CPE301 – SPRING 2019

Design Assignment 2A

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\DA2A

NOTE: I had some issues with the Xplained board regarding the assembly and C code for number 2, so instead, I got it working on my LAB development board and used that. Thank you

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

For number 1 (Assembly),

PORTB connection => Resistor => PB2 LED

For number 1 (C),

PORTB connection => Resistor => PB2 LED

For number 2 (Assembly),

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

For number 2 (C),

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

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

Assembly code for 1:

.org 0

LDI R16, 4 // Used to light up LED(PB2)

OUT DDRB, R16 // PB2 LED is output

LDI R17, 0 // For set or reset of LED (PB2)

LDI R18, 5 // Clock prescaler is set to 1024

STS TCCR1B, R18 // Store prescaler to data space

Loop1:

LDI R18, 0x00 // Reset the counter to 0

STS TCNT1H, R18 // Store high bits from R18

STS TCNT1L, R18 // Store low bits from R18

RCALL DelayON

EOR R17, R16 // XOR to toggle LED

OUT PORTB, R17 // Output value of R17 on PORTB

LDI R18, 0x00 // Reset the counter to 0

STS TCNT1H, R18 // Store high bits from R18

STS TCNT1L, R18 // Store low bits from R18

RCALL DelayOFF

EOR R17, R16 // XOR to toggle LED

OUT PORTB, R17 // Output value of R17 on PORTB

RJMP Loop1 // Repeat Loop1

DelayOFF:

LDS R19, TCNT1H // Load upper bits of counter to R19

LDS R20, TCNT1L // Load lower bits of counter to R20

CPI R20, 0xB2 // Compare lower 8 bits of counter to R20

BRSH LoopOFF // If the lower bits are equal to inputted amount, go to DelayOFF

RJMP DelayOFF

LoopOFF:

CPI R19, 0x11 // Check upper bits to inputted amount

BRLT DelayOFF // Branch if less than the inputted amount

RET // Return from Subroutine

DelayON:

LDS R19, TCNT1H // Load upper bits of counter to R19

LDS R20, TCNT1L // Load lower bits of counter to R20

CPI R20, 0x8C // Compare lower 8 bits of counter to R20

BRSH LoopON // If the lower bits are equal to inputted amount, go to upper bits

RJMP DelayON

LoopON:

CPI R19, 0x1A // Check upper bits to inputted amount

BRLT DelayON // Branch if less than the inputted amount

RET // Return from Subroutine

C code for 1:

#include <avr/io.h>

#define *F\_CPU* 16000000UL // Frequency board running

#include <util/delay.h>

int main(void)

{

DDRB |= (1<<PB2); // Define digital pin13/PORTB5 as an output so we can blink our led

while(1) // Create infinite loop

{

PORTB |= (1<<PB2); // Turn led on

*\_delay\_ms*(435); // Wait 0.435 second

PORTB &= ~(1<<PB2); // Turn led off

*\_delay\_ms*(300); // Wait 0.300 second

}

}

Assembly code for 2:

.ORG 0

LDI R17, 5 // For prescaler to be 1024

LDI R18, 4 // For PC2 be input

LDI R24, 0 // For turning LED off

STS TCCR1B, R17 // Set clock prescaler

SBI DDRB, 2 // P02 is set to output

INPUT: // Input loop

IN R20, PINC // PINC is set to input

AND R20, R18 // OR value of R20 and R18

CP R20, R18 // Compare values or R20 and R18

BREQ START // Branch if equal to START LOOP

RJMP INPUT // Jump back to INPUT if values don't match

START: // Start loop to load high and low values

LDI R21, 0 // Used to set R21 back to 0

STS TCNT1H, R21 // Load high bits from R21

STS TCNT1L, R21 // Load low bits from R21

SBI PORTB, 2 // PB2 is output

LED\_ON1:

LDS R22, TCNT1H // Load upper bits to R22

LDS R23, TCNT1L // Load lower bits to R21

CPI R23, 0x25 // Compare lower bits

BRSH LED\_ON2 // Branch to LED\_ON2 if lower bits exceed value

LED\_ON2:

CPI R22, 0x26 // Compare high bits

BRLT LED\_ON1 // Branch back to LED\_ON1 if less than given value

RESTART:

OUT PORTB, R24 // PORTB is OFF

RJMP INPUT // Jump back to INPUT loop, ready for button to be pressed

C code for 2:

#include <avr/io.h>

#include <util/delay.h>

int main(void)

{

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

TCCR1B = (1<<CS12) | (1<<CS10); // Setting prescaler to 1024 on counter 1 (binary = 101 => 1024)

while(1) // Infinite loop

{

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

{

TCNT1 = 0; // Counter set to zero

while (TCNT1 < 0x2625) // Loop as long as counter is less than TCNT calculated value

{

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

}

}

else

{

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

}

}

return 0;

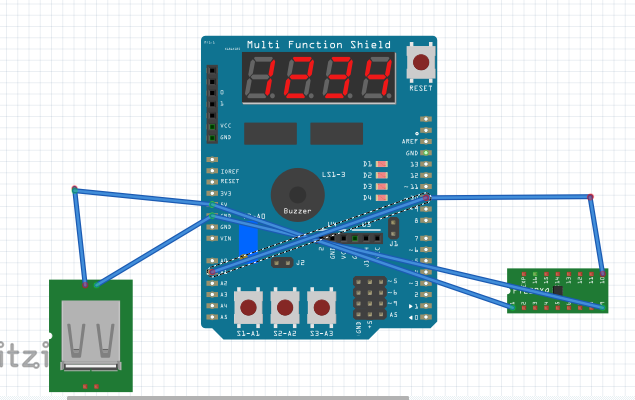
}

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

All code is in number 2.

