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

* Multi-Function Shield
* Switches
* LED
* Xplained Mini

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

Task 1 assembly code:

;

; DA2A\_1\_Assembly.asm

;

; Created: 3/4/2019 6:28:10 PM

; Author : perezr1

;

.org 0x00

;;;;;;;;;;;;;;;;;;;;;;;;Initialize Port & TCCR1B ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

ldi r16, 0x24 ;setting pd2 & pd5

out DDRB, r16 ; enableing pd2 & pd5 as output

ldi r16, 0x04 ; loads 4 in r16

ldi r17, 0x20 ; used to set/reset pd2 & pd5

ldi r20, 0x05 ; prescalar 1024

sts TCCR1B, r20 ; setting up TCCR1B = 00000101 for 1024

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

LetsBegin:

Ldi r20, 0 ; setting up timer counter to 0

;;;;;;;;;Timer Counter needs two registers since its a 16 bit ;;;;;;;;;;;;;;;;;;;;;;;;

sts TCNT1H, r20 ; high byte of the TCNT

sts TCNT1L, r20 ; low byte of the TNCT

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

rjmp delay\_on\_low ; will check the low byte of the TCNT for the 60% DC

ImBack:

eor r17, r16 ; xor to toggle LED

out PORTB, r17 ; enable pd2 & pd5

ldi r20, 0; ; setting up timer counter back to 0 to check for the 40% DC that it is off

sts TCNT1H, r20 ; high byte of the TNCT

sts TCNT1L, r20 ; low byte of the TNCT

rjmp delay\_off\_low ; will check the low byte of the TCNT for the 40% DC

ImBackAgain:

eor r17, r16 ; xor to toggle LED

out PORTB, r17 ; enable pd2 & pd5

rjmp LetsBegin ; repeats all over again

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

delay\_off\_low:

lds r29, TCNT1H ; load upper bytes of timer counter to r29

lds r28, TCNT1L ; load lower bytes of timer counter to r28

cpi r28, 0xB2 ; check to see if lower 8 bits of timer counter is 0x12

brsh delay\_off\_high ; if lower 8 bits of timer counter is 0x12 then jump label

rjmp delay\_off\_low ; otherwise keep checking lower bytes

delay\_off\_high:

cpi r29, 0x11 ; check if upper timer counter is 0x4F

brlt delay\_off\_low; otherwise recheck the lower bytes

rjmp ImBackAgain ; finished checking

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

delay\_on\_low:

lds r29, TCNT1H ; load upper bytes of timer counter to r29

lds r28, TCNT1L ; load lower bytes of timer counter to r28

cpi r28, 0x8C ; check to see if lower 8 bits of timer counter are 0x08

brsh delay\_on\_high

rjmp delay\_on\_low ; otherwise keep checking lower bytes

delay\_on\_high:

cpi r29, 0x1A ; check if upper timer counter have reached desired value

brlt delay\_on\_low ; otherwise recheck the lower bytes

rjmp ImBack

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Task 2 assembly code:

;

; DA2A\_Task2\_Assembly.asm

;

; Created: 3/4/2019 3:08:27 PM

; Author : perezr1

;

.org 0

;;;;;;;;;;;;;;;;;;;;;;;; Initializing Ports ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

ldi r16,0 ; loads 0 in r16

out ddrc, r16 ; will clear

ldi r16, 0xff ; loads 0xff in r16

out ddrb, r16 ; data direction register b

out portb, r16 ; outputs r16 to portb

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Start:

sbic PINC, 2 ; If bit of PC2 is high, then it will skip

rjmp Start ; will keep looping until it push buttom is pushed

rcall Light\_LED\_UP ; Call subroutine to light up LED

LED\_Done:

rjmp Start; will go back to read for push button

;;;;;;;;;;;;;;;; Will Keep LED on for 1.25 seconds;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Light\_LED\_UP:

cbi portb, 2 ; pb2 = 0 ==> its ON

ldi R20, 5; Prescalar 1024

sts TCCR1B, R20 ; Set timer

ldi R20, 0 ; set timer to zero

sts TCNT1H, R20 ; sets upper byte to zero

sts TCNT1L, R20 ; sets lower byte to zero

rjmp Delay\_Low ; subroute for delay

;;;;;;;;;;;;;;;;;;;;;;;;;;; Delay Starts here;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Delay\_Low:

lds R29, TCNT1H ; load upper bytes of timer counter to r29

lds R28, TCNT1L ; load lower bytes of timer counter to r28

cpi R28, 0x4A ; check to see if lower 8 bits of timer counter is 0x4A

brsh Delay\_High ; if lower 8 bits of timer counter is 0x4A than jumps to label

rjmp Delay\_Low ; otherwise keep checking lower bytes

Delay\_High:

cpi R29, 0x4C ; check if upper timer counter have reached desired value

brlt Delay\_Low ; otherwise recheck the lower bytes

sbi portb, 2 ; TURN OFF LED

ret ; returns back from subroutine

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

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

Task 1 C code:

/\*

\* DA2A\_1\_C\_code.c

\*

\* Created: 3/4/2019 7:50:37 PM

\* Author : perezr1

\*/

#include <avr/io.h>

int main(void)

{

DDRB =36; // sets to pd2 and pd5

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

TCCR1B = 5; // 1024

while (1)

{

TCNT1 = 0; // sets clock

while( TCNT1 != 6796) // when 60% DC is on

{

//Delay

}

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

TCNT1 = 0; // resets clock

while( TCNT1 != 4530) // when 40% DC is off

{

// Delay

}

}

}

Task 2 C code:

/\*

\* DA2A\_2\_Ccode.c

\*

\* Created: 3/4/2019 5:54:55 PM

\* Author : perezr1

\*/

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

#include <avr/io.h>

#include <util/delay.h>

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

while (1) // while true

{

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

{

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

*\_delay\_ms*(1250); // delay of 1250ms == 1.25 seconds

}

else

{

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

}

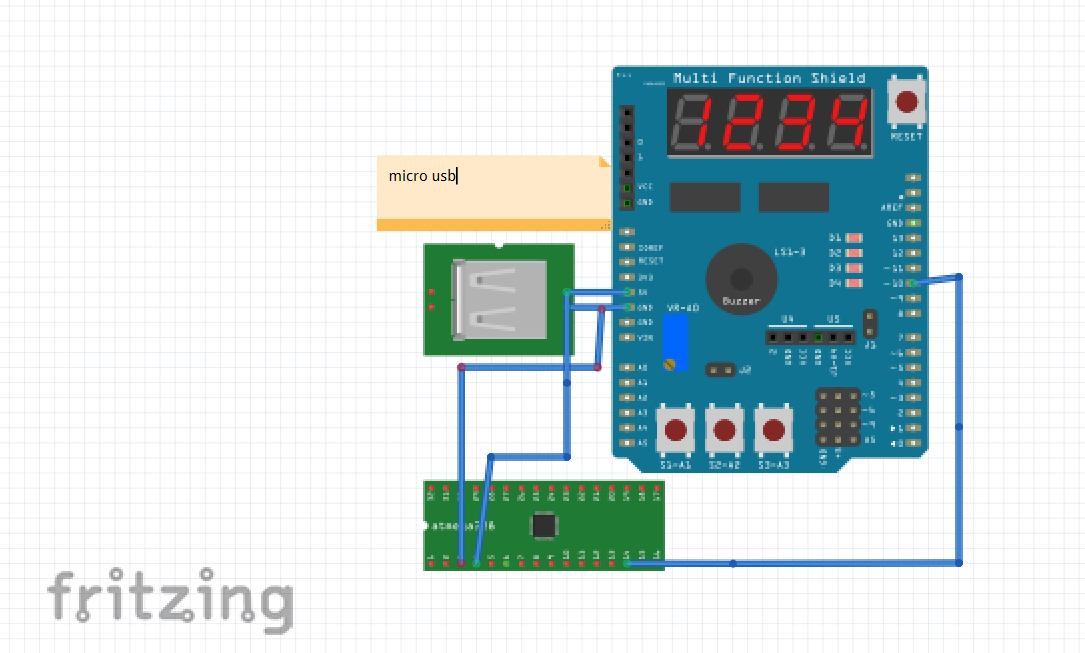
}

return 0;

