Phillip Sortomme

CPE301 – SPRING 2018

Design Assignment X

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

ATmega328P
LED
push button

2. CODE OF TASK 1

```
#include <avr/io.h>
//for the counter we want (1MHz/1024 *0.25)-1 = 243
int main()
{
	DDRB = 0x04;                                    // set PB2 to output
	PORTB = 0x00;          // clear all of PORTB
	TCCR1B = 5;            // set prescaler to 1024
	while(1) {
		if(TCNT1 == 0x00F3){  // when counter = 243
			PORTB ^= 0x04;      // toggle the LED
			TCNT1 = 0x00;       // reset the counter
		}
	}
}
.org 0
   ldi R16,low(RAMEND)
   out SPL,R16
   ldi R16,high(RAMEND)
   out SPH,R16
   SBI DDRB, 2       ; set PB2 as output
   LDI R16, 0
   OUT PORTB, R16    ; set all of PORTB to zero. start LED off
   LDI R17, 5
   STS TCCR1B, R17   ; set timer 1 prescaler to 1024
   LDI R17, 4        ; used to XOR with R16 to toggle LED

begin:
   RCALL delay
   EOR R16, R17
   OUT PORTB, R16
   RJMP begin

delay:
	LDS R18, TCNT1L
   CPI R18, 0xF3
   BRSH checkupper    ; check to see if TCNT1 = 0xF3
   RJMP delay

done:
   LDI R18, 0x00
   STS TCNT1H, R18
   LDI R18, 0x00
	STS TCNT1L, R18
   RET
```

3. CODE OF TASK 2

```
#include <avr/io.h>
#include <util/delay.h>
int main(void)
{
 DDRB = 0x04;
//set pin PB2 to output, using PD2 as input, which is autoset, so no change needed
   while (1)
   {
