Rimon Sawa

CPE301 – SPRING 2018

Design Assignment 2

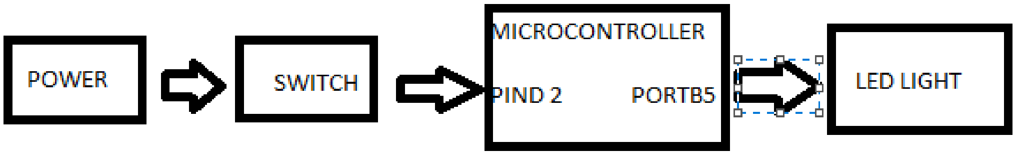
DO NOT REMOVE THIS PAGE DURING SUBMISSION:

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

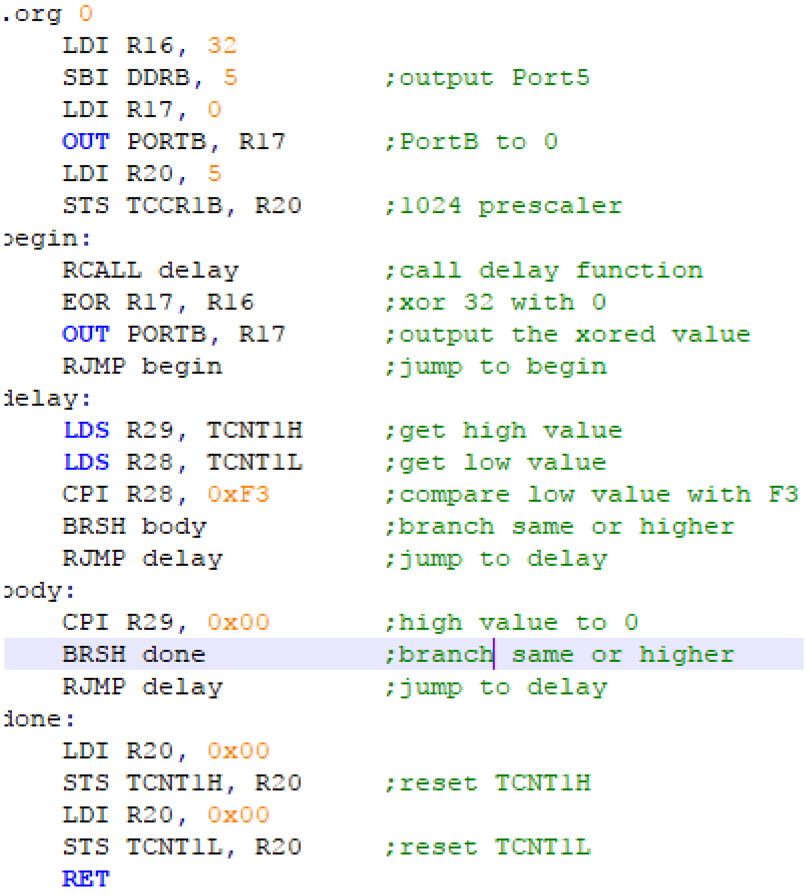
NO	SUBMISSION ITEM	COMPLETED (Y/N)	MARKS (/MAX)
1	COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS		
2.	INITIAL CODE OF TASK 1/A		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 2/B		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 3/C		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 4/D		
3.	INCREMENTAL / DIFFERENTIAL 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 & BLOCK DIAGRAM

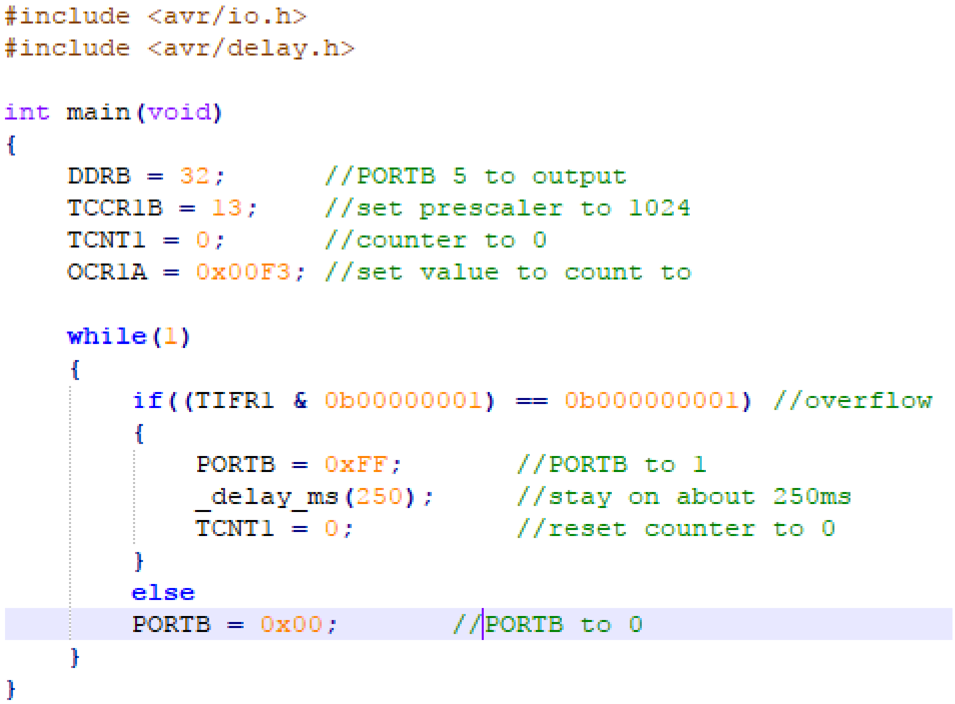
Components: ATmega328p chip, Atmel studio, yellow LED, push button, and 10K resistors.



