CPE301 – SPRING 2019

Design Assignment 2C

Student Name: Mohamad Jundi

Student #: 8000321867

Student Email: jundi@unlv.nevada.edu

Primary Github address: https://github.com/MohamedJundi1994/Submission\_DA.git

Directory: Documents\School\CPE 301\Repository\CPE\_301\DesignAssignments\DA2C

**NOTE:** I had some issues with the Xplained board for #1 - #3 of #2 DA2A, so I used the Lab Board for that portion instead. Thank you.

1. **COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS**

For number 1, 2 and 3 of the 1st problem (Duty Cycle) of DA2A:

PORTB connection => Resistor => PB2 LED

For number 2 (C),

For number 1, 2 and 3 of the 2nd problem (Switch) of DA2A:

Input Button (PINC) => PORTC connection PC2 => PORTB connection => PB2 LED

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

#1 DA2C of #1 DA2A:

// The following Code is #1 of DA2C, covering the topic of #1 (Duty Cycle) from DA2A

#define *F\_CPU* 16000000UL // Frequency of the board

#include <avr/io.h>

int main(void)

{

int ocount = 0; // Declaring overflow counter

DDRB |= (1 << DDB2); // LED 2 is the output

TCCR0A = 0; // Using Normal mode

TCNT0 = 0x00; // Declaring Timer Register set to 0

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

while(1) // Infinite loop

{

while((TIFR0 & 0x01) == 1) // While timer interrupt flag is equal to 1 loop

{

TCNT0 = 0x00; // TCNT0 set to 0x00 and will count up to 254

TIFR0 = 0x01; // TIFR0 is 0x01 so TOV0 is set due to overflow occurrence

ocount++; // Increment overflow counter

if (ocount <= 18) // If overflow counter < 18, keep LED OFF (40%)

PORTB = (0 << DDB2); // Turn OFF LED

else if (ocount >= 27) // If overflow counter is >= 27, LED ON (60%)

PORTB = (1 << DDB2); // Turn ON LED

if (ocount == 44) // If overflow counter is equal to 44, reset overflow counter

ocount = 0; // Overflow counter reset to 0 so operation will restart

}

}

}

#2 DA2C of #1 DA2A:

// The following Code is #2 of DA2C, covering the topic of #1 (Duty Cycle) from DA2A

#define *F\_CPU* 16000000UL // Frequency of the board

#include <avr/io.h>

#include <util/delay.h> // Library must include to use delays

#include <avr/interrupt.h> // Library must be sued to include interrupts

int ocount = 0;

int main(void)

{

DDRB |= (1 << DDB2); // Setting LED 2 as the output

TCCR0A = 0; // Used to verify we are in normal mode

TIMSK0 |= (1 <<TOIE0); // Setting a 1 to TIMSK0 which will enable overflow interrupt

TCNT0 = 0xFF; // Set TCNT to 0xFF

sei(); // Set Global Interrupt Enable

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

while(1)

{

// Infinite while loop waiting on interrupt

}

}

ISR (TIMER0\_OVF\_vect) // Timer0 overflow interrupt

{

while((TIFR0 & 0x02) == 2) // While timer interrupt flag is equal to 2 loop

{

TCNT0 = 0xFF; // TCNT0 set to 0xFF

TIFR0 = 0x02; // TIFR0 is 0x02 so OCF0A is set due to compare match

ocount++; // Increment overflow counter

if (ocount <= 18) // If overflow counter < 18, keep LED OFF (40%)

PORTB = (0 << DDB2); // Turn OFF LED

else if (ocount >= 27) // If overflow counter is >= 27, LED ON (60%)

PORTB = (1 << DDB2); // Turn ON LED

if (ocount == 44) // If overflow counter is equal to 44, reset overflow counter

ocount = 0; // Overflow counter reset to 0 so operation will restart

}

}

#3 DA2C of #1 DA2A:

// The following Code is #3 of DA2C, covering the topic of #1 (Duty Cycle) from DA2A

#define *F\_CPU* 16000000UL // Frequency of the board

#include <avr/io.h>

#include <avr/interrupt.h> // Library must include to use interrupt

int ocount = 0; // Declaring overflow counter

int main(void)

{

DDRB |= (1 << DDB2); // LED 2 is the output

TCCR0A |= (1 << WGM01); // Using CTC Mode

TCNT0 = 0x00; // Declaring Timer Register set to 0

OCR0A = 0xFF; // Top timer value (255) to output compare register

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

TIMSK0 = (1 << OCIE0A); // Compare match is enabled when OCIE0A is set

sei();

while(1)

{

// Infinite Loop

}

}

ISR (TIMER0\_COMPA\_vect) // Timer0 Compare match

{

while((TIFR0 & 0x01) == 1) // While timer interrupt flag is equal to 1 loop

{

TCNT0 = 0x00; // TCNT0 set to 0x00 and will count up to 254

TIFR0 = 0x01; // TIFR0 is 0x01 so TOV0 is set due to overflow occurrence

ocount++; // Increment overflow counter

if (ocount <= 18) // If overflow counter < 18, keep LED OFF (40%)

PORTB = (0 << DDB2); // Turn OFF LED

else if (ocount >= 27) // If overflow counter is >= 27, LED ON (60%)

PORTB = (1 << DDB2); // Turn ON LED

if (ocount == 44) // If overflow counter is equal to 44, reset overflow counter

ocount = 0; // Overflow counter reset to 0 so operation will restart

}

}

#1 DA2C of #2 DA2A:

// The following Code is #1 of DA2C, covering the topic of #2 (Switch) from DA2A

#include <avr/io.h>

int ocount = 0; // Counter initialization

int main(void)

{

DDRB |= (1<<DDB2); // Output is set to PB2

TCCR0A = 0; // Using Normal mode

TCNT0 = 0x00; // Declaring Timer Register set to 0

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

while(1) // Infinite loop

{

while((TIFR0 & 0x01) == 1) // While timer interrupt flag is equal to 1 loop

{

TCNT0 = 0x0A; // TCNT0 set to 0x0A and will count up to 255

TIFR0 = 0x01; // TIFR0 is 0x01 so TOV0 is set due to overflow occurrence

ocount++; // Increment overflow counter

if((PINC & 0x02) == 0x02) // Reading from 1st pin

{

TCNT0 = 0xFF; // Counter set to 0xFF, will lessen counter timing

if (ocount < 36) // Loop as long as counter value is less than calculation for time (1.25s)

{

PORTB = 0xFF; // LED ON while loop runs

ocount = 0; // Counter set back to 0

}

}

else if (ocount > 36) // Loop as long as counter value is greater than calculation for time (1.25s)

{

PORTB = 0x00; // LED OFF if nothing read from 1st pin

ocount = 0; // Counter set back to 0

}

}

}

}

#2 DA2C of #2 DA2A:

// The following Code is #2 of DA2C, covering the topic of #2 (Switch) from DA2A

#include <avr/io.h>

#include <avr/interrupt.h>

int ocount = 0; // Counter initialized

int main(void)

{

DDRB |= (1<<DDB2); // Output is set to PB2

TCCR0A = 0; // Using Normal mode

TCNT0 = 0x00; // Declaring Timer Register set to 0

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

sei(); // Set Global Interrupt Enable

TIMSK0 |= (1 <<TOIE0); // Setting a 1 to TIMSK0 which will enable overflow interrupt

while(1)

