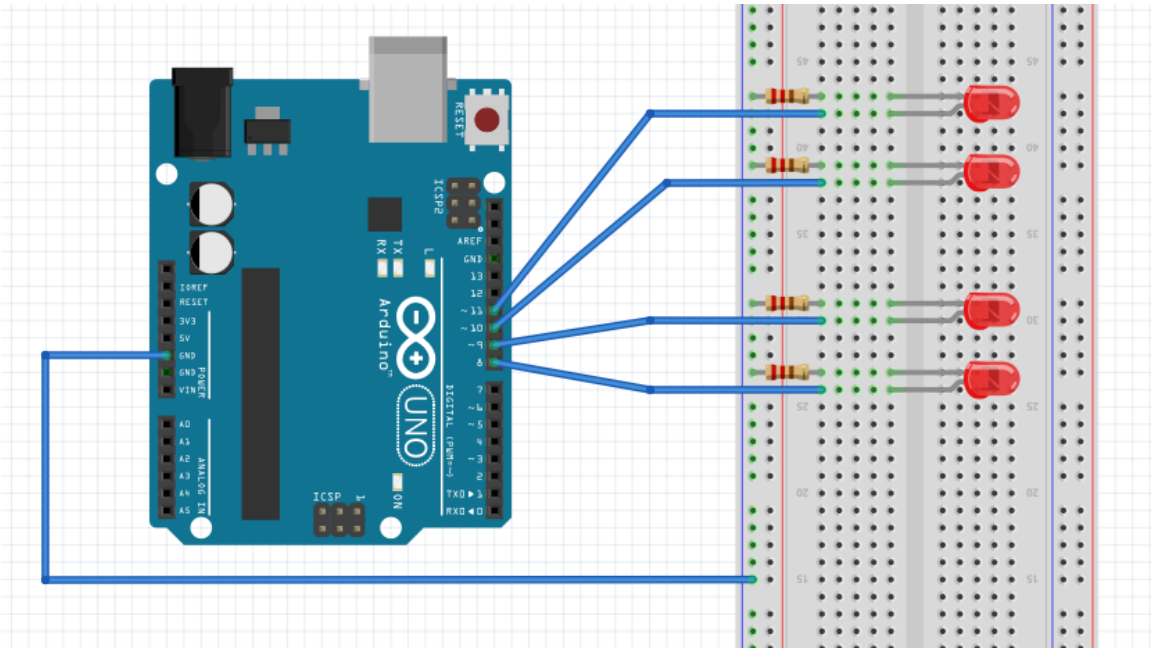
TECH 3230
Lab #3
Bitwise Operators and IF's
Ver 1.0

Purpose: To familiarize students with bitwise and Boolean operators in ANSII C for use in embedded systems

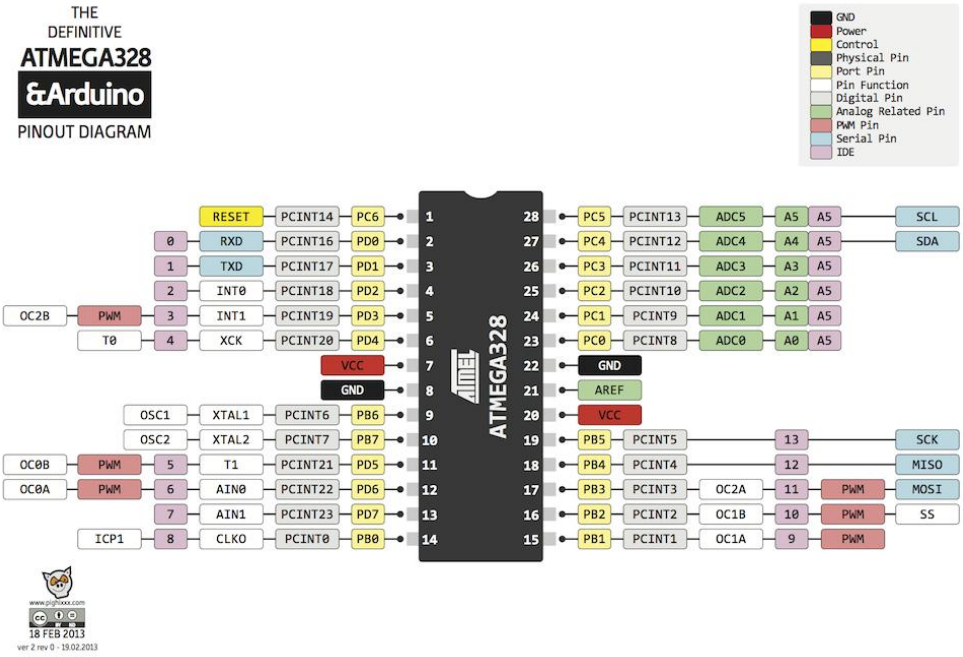
Backround: In embedded systems it is very common to have to manipulate bits within a byte or word. This is used to set up hardware (via registers), to manipulate output pins or to check the status of input pins (I/O). These techniques are typically NOT used when programing for PC's, but are extremely important in embedded programming.

