CPE301 – Fall 2019

Design Assignment 2C

Student Name: Jacob Patrick Reed

Student #: 1008448895

Student Email: reedj35@unlv.nevada.edu

Primary Github address: <https://github.com/reedjacobp>

Directory: <https://github.com/reedjacobp/submission_da>

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 Xplained Mini

Multi-function shield

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

/\*

\* DA2C\_Task1.c

\*

\* Created: 10/9/2019 10:56:08 PM

\* Author : jreed

\*/

#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 on all LEDs

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 61Hz

while(1)

{

count = 0;

TCNT0 = 0x00; //start the timer

while(count < 23) //60% off

{

if(TCNT0 == 0xFF) //check to see if TOP(255) has been reached

{

count++; //upon overflow, count it, then reset timer

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); //once overflow has been counted 23 times, turn off LED by XOR

count = 0; //reset the count

TCNT0 = 0x00; //since timer is constantly incrementing, I must reset right before a loop uses it

while(count < 15) //40% on

{

if(TCNT0 == 0xFF) //check to see if TOP(255) has been reached

{

count++; //upon overflow, count it, then reset timer

TCNT0 = 0x00; //reset timer

}

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

{

PORTB |= (1 << 3);

PORTB &= ~(1 << 2);

count = 0;

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

while(count < 82) //1.333 seconds

{

if(TCNT0 == 0xFF)

{

count++;

TCNT0 = 0x00;

}

}

PORTB |= (1 << 2);

}

}

PORTB ^= (1 << 3); //once overflow has been counted 12 times, turn on LED by XOR

}

return 0;

}

1. **DEVELOPED MODIFIED CODE OF TASK 2**

/\*

\* DA2C\_Task2.c

\*

\* Created: 10/10/2019 10:33:17 PM

\* Author : jreed

\*/

#define F\_CPU 16000000UL

#include <avr/io.h>

#include <avr/interrupt.h>

uint8\_t OVF\_COUNT = 0;

uint8\_t OVF\_LIMIT = 16;

int main(void)

{

//int count = 0;

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

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

PORTB = 0xFF; //turn on all LEDs

TCCR0A = 0x00; //normal operation

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

TCNT0 = 12;

TIMSK0 |= (1 << TOIE0);

sei();

while (1)

{

}

}

ISR (TIMER0\_OVF\_vect)

{

OVF\_COUNT++;

if (OVF\_COUNT == OVF\_LIMIT)

{

PORTB ^= (1 << 3);

PORTB |= (1 << 2);

if(OVF\_LIMIT == 16)

{

OVF\_LIMIT = 24;

}

if(!(PINC & (1 << 3)))

{

OVF\_LIMIT = 85;

PORTB |= (1 << 3);

PORTB &= ~(1 << 2);

}

else

{

OVF\_LIMIT = 16;

}

OVF\_COUNT = 0;

}

TCNT0 = 12;

}

1. **DEVELOPED MODIFIED CODE OF TASK 3**

/\*

\* DA2C\_Task3.c

\*

\* Created: 10/11/2019 12:39:36 AM

\* Author : jreed

\*/

#define F\_CPU 16000000UL

#include <avr/io.h>

#include <avr/interrupt.h>

uint8\_t OVF\_COUNT = 0;

uint8\_t OVF\_LIMIT = 16;

int main(void)

{

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

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

PORTB = 0xFF; //turn on all LEDs

OCR0A = 244; //normal operation

TCNT0 = 0;

TCCR0A |= (1 << WGM01) | (1 << COM1A0);

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

TIMSK0 |= (1 << OCIE0A);

sei();

while (1)

{

}

}

ISR (TIMER0\_COMPA\_vect)

{

OVF\_COUNT++;

if (OVF\_COUNT == OVF\_LIMIT)

{

PORTB ^= (1 << 3);

PORTB |= (1 << 2);

if(OVF\_LIMIT == 16)

{

OVF\_LIMIT = 24;

if(!(PINC & (1 << 3)))

{

OVF\_LIMIT = 85;

PORTB |= (1 << 3);

