Joseph Sharp Halpin

CPE301 – SPRING 2018

Design Assignment 2

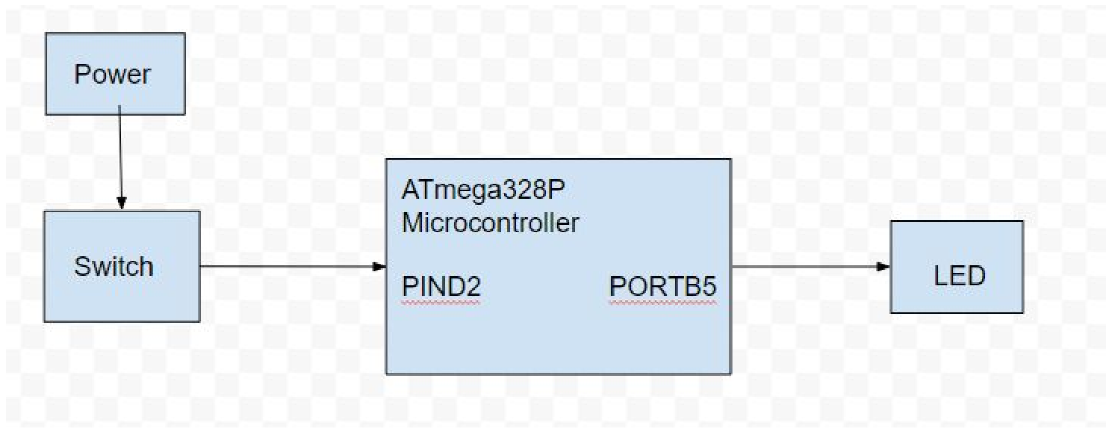
DO NOT REMOVE THIS PAGE DURING SUBMISSION:

The student understands that all required components should be submitted in complete for grading of this assignment.

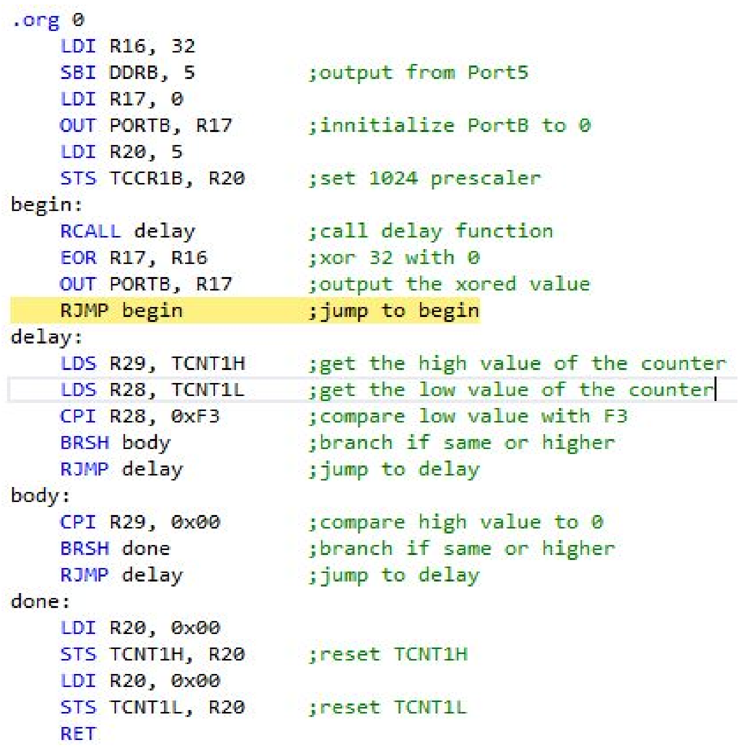
NO	SUBMISSION ITEM	COMPLETED (Y/N)	MARKS (/MAX)
1	COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS		
2.	INITIAL CODE OF TASK 1/A		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 2/B		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 3/C		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 4/D		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 5/E		
4.	SCHEMATICS		
5.	SCREENSHOTS OF EACH TASK OUTPUT		
5.	SCREENSHOT OF EACH DEMO		
6.	VIDEO LINKS OF EACH DEMO		
7.	GOOGLECODE LINK OF THE DA		

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

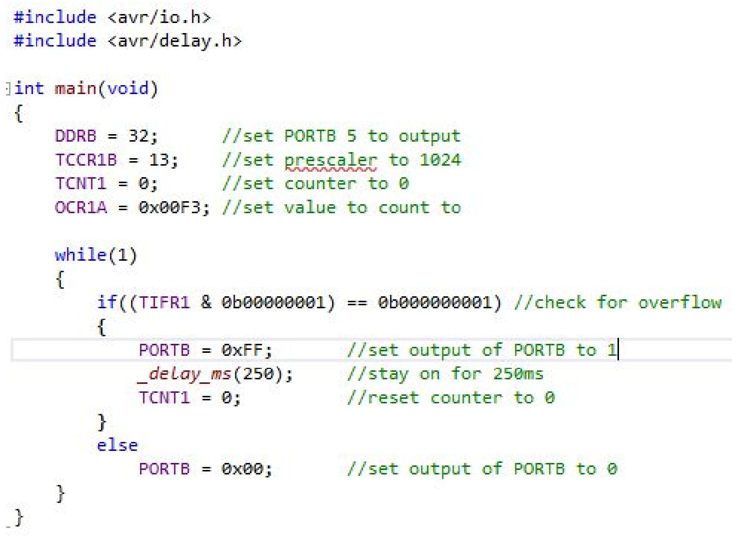
Components used: ATmega328p chip, atmel studio, red LED, push button, and 10K resistors.



