Brian Kiaer

CPE301 – SPRING 2018

Design Assignment 2

DO NOT REMOVE THIS PAGE DURING SUBMISSION:

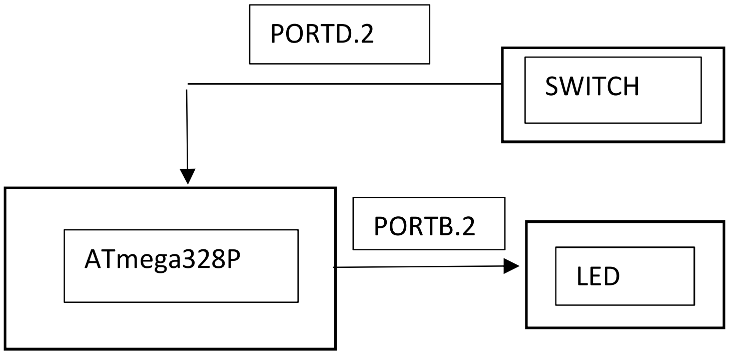
The student understands that all required components should be submitted in complete for grading of this assignment.

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

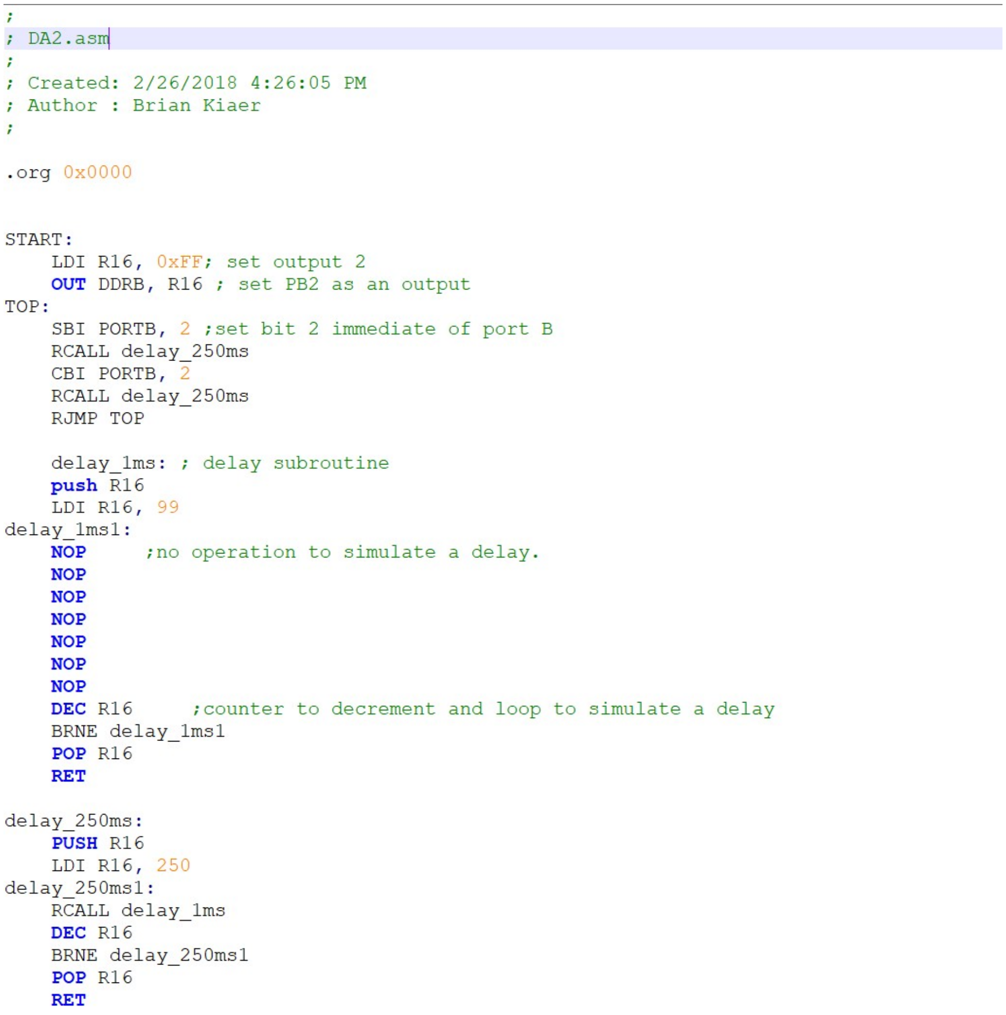
Components:

- Breadboard
- ATmega328P Microcontroller Board
- Yellow LED
- Push Button/Switch