{

// Infinite loop

}

}

ISR (TIMER0\_OVF\_vect) // Timer0 overflow interrupt

{

while((TIFR0 & 0x02) == 2) // While timer interrupt flag is equal to 2 loop

{

TCNT0 = 0x00; // TCNT0 set to 0x00 and will count up to 255

TIFR0 = 0x02; // TIFR0 is 0x02 so TOV0 is set due to overflow occurrence

ocount++; // Increment overflow counter

if((PINC & 0x02) == 0x02) // Reading from 1st pin

{

TCNT0 = 0x00; // TCNT set to count to 255 for counter

if (ocount < 16) // Loop as long as counter is less than calculated timer value

{

PORTB = 0xFF; // LED ON while loop runs

ocount = 0; // Resetting counter back to 0

}

}

else if (ocount > 16) // If counter is greater than 16, LED needs to be OFF

{

PORTB = 0x00; // LED OFF if nothing read from 1st pin

ocount = 0; // Counter set back to 0

}

}

}

#3 DA2C of #2 DA2A:

// The following Code is #3 of DA2C, covering the topic of #2 (Switch) from DA2A

#include <avr/io.h>

#include <avr/interrupt.h>

int ocount = 0; // Counter initialized

int main(void)

{

DDRB |= (1<<DDB2); // Output is set to PB2

TCCR0A |= (1 << WGM01); // Using CTC Mode

TCNT0 = 0x00; // Declaring Timer Register set to 0

OCR0A = 0xFF; // Top timer value (255) to output compare register

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

sei(); // Set Global Interrupt Enable

TIMSK0 = (1 << OCIE0A); // Compare match is enabled when OCIE0A is set

while(1)

{

// Infinite loop

}

}

ISR (TIMER0\_COMPA\_vect) // Timer0 Compare match

{

while((TIFR0 & 0x01) == 1) // While timer interrupt flag is equal to 2 loop

{

TCNT0 = 0x00; // TCNT0 set to 0x00 and will count up to 255

TIFR0 = 0x01; // TIFR0 is 0x02 so TOV0 is set due to overflow occurrence

ocount++; // Increment overflow counter

if((PINC & 0x02) == 0x02) // Reading from 1st pin

{

TCNT0 = 0x00; // TCNT set to count to 255 for counter

if (ocount < 35) // Loop as long as counter is less than calculated timer value

{

PORTB = 0xFF; // LED ON while loop runs

ocount = 0; // Resetting counter back to 0

}

}

else if (ocount > 35) // If counter is greater than 17, LED needs to be OFF

{

PORTB = 0x00; // LED OFF if nothing read from 1st pin

ocount = 0; // Counter set back to 0

}

}

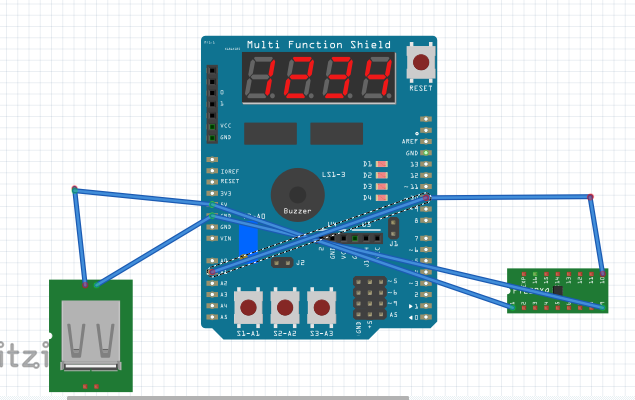
}

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

All code is in number 2.

