Jason Sikorski

CPE301 – SPRING 2016

Design Assignment 2

DO NOT REMOVE THIS PAGE DURING SUBMISSION:

The student understands that all required components should be submitted in complete for grading of this assignment.

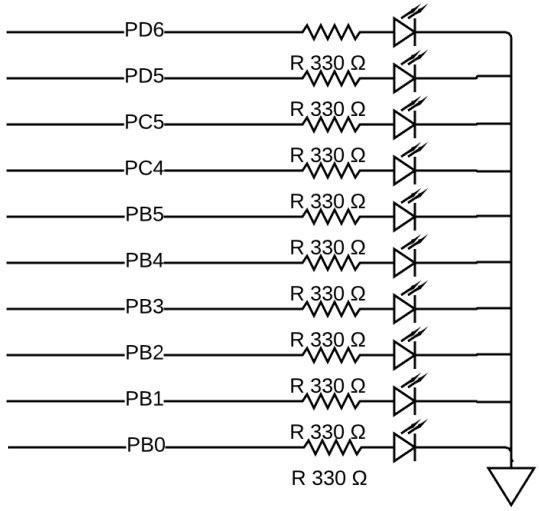
NO	SUBMISSION ITEM	COMPLETED (Y/N)	MARKS (/MAX)
0.	COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS		
1.	INITIAL CODE OF TASK 1/A		
2.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 2/B		
3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 3/C		
4.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 4/D		
5.	SCHEMATICS		
6.	SCREENSHOTS OF EACH TASK OUTPUT		
7.	SCREENSHOT OF EACH DEMO		
8.	VIDEO LINKS OF EACH DEMO		
9.	GITHUB LINK OF THE DA		

0.	COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS		

Components:

330 Ω resistor x 10
LED bar X 1
Atmega328P x 1

Connection Block Diagram:



1.	INITIAL CODE OF TASK 1/A		

main.asm

```
.INCLUDE "header.inc"                   ;Include header file
.ORG 0x00                               ;Begining of code goes here

        INITSTACK                       ;Initialize the stack
        SBI         DDRC, 0                 ;Set PORTC0 to output
        LDI         R17, 0x01           ;Set R17 bit 1 high
        LDI         R16, 0x01           ;Set R16 bit 1 high
        OUT         PORTC, R17          ;Output R17 to PORTC
BEGIN:
        RCALL       DELAY               ;Call delay subroutine
        EOR         R17, R16            ;Toggle R17 bit 1
        OUT         PORTC, R17          ;Output R17 to PORTC
        RJMP        BEGIN
```

header.inc

```
.ORG 0x100
```

```
.MACRO INITSTACK                        ;Macro to initialize Stack Pointer
        LDI     R16, HIGH(RAMEND)
        OUT     SPH, R16                ;Set Stack Pointer to RAMEND
        LDI     R16, LOW(RAMEND)
        OUT     SPL, R16
.ENDMACRO

DELAY:                                          ;Delay subroutine for 0.5 second delay
        PUSH    R20                     ;Save R20 on the stack
        LDI     R20, 0x0B
        STS     TCNT1H, R20
        LDI     R20, 0xDB
        STS     TCNT1L, R20             ;Set TCNT1 for 62,501 cycles
        LDI     R20, 0x00
        STS     TCCR1A, R20
        LDI     R20, 0x03
        STS     TCCR1B, R20             ;Set prescalar to 64
AGAIN:
        IN      R20, TIFR1              ;Get TIFR1 register
        SBRS    R20, TOV1               ;Check TOV1 flag
        RJMP    AGAIN
        LDI     R20, 0x00
        STS     TCCR1B, R20             ;Stop Timer
        LDI     R20, 0x01
        OUT     TIFR1, R20              ;Clear TOV1 flag
        POP     R20                     ;Restore R20
        RET
```

main.c

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

void delay();     // Delay function for 0.5 second delay

int main(void)
{
        DDRB |= (1<<0);         // Set PORTB0 to output
        while(1)
        {
                delay(); // Call delay infinitely
        }
        return 0;
}

void delay()
{
        TCNT1 = 3035;// Set TCNT1 to 62,501 cycles

        TCCR1A = 0;
        TCCR1B |= (1<<CS10)|(1<<CS11);      // Set prescalar to 64
        while((TIFR1 &= 1<<TOV1) != 1);      // Wait for TOV1 flag
        TCCR1B = 0;   // Stop Timer
        TIFR1 |= (1<<TOV1);   // Clear TOV1 flag
```

