CPE301 – FALL 2019

Design Assignment 2C

Student Name: Cody Jones

Student #: 5002863328

Student Email: Jonesc30@unlv.nevada.edu

Primary Github address: https://github.com/Jonesc30/Submission

Directory: Submission

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**

List of Components used

Block diagram with pins used in the Atmega328P

1. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1**

#define F\_CPU 16000000UL

#include <avr/io.h>

int main(void)

{

DDRB = 0xFF; //all registers in DDRB are outputs

DDRC = 0x00; //all registers in DDRC are inputs

PORTB = 0xFF; //turn all LEDs on

TCCR0A = 0x00; //normal operation

TCNT0 = 0x00; //start the timer

TCCR0B |= (1 << CS02) | (1 << CS00); //set prescaler to 1024

int count = 0; //use this to count overflow

//The fout for this is about 60Hz

while (1)

{

count = 0; // reset counter

TCNT0 = 0x00; // reset timer

while (count < 23) // loop until 60% DC is met

{

if(TCNT0 == 0xFF) // if timer overflows

{

count++; // increment counter

TCNT0 = 0x00; // reset timer

}

else if(!(PINC & (1 << 3))) //check if SW3 is pushed

{

PORTB |= (1 << 3); //turn off LED @ PB3

PORTB &= ~(1 << 2); //turn on LED @ PB2

TCNT0 = 0x00; //reset timer if switch is pushed

count = 0;

while(count < 82) //1.333 seconds

{

if(TCNT0 == 0xFF)

{

count++;

TCNT0 = 0x00;

}

}

PORTB |= (1 << 2); //turn off LED @ PB2

}

}

PORTB ^= (1<<3); // toggle LED off

count = 0; // reset counter

TCNT0 = 0x00; // reset timer

while (count < 15) // loop until remaining 40% DC is met

{

if(TCNT0 == 0xFF) // if timer overflows

{

count++; // increment counter

TCNT0 = 0x00; // reset timer

}

else if(!(PINC & (1 << 3))) //check if SW3 is pushed

{

PORTB |= (1 << 3); //turn off LED @ PB3

PORTB &= ~(1 << 2); //turn on LED @ PB2

TCNT0 = 0x00; //reset timer if switch is pushed

count = 0;

while(count < 82) //1.333 seconds

{

if(TCNT0 == 0xFF)

{

count++;

TCNT0 = 0x00;

}

}

PORTB |= (1 << 2); //turn off LED @ PB2

}

}

PORTB ^= (1<<3); // toggle LED on

}

return 0;

}

**INITIAL/MODIFIED/DEVELOPED CODE OF TASK 2**

#define F\_CPU 16000000UL

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/delay.h>

uint8\_t OVFCOUNT = 0; // overflow count multiplier

uint8\_t OVFLIMIT = 16; // overflow count limit, set for 40% of DC

int main(void)

{

DDRB = 0xFF; //all registers in DDRB are outputs

DDRC = 0x00; //all registers in DDRC are inputs

PORTB = 0xFF; //turn all LEDs on

TCCR0A = 0x00; //normal operation

TCNT0 = 12; //start the timer for .625 s, 40 clock cycles

TCCR0B |= (1 << CS02) | (1 << CS00); //set prescaler to 1024

TIMSK0 = (1<<TOIE0); //enable timer0 overflow interrupt

sei(); //enable interrupts

while (1)

{

}

}

ISR (TIMER0\_OVF\_vect)

{

OVFCOUNT++; // increment counter

if (OVFCOUNT == OVFLIMIT)

{

PORTB ^= (1<<3); // toggle LED

PORTB |=(1<<2); // LED off from switch

if(OVFLIMIT == 16)

{

OVFLIMIT = 24; // set ovf limit for 60% of DC

if(!(PINC & (1 << 3))) //check if SW3 is pushed

{

PORTB &= ~(1<<2); //LED on

OVFLIMIT = 85; //when switch is pressed it goes through 85 cycles for 1.3333 sec

PORTB |= (1<<3); //LED off

}

}

else

{

OVFLIMIT = 16; // set ovf limit for 40% of DC

if(!(PINC & (1 << 3))) //check if SW3 is pushed

{

PORTB &= ~(1<<2); //LED on

OVFLIMIT = 85; //when switch is pressed it goes through 85 cycles for 1.3333 sec

PORTB |= (1<<3); //LED off

}

}

OVFCOUNT = 0; // reset counter

}

TCNT0 = 12; // timer value for .625 sec

}

**INITIAL/MODIFIED/DEVELOPED CODE OF TASK 3**

#define F\_CPU 16000000UL

#include <avr/io.h>

#include <avr/interrupt.h>

#include <util/delay.h>

uint8\_t OVFCOUNT = 0; // overflow count multiplier

uint8\_t OVFLIMIT = 16; // overflow count limit, set for 40% of DC

int main(void)

