CprE 288 Spring 2014 – Homework 4
Due Fri. Feb. 14

Notes:
• Homework answers must be typed using a word editor.
• Homework is individual work. Adhere to the University’s policy relating to the integrity of scholarship. See http://catalog.iastate.edu/academiclife/#regulationstext, “Academic Dishonesty”.
• Late homework is accepted within three days from the due date. Late penalty is 10% per day.

Note: Unless otherwise specified, all problems assume the ATMega128 is being used

Question 1 (10 pts): Binary operations, 2pts each

Write a single line of C code to implement each of the following. Assume the following declarations

```c
unsigned char ch;
int n;
```

a. Set bits 5, 3, 0 of ch to 1, and preserve the remaining bits.
```c
ch = (ch | 0b00101001);
```

b. Clear bits 15, 14, 1, 0 of n to 0, and preserve the remaining bits.
```c
n = (n & 0b0011111111110011);
```

c. Toggle bits 15 through and including 4 of n, and preserve the remaining bits.
```c
n = (n ^ 0b1111111111110000)
```

d. For n, Set bits 13, 12 to 1, Clear bits 3, 2 to 0, Toggle bits 11, 7, and preserve the remaining bits.
```c
n = ((n | 0b0011000000000000) & 0b1111111111110001) ^ 0b0000100010000000
```

e. For ch, rotate the bits to the left one position (bit 7 should end up at bit position 0), and then clear the upper 4 bits of the result, while preserve the remaining bits.
```c
ch = (ch << 1 | ch>> 7) & 0b00001111
```
Question 2 (10 pts): Bit checking, 2 pts each

Write a **single** `if` condition that will evaluate to true for the following checks. Assume the following declarations:

```c
unsigned char ch;
int n;
```

a. For `ch`, any of bits 6, 4, 2, 0 are Set to 1.

```c
if(((ch&0b01000000)==0b01000000) || ((ch&0b00010000)==0b00010000) || ((ch&0b00001000)==
0b0000100) || ((ch&0b00000100)==0b00000100))
```

d. For `n`, all bits 13, 12, 7, 6 are Cleared to 0.

```c
if((n|0b11001111)==0b11001111) || ((n|0b01111111)==0b01111111) ||
((n|0b11111111)==0b11111111) || ((n|0b11111111)==0b11111111)
```
Question 3 (10 pts): Code evaluation, 2 pts each

For each C code fragment, give the final value of the indicated variable after the code fragment is run. If the final value cannot be determine, then give N/A.

a. Final value of ch is __0xa0_____ (in hex)
   char ch = 0x55;
   ch = ch << 5;

b. Final value of int_ptr is __0xFC02____ (in hex)
   int my_int = 0x55; // Assume my_int is located at address 0xFC00
   int *int_ptr;

   my_int = my_int >> 3;
   int_ptr = &my_int;
   int_ptr++;

c. Final value of my_int is __0x33____ (in hex)
   int my_int = 0x55; // Assume my_int is located at address 0xFFFE
   int *int_ptr;

   int_ptr = &my_int;
   int_ptr++;
   my_int++;

   if(int_ptr){
     my_int = 0x44;
   }
   else {
     my_int = 0x33;
   }

d. Final value of flag is __0______ (in decimal)
   signed char ch = -128;
   char flag = 0;

   ch--;

   if(50 > ch){
     flag = 1;
   }

e. Final value of ch is __5______ (in decimal)
   char ch = 5;
char *ch_ptr = &ch;
ch_ptr = ch_ptr << 16;
if(ch_ptr > 0x7FC0){
    ch = 1;
}

**Question 4 (10 pts): Coding practice**

Complete the following function, `max_consecutive_1s`, so that it returns the maximum number of consecutive 1’s in the variable that is passed to it.

For example, for 0x98FF5588 the max number of consecutive 1’s is 8.
Another example, for 0xF007FE05 the max number of consecutive 1’s is 10.

```c
char max_consecutive_1s(unsigned int x)
{
    char max, output;
    int i;
    for(i=31;i>=0;i--){
        if(!(x&(1<<i)))
            max=0;
        else{
            max++;
        }
        if(output<max)
            output = max;
    }
    return output;
}
```
Question 5 (10 pts): I/O Ports, 5pts each. Note each part is independent.

a. Fill in the figure with arrows to indicate the direction (i.e. input or output) for each wire.

DDRA = 0xCC;
DDRB = 0xF3;
DDRC = 0xF0;
DDRD = 0xF0;
DDRE = 0x00;

b. Write C code that will read in button values, and send values to LEDs, based on the figure below. Make sure that your code first sets up the directions of the wires as shown in the figure. Add any variables that you think you may need.
There are 4 buttons (B0, B1, B2, B3), whose values should be stored in elements 0 to 3 of buttons (i.e. button B0 should be stored in button[0]). There are four LEDs (L0, L1, L2, L3) that can have a value of 1 or 0 written (i.e. sent) to them. All wires that do not have an LED or button connected should have their values preserved (i.e. you code should not change their value – they should persist).

```c
void main(void)
{
    unsigned char buttons[4];

    DDRC = 0b10111110;
    DDRD = 0b00001111;

    Buttons[0] = (PINC >> 6) & 1;
    Buttons[1] = (PINC >> 0) & 1;
    Buttons[2] = (PIND >> 7) & 1;
    Buttons[3] = (PIND >> 4) & 1;

    PORTC = (PORTC & 0b11101111) ^ 0b00000100;
    PORTD = (PORTD & 0b11111011) | 0b00000001;
}
```