Procedure:

1. Connect the following circuit to the Arduino:



2. Note that the pins are connected to Arduino pins 8-11. Looking at the Arduino Pinout (PINK numbers below) you will see that these pins are PB0-PB3 (PORTB bits 0-3)



[NOTE: the IC pin numbers (written on IC) and Arduino pin numbers (in PINK) are NOT the same]

3. Once the hardware is wired, we will write a program that manipulates the LED's (even though printf is not required for this program, set it up anyway incase you want to debug the program using printf statements)
    a. Since we want to use these pins as output, we need to set them as outputs using register DDRB:

18.4.3. Port B Data Direction Register

When addressing I/O Registers as data space using LD and ST instructions, the provided offset must be used. When using the I/O specific commands IN and OUT, the offset is reduced by 0x20, resulting in an I/O address offset within 0x00 - 0x3F.

Name: DDRB
Offset: 0x24
Reset: 0x00
Property: When addressing as I/O Register: address offset is 0x04

Bit	7	6	5	4	3	2	1	0
	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – DDRBn: Port B Data Direction [n = 7:0]

The DDxn bits in the DDRx Register select the direction of this pin. If DDxn is written to '1', Pxn is configured as an output pin. If DDxn is written to '0', Pxn is configured as an input pin.

Since we want to use the bottom 4 bits as ouputs (to turn on LED's) we will set DDRB=0x0F. This should be done after atmel_start_init();

    b. We will now manipulate the register PORTB to turn on/off various leds.

18.4.2. Port B Data Register

When addressing I/O Registers as data space using LD and ST instructions, the provided offset must be used. When using the I/O specific commands IN and OUT, the offset is reduced by 0x20, resulting in an I/O address offset within 0x00 - 0x3F.

Name: PORTB
Offset: 0x25
Reset: 0x00
Property: When addressing as I/O Register: address offset is 0x05

Bit	7	6	5	4	3	2	1	0
	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Bits 7:0 – PORTBn: Port B Data [n = 0:7]

PORTxn as an Output:

If a '1' is written to the bit when the pin is configured as an output pin, the port pin is driven high. If a '0' is written to the bit when the pin is configured as an output pin, the port pin is driven low.

c. Start off by setting PORTB=0x00 (all off)
d. We will want to pause after each output change, to do this, #include <util/delay.h> after the #include <atmel_start.h> then you can use the function _delay_ms(1000); to delay a half second (1000ms = 1.0 sec)
e. Now do the following in order (with a 1000ms delay after each change in output):
    i. Declare a variable called "i"
    ii. Use a bitwise OR to turn on bits PORTB2 and PORTB0
    iii. Use a bitwise XOR to turn off PORTB2 and PORTB0 and turn on PORTB3 and PORTB1
    iv. Use a bitwise NOT to invert the LED's
    v. Use a bitwise XOR to turn off PORTB3 and PORTB1 and turn on PORTB2 and PORTB0
    vi. Use a bitwise NOT to invert the LED's again
    vii. Use a bitwise AND to turn off all but PORTB0
    viii. Use a FOR LOOP (using i as the counter) and a SHIFT BIT LEFT to make it turn on PORTB1, PORTB2 and PORTB3 (pause between each one)
    ix. Use a FOR LOOP (using i as the counter) and a SHIFT BIT RIGHT to make it turn on PORTB2, PORTB1, PORTB0 (pause between each one)
    x. Make i decimal 10 and send it to PORTB

xi. Now do a series of if statements as per table below:

if statement	**True**	**False**
If(0)	PORTB=0x0F	PORTB=0x00
if(1)	PORTB=0x0F	PORTB=0x00
if(10)	PORTB=0x0F	PORTB=0x00
If(i==10)	PORTB=0x0F	PORTB=0x00
if(i)	PORTB=0x0F	PORTB=0x00
if(-i)	PORTB=0x0F	PORTB=0x00
If(i!=10)	PORTB=0x0F	PORTB=0x00

Make sure to do a pause for 1000ms after the if structure

Between each if structure, make PORTB=0x09 and pause 1000ms (as a marker for the end of an IF structure)

f. Put a comment by each if statement indicating it was true or false
g. Demo to instructor
h. Zip project and submit via online submission system.