CPE301 – SPRING 2019

Design Assignment 2A

Student Name: Ron Joshua Recrio

Student #: 5003825419

Student Email: recrio@unlv.nevada.edu

Primary Github address: https://github.com/recrio/submissions

Directory: /DesignAssignments/DA2A

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**

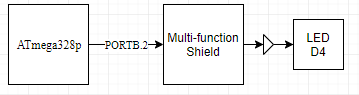
List of Components used:

ATmega328p Xplained Mini

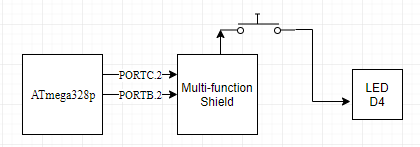
Multifunction Shield

Block diagram with pins used in the Atmega328P

Task 1:



Task 2:



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

.ORG 0x0000

init:

LDI R16, 4 ; 0b000100 or PB2

OUT DDRB, R16 ; PB2 set as an output

LDI R17, 0 ; Used for toggling PB2

LDI R20, 5 ; 0b101 for 1024 prescaler

STS TCCR1B, R20 ; prescaler set

LDI R20, 0 ; to be used for TCNT1 reset

main:

RCALL TCNT1\_RESET ; reset TCNT1

RCALL delay\_ON ; call for .435 second delay

EOR R17, R16 ; XOR toggles the LED

OUT PORTB, R17 ; should turn off the LED

RCALL TCNT1\_RESET ; reset TCNT1

RCALL delay\_OFF ; call for .290 second delay

EOR R17, R16 ; toggle LED again

OUT PORTB, R17 ; should turn on the LED

RJMP main ; repeat

TCNT1\_RESET:

STS TCNT1H, R20 ; timer/counter

STS TCNT1L, R20 ; reset

RET

check\_high\_ON:

CPI R27, 0x1A ; check if high byte of timer matches 0x1A

BRLT delay\_ON ; if not, recheck timer

RET ; return to main

delay\_OFF: ; TCNT should be 4531 or 0x11B3

LDS R27, TCNT1H ; R27 = high byte of timer/counter

LDS R26, TCNT1L ; R26 = low byte of timer/counter

CPI R26, 0xB3 ; check if low byte of timer matches 0x8C

BRSH check\_high\_OFF ; if low bytes match, check high byte next

RJMP delay\_OFF ; repeat until matching

check\_high\_OFF:

CPI R27, 0x11 ; check if high byte of timer matches 0x1A

BRLT delay\_OFF ; if not, recheck timer

RET ; return to main

delay\_ON: ; TCNT should be 6796 or 0x1A8C

LDS R27, TCNT1H ; R27 = high byte of timer/counter

LDS R26, TCNT1L ; R26 = low byte of timer/counter

CPI R26, 0x8C ; check if low byte of timer matches 0x8C

BRSH check\_high\_ON ; if low bytes match, check high byte next

RJMP delay\_ON ; repeat until matching

1. **VERIFICATION OF TASK 1 USING C**

#include <stdio.h>

#include <avr/io.h>

int main(void)

{

DDRB = (1<<2); //Make PB2 Output

TCCR1B = 5; // set prescaler to 1024

while (1)

{

TCNT1 = 0; // set timer/counter to 0

while (TCNT1 != 6796) {

//do nothing

}

PORTB ^= (1<<2); // toggle PB2 using xor

TCNT1 = 0; // reset again

while (TCNT1 != 4531) {

// do nothing

}

PORTB ^= (1<<2); // toggle PB2 using xor

}

}

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

.ORG 0

init:

CBI DDRC, 2 ; PC2 as input, Clears DDRC2

LDI R16, 4 ; PB2

OUT DDRB, R16 ; SET PB2 as output

LDI R17,0 ; toggle for PC2

LDI R20,5 ; 0b101 for prescaler 1024

STS TCCR1B, R20 ; Set prescaler to 1024

LDI R20, 0 ; For counter/timer reset

EOR R17, R16 ; Toggle with XOR so that it starts as OFF

main:

OUT PORTB, R17 ; Turn off LED

SBIC PINC, 2 ; Skip if PC2 is pressed

RJMP keep\_checking ; if not then jump

EOR R17, R16 ; Toggle again

OUT PORTB, R17 ; Turn on LED

EOR R17, R16 ; Toggle back

STS TCNT1H, R20 ; Reset

STS TCNT1L, R20 ; Timer/Counter

RCALL delay ; call delay of 1.25s

RJMP main ; Loop again

keep\_checking:

