CPE301 – SPRING 2019

Design Assignment 2C

Student Name: Allis Hierholzer
Student #: 2000160356
Student Email: hierholz@unlv.nevada.edu
Primary Github address: https://github.com/acexhp/submission_da.git
Directory: Repository/cpe301/DesignAssignment/DA2C

Task:

The goal of the assignment is use GPIO and delays using Timers and Interrupts:

1. Implement Design Assignment 2A using Timer 0 – normal mode. Count OVF occurrence if needed. Do not use interrupts.

2. Implement Design Assignment 2A using TIMER0_OVF_vect interrupt mechanism in normal mode.

3. Implement Design Assignment 2A using TIMER0_COMPA_vect interrupt mechanism in CTC mode.

Submission:

The following are required for successful completion of the design assignment:

a. AVR C code that has been compiled and working for all four tasks. Verify the period and duty cycle of the waveforms in simulation and emulation.

b. The C code should be well documented with explanation of every instruction.

c. A word document that contains the code with comments, complete schematics, that includes the AVR, components connected on the breadboard and LED should be included. Follow the template provided.

d. A snapshot of the board with connected components and a video of the complete LED bar blink sequence should be recorded and uploaded to Youtube and the line to be provided for each task.

e. The git directory should have DA2\DA2T1, DA2\DA2T2, … folders, with one doc file and video link file.

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

- ATMEGA328P XPLAINED MINI
- MULTIFUNCTION SHIELD
- ATMEL STUDIO 7.0
- Oscilloscope

2. INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1/A

1A

```
#define F_CPU 16000000UL                         //clock runs at 16 MHz
#include <avr/io.h>
#include<util/delay.h>

int main()
{
       int overflow = 0;                         //initialize overflow
       DDRB |= (1 << DDB2);                      //PB2 as output
       TCCR0A = 0;
       TCNT0 = 0x00;                             //start timer
       TCCR0B = (1 << CS02) | (1 << CS00);       //pre-scaler = 1024


       while (1){
              while ((TIFR0 & 0x01) == 0);       //detects overflow
              TCNT0 = 0x00;                      //resets counter
              TIFR0 = 0x01;                      //reset overflow flag
              overflow++;                        //inc overflow
              if (overflow <= 26)                //led on
                     PORTB = (0 << DDB2);
                     else PORTB = (1 << DDB2);
              if (overflow == 44) {
                     overflow = 0;               //turns off led
              }
       }
}
```

1B

```
#define F_CPU 16000000UL                         //clock runs at 16 MHz
#include <avr/io.h>
#include<util/delay.h>

int overflow = 0;                                //initialize overflow

int main()
{
       DDRB |= (1<<2);                           //set PORTB2 as output
       PORTB |= (1<<2);                          //Turn LED off
       DDRC &= (0<<2);                           // set PORTC1 for input
       PORTC |= (1<<2);                          // enable pull-up

       TCCR0A = 0;
       TCCR0B = (1 << CS02) | (1 << CS00);       //pre-scaler = 1024

       //when the PINC is pressed, LED pulses
       while (1) {
              if (!(PINC & (1<<PINC1)))
              {
                     overflow = 0;
                     TCNT0 = 0;
              }

              while ((TIFR0 & 0x01) == 0);      //detects overflow
              TCNT0 = 0x00;                     //resets counter
              TIFR0 = 0x01;                     //reset overflow flag
              overflow++;                       //inc overflow
              if (overflow <= 69)               //led turns on
              PORTB = (0 << DDB2);
              else PORTB = (1 << DDB2);         //led off
       }

       return 0;
}
```

2A

```
#define F_CPU 16000000UL                        //clock runs at 16 MHz
#include <avr/io.h>
#include<avr/interrupt.h>

int overflow = 0;                               //initialize overflow

int main(void)
{
       DDRB |= (1 << DDB2);                     //PB2 as output
       TIMSK0 |= (1 << TOIE0);                  //enables interrupt
       TCNT0 = 0;                               //start counter
       sei();                                   //enables interrupt
       TCCR0B = (1 << CS02) | (1 << CS00);      //pre-scaler = 1024
       while (1)
       {

       }
}

ISR (TIMER0_OVF_vect)                           //timer0 overflow interrupt
{
              while ((TIFR0 & 0x01) == 0);      //detects overflow
              TCNT0 = 0x00;                     //resets counter
              TIFR0 = 0x01;                     //reset overflow flag
              overflow++;                       //inc overflow
              if (overflow <= 13)               //led on (13 instead of 26 because...
              PORTB = (0 << DDB2);              //... overflow is being doubled)
              else PORTB = (1 << DDB2);
              if (overflow == 22) {
                     overflow = 0;              //turns off led
              }
}
```

2B

```
#define F_CPU 16000000UL                        //clock runs at 16 MHz
#include <avr/io.h>
#include<avr/interrupt.h>

int overflow = 0;                               //initialize overflow
int main(void)
{
        DDRB |= (1 << DDB2);                        //PB2 as output
        TIMSK0 |= (1 << TOIE0);                     //enables interrupt
        TCNT0 = 0;                                  //start counter
        sei();                                      //enables interrupt
        TCCR0B = (1 << CS02) | (1 << CS00);         //pre-scaler = 1024
        while (1)
        {

        }
}

ISR (TIMER0_OVF_vect)                               //timer0 overflow interrupt
{
while (1) {
                if (!(PINC & (1<<PINC1)))
                {
                        overflow = 0;
                        TCNT0 = 0;
                }

                while ((TIFR0 & 0x01) == 0);        //detects overflow
                TCNT0 = 0x00;                       //resets counter
                TIFR0 = 0x01;                       //reset overflow flag
                overflow++;                         //inc overflow
                if (overflow <= 69)                 //led turns on
                PORTB = (0 << DDB2);
                else PORTB = (1 << DDB2);           //led off
        }

                }
```

