CPE301 – FALL 2019

Design Assignment 2C

Student Name: Andrew Buchanan

Student #:5003154346

Student Email: buchaa2@unlv.nevada.edu

Primary Github address: <https://github.com/buchaa2/103EPC>

Directory:

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

//DA2A task1

#define *F\_CPU* 16000000UL

#include <avr/io.h>

int main(void)

{

DDRB =0xFF; // set PORTB registers as outputs for output (LED)

DDRC =0x00; // set PORTC registers as inputs for input switches

PORTB =0xFF; // turn all the LEDs in B on

TCCR0A = 0; // normal operation

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

int count = 0; // overflow counter

while (1) {

count = 0; // reset counter

TCNT0 = 0x00; // reset timer

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

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

count++; // increment counter

TCNT0 = 0x00; // reset timer

}

}

PORTB =0x00 ; // toggle LED off

count = 0; // reset counter

TCNT0 = 0x00; // reset timer

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

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

count++; // increment counter

TCNT0 = 0x00; // reset timer

}

}

PORTB =0xFF; // toggle LED on

}

}

-----------------------------------------------------------------------------------------

//DA2A task2

#define *F\_CPU* 16000000UL

#include <avr/io.h>

int main(void)

{

DDRC &= (0<<3); // connect PORTC.3 to switch as input

PORTC |= (1<<3); // enable pull-up

DDRB = 0xFF; // set PORTB.2 for output (LED)

TCCR0A = 0; // normal operation

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

int count = 0; // overflow counter

while (1) {

if(!(PINC & (1<<PINC3))) { // check if pin C 3 is low

PORTB =0x00; // set LED on

count = 0; // reset counter

TCNT0 = 0x00; // reset timer

while(count < 82) { // loop until 1.33 sec delay met

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

count++; // increment counter

TCNT0 = 0x00; // reset timer

}

}

}

else {

PORTB = 0xFF; // set LED off

}

}

return 0;

}

-----------------------------------------------------------------------------------------

//DA2A task 1

#include <avr/io.h>

#include <stdio.h>

#include <avr/interrupt.h>

//declare vital variables

int ovFlowCount = 0;

int dc60 = 23; //overflow count to reach is 23

int dc40 = 15; //overflow count to reach is 15

int flag = 1;

int main(void)

{

DDRB |= (1<<2)|(1<<5); // Set all PORTB out

PORTB |= (1<<5); // Set PORTB

TCCR0B |= (1<<CS00)|(1<<CS02); // Set pre-scaler to 1024

TIMSK0 |= (1<<TOIE0); //enable timer interrupts

sei(); //enable interrupts

TCNT0 = 0; //initialize clock

while(1);

}

ISR(TIMER0\_OVF\_vect){

ovFlowCount ++;

if (flag == 1 && ovFlowCount == dc40){

PORTB &= ~(1<<2);

ovFlowCount = 0; //reset overflow counter

flag = 0;

}

else if (flag == 0 && ovFlowCount == dc60){

PORTB |= (1<<2);

ovFlowCount = 0; //reset overflow counter

flag = 1;

}

}

-----------------------------------------------------------------------------------------

/DA2A task2

#include <avr/io.h>

#include <stdio.h>

#include <avr/interrupt.h>

int flag = 0;

//FOR 1.333s Overflow count = 82

int main(void){

//initialize registers

DDRB |= (1<<2)|(1<<5); //set portb 2 to output

PORTB |= (1<<2)|(1<<5); //set pinb2 to high

DDRC &= (0<<2); //set ddrc to input

PORTC |= (0<<2); //set portc2 to low

TCCR0B |= (1<<CS00)|(1<<CS02); // Set pre-scaler to 1024

//int flag = 0;

TIMSK0 |= (1<<TOIE0); //enable timer interrupt

sei(); //enable interrupt

while (1) {

if (!(PINC & (1 << PINC2))) //check for button press

{

flag = 1; //if pressed then turn on led

}

else {

PORTB |= (1<<2); //if not pressed then keep led off

flag = 0; //keep flag off to keep from false positives

}

}

return 0;

