CPE301 – FALL 2019

Design Assignment 2C

Student Name: Dillon Archibald
Student #: 5004439916
Student Email: Archid1@unlv.nevada.edu
Primary Github address:
Directory: https://github.com/Dil-bert/Alabaster.git

Submit the following for all Labs:

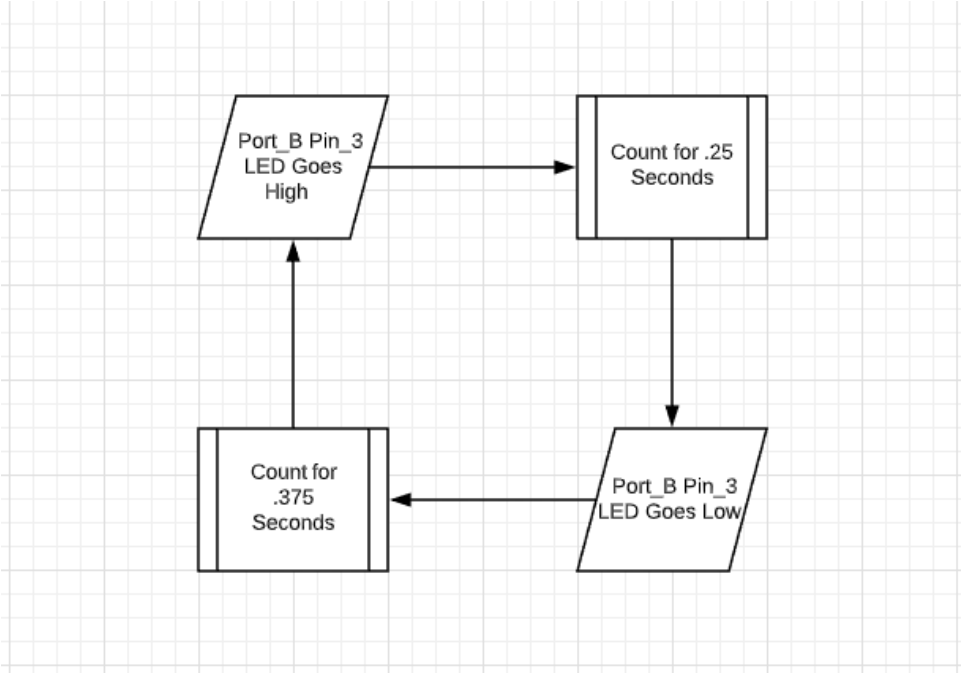
1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder 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).

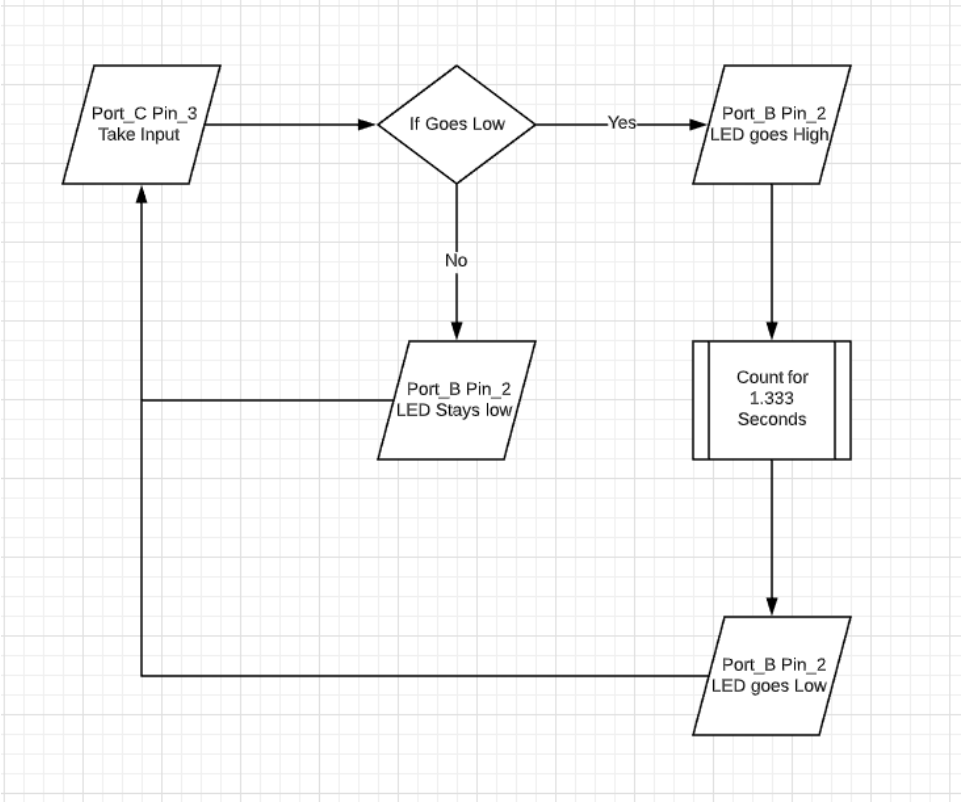
1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