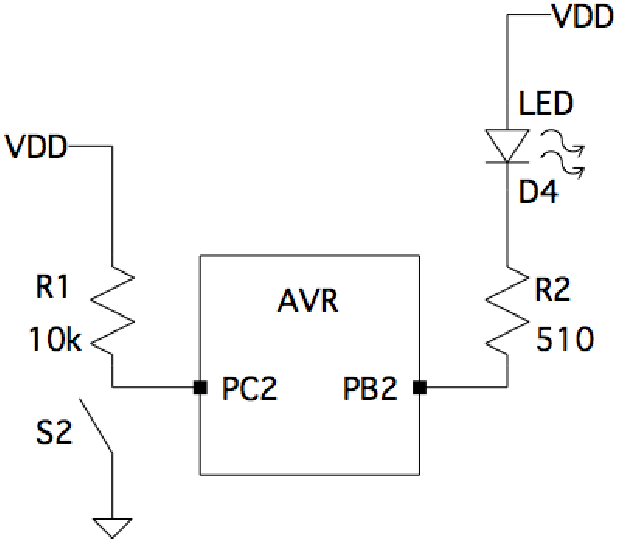
<u>3A</u>

```
#define F_CPU 16000000UL                            //clock runs at 16 MHz
#include <avr/io.h>
#include<avr/interrupt.h>

int overflow = 0;                                   //initialize overflow

int main(void)
{
        DDRB |= (1 << DDB2);                        //PB2 as output
        TCNT0 = 0;                                  //start counter
        OCR0A = 255;                                //load compare reg value
        TCCR0A |= (1 << WGM01);                     //set to ctc mode
        TIMSK0 |= (1 << OCIE0A);                    //set interrupt on compare match
        TCCR0B = (1 << CS02) | (1 << CS00);         //pre-scaler = 1024
        sei();                                      //enables interrupt
        while (1)
        {

        }
}

ISR (TIMER0_COMPA_vect)                             //timer0 overflow interrupt
{
                while ((TIFR0 & 0x02) == 0);        //detects overflow
                TCNT0 = 0x00;                       //resets counter
                TIFR0 = 0x02;                   //reset overflow flag
                overflow++;                     //inc overflow
                if (overflow <= 13)             //led on (13 instead of 26 because...
                PORTB = (0 << DDB2);            //... overflow is being doubled)
                else PORTB = (1 << DDB2);
                if (overflow == 22) {
                        overflow = 0;           //turns off led
                }
}
```

3B

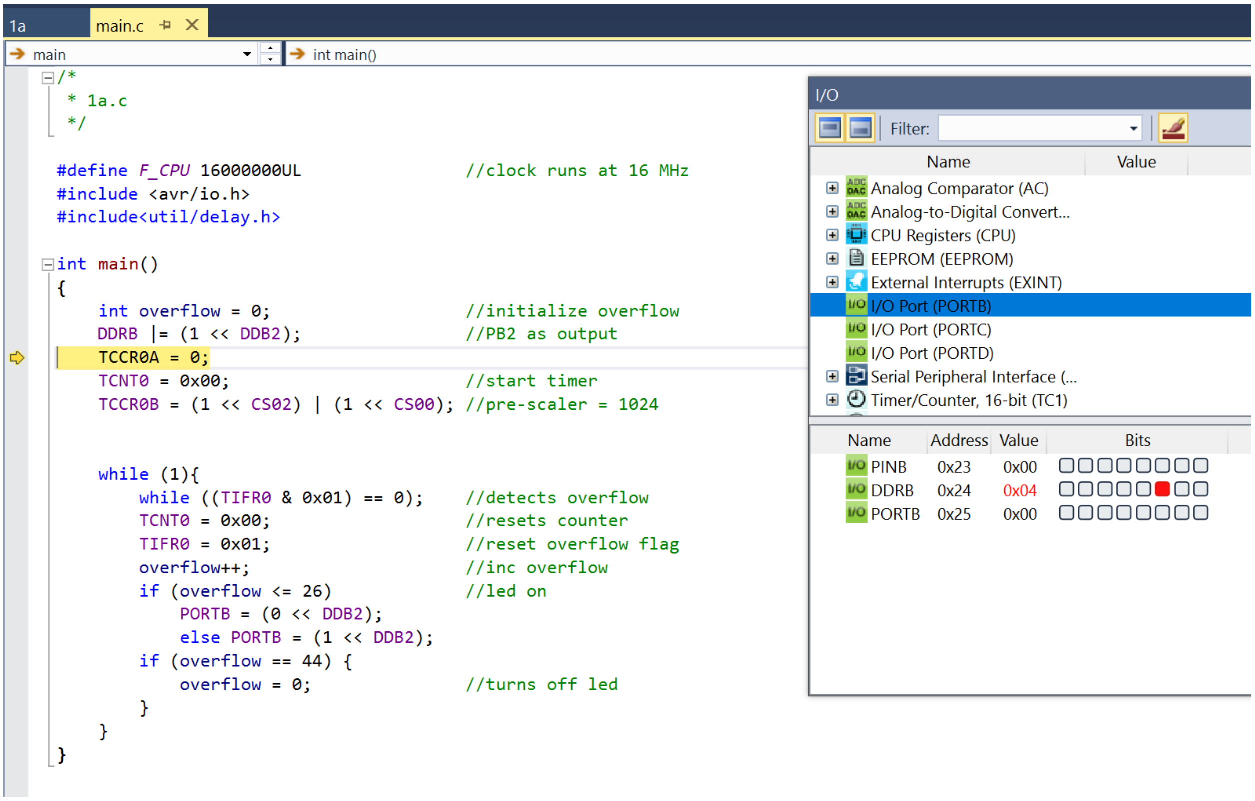
```
#define F_CPU 16000000UL                        //clock runs at 16 MHz
#include <avr/io.h>
#include<avr/interrupt.h>

int overflow = 0;                               //initialize overflow

int main(void)
{
        DDRB |= (1 << DDB2);                    //PB2 as output
        TCNT0 = 0;                              //start counter
        OCR0A = 255;                            //load compare reg value
        TCCR0A |= (1 << WGM01);                 //set to ctc mode
        TIMSK0 |= (1 << OCIE0A);                //set interrupt on compare match
        TCCR0B = (1 << CS02) | (1 << CS00);     //pre-scaler = 1024
        sei();                                  //enables interrupt
        while (1)
        {

        }
}

ISR (TIMER0_COMPA_vect)                         //timer0 overflow interrupt
{
while (1) {
        if (!(PINC & (1<<PINC1)))
        {
                overflow = 0;
                TCNT0 = 0;
        }

        while ((TIFR0 & 0x02) == 0);            //detects overflow
        TCNT0 = 0x00;                           //resets counter
        TIFR0 = 0x02;                           //reset overflow flag
        overflow++;                             //inc overflow
        if (overflow <= 69)                     //led turns on
        PORTB = (0 << DDB2);
        else PORTB = (1 << DDB2);               //led off
}

}
```

