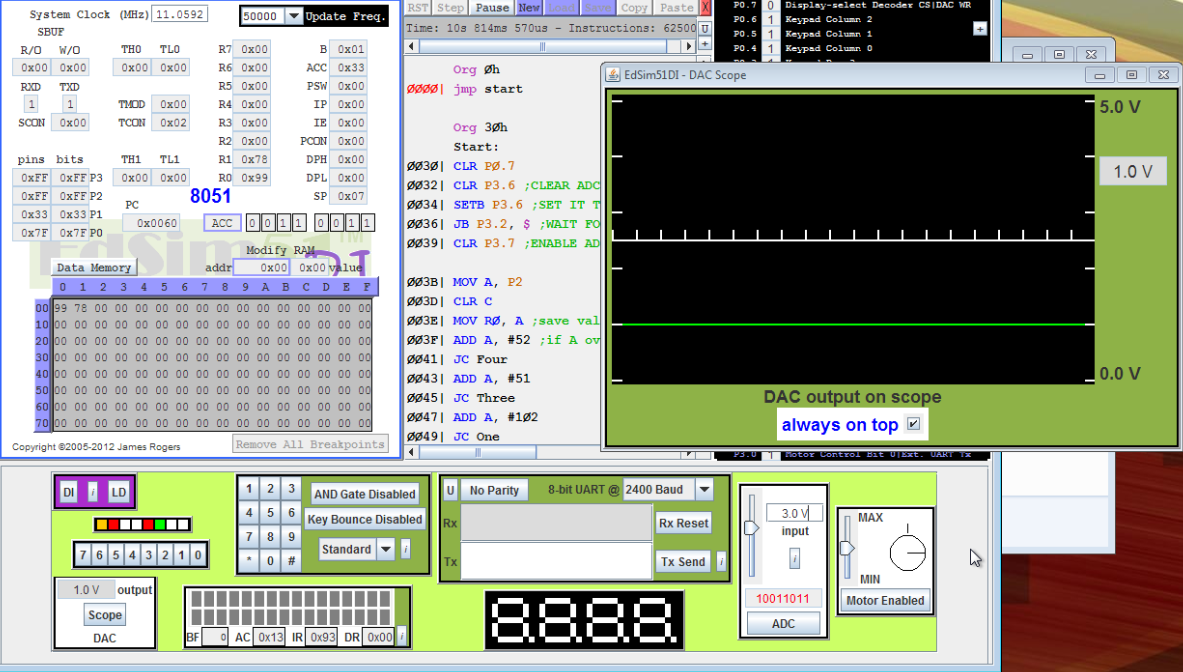
Output at 1V is 1V:



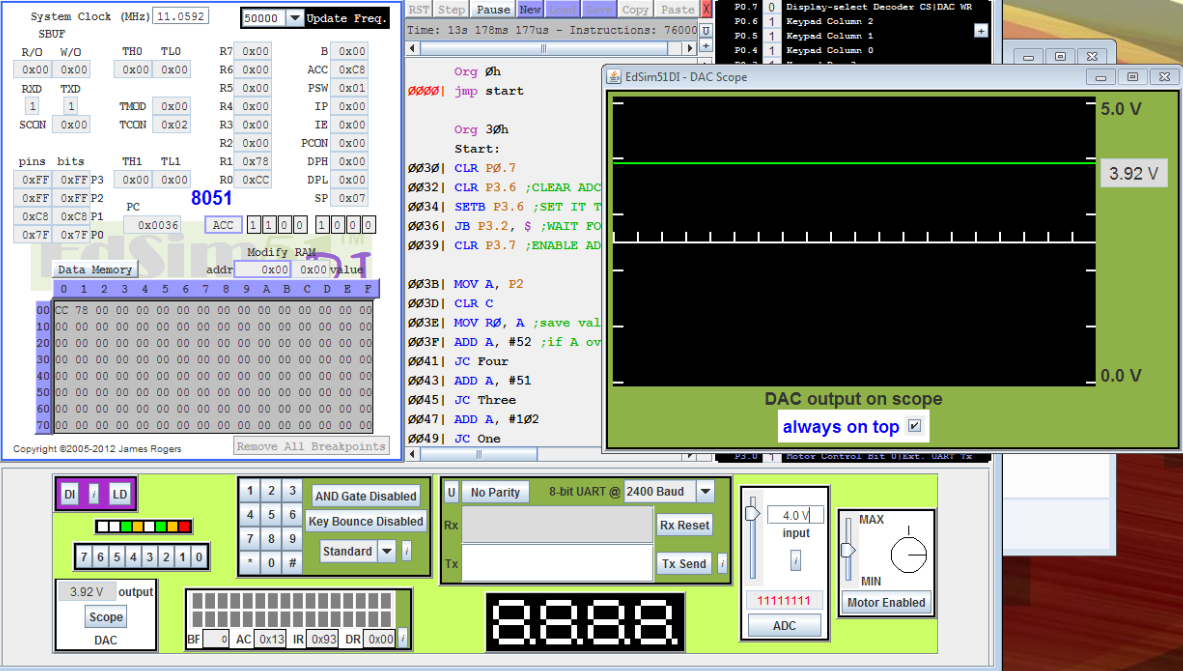
Output at 2V is 1V:



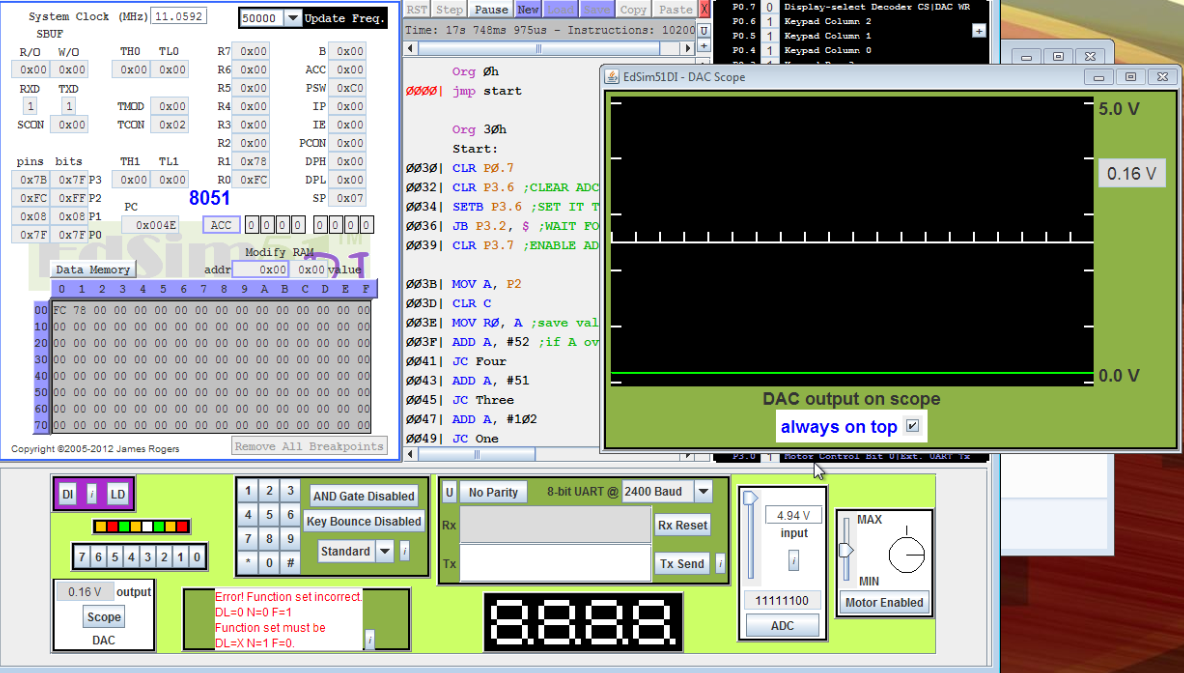
Output at 3V is 1V:



Output at 4V is 4V:



Output at 5V is 0V:



**Conclusion:**

This lab taught the basic functions necessary for converting from analog to digital and back. Also equally important, it taught basic signal processing in assembly. While in C++, these instructions could have been implemented much quicker using if statements, these statements had to be implemented in assembly through manipulation of flags, such as the carry flag. Basic multiplication and addition functions are much more difficult to program too because of the availability of only registers A and B to execute thee functions, as well as their limit of only being able to process 8 bit numbers. While these instructions are less intuitive to program, they are overall quicker to execute in assembly than in C because of more control over each machine cycle that is allowed in assembly.

**Code:**

Org 0h  
jmp start

Org 30h  
Start:  
CLR P0.7  
CLR P3.6 ;CLEAR ADC WR LINE  
SETB P3.6 ;SET IT TO GENERATE A POSITIVE EDGE TO START CURRENT A/D CONVERSION  
JB P3.2, $ ;WAIT FOR COMPLETION OF CURRENT CONVERSION  
CLR P3.7 ;ENABLE ADC OUTPUT LINES TO ALLOW READING THE A/D CONVERSION RESULT

MOV A, P2

CLR C  
MOV R0, A ;save value of Vin  
ADD A, #52 ;if A overflows, Vin is greater than 4V  
JC Four  
ADD A, #51 ; If A overflows, Vin is greater than 3  
JC Three   
ADD A, #102 ; If A overflows, Vin is greater than1  
JC One   
JMP Zero

Four:  
CLR A  
MOV A, #0FFh ;20-4Vin = 4(5V-Vin)  
SUBB A, R0 ;implementing 5V-Vin  
MOV B, #4  
MUL AB  
MOV P1, A  
JMP Finish

Three:  
CLR A  
MOV B, #3  
MOV A, R0  
MUL AB  
CLR C  
SUBB A, #098h  
MOV P1, A  
JMP Finish

One:  
mov P1, #033h ; Constant term for 1<V<3  
JMP Finish

Zero:  
CLR A  
MOV A, R0  
MOV B, #4  
MUL AB  
MOV R1, A ;saves value of 4\*Vin  
CLR A  
MOV A, #0FFh  
SUBB A, R1  
MOV P1, A  
JMP Finish

Finish:  
SETB P3.7  
JMP START

End