CPE301 – SPRING 2023

Design Assignment 3

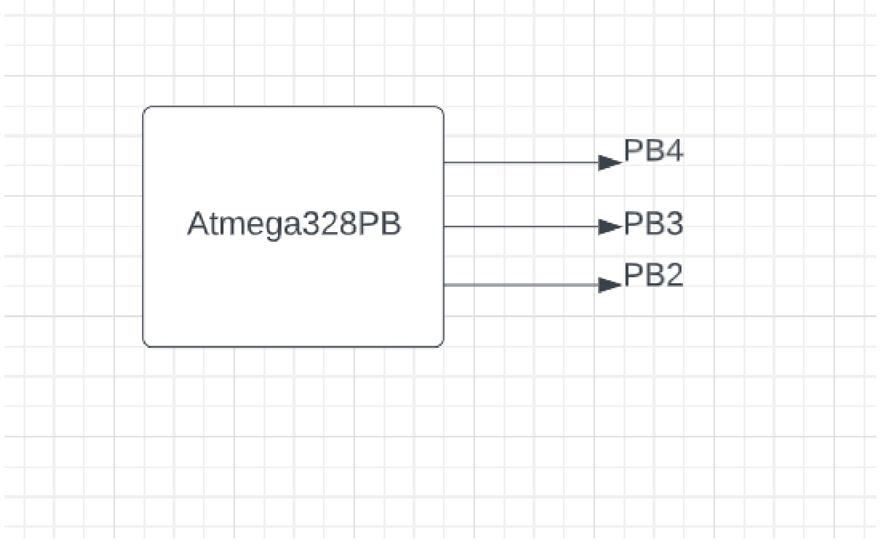
Student Name: David Lenzin
Student #: 2001654470
Student Email: lenzin@unlv.nevada.edu
Primary Github address: https://github.com/dlenzin15/submissions
Directory: submissions/DA3

Submit the following for all Labs:

1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder ESD301/DA, sub-folder named LABXX, with one document and one video link file for each lab, place modified asm/c files named as LabXX-TYY.asm/c.
3. If multiple asm/c files or other libraries are used, create a folder LabXX-TYY and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) and other include files, c) text file with youtube video links (see template).

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

Atmega328PB board was used. Pins PB2, PB3, and PB4 were used. See schematic for pinout.



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

```c
/*
 * GccApplication1.c
 *
 * Created: 3/10/2023 3:26:15 PM
 * Author : david
 */

#define F_CPU 16000000UL  //defines clock freq = 16 MHz
#include "util/delay.h"
#include <avr/io.h>

#define DELAY_TCNT_VALUE 20        //TCNT = ((16 MHz / 256)*(1/3000)) - 1 = 19.83. Rounding up to 20.
#define DELAY_COUNTER 3125         // Timer delay is 0.320 ms. 1 second / 0.320 ms = 3125

int main(void)
{
       DDRB |= (1<<4);             //Set PB4 to an output
       PORTB |= (1<<4);            // Turn LED off initially
       TCCR0A = 0;                 // Set timer to Normal mode
       TCCR0B |= (1 << CS02);      // set prescalar to 256
       TCNT0 = 0x00;               //Initialize the TCNT register to start the timer

       int overflows = 0;          //Counter to track the how many times the timer overflows
       while (1)
    {
              if (TCNT0 == DELAY_TCNT_VALUE) {
                     overflows++;  //If the TCNT has overflown, increment the counter

             //If the timer has overflown enough times, reset the counter and toggle the LED
              if (overflows >= DELAY_COUNTER) {
```

```
                                overflows = 0;
                                PORTB ^= (1<<4);
                        }
                        TCNT0 = 0;                  //Reset the TCNT register
                }
    }
        return 0;
}
```

