CPE301 – SPRING 2021

Design Assignment 2A

Student Name: Brian Wolak

Student #: 2000509437

Student Email: wolak@unlv.nevada.edu

Primary Github address: <https://github.com/brianwolak/submission_da.git>

Directory: [submission\_da/DA\_2 at main · brianwolak/submission\_da (github.com)](https://github.com/brianwolak/submission_da/tree/main/DA_2)

**Task 1:**

The goal of task one was to design a subroutine to generate a waveform producing an output on PORT B pin 3 with a 75% duty cycle and 1 second overall period. Given below are the necessary codes is AVR assembly and C language along with waveform and screen clip outputs.

Video Link:

<https://youtu.be/ZzgsyIp5obk>

Assembly Code:

.org 0

INITIALIZE:

ldi r16, 0x00 ;load r16 with 0 value

ldi r17, 0x08 ;load r17 with 0x08 value

ldi r18, 0xff ;load r18 with 0xff value

ldi r19, 1 ;load r19 with 1 value

ldi r20, 5 ;load r20 with 5 for prescale value

ldi r21, 0xC6 ;load r21 for 75% T1 low value

ldi r26, 0x2D ;load r26 for 75% T1 high value

ldi r27, 0x3D ;load r27 for 1 sec T1 high value

sts TCCR1A, r16 ;timer 1 setup from r16

sts TCCR1B, r20 ;prescalar setup of 1024

out DDRB, r18 ;setting DDRB at output

BEGIN:

out PORTB, r16 ;set DDRB output to 0

sts TCNT1H, r16 ;set timer1 high bits to zero

sts TCNT1L, r16 ;set timer1 low bits to zero

DELAY:

lds r22, TCNT1L ;T1L 75%

lds r23, TCNT1L ;T1L 100%

lds r24, TCNT1H ;T1H 75%

lds r25, TCNT1H ;T1H 100%

cp r22, r21 ;compare r22 and r16

breq SUB1 ;go to SUB if equal

cp r23, r17 ;compare r23 and r17

breq SUB2 ;go to SUB2 if equal

rjmp DELAY ;restart delay

SUB1:

cp r24, r26 ;compare 75% high timer values

breq TOGGLE ;to TOGGLE if same or higher

rjmp DELAY ;back to delay

SUB2:

cp r25, r27 ;compare T1 full second time values

breq BEGIN ;if less than jump to DELAY

rjmp DELAY ;back to BEGIN

TOGGLE:

out PORTB, r17 ;output 1 to PORTB

rjmp DELAY ;back to delay

C Code:

#define *F\_CPU* 16000000UL

#include <avr/io.h>

#include <util/delay.h>

int main(void)

{

DDRB = 0xff; //initialize PORTB as output

while (1)

{

PORTB &= (1<<3); //PORTB pin 3 set to output

*\_delay\_ms*(750); //delay 750ms

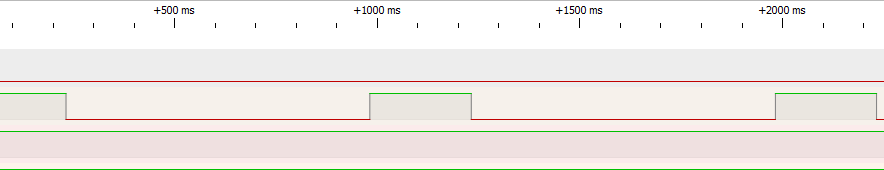
PORTB |= ~(1<<3); //PORTB pin 3 turns off

*\_delay\_ms*(250); //delay 250ms

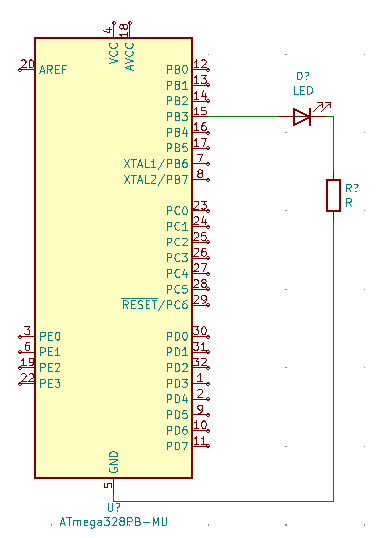
}

}

Waveform Output:

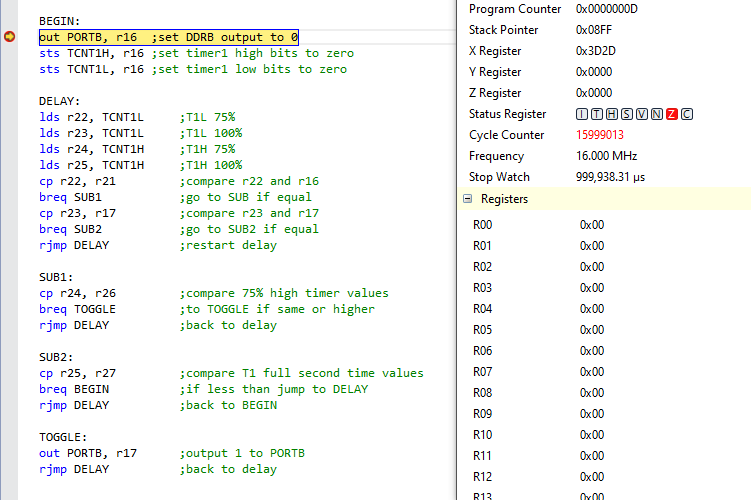


*Above we can see the 1 second period with a 75% duty cycle using a logic analyzer to capture the output*

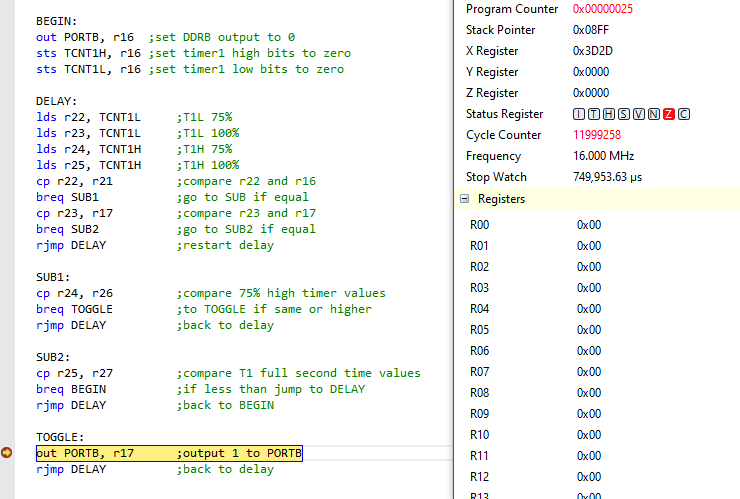


*KiCad Schematic of DA\_2A Task #1*

Atmel Output:



*Here we see the full one second period displayed as the program completes one full period*

**

*Here we see the 75% duty cycle showing 749,953us seconds in the processor status*

**Task 2:**

Task two involved an expansion of task one with the addition of a pushbutton connected to PORT C pin 3 that when pressed would illuminate an LED connected to PORT B pin two for 2 seconds. Below is the associated AVR Assembly and C codes along with waveform output.

Video Link:

<https://youtu.be/rK2zF2bcbbg>

Assembly Code:

.org 0

INITIALIZE:

ldi r16, 0x0C ;load r16 with 0 value

ldi r17, 0x08 ;load r17 with 0x08 value

ldi r18, 0xff ;load r18 with 0xff value

ldi r19, 1 ;load r19 with 1 value

ldi r20, 5 ;load r20 with 5 for prescale value

ldi r21, 0xC6 ;load r21 for 75% T1 low value

ldi r26, 0x2D ;load r26 for 75% T1 high value

ldi r27, 0x3D ;load r27 for 1 sec T1 high value

sts TCCR1A, r16 ;timer 1 setup from r16

sts TCCR1B, r20 ;prescalar setup of 1024

out DDRB, r16 ;setting DDRB at output

ldi r28, 0x08 ;pull up reisitor setup value

out PORTC, r28 ;pull up reisitor set

ldi r29, 0x00 ;counter set

BEGIN:

cbi PORTB, 3 ;set DDRB output to 0

sts TCNT1H, r16 ;set timer1 high bits to zero

sts TCNT1L, r16 ;set timer1 low bits to zero

cpi r29, 0x02 ;check if two cycles completed

breq LEDOFF ;start LEDOFF cycles if 2 seconds completed

in r30, PINC ;read PINC

andi r30, 0x08 ;mask input

cpi r30, 0x08 ;button was pressed if equal

breq LEDON ;start LEDON 2 second cycle

rjmp DELAY

DELAY:

lds r22, TCNT1L ;T1L 75%

lds r23, TCNT1L ;T1L 100%

lds r24, TCNT1H ;T1H 75%

lds r25, TCNT1H ;T1H 100%

cp r22, r21 ;compare r22 and r16

breq SUB1 ;go to SUB if equal

cp r23, r17 ;compare r23 and r17

breq SUB2 ;go to SUB2 if equal

rjmp DELAY ;restart delay

SUB1:

cp r24, r26 ;compare 75% high timer values

breq TOGGLE ;to TOGGLE if same or higher

rjmp DELAY ;back to delay

SUB2:

cp r25, r27 ;compare T1 full second time values

brlt DELAY ;if less than jump to DELAY

inc r29 ;increment 2 second counter

rjmp BEGIN ;back to BEGIN

TOGGLE:

sbi PORTB, 3 ;output 1 to PORTB

rjmp DELAY ;back to delay

LEDON:

sbi PORTB, 2 ;start 2 sec LED on sequence

rjmp DELAY

LEDOFF:

cbi PORTB, 2 ;stop LED 2 sec cycle

ldi r29, 0x00 ;reset counter

rjmp BEGIN

C Code:

#define *F\_CPU* 16000000UL //define CPU

#include <avr/io.h>

#include <util/delay.h> //delay library

int main(void){

DDRB = 0xff; //setting PORTB as output

PORTC |= (1<<2); //setting PORTB pullup on PIN2

int count = 0; //counter variable

while (1)

{

count = 0; //reset count

PORTB |= (1<<3); //turn on PB.3 LED

*\_delay\_ms*(250); //delay 250ms

PORTB ^= (1<<3); //turn PB.3 LED off

*\_delay\_ms*(750); //delay 750ms

if((PINC & 0x08) ==0x08){

while (count < 1){

PORTB |= (1<<2); //turn on PB.2 LED

PORTB |= (1<<3); //turn on PB.3 LED

*\_delay\_ms*(250);

PORTB ^= (1<<3);

*\_delay\_ms*(750); //delay 750ms

PORTB |= (1<<3); //turn off PB.3 LED

*\_delay\_ms*(250); //delay 250ms

PORTB ^= (1<<3); //turn on PB.2 LED

*\_delay\_ms*(750); //delay 750ms

PORTB = 0x00; //turn off all PB LEDS

count++; //increment count

}

}

else{

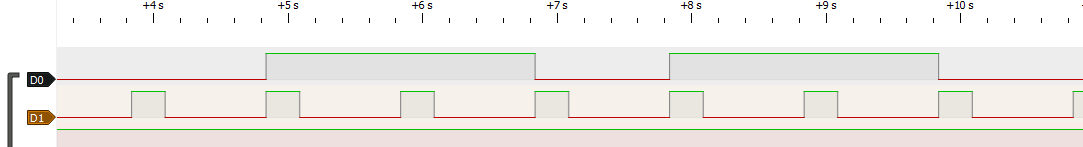
count = 0; //reset count

}

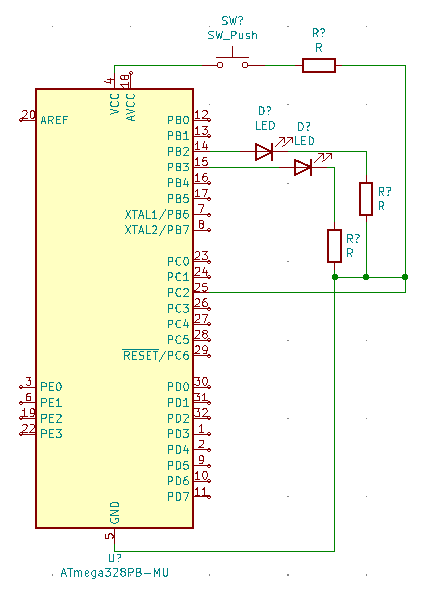
}

}

Waveform Output:

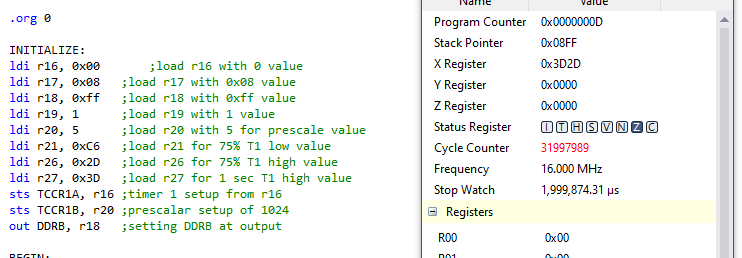


*Above we can see again the original 1 second waveform with 1 second period and 75% duty cycle this time with the addition of a two second waveform beginning upon button press and leaving the original 1 second waveform unaffected*

****

*KiCad Schematic of DA\_2A Task #2*

Atmel Output:



*Finally we see the full 2 second delay after button press in the processor status*

1. **GITHUB LINK OF THIS DA**

[submission\_da/DA\_2 at main · brianwolak/submission\_da (github.com)](https://github.com/brianwolak/submission_da/tree/main/DA_2)