2. CODE TASK 1/A & 1/C

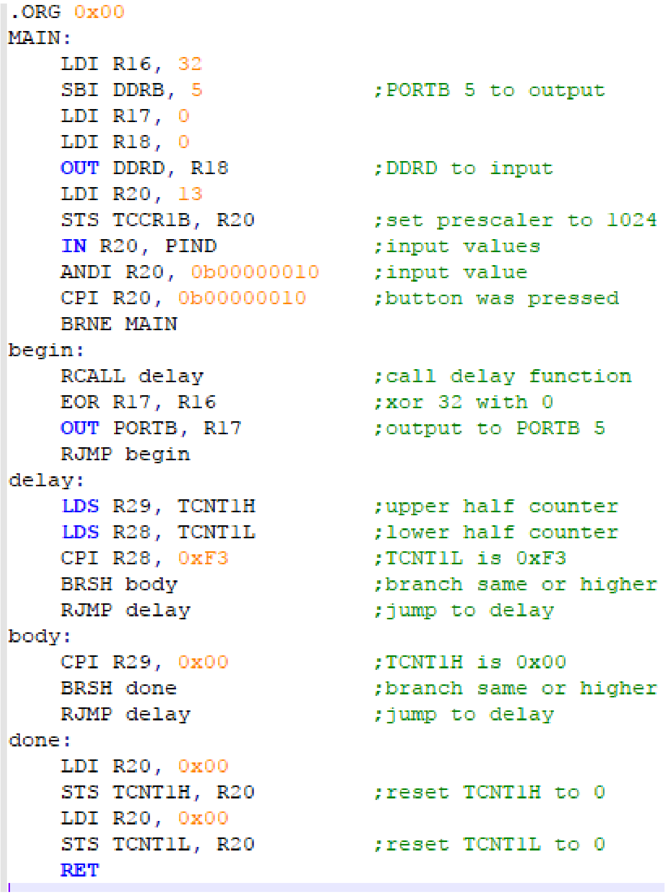


```
.org 0
    LDI R16, 32
    SBI DDRB, 5          ;output Port5
    LDI R17, 0
    OUT PORTB, R17       ;PortB to 0
    LDI R20, 5
    STS TCCR1B, R20      ;1024 prescaler
begin:
    RCALL delay          ;call delay function
    EOR R17, R16         ;xor 32 with 0
    OUT PORTB, R17       ;output the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

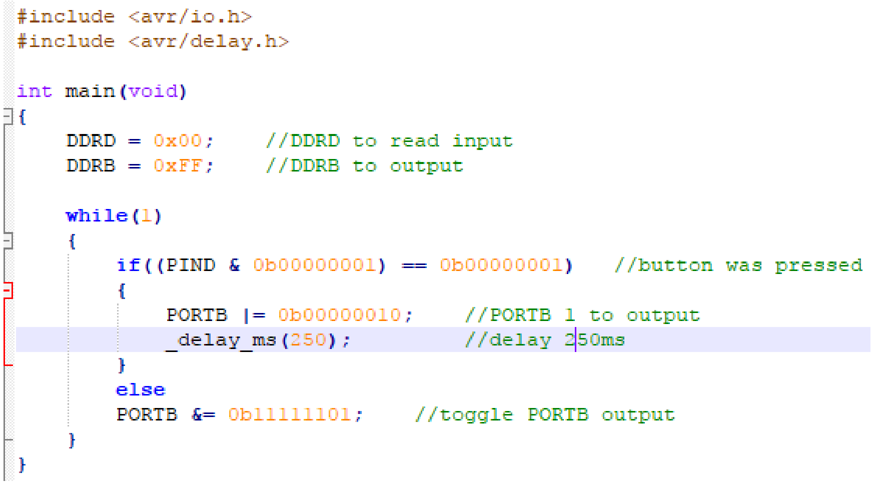


```c
#include <avr/io.h>
#include <avr/delay.h>

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

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

CODE TASK 2/A & 2/C

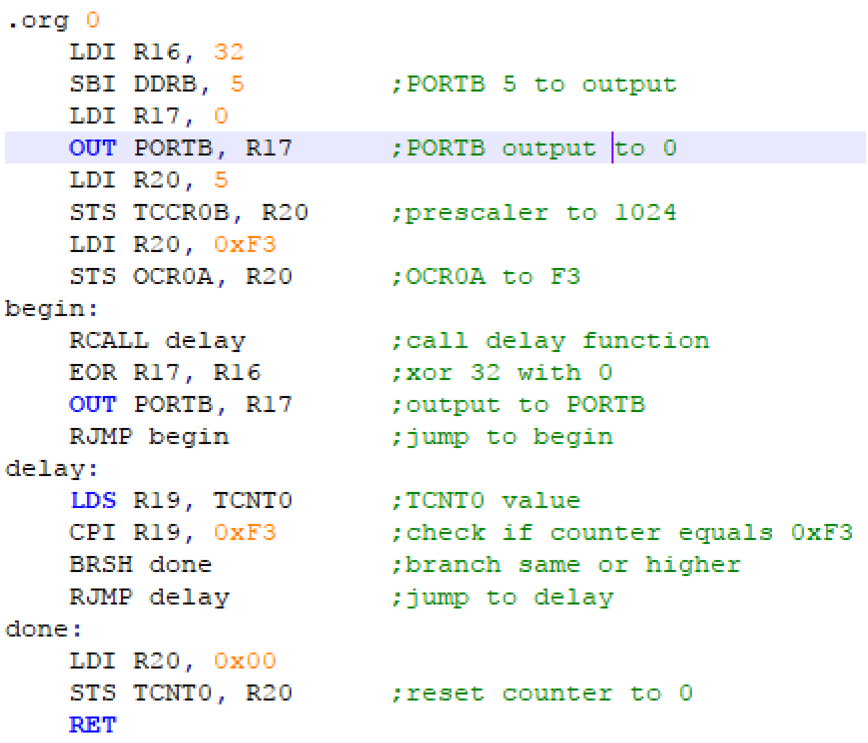


```
.ORG 0x00
MAIN:
    LDI R16, 32
    SBI DDRB, 5             ;PORTB 5 to output
    LDI R17, 0
    LDI R18, 0
    OUT DDRD, R18           ;DDRD to input
    LDI R20, 13
    STS TCCR1B, R20         ;set prescaler to 1024
    IN R20, PIND            ;input values
    ANDI R20, 0b00000010    ;input value
    CPI R20, 0b00000010     ;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         ;upper half counter
    LDS R28, TCNT1L         ;lower half counter
    CPI R28, 0xF3           ;TCNT1L is 0xF3
    BRSH body               ;branch same or higher
    RJMP delay              ;jump to delay
body:
    CPI R29, 0x00           ;TCNT1H is 0x00
    BRSH done               ;branch 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
```

