Department of Electrical & Computer Engineering
ECED – 3901 Design Methods II
Assignment #3
Due: June 18th, 2015 @ 12:30 PM - Submitted via BBLearn Website (PDF files only), OR printed files in 3901 Mail-Slot at ECED Office

1. Write a C program in AVR Studio to generate a 1.152kHz square wave output on Port B, pin 4 of the ATMega Development Board (fosc = 14.7456MHz). Use the 8-bit Timer/Counter0 Register with Clear Timer on Compare mode and a clock source having a prescaler of 256. Comment each line of the code so that it is apparent that you understand the code.

   See the ECED3204 Lab #4 (at https://github.com/colinoflynn/eced3204/blob/master/labs/lab4_timer/LAB4.pdf) for help with setup of the PWM.

   Download the program to your AVR and test it works. Include a capture of the generated waveform along with the C code.

2. The following code should print '1', but prints '0':

```
#include <stdio.h>

volatile unsigned int running = 1;

int motor_do(unsigned int distance);

int main(void)
{
    printf("Testing routine\n");
    printf("Expected 1, actual: %d\n", motor_do(5));
}

int motor_do(unsigned int distance){
    //If motor not running and object present, turn on motor
    if (running == 0 & distance){
        return 1;
    } else {
        return 0;
    }
}
```

   Determine and fix the bug in this program. You can use the following resources:
   1. A C compiler (such as Atmel Studio, or http://cpp.sh)
   2. The PC-Lint online interface at http://gimpel-online.com//cgi-bin/genPage.py?srcFile=diy.c&cgiScript=analyseCode.py&title=Do-It-Yourself+Example+%28C%29&intro=This+example+allows+you+to+specify+your+own+C+code.&compilerOption=online32.lnt&includeOption={{quotedIncludeOption}}

3. The following code is supposed to take the average of several ADC readings. It seems to sometimes produce slightly incorrect values, and sometimes produce *extremely* incorrect values. Find the two main bugs causing these problems:

```
#include <avr/io.h>
#include <stdio.h>
#include <stdint.h>
#include "sensors.h"


int main(void)
{
    adc_setup();
    PORTB |= 1<<0;
    DDRB |= 1<<0;

    int16_t adc_data[16];
    unsigned int count = 16;

    for(uint8_t i = 0; i < count; i++){
        adc_data[i++] = read_sensor();
    }

    int16_t sum = 0;

    for(uint8_t i = 0; i < count; i++){
        sum += adc_data[i];
    }

    printf("Average reading: %d\n", sum/count);

    return 0;
}
```

As several hints:

1. You cannot copy-paste this code into the PC-Lint software as-is, since it has references to AVR-specific functions/registers. Instead you might need to comment out such calls… for example:

   ```
   adc_setup();
   PORTB |= 1<<0;
   DDRB |= 1<<0;
   ```

   Could be commented out:

   ```
   //adc_setup();
   //PORTB |= 1<<0;
   //DDRB |= 1<<0;
   ```

2. The function read_sensor() along with the sensor.h header need to be replaced. For example you could simply comment out sensors.h, and add a stub function:

```
int16_t read_sensor(void)
{
    static int i;
    int testdata[16] = {-100, 3469, -5000, -500, 454, 55, 52};
    return testdata[i++];
}
```

3. With the above entered you could try running the program in an online C compiler such as cpp.sh, which when fixed should give you the **correct average of -98**.

4. Using the Online PC-Lint page might give you the best chance of finding these errors! See Q2 for an introduction to this software.