```
        PORTB ^= (1<<0);        // Toggle PORTB0
        return;
}
```

2.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 2/B		

main.asm

```
.INCLUDE "header.inc"                   ;Inculde header
.ORG 0x00                               ;Code begins here

        INITSTACK                       ;Initialize the stack

        LDI         R18, 0xFF           ;Set R18 bits 0-7 high
        OUT         DDRB, R18           ;Set PORTB to output
        OUT         DDRC, R18           ;Set PORTC to output
BEGIN:
        RCALL       DELAY               ;Call delay subroutine
        SUBI        R17, -1             ;Increment R17
        OUT         PORTB, R17          ;Output R17 to PORTB
        MOV         R16, R17            ;Copy R17 to R16
        LSR         R16                 ;Shift R16 right
        LSR         R16                 ;Shift R16 right
        OUT         PORTC, R16          ;Output R16 to PORTC
        RJMP        BEGIN
```

main.c

```
char delay(char);           // Delay function for 0.5 second delay, returns updated counter

int main(void)
{
        DDRB = 0xFF; // Set PORTB to output
        DDRC = 0xFF; // Set PORTC to output
        char count = 0;  // 8 bit counter variable
        while(1)
        {
                count = delay(count);     // Call delay function on counter variable
                PORTB = count;            // Output counter bits 0-5 on PORTB0-5
                PORTC = count >> 2;       // Output counter bits 6-7 on PORTC4-5
        }
        return 0;
}

char delay(char a)
{
        TCNT1 = 3035;// Set TCNT1 to 62,501 cycles
        TCCR1A = 0;
        TCCR1B |= (1<<CS10)|(1<<CS11);      // Set prescalar to 64
        while((TIFR1 &= 1<<TOV1) != 1);     // Wait for TOV1 flag
        TCCR1B = 0;   // Stop timer
        TIFR1 |= (1<<TOV1);   // Clear TOV1 flag
        a++;     // increment counter variable
```

```
    return a;           // return counter variable
}
```

3.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 3/C		

main.asm

```
.INCLUDE "header.inc"                   ;Include header file
.ORG 0x00                               ;Code begins here

    INITSTACK                           ;Initialize the stack

    LDI     R18, 0xFF       ;Set R18 bit 0-7 high
    OUT     DDRB, R18       ;Set PORTB to output
    OUT     DDRC, R18       ;Set PORTC to output
    OUT     DDRD, R18       ;Set PORTD to output
    LDI     R17, 0x00       ;Initialize R17 to 0
    LDI     R16, 0x00       ;Initialize R16 to 0
    LDI     R19, 0x00       ;Initialize R19 to 0
    LDI     R21, 0x00       ;Initialize R21 to 0
    LDI     R18, 0x60       ;Set R18 bit 5&6 high
    LDI     R20, 0x20       ;Set R20 bit 5 high
BEGIN:
    RCALL   DELAY           ;Call delay subroutine
    SUBI    R17, -1         ;Increment R17 (counter)
    SUBI    R19, -1         ;Increment R19 (counter for 5's)
    OUT     PORTB, R17      ;Output R17 to PORTB
    MOV     R16, R17        ;Copy R17 to R16
    LSR     R16             ;Shift R16 Right
    LSR     R16             ;Shift R16 Right
    ANDI    R16, 0x30       ;Clear R16 bits 1-3 & 6-7
    CPI     R19, 5          ;Check if 5's counter = 5
    BRNE    NOT5
    EOR     R21, R20        ;Toggle R21 bit 5
NOT5:
    CPI     R19, 10         ;Check if 5's counter = 10
    BRNE    NOT10
    EOR     R21, R18        ;Toggle R21 bit 5&6
    LDI     R19, 0
NOT10:
    OUT     PORTC, R16      ;Output R16 to PORTC
    OUT     PORTD, R21      ;Output R21 to PORTD
    RJMP    BEGIN
```

main.c

