Brian Lopez

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

For Tasks 1, 3, and 4:

- Atmega328P
- Programmer for the chip
- One 5K resistor that is connected to port B.2
- One LED light

For Tasks 2, and 5:

- Same as above
- Another 5K resistor that is connected to port D.2
- One pushbutton, one side connects to the resistor and the other side to ground

2. CODE OF TASK 1/A

```
; Design Assignment 2, task 1
; Blinks an LED with a DC of 50% and a period of .5 sec

start:
    SBI DDRB, 2        ; set port b.2 as an output port
    LDI R17, 0x00
    OUT PORTB, R17     ; turn led off first
myLp:
    RCALL delay        ; make a delay
    LDI R17, 0x80
    OUT PORTB, R17     ; turn led on
    RCALL delay        ; make another delay
    LDI R17, 0x00
    OUT PORTB, R17     ; turn led off
    RJMP myLp

delay:
    LDI R18, 125
delLp1:
    LDI R19, 228
delLp2:
    NOP
    NOP
    NOP
    NOP
    DEC R19
    BRNE delLp2
    DEC R18
    BRNE delLp1
    RET
```

3. CODE OF TASK 1/B

```
#include <avr/io.h>
#define F_CPU 100000UL
#include <util/delay.h>

/*
      DA2_task1 in c
      This will toggle an led on/off with a period of .5 sec.
```

```
*/

int main(void)
{
    DDRB = (1<<PB7);   // set pb7 as an output
       while(1)
       {
              PORTB ^= (1<<PB7);   // flip port7 on/off
              _delay_ms(2500);     // delay for .25 sec
              _delay_ms(1);
       }
}
```

4. CODE OF TASK 2/A

```
; Design Assignment 2, task 2 in assembly


start:
    CBI DDRD, 2         ; set port d.2 as an input
    SBI DDRB, 2
    LDI R17, 0x04
    OUT PORTB, R17      ; load 1 to port b.2
again:
    SBIC PIND, 2        ; check if d.2 is clear (0). means button is pressed
    RJMP ledOff         ; if not pressed, just turn the led off by making it an input
    SBI PORTB, 2        ; Button is pressed, turn led on by setting is as output
    LDI R21, 4          ; The timer below is for ~0.25 seconds, so repeat that 4 times
to get one second
timer:
    LDI R20, 60
    OUT TCNT0, R20      ; load value for timer to start
    LDI R20, 0x00
    OUT TCCR0A, R20     ; normal mode
    LDI R20, (1<<CS02 | 1<<CS00)   ; 1024 prescalar
    OUT TCCR0B, R20
tLp:
    IN R20, TIFR0         ; read timer value
    SBRS R20, 0           ; if not done, keep checking
    RJMP tLp
    LDI R20, 0x0
    OUT TCCR0B, R20       ; stop the timer
    LDI R20, (1<<TOV0)
    OUT TIFR0, R20        ; clear the flag
    DEC R21
    CPI R21, 0            ; check if this has been done 4 times, if not
    BRNE timer            ; repeat
    CBI PORTB, 2          ; after 1 second, turn led off again by setting it as an
input.
    RJMP again
ledOff:
    CBI PORTB, 2          ; turns the led off by setting it as an input.
    RJMP again
```

5. CODE OF TASK 2/B

```
// Design Assignment 2, task 2 in c

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

```
int main(void)
{
    DDRB |= (1<<2);          // set pinb.2 as an output
    DDRD |= (0<<2);          // set pind.2 as an input
    PORTB = 0x00;            // initially have the led off

    while(1)
    {
           if (!(PIND & 0x04))          // wait until the switch is pressed (pind.2
will be 0 when pressed)
           {
                  PORTB ^= (1<<2);              // turn on the led
                  for(int i = 0; i < 4; i++)    // the timer goes on for .25 seconds,
so do that 4 times to get one second
                  {
                         TCNT0 = 61;               // load start value
                         TCCR0A = 0x00;            // normal mode
                         TCCR0B = 0x05;            // prescalar of 1024
                         while((TIFR0 & 0x01) == 0)
                         {
                                // wait until overflow occurs
                         }
                         TCCR0B = 0;              // turn off the timer
                         TIFR0 |= 1<<TOV0;        // reset the flag
                  }
                  PORTB ^= (1<<2);             // once the for loop is done, turn the
led off
           }
    }
}
```

6. CODE FOR TASK 3/A