3. SCHEMATICS



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

1A

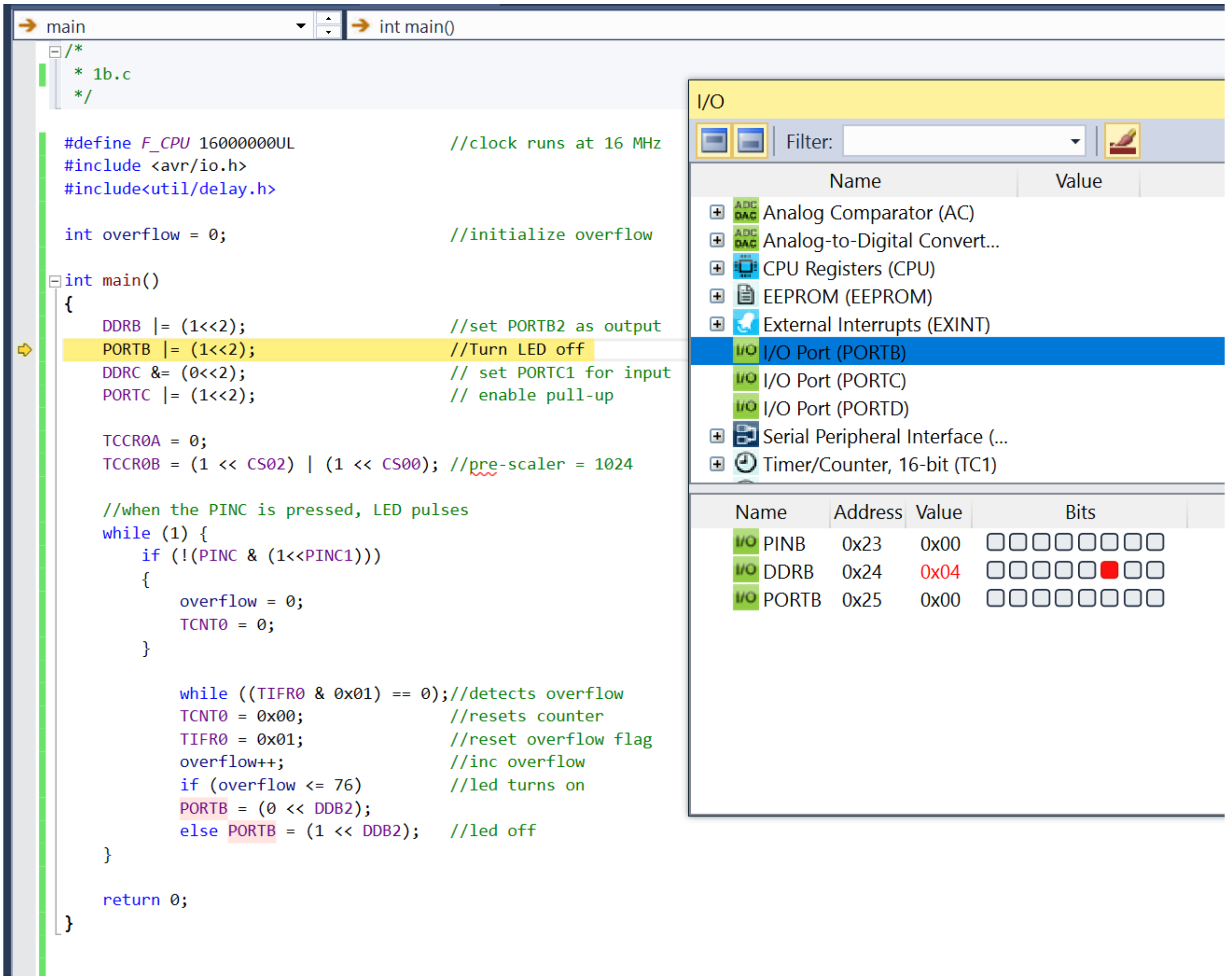


```c
/*
 * 1a.c
 */

#define F_CPU 16000000UL                //clock runs at 16 MHz
#include <avr/io.h>
#include<util/delay.h>

int main()
{
    int overflow = 0;                   //initialize overflow
    DDRB |= (1 << DDB2);                //PB2 as output
    TCCR0A = 0;
    TCNT0 = 0x00;                       //start timer
    TCCR0B = (1 << CS02) | (1 << CS00); //pre-scaler = 1024


    while (1){
        while ((TIFR0 & 0x01) == 0);    //detects overflow
        TCNT0 = 0x00;                   //resets counter
        TIFR0 = 0x01;                   //reset overflow flag
        overflow++;                     //inc overflow
        if (overflow <= 26)             //led on
            PORTB = (0 << DDB2);
            else PORTB = (1 << DDB2);
        if (overflow == 44) {
            overflow = 0;               //turns off led
        }
    }
}
```