SBI PORTB, 2 ; PB2 = 1, set it high so it will not light up

RJMP main ; Loop back to check for button press

delay: ; TCNT1 should be 19,531 or 0x4C4B

LDS R27, TCNT1H ; high byte of timer/counter

LDS R26, TCNT1L ; low byte of timer/counter

CPI R26,0x4B ; check if low byte matches

BRSH check\_high ; if matching, then check high byte

RJMP delay ; else keep checking

check\_high:

CPI R27,0x4C ; check high byte

BRLT delay ; if not matching , keep checking

RET ; if timer/counter is finally matching, return

1. **VERIFICATION USING C OF TASK 2**

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

#include <avr/io.h>

#include <util/delay.h>

int main(void)

{

DDRB |= (1<<2); //PB2 is an output, XXXX X1XX

PORTB |= (1<<2); //PB2 set as high to turn off LED, XXXX X1XX

DDRC &= (0<<2); //PC2 is an input, XXXX X0XX

PORTC |= (0<<2); //PC2 set as low or unpressed, XXXX X0XX

while (1) {

if (!(PINC & (1 << PINC2))){ // If button is pressed

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

*\_delay\_ms*(1250); // delay for 1250ms

}

else { // if button is not pressed

PORTB |= (1<<2); // set PB2 to high or LED off

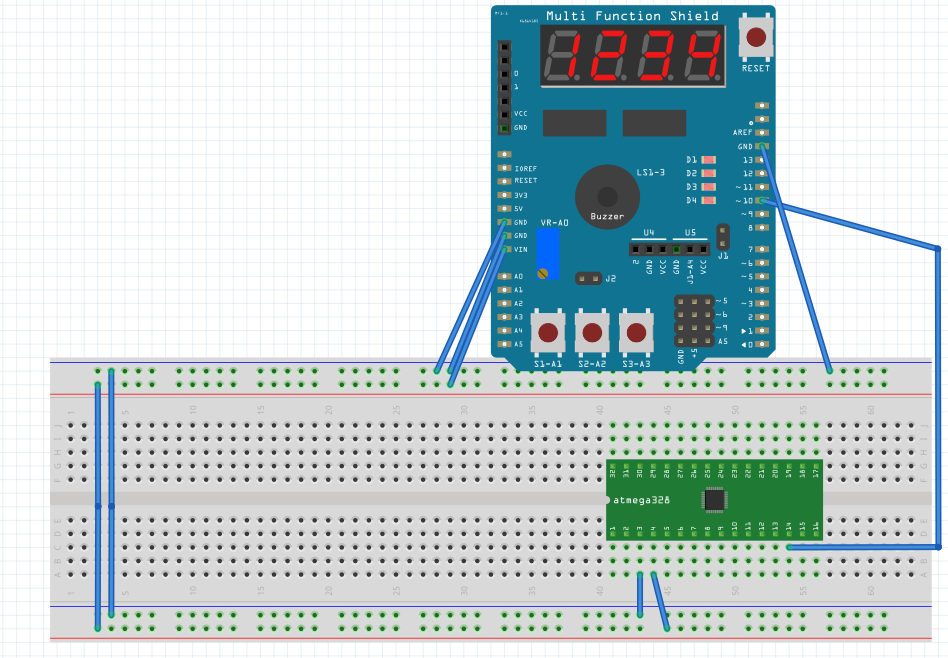
}

}

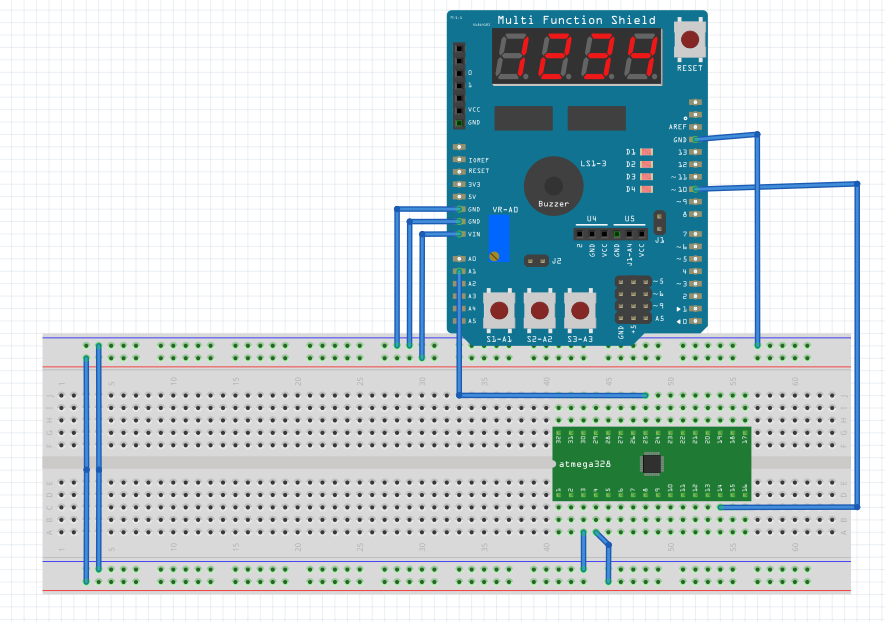
}

1. **SCHEMATICS**

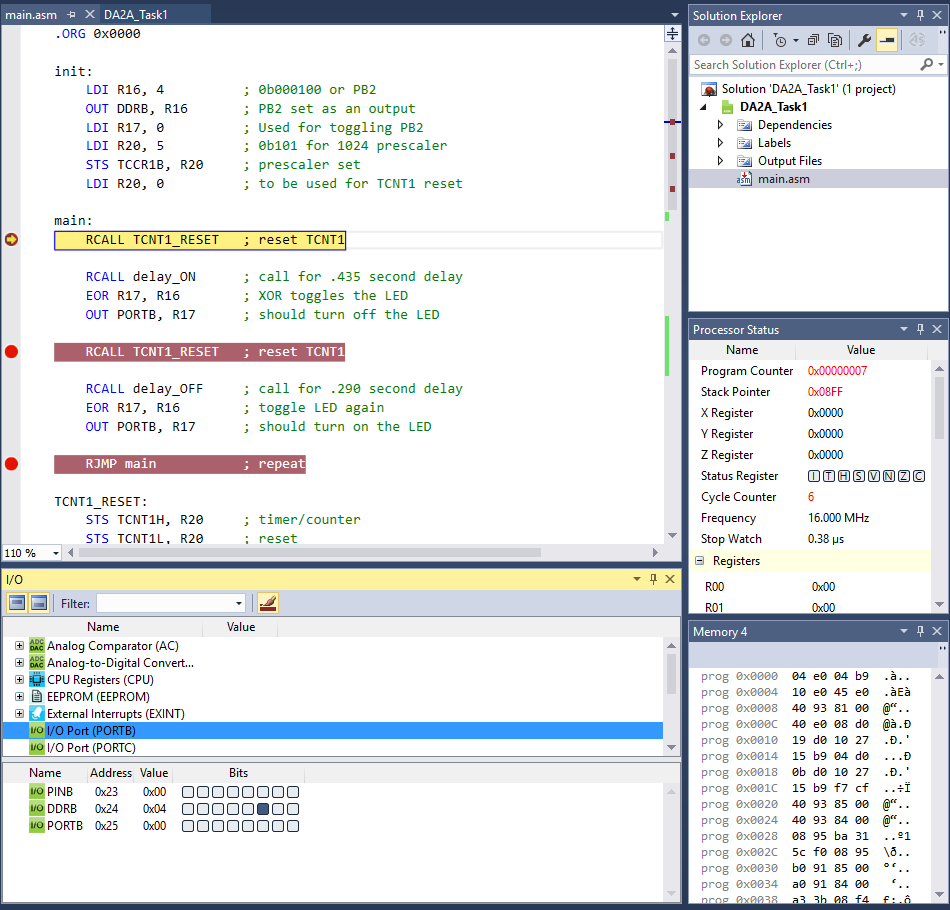
**Task 1:**



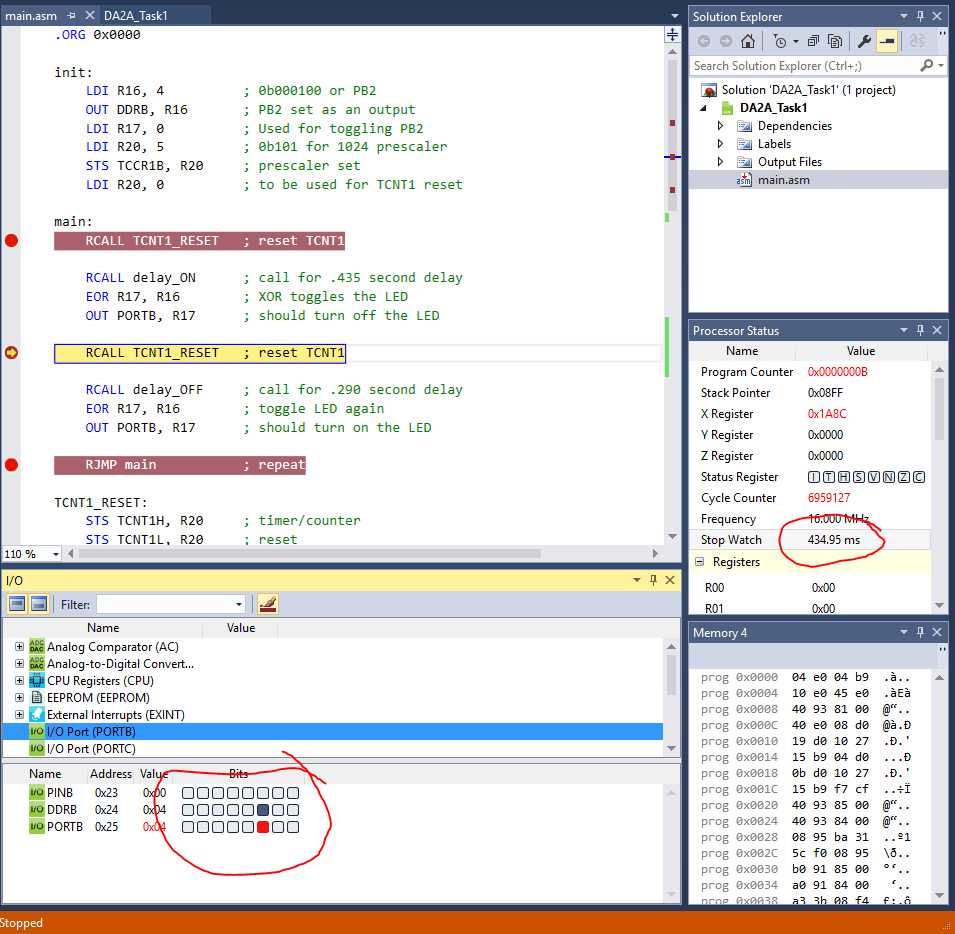
**Task 2:**



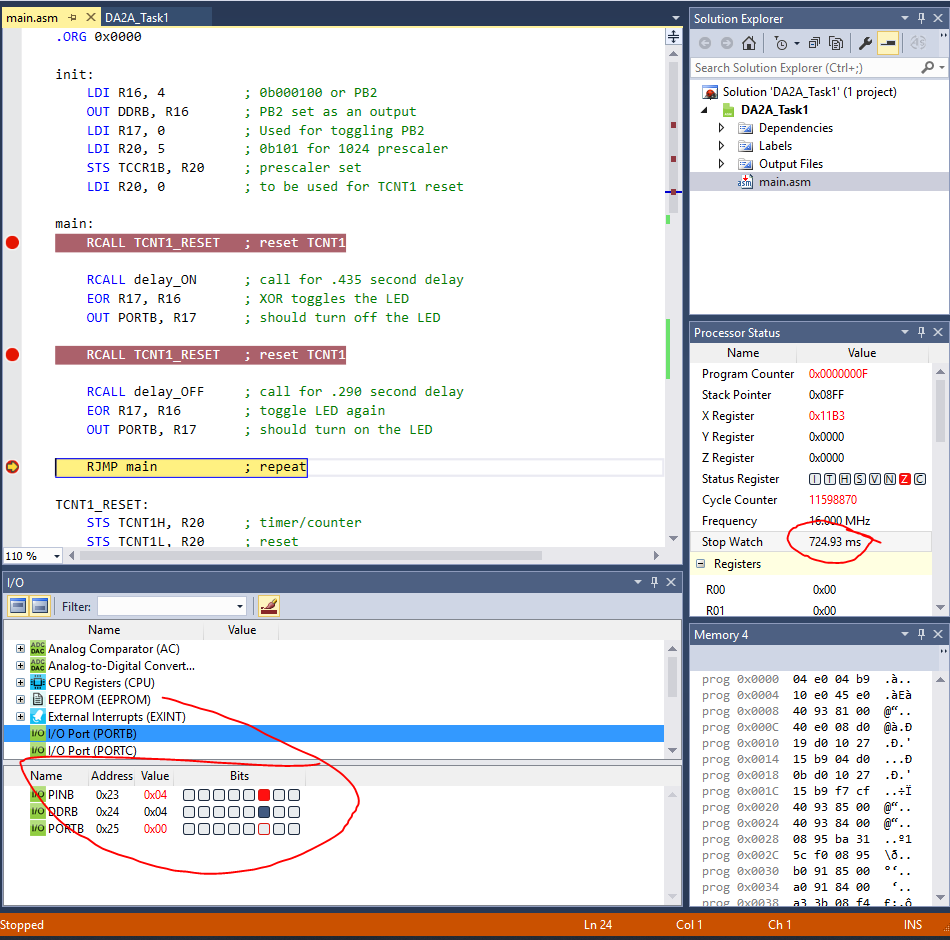
1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

