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

* LED
* Push button
* Multi-Function Shield
* Xplained Mini

1. **Task 1/A**

/\*

\* DA2C\_Task1A.c

\*

\* Created: 3/20/2019 6:10:30 PM

\* Author : perezr1

\*/

#include <avr/io.h>

// TCNT\_ON = (((16MHz/1024)\*0.435) -1 ) = 6796

// TCNT\_OFF = (((16MHz/1024)\*0.29) -1 ) = 4530

int main(void)

{

DDRB =36; // sets to pd2 and pd5

PORTB |= (1<<5); // turns off pd5

TCCR0B = 5; // 1024

while (1)

{

TCNT0 = 0; // sets clock

int count = 0;

// when 60% DC is on

while(count!=27) // 6796/255 = 27

{

while ( TCNT0 != 255)

{

// delay

}

TCNT0 =0; // reset timer

count++; // count += 1

}

PORTB ^= (1<<2); // toggle portb

count = 0; // resets count

TCNT0 = 0; // resets clock

// when 40% DC is off

while(count!=18) // 4530/255 = 18

{

while (TCNT0 != 255)

{

// delay

}

TCNT0 =0; // resets timer

count++; // count += 1

}

}

}

1. **Task 1/B**

/\*

\* DA2C\_Task1B.c

\*

\* Created: 3/20/2019 6:52:39 PM

\* Author : perezr1

\*/

#define *F\_CPU* 16000000UL // sets frequency to 16MHz

#include <avr/io.h>

// // TCNT0 = (((16MHz/1024)\*1.25) -1 ) = 19530

int main(void)

{

DDRB = 36; // sets to pd2 and pd5

PORTB |= (1<<5); // turns off pd5

PORTB |= (1<<2); // turns off pd2

DDRC &= (0<<2); // sets it to read

PORTC |= (0<<2); // has portC clear

TCCR0B = 5; // 1024

while (1) // while true

{

TCNT0 = 0; // timer is set to zero

int counter = 0; // resets counter

if(!(PINC&(1<<PINC2))) // will detect if the push button was pushed

{

while(counter < 77) // 19530/255 = 77

{

while (TCNT0 != 255)

{

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

}

TCNT0 =0; // resets timer

counter++; // counter += 1

}

counter = 0; // resets counter

}

else

{

PORTB |= (1<<2); // will keep the LED off

}

}

return 0;

}

1. **Task 2/A**

/\*

\* DA2C\_Task2A.c

\*

\* Created: 3/20/2019 9:06:40 PM

\* Author : perezr1

\*/

#define *F\_CPU* 16000000UL // sets frequency to 16MHz

#include <avr/io.h>

#include <avr/interrupt.h>

// // TCNT0 = (((16MHz/1024)\*1.25) -1 ) = 19530

int main(void)

{

DDRB =36; // sets to pd2 and pd5

PORTB |= (1<<5); // turns off pd5

TCCR0B = 5; // 1024

TIMSK0 = (1<< TOIE0);

sei();

while (1)

{

// while true

}

}

ISR (TIMER0\_OVF\_vect)

{

TCNT0 = 0; // sets clock

int count = 0; // keeps count of overflow

// when 60% DC is on

while(count!=27) // 6796/255 = 27

{

while ( TCNT0 != 255)

{

// delay

}

TCNT0 =0; // reset timer

count++; // count += 1

}

PORTB ^= (1<<2); // toggle portb

count = 0; // resets count

TCNT0 = 0; // resets clock

// when 40% DC is off

while(count!=18) // 4530/255 = 18

{

while (TCNT0 != 255)

{

// delay

}

TCNT0 =0; // resets timer

count++; // count += 1

}

}

1. **Task 2/B**

/\*

\* DA2C\_Task2B.c

\*

\* Created: 3/20/2019 8:31:40 PM

\* Author : perezr1

\*/

#define *F\_CPU* 16000000UL // sets frequency to 16MHz

#include <avr/io.h>

#include <avr/interrupt.h>

// // TCNT0 = (((16MHz/1024)\*1.25) -1 ) = 19530

int main(void)