Name	Value
Analog Comparator (AC)	
Analog-to-Digital Convert...	
CPU Registers (CPU)	
EEPROM (EEPROM)	
External Interrupts (EXINT)	
I/O Port (PORTB)	
I/O Port (PORTC)	
I/O Port (PORTD)	
Serial Peripheral Interface (...	
Timer/Counter, 16-bit (TC1)	

Name	Address	Value	Bits
PINB	0x23	0x00	□□□□□□□□
DDRB	0x24	0x04	□□□□□■□□
PORTB	0x25	0x00	□□□□□□□□

1B

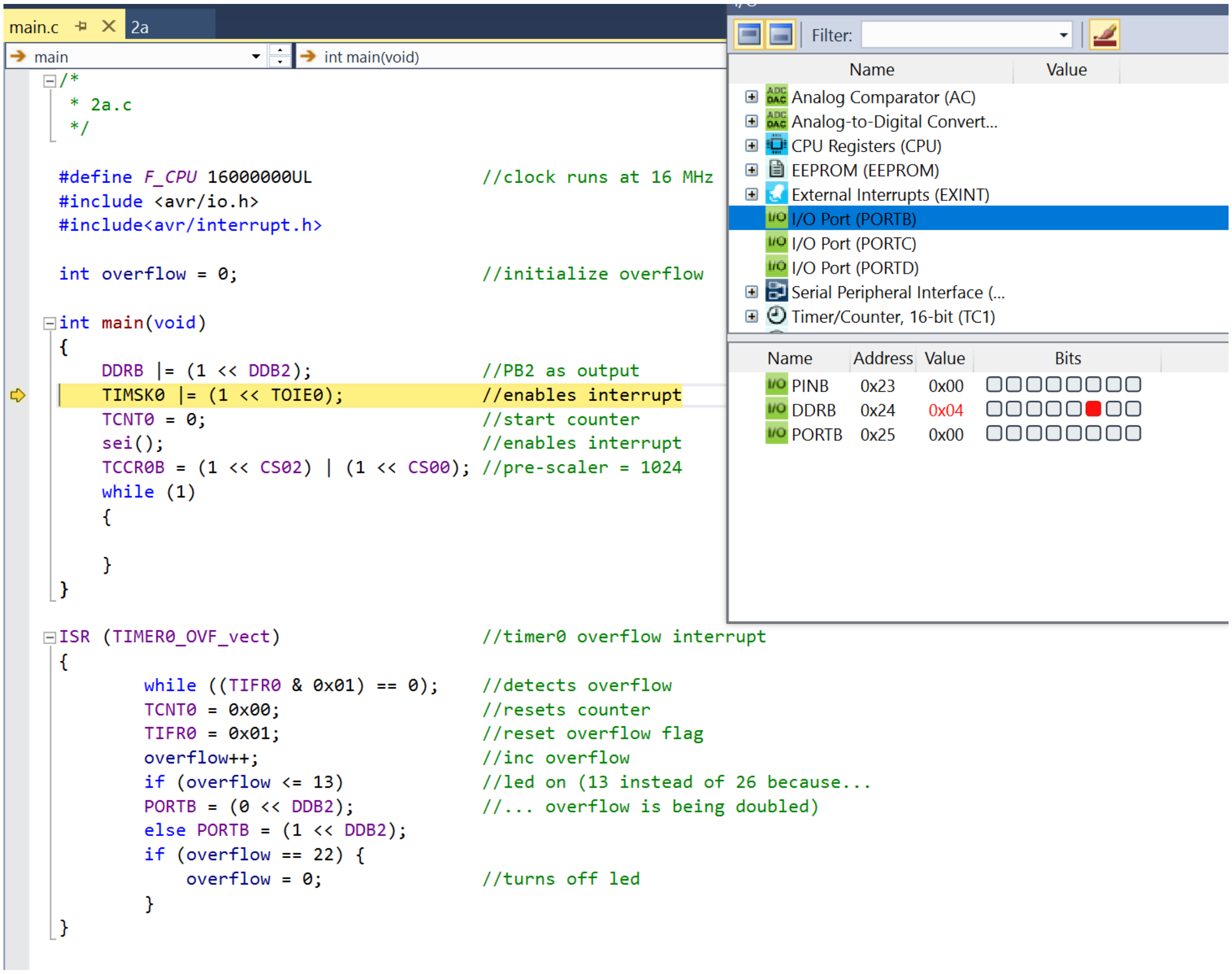


```
/*
 * 1b.c
 */

#define F_CPU 16000000UL                //clock runs at 16 MHz
#include <avr/io.h>
#include<util/delay.h>

int overflow = 0;                       //initialize overflow

int main()
{
    DDRB |= (1<<2);                     //set PORTB2 as output
    PORTB |= (1<<2);                    //Turn LED off
    DDRC &= (0<<2);                     // set PORTC1 for input
    PORTC |= (1<<2);                    // enable pull-up

    TCCR0A = 0;
    TCCR0B = (1 << CS02) | (1 << CS00); //pre-scaler = 1024

    //when the PINC is pressed, LED pulses
    while (1) {
        if (!(PINC & (1<<PINC1)))
        {
            overflow = 0;
            TCNT0 = 0;
        }

            while ((TIFR0 & 0x01) == 0);//detects overflow
            TCNT0 = 0x00;               //resets counter
            TIFR0 = 0x01;               //reset overflow flag
            overflow++;                 //inc overflow
            if (overflow <= 76)         //led turns on
            PORTB = (0 << DDB2);
            else PORTB = (1 << DDB2);   //led off
    }

    return 0;
}
```