```
int main(void)
{
    DDRB = 0xFF; // Set PORTB to output
    DDRC = 0xFF; // Set PORTC to output
    DDRD |= (1<<5) | (1<<6);        // Set PORTD5&6 to output
    char count = 0;  // Variable for 8 bit counter
    while(1)
```

```
	{
		count = delay(count); // Update counter by calling delay
		PORTB = count;		// Output count bits 0-5 to PINB0-5
		PORTC = count >> 2;	// Output count bits 6-7 to PINC4-5
		if(count%5 == 0)		// Check if count is div by 5
			PORTD ^= (1<<5);	// Toggle PIND5
		if(count%10 == 0)		// Check if count is div by 10
			PORTD ^= (1<<6);	// Toggle PIND6
	}
	return 0;
}
```

4.	INCREMENTAL / DIFFERENTIAL CODE OF TASK 4/D		

header.inc

```
.ORG 0x00
	RJMP	MAIN		;Set Reset Interrupt
.ORG 0x1A
	RJMP	TIM1_OVF	;Set Timer1 Overflow Interrupt

TIM1_OVF:				;Timer1 Overflow Interrupt Subroutine
	PUSH	R20		;Save R20
	LDS	R20, SREG	;Save SREG
	PUSH	R20
	LDI	R20, 0x00
	STS	TCCR1B, R20	;Stop Timer1
	LDI	R20, 1<<TOV1
	OUT	TIFR1, R20	;Clear TOV1 flag
	LDI	TEMP, 0			;Set TEMP reg to 0
	RCALL	DELAY		;Call Delay subroutine
	POP	R20
	STS	SREG, R20	;Restore SREG
	POP	R20		;Restore R20
	RETI


.ORG 0x500

.DEF	COUNT1 = R17		;Count1 Variable in R17
.DEF	COUNT5 = R19		;Count5 Variable in R19
.DEF	TEMP = R23		;Temp Register in R23 (This Reg is not saved)

.MACRO INITSTACK			; Initialize Stack Point to RAMEND
	LDI	R16, HIGH(RAMEND)
	OUT	SPH, R16
	LDI	R17, LOW(RAMEND)
	OUT	SPL, R16
.ENDMACRO

DELAY:					;Delay Subroutine for 0.5 sec delay
	PUSH	R20		;Save R20
	LDI	R20, 0x0B
```