{

DDRB = 36; // sets to pd2 and pd5

PORTB |= (1<<5); // turns off pd5

PORTB |= (1<<2); // turns off pd2

DDRC &= (0<<2); // sets it to read

PORTC |= (0<<2); // has portC clear

TCCR0B = 5; // 1024

TIMSK0 = (1<< TOIE0);

sei();

while (1)

{

// while true

}

}

ISR (TIMER0\_OVF\_vect)

{

TCNT0 = 0; // timer is set to zero

int counter = 0; // resets counter

if(!(PINC&(1<<PINC2))) // will detect if the push button was pushed

{

while(counter < 77) // 19530/255 = 77

{

while (TCNT0 != 255)

{

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

}

TCNT0 =0; // resets timer

counter++; // counter += 1

}

counter = 0; // resets counter

}

else

{

PORTB |= (1<<2); // will keep the LED off

}

}

1. **Task 3/A**

/\*

\* DA2C\_Task3A.c

\*

\* Created: 3/20/2019 10:30:14 PM

\* Author : perezr1

\*/

#include <avr/io.h>

#include <avr/interrupt.h>

#define *F\_CPU* 16000000UL

// TCNT\_ON = (((16MHz/1024)\*0.435) -1 ) = 6796

// TCNT\_OFF = (((16MHz/1024)\*0.29) -1 ) = 4530

int main(void)

{

DDRB =36; // sets to pd2 and pd5

PORTB |= (1<<5); // turns off pd5

OCR0A = 255;// max

TCCR0A = 2; // CTC

TCCR0B = 5; // 1024

TIMSK0 = (1<<OCIE0A);

sei();

while (1)

{

}

}

ISR (TIMER0\_COMPA\_vect)

{

TCNT0 = 0; // sets clock

int count = 0;// sets counter

// when 60% DC is on

while(count != 27) // 6796/255 = 27

{

while ((TIFR0 & (1<<OCF0A)) == 0)

{

// delay

}

TCNT0 = 0; // reset timer

TIFR0 |= (1<<OCF0A);

count++; // count += 1

}

PORTB ^= (1<<2); // toggle portb

count = 0; // resets count

TCNT0 = 0; // resets clock

//TIFR0 |= (1<<OCF0A);

// when 40% DC is off

while(count != 18) // 4530/255 = 18

{

while ((TIFR0 & (1<<OCF0A)) == 0)

{

// delay

}

TCNT0 = 0; // resets timer

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

count++; // count += 1

}

}

1. **Task 3/B**

/\*

\* DA2C\_Task3B.c

\*

\* Created: 3/20/2019 9:06:40 PM

\* Author : perezr1

\*/

#define *F\_CPU* 16000000UL // sets frequency to 16MHz

#include <avr/io.h>

#include <avr/interrupt.h>

// // TCNT0 = (((16MHz/1024)\*1.25) -1 ) = 19530

int main(void)

{

DDRB = 36; // sets to pd2 and pd5

PORTB |= (1<<5); // turns off pd5

PORTB |= (1<<2); // turns off pd2

DDRC &= (0<<2); // sets it to read

PORTC |= (0<<2); // has portC clear

OCR0A = 255;// overflow

TCCR0A = 2;// CTC mode

TCCR0B = 5; // 1024

TIMSK0 = (1<<OCIE0A);

sei();

while (1)

{

// while true

}

}

ISR (TIMER0\_COMPA\_vect)

{

TCNT0 = 0; // timer is set to zero

int counter = 0; // resets counter

if(!(PINC&(1<<PINC2))) // will detect if the push button was pushed

{

while(counter < 77) // 19530/255 = 77

{

while ((TIFR0 & (1<<OCF0A)) == 0)