Name	Value
Analog Comparator (AC)	
Analog-to-Digital Convert...	
CPU Registers (CPU)	
EEPROM (EEPROM)	
External Interrupts (EXINT)	
I/O Port (PORTB)	
I/O Port (PORTC)	
I/O Port (PORTD)	
Serial Peripheral Interface (...	
Timer/Counter, 16-bit (TC1)	

Name	Address	Value	Bits
PINB	0x23	0x00	□□□□□□□□
DDRB	0x24	0x04	□□□□□■□□
PORTB	0x25	0x00	□□□□□□□□

2A

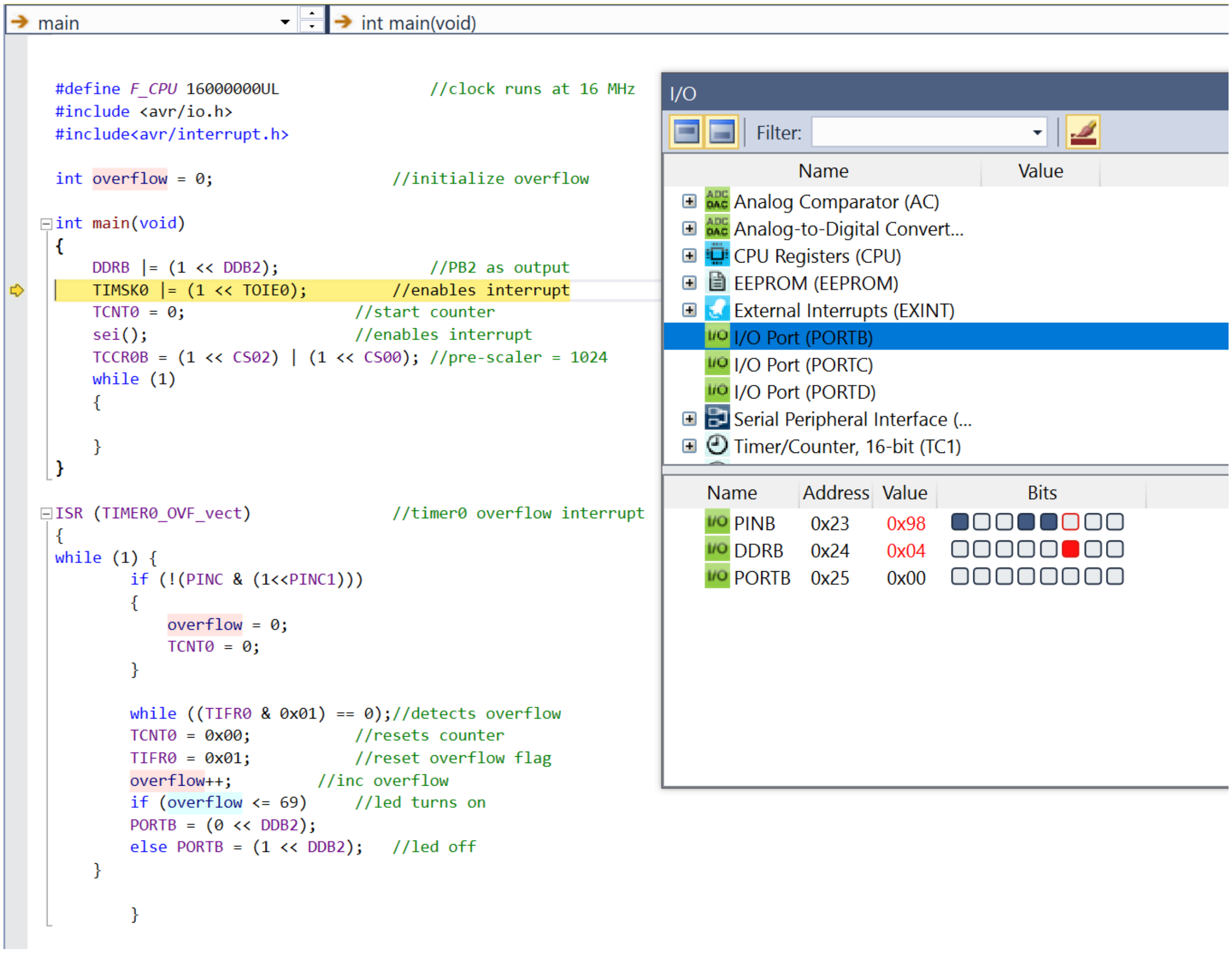


```
/*
 * 2a.c
 */

#define F_CPU 16000000UL                //clock runs at 16 MHz
#include <avr/io.h>
#include<avr/interrupt.h>

int overflow = 0;                       //initialize overflow

int main(void)
{
    DDRB |= (1 << DDB2);                //PB2 as output
    TIMSK0 |= (1 << TOIE0);             //enables interrupt
    TCNT0 = 0;                          //start counter
    sei();                              //enables interrupt
    TCCR0B = (1 << CS02) | (1 << CS00); //pre-scaler = 1024
    while (1)
    {

    }
}

ISR (TIMER0_OVF_vect)                   //timer0 overflow interrupt
{
        while ((TIFR0 & 0x01) == 0);    //detects overflow
        TCNT0 = 0x00;                   //resets counter
        TIFR0 = 0x01;                   //reset overflow flag
        overflow++;                     //inc overflow
        if (overflow <= 13)             //led on (13 instead of 26 because...
        PORTB = (0 << DDB2);            //... overflow is being doubled)
        else PORTB = (1 << DDB2);
        if (overflow == 22) {
            overflow = 0;               //turns off led
        }
}
```