{

DDRB = 0xFF; //all registers in DDRB are outputs

DDRC = 0x00; //all registers in DDRC are inputs

PORTB = 0xFF; //turn all LEDs on

OCR0A = 244; //TOP=256-12

TCNT0 = 0; //start the timer

TCCR0A |= (1<<WGM01);

TCCR0B |= (1 << CS02) | (1 << CS00); //set prescaler to 1024, CTC mode

TIMSK0 |= (1<<OCIE0A); //enable timer0 overflow interrupt

sei(); //enable interrupts

while (1)

{

}

}

ISR (TIMER0\_COMPA\_vect)

{

OVFCOUNT++; // increment counter

if (OVFCOUNT == OVFLIMIT)

{

PORTB ^= (1<<3); // toggle LED

PORTB |=(1<<2); // LED off from switch

if(OVFLIMIT == 16)

{

OVFLIMIT = 24; // set ovf limit for 60% of DC

if(!(PINC & (1 << 3))) //check if SW3 is pushed

{

PORTB &= ~(1<<2); //LED on

OVFLIMIT = 85; //when switch is pressed it goes through 85 cycles for 1.3333 sec

PORTB |= (1<<3); //LED off

}

}

else

{

OVFLIMIT = 16; // set ovf limit for 40% of DC

if(!(PINC & (1 << 3))) //check if SW3 is pushed

{

PORTB &= ~(1<<2); //LED on