List of Components used
Block diagram with pins used in the Atmega328P

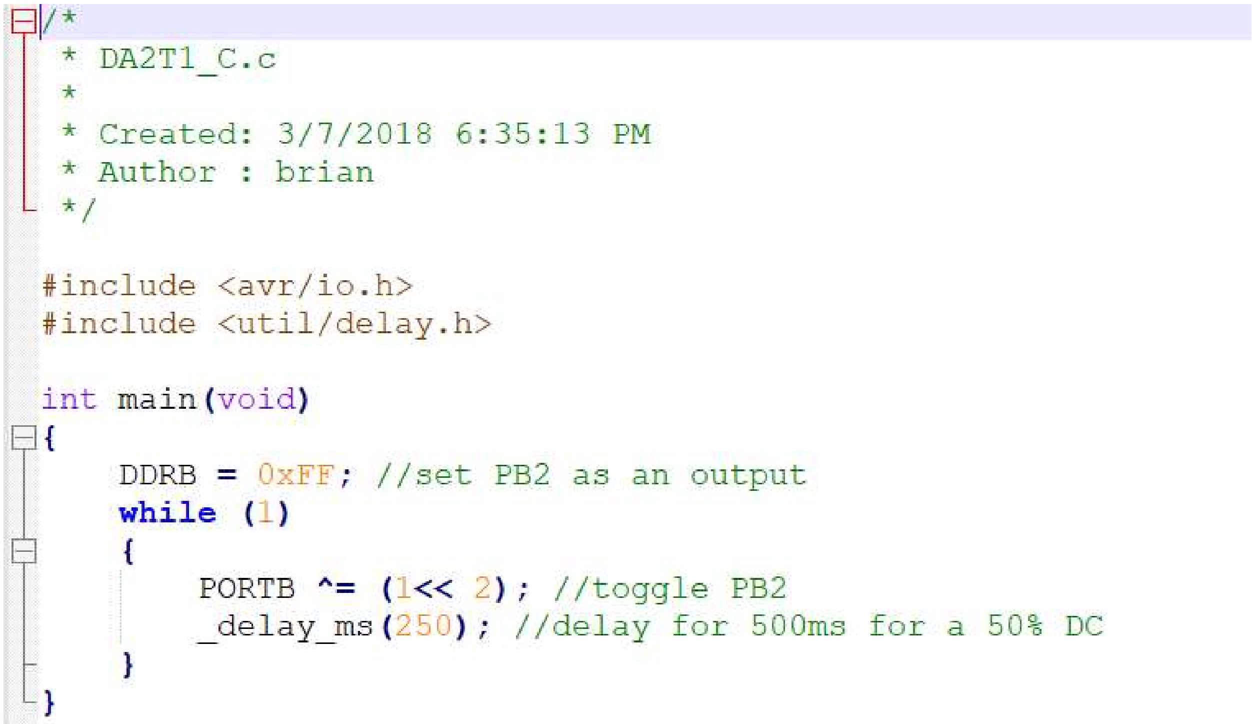
2. DEVELOPED CODE OF TASK 1/A IN AVR



```
;
; DA2.asm
;
; Created: 2/26/2018 4:26:05 PM
; Author : Brian Kiaer
;

.org 0x0000


START:
    LDI R16, 0xFF; set output 2
    OUT DDRB, R16 ; set PB2 as an output
TOP:
    SBI PORTB, 2 ;set bit 2 immediate of port B
    RCALL delay_250ms
    CBI PORTB, 2
    RCALL delay_250ms
    RJMP TOP

    delay_1ms: ; delay subroutine
    push R16
    LDI R16, 99
delay_1ms1:
    NOP     ;no operation to simulate a delay.
    NOP
    NOP
    NOP
    NOP
    NOP
    NOP
    DEC R16     ;counter to decrement and loop to simulate a delay
    BRNE delay_1ms1
    POP R16
    RET

delay_250ms:
    PUSH R16
    LDI R16, 250
delay_250ms1:
    RCALL delay_1ms
    DEC R16
    BRNE delay_250ms1
    POP R16
    RET
```

Above is Task 1 written in AVR Assembly. The purpose of this developed code is to create a delay subroutine. By using the delay subroutine, the code is responsible for creating a Waveform on Port B 2 with a 50% Duty Cycle.

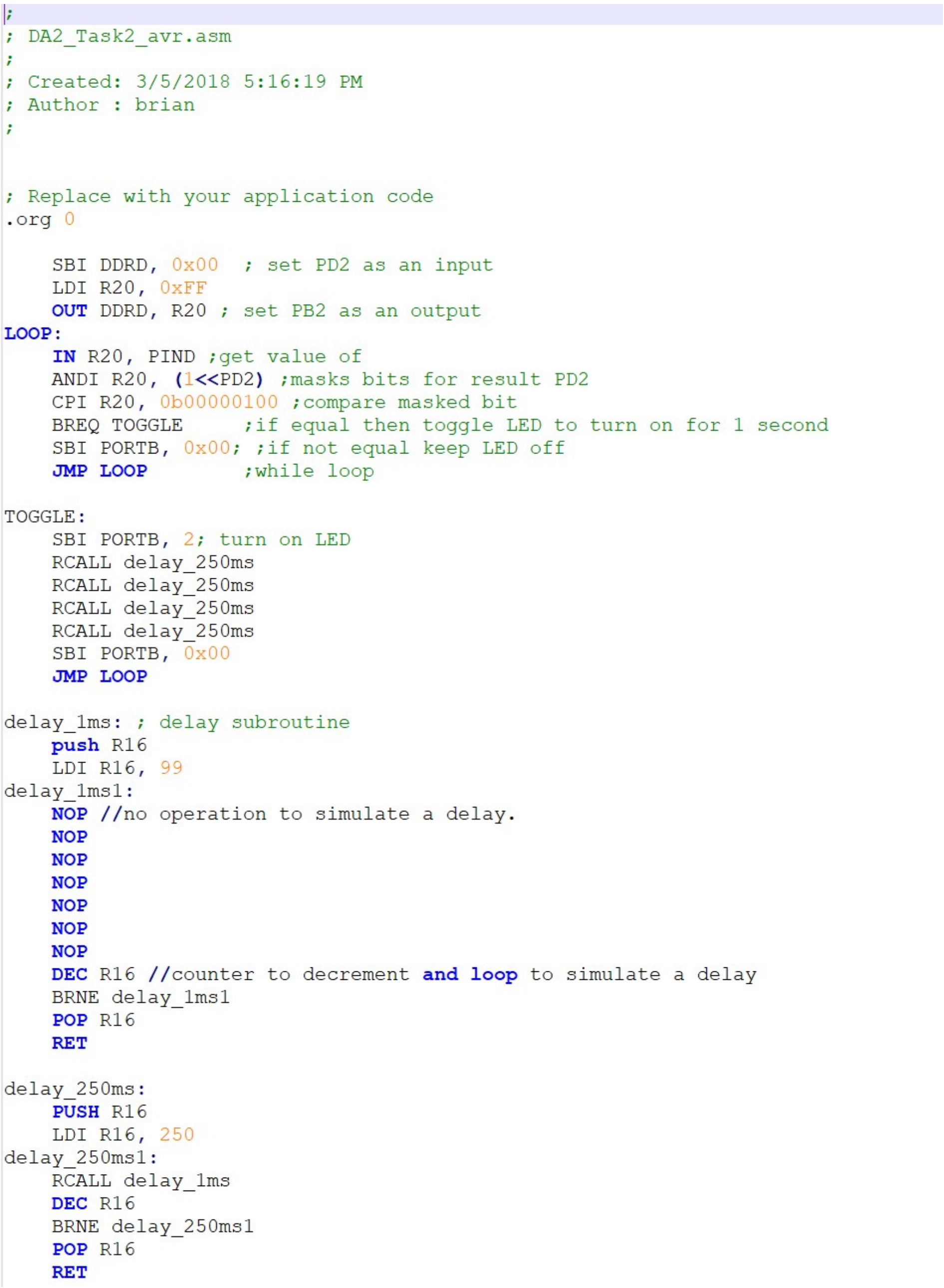
3. DEVELOPED CODE OF TASK 1/B IN C



```
/*
 * DA2T1_C.c
 *
 * Created: 3/7/2018 6:35:13 PM
 * Author : brian
 */

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

int main(void)
{
    DDRB = 0xFF; //set PB2 as an output
    while (1)
    {
        PORTB ^= (1<< 2); //toggle PB2
        _delay_ms(250); //delay for 500ms for a 50% DC
    }
}
```

Above is the C code implementing Task 1. This is responsible for generating a waveform with a 50% Duty Cycle and with a period of 0.5 seconds. The waveform will toggle an LED that is connected to Port B Pin 2 (PB2).