List of Components used

- 1xAtmega328P
- 1xBreadboard
- 1xLED
- 4xWires

Block diagram with pins used in the Atmega328P





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

N/A

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

```
/*
* DA2c_1_a.c
*
* Created: 10/16/2019 8:37:25 PM
* Author : Dilbert
*/

#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
   unsigned short count = 0;

   DDRB |=(1<<DDB2);    // Set PortB.2 as an output
   PORTB = (0<<DDB2);    // Set PortB.2 Low (LED off)
   TCCR0A = 0;           // Timer0, Normal mode, internal clk
```

```
   TCNT0 = 0X00;       // Setting the count register to 0[count goes from 0-255]
   TCCR0B |= 0X05;       // Setting the Prescaler to 1024 and starting the timer

   while (1) {
      PORTB ^=(1<<DDB2);                  // Toggle PortB.2 (Turn on)
      while(count != 15){                 // Run timer 81 times
         while((TIFR0 & 0X01) == 0){};    // run timer until overflow
            TCNT0 = 0X00;                 // reset the timer count
            TIFR0 = 0X01;                // reset the tiemr over flow flag
            count = count + 1;            // increment the count of timer overflows
         }
      count = 0;                         // Reset the count of overflows
      PORTB ^=(1<<DDB2);                  // Toggle PortB.2 (Turn off)
      while(count != 23){                 // Run timer 81 times
         while((TIFR0 & 0X01) == 0){};    // run timer until overflow
         TCNT0 = 0X00;                    // reset the timer count
         TIFR0 = 0X01;                   // reset the tiemr over flow flag
         count = count + 1;               // increment the count of timer overflows
         }
      count = 0;                         // Reset the count of overflows
   }
}


/*
* DA2c_1_b.c
*
* Created: 10/12/2019 6:16:22 PM
* Author : Dilbert
*/
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
   unsigned short count = 0; // Counting variable

   DDRC |=(0<<DDC3);    // Set PortC.3 as an input
   PORTC =(1<<DDC3);    // Set PortC.3 pull up High
   DDRB |=(1<<DDB2);    // Set PortB.2 as an output
   PORTB =(0<<DDB2);    // Set PortB.2 Low (LED off)

   TCCR0A = 0;          //Timer0, Normal mode, internal clk
   TCNT0 = 0X00;       //Setting the count register to 0[count goes from 0-255]
   TCCR0B |= 0X05;       //Setting the Prescaler to 1024 and starting the timer
```