        if ((PIND&0x04) == 4)                    // if PD2 is high
                {
                        PORTB = 0x04;                           // turn on LED
                        for (int i = 0; i < 10; i++)
                        {
                                _delay_ms(100);  // use 10 100ms delays to get a 1 second delay
                        }
                        PORTB = 0x00;
// if button is still depressed, wait until released to start again
                        while((PIND&0x04) == 4)
                        {}
                }
                else
                        PORTB = 0x00;
        }
}
```

ASSEMBLY CODE

```
.org 0
        ldi R16,low(RAMEND)
        out SPL,R16
        ldi R16,high(RAMEND)
        out SPH,R16
        SBI DDRB, 2     ; set PB2 as output
        LDI R16, 0x00
        OUT PORTB, R16   ; set all of PORTB to zero. start LED off
        LDI R17, 0x05
        STS TCCR1B, R17
        LDI R17, 0x04                   ; use R17 to AND with R16 to determin if bit 2 is 1
begin:
        IN R16, PIND
        AND R16, R17
        CPI     R16, 0x04
        BREQ blink                      ; if PD2 is 1 then branch to led blink
        RJMP begin
blink:
        LDI R18, 4
        OUT PORTB, R18
        LDI R18, 0
        STS TCNT1H, R18
        STS TCNT1L, R18   ; reset timer 1 to use for a 1 second delay