**TASK 1:** Initial State (Note that PORTB2 as 0 means LED is turned ON)

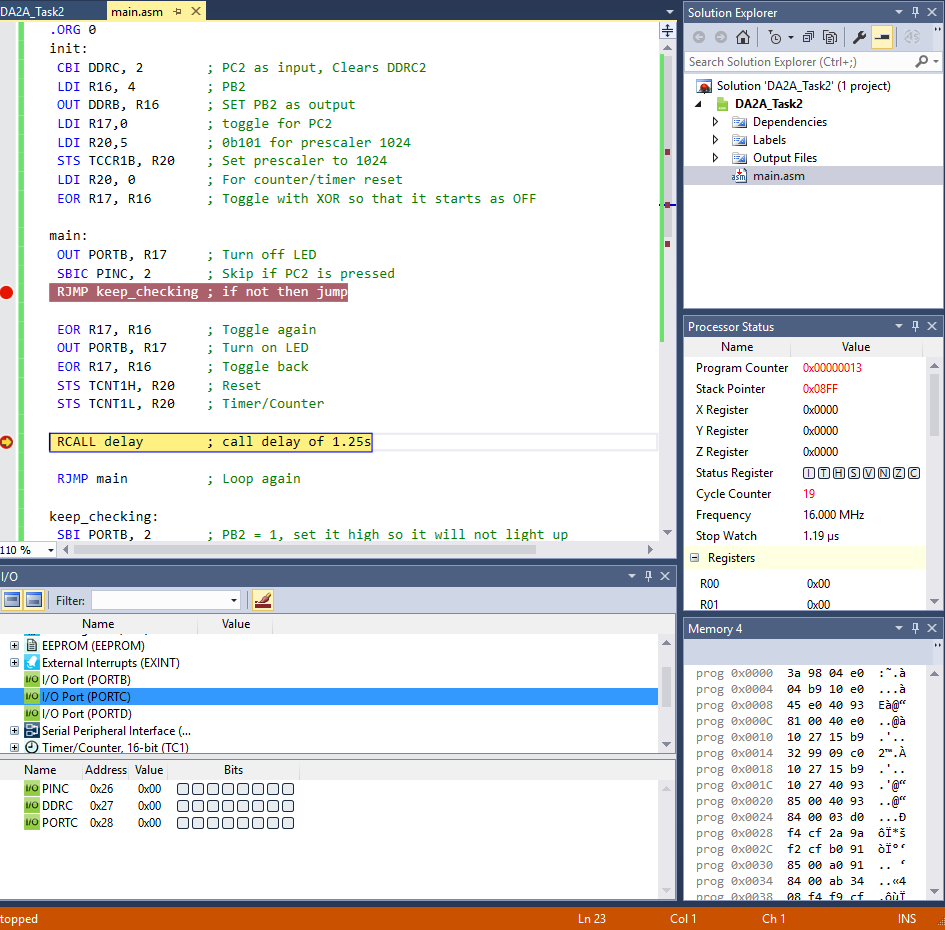
**TASK 1:** After .435 Sec (PORTB2 is 1 which means LED is OFF)