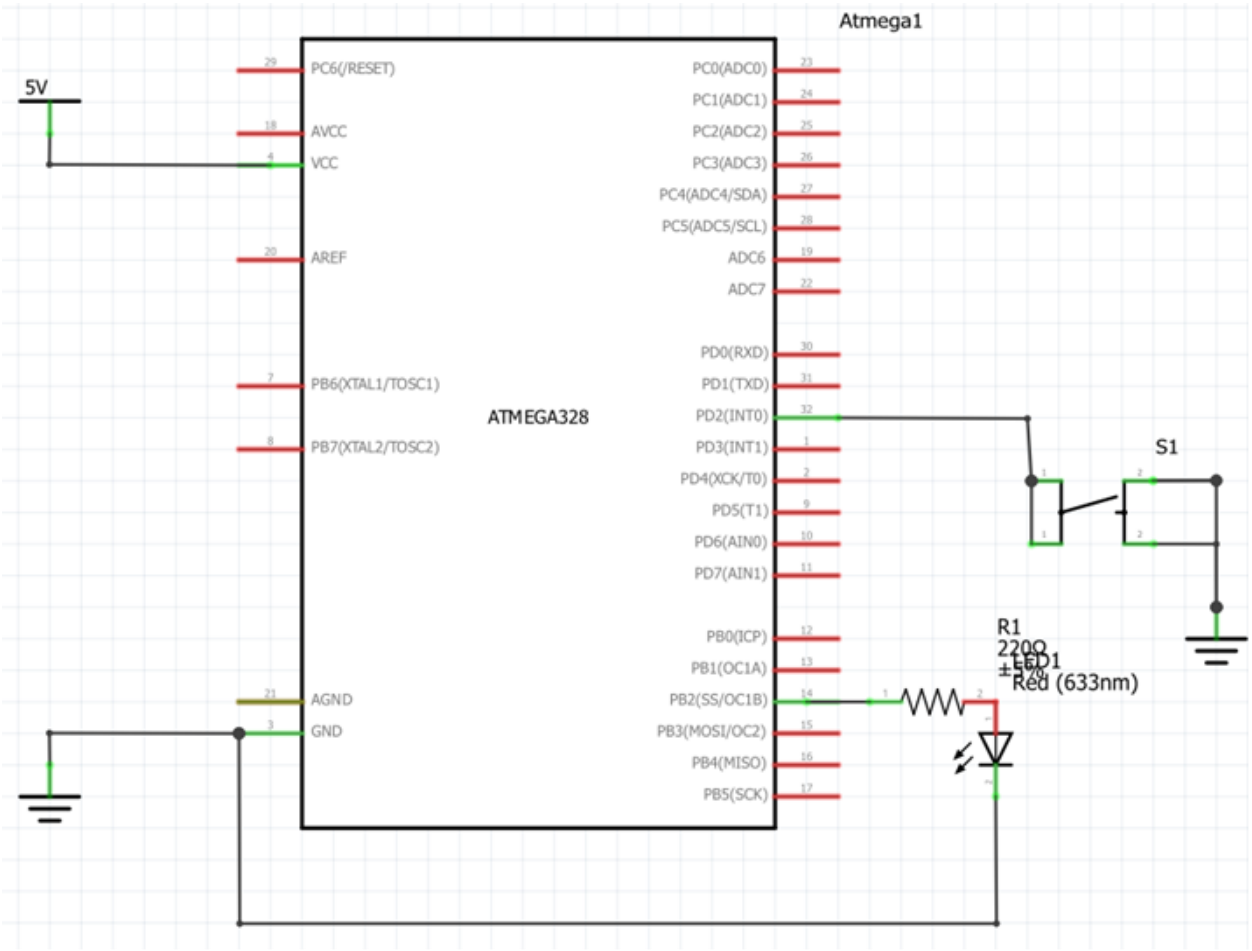
```
   while (1) {
      if(!(PINC & (1 << 3))){                  //If input PinC.3 == 0
         PORTB ^=(1<<DDC2);                    //Toggle PortB.2 (Turn on)
         while(count != 81){                   //Run timer 81 times
            while((TIFR0 & 0X01) == 0){};    // run timer until overflow
            TCNT0 = 0X00;                     //reset the timer count
            TIFR0 = 0X01;                     //reset the tiemr over flow flag
            count = count + 1;                //increment the count of timer overflows
            }
         PORTB ^=(1<<DDB2);                   //Toggle PortB.2 (Turn off)
         }
      count = 0;                              //Reset the count of overflows
      }
}
/*
* DA2c_2_a.c
*
* Created: 10/16/2019 8:57:56 PM
* Author : Dilbert
*/

#define F_CPU 16000000UL

#include <avr/io.h>
#include <avr/interrupt.h>

unsigned short county;        // Global counter variable
int main(void)
{
   county = 0;               // Set count to zero
   DDRB = 1<<2;              // Set PortB.2 as an output
   PORTB = 1<<2;             // Set PortB.2 high (Turn on LED)

   TCNT0 = 0X00;             // Clear the timer count register
   TIMSK0 = (1<<TOIE0);    // Turn on the timer overflow interrupt
   TCCR0A = 0X00;            // Set the Timer control register.A = 0x00 (Normal mode)
   TCCR0B = 0X05;            // Set the Timer control register.B = 0x00 (Prescaler 1024)
   sei();                    // Global interrupts Enable
   while(1){
      if(county == 15){    // If count equals 15 (40% of period)
         PORTB = 0<<2;    // Set PortB.2 Low (Turn off LED)
      }
      if(county == 38){    // If count equals 38 (end of period)
         PORTB = 1<<2;    // Set PortB.2 High (Turn on LED)
         county = 0;        // Reset Counter to 0
      }
```

```
   };
}

ISR (TIMER0_OVF_vect){        // Overflow Interrupt service request function
   county = county + 1;    // Count increment Plus one
   TCNT0 = 00;              // Reset the timer count register to 0x00
}
/*
* DA2c_2_b.c
*
* Created: 10/15/2019 8:52:37 AM
* Author : Dilbert
*/
#define F_CPU 16000000UL

#include <avr/io.h>
#include <avr/interrupt.h>

unsigned short county;        // Global Count Variable
int main(void)
{
   DDRB = 1<<2;           // Set PortB.2 as an output
   PORTB = 0<<2;           // Set PortB.2 Low (LED off)
   DDRC = 1<<3;           // Set PortC.3 as an input
   PORTC = 1<<3;           // Set PortC.3 Pull up high

   county = 0;              // Set global count to 0

   TCNT0 = 0X00;           // Set the Timer count register to zero
   TIMSK0 = (1<<TOIE0);    // Set the Timer Overflow interrupt enabled
   sei();                  // Global interrupt enable
   while(1){
      if(county == 82){        // If Count = 82
         county = 0;           // Set global counter to 0
         PORTB = 0<<2;        // Set PortB.2 Low (LED off)
      }
      if(!(PINC & (1 << 3))){    // If PortC.3 Goes low (Button pressed)
         county = 0;           // Set global counter to 0
         PORTB = 1<<2;        // Set PortB.2 High (LED on)
         TCCR0A = 0X00;        // Set Timer Control Register.A = 0x00 (Normal Mode)
         TCCR0B = 0X05;        // Set Timer Control Register.B = 0x05 (Prescaler 1024)
      }
   };
}
ISR (TIMER0_OVF_vect){        // Timer Overflow interrupt Service Request
   county = county + 1;    // Increment Count + 1
```

