**ECE-3226-50:** Lab #5

LEDs, Switches, and Delay Loops

Robert Campbell

**Objective:**

The purpose of this experiment is to familiarize and gain experience applying code to ATmega32 hardware with a debugger attached. Specifically stepping through applied code on the hardware, observing the processing speed by way of the built-in LEDs and buttons, and through those observations the importance of delay loops to allow for smoother human interaction with the hardware.

**Equipment:**

AVR Studio 7

STK500 Starter Kit

JTAG mkII Debugger

**Procedure:**

Part 2: Using LEDs and Delay Loops

A: A program that will turn on LEDs 7,6,1, and 0, or the binary representation of 0xC3.

start:

ldi r16, 0xff

Out DDRB, r16; Sets all of port b to be output

cbi portb, 0

cbi portb, 1

sbi portb, 2

sbi portb, 3

sbi portb, 4

sbi portb, 5

cbi portb, 6

cbi portb, 7 ; sets portb to 1100 0011 or 0xC3 with active low

end: rjmp end

Observations: The desired LEDs lit up, displaying 0xC3 in binary (11000011).

D: A program that that cycles, one LED at a time, from low-bit (LED0) to high-bit (LED7).

start:

ldi r16, 0xff

out DDRB, r16

ldi r16, 0xFE;

sec

call loop;

end: rjmp end

loop:

out portb, r16;

rol r16;

rjmp loop

Observations: All 8 LEDs appear to be lit at once, instead of cycling.

**Question 2\_1:** What is happening on the board? Why?

*All the LEDs appear to be lit at once instead of cycling. It is, in fact, cycling as expected, but too fast to observe individually.*

F: Modify program D by adding a delay loop that takes .5 seconds each time an LED is lit. It should take 4 seconds to light each LED.

start:

ldi r16, 0xff

out DDRB, r16

ldi r16, 0xFE;

sec

call loop;

end: rjmp end

loop:

out portb, r16;

rol r16;

clr r0

ldi r18,0x90

ldi r19, 0x86

ldi r20, 0x01 ; loads a total value of 100,000

in r17, sreg;

call delay

rjmp loop

delay:

subi r18,1;

sbc r19, r0;

sbc r20, r0;

brne delay; delays for 5microseconds \* value of R20:R18

; 100000 \* 5 = 500,000microseconds

out sreg, r17

ret

Observations: Included in Question 2\_2.

**Question 2\_2:** Does the program work as expected? State your observations.

*It does. Each LED is lit for approximately .5 seconds, with one cycle of lighting each LED taking approximately 4 seconds.*

Part 3: Using Switches and Delay Loops

A: A program that will light an LED for as long as the corresponding button is pressed down.

start:

;PORTD = inputs

;PORTB = outputs

Clr r16

clr r17

clr r18 ; null register

Out DDRD, r16

Ldi r16, 0xff

Out DDRB, r16

Out Portb, r16; default output

loop:

in r17, pind

cp r17, r16 ; Checks if all buttons are off (active low, 0xff)

brne turnon

out portb, r16;

rjmp loop;

turnon:

sbis pind, 0

cbi portb, 0

sbis pind, 1

cbi portb, 1

sbis pind, 2

cbi portb, 2

sbis pind, 3

cbi portb, 3

sbis pind, 4

cbi portb, 4

sbis pind, 5

cbi portb, 5

sbis pind, 6

cbi portb, 6

sbis pind, 7

cbi portb, 7

rjmp loop

Observations: When one or multiple buttons were pressed, the corresponding LED or LEDS would light up. They would turn off when the button was released.

E: A program that increments an 8-bit binary value, to be displayed on the board LEDs with LED7 being the most significant bit, when the specified button (SW7) is pressed. Include delay loops to account for any bouncing when pressing the button.

start:

;PIND, 7 = input

;PORTB = outputs

ldi r16, low(ramend)

out spl, r16

ldi r16, high(ramend)

out sph, r16

Clr r16

Out DDRD, r16

Ldi r16, 0xff

Out DDRB, r16

Out portb, r16

LoopPress:

Ldi r17, 1

Call delay

Sbic pind,7

Rjmp LoopPress

LoopRelease:

Ldi r17, 1

Call delay

Sbis pind, 7

Rjmp LoopRelease

Dec r16

Out portb, r16

Rjmp LoopPress

Delay:

Dec r17

Brne delay

ret

Observations: The LEDs increment from 0 to 255 each time SW7 is pressed and rolls back over to 0 when pressed one more time.

**Question 3\_1** Experiment with different delay times. What is an approximate minimum delay time that eliminates debouncing (without unnecessarily extending the delay time)?

*We iterated down from a 600 microsecond delay all the way down to having no delay. Even with no delay in place, there was only very minor bouncing. Using an approximately 3 microsecond delay eliminated any bouncing.*

**Discussion/Conclusion:**

In this experiment, we practiced using different ports as inputs and outputs on the STK500 board, while processing the inputs and outputs with assembly code that we wrote. Using those inputs and outputs, we observed the importance of using delays both with the bouncing of the input buttons and the looping LED output being too fast to observe.

One problem that we ran into was the looping LED display slowly growing, lighting more LEDs in the cycle, until eventually all LEDs were lit. The issue ended up being in the logic in ROL relying on the carry-bit when there were also subtraction with carry instructions happening later in the code. To resolve the problem, we ended up preserving the SREG in register 17 before the delay call and restoring it at the end of the delay loop.