PORTB &= ~(1 << 2);

}

}

else

{

OVF\_LIMIT = 16;

if(!(PINC & (1 << 3)))

{

OVF\_LIMIT = 85;

PORTB |= (1 << 3);

PORTB &= ~(1 << 2);

}

}

OVF\_COUNT = 0;

}

TCNT0 = 0;

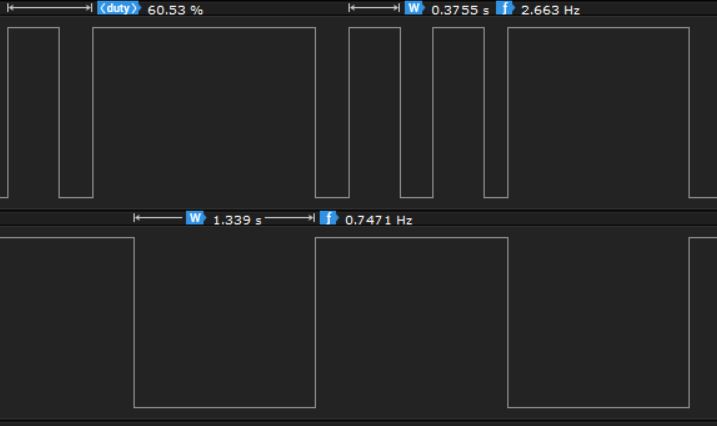
}

1. **SCHEMATICS**

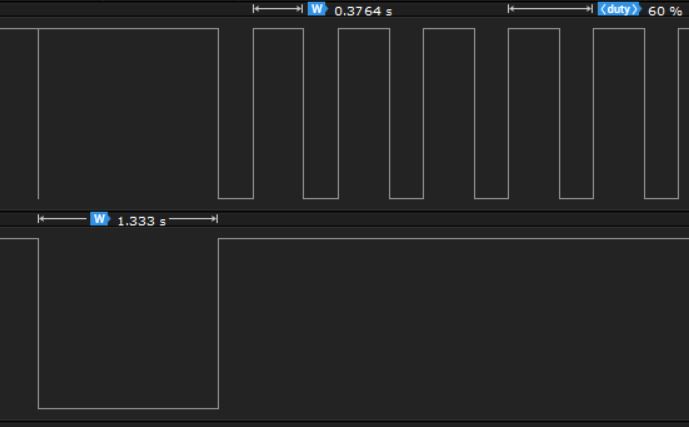
N/A

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

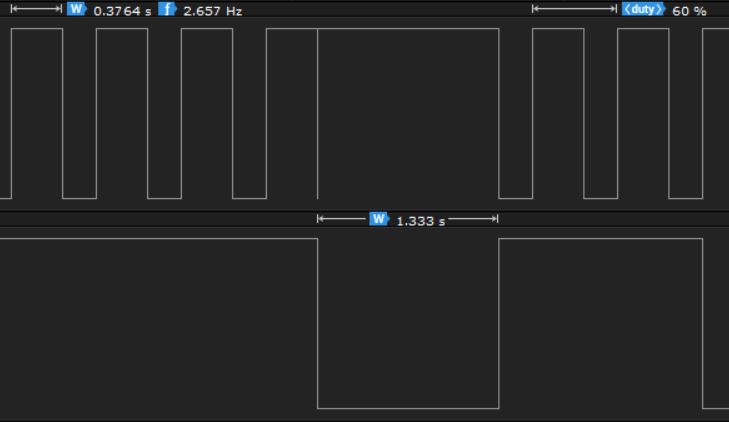
**Task 1 Waveform Screenshot**



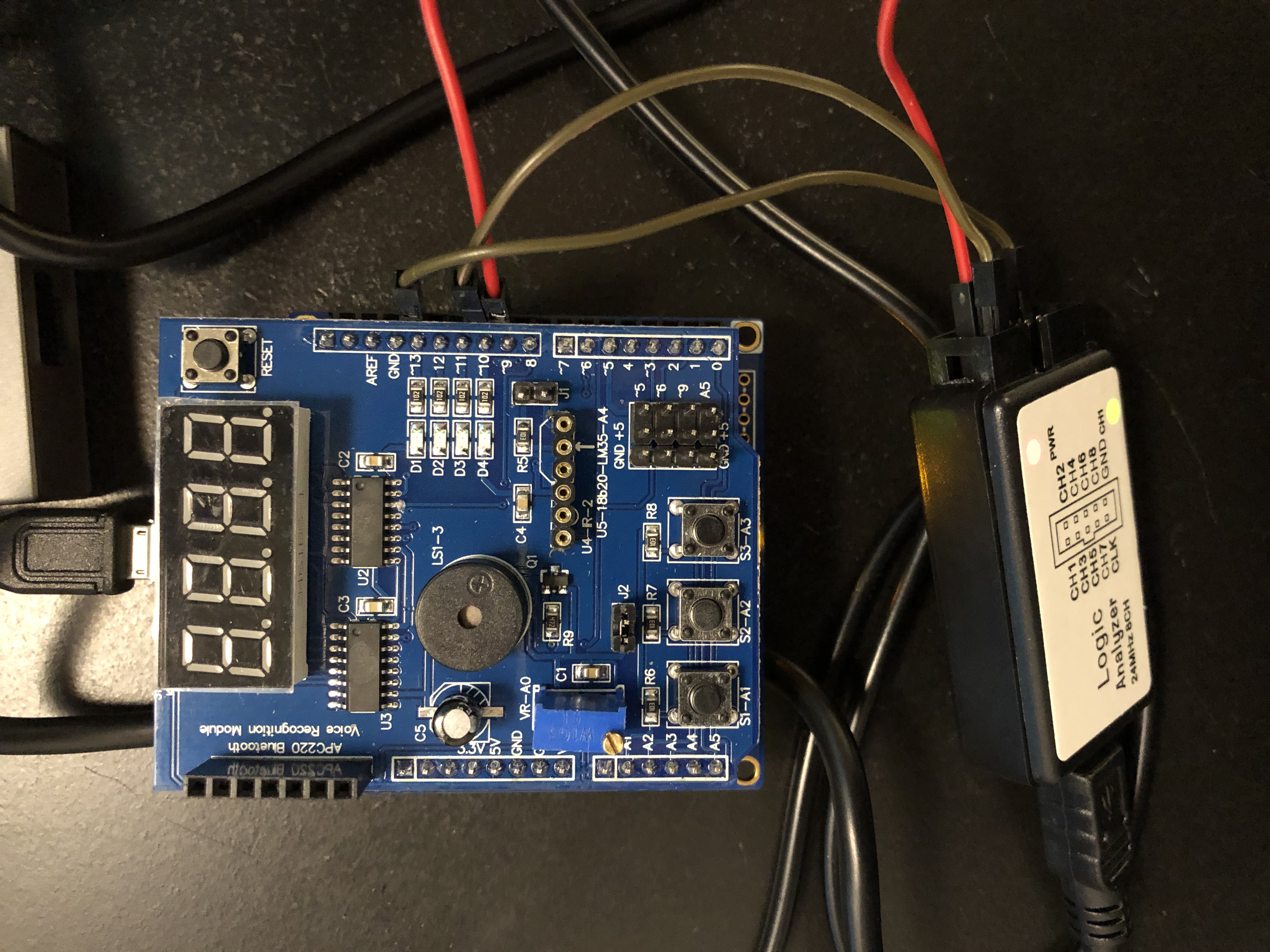
**Task 2 Waveform Screenshot**



**Task 3 Waveform Screenshot**



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



1. **VIDEO LINKS OF EACH DEMO**

**Task 1** : <https://youtu.be/oHMrpV3Dsxo>

**Task 2** : <https://youtu.be/qoYsYyKb3F0>

**Task 3** : https://youtu.be/V4uJRaN4pp0

1. **GITHUB LINK OF THIS DA**

<https://github.com/reedjacobp/submission_da/tree/master/DesignAssignments/DA2C>

**Student Academic Misconduct Policy**

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

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

Jacob Reed