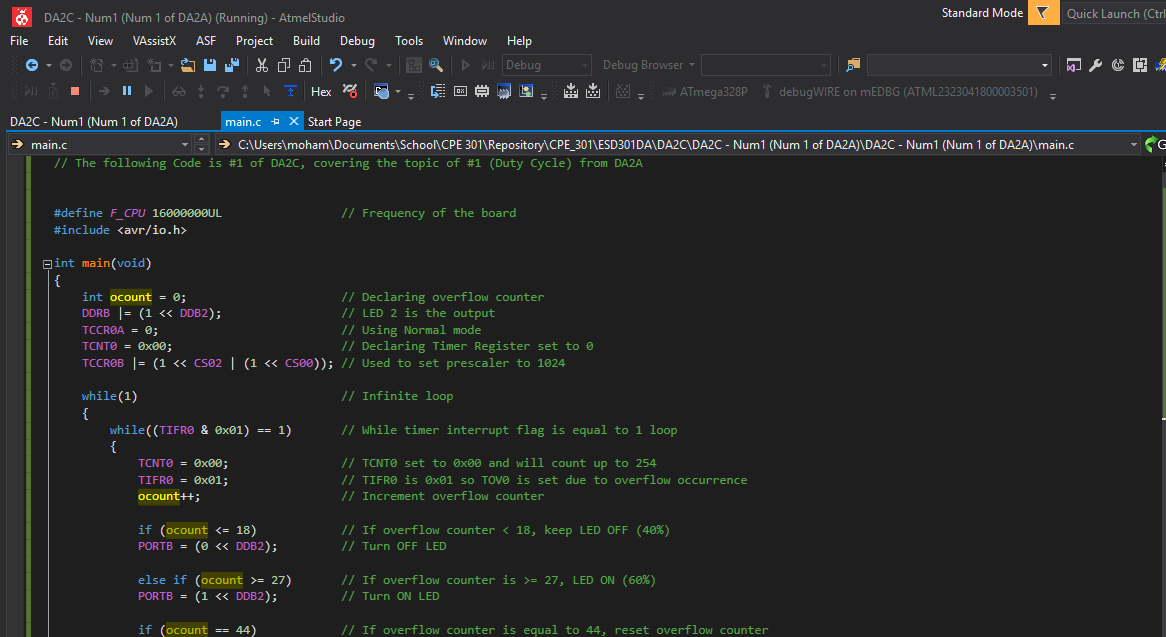
1. **SCHEMATICS**

Schematic built using Fritzing,



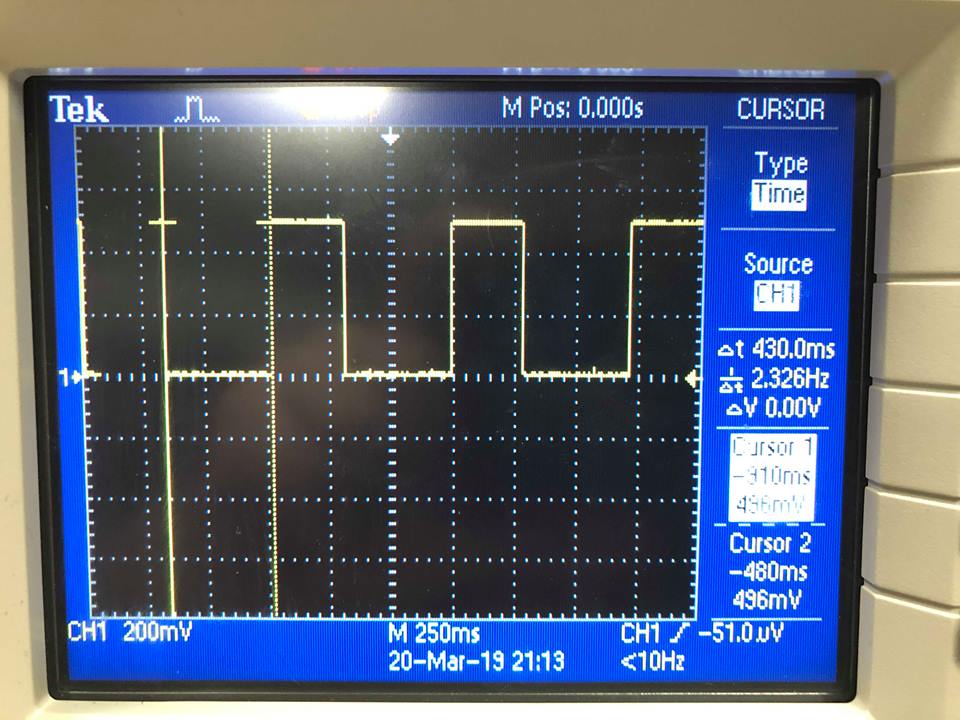
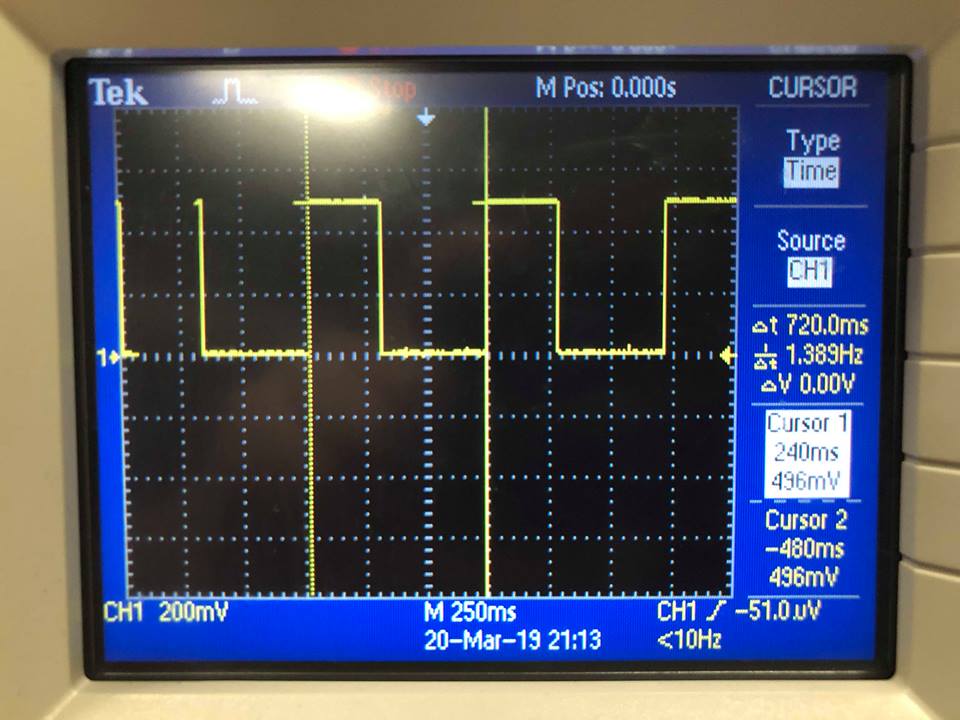
1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

ATMEL SCREENSHOT (#1 DA2C of #1 DA2A):

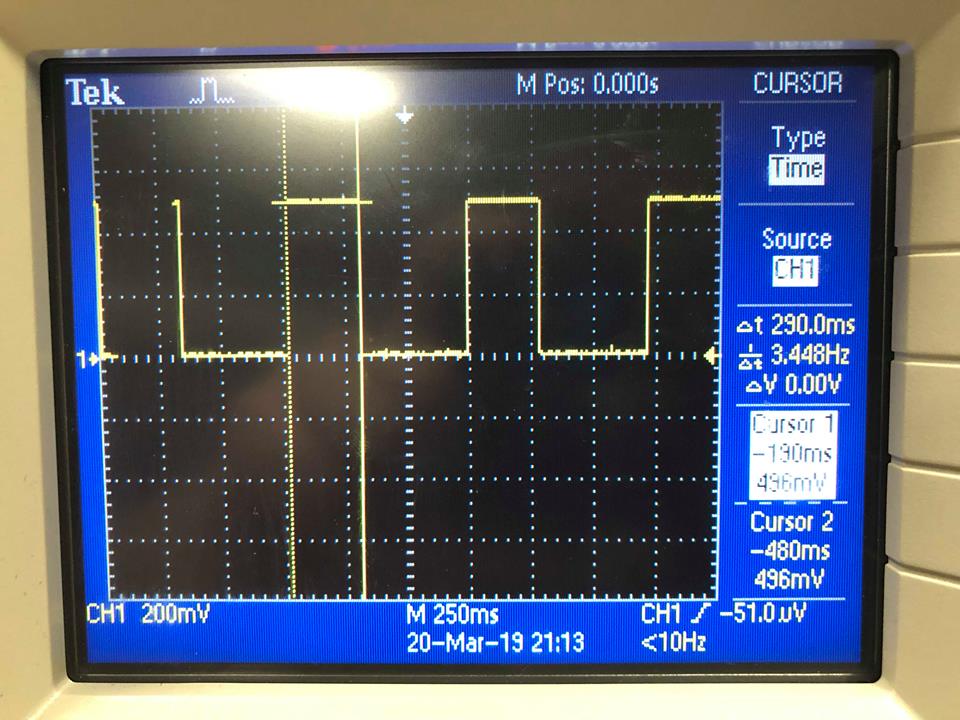


Oscilloscope Verification:

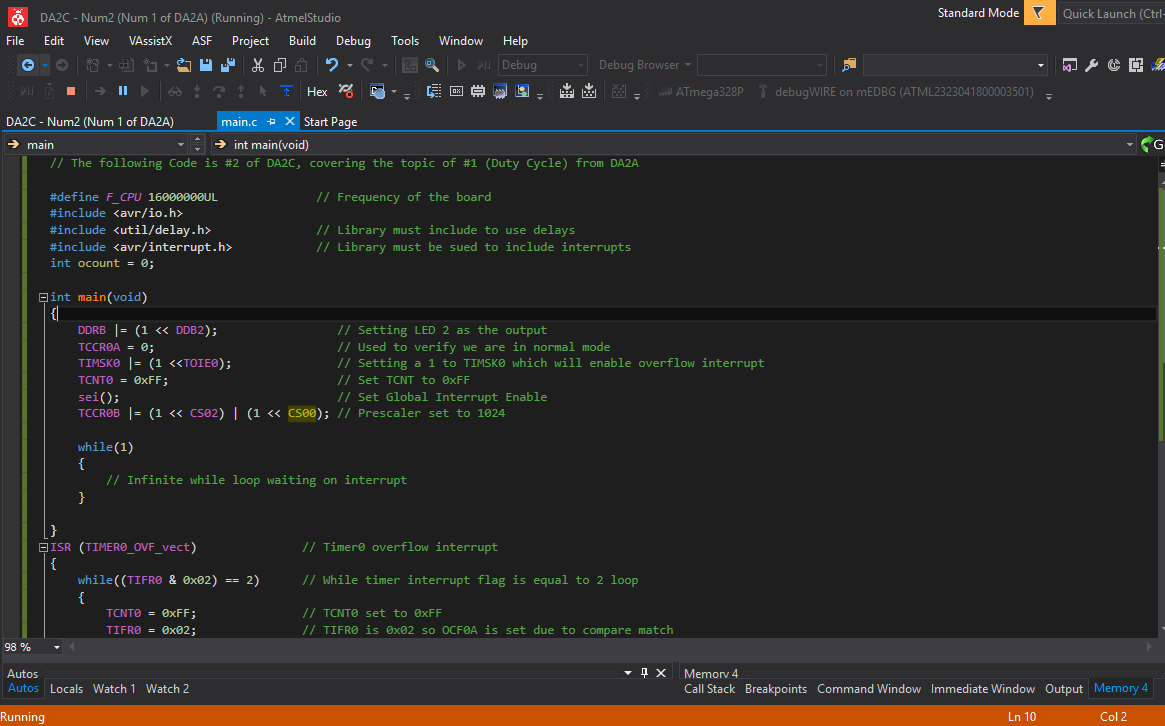
Total Period: LED ON time:



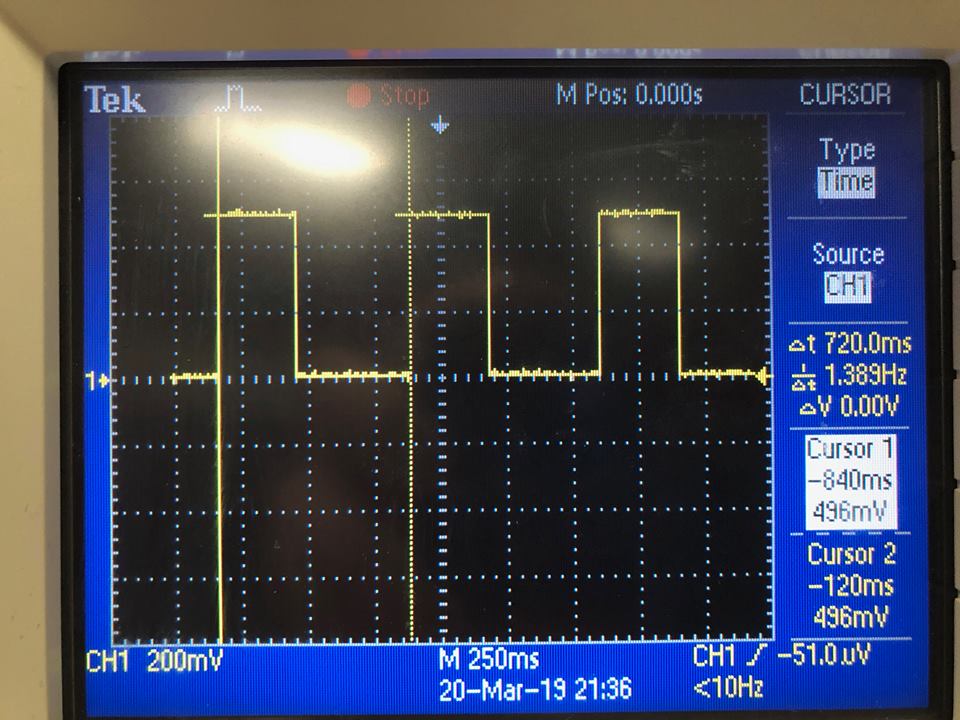
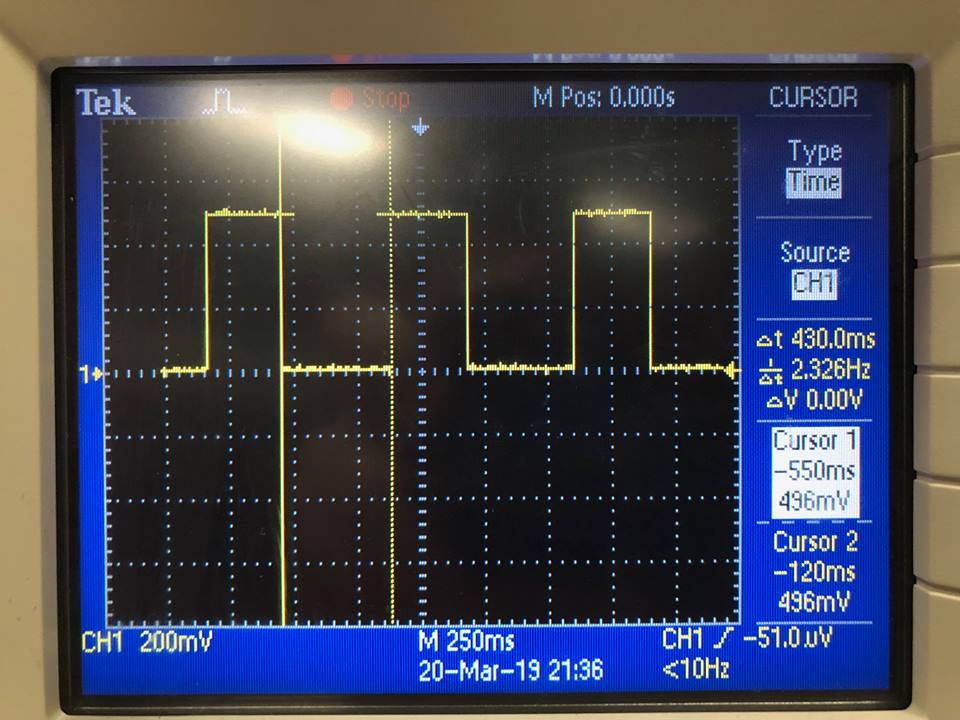
LED OFF time:



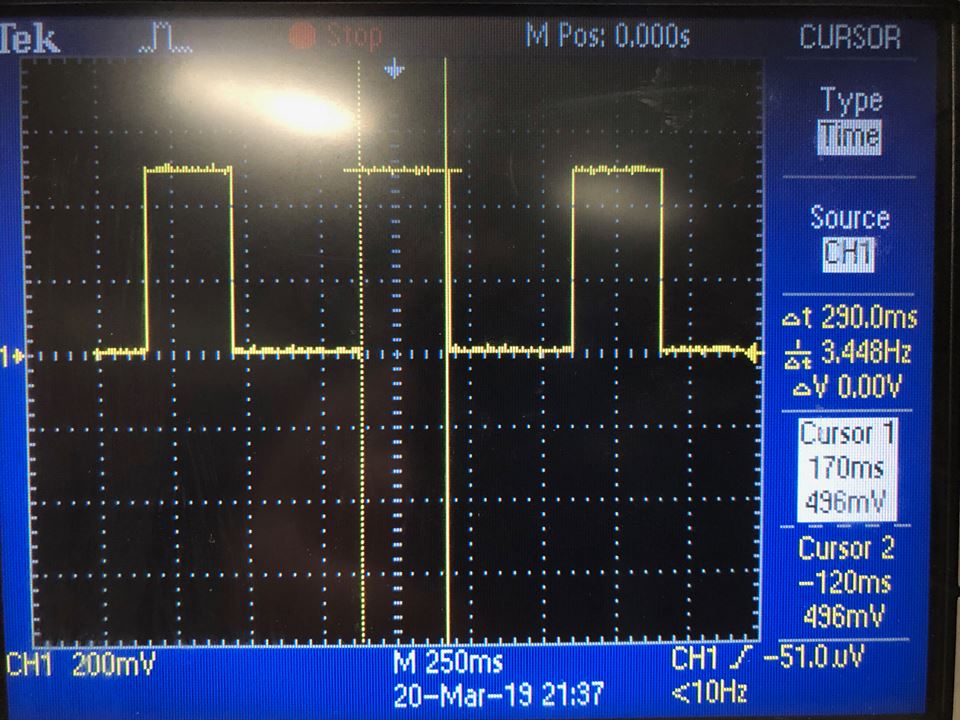
ATMEL SCREENSHOT (#2 DA2C of #1 DA2A):