2. DEVELOPED CODE OF TASK 1/A AND 1/C

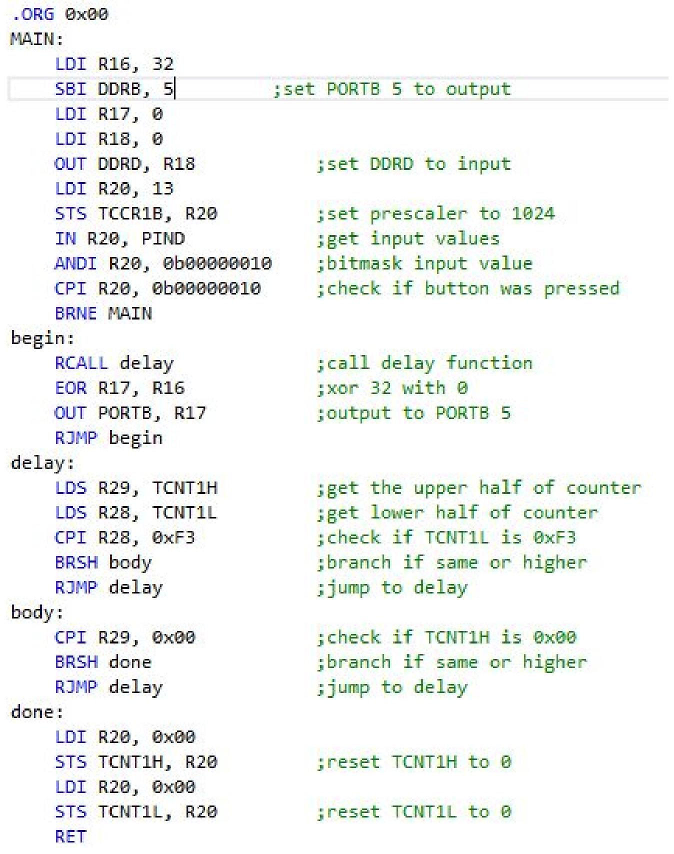


```
.org 0
    LDI R16, 32
    SBI DDRB, 5         ;output from Port5
    LDI R17, 0
    OUT PORTB, R17      ;innitialize PortB to 0
    LDI R20, 5
    STS TCCR1B, R20     ;set 1024 prescaler
begin:
    RCALL delay         ;call delay function
    EOR R17, R16        ;xor 32 with 0
    OUT PORTB, R17      ;output the xored value
    RJMP begin          ;jump to begin
delay:
    LDS R29, TCNT1H     ;get the high value of the counter
    LDS R28, TCNT1L     ;get the low value of the counter
    CPI R28, 0xF3       ;compare low value with F3
    BRSH body           ;branch if same or higher
    RJMP delay          ;jump to delay
body:
    CPI R29, 0x00       ;compare high value to 0
    BRSH done           ;branch if same or higher
    RJMP delay          ;jump to delay
done:
    LDI R20, 0x00
    STS TCNT1H, R20     ;reset TCNT1H
    LDI R20, 0x00
    STS TCNT1L, R20     ;reset TCNT1L
    RET
```

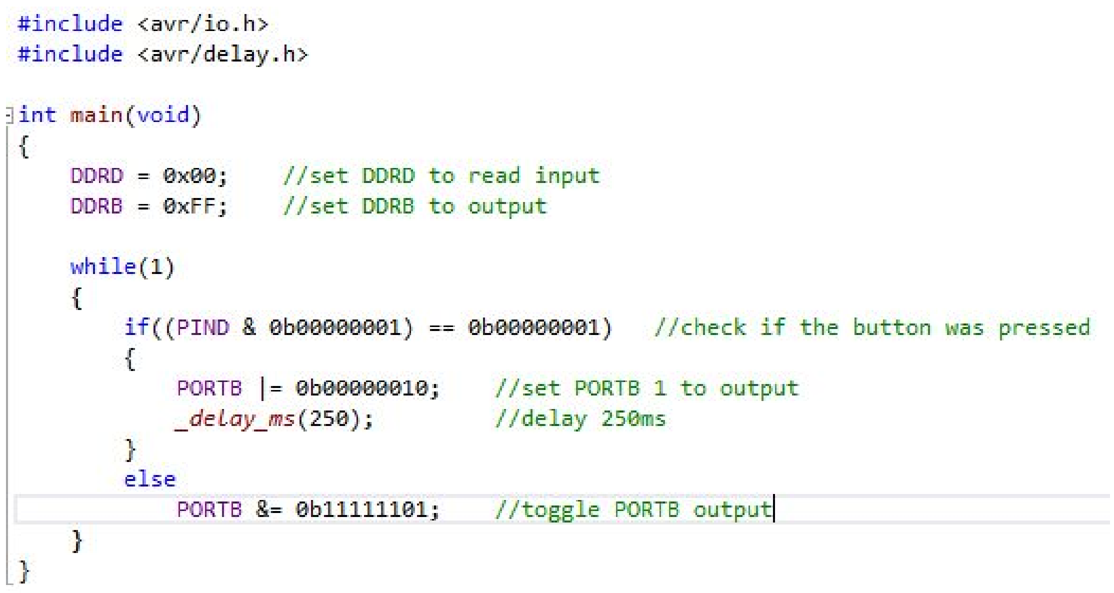


```
#include <avr/io.h>
#include <avr/delay.h>

int main(void)
{
    DDRB = 32;      //set PORTB 5 to output
    TCCR1B = 13;    //set prescaler to 1024
    TCNT1 = 0;      //set counter to 0
    OCR1A = 0x00F3; //set value to count to

    while(1)
    {
        if((TIFR1 & 0b00000001) == 0b000000001) //check for overflow
        {
            PORTB = 0xFF;       //set output of PORTB to 1
            _delay_ms(250);     //stay on for 250ms
            TCNT1 = 0;          //reset counter to 0
        }
        else
            PORTB = 0x00;       //set output of PORTB to 0
    }
}
```