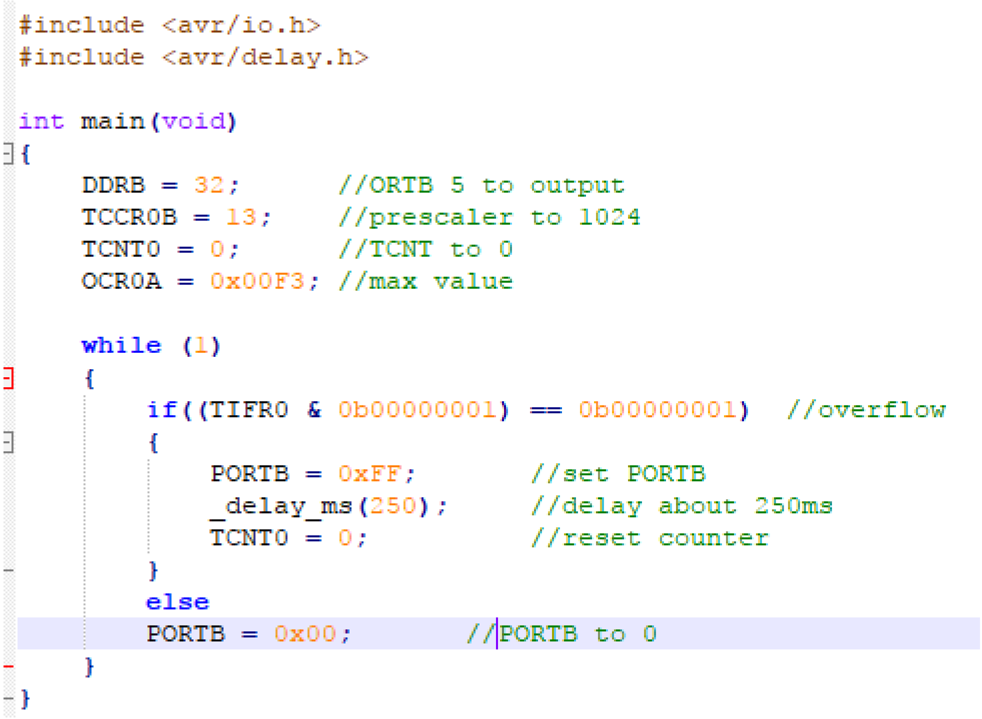


```
#include <avr/io.h>
#include <avr/delay.h>

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

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

CODE TASK 3/A & 3/C

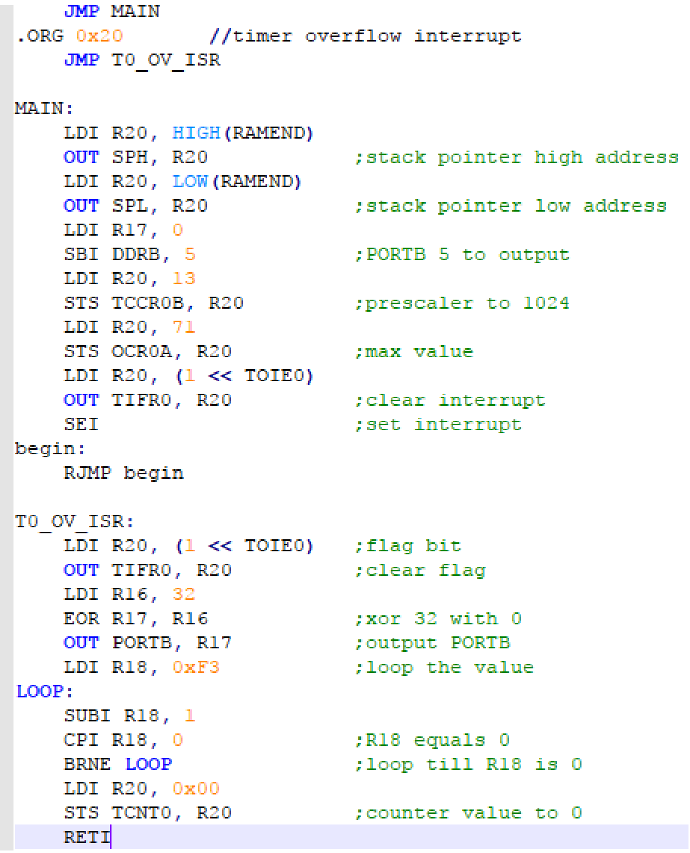