}

ISR(TIMER0\_OVF\_vect){

int ovFlowCount = 0;

int ovCheck = 0;

int delayCount = 82; //amount of overflows to reach desired delay

if (flag == 1){

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

TCNT0 = 0; //start counter at 0

while(ovFlowCount <= delayCount ){

ovCheck = TIFR0 & 0x01; //variable to check if overflow met.

if (ovCheck == 1){

ovFlowCount ++; //increase overflow count

TIFR0 |= (1<<TOV0); //reset sreg flag

}

}

ovFlowCount = 0;

PORTB |= (1<<2);

flag = 0;

}

}

-----------------------------------------------------------------------------------------

//DA2A task1

#include <avr/io.h>

#include <stdio.h>

#include <avr/interrupt.h>

//declare vital variables

int matchCount = 0;

int dc60 = 23; //duty cycle of 60%

int dc40 = 15; //duty cycle of 40%

int flag = 1; //flag used to toggle between duty cycles

int cycleMatch = 244; //compare to 244 and clear each time

int main(void)

{

DDRB |= (1<<2)|(1<<5); // Set all PORTB out

PORTB |= (1<<5); //turn off LED 5

TCCR0A |= (1<<WGM01); //set ctc mode on OCRA

TCCR0B |= (1<<CS00)|(1<<CS02); // Set pre-scaler to 1024

TIMSK0 |= (1<<OCIE0A); //enable compare interrupts

OCR0A = cycleMatch;

sei(); //enable interrupts

while(1); //infinite loop

}

ISR(TIMER0\_COMPA\_vect){

matchCount ++;

if (flag == 1 && matchCount == dc60)//if the flag and the matchCount equals the cycle for 60%

{

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

matchCount = 0; //reset overflow counter

flag = 0; //set the flag for the second portion of the wave.

}

else if (flag == 0 && matchCount == dc40)//if the flag and the matchCount equals the cycles for 40%

{

PORTB |= (1<<2); //toggle LED

matchCount = 0; //reset overflow counter

flag = 1;

}

}

-----------------------------------------------------------------------------------------

//DA2A task2

#include <avr/io.h>

#include <stdio.h>

#include <avr/interrupt.h>

//set global variables

int flag = 0;

int matchCount = 0;

int cycleMatch = 244;

int main(void){

//initialize registers

DDRB |= (1<<2)|(1<<5); //set portb 2 to output

PORTB |= (1<<2)|(1<<5); //set pb2 to high

DDRC &= (0<<2); //set ddrc to input

PORTC |= (0<<2); //set portc2 to low

//Set modes

TCCR0A |= (1<<WGM01); //set ctc mode on OCRA

TCCR0B |= (1<<CS00)|(1<<CS02); // Set pre-scaler to 1024

TIMSK0 |= (1<<OCIE0A); //enable compare interrupts

OCR0A = cycleMatch;

sei(); //enable interrupt

while (1) {

if (!(PINC & (1 << PINC2))) //check for button press

{

flag = 1; //if pressed then turn on led

}

else {

PORTB |= (1<<2); //if not pressed then keep led off

flag = 0;

}

}

return 0;

}

ISR(TIMER0\_COMPA\_vect){

int ovFlowCount = 0;

int matchCheck = 0;

int delayCount = 85; //amount of overflows to reach desired delay

if (flag == 1)

{

flag = 0; //toggle flag

PORTB &= ~(1<<2);

TCNT0 = 0; //start counter at 0

while(matchCount <= delayCount )