DEVELOPED CODE OF TASK 2/A AND 2/C

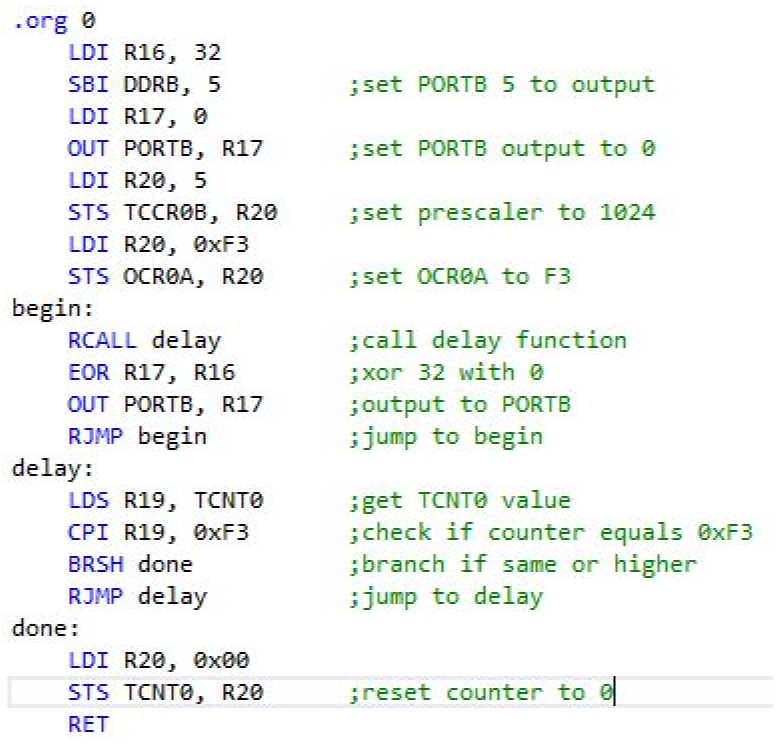


```
.ORG 0x00
MAIN:
    LDI R16, 32
    SBI DDRB, 5         ;set PORTB 5 to output
    LDI R17, 0
    LDI R18, 0
    OUT DDRD, R18           ;set DDRD to input
    LDI R20, 13
    STS TCCR1B, R20         ;set prescaler to 1024
    IN R20, PIND            ;get input values
    ANDI R20, 0b00000010    ;bitmask input value
    CPI R20, 0b00000010     ;check if button was pressed
    BRNE MAIN
begin:
    RCALL delay             ;call delay function
    EOR R17, R16            ;xor 32 with 0
    OUT PORTB, R17          ;output to PORTB 5
    RJMP begin
delay:
    LDS R29, TCNT1H         ;get the upper half of counter
    LDS R28, TCNT1L         ;get lower half of counter
    CPI R28, 0xF3           ;check if TCNT1L is 0xF3
    BRSH body               ;branch if same or higher
    RJMP delay              ;jump to delay
body:
    CPI R29, 0x00           ;check if TCNT1H is 0x00
    BRSH done               ;branch if same or higher
    RJMP delay              ;jump to delay
done:
    LDI R20, 0x00
    STS TCNT1H, R20         ;reset TCNT1H to 0
    LDI R20, 0x00
    STS TCNT1L, R20         ;reset TCNT1L to 0
    RET
```

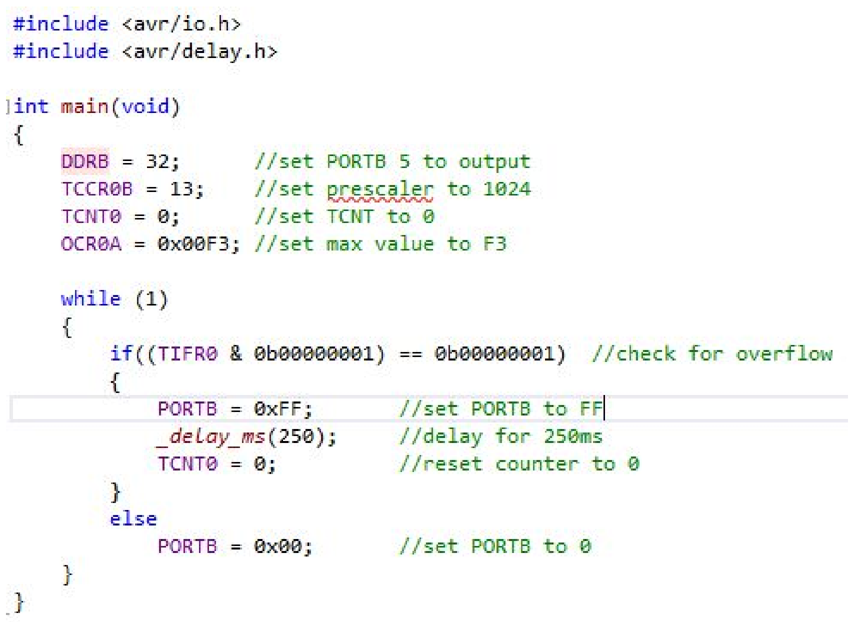


```c
#include <avr/io.h>
#include <avr/delay.h>

int main(void)
{
    DDRD = 0x00;    //set DDRD to read input
    DDRB = 0xFF;    //set DDRB to output

    while(1)
    {
        if((PIND & 0b00000001) == 0b00000001)   //check if the button was pressed
        {
            PORTB |= 0b00000010;    //set PORTB 1 to output
            _delay_ms(250);         //delay 250ms
        }
        else
            PORTB &= 0b11111101;    //toggle PORTB output
    }
}
```

DEVELOPED CODE OF TASK 3/A AND 3/C

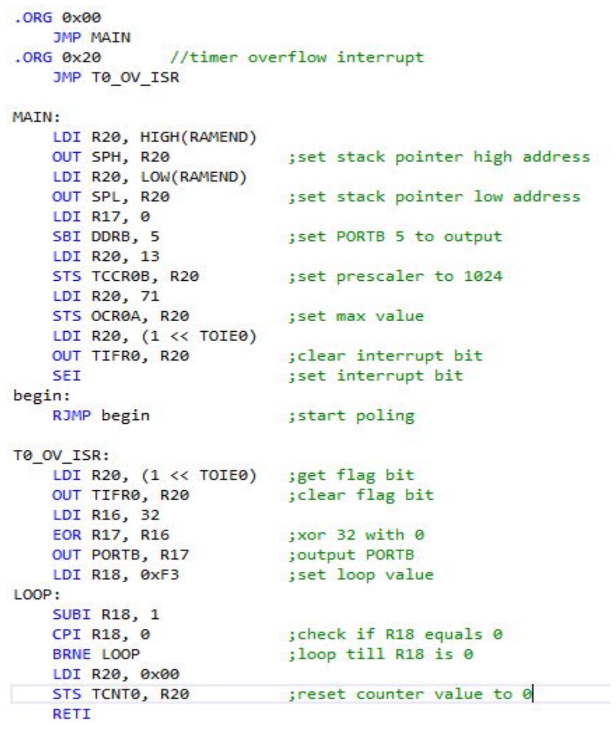


```asm
.org 0
    LDI R16, 32
    SBI DDRB, 5         ;set PORTB 5 to output
    LDI R17, 0
    OUT PORTB, R17      ;set PORTB output to 0
    LDI R20, 5
    STS TCCR0B, R20     ;set prescaler to 1024
    LDI R20, 0xF3
    STS OCR0A, R20      ;set OCR0A to F3
begin:
    RCALL delay         ;call delay function
    EOR R17, R16        ;xor 32 with 0
    OUT PORTB, R17      ;output to PORTB
    RJMP begin          ;jump to begin
delay:
    LDS R19, TCNT0      ;get TCNT0 value
    CPI R19, 0xF3       ;check if counter equals 0xF3
    BRSH done           ;branch if same or higher
    RJMP delay          ;jump to delay
done:
    LDI R20, 0x00
    STS TCNT0, R20      ;reset counter to 0
    RET
```