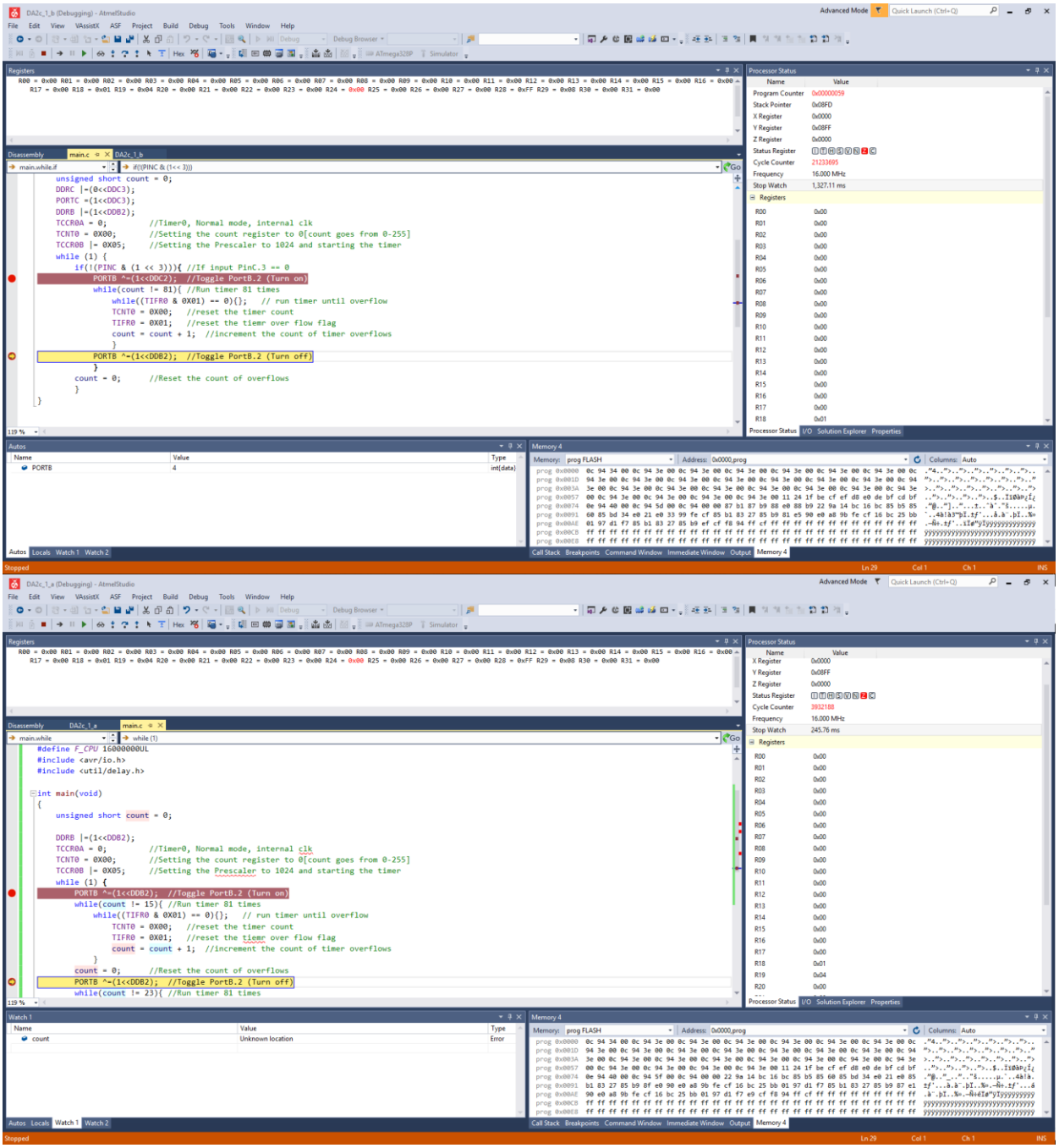
```
   TCNT0 = 00;              // Reset the timer count register to zero
}

/*
* DA2c_3_a.c
*
* Created: 10/16/2019 10:01:39 PM
* Author : Dilbert
*/

#include <avr/io.h>

#define F_CPU 16000000UL

#include <avr/io.h>
#include <avr/interrupt.h>

unsigned short county;

int main(void){
      county = 0;              // Global count variable set to zero
   DDRB = (0<<2);           // Set PortB.2 as an output
      PORTB = (1<<2);          // Set PortB.2 high (Turn on LED)
   OCR0A = 0XFF;            // Set the output compare register to 0xFF
      TCCR0A = 0X02;           // Set the Timer compare register.B WGM01 = 1 (CTC mode)
      TCCR0B = 0X05;           // Set the Timer compare register.A CS02 & CS00 = 1 (Prescaler
1024)
      TIMSK0 = (1<<OCIE0A);    // Set the Timer mask to ouptut the compare flag interrupt
      sei();                   // Global interrupt enable
   while (1)
   {
       if(county == 15){    // If the count goes to 15 enter (after 40% of period)
          PORTB = (0<<2);    // Set PortB.2 Low (turn off LED)
       }
       if(county == 38){    // If the count goes to 38 enter (after completion of period)
          PORTB = (1<<2);    // Set PortB.2 High (turn on LED)
          county = 0;        // Set count to zero
       }
   }
}

ISR (TIMER0_COMPA_vect){    // Compare interrupt service request function
   county = county + 1;    // Count increment + 1
   TIFR0 |= (1<<OCF0A);    // Reset the compare flag interrupt register
}
```