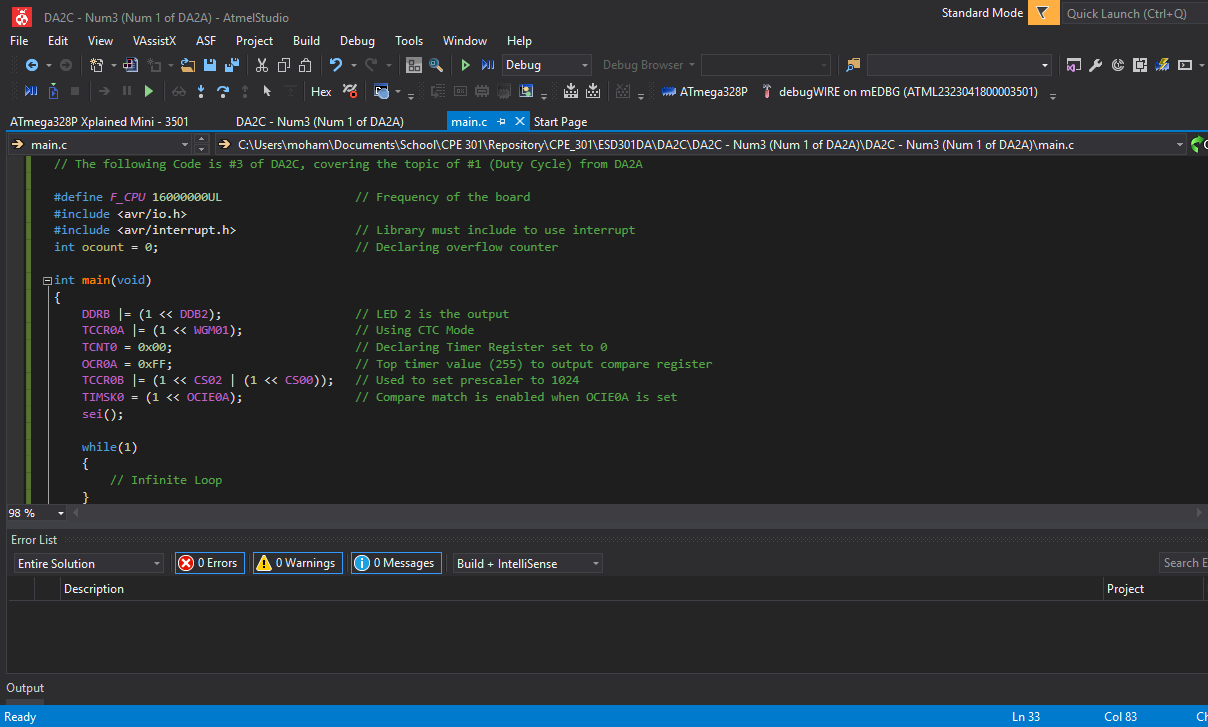
Oscilloscope Verification:

Total Period: LED ON time:****

LED OFF time:

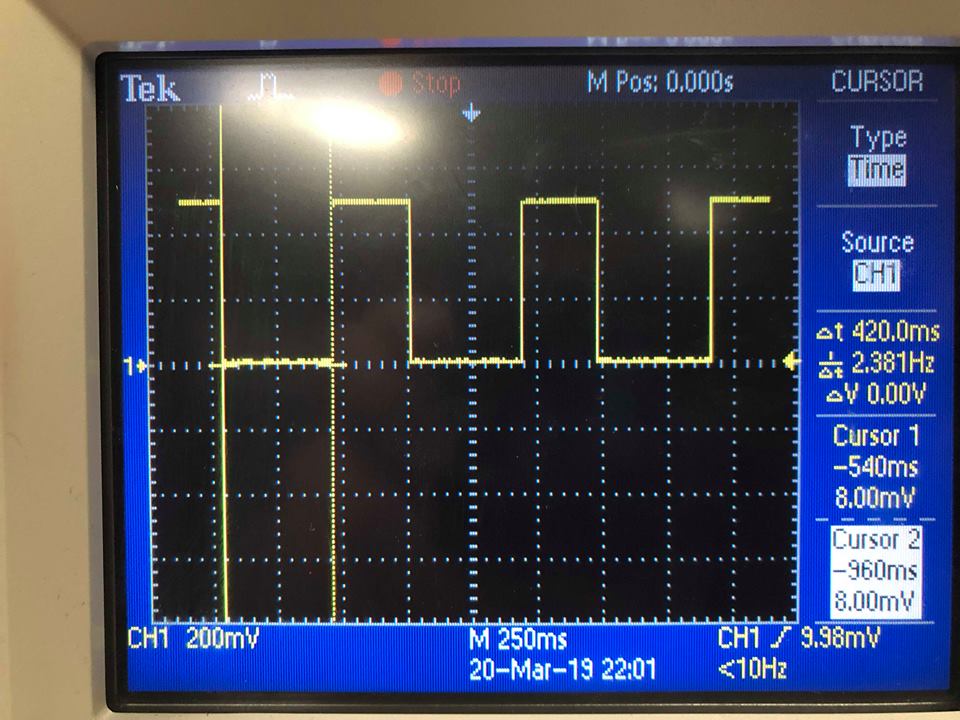
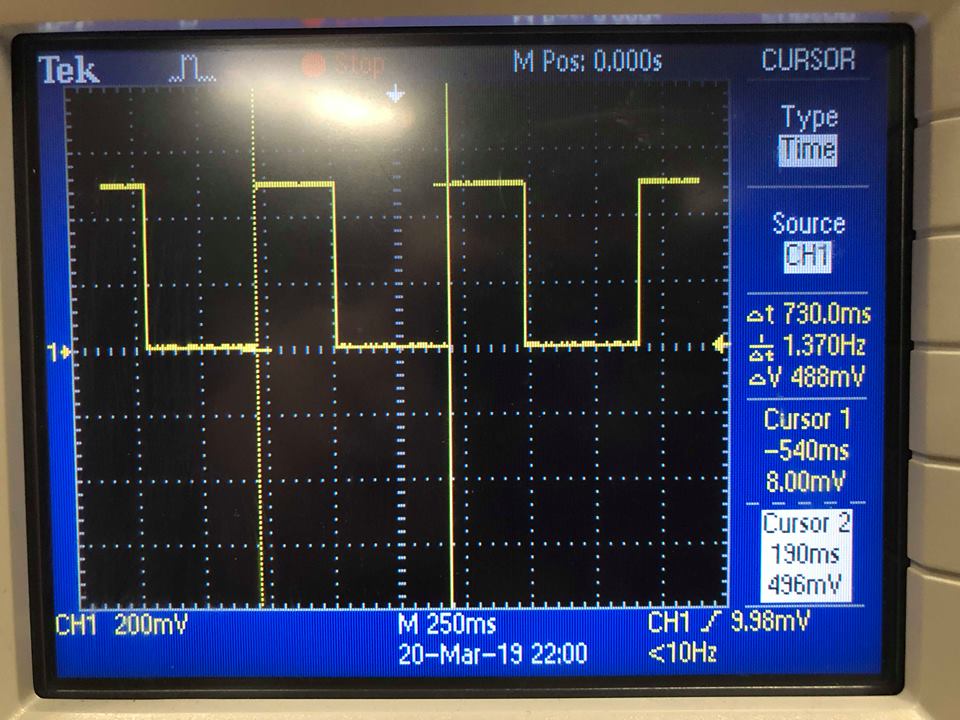


