**Austin Sypolt**

**ECE 362**

**Post-Lab #5**

**Introduction:**

In this lab we will learn how to use logic and bit instructions to change LEDs as well as learning the application of a hex keypad and further lessons in utilizing the stepper motor on our HSC12 boards.

**Lab 5.1.1: Logic Instructions**

Objective/Purpose: Clears bit 6 of VAR\_1 and stores it back into VAR\_1. Then sets bit 4 of VAR\_1 and stores it in VAR\_1. Then using the AND instruction it will wait for bit 1 (switch 2) of the dipswitches to go low then send VAR\_1 to the LEDs. Finally testing for switch 2 to wait until it goes high after going low.

Code:

XDEF Entry

XREF \_\_SEG\_END\_SSTACK

Variables: Section

var1: ds.b 1

Constants: Section

port\_s: equ $248

Code: Section

main:

\_Startup:

Entry:

Movb #$FF, $24a

Adda var1

anda #%10111111

oraa #%00000100

nop

Variables: Section

var1: ds.b 1

Constants: Section

port\_s: equ $248

port\_t: equ $240

Code: Section

main:

\_Startup:

Entry:

Movb #$FF, $24a

loop: ldaa port\_t

anda #%00000100

bne loop

loop1: ldaa port\_t

anda #%00000100

beq loop1

staa port\_s

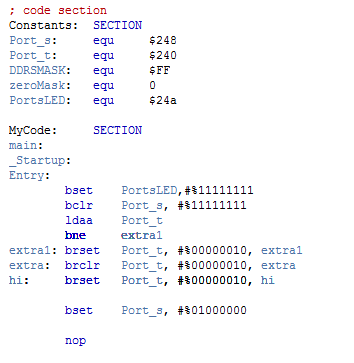
nop

**Lab 5.1.2: Bit Instructions**

Objective/Purpose:

Use BSET and BCLR to initialize and turn off LEDs/DDR in port S.

Code:



**Lab 5.2: Hex Keypad**

Objective/Purpose:

Initializing Port U so that the upper 4 bits are outputs and the lower 4 are inputs. Then the value pressed on the hex keypad should be read in to the right bits and then oscillate with output pressed on the hex keypad.

Code:

XDEF Entry, \_Startup, main

XREF \_\_SEG\_END\_SSTACK ; symbol defined by the linker for the end of the stack

Constants: Section

sequence dc.b $70, $B0, $D0, $E0

port\_u equ $268

lookup dc.b $07, $0B, $0D, $0E

padval dc.b $0, $1, $2, $3, $4, $5, $6, $7, $8, $9, $a, $b, $C, $D, $E, $F

port\_s equ $248

keypadval dc.b $eb, $77, $7b, $7d, $b7, $bb, $bd, $d7, $db, $dd, $e7, $ed, $7e, $be, $de, $ee

MyCode: SECTION

main:

\_Startup:

Entry:

movb #$0F, $24a

movb #$F0, $26A

movb #$F0, $26D

movb #$0F, $26c

lds #\_\_SEG\_END\_SSTACK

reset: ldx #$0

again: ldaa sequence,x

inx

cpx #$5

beq reset

staa port\_u

lds $C027

JSR Debounce

ldaa port\_u

cmpa #$0F

beq again

ldaa port\_u

lds $C037

JSR Keypad

bra again

Debounce: ldy #1000

delay: dey

bne delay

RTS

Keypad:

ldy #0

more: ldab keypadval,y

cpy #$10

beq back

cba

bne no

beq yes

no:

iny

bra more

yes:

ldaa padval,y

staa port\_s

bra back

back:

RTS

**Lab 5.3: Stepper Motor**

Objective/Purpose:

The purpose of this section was to move our stepper motor clockwise while bit 1 is up, counterclockwise while bit 2 is up (nothing is both are up), and then read in delay files to change the speed of our stepper motor dependent on the bit 3 dipswitch.

Code:

 XDEF DelayFast, DelaySlow, Entry, \_Startup, main

 XREF Delay, DelayS, \_\_SEG\_END\_SSTACK      ; symbol defined by the linker for the end of the stack

Constants:    Section

sequence    dc.b    0, $0a, $12, $14, $0c

port\_p     equ     $258

port\_t     equ     $240

DelayFast   dc.w    15000

DelaySlow   dc.w    60000

; code section

MyCode:     SECTION

main:

\_Startup:

Entry:

        LDS    #\_\_SEG\_END\_SSTACK

        movb   #$1e, $25a

        ldx    #0

again:  brset    port\_t, #%00000100, h

        brclr    port\_t, #%00000100, l

    h:  JSR    Delay

        bra    check

    l:  JSR    DelayS

        bra    check

check:  brset port\_t, #%00000001, dcheck

        brclr port\_t, #%00000010, check

        bra     ccw

dcheck: brset    port\_t, #%00000010, check

        bra     cw

ccw:    ldaa     sequence,x

        staa     port\_p

        dex

        cpx    #0000

        beq    reset1

        bra    again

reset1: ldx    #4

        bra    again

cw:     ldaa   sequence,x

        staa   port\_p

        inx

        cpx    #0005

        beq    reset2

        bra    again

reset2: ldx    #1

        bra    again

        Nop

\*Delay Fast Text File\*

    XDEF    Delay

    XREF    DelayFast

Delay:     ldy    DelayFast

    go:    dey

           bne    go

           RTS

\*Delay Slow Text File\*

    XDEF    DelayS

    XREF    DelaySlow

DelayS:    ldy    DelaySlow

    go:    dey

           bne    go

           RTS

**Conclusion:**

Throughout this lab we worked with the use of logical instruction. This allows us to program our HSC12 boards with more versatility, as well as learning additional branching instructions that are useful when dealing with the position of our dipswitches. Additionally, we learned how to retrieve inputs from our hex keypad and then output those values onto our LEDs, using this in the future we could program our keypads to do other things now that the implementation is understood. Finally, we worked on the use of the stepper motor on our circuit boards. We learned how to properly send a sequence to it that allowed for it to continuously spin, as well as adding the functionality to change the direction it spun dependent on the bit 1 and 2 dipswitches. The ability to speed up and slow down the stepper motor was also implemented with the use of the bit 3 dipswitch and changing delay values.

Difficulties came about while working with the hex keypad, as although we were able to display a singular value onto the LEDs, we were never able to implement the shift left and right features mentioned in the lab manual.