delay:
```

```
        LDS R18, TCNT1L
        LDS R19, TCNT1H
        CPI R18, 0XCF
        BRSH delay2
        RJMP delay
delay2:
        CPI R19, 0x03
        BRSH turnoff
        RJMP delay
turnoff:
        LDI R18, 0
        OUT PORTB, R18
        RJMP begin
```

4. CODE OF TASK 3

```
#include <avr/io.h>
int main(void){
        DDRB = 0x04;            // set PB2 as output
        PORTB = 0x00;           // set all of PORTB to zero. start LED off
        TCCR0B = 0x05;          // set timer 0 prescaler to 2014
        while(1)
        {
                if(TCNT0 == 0xF3){  // when counter = 243
                        PORTB ^= 0x04;      // toggle the LED
                        TCNT0 = 0x00;       // reset the counter
                }
        }
}
```

ASSEMBLY CODE

```
.org 0
  SBI DDRB, 2      ; set PB2 as output
  LDI R16, 0
  OUT PORTB, R16    ; set all of PORTB to zero. start LED off
  LDI R17, 5
  STS TCCR0B, R17   ; set timer 0 prescaler to 1024
  LDI R17, 4       ; used to XOR with R16 to toggle LED
begin:
  RCALL delay
  EOR R16, R17
  OUT PORTB, R16
  RJMP begin
delay1:
  LDS R18, TCNT0
  CPI R18, 0xF3
  BRSH overflow     ; check to see if TCNT0 = 255
  RJMP delay1
overflow:
  LDI R18, 0x00
  STS TCNT0, R18    ; reset TCNT0
delay2:
  LDS R18, TCNT0
  CPI R18, 0xE8     ;check to see if TCNT0 = 232, making total time 487