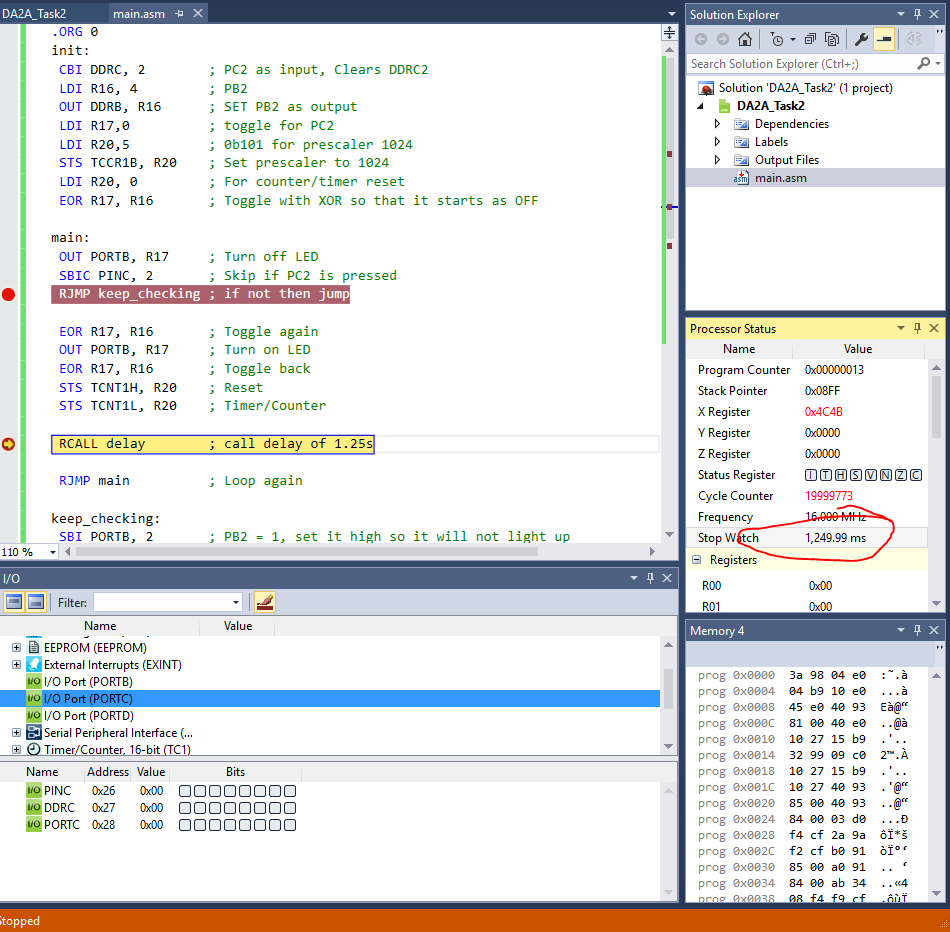
**TASK 1:** After .290 Sec



**TASK 2: INITIAL BUTTON = 0 (or PRESSED)**

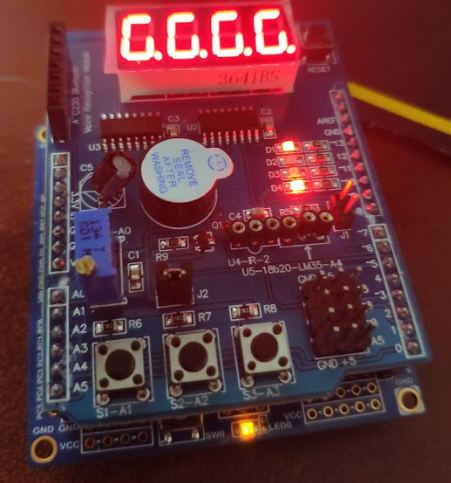


**TASK 2: AFTER DELAY**

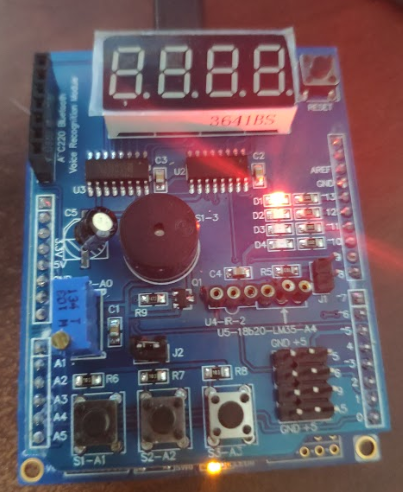
****

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

**TASK 1:**



**TASK 2:**

****

1. **VIDEO LINKS OF EACH DEMO**

DA2A Task 1- <https://youtu.be/6_uJ5-JgL44>

DA2A Task 2 - <https://youtu.be/mwjTFpltsVw>

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”.

Ron Joshua Recrio