Willis Allstead
CPE 301-1001
Lab #7
October 30, 2016

Assignment Description:

In this lab we were to learn how to use serial communication and timers to control the input to a speaker. We were supposed to generate frequencies of different notes using this speaker. We could get extra credit if we did the sharp (#) frequencies as well.

Problems Encountered:

My one problem was easily solved by fixing the address I was writing data port b to. I had it set wrong, but somehow I could still faintly hear out of the speaker. After some debugging and help from the lab TAs we diagnosed the simple issue.

Lessons Learned:

I for some reason did not know how simple it was to make a speaker play a sound. I don't claim to know how that speaker was built, but I didn't know it was as simple as running a certain frequency through a wire directly to the speaker. It was interesting to learn how this is done.

Description of Completed Lab: (compiled code below)

ANSI-C_Lab_7

```
volatile unsigned char *myTCCR1A = (unsigned char *) 0x80; // Timer/Counter Control Register A
volatile unsigned char *myTCCR1B = (unsigned char *) 0x81; // Timer/Counter Control Register B
volatile unsigned char *myTCCR1C = (unsigned char *) 0x82; // Timer/Counter Control Register C
volatile unsigned char *myTIMSK1 = (unsigned char *) 0x6F; // Timer/Counter Interrupt Mask Register
volatile unsigned int  *myTCNT1  = (unsigned  int *) 0x84; // Timer/Counter Count Register (low & high)
volatile unsigned char *myTIFR1 =  (unsigned char *) 0x36; // Timer/Counter Interrupt Flag Register
volatile unsigned char *portDDRB = (unsigned char *) 0x24;
volatile unsigned char *portB =    (unsigned char *) 0x25;
volatile byte byteRead =0;

void myDelayFor(int);

void setup() {
  /* Initialize Timer1 for NORMAL mode */
  *myTCCR1A = 0;
  *myTCCR1B = 0;
  *myTCCR1C = 0;
  *myTIMSK1 = 0; // Timer 1 should have no interrupts

  *portDDRB |= 0x40;  // Initialize GPIO PortB as output

  Serial.begin(115200);
}

// the loop function runs over and over again forever
void loop() {

   /*  check if data has been sent from the computer: */
  if (Serial.available()) {
    /* read the most recent byte */
    byteRead = Serial.read();
  }

  switch(byteRead) {
    case 'a':
      myDelayFor(18182);
      break;
    case 'A':
      myDelayFor(17167); // A#
      break;
    case 'b':
      myDelayFor(16194);
      break;
    case 'c':
      myDelayFor(15296);
      break;
    case 'C':
      myDelayFor(14440); // C#
      break;
    case 'd':
      myDelayFor(13629);
      break;
    case 'D':
      myDelayFor(12821); // D#
      break;
    case 'e':
      myDelayFor(12140);
      break;
    case 'f':
      myDelayFor(11461);
      break;
    case 'F':
      myDelayFor(10811); // F#
      break;
    case 'g':
      myDelayFor(10204);
      break;
    case 'G':
      myDelayFor(9627); // G#
      break;
    default:
      Serial.write(byteRead);
      myDelayFor(18182);
  }
}

void myDelayFor(int numTicks) {
  *myTCCR1B &= 0xF8; // turn timer to OFF
  *myTCNT1 = (unsigned int) (65536 - numTicks); // for A
  *myTCCR1B |= 0x01; // turn timer ON with pare-scalar of 1
  while(!(*myTIFR1 & 0x01)); // once overflow flag is 1, we will stop
  *myTCCR1B &= 0xF8; // turn timer to OFF (prof. Egbert says this is optional)
  *myTIFR1 |= 0x01; // clear overflow flag bit (by setting to 1 for some reason)

  *portB ^= 0x40;
}
```

Basically to achieve what the lab was asking for I had to create a timer function, pass into that function the amount of ticks to subtract from 65536 for the timer, then run a loop to see if anything changed in the serial communication. In my setup() i just set up the timer in normal mode and started the Serial at a baud of 115200 (I could have used 9600 but a TA had me change it). I also initialized the Port B as output. Then I could use the loop() to read the byte from the serial terminal. Once that byte was read I could use it in the switch statement to decide what frequency I had to generate.

I used a pre-scalar of 1 as suggested by Frank in the lab because that way I didn't have to change the math for each frequency I wanted to generate.

My circuit was simply the arduino portB pin 6 hooked up to a resistor then alligator clips to a speaker. Of course that speaker was grounded as well, to finish the circuit.