OVFLIMIT = 85; //when switch is pressed it goes through 85 cycles for 1.3333 sec

PORTB |= (1<<3); //LED off

}

}

OVFCOUNT = 0; // reset counter

}

TCNT0 = 0; // timer value

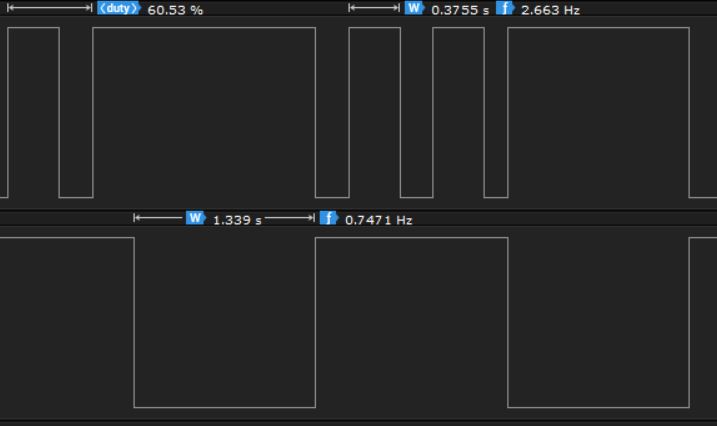
}

1. **SCHEMATICS**

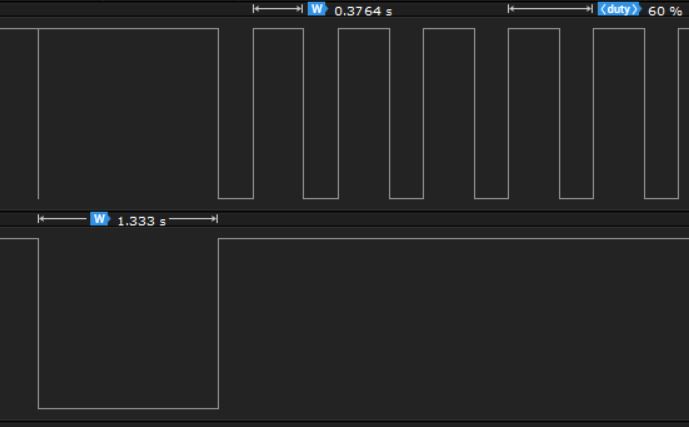
Use fritzing.org

1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

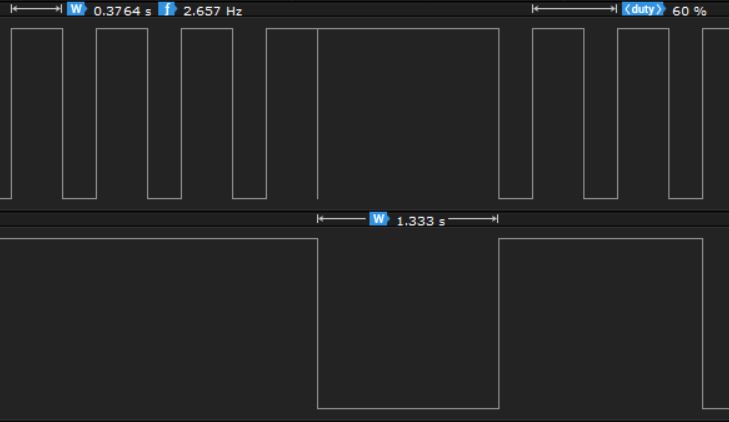
Task 1



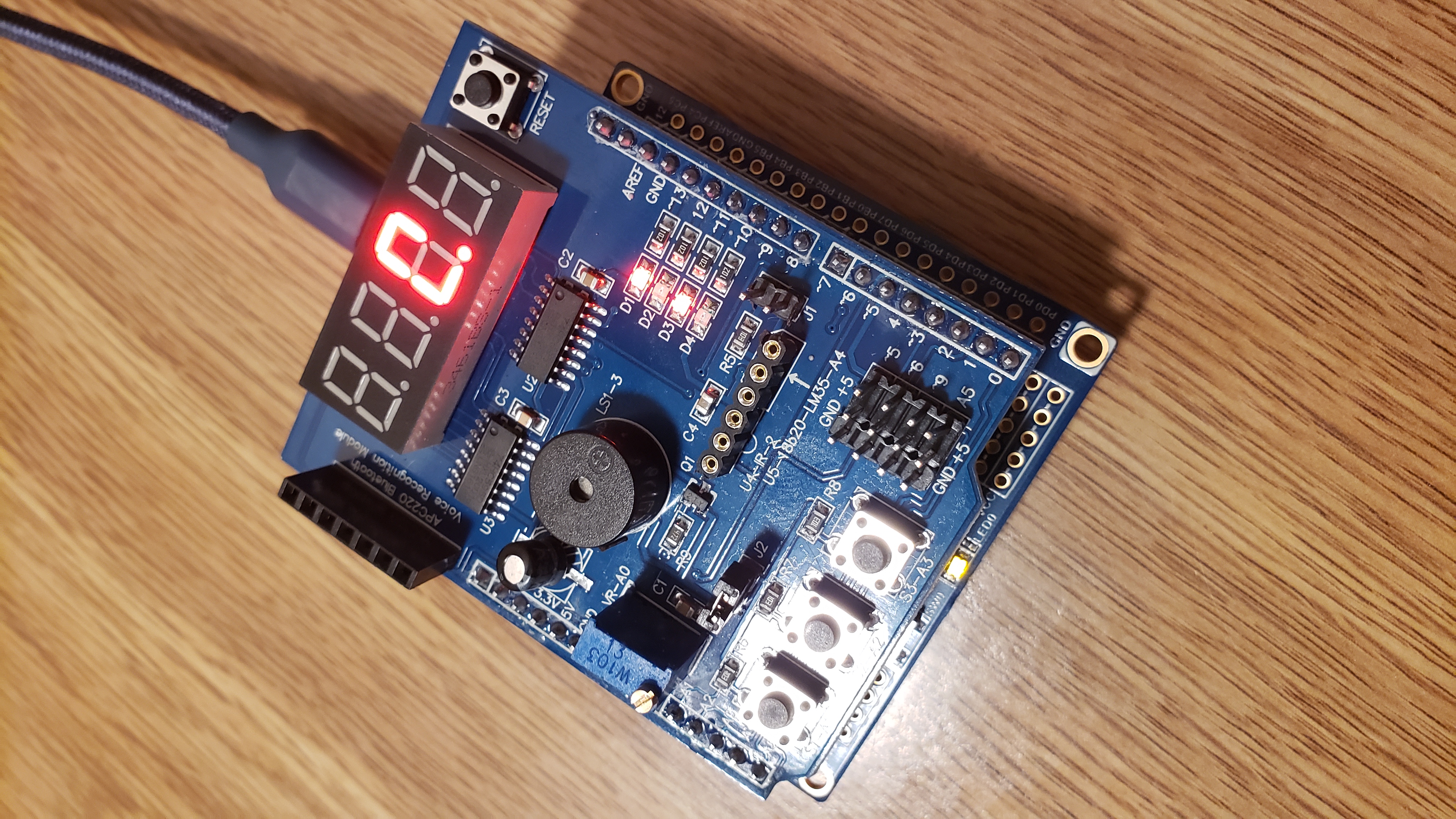
Task 2



Task 3



1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**



1. **VIDEO LINKS OF EACH DEMO**

Task 1: <https://www.youtube.com/watch?v=Tqrw04tKO8w>

Task 2: <https://www.youtube.com/watch?v=E3nyYXMj7pA>

Task 3: <https://www.youtube.com/watch?v=z52RTvPqKpg>

1. **GITHUB LINK OF THIS DA**

<https://github.com/Jonesc30/Submission/tree/master/DesignAssignments>

**Student Academic Misconduct Policy**

<http://studentconduct.unlv.edu/misconduct/policy.html>

“This assignment submission is my own, original work”.

NAME OF THE STUDENT