```
.org 0
    LDI R16, 32
    SBI DDRB, 5         ;PORTB 5 to output
    LDI R17, 0
    OUT PORTB, R17      ;PORTB output to 0
    LDI R20, 5
    STS TCCR0B, R20     ;prescaler to 1024
    LDI R20, 0xF3
    STS OCR0A, R20      ;OCR0A to F3
begin:
    RCALL delay         ;call delay function
    EOR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

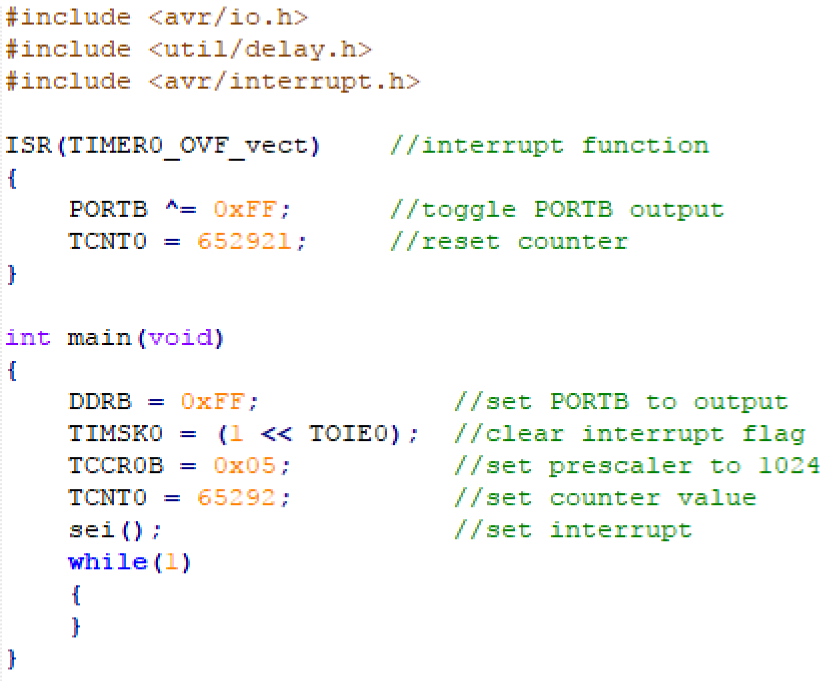


```
#include <avr/io.h>
#include <avr/delay.h>

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

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

CODE TASK 4/A & 4/C

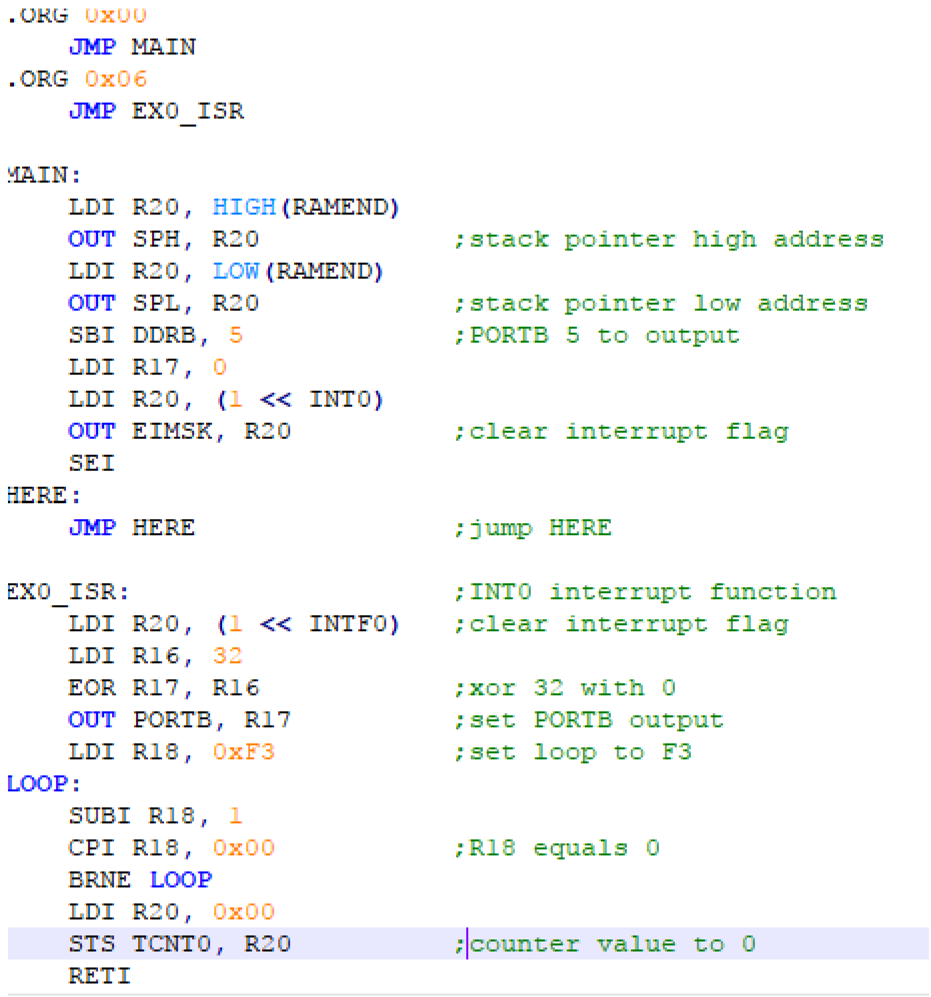


