**Casey Sobecks**

**ECE 362**

**Post-Lab #6**

**Introduction:**

The purpose of this laboratory is to understand the usage and implementations of registers. Along with this, the application of stacks as a way to pass information to and from a subroutine will be taught. More specific to the microcontroller used, the HCS12’s method of banking memory and paging should be understood by the end of the laboratory

**Lab 6.1:**

Objective/Purpose:

The purpose of this lab is to understand how stacks are manipulated by pushing and pulling values to and from registers.

Explanation:

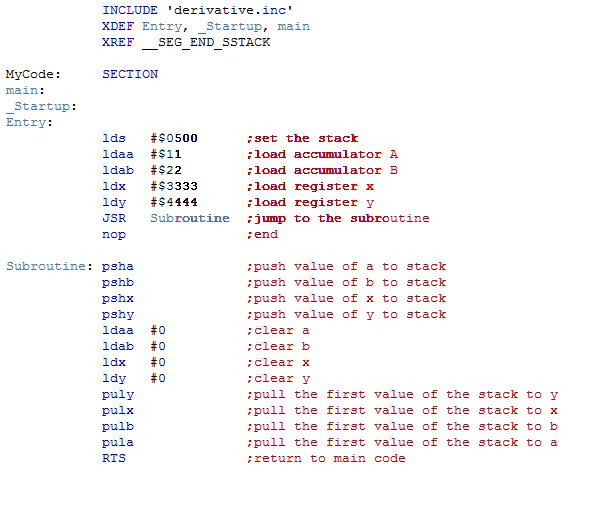
First the stack is set and the values are loaded into the registers and accumulators. Then the program jumps to the subroutine where it pushes the values in each register and then loads zero to each of them. After, it pulls the values in the order that it pushed them.

Results:

The code worked as expected. The stack is as follows



Code:



**Lab 6.2:**

Objective/Purpose:

The purpose of this lab is to use the call instruction to structure a stack. This will be used to pass parameters to and from the stack.

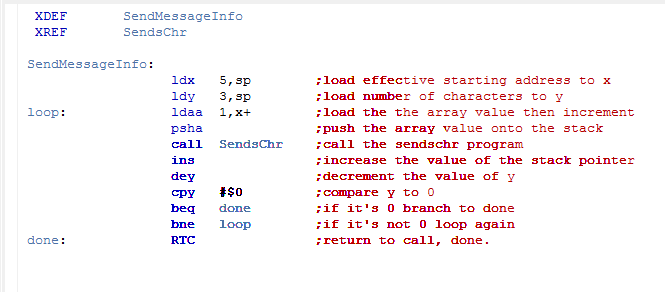
Explanation:

The code loads the values of x and y from the stack and then moves to the loop. Here it loads a value to a and pushes it. It calls the sendchr function and increases the value of the stack pointer. Then it decreases y and checks to see if the loop is complete.

Results:

The program works as expected. It send the data and ran through the information completely

Code:



**Lab 6.3.:**

Objective/Purpose:

The purpose of this lab is to use assembly language to read values from the potentiometer and relay them to the LCD screen on the provided board.

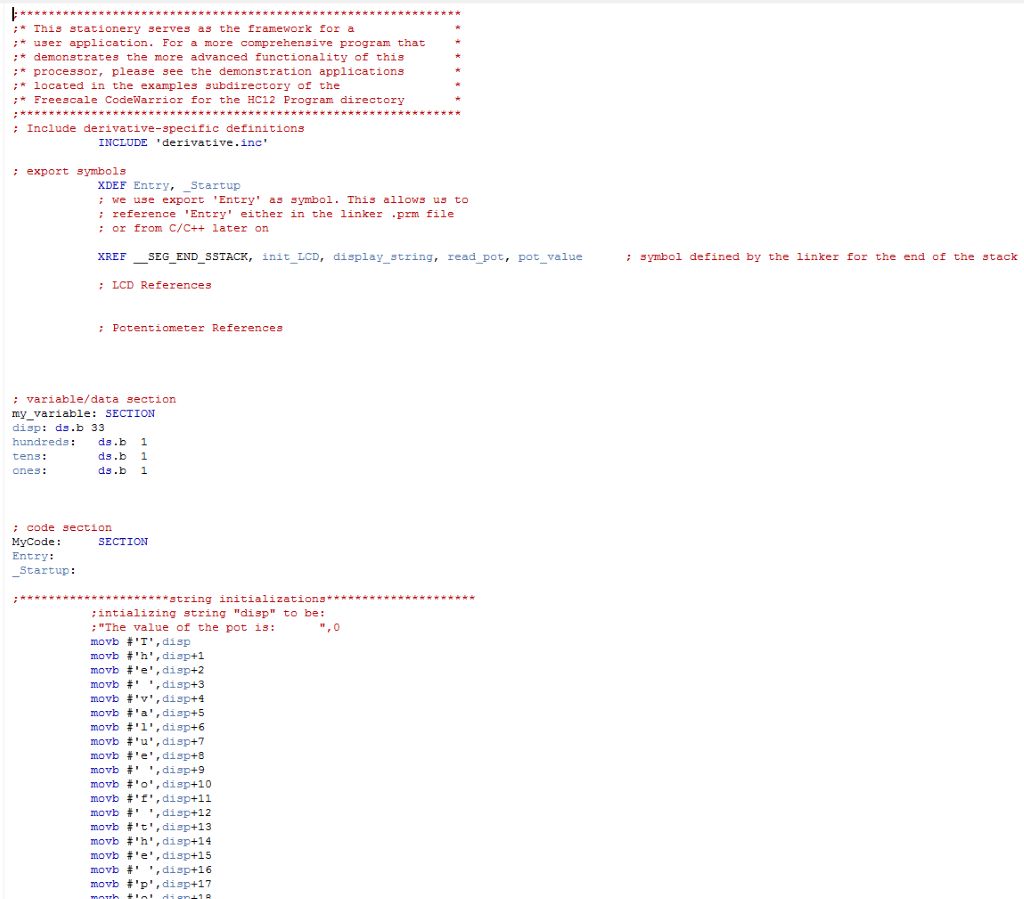
Explanation:

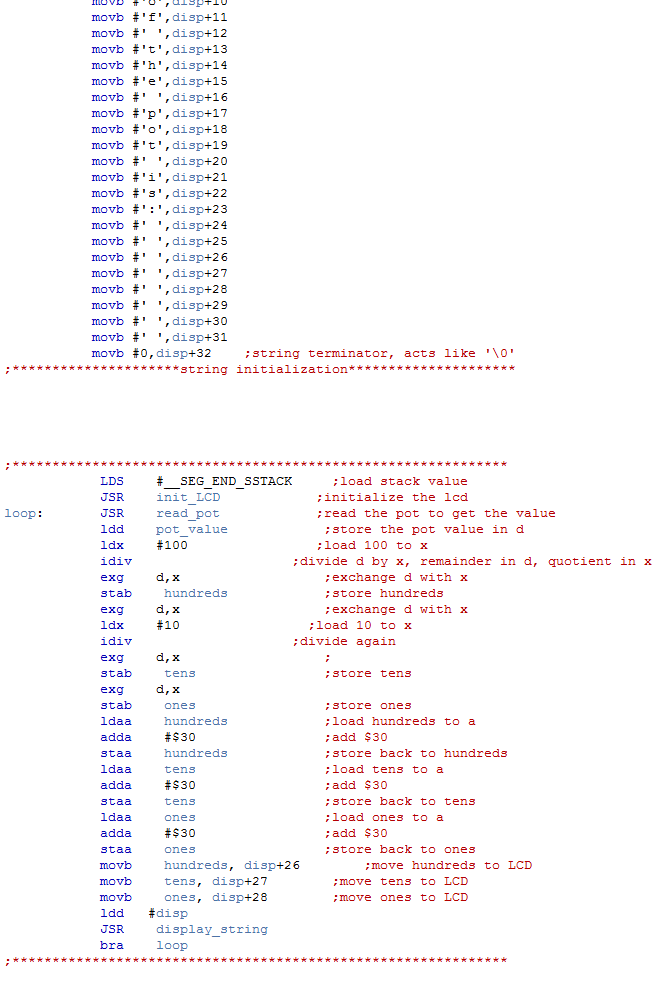
The code first loads the variables and then runs the given information. From there, it loads the stack and jumps around to get the pot value. This value is divided by 100 and then 10 to get the value in the hundreds, tens, and ones place. The values are then moved into 3 sequential lcd places to be displayed.

Results:

The screen printed out the potentiometer value correctly. Initially, the screen would not display and was flashing weird patterns, but this was fixed in the debugger.

Code:





LDS #\_\_SEG\_END\_SSTACK ;load stack value

JSR init\_LCD ;initialize the lcd

JSR display\_string ;initialize the lcd to display

loop: JSR read\_pot ;read the pot to get the value

ldd pot\_value ;store the pot value in d

ldx #100 ;load 100 to x

idiv ;divide d by x, remainder in d, quotient in x

exg d,x

stab hundreds ;store hundreds

exg d,x

ldx #10 ;load 10 to x

idiv ;divide again

exg d,x

stab tens ;store tens

exg d,x

stab ones ;store ones

ldaa hundreds ;load hundreds to a

adda #$30 ;add $30

staa hundreds ;store back to hundreds

ldaa tens ;load tens to a

adda #$30 ;add $30

staa tens ;store back to tens

ldaa ones ;load ones to a

adda #$30 ;add $30

staa ones ;store back to ones

movb hundreds, disp+26 ;move hundreds to LCD

movb tens, disp+27 ;move tens to LCD

movb ones, disp+28 ;move ones to LCD

JSR display\_string

bra loop

**Conclusion:**

In this lab the concepts of passing and pulling on the stack was learned. Along with this, the implementation of the pot was learned. This will be important later on because the pot will be used to modify values or change the program in real time. One of the difficult parts to implement was the use of the LCD screen. The biggest snag I ran into was simply displaying the program. I forgot to include the ldd #disp which meant that I wasted a bit of time trying to fix an easy issue. Overall this lab did not pose to many difficulties other than the time spent understand the use of the implementation of the stack and reading the prelab.