  BRSH done
```

```
done:
   LDI R18, 0x00
   STS TCNT0, R18
   RET
```

5. CODE OF TASK 4

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
ISR(TIMER0_OVF_vect)
{
        PORTB ^= 0xFF;                  // toggle LED
        TCNT0 = 0x0B ;                  // reset counter
}
int main()
{
        DDRB = 0x04;                    // set PD2 to output
        TIMSK0 = (1 << TOIE1);      // enable timer1 interrupt
        TCCR0B = 5;                             // set prescaler to 1024
        TCNT0 = 0x0B;                   // set timer to 255- 244 = 11
        sei();                                  // enable global interrupt
        while(1) {}
}
```

ASSEMBLY CODE

```
.org 0
        rjmp start
.org OVF0addr
   rjmp  timer0ovf
start:
        ldi             R16,low(RAMEND)
        out             SPL,R16
        ldi             R16,high(RAMEND)
        out             SPH,R16
        SBI             DDRB, 2                 ; set PD2 as output
        LDI             R16, 0
        OUT             PORTB, R16              ; set all of PORTD to zero. start LED off
        LDI             R17, 5
        OUT             TCCR0B, R17             ; set timer 0 prescaler to 1024
        LDI             R17, 1
        STS             TIMSK0, R17             ; enable the overflow interrupt
        LDI             R17, 4                  ; used to XOR with R16 to toggle