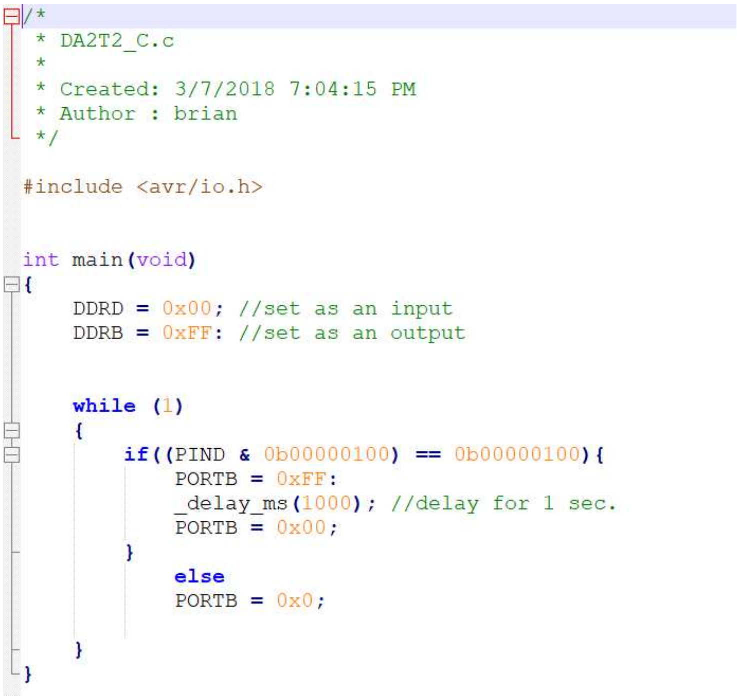
4. DEVELOPED CODE OF TASK 2/A IN AVR



```
;
; DA2_Task2_avr.asm
;
; Created: 3/5/2018 5:16:19 PM
; Author : brian
;


; Replace with your application code
.org 0

    SBI DDRD, 0x00  ; set PD2 as an input
    LDI R20, 0xFF
    OUT DDRD, R20 ; set PB2 as an output
LOOP:
    IN R20, PIND ;get value of
    ANDI R20, (1<<PD2) ;masks bits for result PD2
    CPI R20, 0b00000100 ;compare masked bit
    BREQ TOGGLE     ;if equal then toggle LED to turn on for 1 second
    SBI PORTB, 0x00; ;if not equal keep LED off
    JMP LOOP        ;while loop

TOGGLE:
    SBI PORTB, 2; turn on LED
    RCALL delay_250ms
    RCALL delay_250ms
    RCALL delay_250ms
    RCALL delay_250ms
    SBI PORTB, 0x00
    JMP LOOP

delay_1ms: ; delay subroutine
    push R16
    LDI R16, 99
delay_1ms1:
    NOP //no operation to simulate a delay.
    NOP
    NOP
    NOP
    NOP
    NOP
    NOP
    DEC R16 //counter to decrement and loop to simulate a delay
    BRNE delay_1ms1
    POP R16
    RET

delay_250ms:
    PUSH R16
    LDI R16, 250
delay_250ms1:
    RCALL delay_1ms
    DEC R16
    BRNE delay_250ms1
    POP R16
    RET
```

This code is to implement Task 2 written in AVR Assembly. This code is responsible for connecting a switch to Port D.2 as an active-high pull-up activated transistor, when it is pressed or switched on an LED at Port B.2 will turn light up for a delay of 1 second and then turn back off. This is without using interrupts.

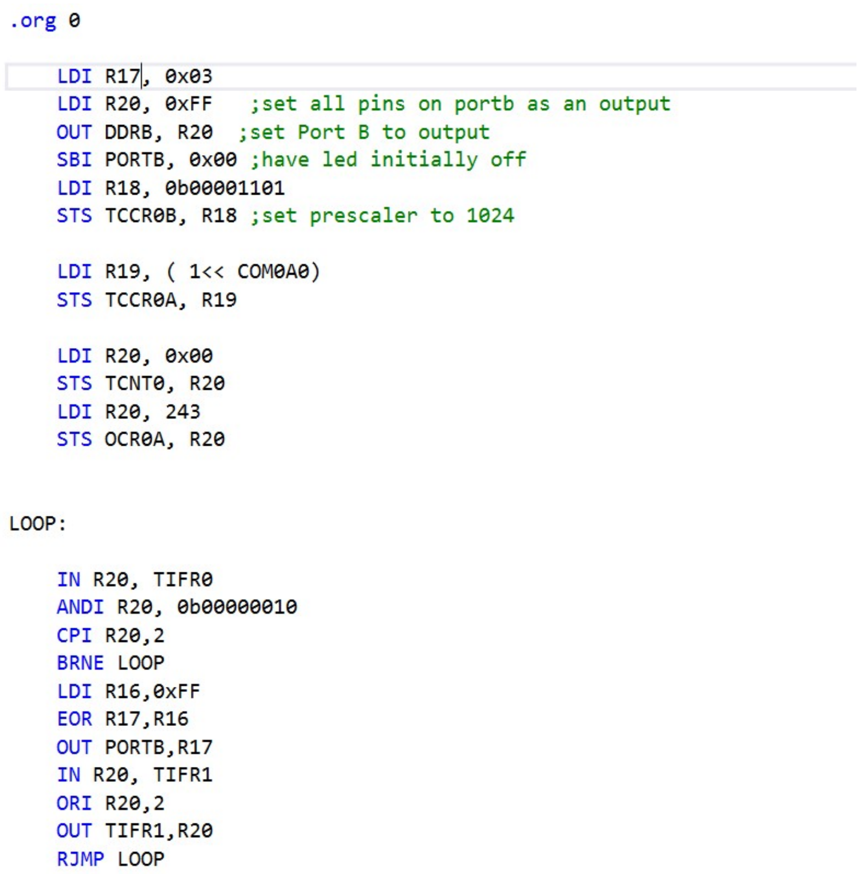
5. DEVELOPED CODE FOR TASK 2/B IN C



```c
/*
 * DA2T2_C.c
 *
 * Created: 3/7/2018 7:04:15 PM
 * Author : brian
 */

#include <avr/io.h>


int main(void)
{
    DDRD = 0x00; //set as an input
    DDRB = 0xFF: //set as an output


    while (1)
    {
        if((PIND & 0b00000100) == 0b00000100){
            PORTB = 0xFF:
            _delay_ms(1000); //delay for 1 sec.
            PORTB = 0x00;
        }
            else
            PORTB = 0x0;

    }
}
```