```
    JMP MAIN
.ORG 0x20       //timer overflow interrupt
    JMP T0_OV_ISR

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

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

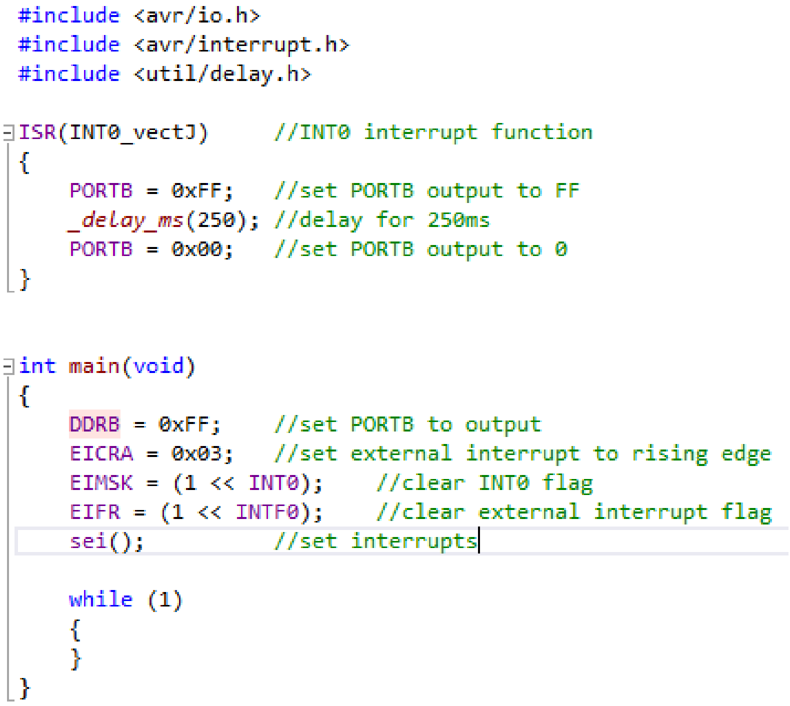


```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

ISR(TIMER0_OVF_vect)    //interrupt function
{
    PORTB ^= 0xFF;      //toggle PORTB output
    TCNT0 = 652921;     //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)
    {
    }
}
```

CODE TASK 5/A & 5/C



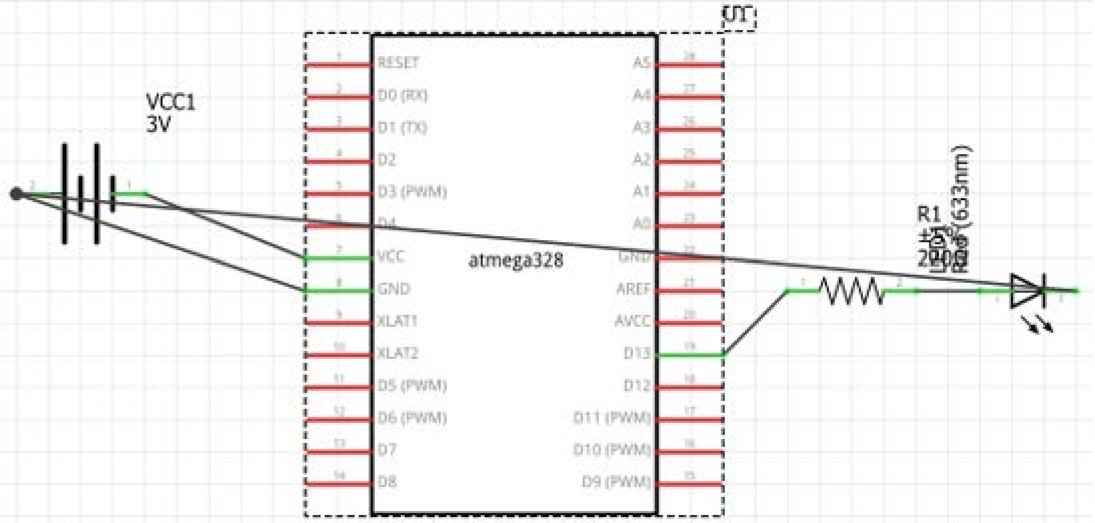
```
.ORG 0x00
    JMP MAIN
.ORG 0x06
    JMP EX0_ISR

MAIN:
    LDI R20, HIGH(RAMEND)
    OUT SPH, R20                ;stack pointer high address
    LDI R20, LOW(RAMEND)
    OUT SPL, R20                ;stack pointer low address
    SBI DDRB, 5                 ;PORTB 5 to output
    LDI R17, 0
    LDI R20, (1 << INT0)
    OUT EIMSK, R20              ;clear interrupt flag
    SEI
HERE:
    JMP HERE                    ;jump 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               ;R18 equals 0
    BRNE LOOP
    LDI R20, 0x00
    STS TCNT0, R20              ;counter value to 0
    RETI
```