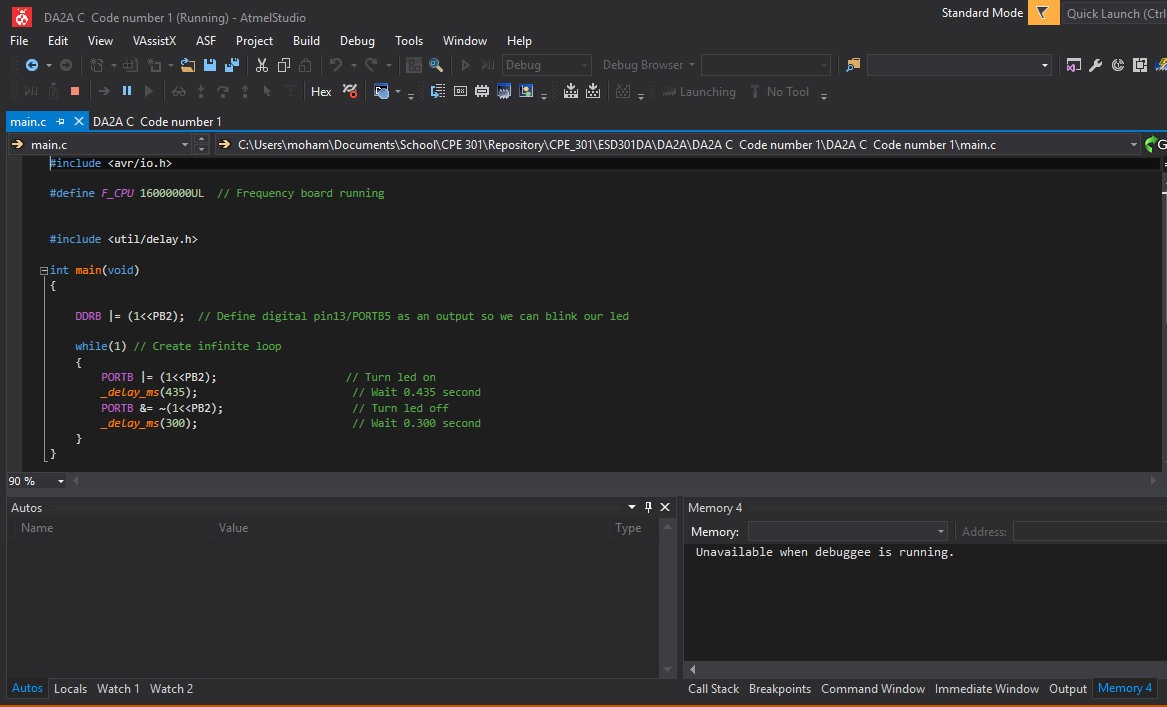
1. **SCHEMATICS**

Highlighted connection is for number 2 including switch. Without that connection, it is for number 1.