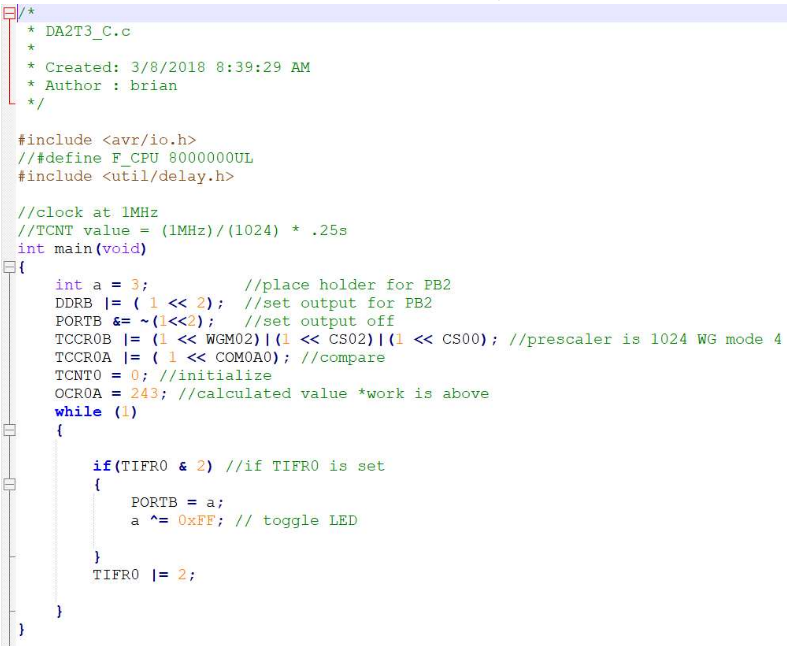
Above is the code for Task 2/B written in C. This code has an active-high pull-up transistor connected to PORTD.2 and when it is pressed, an LED will light up that is connected to PORTB.2 for 1 second.

6. DEVELOPED CODE FOR TASK 3/A IN AVR



```
.org 0

    LDI R17, 0x03
    LDI R20, 0xFF   ;set all pins on portb as an output
    OUT DDRB, R20  ;set Port B to output
    SBI PORTB, 0x00 ;have led initially off
    LDI R18, 0b00001101
    STS TCCR0B, R18 ;set prescaler to 1024

    LDI R19, ( 1<< COM0A0)
    STS TCCR0A, R19

    LDI R20, 0x00
    STS TCNT0, R20
    LDI R20, 243
    STS OCR0A, R20


LOOP:

    IN R20, TIFR0
    ANDI R20, 0b00000010
    CPI R20,2
    BRNE LOOP
    LDI R16,0xFF
    EOR R17,R16
    OUT PORTB,R17
    IN R20, TIFR1
    ORI R20,2
    OUT TIFR1,R20
    RJMP LOOP
```

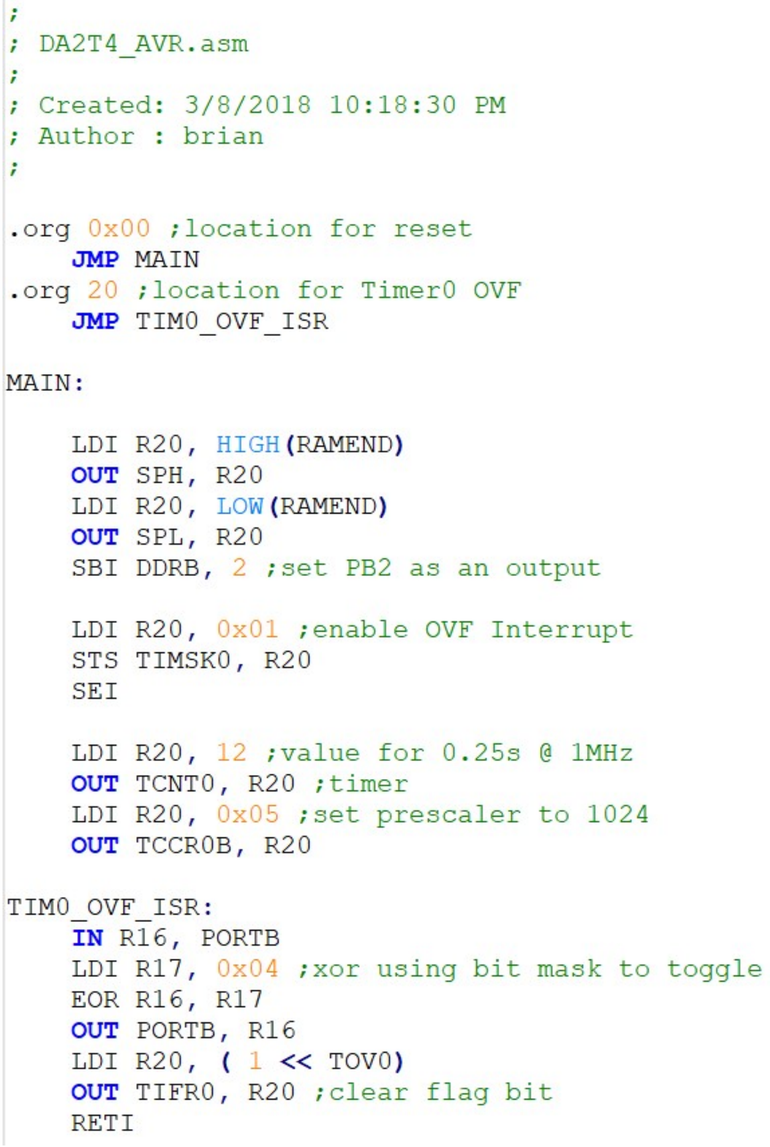
7. DEVELOPED CODE FOR TASK 3/B IN C



```
/*
 * DA2T3_C.c
 *
 * Created: 3/8/2018 8:39:29 AM
 * Author : brian
 */

#include <avr/io.h>
//#define F_CPU 8000000UL
#include <util/delay.h>

//clock at 1MHz
//TCNT value = (1MHz)/(1024) * .25s
int main(void)
{
    int a = 3;           //place holder for PB2
    DDRB |= ( 1 << 2);   //set output for PB2
    PORTB &= ~(1<<2);    //set output off
    TCCR0B |= (1 << WGM02)|(1 << CS02)|(1 << CS00); //prescaler is 1024 WG mode 4
    TCCR0A |= ( 1 << COM0A0); //compare
    TCNT0 = 0; //initialize
    OCR0A = 243; //calculated value *work is above
    while (1)
    {

        if(TIFR0 & 2) //if TIFR0 is set
        {
            PORTB = a;
            a ^= 0xFF; // toggle LED

        }
        TIFR0 |= 2;

    }
}
```

Above is the C Code written for Task 3/B. This is responsible for implementing Task 1 using Timer0. The code is responsible for implementing CTC mode. By comparing Timer0 with OCR0A.