{

matchCheck = TIFR0 & 0x02; //variable to check if compare is met

if (matchCheck == 0x02){

matchCount ++; //increase match count

TIFR0 |= (1<<OCF0A); //reset sreg flag

}

}

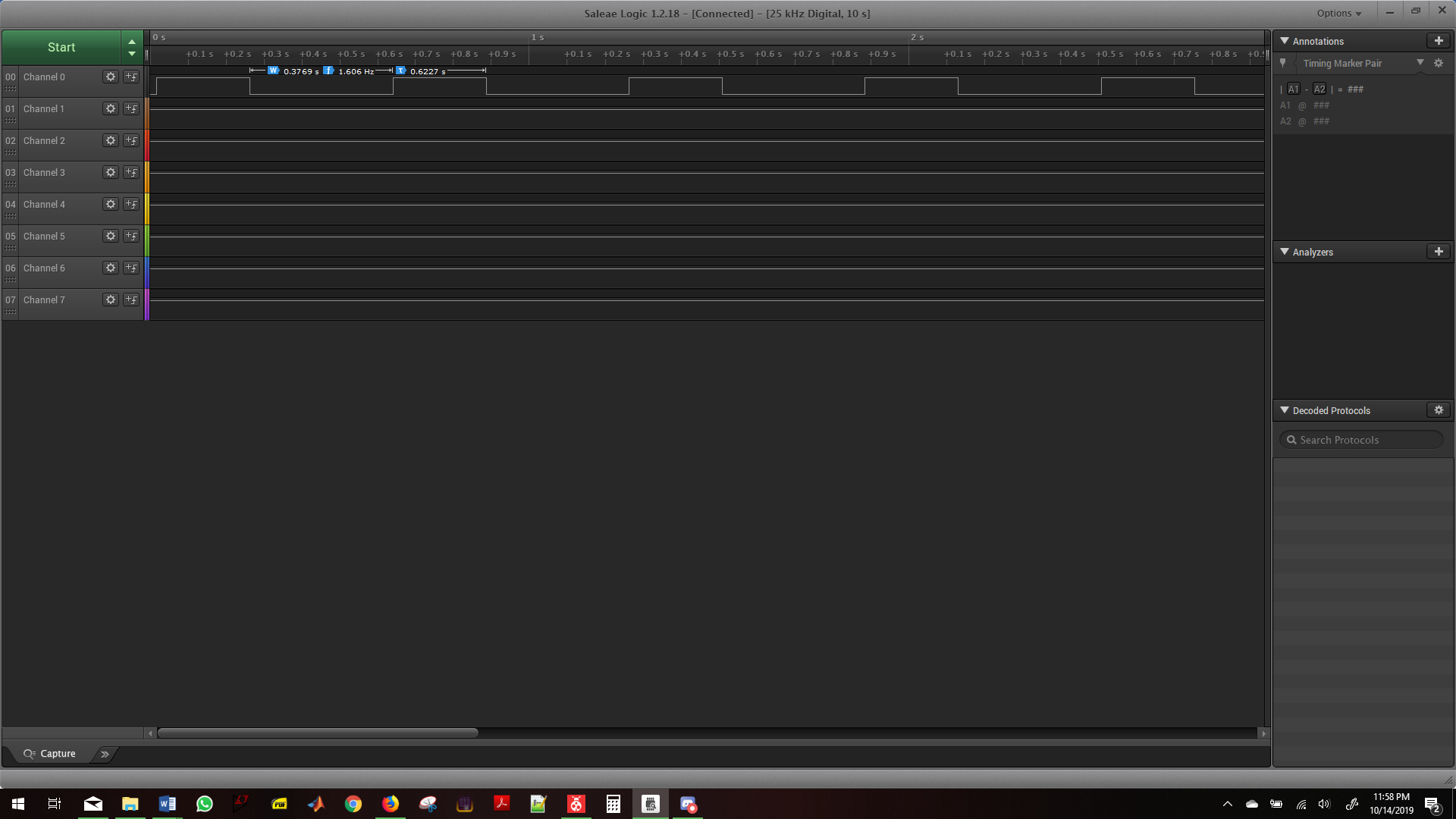