```
        STS     TCNT1H, R20
        LDI     R20, 0xDB
        STS     TCNT1L, R20     ;Set TCNT1 to 62,501 cycles
        LDI     R20, 0x00
        STS     TCCR1A, R20
        LDI     R20, 0x03
        STS     TCCR1B, R20     ;Set prescalar to 64
        LDI     R20, 0x01
        STS     TIMSK1, R20     ;Enable Timer1 Overflow Interrupt
        POP     R20
        RET
```

main.asm

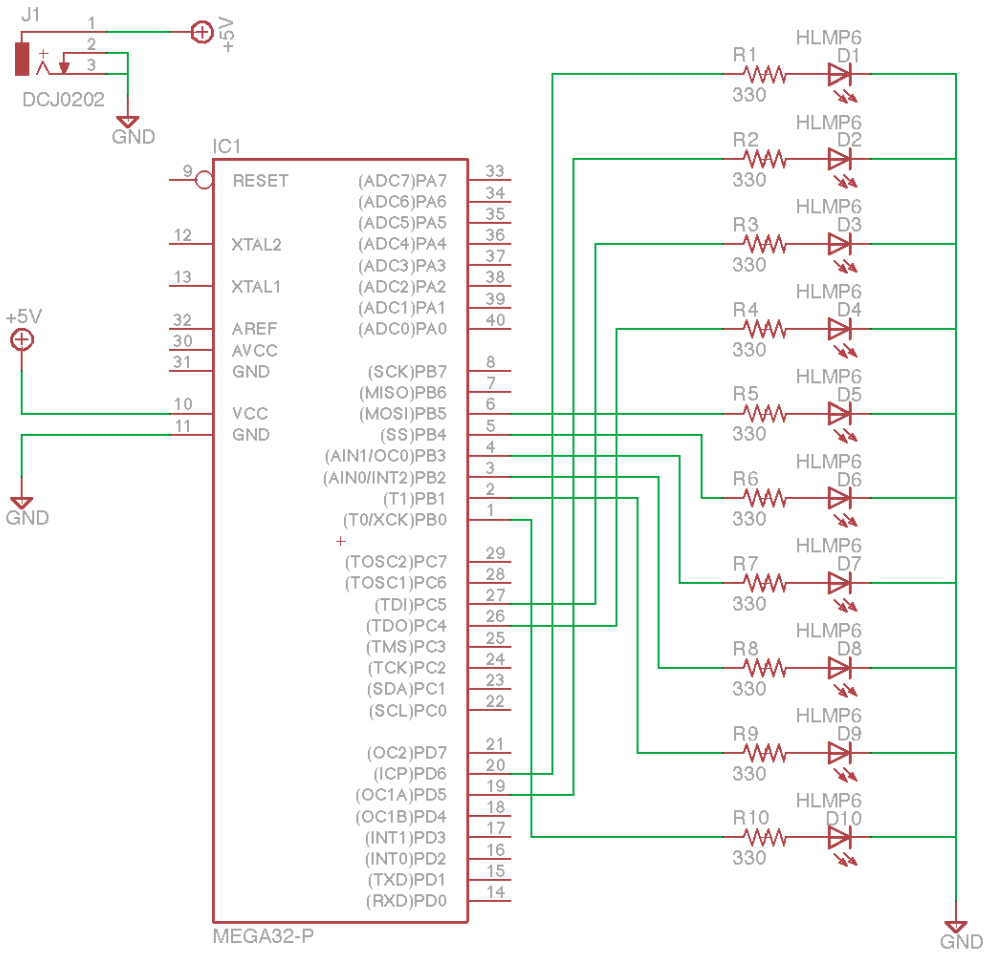
```
.INCLUDE "header.inc"                   ;Include header file

.ORG 0x100                              ;Begining of code goes here

MAIN:                                   ;Label for reset interrupt
        INITSTACK                       ;Initalize the stack

        LDI     R18, 0xFF               ;Set R18 bits 0-7 high
        OUT     DDRB, R18               ;Set PORTB to output
        OUT     DDRC, R18               ;Set PORTC to output
        OUT     DDRD, R18               ;Set PORTD to output
        LDI     COUNT1, 0x00            ;Initialize Count1 to 0
        LDI     COUNT5, 0x00            ;Initialize Count5 to 0
        LDI     R16, 0x00               ;Initialize R16 to 0
        LDI     R18, 0x60               ;Set R18 bits 5&6 high
        LDI     R20, 0x00               ;Initialize R20 to 0
        LDI     R21, 0x00               ;Initialize R21 to 0
        LDI     R22, 0x20               ;Set R22 bit 5 high
        SEI                             ;Enable Global Interrupts
        RCALL   DELAY                   ;Call Delay subroutine