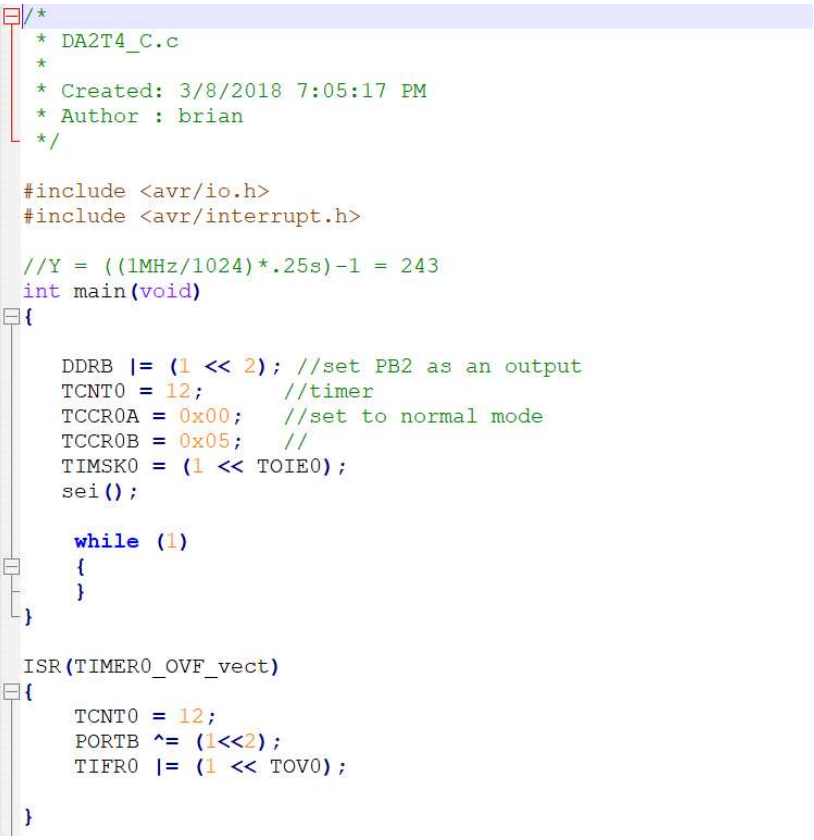
8. DEVELOPED CODE FOR TASK 4/A in AVR



```
;
; DA2T4_AVR.asm
;
; Created: 3/8/2018 10:18:30 PM
; Author : brian
;

.org 0x00 ;location for reset
    JMP MAIN
.org 20 ;location for Timer0 OVF
    JMP TIM0_OVF_ISR

MAIN:

    LDI R20, HIGH(RAMEND)
    OUT SPH, R20
    LDI R20, LOW(RAMEND)
    OUT SPL, R20
    SBI DDRB, 2 ;set PB2 as an output

    LDI R20, 0x01 ;enable OVF Interrupt
    STS TIMSK0, R20
    SEI

    LDI R20, 12 ;value for 0.25s @ 1MHz
    OUT TCNT0, R20 ;timer
    LDI R20, 0x05 ;set prescaler to 1024
    OUT TCCR0B, R20

TIM0_OVF_ISR:
    IN R16, PORTB
    LDI R17, 0x04 ;xor using bit mask to toggle
    EOR R16, R17
    OUT PORTB, R16
    LDI R20, ( 1 << TOV0)
    OUT TIFR0, R20 ;clear flag bit
    RETI
```

Above is the code written in AVR Assembly, this code is responsible for implementing Task 1 but with the Timer0 Overflow Interrupt.

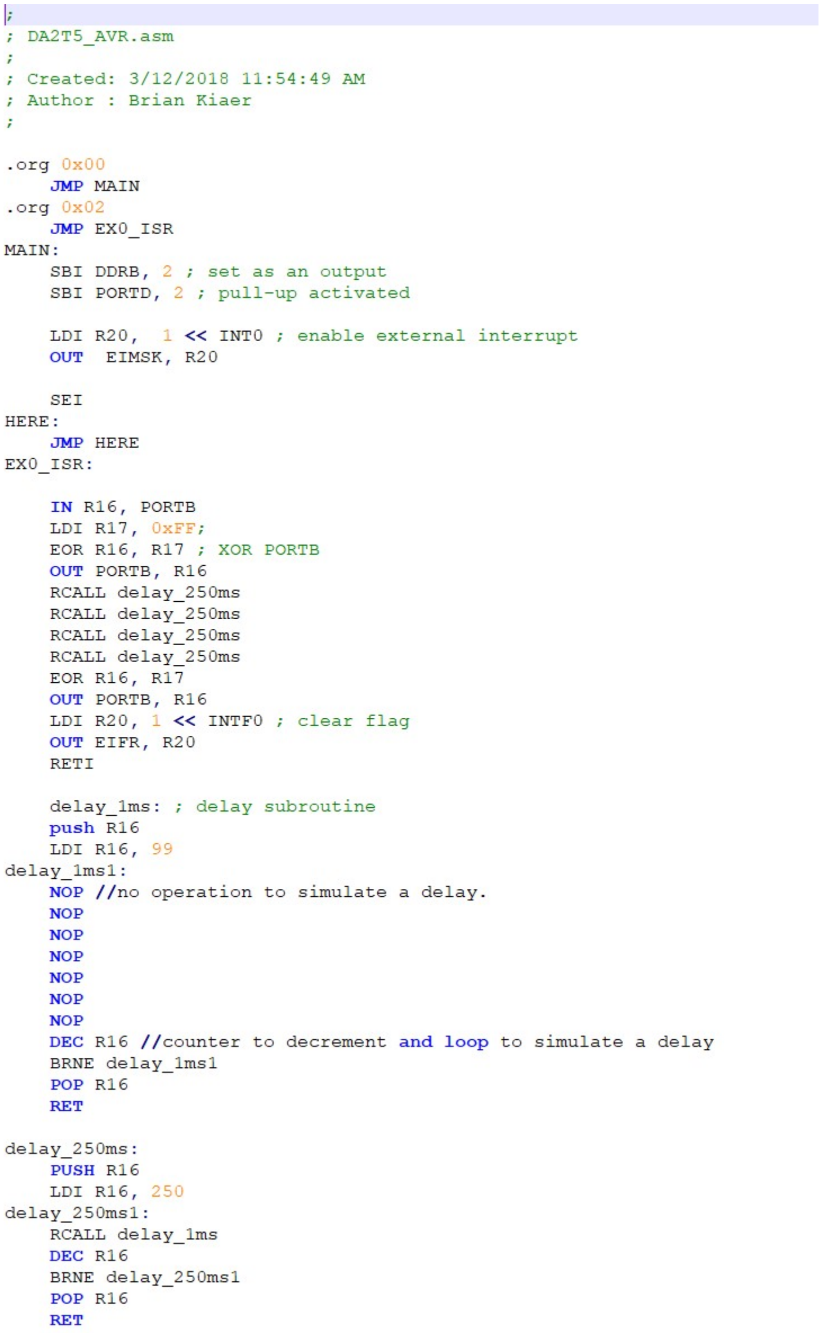
9. DEVELOPED CODE FOR TASK 4/B IN C



```
/*
 * DA2T4_C.c
 *
 * Created: 3/8/2018 7:05:17 PM
 * Author : brian
 */

#include <avr/io.h>
#include <avr/interrupt.h>

//Y = ((1MHz/1024)*.25s)-1 = 243
int main(void)
{

   DDRB |= (1 << 2); //set PB2 as an output
   TCNT0 = 12;       //timer
   TCCR0A = 0x00;    //set to normal mode
   TCCR0B = 0x05;    //
   TIMSK0 = (1 << TOIE0);
   sei();

    while (1)
    {
    }
}

ISR(TIMER0_OVF_vect)
{
    TCNT0 = 12;
    PORTB ^= (1<<2);
    TIFR0 |= (1 << TOV0);

}
```