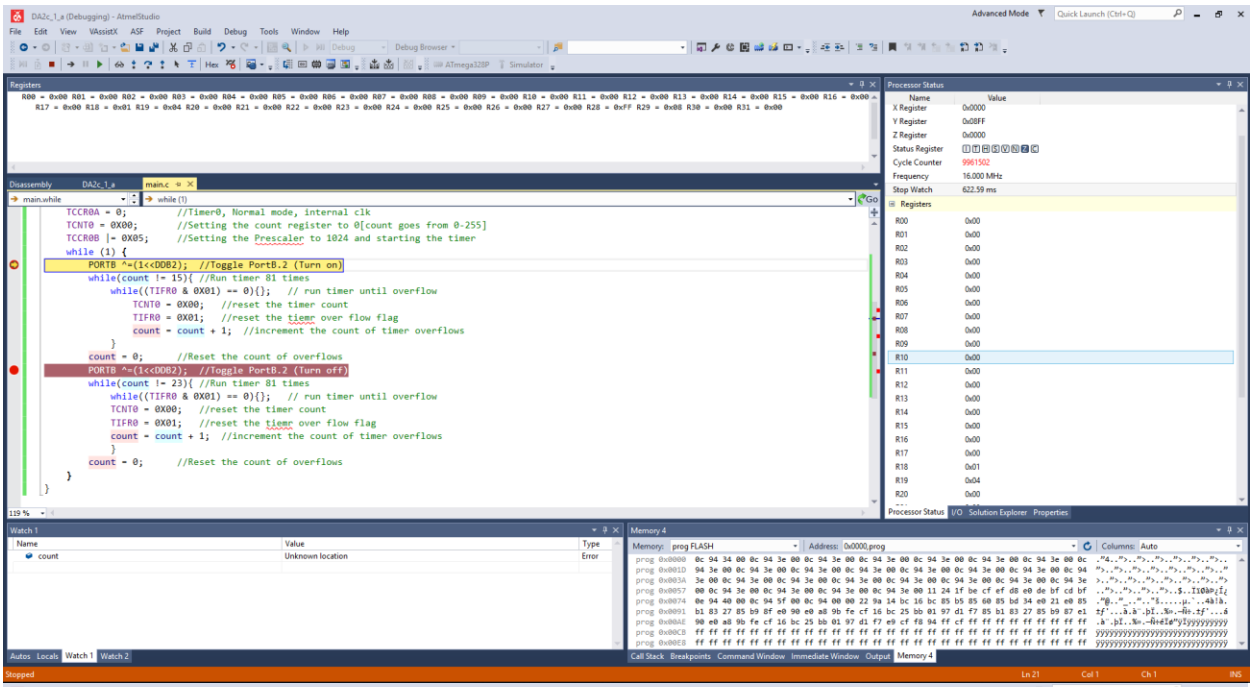
```
/*
* DA2c_3_b.c
*
* Created: 10/16/2019 10:55:44 PM
* Author : Dilbert
*/

#define F_CPU 16000000UL

#include <avr/io.h>
#include <avr/interrupt.h>

unsigned short county;

int main(void){                    //----------------------------------------
   DDRB = (1<<2);       // SET B2 AS OUTPUT
   PORTB = (0<<2);       // SET B2 OUTPUT LOW
   DDRC = (0<<3);       // SET C3 AS INPUT
   PORTC = (1<<3);       // SET C3 PULL UP RESISTOR HIGH
     county = 0;          // GLOBAL COUNT VARIABLE = 0

   while (1)
   {
      if(!(PINC & (1 << 3))){        // If pinC.3 goes low
         PORTB = (1<<DDB2);        // Toggle PortB.2 (Turn on)
         county = 0;               // Ensure Count variable is zero
           OCR0A = 0XFF;           // Set the compare register to 0xFF
           TCCR0A = 0X02;           // Control RegisterA WGM01 = 1 (CTC mode)
           TCCR0B = 0X05;           // Control RegisterB CS02 and CS00 = 1 (1024
PRESCALER)
           TIMSK0 = (1<<OCIE0A);    // Set the Timer mask for compare interrupt flag
           sei();
      }
      if(county == 81){        // Run timer 81 times
         PORTB =(0<<DDB2);    // Toggle PortB.2 low (Turn off)
         county = 0;           // Set Counter to zero.
      }
   }
}

ISR (TIMER0_COMPA_vect){    // Timer compare flag interrupt service request
   county = county + 1;    // Increment the counter 1
   TIFR0 |= (1<<OCF0A);    // Toggle the compare flag
}
```