COUNT:
        LDI     TEMP, 0x01              ;Set TEMP bit 1 high
        SUBI    COUNT1, -1              ;Increment Count1
        SUBI    COUNT5, -1              ;Increment Count5
        OUT     PORTB, COUNT1           ;Output Count1 to PORTB
        MOV     R16, COUNT1             ;Copy Count1 to R16
        LSR     R16                     ;Right Shift R16
        LSR     R16                     ;Right Shift R16
        ANDI    R16, 0x30               ;Clear R16 bits 0-3 and 6-7
        CPI     COUNT5, 5               ;Check if Count5 = 5
        BRNE    NOT5
        EOR     R21, R22                ;Toggle R21 bit 5
NOT5:
        CPI     COUNT5, 10              ;Check if Count5 = 10
        BRNE    NOT10
        EOR     R21, R18                ;Toggle R21 bit 5&6
        LDI     COUNT5, 0               ;Reset Count5
NOT10:
        OUT     PORTC, R16              ;Output R16 to PORTC
```

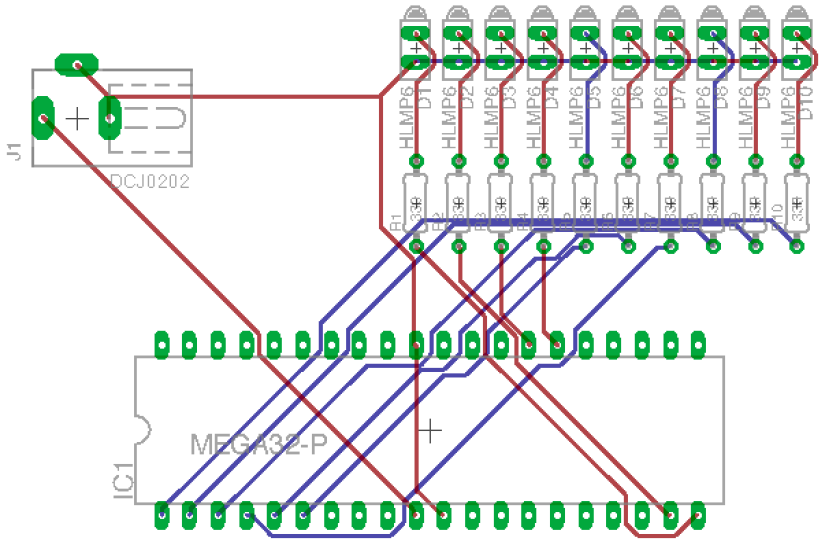
```
        OUT         PORTD, R21          ;Output R21 to PORTD