3. DEVELOPED MODIFIED CODE OF TASK 1B

```c
/*
 * DA3_Task1B.c
 *
 * Created: 3/10/2023 5:15:19 PM
 * Author : david
 */

#define F_CPU 16000000UL  //defines clock freq = 16 MHz
#include "util/delay.h"
#include <avr/io.h>
#include <avr/interrupt.h>

/************************ Calculations *************************************/
/* Top value: ((16 MHz / 64) * 0.999 ms) - 1 = 248.75. Rounding up to 249   */
/* TCNT1 Value: (2^16 - 1) = 65535. 65535 - 249 = 65286, which is 0xFF06    */
/****************************************************************************/

#define TCNT1L_VALUE 0x06
#define TCNT1H_VALUE 0xFF

ISR(TIMER1_OVF_vect)
{
	static int overflows = 0;//Counter to track to overflows
	overflows++;
	if (overflows >= 1000)		//1000 overflows is approximately 1 second
	{
		PORTB ^= (1<<3);		//Toggle the LED
		overflows = 0;			//Reset counter
	}
	
	TCNT1L = TCNT1L_VALUE;
	TCNT1H = TCNT1H_VALUE;	//Load the TCNT1 Register with 0xFF06
	
}

int main(void)
{
	DDRB |= (1<<3);	//Set PB5 as an output
	PORTB |= (1<<3);	//Turn off LED initially
	
	TCCR1B |= (1 << CS11) | (1 << CS10); //Set prescalar to 64. Mode is normal by default
	TCNT1L = TCNT1L_VALUE;
	TCNT1H = TCNT1H_VALUE;	//Load the TCNT1 Register with 0xFF06
	
	TIMSK1 |= (1 << TOIE1); //Enable Timer1 overflow interrupt
	
	sei();		//Enable global interrupts
	
    while (1);     //Wait for interrupts

return 0;
}
```