Name	Value
Analog Comparator (AC)	
Analog-to-Digital Convert...	
CPU Registers (CPU)	
EEPROM (EEPROM)	
External Interrupts (EXINT)	
I/O Port (PORTB)	
I/O Port (PORTC)	
I/O Port (PORTD)	
Serial Peripheral Interface (...	
Timer/Counter, 16-bit (TC1)	

Name	Address	Value	Bits
PINB	0x23	0x00	□□□□□□□□
DDRB	0x24	0x04	□□□□□■□□
PORTB	0x25	0x00	□□□□□□□□

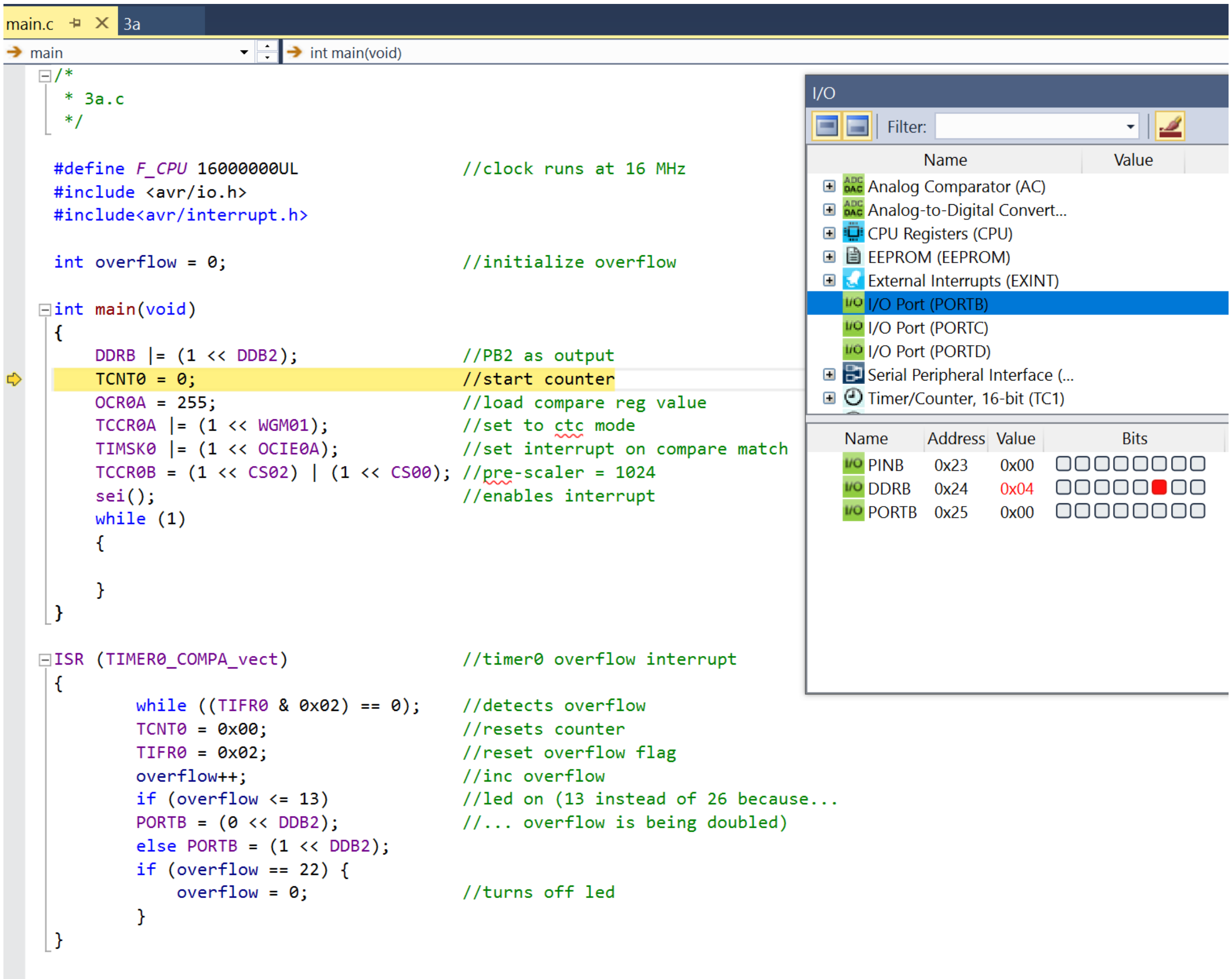
2B



```
#define F_CPU 16000000UL                //clock runs at 16 MHz
#include <avr/io.h>
#include<avr/interrupt.h>

int overflow = 0;                   //initialize overflow

int main(void)
{
    DDRB  |= (1 << DDB2);               //PB2 as output
    TIMSK0 |= (1 << TOIE0);         //enables interrupt
    TCNT0 = 0;                  //start counter
    sei();                      //enables interrupt
    TCCR0B = (1 << CS02) | (1 << CS00); //pre-scaler = 1024
    while (1)
    {

    }
}

ISR (TIMER0_OVF_vect)               //timer0 overflow interrupt
{
while (1) {
        if (!(PINC & (1<<PINC1)))
        {
            overflow = 0;
            TCNT0 = 0;
        }

        while ((TIFR0 & 0x01) == 0);//detects overflow
        TCNT0 = 0x00;           //resets counter
        TIFR0 = 0x01;           //reset overflow flag
        overflow++;         //inc overflow
        if (overflow <= 69)     //led turns on
        PORTB = (0 << DDB2);
        else PORTB = (1 << DDB2);   //led off
    }

        }
```