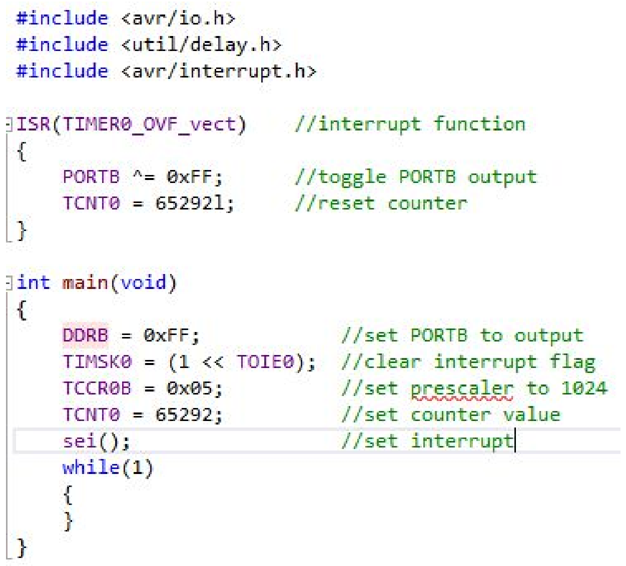


```c
#include <avr/io.h>
#include <avr/delay.h>

int main(void)
{
    DDRB = 32;      //set PORTB 5 to output
    TCCR0B = 13;    //set prescaler to 1024
    TCNT0 = 0;      //set TCNT to 0
    OCR0A = 0x00F3; //set max value to F3

    while (1)
    {
        if((TIFR0 & 0b00000001) == 0b00000001)  //check for overflow
        {
            PORTB = 0xFF;       //set PORTB to FF
            _delay_ms(250);     //delay for 250ms
            TCNT0 = 0;          //reset counter to 0
        }
        else
            PORTB = 0x00;       //set PORTB to 0
    }
}
```

DEVELOPED CODE FOR TASK 4/A AND 4/C

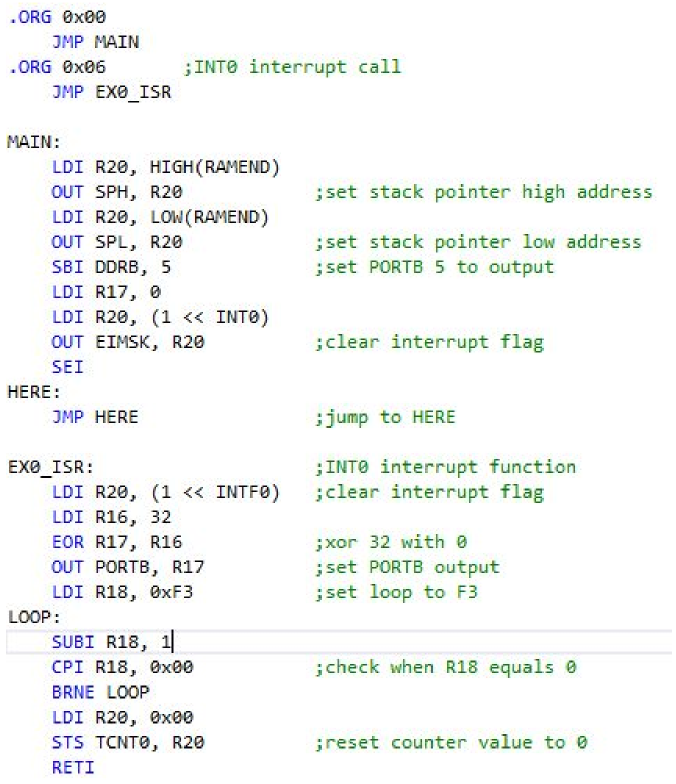


```asm
.ORG 0x00
    JMP MAIN
.ORG 0x20       //timer overflow interrupt
    JMP T0_OV_ISR

MAIN:
    LDI R20, HIGH(RAMEND)
    OUT SPH, R20            ;set stack pointer high address
    LDI R20, LOW(RAMEND)
    OUT SPL, R20            ;set stack pointer low address
    LDI R17, 0
    SBI DDRB, 5             ;set PORTB 5 to output
    LDI R20, 13
    STS TCCR0B, R20         ;set prescaler to 1024
    LDI R20, 71
    STS OCR0A, R20          ;set max value
    LDI R20, (1 << TOIE0)
    OUT TIFR0, R20          ;clear interrupt bit
    SEI                     ;set interrupt bit
begin:
    RJMP begin              ;start poling

T0_OV_ISR:
    LDI R20, (1 << TOIE0)   ;get flag bit
    OUT TIFR0, R20          ;clear flag bit
    LDI R16, 32
    EOR R17, R16            ;xor 32 with 0
    OUT PORTB, R17          ;output PORTB
    LDI R18, 0xF3           ;set loop value
LOOP:
    SUBI R18, 1
    CPI R18, 0              ;check if R18 equals 0
    BRNE LOOP               ;loop till R18 is 0
    LDI R20, 0x00
    STS TCNT0, R20          ;reset counter value to 0
    RETI
```