4. DEVELOPED MODIFIED CODE OF TASK 1C:

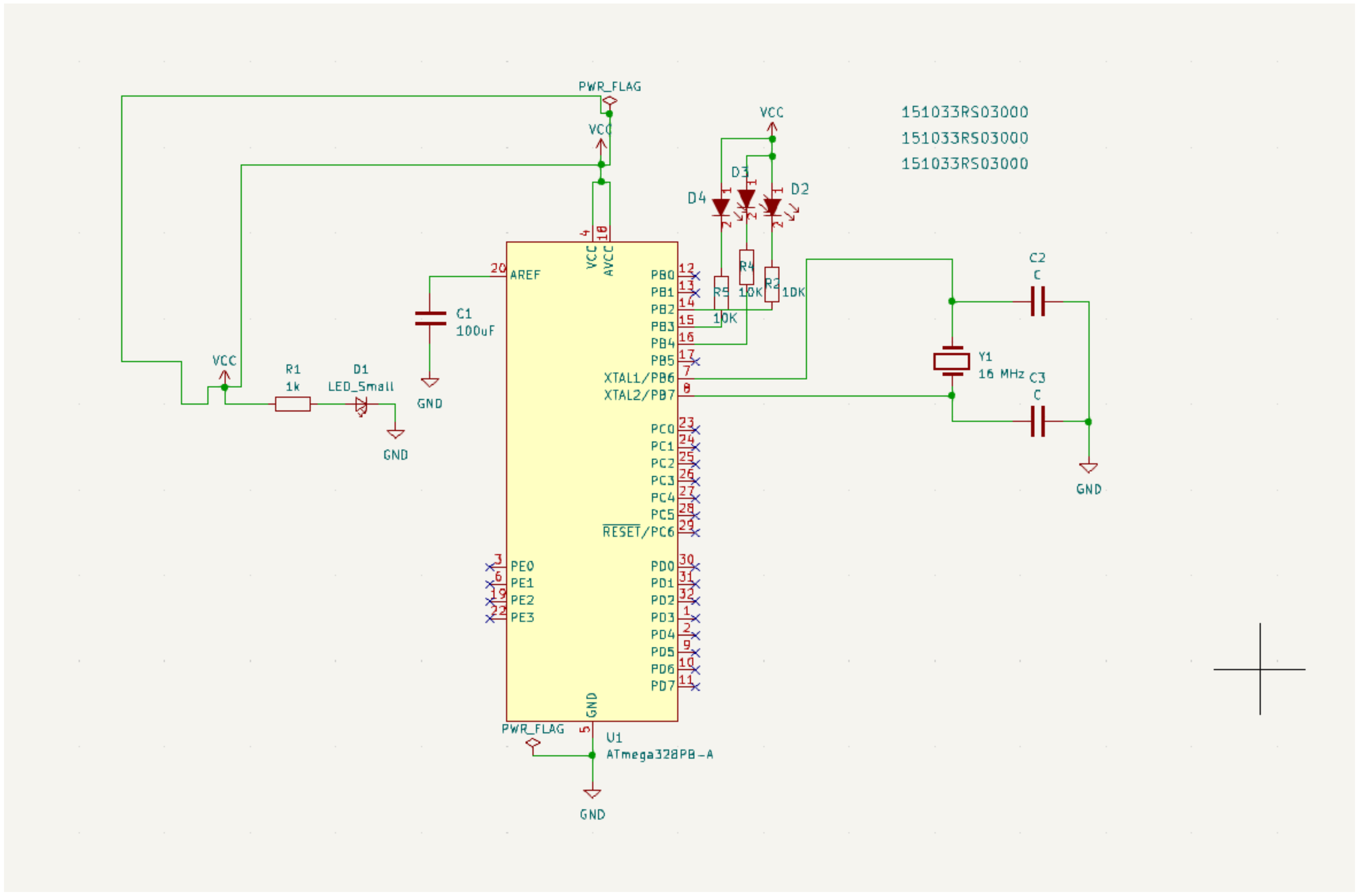
```c
/*
 * DA3_Task1C.c
 *
 * Created: 3/10/2023 6:07:02 PM
 * Author : david
 */

#define F_CPU 16000000UL  //defines clock freq = 16 MHz
#include "util/delay.h"
#include <avr/io.h>
#include <avr/interrupt.h>

/******************Calculations:*****************************/
/* 2/3000 = 0.666 ms. Target frequency = 3000/2 = 1500 Hz     */
/*                                                            */
/* OCR2A = (16 MHz / (2*64*1500)) = 83.3 Rounding down to 83  */
/*                                                            */
/*  However, this generates a pulse every 0.333 ms to make    */
/*     a period of 0.666 ms.                                  */
/*                                                            */
/* 83*2 = 166.                                                */
/************************************************************/

#define OCR2A_VALUE 166
#define OVERFLOW_MAX 2001   // (1.333s / 0.666 ms) = 2001.5

ISR(TIMER2_COMPA_vect)
{
       static int overflows = 0;   //Counter to track overflows
       overflows++;                //Increment the counter everytime the interrupt is triggered
       if (overflows == OVERFLOW_MAX) {
              PORTB ^= (1<<2);     //Toggle the LED
              overflows = 0;       //Reset the counter
       }
}

int main(void)
{
       DDRB |= (1<<2);             //Set PB2 to an output
       PORTB |= (1<<2);            //Initially turn LED off

       TCCR2A |= (1 << WGM21);     //Set Timer1 to CTC mode
       TCCR2B |= (1 << CS22);      //Set prescalar to 64
       OCR2A = OCR2A_VALUE;        //Load compare register value

       TIMSK2 |= (1 << OCIE2A);    //Set interrupt on compare match
       sei();                      // enable interrupts

    while (1);                     //Wait for interrupts
       return 0;
}
```