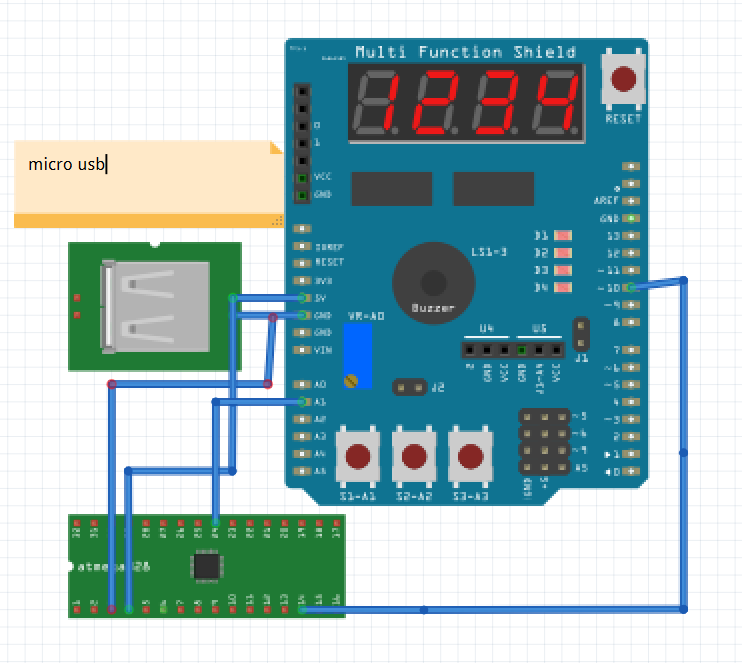
}

1. **SCHEMATICS**

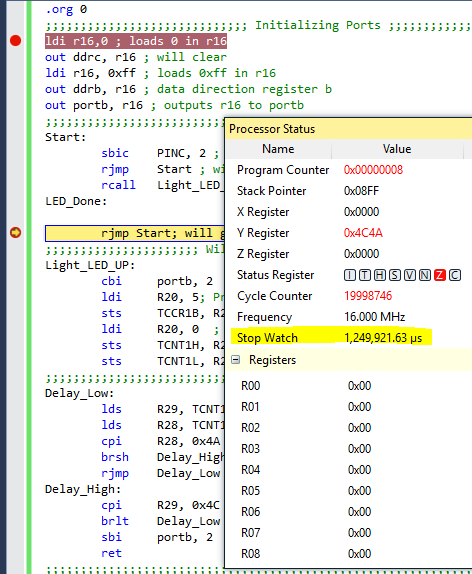
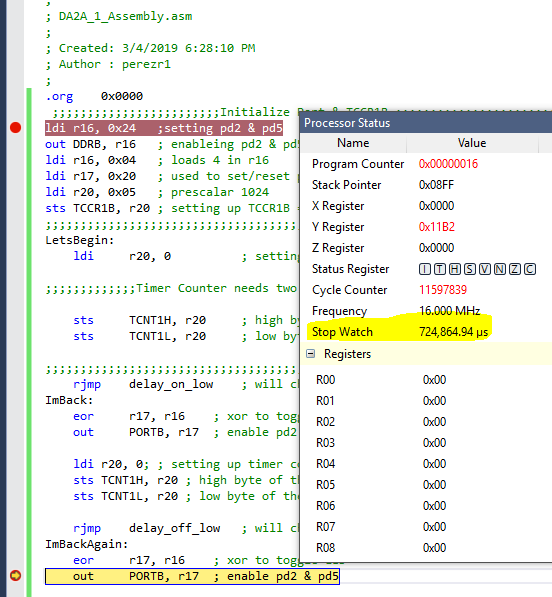
Task 1:



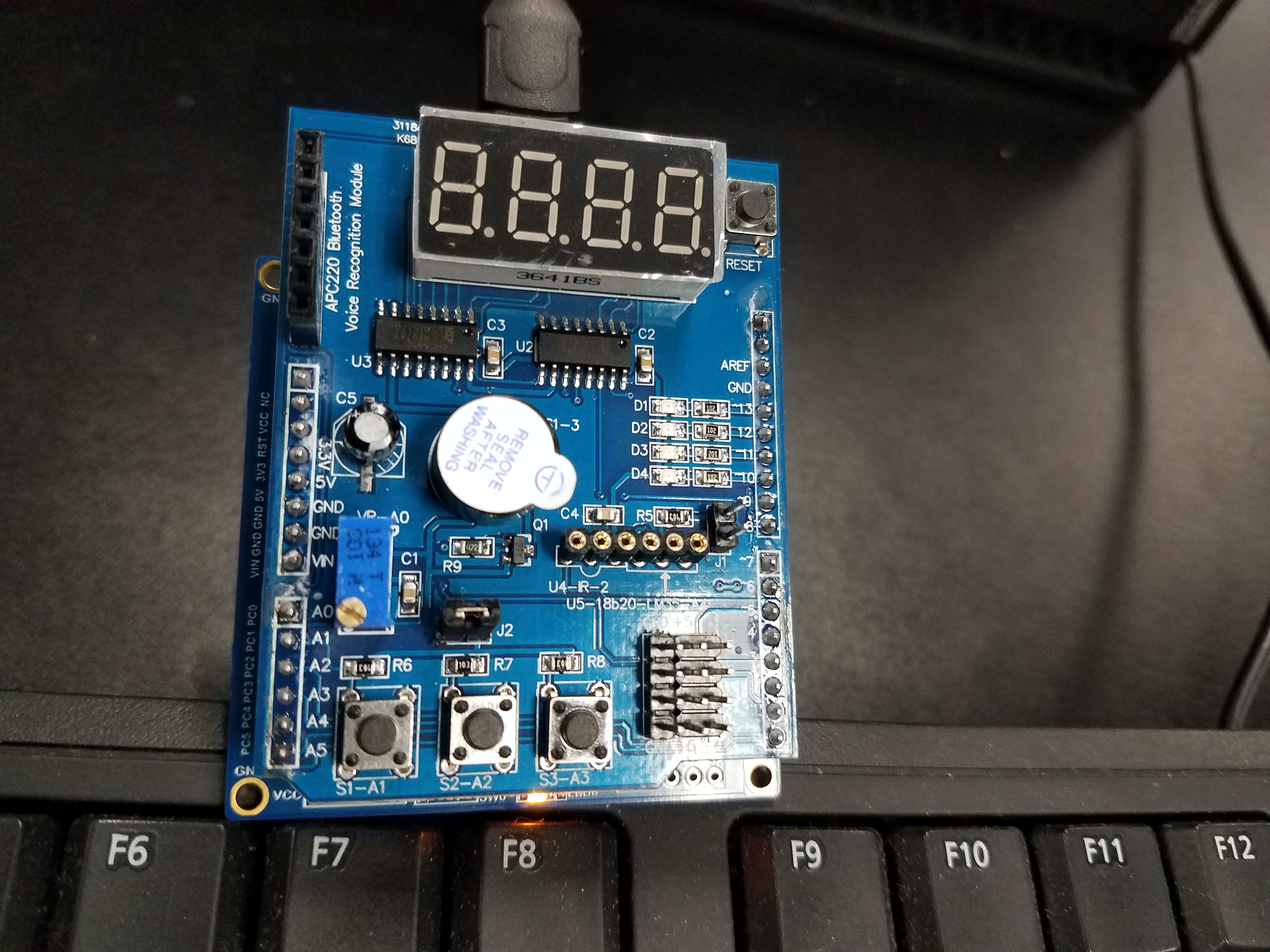
Task 2:



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



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



1. **VIDEO LINKS OF EACH DEMO**

Task 1 (Assembly): <https://www.youtube.com/watch?v=ZVinzFpczrM>

Task 1 (C code): <https://www.youtube.com/watch?v=BmeGOI4tsJ8>

Task 2 (Assembly): <https://www.youtube.com/watch?v=xkjkshdlT2I>

Task 2 (C code): <https://www.youtube.com/watch?v=tgYo88D9zp0>

1. **GITHUB LINK OF THIS DA**

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

**Student Academic Misconduct Policy**

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

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

RICKY PEREZ