        LDI             R18, 0X0B
        OUT             TCNT0, R18              ; set TCNT0 to 0x0B, loading TCNT0 to 11
        SEI
wait:
        Rjmp wait
       ; ************timer1 overflow subroutine**************
timer0ovf:
        EOR   R16, R17
        OUT   PORTB, R16                        ; toggle LED
        LDI   R18, 0X0B
```

```
        OUT   TCNT0, R18                ; set TCNT0 to 0x0B, reloading TCNT0 to 11
        RETI
```

6. CODE OF TASK 5

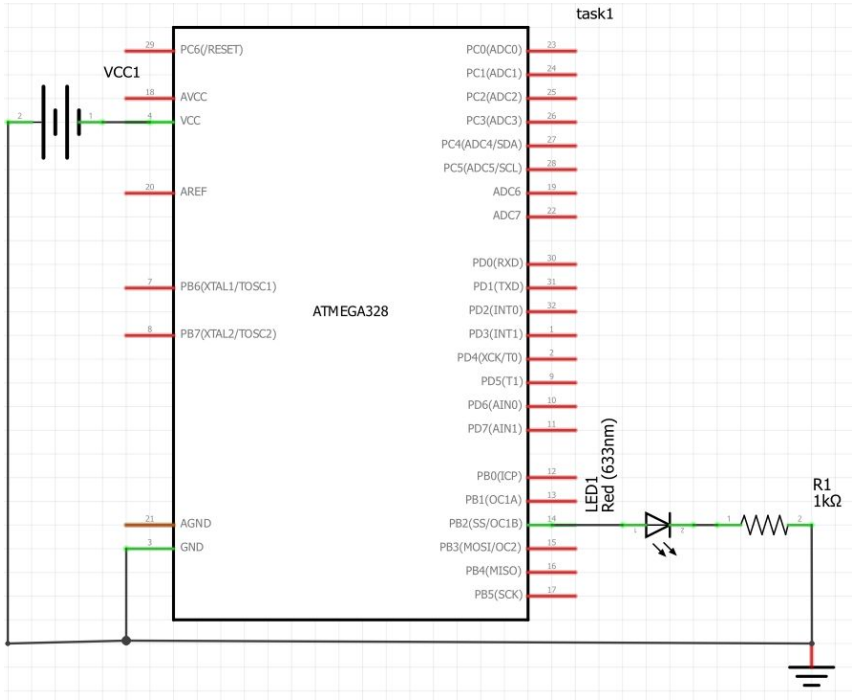
```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
ISR(INT0_vect)
{
        PORTB = 0X04;
        for (int i = 0; i < 10; i++)
        {
                _delay_ms(100);          // use 10 100ms delays to get a 1 second delay
        }
        PORTB = 0x00;
}
int main(void)
{
        DDRB = 0x04;     //set pin PB2 to output, using PD2 as input, which is autoset, so no change needed
        EIMSK = 0X01;
        EIFR = 0X01;
        EICRA = 0X03;
        sei();
        while (1)
        {}
}
```

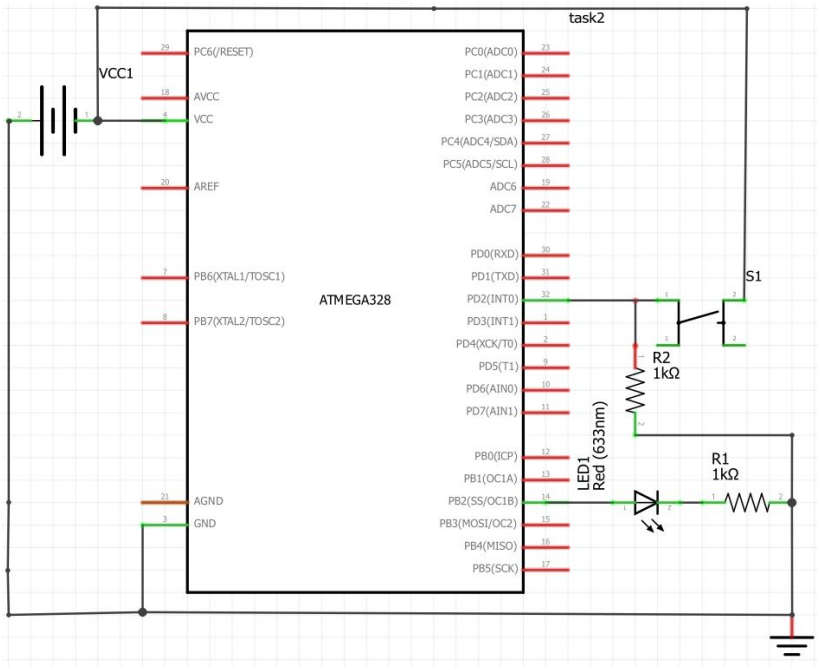
ASSEMBLY CODE

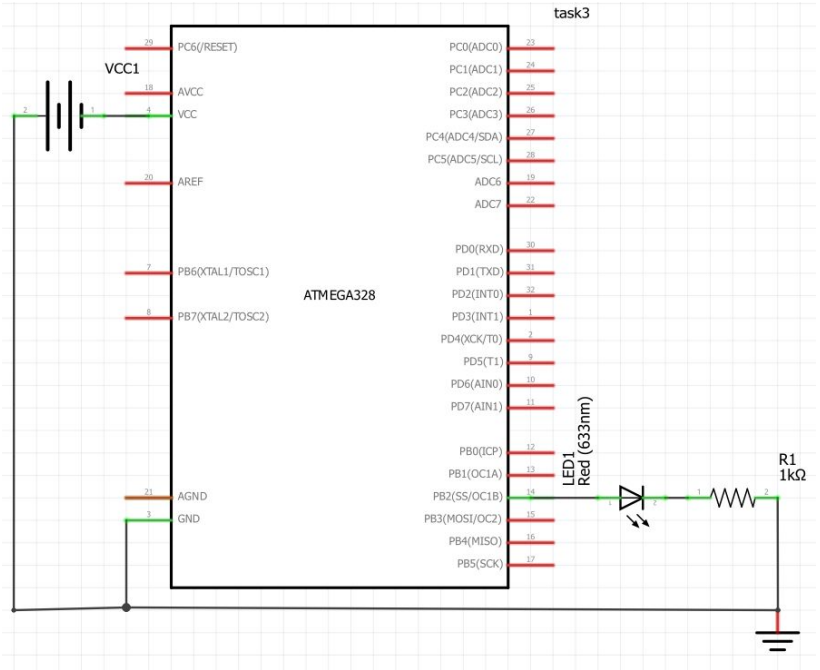
```
.org 0
        rjmp start
.org INT0addr
        rjmp  interupt0
start:
        ldi             R16,low(RAMEND)
        out             SPL,R16
        ldi             R16,high(RAMEND)
        out             SPH,R16