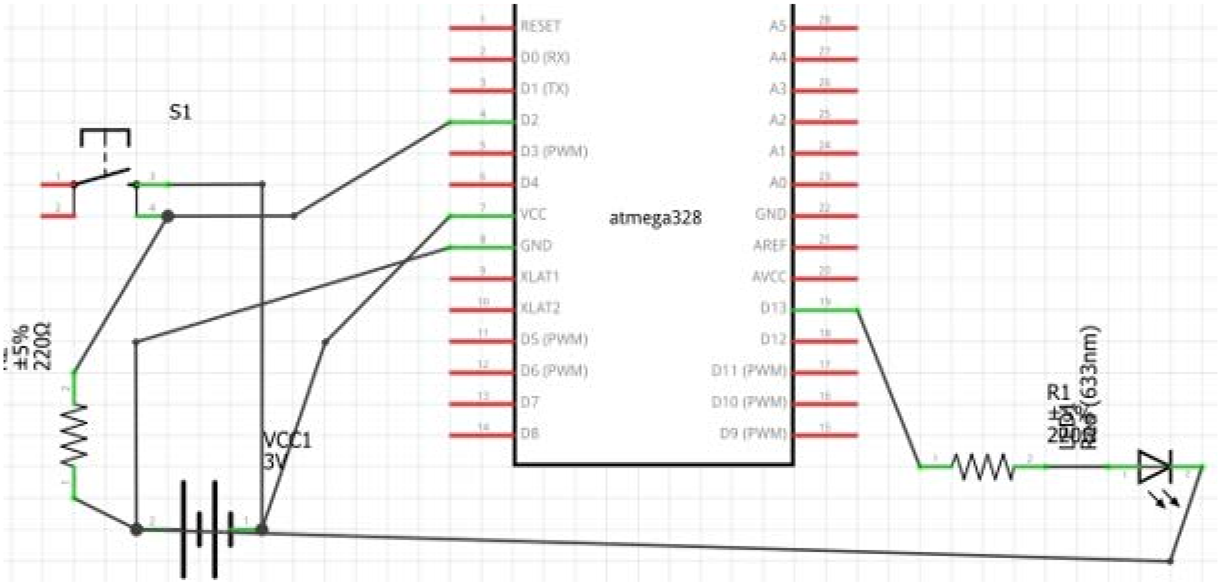
```
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

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

1, 3, & 4 schematic:

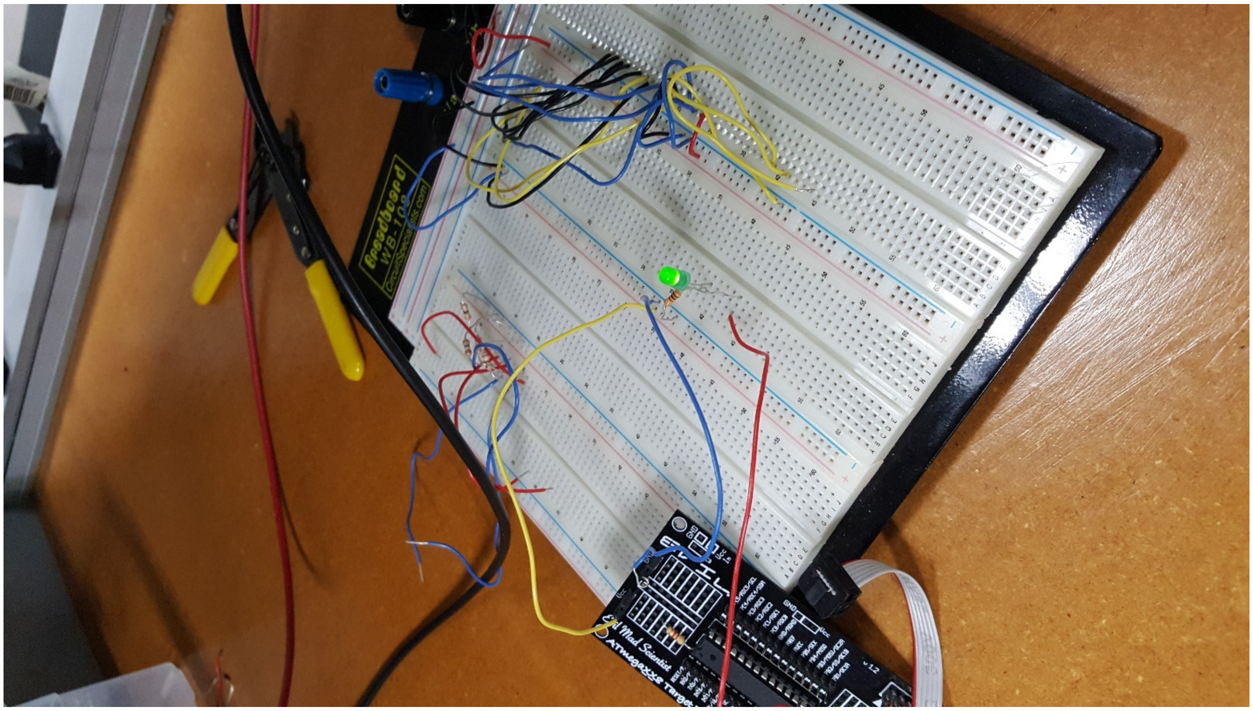


2 & 5 schematic:

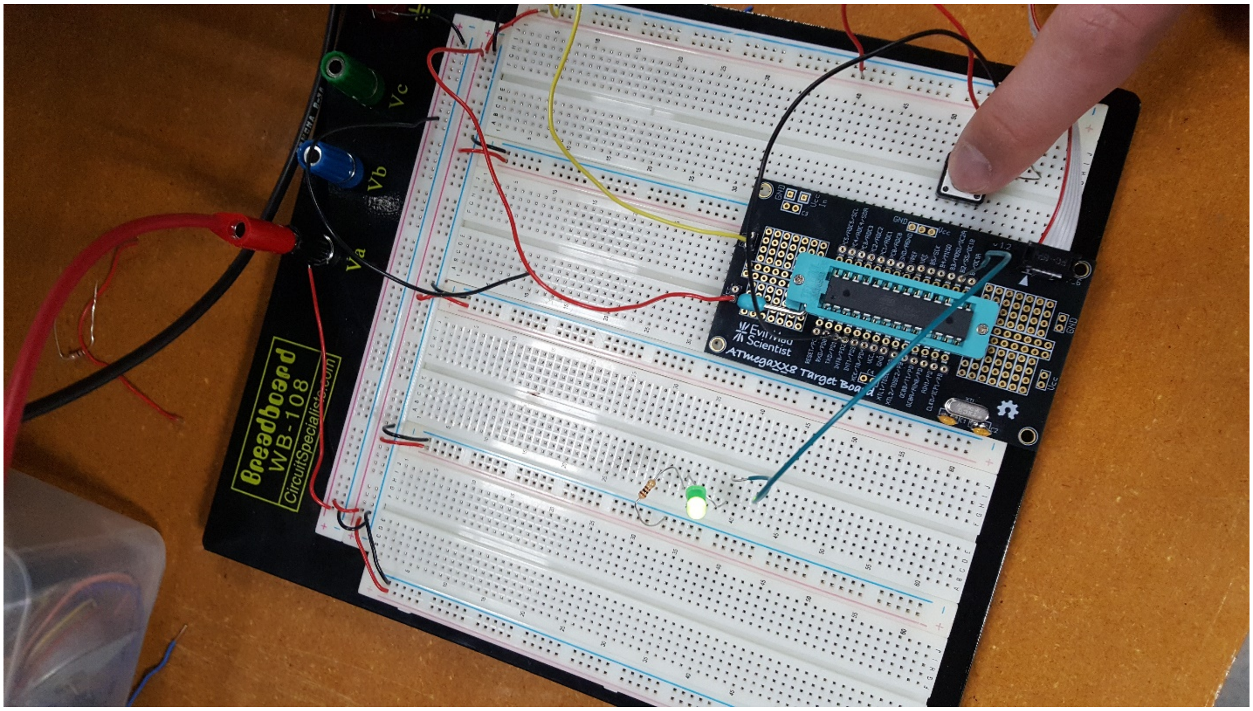


4. PICTURES OF CIRCUIT