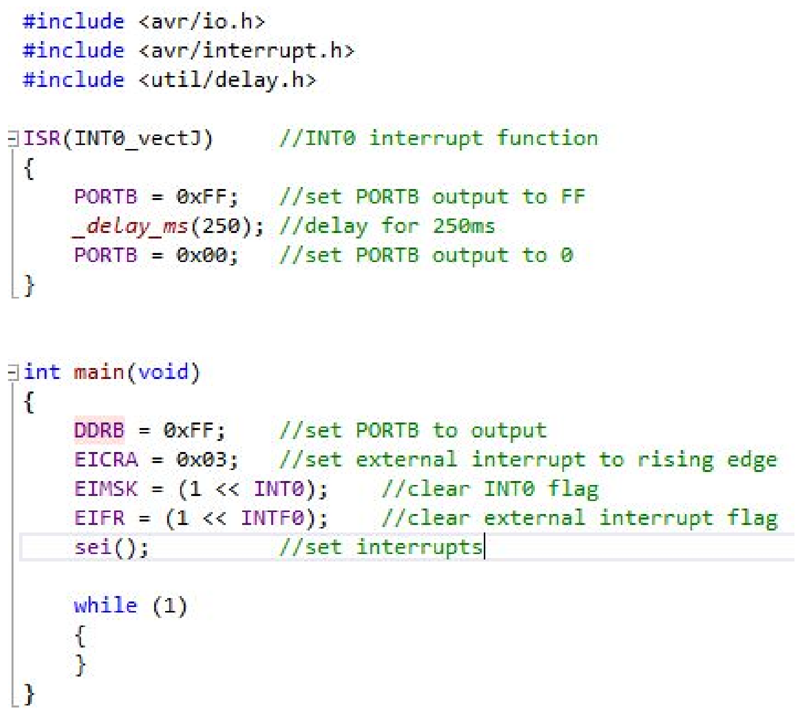


```c
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

ISR(TIMER0_OVF_vect)    //interrupt function
{
    PORTB ^= 0xFF;      //toggle PORTB output
    TCNT0 = 65292l;     //reset counter
}

int main(void)
{
    DDRB = 0xFF;            //set PORTB to output
    TIMSK0 = (1 << TOIE0);  //clear interrupt flag
    TCCR0B = 0x05;          //set prescaler to 1024
    TCNT0 = 65292;          //set counter value
    sei();                  //set interrupt
    while(1)
    {
    }
}
```

DEVELOPED CODE FOR TASK 5/A AND 5/C



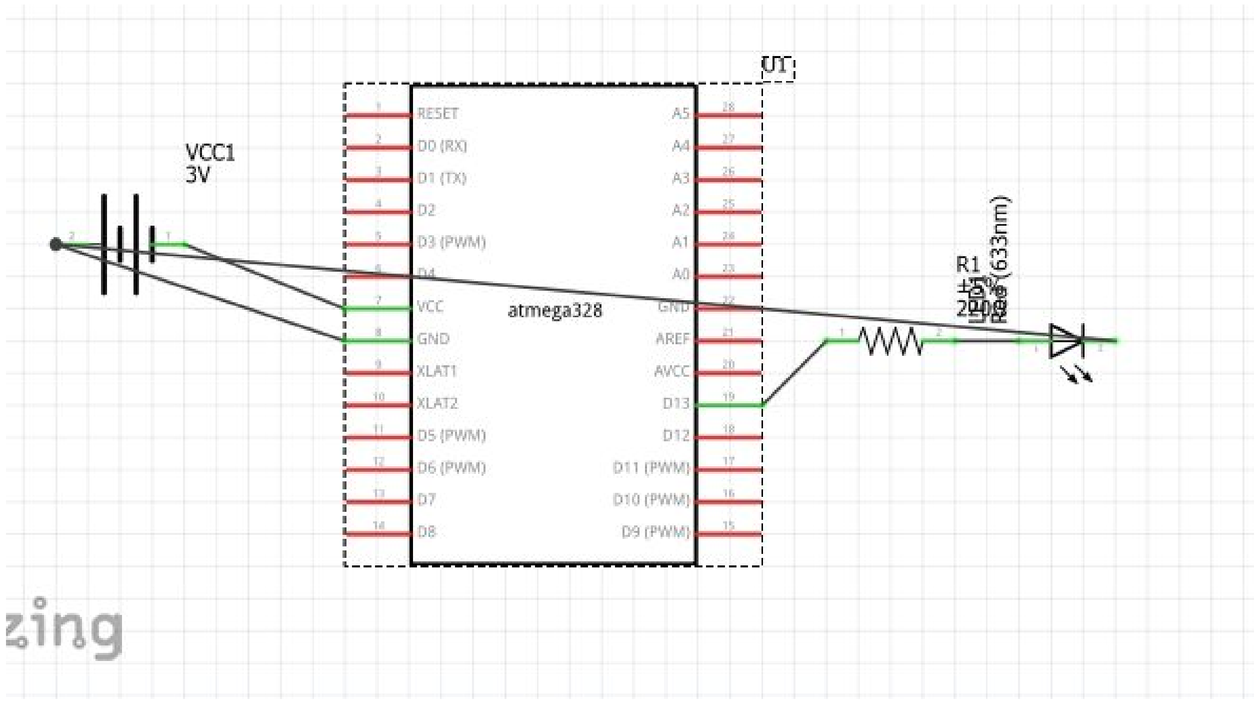
```asm
.ORG 0x00
    JMP MAIN
.ORG 0x06       ;INT0 interrupt call
    JMP EX0_ISR

MAIN:
    LDI R20, HIGH(RAMEND)
    OUT SPH, R20           ;set stack pointer high address
    LDI R20, LOW(RAMEND)
    OUT SPL, R20           ;set stack pointer low address
    SBI DDRB, 5            ;set PORTB 5 to output
    LDI R17, 0
    LDI R20, (1 << INT0)
    OUT EIMSK, R20         ;clear interrupt flag
    SEI
HERE:
    JMP HERE               ;jump to HERE

EX0_ISR:                   ;INT0 interrupt function
    LDI R20, (1 << INTF0)  ;clear interrupt flag
    LDI R16, 32
    EOR R17, R16           ;xor 32 with 0
    OUT PORTB, R17         ;set PORTB output
    LDI R18, 0xF3          ;set loop to F3
LOOP:
    SUBI R18, 1
    CPI R18, 0x00          ;check when R18 equals 0
    BRNE LOOP
    LDI R20, 0x00
    STS TCNT0, R20         ;reset counter value to 0
    RETI
```