         SBI            DDRB, 2                 ; set PD2 as output
         LDI            R16, 0
         OUT            PORTB, R16              ; set all of PORTD to zero. start LED off
        LDI             R16, 1
        OUT             EIMSK, R16              ; set interrupt0 mask
        OUT             EIFR, R16               ; set interrupt0 flag
        LDI             R16, 4
        STS             EICRA, R16              ; set interrupt control register to rising edge
        LDI             R16, 5
        OUT             TCCR0B, R16             ; set timer0 prescaler to 1024. timer used for 1 second delay
        SEI
wait:
   Rjmp wait
        ; ************interrupt0 subroutine**************
interupt0:
        LDI             R16, 4
         OUT            PORTB, R16
```

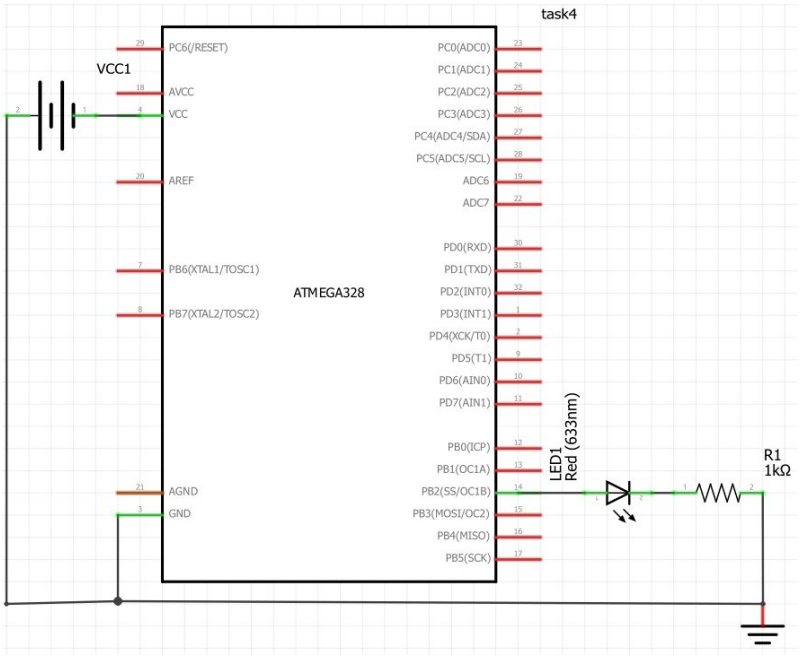
```
        LDI     R16, 0
        OUT     TCNT0, R16          ; reset timer for delay
        LDI     R17, 0
; since timer0 is 8 bit, and we need total value of 975 multiple loops are needed.
; using r17 as a counter, when 255 has been reached 3 times (765)
; go to a final loop for the last 210 counts
delay:
        IN      R18, TCNT0
        CPI     R18, 0XFF
        BRSH    checkcount
        RJMP    delay
checkcount:
        LDI     R16, 0
        OUT     TCNT0, R16          ; reset timer for the rest of delay