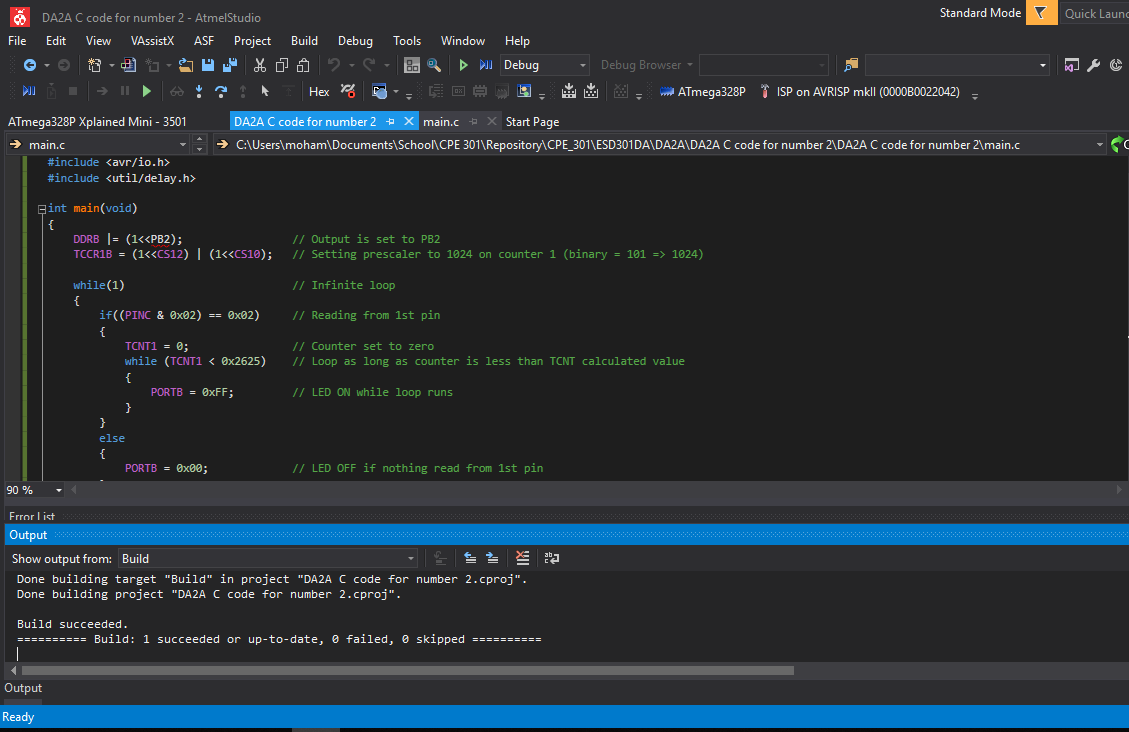


1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

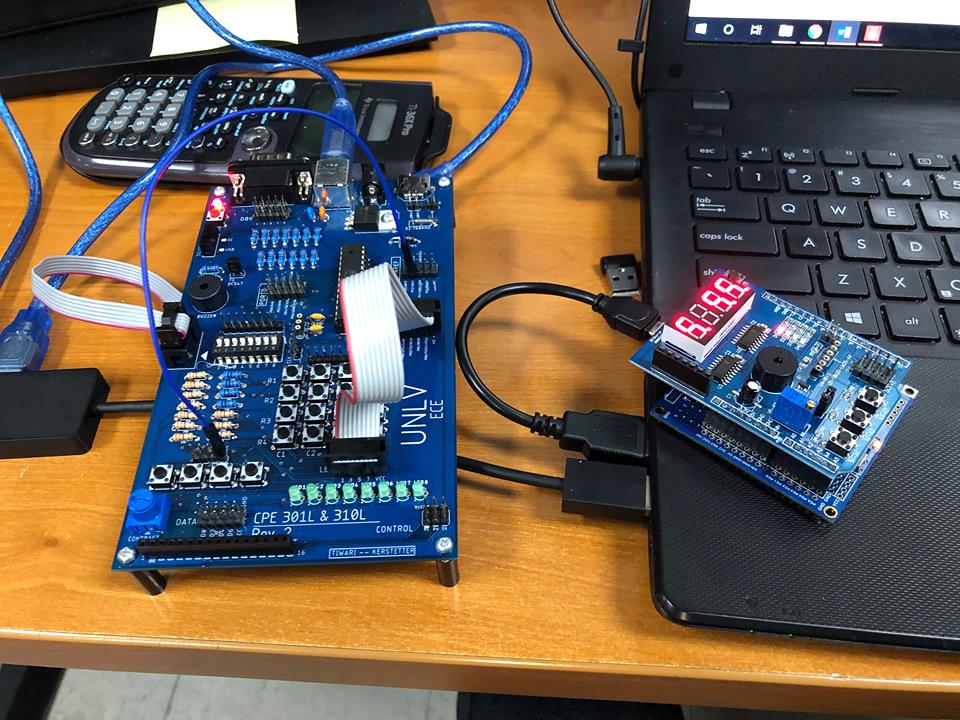
C for number 1



C for number 2



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

****

1. **VIDEO LINKS OF EACH DEMO**

Assembly for 1:

<https://www.youtube.com/watch?v=9yT5PHNzeFY&feature=youtu.be&fbclid=IwAR2dJe46ljY7DJv1n6lFRC6201aZX1EZf1oD6ds45DUkTkTv59p-_mp9ol4>

C for 1:

<https://www.youtube.com/watch?v=zTkdThMMLAA&feature=youtu.be&fbclid=IwAR3U9xvPV3lH2zuWnG1VlBgmzbCFrNSeFCYVlB0DIFbs0J_oeT8xZnfoAMA>

Assembly for 2:

<https://www.youtube.com/watch?v=So4Vuc5SbJs&feature=youtu.be&fbclid=IwAR2EWtQX0V9yTulq_tk1KFH9vx7bM9F_n_71B5wviVQwSOquXS8nT9i6LPY>

C for 2:

<https://www.youtube.com/watch?v=d0TKvKUDbNI&feature=youtu.be&fbclid=IwAR2c_grhoD4OoNrINykCd3KL9x9-pYRWt-hsLm077vrzByYA_ZrtR0nt7qM>

1. **GITHUB LINK OF THIS DA**

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

This assignment submission is my own, original work.

MOHAMAD JUNDI