{

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

}

TCNT0 =0; // resets timer

TIFR0 |= (1<<OCF0A);//resets

counter++; // counter += 1

}

counter = 0; // resets counter

}

else

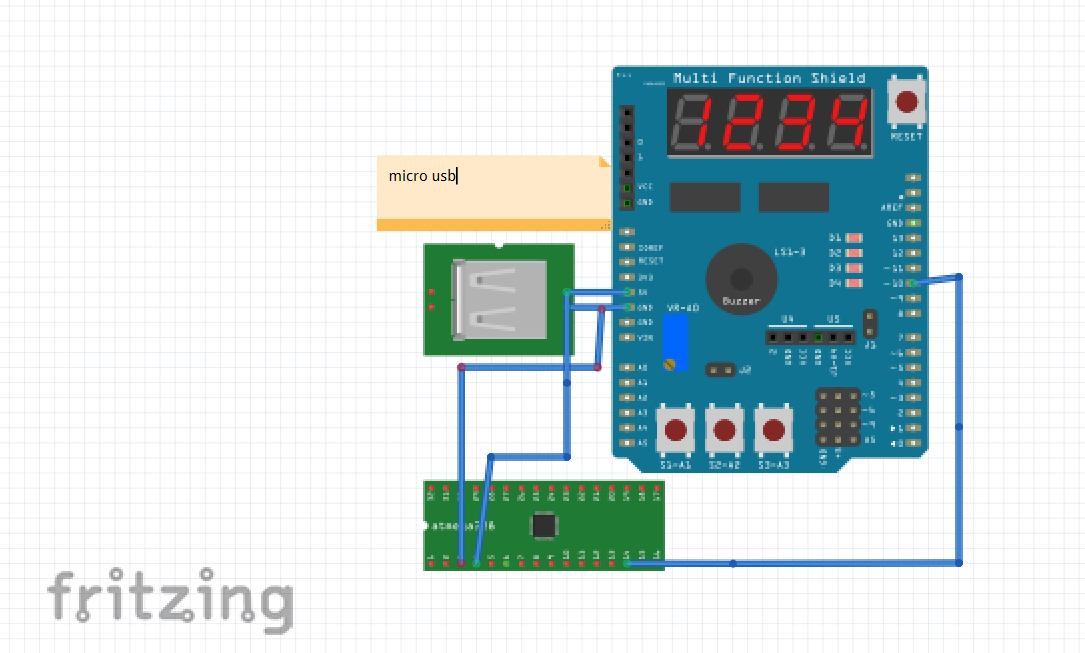
{

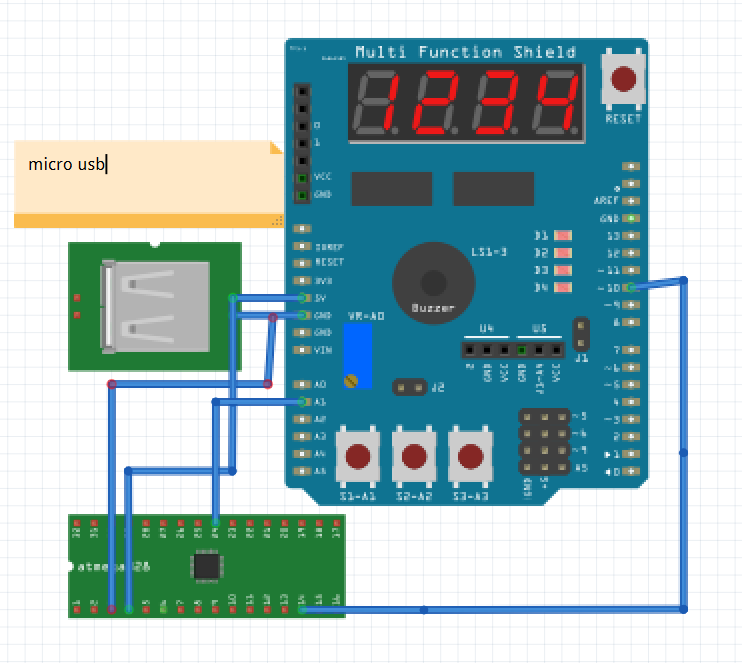
PORTB |= (1<<2); // will keep the LED off