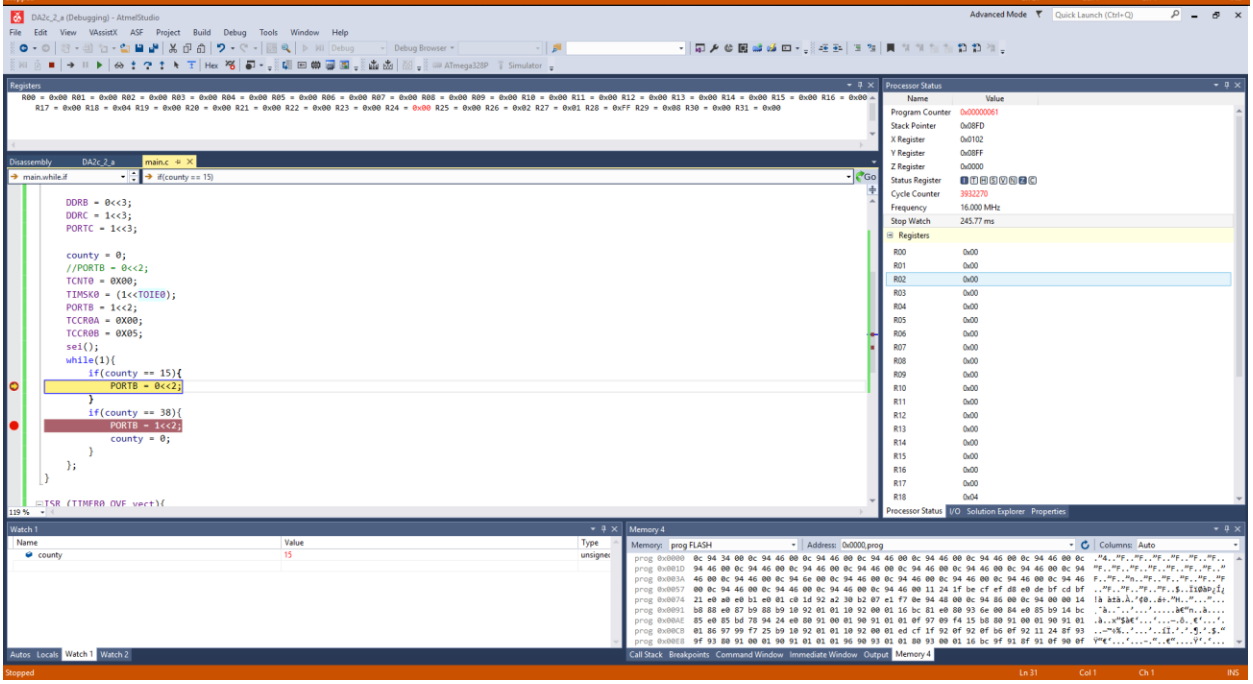
4. SCHEMATICS

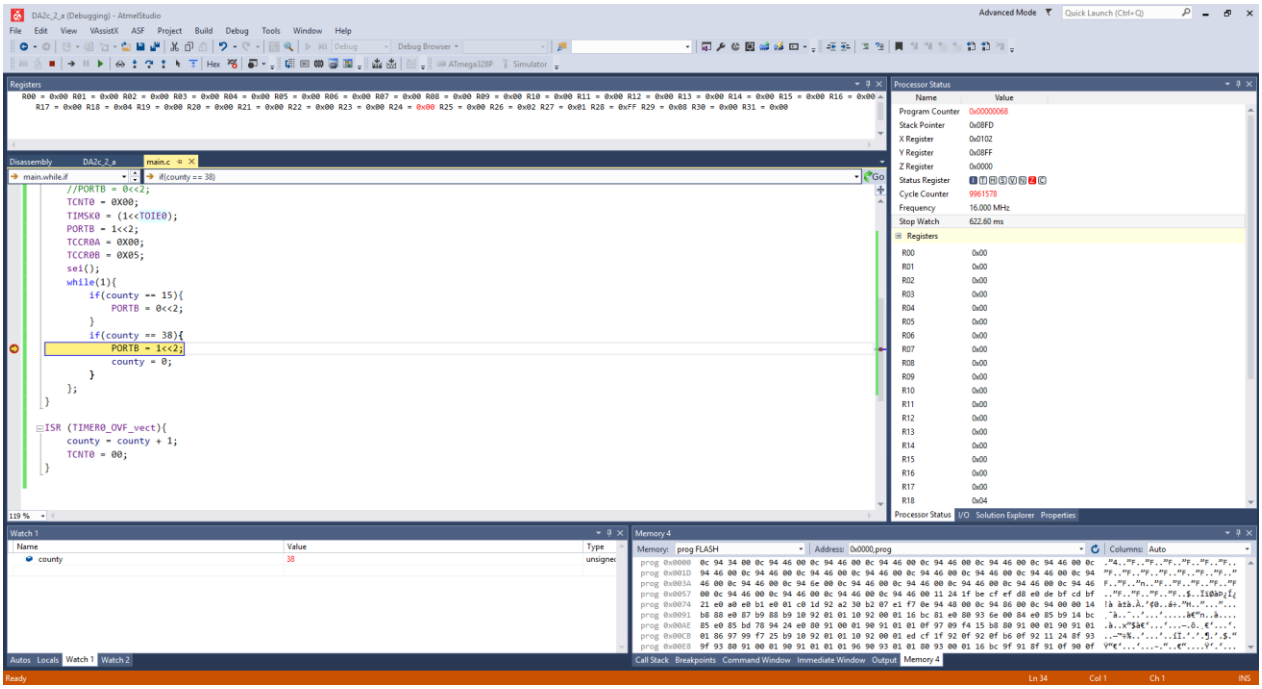


5. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)