```
start:
    SBI DDRB, 7
    LDI R16, 0x80
    LDI R17, 0
    OUT PORTB, R17
begin:
    LDI R20, 61
    OUT TCNT0, R20    ; load timer 0
    LDI R20, 0x00
    OUT TCCR0A, R20   ; normal mode for timer 0
    LDI R20, (1<<CS02 | 1<<CS00)  ; prescalar of 1024
    OUT TCCR0B, R20
loop:
    IN R20, TIFR0     ; check if overflow occurs
    SBRS R20, 0
    RJMP loop
    LDI R20, 0x0
    OUT TCCR0B, R20    ; stop the timer
    LDI R20, (1<<TOV0)
    OUT TIFR0, R20     ; clear the flag
    EOR R17, R16
    OUT PORTB, R17     ; flip led on/off
    RJMP begin
```

7. CODE FOR TASK 3/B

```c
/*
 Design Assignment 2, Task 3 in C
 */

#include <avr/io.h>

int main(void)
{
    DDRB |= (1<<2);
    PORTB = 0;
    while(1)
    {
        TCNT0 = 61;
        TCCR0A = 0x00;              // normal mode
        TCCR0B = 0x05;              // 1024 prescalar
        while((TIFR0 & 0x01) == 0)
        {
            // wait until overflow occurs
        }
        TCCR0B = 0;                 // stop the clock
        TIFR0 |= 1<<TOV0;           // clear the flag
        PORTB ^= (1<<2);            // switch the led on/off
    }
}
```

8. CODE FOR TASK 4/A

```
; Design assignment 2, task 4 in assembly


.org 0x00
    RJMP start
.org 0x20
    RJMP T0_overflow

start:
    SBI DDRB, 2                 ; set pin 2 of port b as an output
    LDI R17, 0
    LDI R16, 0x04
    OUT PORTB, R17
begin:
    LDI R19, 61                  ; for timer that takes 25ms
    OUT TCNT0, R19
    LDI R19, 0x00                ; load normal mode into the timer
    OUT TCCR0A, R19
    LDI R20, (1<<CS02 | 1<<CS00)   ; prescalar of 1024
    OUT TCCR0B, R20
    LDI R20, 0x01
    STS TIMSK0, R20              ; enable the timer interupt
    SEI                          ; enable global interupts
again:
    RJMP again

T0_overflow:
    LDI R20, (1<<TOV0)           ; reset the interupt
    OUT TIFR0, R20
    EOR R17, R16
```

```
    OUT PORTB, R17                ; invert the value to the led
    LDI R20, 61                   ; restart the timer
    OUT TCNT0, R20
    RETI
```

9. CODE FOR TASK 4/B

```
/*
 Design Assignment 2, Task 4 in C
 */

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


int main(void)
{
    DDRB |= (1<<2);    // set portb.2 as an output
    PORTB = 0x00;      // turn led off at beginning
    TCNT0 = 61;        // load timer value
    TCCR0A = 0x00;     // normal mode
    TCCR0B = 0x05;     // 1024 prescalar
    TIMSK0 = (1<<TOIE0); // set overflow interrupt
    sei();             // turn on global interrupts
    while(1)
    {
        // do nothing now :)
    }
}

ISR (TIMER0_OVF_vect)
{
    TCNT0 = 61;      // reload timer value
    PORTB ^= (1<<2);   // switch led on/off
    TIFR0 |= (1<<TOV0);   // clear the flag
}
```

10. CODE FOR TASK 5/A

```
.ORG 0x00
    JMP main
.ORG 0x02
    JMP int0Chng
.ORG 0x16
    JMP time1CTC

main:
    LDI R16, HIGH(RAMEND)        ; initialize stack
    OUT SPH, R16
    LDI R16, LOW(RAMEND)
    OUT SPL, R16

    SBI DDRB, 2                  ; set portb.2 as an output
    CBI PORTB, 2                 ; turn led off initially

    LDI R20, HIGH(3125)          ; set ctc compare value (~1 second)
    STS OCR1AH, R20
    LDI R20, LOW(3125)
    STS OCR1AL, R20
    LDI R20, 1<<OCIE1A
```

```
    STS TIMSK1, R20             ; enable timer 1 interrupt

    CBI DDRD, 2                 ; set portd.2 as an input
    LDI R20, 0x00
    STS EICRA, R20
    LDI R20, 1<<INT0
    OUT EIMSK, R20              ; enable int0 interrupt

    SEI                         ; enable global interrupts
lp:
    RJMP lp                     ; be here forever!!

int0Chng:
    SBI PORTB, 2                ; turn on the led
    LDI R20, (1<<WGM12 | 1<<CS12)   ; turn on timer 1 in ctc mode
    STS TCCR1B, R20             ; and prescalar of 256