WAIT:
        CPI         TEMP, 0             ;Loop here while TEMP = 1
        BREQ        COUNT
        RJMP        WAIT
```

main.c

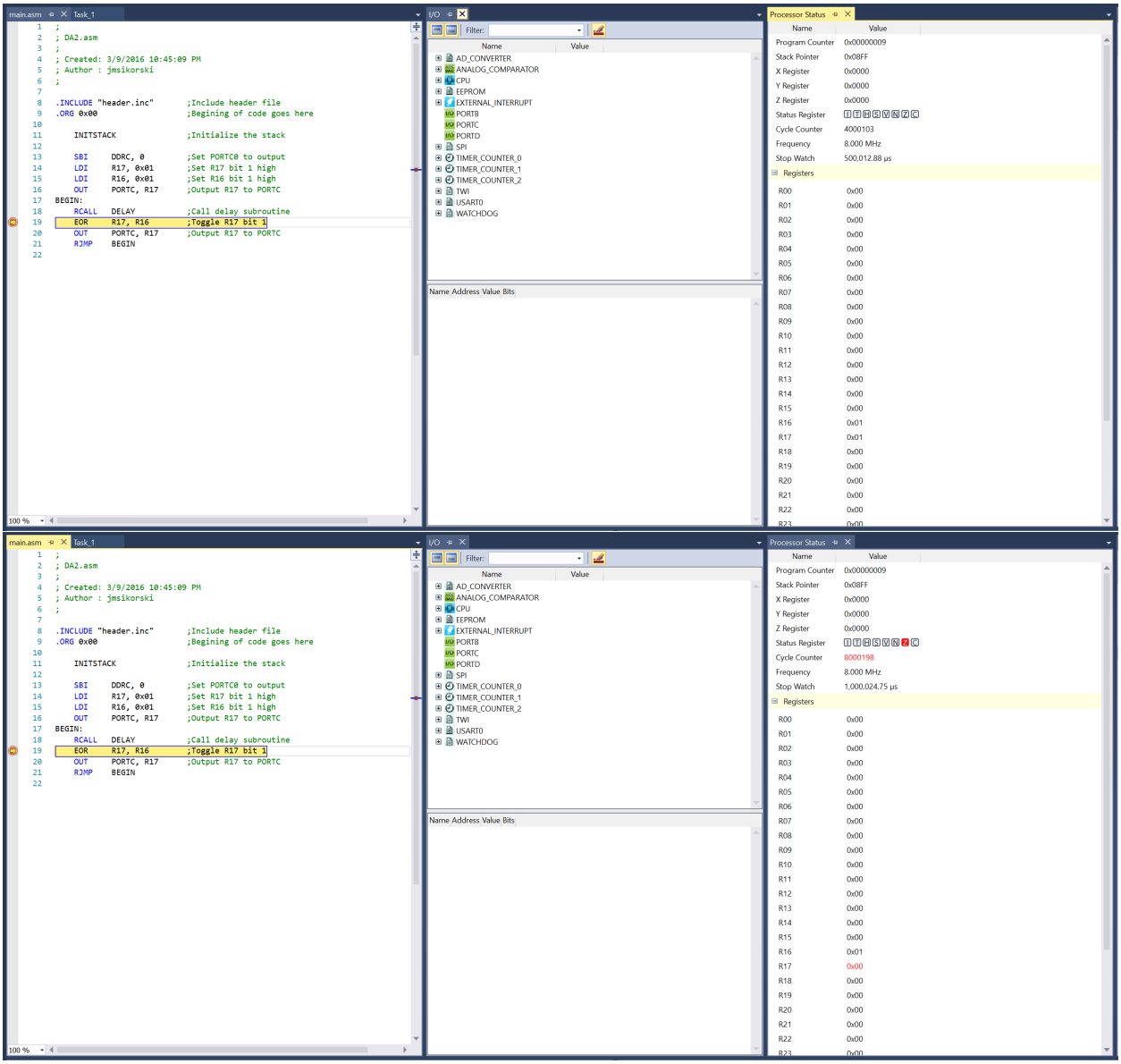
```
ISR(TIMER1_OVF_vect)        // Timer 1 overflow interrupt
{
        PORTC |= (1<<0);        // Set PORTC0 to 1, flag for Timer1 completion
}

int main(void)
{
        DDRB = 0xFF; // Set PORTB to output
        DDRC = 0xFF; // Set PORTC to output
        DDRD |= (1<<5) | (1<<6);        // Set PORTD5&6 to output
        PORTC |= (1<<0);        // Initialize Timer1 flag to 1
        TIMSK1 |= (1<<TOIE1);           // Enable Timer1 interrupts
        char count = 0;  // Variable for 8 bit counter
        sei();    // Enable Global Interrupts
        while(1)
        {
                if((PORTC & (1<<0)) == 1)       // Check Timer1 completion flag
                {
                        count++;            // Increment counter
                        PORTB = count;          // Output counter to port B(bits 0-5)
                        PORTC = count >> 2; // Output counter to port C(bits 6 & 7)
                        if(count%5 == 0)            // Check if count is div by 5
                                PORTD ^= (1<<5);        // Toggle PORTD5
                        if(count%10 == 0)       // Check if count is div by 10
                                PORTD ^= (1<<6);        // Toggle PORTD6
                        delay(); // Start Delay
                        PORTC &= 0xFE;          // Clear Timer1 completion flag
                }
        }
        return 0;
}