        LDI     R16, 1
        ADD     R17, R16
        CPI     R17, 3
        BREQ    delay
delay2:
        IN      R18, TCNT0
        CPI     R18, 0xD2
        BRSH    turnoff
        RJMP    delay2
turnoff:
        LDI     R18, 0
        OUT     PORTB, R18
        RETI
```

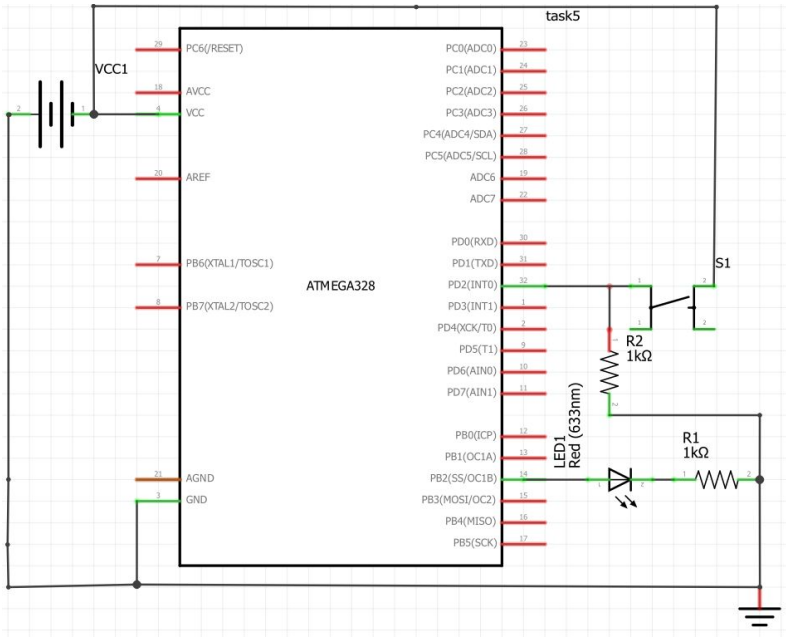
7. SCHEMATICS




task2
VCC1
PC6(/RESET)
AVCC
VCC
AREF
PB6(XTAL1/TOSC1)
PB7(XTAL2/TOSC2)
AGND
GND
ATMEGA328
PC0(ADC0)
PC1(ADC1)
PC2(ADC2)
PC3(ADC3)
PC4(ADC4/SDA)
PC5(ADC5/SCL)
ADC6
ADC7
PD0(RXD)
PD1(TXD)
PD2(INT0)
PD3(INT1)
PD4(XCK/T0)
PD5(T1)
PD6(AIN0)
PD7(AIN1)
PB0(ICP)
PB1(OC1A)
PB2(SS/OC1B)
PB3(MOSI/OC2)
PB4(MISO)
PB5(SCK)
S1
R2
1kΩ
LED1
Red (633nm)
R1
1kΩ

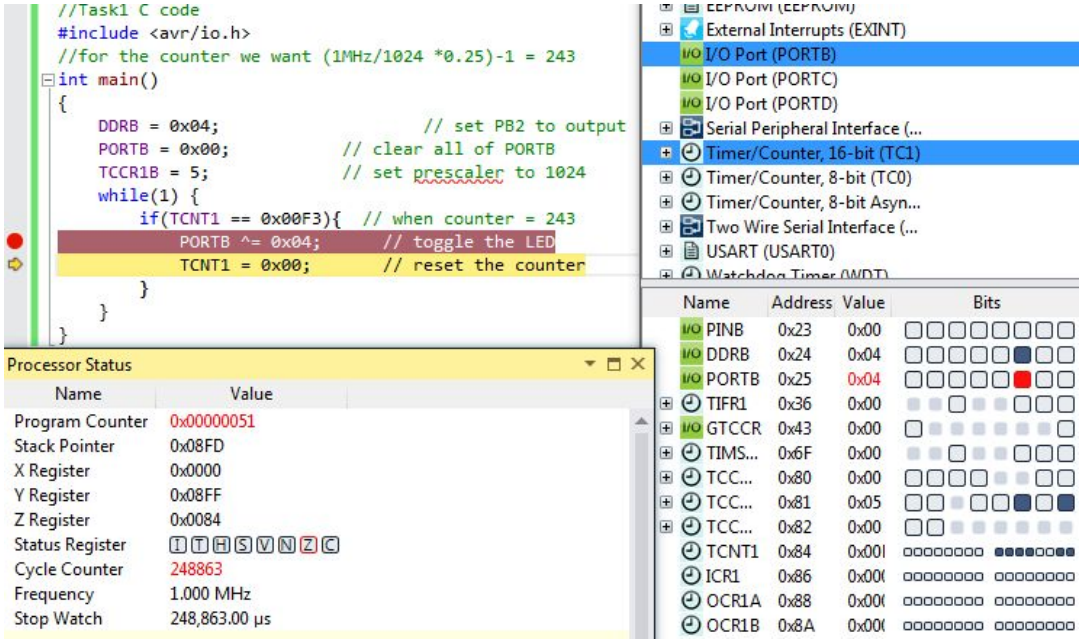

task3
VCC1
PC6(/RESET)
AVCC
VCC
AREF
PB6(XTAL1/TOSC1)
PB7(XTAL2/TOSC2)
AGND
GND
ATMEGA328
PC0(ADC0)
PC1(ADC1)
PC2(ADC2)
PC3(ADC3)
PC4(ADC4/SDA)
PC5(ADC5/SCL)
ADC6
ADC7
PD0(RXD)
PD1(TXD)
PD2(INT0)
PD3(INT1)
PD4(XCK/T0)
PD5(T1)
PD6(AIN0)
PD7(AIN1)
PB0(ICP)
PB1(OC1A)
PB2(SS/OC1B)
PB3(MOSI/OC2)
PB4(MISO)
PB5(SCK)
LED1
Red (633nm)
R1
1kΩ





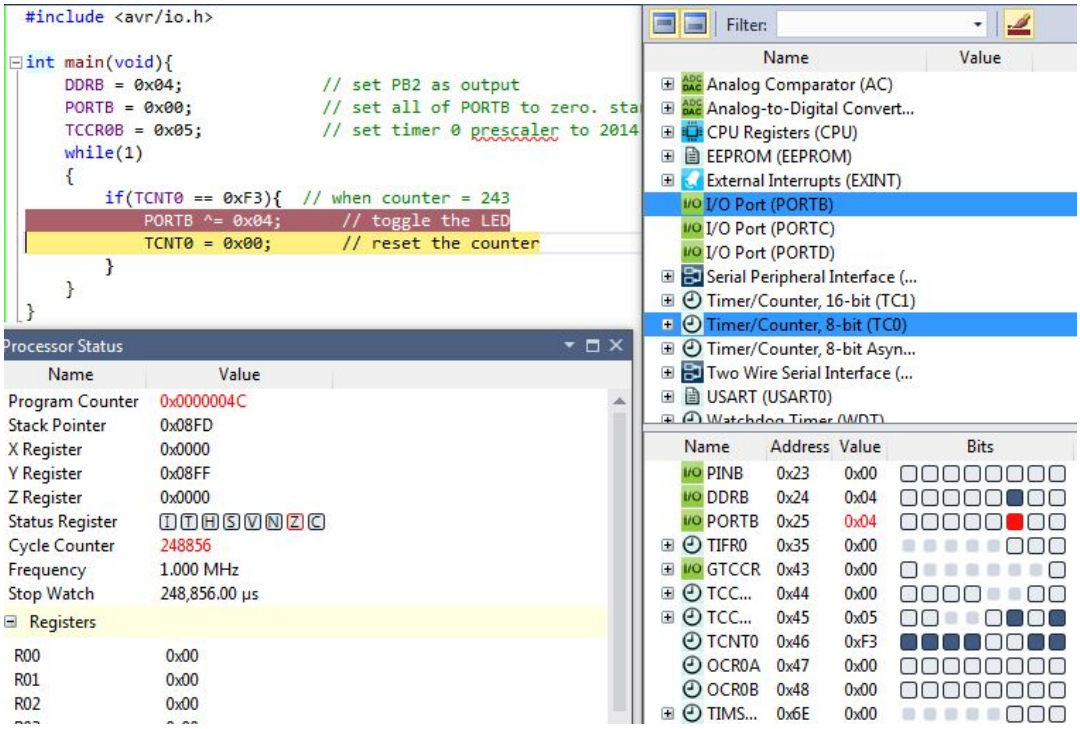
8. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

Task 1



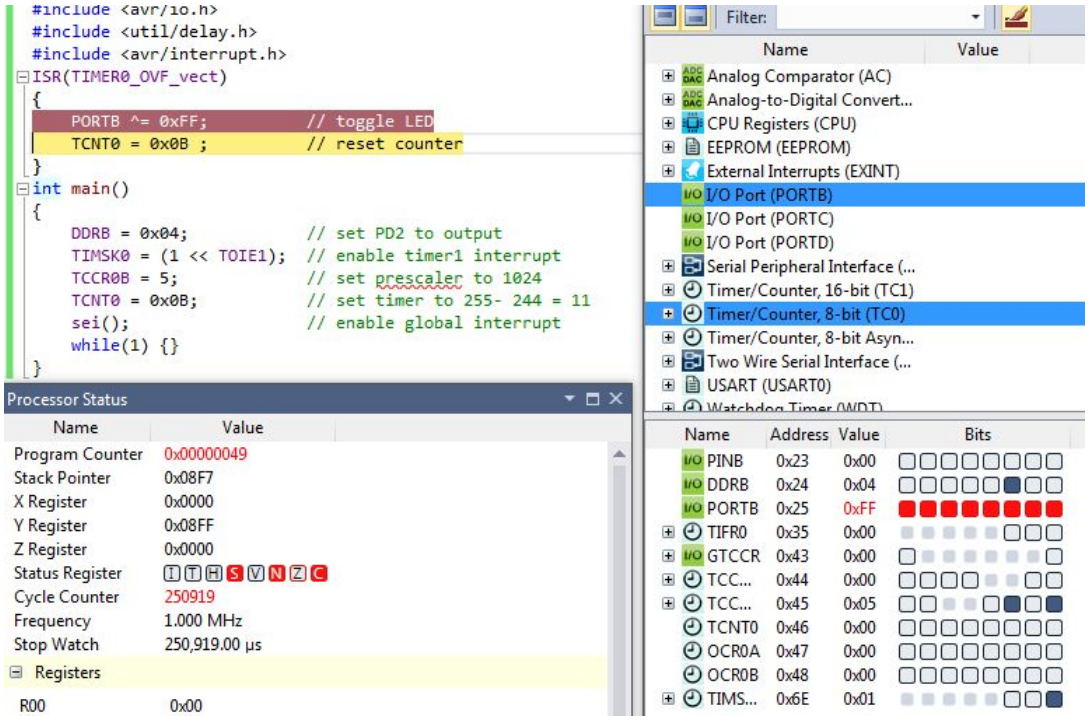
I could not get an acurate simulation of task 2

Task 3



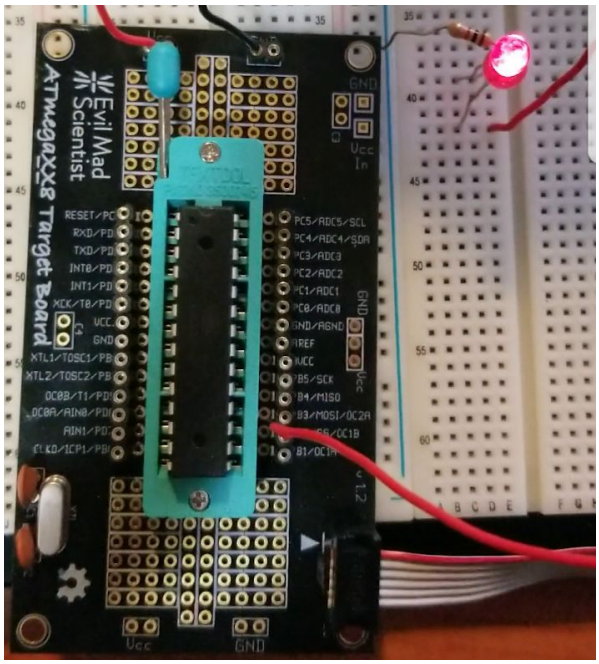
task 4

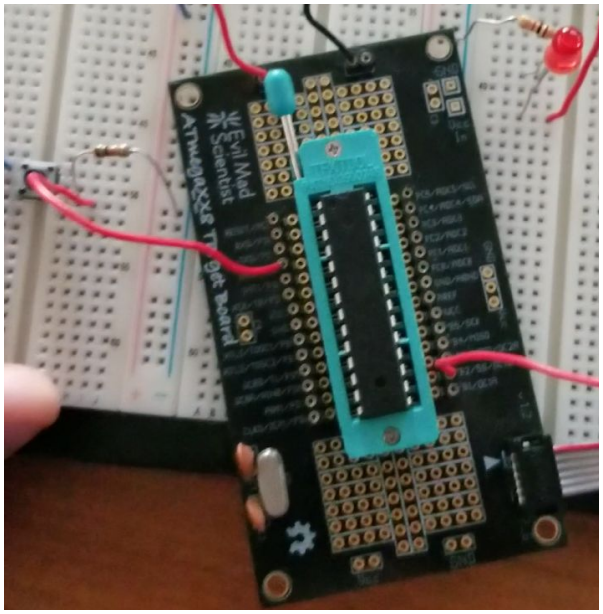