    LDI R20, 1<<INTF0
    OUT EIFR, R20               ; clear int0 flag
    RETI

time1CTC:
    CBI PORTB, 2                ; turn off led
    LDI R20, 1<<OCF1A
    STS TIFR1, R20              ; clear timer1 flag
    LDI R20, 0x00
    STS TCCR1B, R20             ; turn off timer
    RETI
```

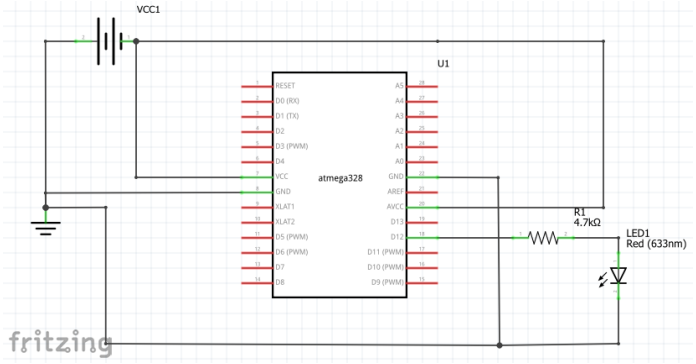
11. CODE FOR TASK 5/B:

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

int main(void)
{
    DDRB |= (1<<2);      // set pin b.2 as an output
    PORTD = 1<<2;
    EICRA = 0x02;        // falling edge
    EIMSK = (1<<INT0);   // turn on int0 interrupt
    sei();               // enable global interrupts

    while(1)
    {

    }
}

ISR (TIMER1_COMPA_vect)
{
    PORTB ^= (1<<2);         // turn off the LED
    TIFR1 |= (1<<OCF1A);     // clear the timer flag
    TCCR1B = 0;              // turn off the timer
}

ISR(INT0_vect)
{
    PORTB ^=  (1<<2);       // turn on the LED
    OCR1A = 3125;           // Load the compare value
```

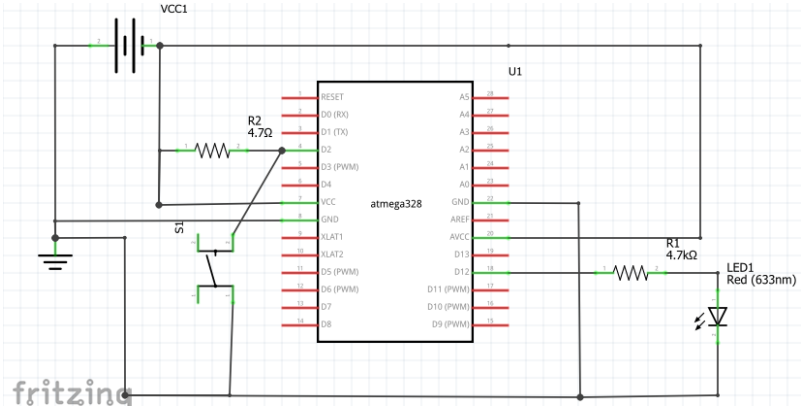
```
        TCNT1 = 0;
        TCCR1B |= (1<<WGM12);    // CTC mode
        TIMSK1 |= (1<<OCIE1A);  // turn on timer1 interrupt
        TCCR1B |= (1<<CS12); // prescalar of 256
        EIFR |= (1<<INTF0);      // clear int0 flag
    }
```

12. SCHEMATICS

For Tasks 1, 3, and 4:

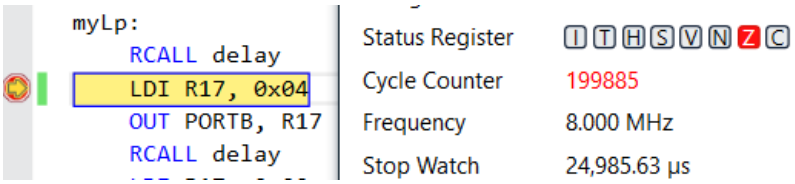


For Tasks 2 and 5:



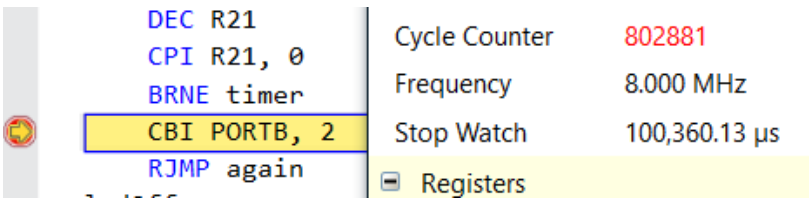
13. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

For Task 1:



change is done after the first LED change, so .25 seconds on and .25 seconds off

For Task 2:



Measured from button press to the LED turning off

For Task 3:

```
EOR R17, R16
OUT PORTB, R17
RJMP begin
```

Cycle Counter	199698
Frequency	8.000 MHz
Stop Watch	24,962.25 µs

Measured the same as task 1

For Task 4:

```
OUT TIFR0, R20
EOR R17, R16
OUT PORTB, R17
LDI R20, 61
```

Cycle Counter	199702
Frequency	8.000 MHz
Stop Watch	24,962.75 µs

For Task 5:

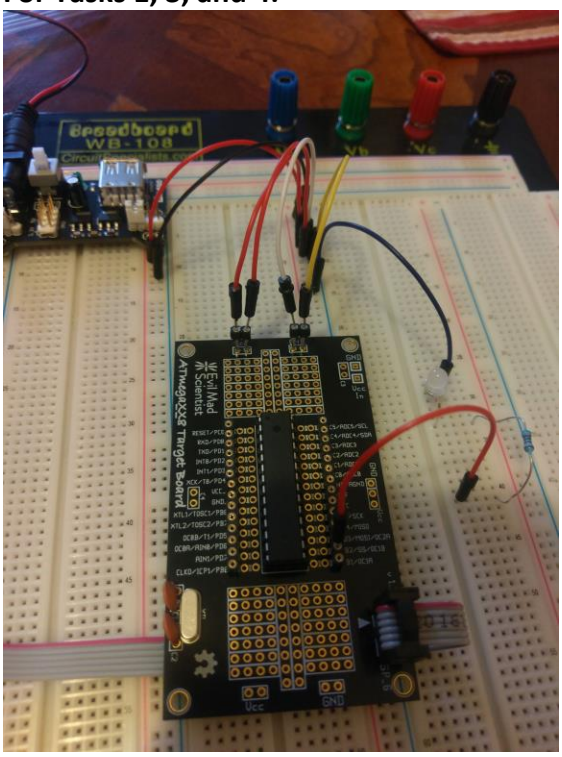
```
time1CTC:
CBI PORTB, 2
LDI R20, 1<<OCF1A
STS TIFR1, R20
```

Cycle Counter	800304
Frequency	8.000 MHz
Stop Watch	100,038.00 µs

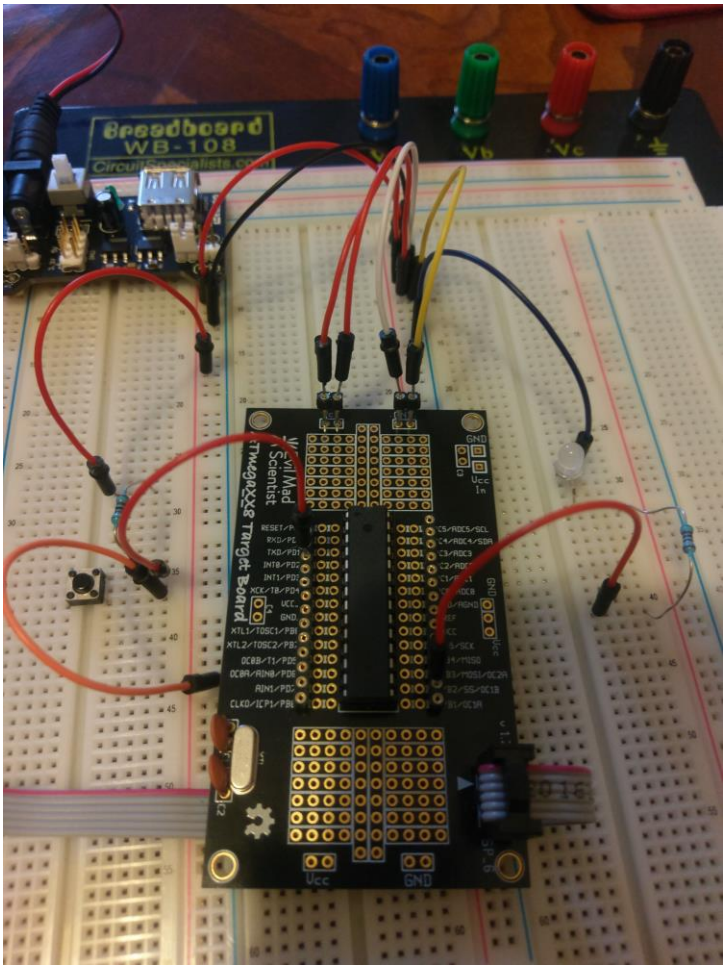
Measured from button press to LED turning off

14. SCREENSHOT OF EACH DEMO (BOARD SETUP)

For Tasks 1, 3, and 4:



For Tasks 2 and 5:



15. VIDEO LINKS OF EACH DEMO

Playlist that contains all of the videos:
https://www.youtube.com/watch?v=DEEne5UPYv8&index=10&list=PL_kN1D7twBrzRnyjlp5erD3DDkbXzAMii

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

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