void delay()
{
        TCNT1 = 3035;// Set TCNT1 for 62,501 cycles
        TCCR1A = 0;
        TCCR1B |= (1<<CS10)|(1<<CS11);      // Set prescalar to 64
        return;
}
```

5.	SCHEMATICS		





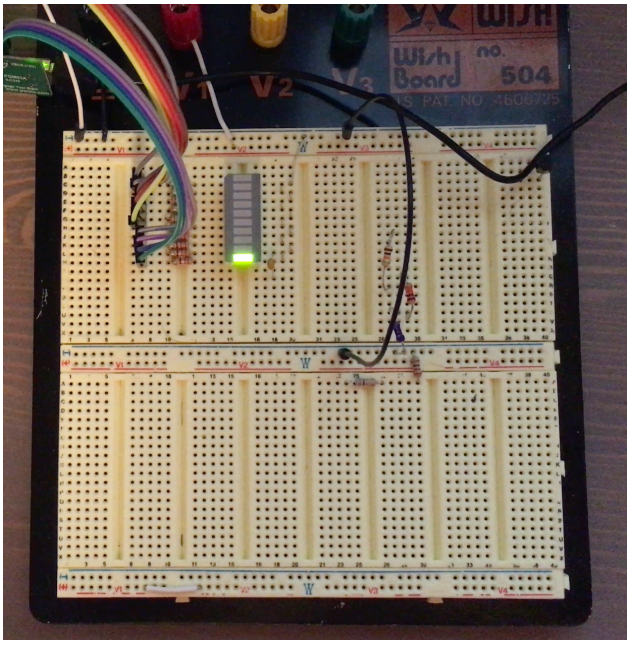
6.	SCREENSHOTS OF EACH TASK OUTPUT		



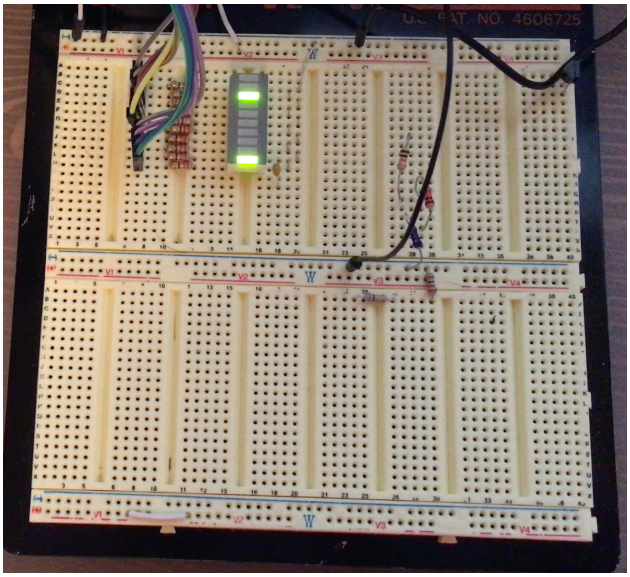
Calculated period: 500.01187 µs

7.	SCREENSHOTS OF EACH TASK DEMO		

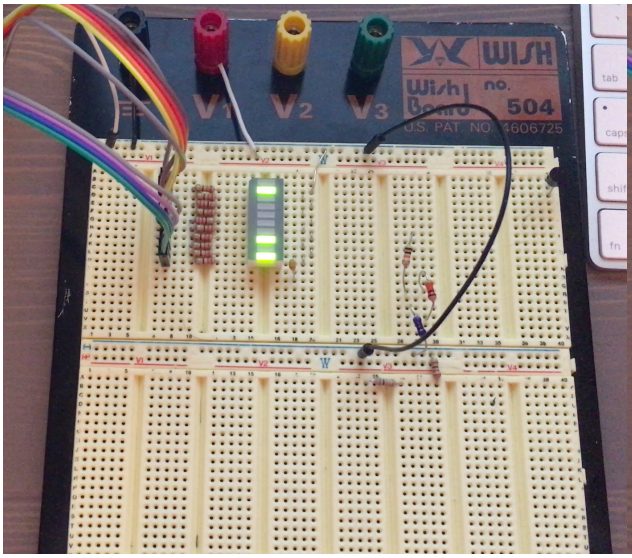
TASK 1:



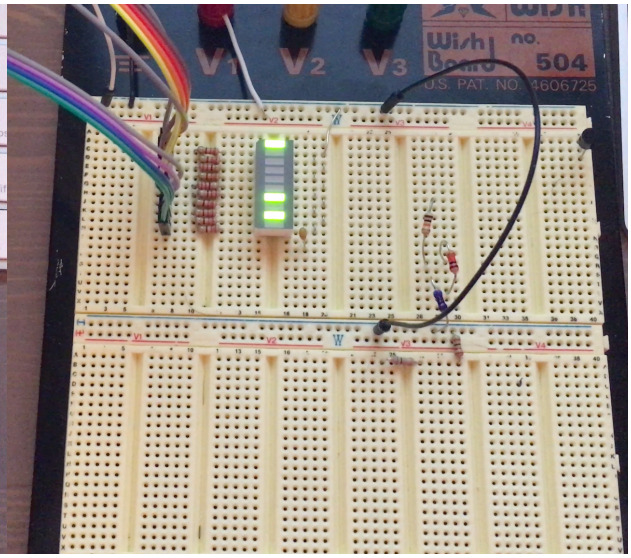
TASK2:



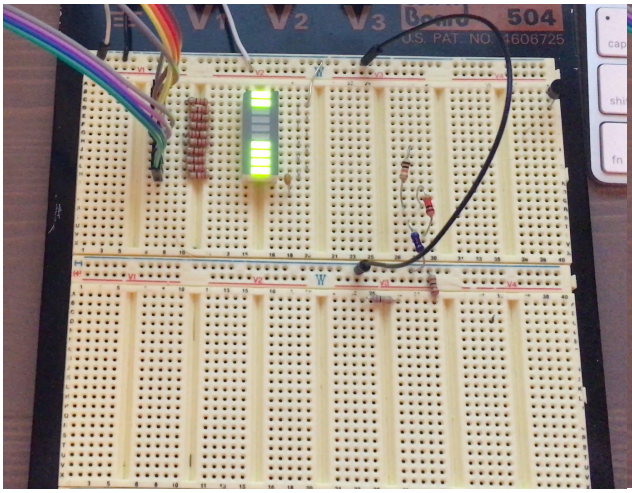
TASK3 & 4:



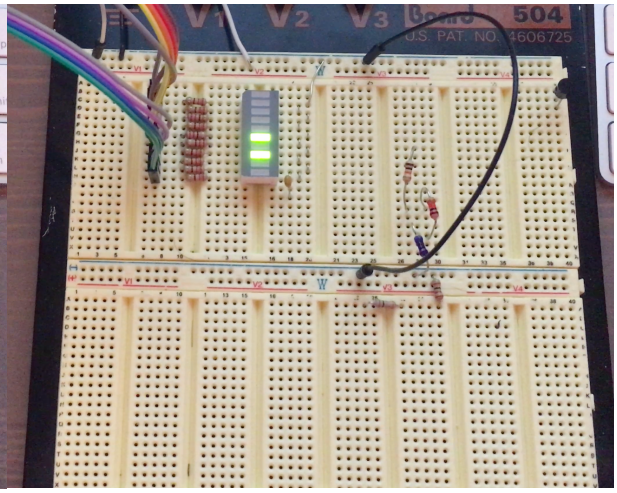
Output = 5



Output = 10



Output = 15



Output = 20

8.	VIDEO LINKS OF EACH DEMO		
Task 1: https://youtu.be/v6YsDAOqKJ0 Task 2: https://youtu.be/hhcdGv-t3o8 Task 3: https://youtu.be/wC3huNwjNAY			
9.	GITHUB LINK OF THE DA		
https://github.com/jmsikorski/UNLVCpE301Sp16/tree/master/DA2			

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

"This assignment submission is my own, original work".
JASON M. SIKORSKI