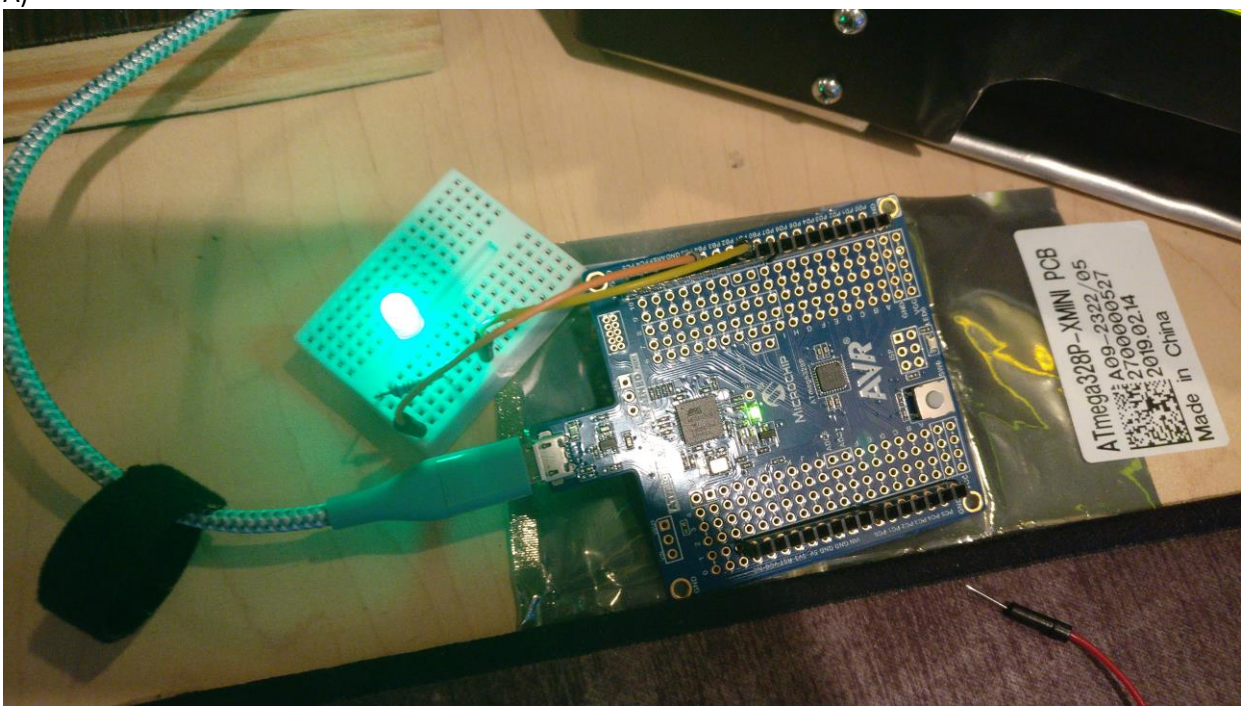

DA2c_1_a (Debugging) - AtmelStudio
Registers
R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00 R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00 R14 = 0x00 R15 = 0x00 R16 = 0x00
R17 = 0x00 R18 = 0x01 R19 = 0x04 R20 = 0x00 R21 = 0x00 R22 = 0x00 R23 = 0x00 R24 = 0x00 R25 = 0x00 R26 = 0x00 R27 = 0x00 R28 = 0xFF R29 = 0x08 R30 = 0x00 R31 = 0x00
Processor Status
Disassembly
DA2c_1_a
main.c
main.while
while (1)
TCCR0A = 0; //Timer0, Normal mode, internal clk
TCNT0 = 0X00; //Setting the count register to 0[count goes from 0-255]
TCCR0B |= 0X05; //Setting the Prescaler to 1024 and starting the timer
while (1) {
PORTB ^=(1<<DDB2); //Toggle PortB.2 (Turn on)
while(count != 15){ //Run timer 81 times
while((TIFR0 & 0X01) == 0){}; // run timer until overflow
TCNT0 = 0X00; //reset the timer count
TIFR0 = 0X01; //reset the tiemr over flow flag
count = count + 1; //increment the count of timer overflows
}
count = 0; //Reset the count of overflows
PORTB ^=(1<<DDB2); //Toggle PortB.2 (Turn off)
while(count != 23){ //Run timer 81 times
while((TIFR0 & 0X01) == 0){}; // run timer until overflow
TCNT0 = 0X00; //reset the timer count
TIFR0 = 0X01; //reset the tiemr over flow flag
count = count + 1; //increment the count of timer overflows
}
count = 0; //Reset the count of overflows
}
}
Watch 1
Name
Value
Type
count
Unknown location
Error
Memory 4
Memory: prog FLASH
Address: 0x0000.prog
Columns: Auto
Autos
Locals
Watch 1
Watch 2
Call Stack
Breakpoints
Command Window
Immediate Window
Output
Memory 4
Stopped
Ln 21
Col 1
Ch 1
INS