}

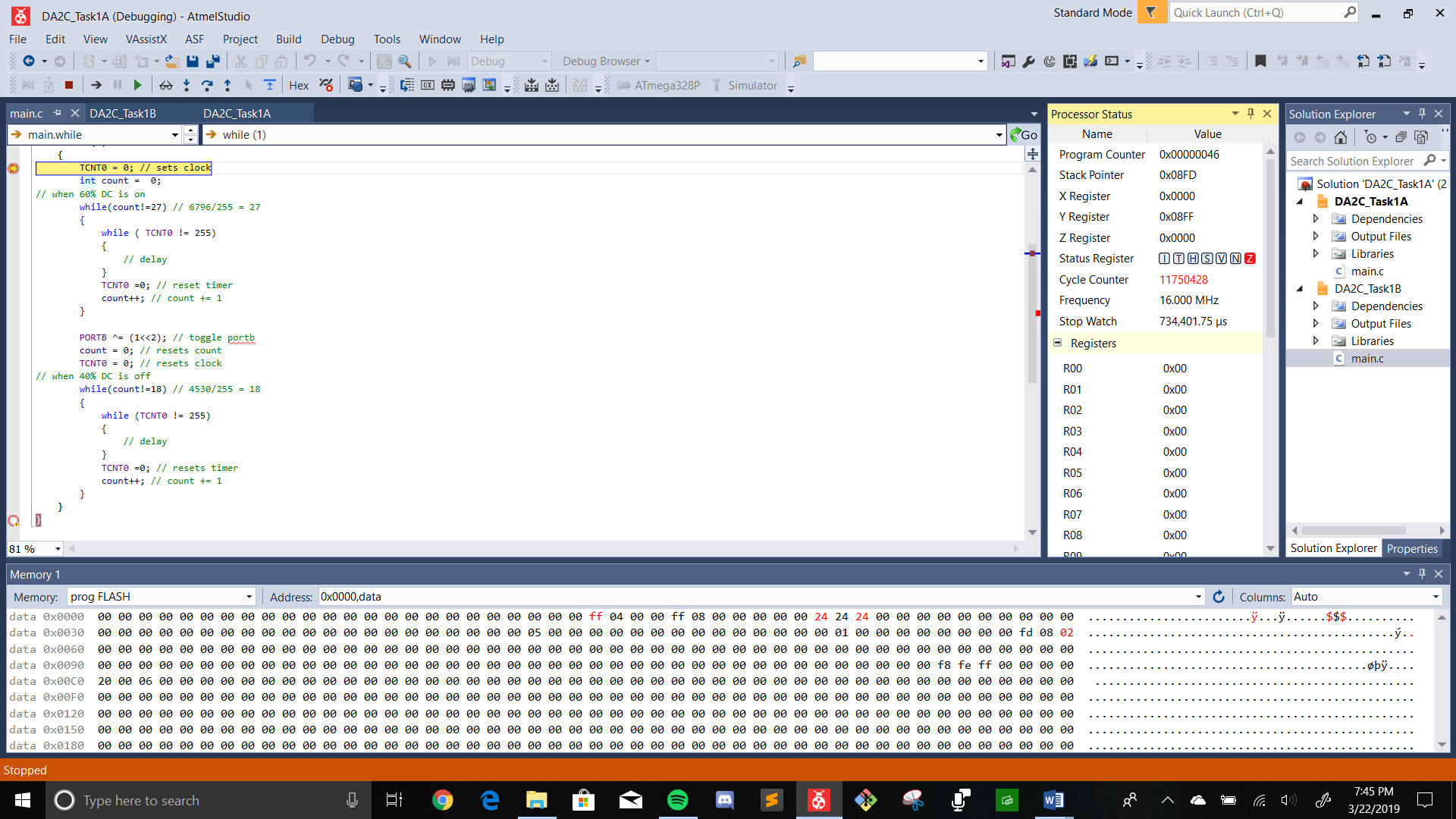
}

1. **SCHEMATICS**

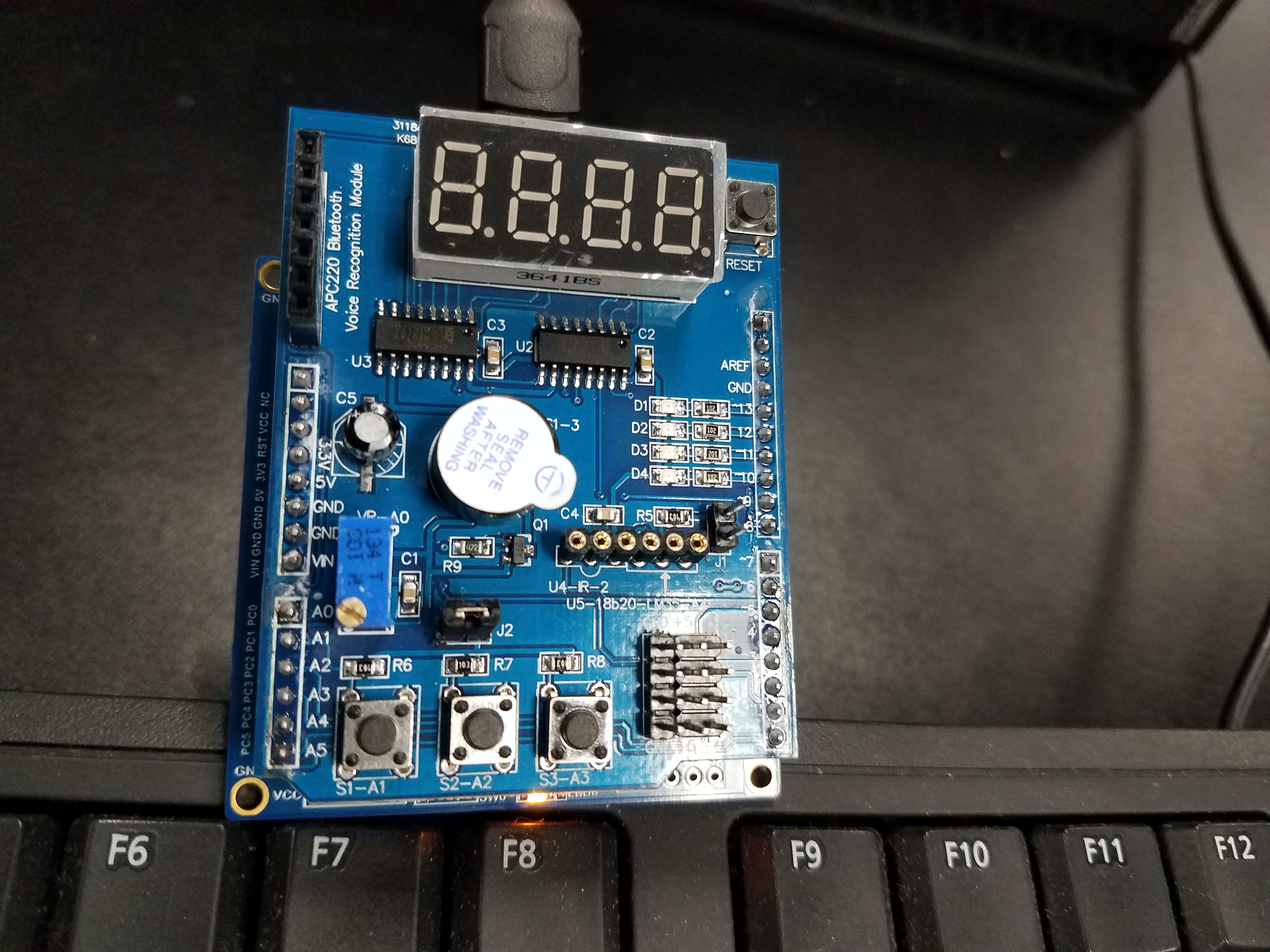




1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**



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



1. **VIDEO LINKS OF EACH DEMO**

Task 1 : <https://youtu.be/z84JTwUb8HY>

Task 2: <https://youtu.be/9t0OyHYBj3E>

Task 3: <https://youtu.be/HogiszChmZY>

1. **GITHUB LINK OF THIS DA**

<https://github.com/RickyPerez79/submission_da.git>

**Student Academic Misconduct Policy**

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

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

RICKY PEREZ