```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
ISR(TIMER0_OVF_vect)
{
    PORTB ^= 0xFF;          // toggle LED
    TCNT0 = 0x0B ;          // reset counter
}
int main()
{
    DDRB = 0x04;            // set PD2 to output
    TIMSK0 = (1 << TOIE1);  // enable timer1 interrupt
    TCCR0B = 5;             // set prescaler to 1024
    TCNT0 = 0x0B;           // set timer to 255- 244 = 11
    sei();                  // enable global interrupt
    while(1) {}
}
```

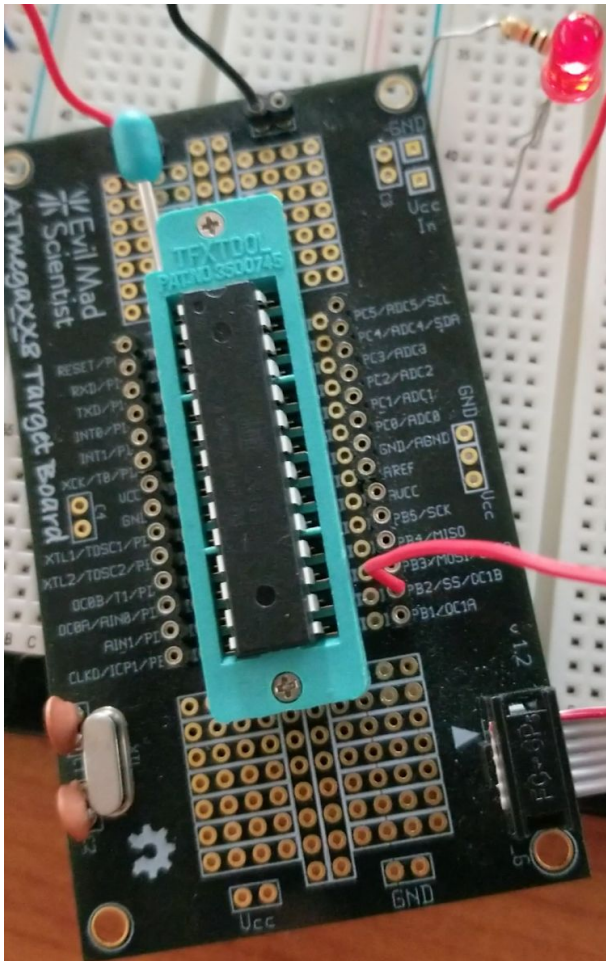


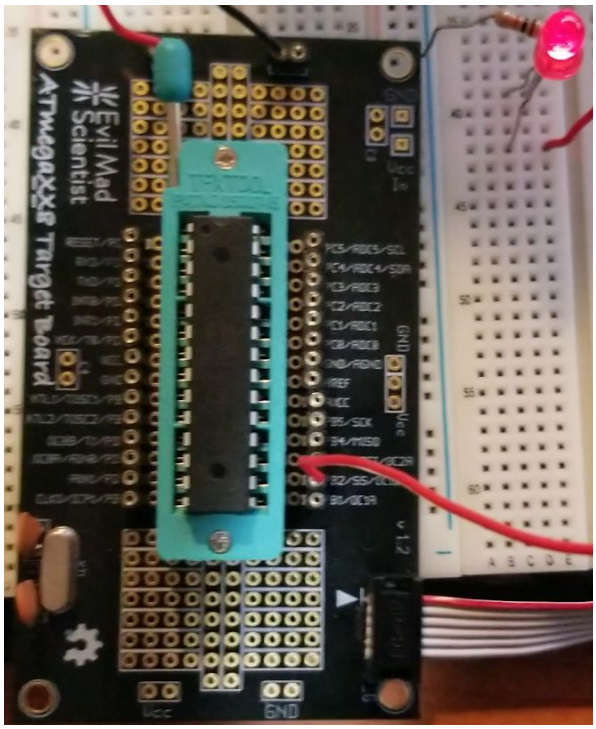
I could not get an accurate simulation of task 5

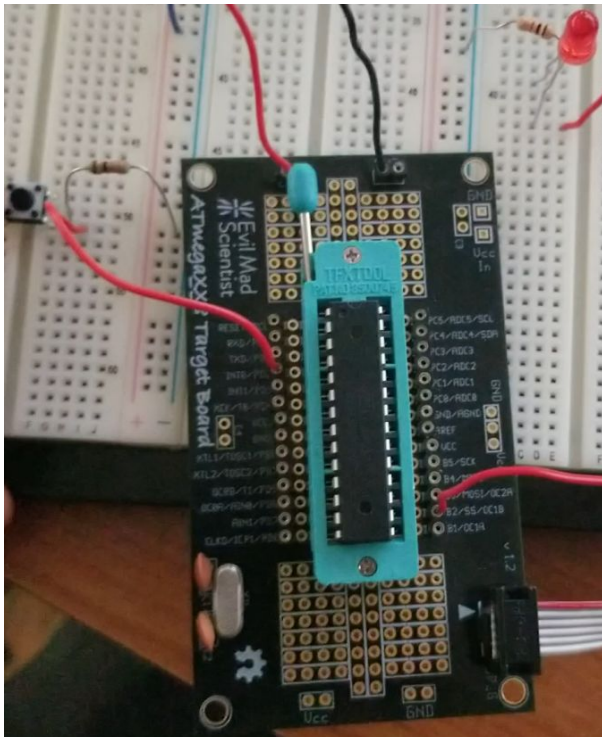
9. SCREENSHOT OF EACH DEMO (BOARD SETUP)






Evil Mad Scientist
ATmega8/X8 Target Board
GND
Vcc
In
PC5/ADC5/SCL
PC4/ADC4/SDA
PC3/ADC3
PC2/ADC2
PC1/ADC1
PC0/ADC0
GND/AGND
AREF
AVCC
PB5/SCK
PB2/SS/OC1B
PB1/OC1A
v1.2
Vcc
GND


Evil Mad Scientist
ATmega8/X8 Target Board
GND
Vcc
In
v1.2
Vcc
GND



10. VIDEO LINKS OF EACH DEMO

task 1
https://youtu.be/ckgXGJZ8pak

task 2
https://youtu.be/YCDicUKqlfY

task 3
https://youtu.be/HmIoemgSRps

task 4
https://youtu.be/rWyIhevbh38

task 5
https://youtu.be/K9bU9p78BZ0

11. GITHUB LINK OF THIS DA

https://github.com/Pogoptomus/CPE301.git

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

"*This assignment submission is my own, original work*".
Phillip Sortomme