Name	Address	Value	Bits
PINB	0x23	0x98	■□□■■□□□
DDRB	0x24	0x04	□□□□□■□□
PORTB	0x25	0x00	□□□□□□□□

3A

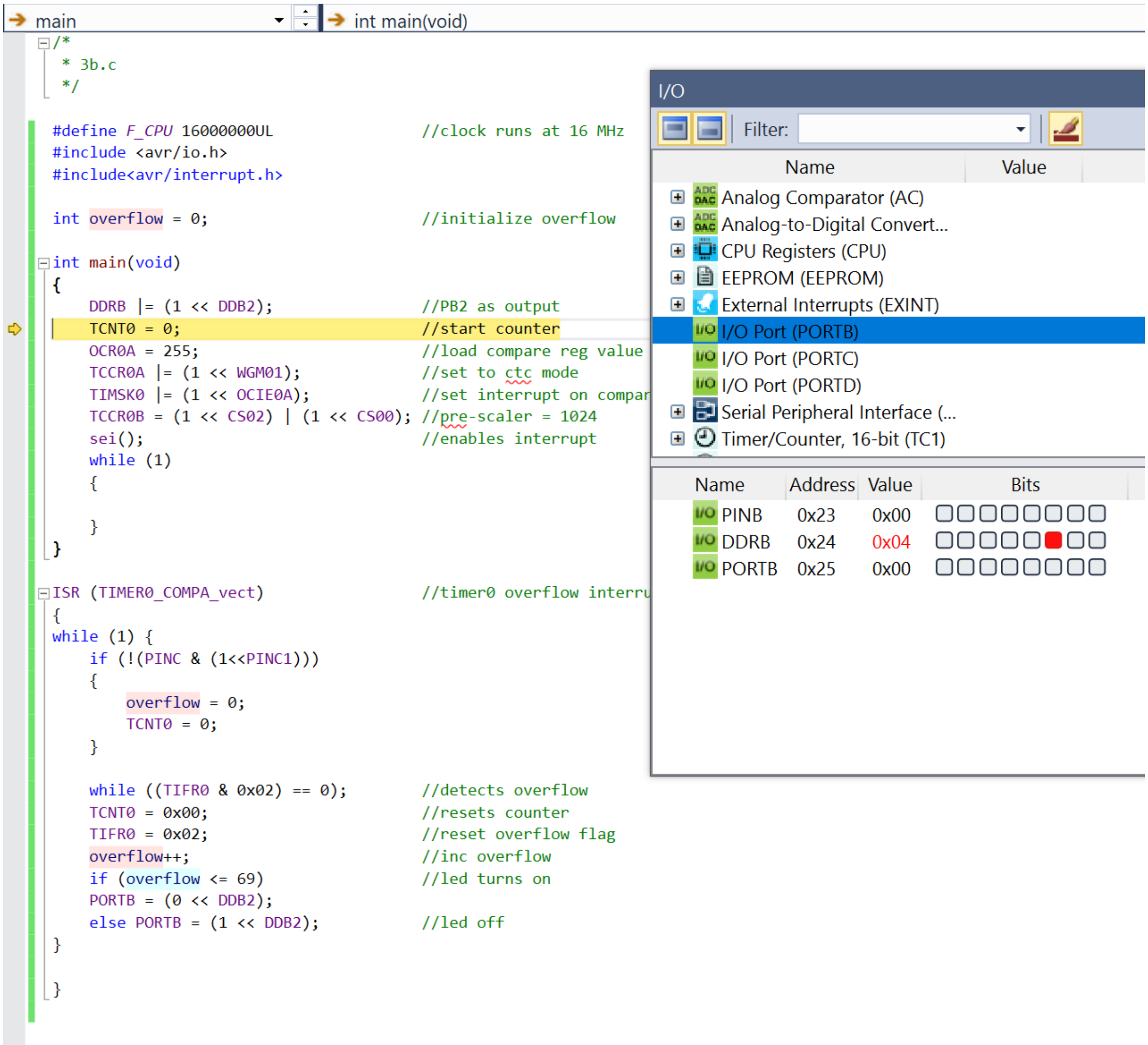


```
/*
 * 3a.c
 */

#define F_CPU 16000000UL                //clock runs at 16 MHz
#include <avr/io.h>
#include<avr/interrupt.h>

int overflow = 0;                       //initialize overflow

int main(void)
{
    DDRB |= (1 << DDB2);                //PB2 as output
    TCNT0 = 0;                          //start counter
    OCR0A = 255;                        //load compare reg value
    TCCR0A |= (1 << WGM01);             //set to ctc mode
    TIMSK0 |= (1 << OCIE0A);            //set interrupt on compare match
    TCCR0B = (1 << CS02) | (1 << CS00); //pre-scaler = 1024
    sei();                              //enables interrupt
    while (1)
    {

    }
}

ISR (TIMER0_COMPA_vect)                 //timer0 overflow interrupt
{
        while ((TIFR0 & 0x02) == 0);    //detects overflow
        TCNT0 = 0x00;                   //resets counter
        TIFR0 = 0x02;                   //reset overflow flag
        overflow++;                     //inc overflow
        if (overflow <= 13)             //led on (13 instead of 26 because...
        PORTB = (0 << DDB2);            //... overflow is being doubled)
        else PORTB = (1 << DDB2);
        if (overflow == 22) {
            overflow = 0;               //turns off led
        }
}
```

Name	Value
Analog Comparator (AC)	
Analog-to-Digital Convert...	
CPU Registers (CPU)	
EEPROM (EEPROM)	
External Interrupts (EXINT)	
I/O Port (PORTB)	
I/O Port (PORTC)	
I/O Port (PORTD)	
Serial Peripheral Interface (...	
Timer/Counter, 16-bit (TC1)	