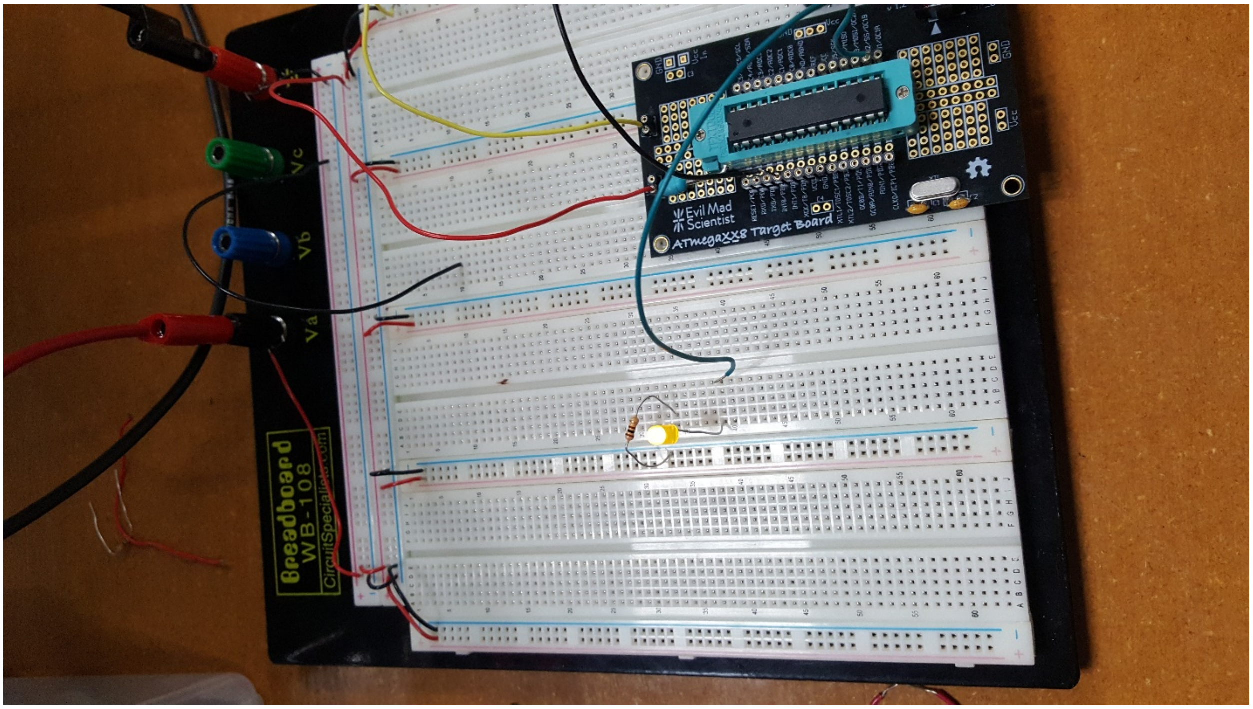
TASK 1:



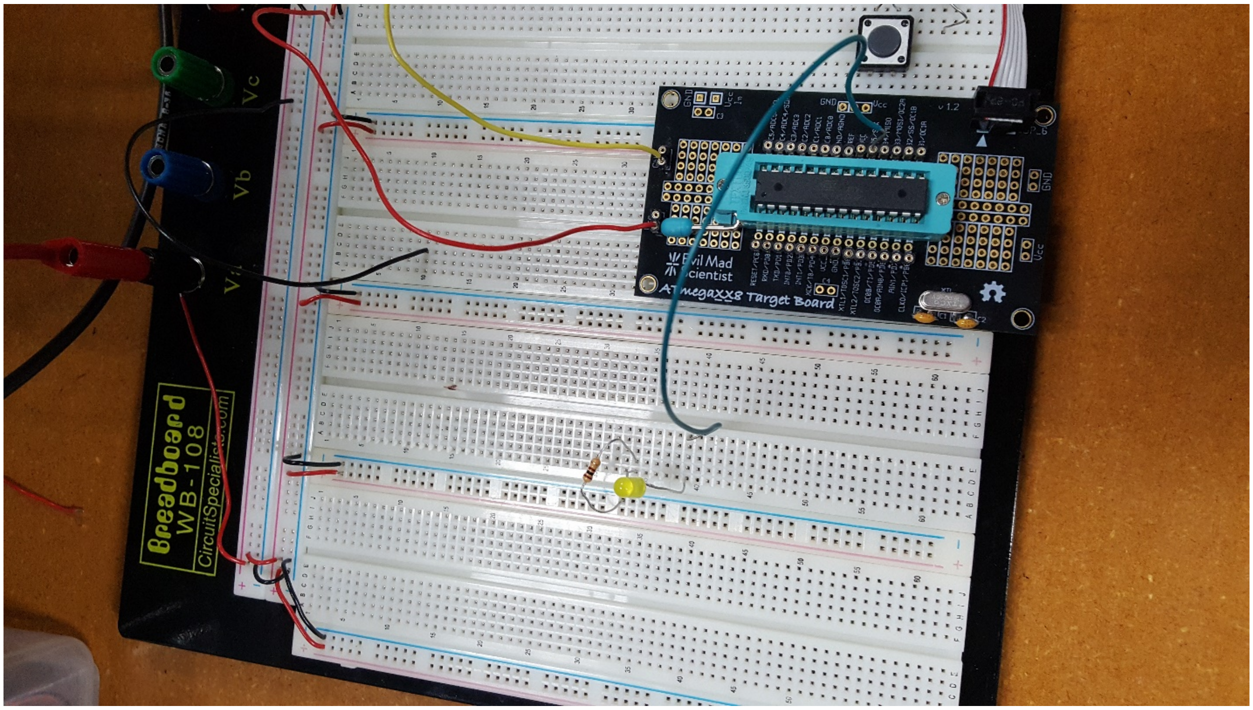
TASK 2:


Breadboard
WB-108
CircuitSpecialists.com
Vc
Vb
Va
Evil Mad Scientist
ATmegaXX8 Target Board

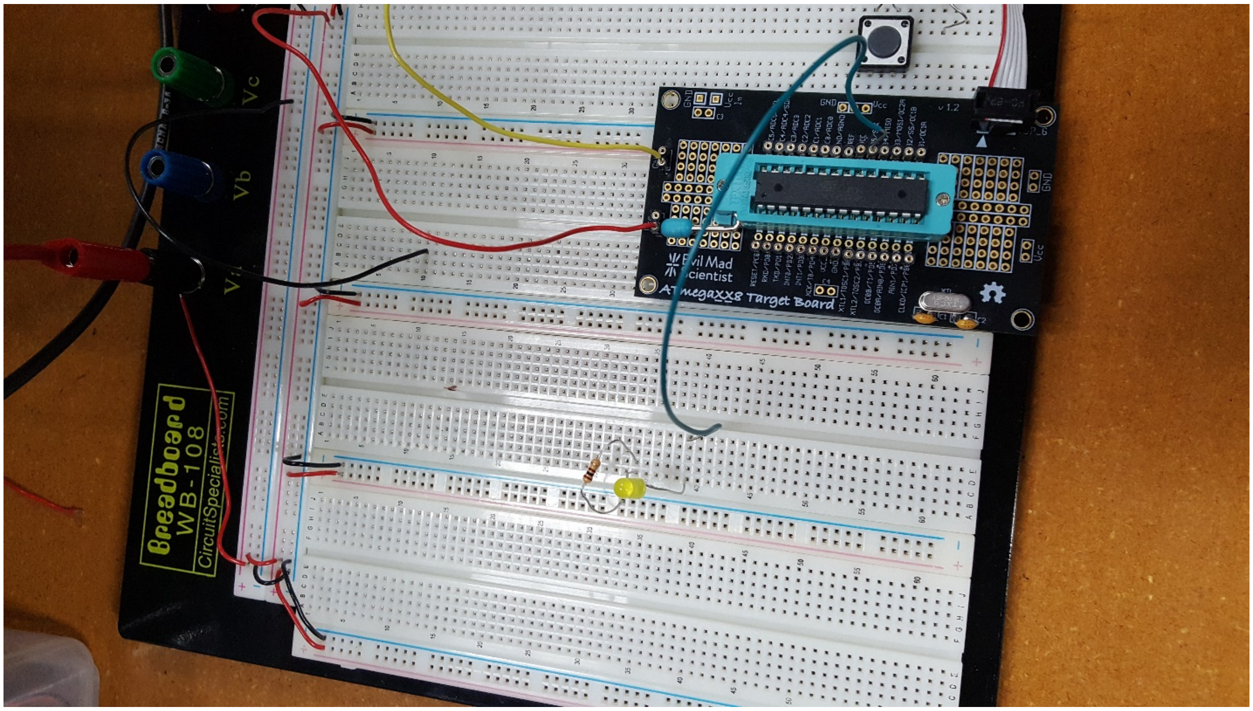
TASK 3:


Breadboard
WB-108
CircuitSpecialists.com
Vc
Vb
Va
Evil Mad Scientist
ATmegaXX8 Target Board

TASK 4:


Vc
Vb
Breadboard
WB-108
CircuitSpecialists.com
Evil Mad Scientist
ATmegaXX8 Target Board

TASK 5:


Vc
Vb
Breadboard
WB-108
CircuitSpecialists.com
Evil Mad Scientist
ATmegaXX8 Target Board

5. VIDEO LINKS OF EACH DEMO

https://youtu.be/D5utjyF5JT0

6. GITHUB LINK OF THIS DA

https://github.com/rsawa91