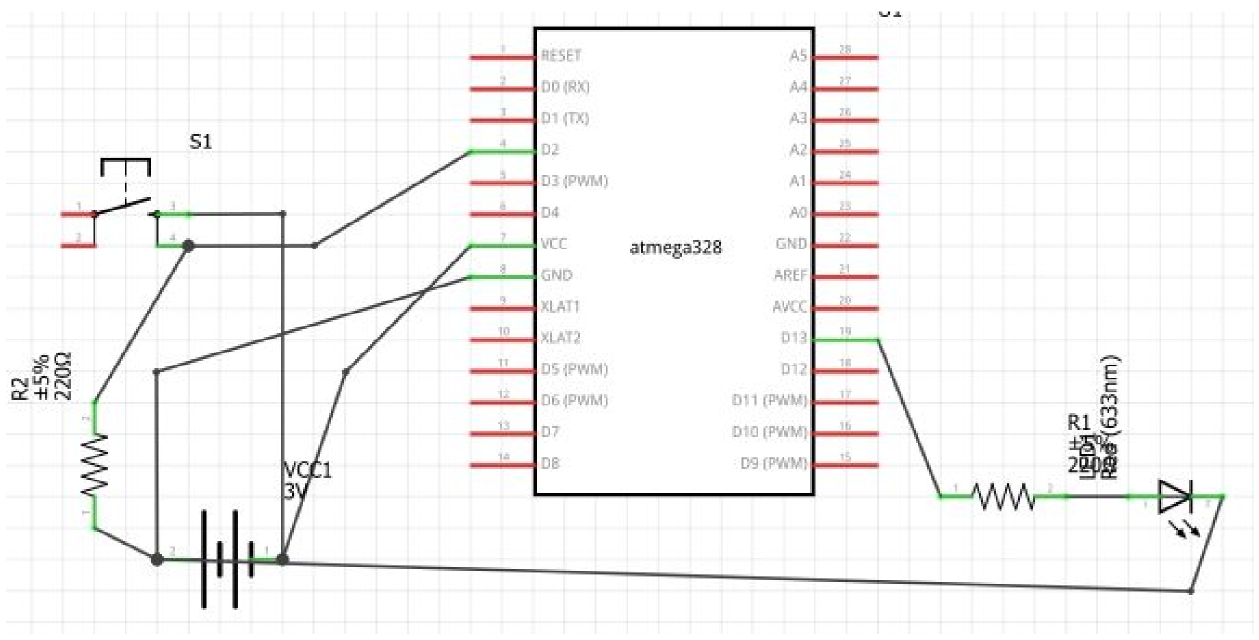
```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

ISR(INT0_vectJ)     //INT0 interrupt function
{
    PORTB = 0xFF;   //set PORTB output to FF
    _delay_ms(250); //delay for 250ms
    PORTB = 0x00;   //set PORTB output to 0
}


int main(void)
{
    DDRB = 0xFF;    //set PORTB to output
    EICRA = 0x03;   //set external interrupt to rising edge
    EIMSK = (1 << INT0);    //clear INT0 flag
    EIFR = (1 << INTF0);    //clear external interrupt flag
    sei();          //set interrupts

    while (1)
    {
    }
}
```

3. SCHEMATICS

Task 1, 3, and 4 schematic:

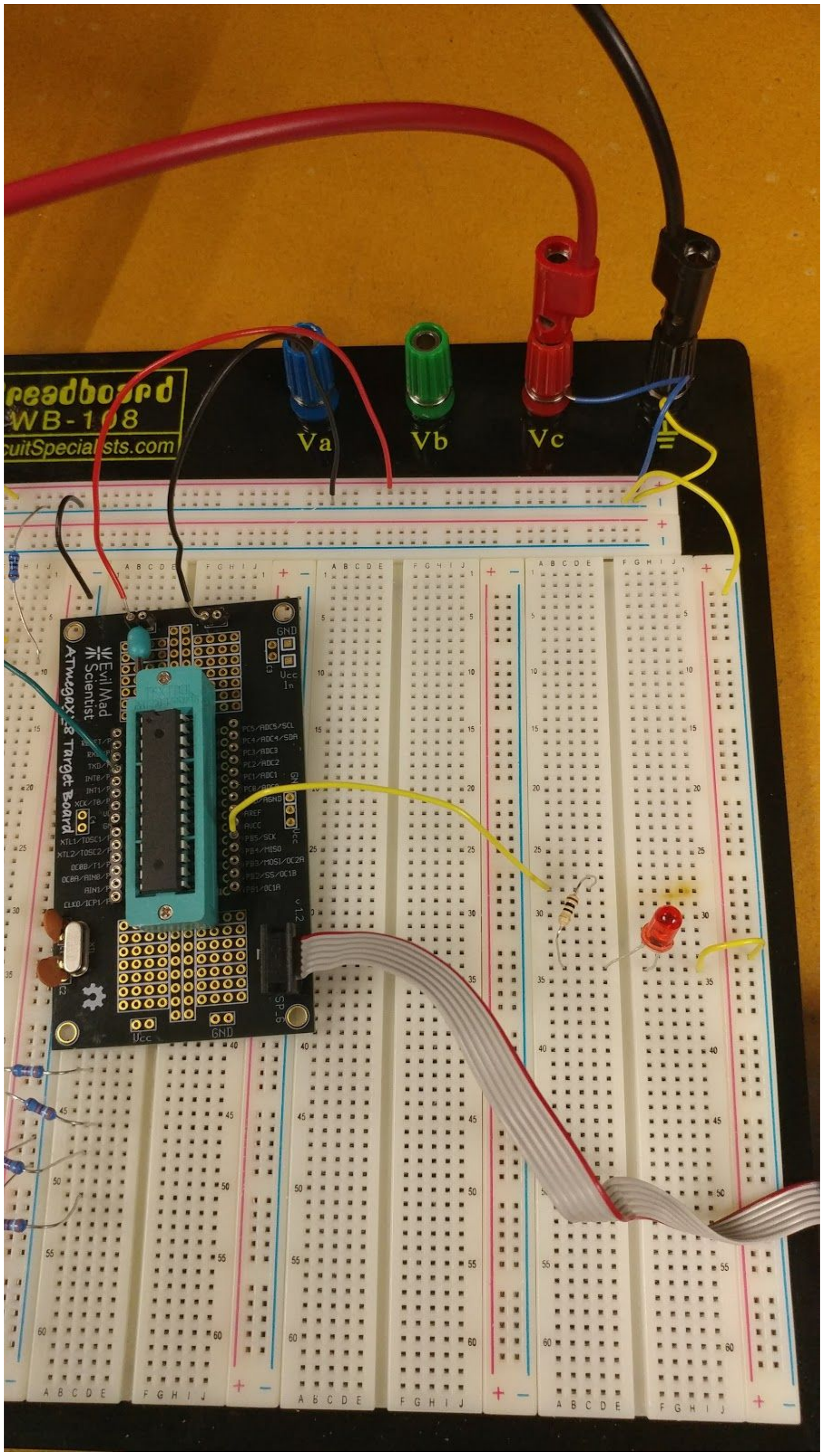


Task 2 and 5 schematic:

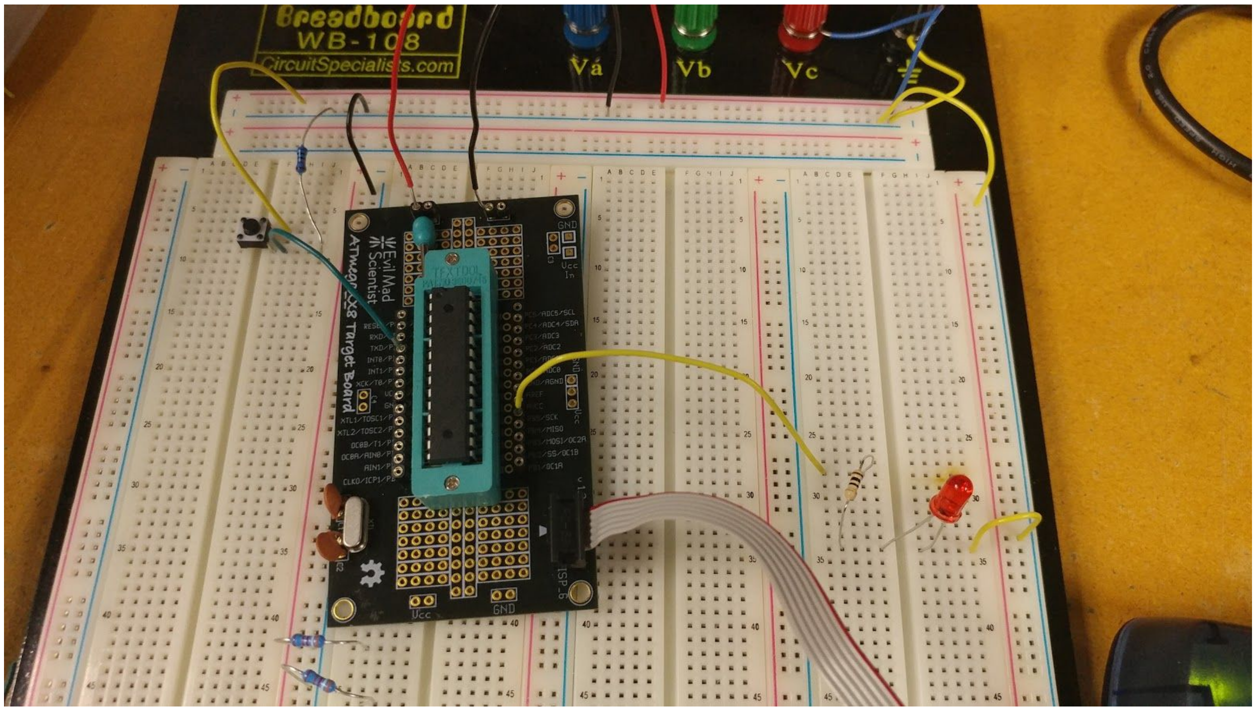

S1
atmega328
RESET
D0 (RX)
D1 (TX)
D2
D3 (PWM)
D4
VCC
GND
XLAT1
XLAT2
D5 (PWM)
D6 (PWM)
D7
D8
A5
A4
A3
A2
A1
A0
GND
AREF
AVCC
D13
D12
D11 (PWM)
D10 (PWM)
D9 (PWM)
R2
±5%
220Ω
VCC1
3V
R1
±5%
220Ω
LED1
Red (633nm)

4. PICTURES OF CIRCUIT SETUP

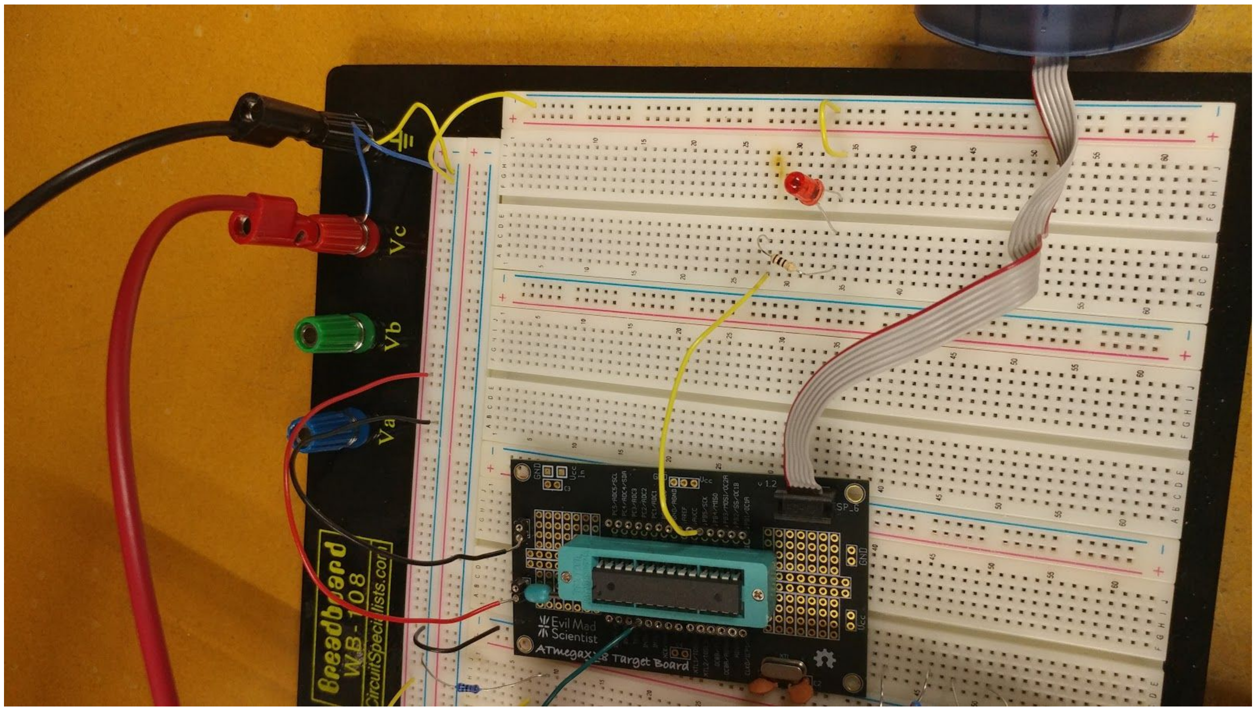
TASK 1:



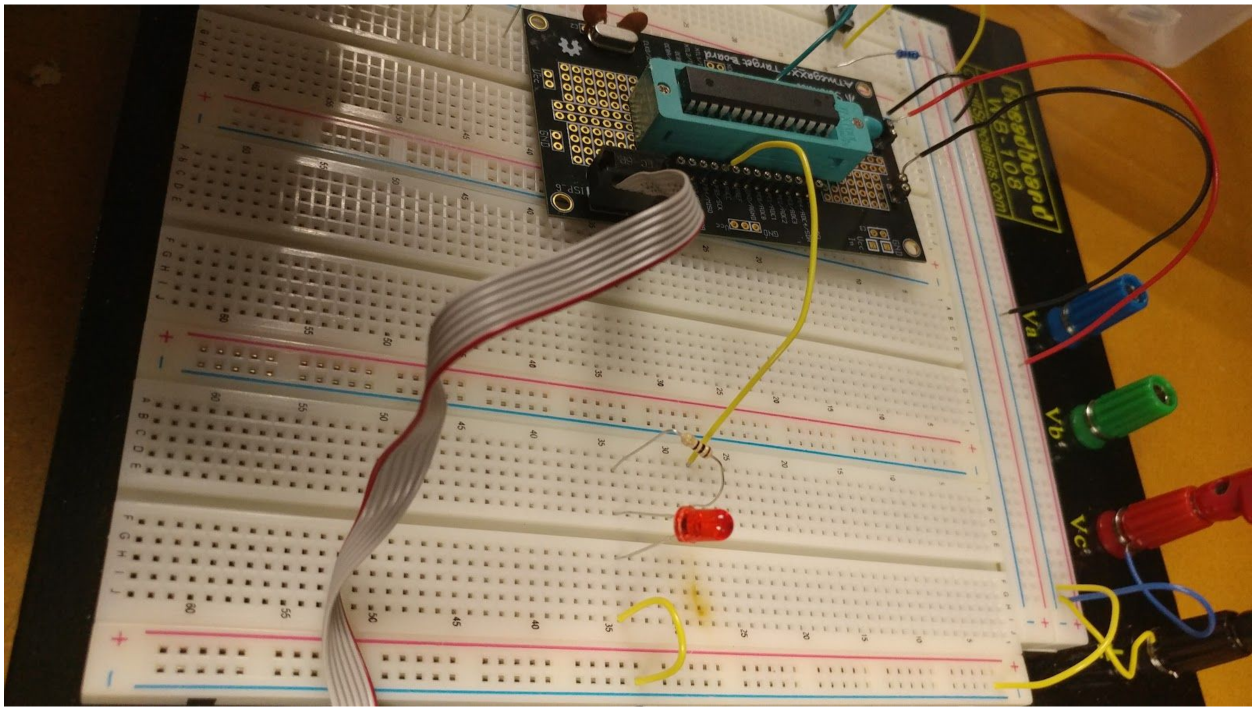
TASK 2:



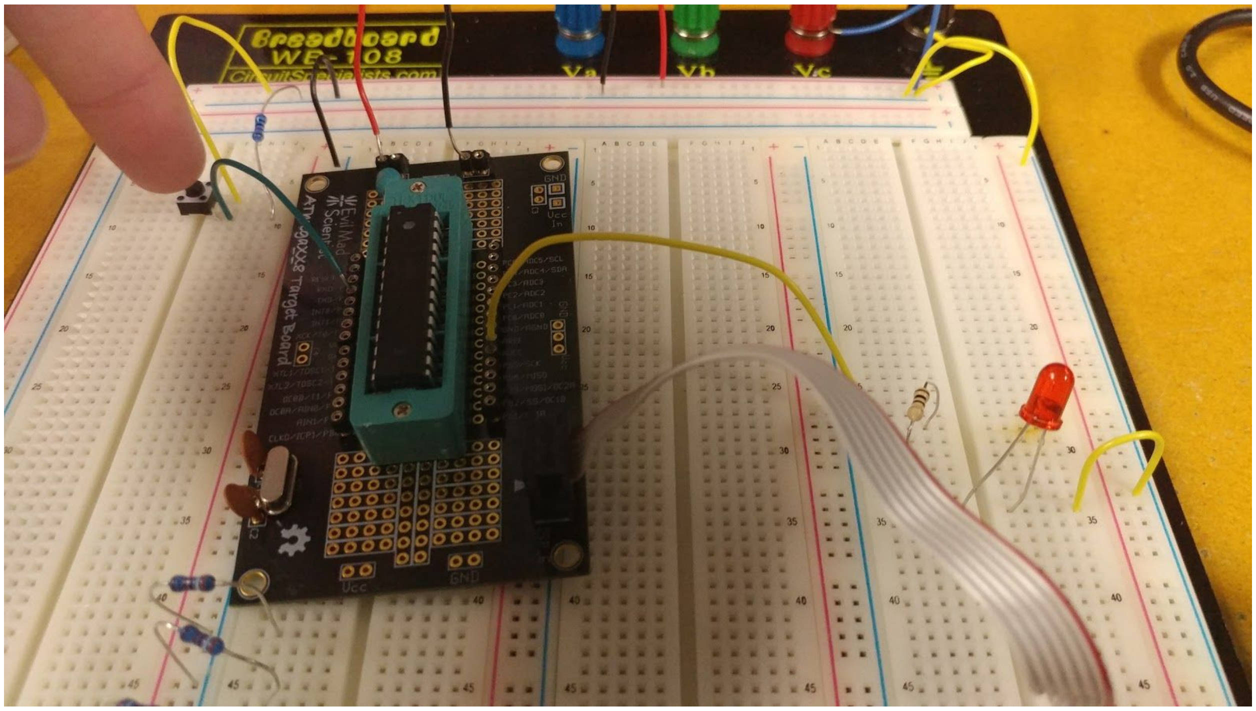
TASK 3:



TASK 4:



TASK 5:



5. VIDEO LINKS OF EACH DEMO

https://youtu.be/kPHP71xCFUY

6. GITHUB LINK OF THIS DA

https://github.com/jsharpin/My-Repos

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"This assignment submission is my own, original work".
Joseph Sharp Halpin