ATMEL SCREENSHOT (#3 DA2C of #1 DA2A):

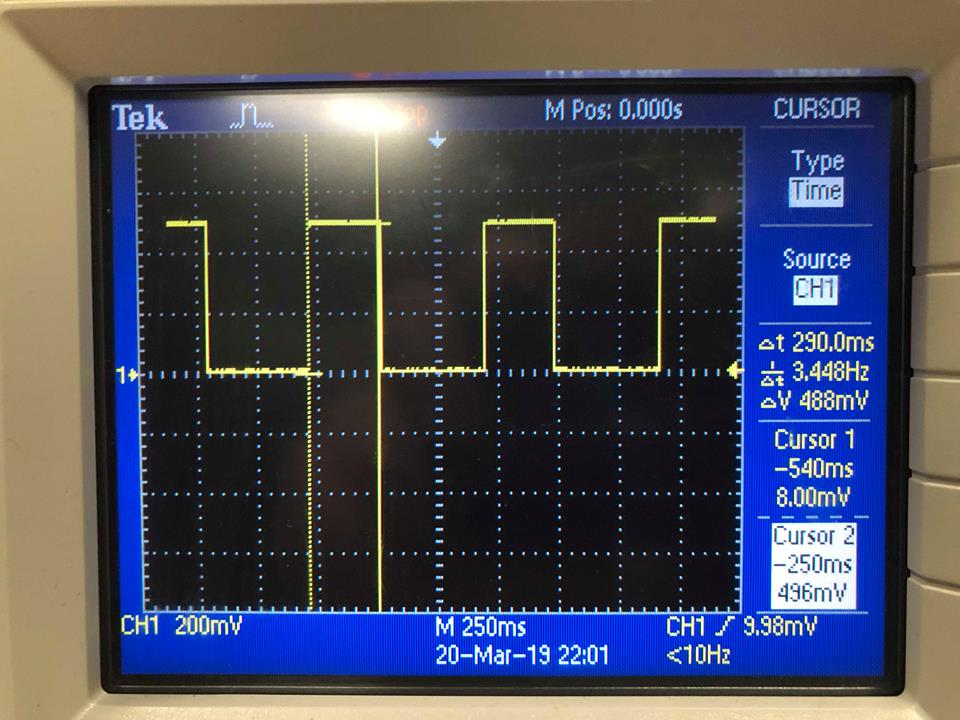


Oscilloscope Verification:

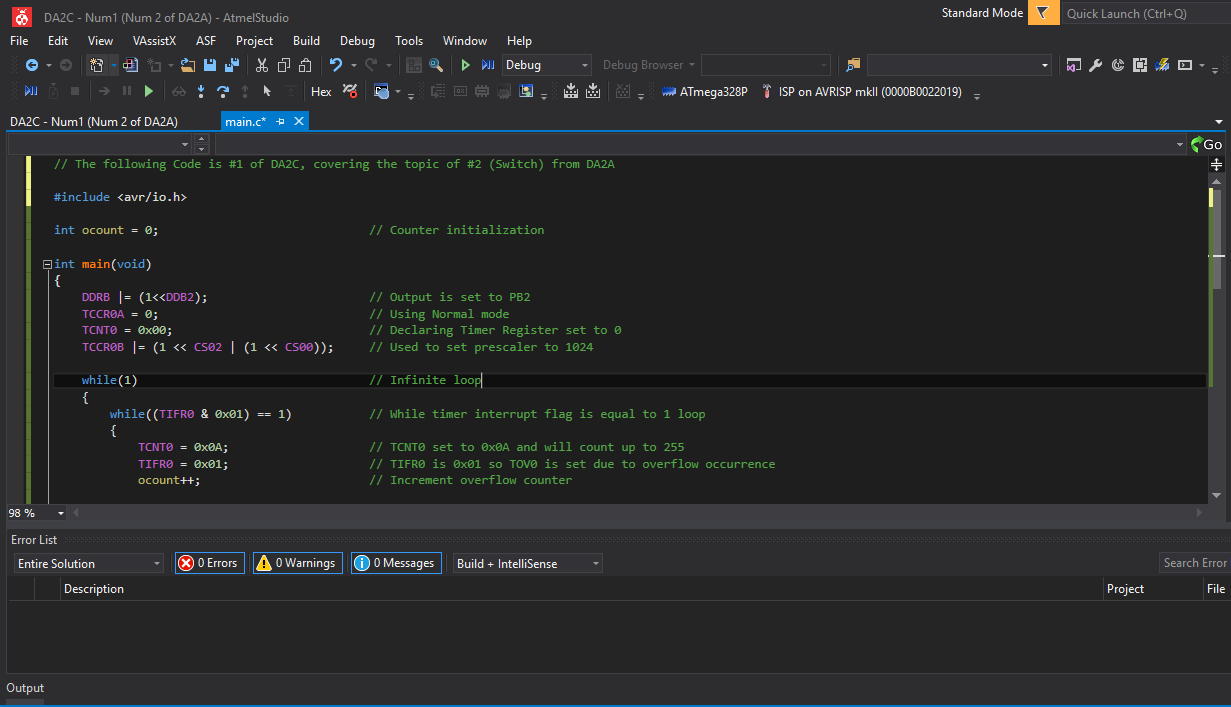
Total Period: LED ON time:



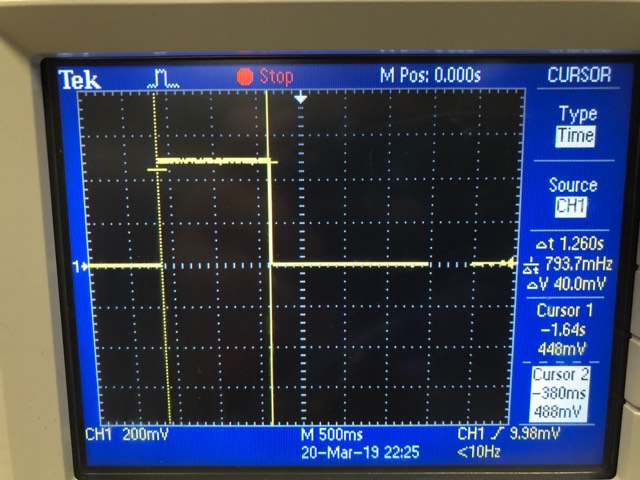
LED OFF time:



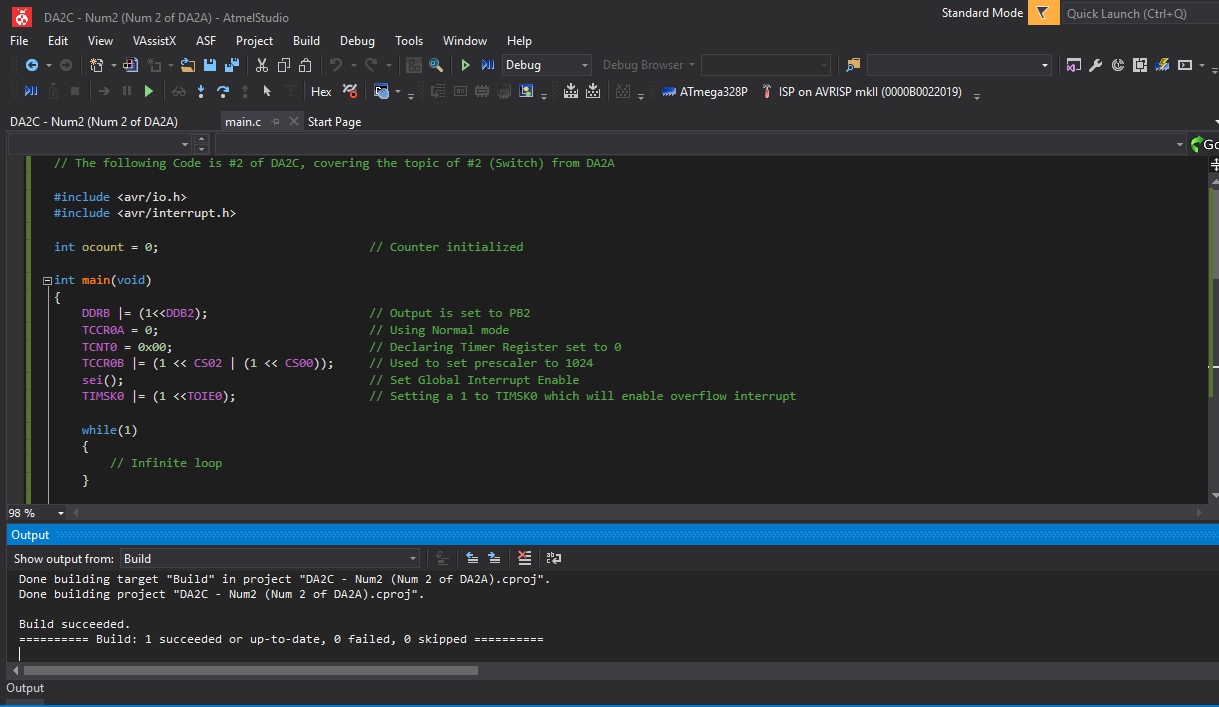
ATMEL SCREENSHOT (#1 DA2C of #2 DA2A):



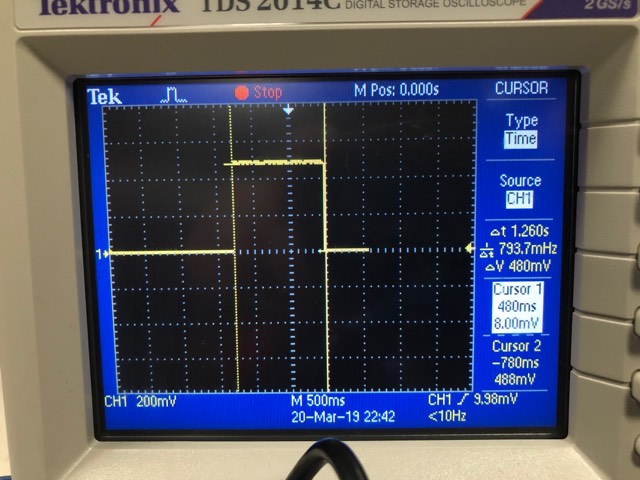
Oscilloscope Verification:



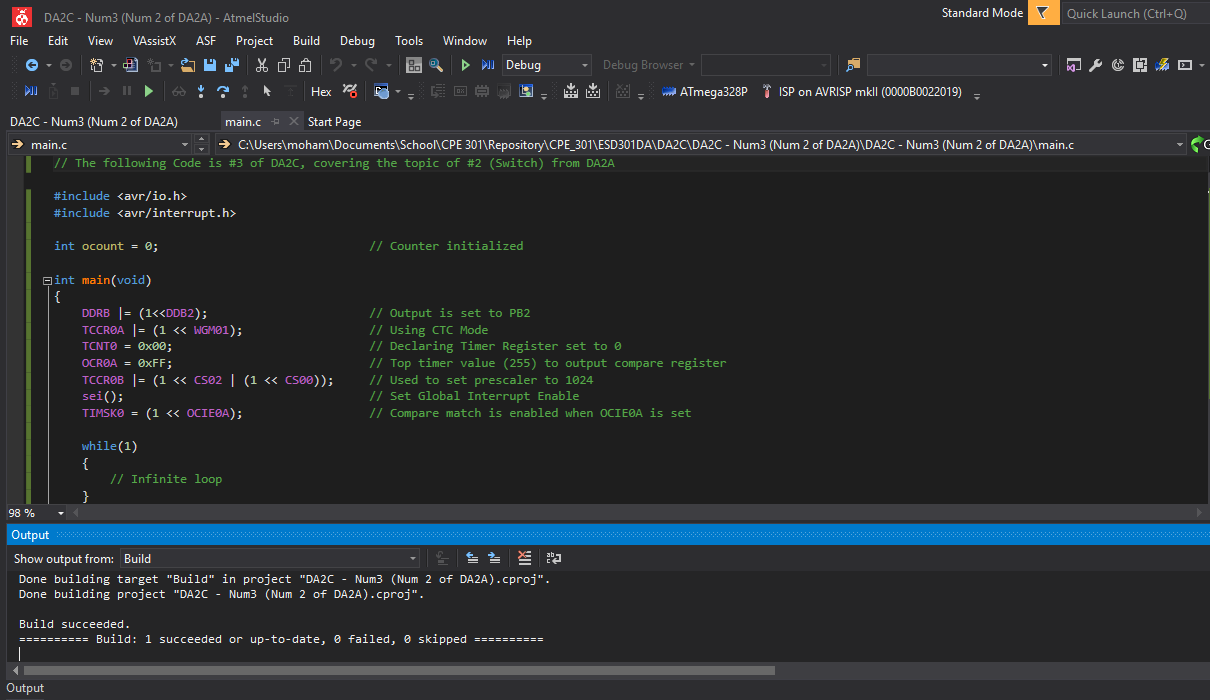
ATMEL SCREENSHOT (#2 DA2C of #2 DA2A):



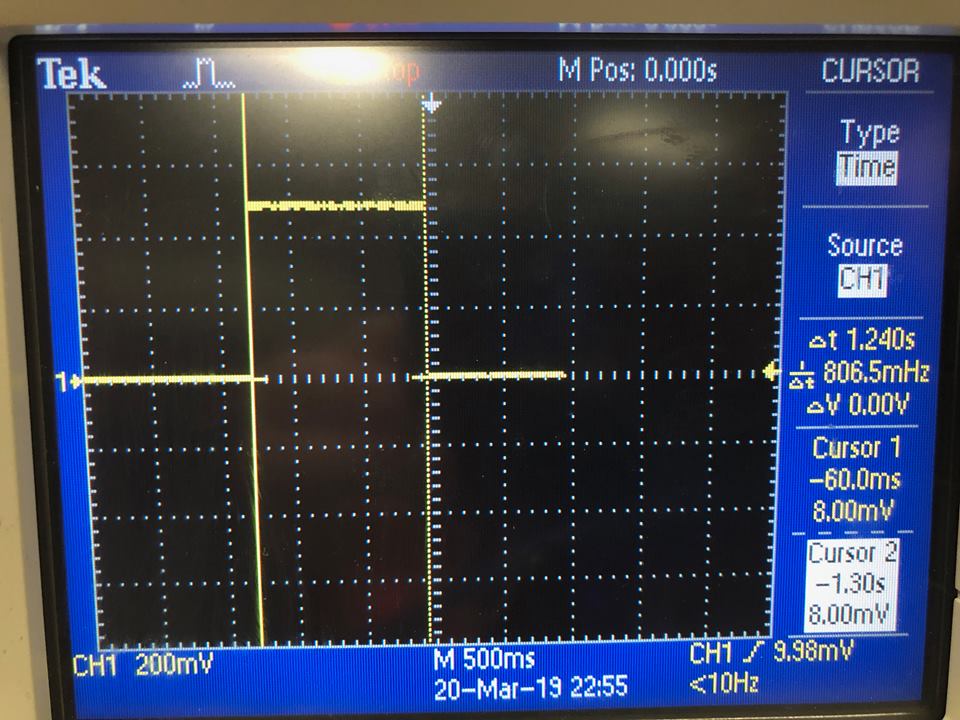
Oscilloscope Verification:



ATMEL SCREENSHOT (#3 DA2C of #2 DA2A):

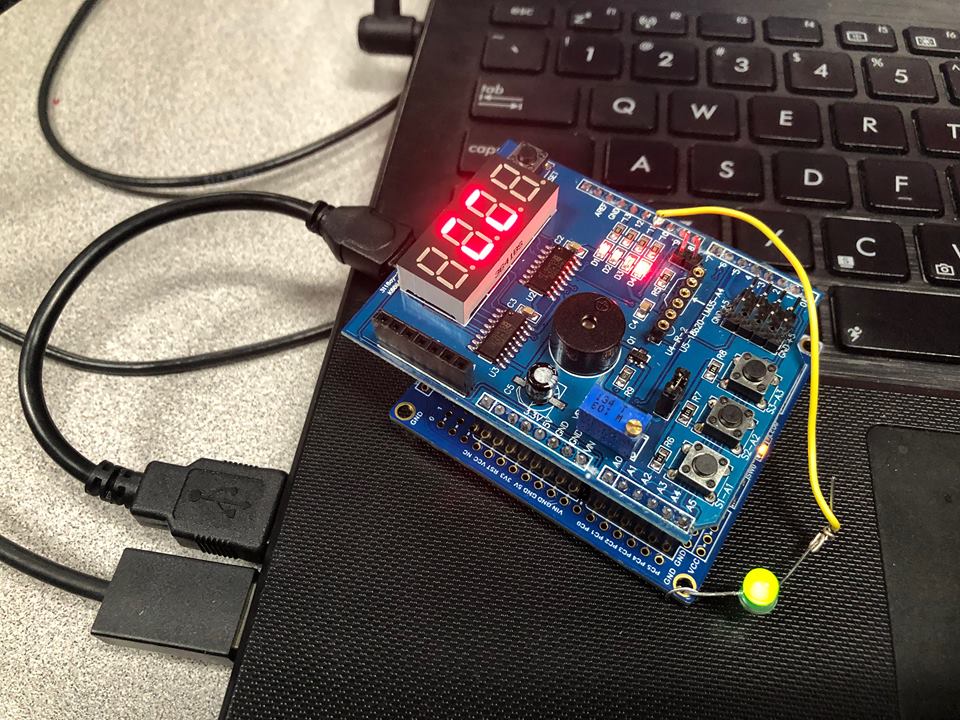


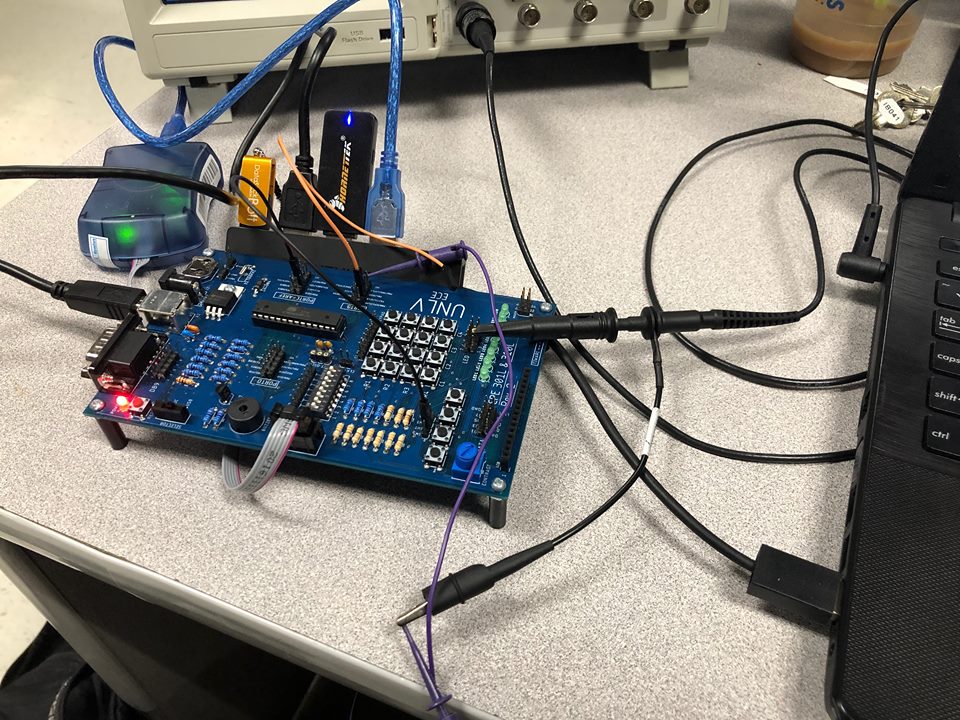
Oscilloscope Verification:



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

Board Setup for #1 - #3 of DA2C for #1 of DA2A:



Board Setup for #1 - #3 of DA2C for #2 of DA2A:

1. **VIDEO LINKS OF EACH DEMO**

#1 DA2C of #1 DA2A:

<https://www.youtube.com/watch?v=wrPu9FWq3f4&feature=youtu.be&fbclid=IwAR3sIL3pU-W7bI1yrvUj9Vh6uOw-VFxyKNcvk3JyDn5iLPLTxznyGKq0bH0>

#2 DA2C of #1 DA2A:

<https://www.youtube.com/watch?v=qSQe4ghyA3I&feature=youtu.be&fbclid=IwAR1SrUYPxo0CGK7Nl8MZA1wgUSXQwppSNQ0XzNDa9B3aJ-BoawQ1ULQjiB8>

#3 DA2C of #1 DA2A:

<https://www.youtube.com/watch?v=K3DeWgxdIkU&feature=youtu.be&fbclid=IwAR16H_CCyOcWzEfOYJXYKB7Q1gmeg7OdQQUDP1zuLje2dj70l4gGAJlM928>

#1 DA2C of #2 DA2A:

<https://www.youtube.com/watch?v=mHm86dH5Lqg&feature=youtu.be&fbclid=IwAR3TLTZ_C27I49gMPK4Czq_pAErI4SIHBwN3aoQTteMhDr0_MvyUSSMIsRk>

#2 DA2C of #2 DA2A:

<https://www.youtube.com/watch?v=tstjrCeG3AM&feature=youtu.be&fbclid=IwAR02bRJ3I5037JeNabbuzP2h3__X8KF6f1XFV9qzteIJ4gjrZ21B6pCsXWY>

#3 DA2C of #2 DA2A:

<https://www.youtube.com/watch?v=AB2zm5yGuG4&feature=youtu.be&fbclid=IwAR23Qe7oev_c1hS13m9ACwiS0jTGlXnkScjg8r2qIXNMutvMj4xXnesjup0>

1. **GITHUB LINK OF THIS DA**

Link: https://github.com/MohamedJundi1994/Submission\_DA.git

This assignment submission is my own, original work.

MOHAMAD JUNDI