The above code has the same implementation as Task 4/A but is written in C. I calculated the Y value as 243 and subtracted 255 from that to have the TCNT0 timer start at 12.

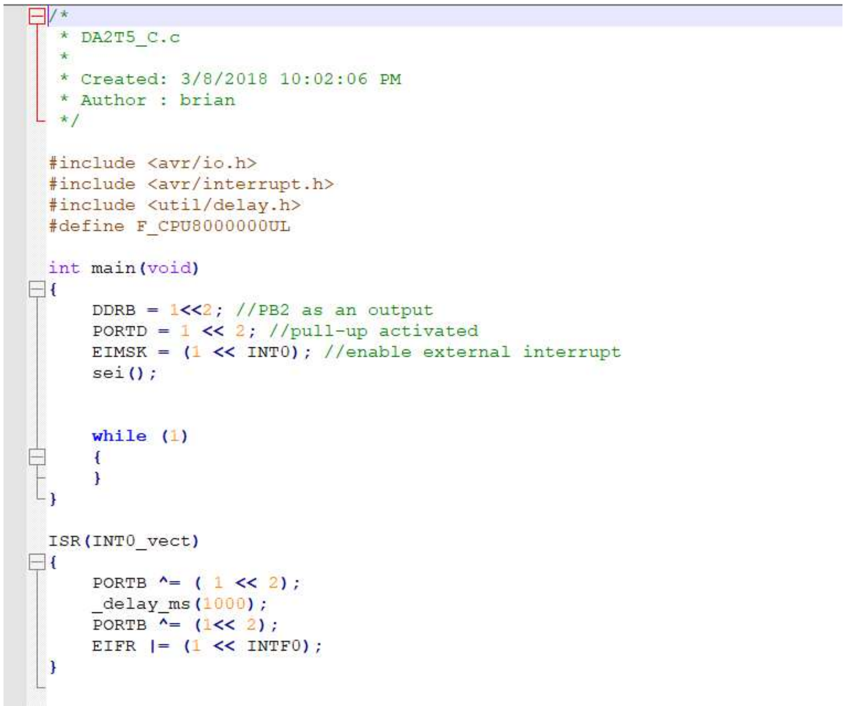
10. DEVELOPED CODE FOR TASK 5/A IN AVR



```
;
; DA2T5_AVR.asm
;
; Created: 3/12/2018 11:54:49 AM
; Author : Brian Kiaer
;

.org 0x00
    JMP MAIN
.org 0x02
    JMP EX0_ISR
MAIN:
    SBI DDRB, 2 ; set as an output
    SBI PORTD, 2 ; pull-up activated

    LDI R20,  1 << INT0 ; enable external interrupt
    OUT  EIMSK, R20

    SEI
HERE:
    JMP HERE
EX0_ISR:

    IN R16, PORTB
    LDI R17, 0xFF;
    EOR R16, R17 ; XOR PORTB
    OUT PORTB, R16
    RCALL delay_250ms
    RCALL delay_250ms
    RCALL delay_250ms
    RCALL delay_250ms
    EOR R16, R17
    OUT PORTB, R16
    LDI R20, 1 << INTF0 ; clear flag
    OUT EIFR, R20
    RETI

    delay_1ms: ; delay subroutine
    push R16
    LDI R16, 99
delay_1ms1:
    NOP //no operation to simulate a delay.
    NOP
    NOP
    NOP
    NOP
    NOP
    NOP
    DEC R16 //counter to decrement and loop to simulate a delay
    BRNE delay_1ms1
    POP R16
    RET

delay_250ms:
    PUSH R16
    LDI R16, 250
delay_250ms1:
    RCALL delay_1ms
    DEC R16
    BRNE delay_250ms1
    POP R16
    RET
```

The code above is written in AVR Assembly, Task 5 is responsible for implementing Task 2 but with an External Interrupt. The external interrupt is connected to PIN INT0 and when pressed the LED connected to PB2 will turn on for 1 second and then turn off.