5. DA3 Complete Code:

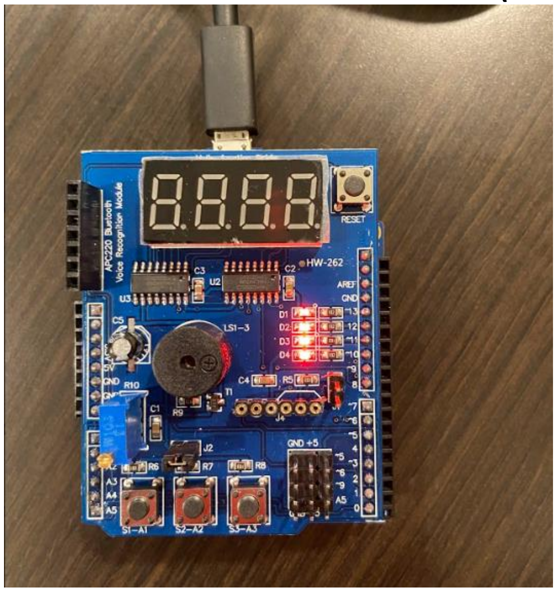
```c
/*
 * DA3_Complete.c
 *
 * Created: 3/14/2023 6:29:49 PM
 * Author : david
 */

#define F_CPU 16000000UL  //defines clock freq = 16 MHz
#include "util/delay.h"
#include <avr/io.h>
#include <avr/interrupt.h>

#define DELAY_TCNT_VALUE 20          //TCNT = ((16 MHz / 256)*(1/3000)) - 1 = 19.83. Rounding up to 20. Results in a
delay of 0.320 ms
#define DELAY_COUNTER 3125           // Timer delay is 0.320 ms. 1 second / 0.320 ms = 3125

#define TCNT1L_VALUE 0x06
#define TCNT1H_VALUE 0xFF

#define OCR2A_VALUE 166
#define OVERFLOW_MAX 2001    // (1.333s / 0.666 ms) = 2001.5


ISR(TIMER1_OVF_vect)
{
	static int overflows = 0;    //Counter to track to overflows
	overflows++;
	if (overflows >= 1000)               //1000 overflows is approximately 1 second
	{
		PORTB ^= (1<<3);             //Toggle the LED
		overflows = 0;               //Reset counter
	}

	TCNT1L = TCNT1L_VALUE;
	TCNT1H = TCNT1H_VALUE;       //Load the TCNT1 Register with 0xFF06

}

ISR(TIMER2_COMPA_vect)
{
	static int overflows = 0;    //Counter to track overflows
	overflows++;                         //Increment the counter everytime the interrupt is triggered
	if (overflows == OVERFLOW_MAX) {
		PORTB ^= (1<<2);     //Toggle the LED
		overflows = 0;       //Reset the counter
	}
}


int main(void)
{
	//Initialize Ports:
	DDRB |= (1<<4) | (1<<3) | (1<<2);    //Set PB4, PB3, and PB2 to output
	PORTB |= (1<<4) | (1<<3) | (1<<2);   // Turn LEDs off initially

	//Initialize Timers
	//Timer 0
	TCCR0A = 0;                          // Set timer to Normal mode
	TCCR0B |= (1 << CS02);       // set prescalar to 256
	TCNT0 = 0x00;                //Initialize the TCNT register to start the timer
```

```
       //Timer 1
       TCCR1B |= (1 << CS11) | (1 << CS10); //Set prescalar to 64. Mode is normal by default
       TCNT1L = TCNT1L_VALUE;
       TCNT1H = TCNT1H_VALUE;       //Load the TCNT1 Register with 0xFF06

       //Timer 2
       TCCR2A |= (1 << WGM21);      //Set Timer1 to CTC mode
       TCCR2B |= (1 << CS22);       //Set prescalar to 64
       OCR2A = OCR2A_VALUE;  //Load compare register value

       //Initialize interrupts
       TIMSK1 |= (1 << TOIE1);              //Enable Timer1 overflow interrupt
       TIMSK2 |= (1 << OCIE2A);     //Set interrupt on compare match
       sei();                                      // enable interrupts

       int overflows = 0;           //Counter to track the how many times the timer overflows
       while (1)
       {
              if (TCNT0 == DELAY_TCNT_VALUE) {
                     overflows++;   //If the TCNT has overflown, increment the counter
                     if (overflows >= DELAY_COUNTER) {    //If the timer has overflown enough times, reset the counter
and toggle the LED
                            overflows = 0;
                            PORTB ^= (1<<4);
                     }
                     TCNT0 = 0;                    //Reset the TCNT register
              }
       }
       return 0;
}
```

6. SCHEMATICS



7. SCREENSHOT OF EACH DEMO (BOARD SETUP)



- The board uses no external hardware in this design assignment, so the output of all three tasks is to blink the D2, D3, and D4 LEDs.

8. VIDEO LINKS OF EACH DEMO

https://youtu.be/zyBwMItQe3I

9. GITHUB LINK OF THIS DA

https://github.com/dlenzin15/submissions/tree/main/DA3

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

"This assignment submission is my own, original work".
David Lenzin