matchCount = 0; //reset match count

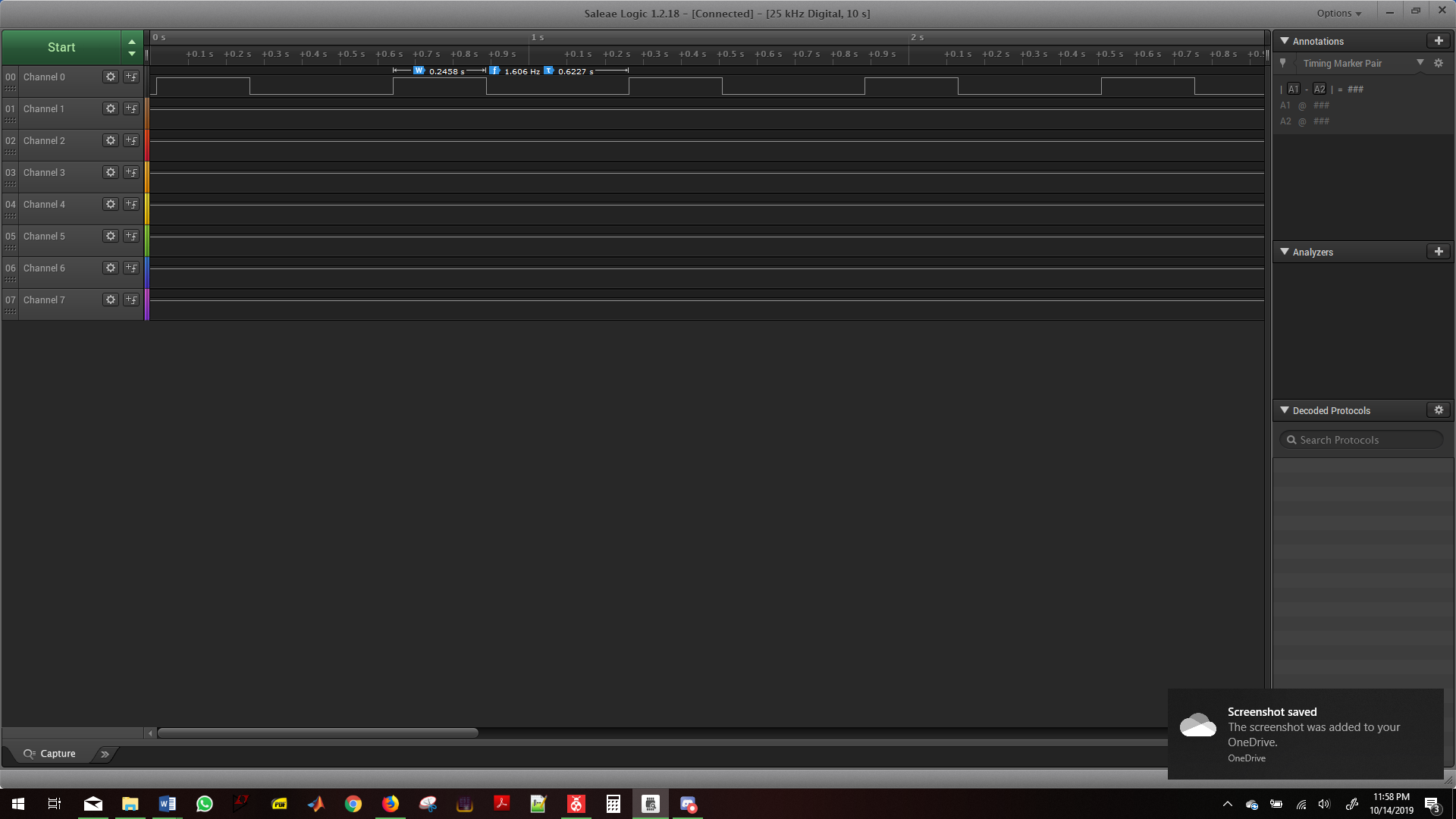
PORTB |= (1<<2); //turn off LED

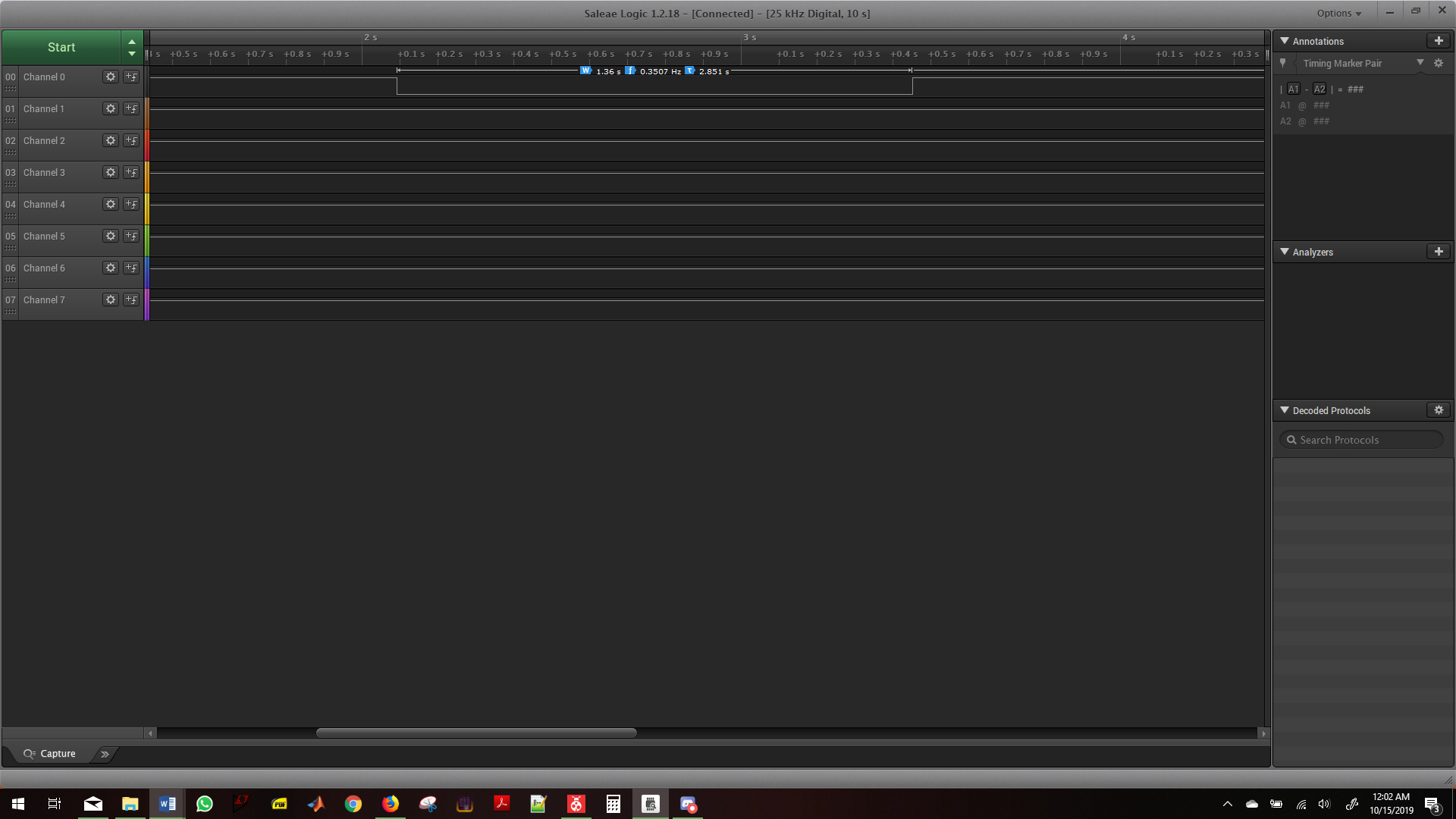
}

}

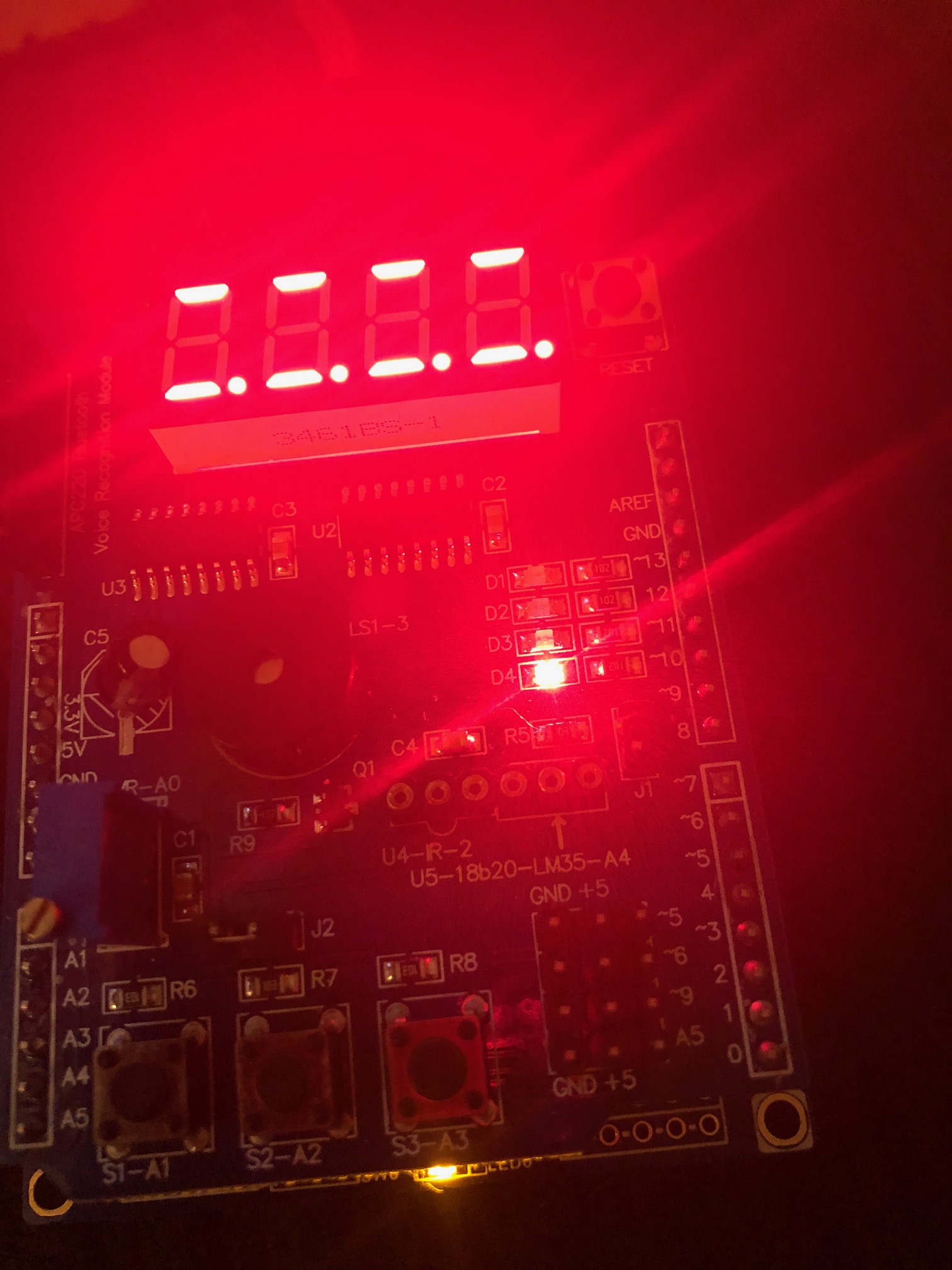
1. **DEVELOPED MODIFIED CODE OF TASK 2/A from TASK 1/A**
2. **SCHEMATICS**
3. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

****





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

****

1. **VIDEO LINKS OF EACH DEMO**

<https://www.youtube.com/watch?v=mo6vX_DPnKQ>

1. **GITHUB LINK OF THIS DA**

**Student Academic Misconduct Policy**

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

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

Andrew Buchanan