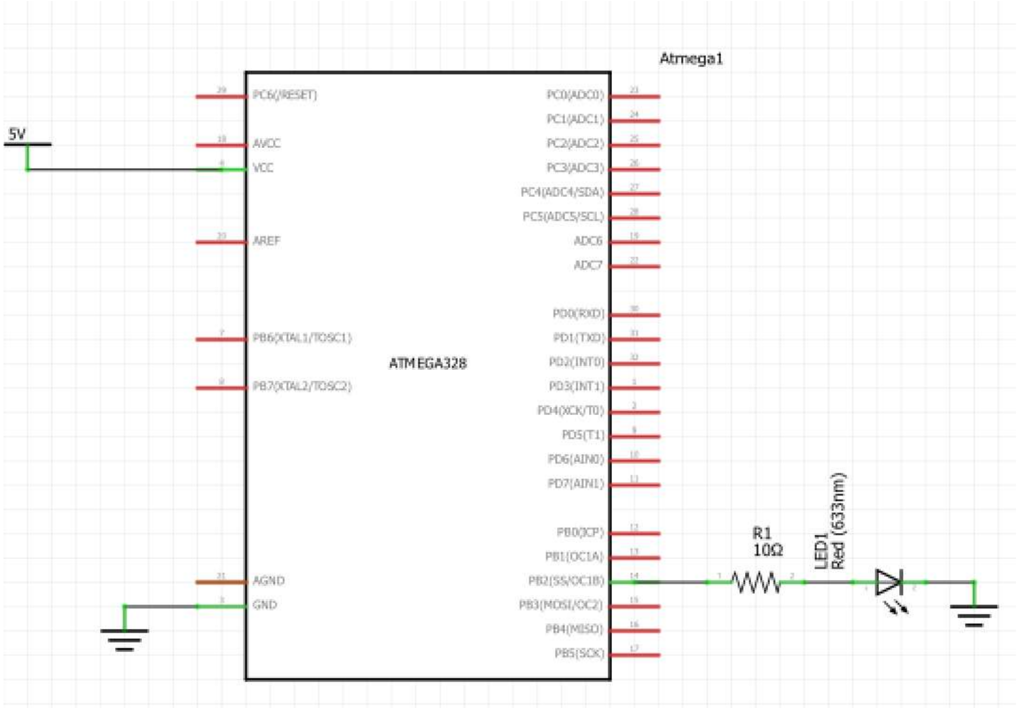
11. DEVELOPED CODE FOR TASK 5/B IN C



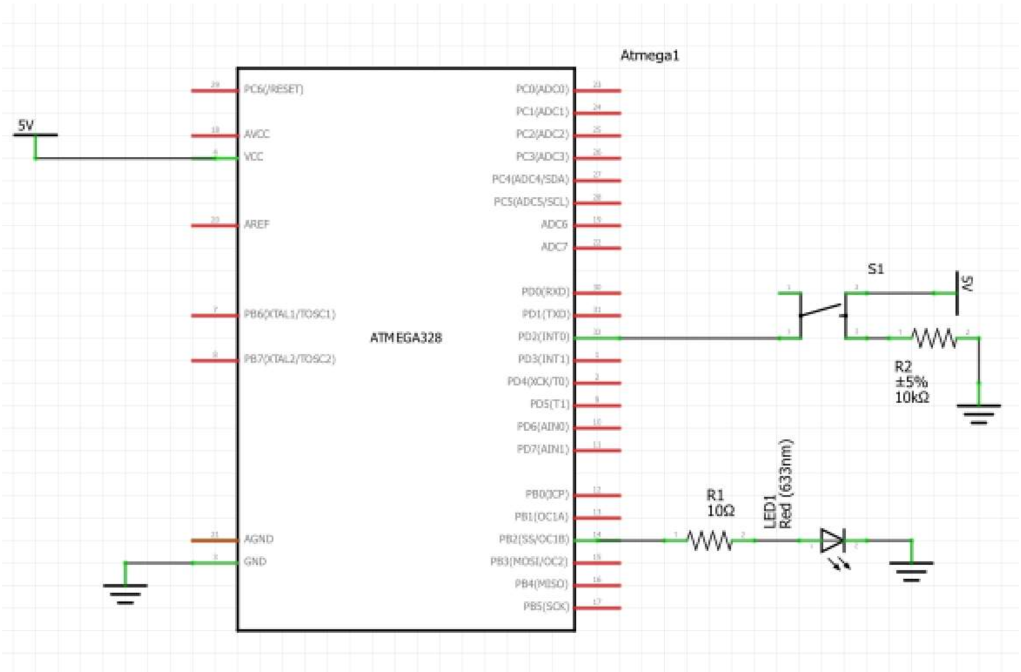
```
/*
 * DA2T5_C.c
 *
 * Created: 3/8/2018 10:02:06 PM
 * Author : brian
 */

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#define F_CPU8000000UL

int main(void)
{
    DDRB = 1<<2; //PB2 as an output
    PORTD = 1 << 2; //pull-up activated
    EIMSK = (1 << INT0); //enable external interrupt
    sei();


    while (1)
    {
    }
}

ISR(INT0_vect)
{
    PORTB ^= ( 1 << 2);
    _delay_ms(1000);
    PORTB ^= (1<< 2);
    EIFR |= (1 << INTF0);
}
```

Above is the code for Task 5 written in C. This is responsible for implementing an external interrupt and delaying an LED for 1 second when the button is pressed.

12. SCHEMATICS



This schematic above was used to implement Tasks 1, 3, and 4.



This schematic is used for Tasks 2 and 5.

13. SCREENSHOT OF EACH DEMO

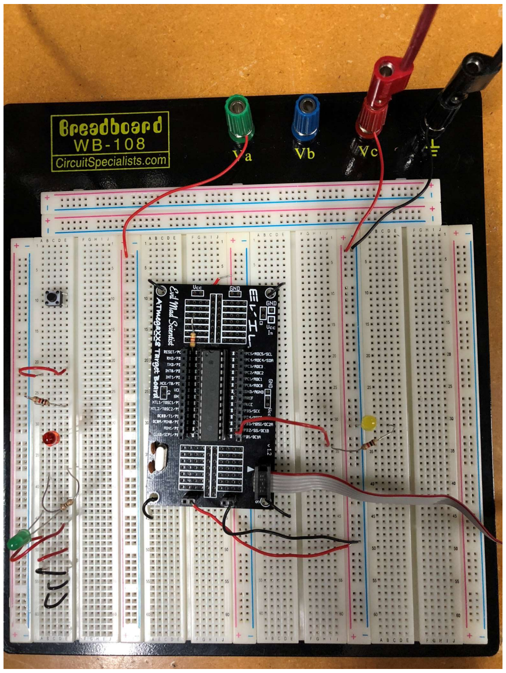


Figure 1. Schematic for Task 1 for AVR Assembly and C.