Name	Address	Value	Bits
PINB	0x23	0x00	□□□□□□□□
DDRB	0x24	0x04	□□□□□■□□
PORTB	0x25	0x00	□□□□□□□□

3B

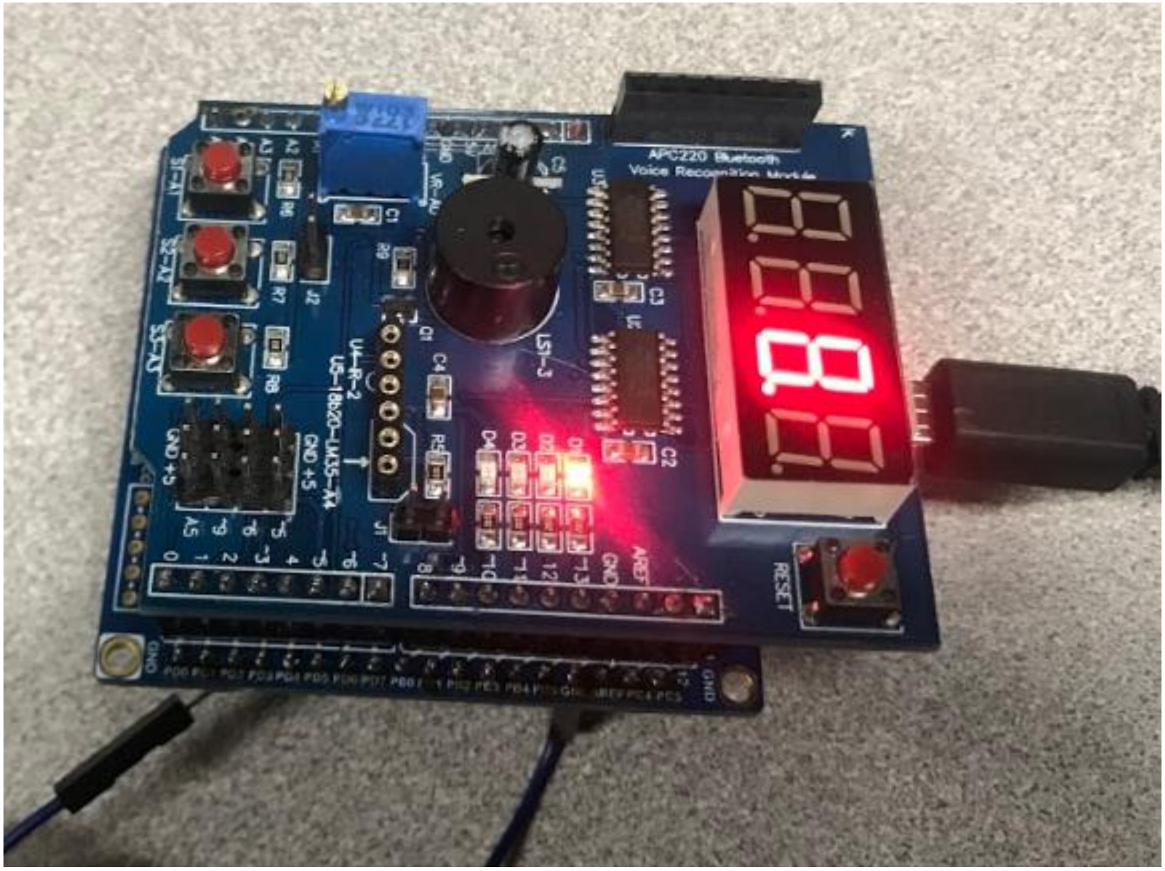


```
/*
 * 3b.c
 */

#define F_CPU 16000000UL                //clock runs at 16 MHz
#include <avr/io.h>
#include<avr/interrupt.h>

int overflow = 0;                       //initialize overflow

int main(void)
{
    DDRB |= (1 << DDB2);                //PB2 as output
    TCNT0 = 0;                          //start counter
    OCR0A = 255;                        //load compare reg value
    TCCR0A |= (1 << WGM01);             //set to ctc mode
    TIMSK0 |= (1 << OCIE0A);            //set interrupt on compar
    TCCR0B = (1 << CS02) | (1 << CS00); //pre-scaler = 1024
    sei();                              //enables interrupt
    while (1)
    {

    }
}

ISR (TIMER0_COMPA_vect)                 //timer0 overflow interru
{
while (1) {
    if (!(PINC & (1<<PINC1)))
    {
        overflow = 0;
        TCNT0 = 0;
    }

    while ((TIFR0 & 0x02) == 0);        //detects overflow
    TCNT0 = 0x00;                       //resets counter
    TIFR0 = 0x02;                       //reset overflow flag
    overflow++;                         //inc overflow
    if (overflow <= 69)                 //led turns on
    PORTB = (0 << DDB2);
    else PORTB = (1 << DDB2);           //led off
}

}
```

Name	Address	Value	Bits
PINB	0x23	0x00	□□□□□□□□
DDRB	0x24	0x04	□□□□□🟥□□
PORTB	0x25	0x00	□□□□□□□□

5. SCREENSHOT OF EACH DEMO (BOARD SETUP)

The board set up is the same for all exercises



6. VIDEO LINKS OF EACH DEMO

1A

https://youtu.be/7cCjv1m9Di4

1B

https://youtu.be/z2lXkrFeqw0

2A

https://youtu.be/m-iaM6LmcC8

2B

https://youtu.be/j0gw1afVVrg

3A

https://youtu.be/hrxYflyjL0o

3B

https://youtu.be/MIzcfgecc1g

7. GITHUB LINK OF THIS DA

https://github.com/acexhp/submission_da.git

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

"*This assignment submission is my own, original work*".
Allis Hierholzer