DA2c_2_a (Debugging) - AtmelStudio
Registers
R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00 R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00 R14 = 0x00 R15 = 0x00 R16 = 0x00
R17 = 0x00 R18 = 0x04 R19 = 0x00 R20 = 0x00 R21 = 0x00 R22 = 0x00 R23 = 0x00 R24 = 0x00 R25 = 0x00 R26 = 0x02 R27 = 0x01 R28 = 0xFF R29 = 0x08 R30 = 0x00 R31 = 0x00
Processor Status
Disassembly
DA2c_2_a
main.c
main.while.if
if(county == 15)
DDRB = 0<<3;
DDRC = 1<<3;
PORTC = 1<<3;
county = 0;
//PORTB = 0<<2;
TCNT0 = 0X00;
TIMSK0 = (1<<TOIE0);
PORTB = 1<<2;
TCCR0A = 0X00;
TCCR0B = 0X05;
sei();
while(1){
if(county == 15){
PORTB = 0<<2;
}
if(county == 38){
PORTB = 1<<2;
county = 0;
}
};
ISR (TIMER0_OVF_vect){
Watch 1
Name
Value
Type
county
15
unsigned
Memory 4
Memory: prog FLASH
Address: 0x0000.prog
Columns: Auto
Autos
Locals
Watch 1
Watch 2
Call Stack
Breakpoints
Command Window
Immediate Window
Output
Memory 4
Stopped
Ln 31
Col 1
Ch 1
INS

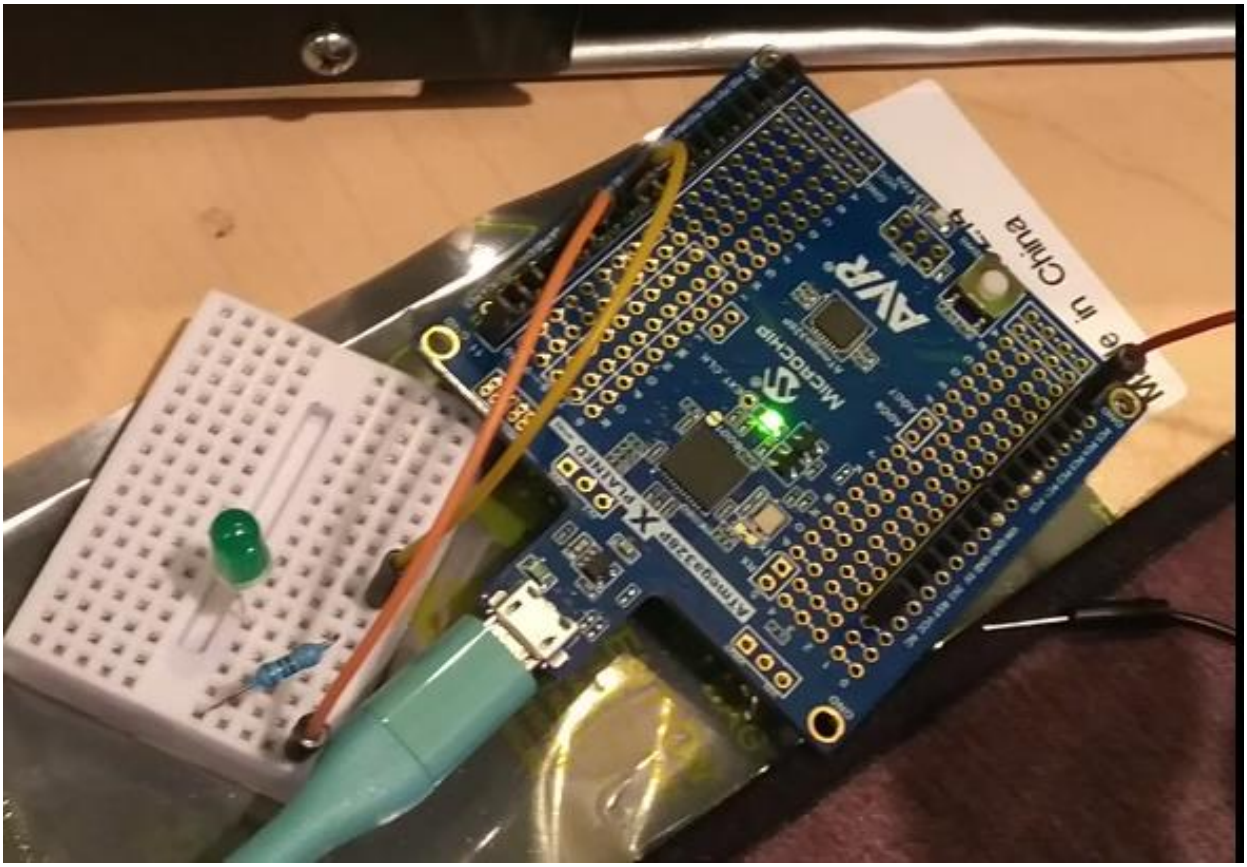


6. SCREENSHOT OF EACH DEMO (BOARD SETUP)

A)



B)



7. VIDEO LINKS OF EACH DEMO

https://youtu.be/Xa5l69hDQ1k
https://youtu.be/zoWdS4xFqiI
https://youtu.be/lmDZS8CBW1M
https://youtu.be/aWjO_K6-Hw4
https://youtu.be/07uVIZxCxxM
https://youtu.be/K1sqPyIqINE

8. GITHUB LINK OF THIS DA

Alabaster/DesignAssignments/**DA2c**/

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"*This assignment submission is my own, original work*".
Dillon Archibald