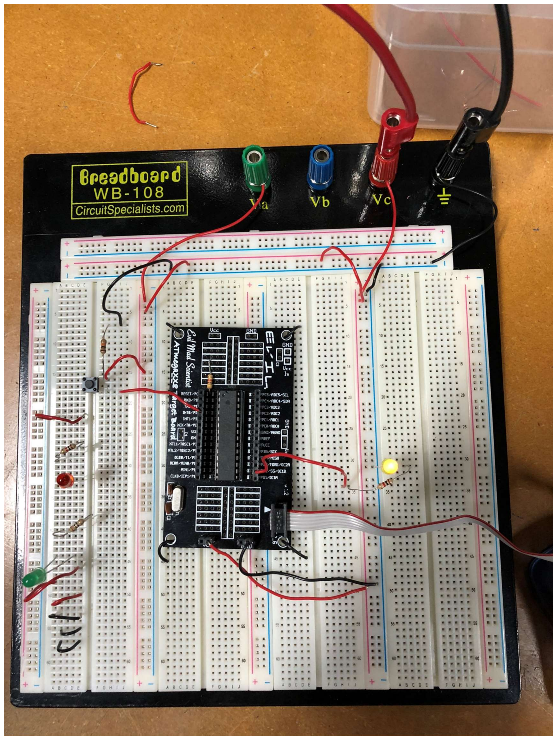


Figure 2. Schematic for Task 2 in AVR and C Code.

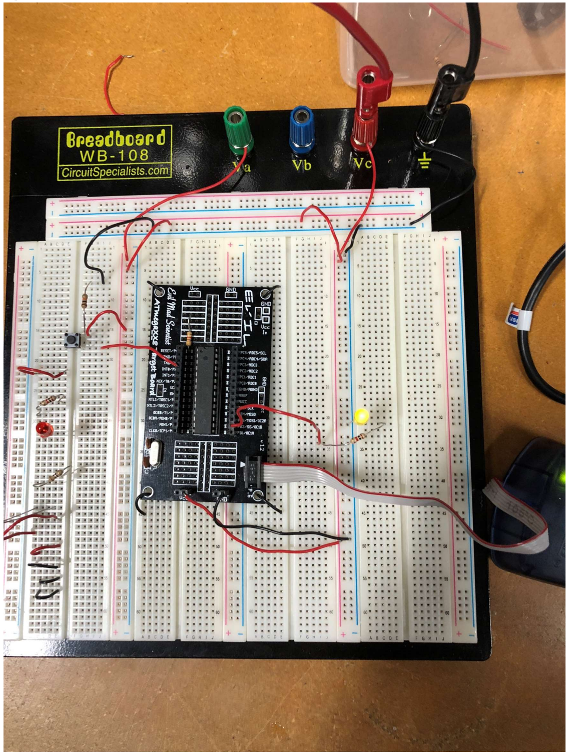


Figure 3. Schematic I used to test Task 3 in AVR and C Code.

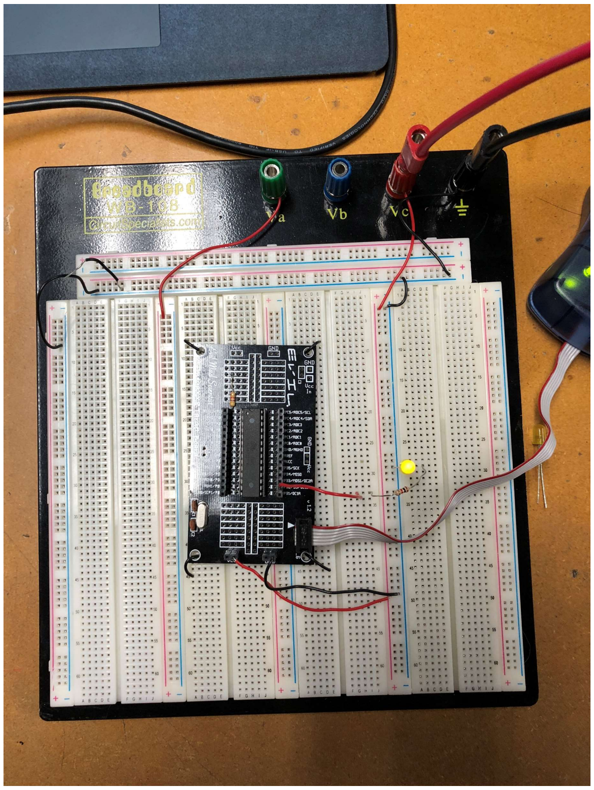


Figure 4. Schematic for Task 4 in AVR and C Code.

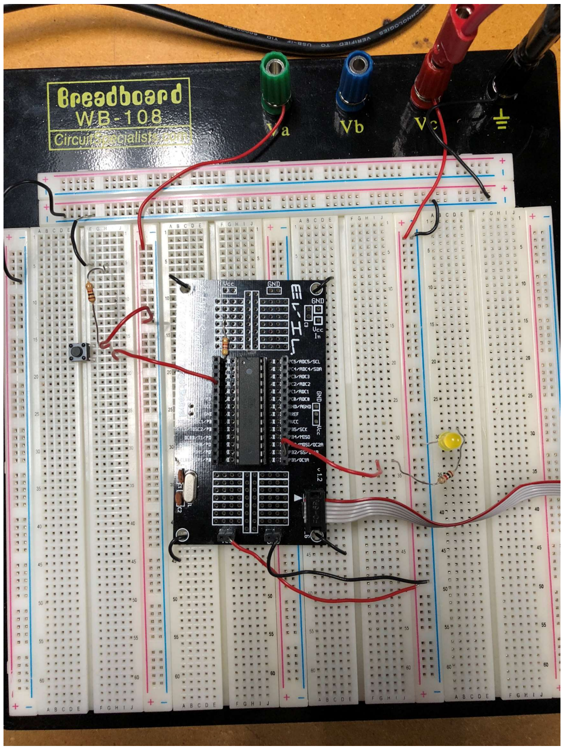


Figure 5. Schematic on Breadboard for Task 5 for AVR and C Code

14. **YOUTUBE VIDEO LINKS**

Task 1 - https://www.youtube.com/watch?v=JSzn9tOtVw8

Task 2 – https://www.youtube.com/watch?v=4ZI6pd2mjAw

Task 3 – https://www.youtube.com/watch?v=4bw8KsGgASE

Task 4 – https://www.youtube.com/watch?v=Ndr4dWiZFy8

Task 5 – https://www.youtube.com/watch?v=PHANe1GfLO4

15. DA LINK

https://github.com/bkiaer/DA2

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

"This assignment submission is my own, original work".
Brian Kiaer