Noor Blum

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. | C CODE WITH COMMENTS FOR TASK 1/A | Y |  |
| 1. | C CODE WITH COMMENTS FOR TASK 1/B | Y |  |
| 2. | C CODE WITH COMMENTS FOR TASK 1/C | Y |  |
| 3. | C CODE WITH COMMENTS FOR TASK 1/D | Y |  |
| 4. | ASM CODE WITH COMMENTS FOR TASK 1/A | Y |  |
| 5. | ASM CODE WITH COMMENTS FOR TASK 1/B | Y |  |
| 6. | ASM CODE WITH COMMENTS FOR TASK 1/C | Y |  |
| 7. | ASM CODE WITH COMMENTS FOR TASK 1/D | Y |  |
| 8. | SCHEMATIC | Y |  |
| 9. | SNAPSHOT OF THE BOARD WITH CONNECTED COMPONENTS | Y |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 0. | C CODE WITH COMMENTS FOR TASK 1/A |  |  |

#include <avr/io.h> //Header file for AVR device specific I/O Definition

#define F\_CPU 8000000 // Clock Speed as 8MHz

#include <util/delay.h> //Header file for incorporating delay routines.

#include <avr/interrupt.h>

void init\_port()

{

DDRC |= 0x23; // SET PORTC.0 PORTC.5 as output

DDRD &= ~(0x10); // SET PORTD.4 (PIN TO) as input

DDRB = 0xff; // SET PORTB as output

}

int main(void)

{

init\_port();

/\* PROGRAM CODE FOR GENERATING 50% DUTY CYCLE\*/

while(1)

{

\_delay\_ms(250);

PORTC &= ~(0x01);

\_delay\_ms(250);

PORTC |= 0x01;

}

}

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | C CODE WITH COMMENTS FOR TASK 1/B |  |  |

#include <avr/io.h> //Header file for AVR device specific I/O Definition

#define F\_CPU 8000000 // Clock Speed as 8MHz

#include <util/delay.h> //Header file for incorporating delay routines.

#include <avr/interrupt.h>

void init\_port()

{

DDRC |= 0x23; // SET PORTC.0 PORTC.5 as output

DDRD &= ~(0x10); // SET PORTD.4 (PIN TO) as input

DDRB = 0xff; // SET PORTB as output

}

void init\_counter0()

{

OCR0A = 0x04; // for Input capture value at 5th pulse

OCR0B = 0x09; // for Input capture value at 10th pulse

TIFR0 |= (1<<OCF0A)|(1<<OCF0B); // clear the capture flag

TCCR0B |= 0x07; // set clock as external

}

int main(void)

{

init\_port();

init\_counter0();

while(1)

{

/\* PROGRAM TO GENERATE 50% DUTY CYCLE SIGNAL \*/

\_delay\_ms(250);

PORTC &= ~(0x01);

\_delay\_ms(250);

PORTC |= 0x01;

/\* DISPLAY THE COUNT ON PORTB \*/

PORTB = TCNT0;

}

}

}

|  |  |  |  |
| --- | --- | --- | --- |
| 2. | C CODE WITH COMMENTS FOR TASK 1/C |  |  |

#include <avr/io.h> //Header file for AVR device specific I/O Definition

#define F\_CPU 8000000 // Clock Speed as 8MHz

#include <util/delay.h> //Header file for incorporating delay routines.

#include <avr/interrupt.h>

void init\_port()

{

DDRC |= 0x31; // SET PORTC.0, PORTC.4, PORTC.5 as output

DDRD &= ~(0x10); // SET PORTD.4 (PIN TO) as input

DDRB = 0xff; // SET PORTB as output

}

void init\_counter0()

{

OCR0A = 0x04; // for Input capture value at 5th pulse

OCR0B = 0x09; // for Input capture value at 10th pulse

TIFR0 |= (1<<OCF0A)|(1<<OCF0B); // clear the capture flag

TCCR0B |= 0x07; // set clock as external

}

int main(void)

{

init\_port();

init\_counter0();

while(1)

{

/\* Generating Pulse with 50% duty cycle \*/

\_delay\_ms(250);

PORTC &= ~(0x01);

\_delay\_ms(250);

PORTC |= 0x01;

/\* Display the counter value on PORTB \*/

PORTB = TCNT0;

/\* Toggling LEDs on every 5th Count and 10th COunt at PORTC 4th and 5th PIN \*/

if ((TIFR0&(1<<OCF0A))== (1<<OCF0A)) // Toggle the LED after 5th Pulse on PORTC pin

{

PORTC^=(1<<PC4);

OCR0A+= 0x05;

TIFR0 |= (1<<OCF0A);

}

if ((TIFR0&(1<<OCF0B))== (1<<OCF0B)) // Toggle the LED after every 10th Pulse on PORTC pin

{

PORTC^=(1<<PC5);

PORTC^=(1<<PC4);

OCR0B+= 0x0A;

TIFR0 |= (1<<OCF0B);

}

}

}

|  |  |  |  |
| --- | --- | --- | --- |
| 3. | C CODE WITH COMMENTS FOR TASK 1/D |  |  |

#include <avr/io.h> //Header file for AVR device specific I/O Definition

#define F\_CPU 8000000 // Clock Speed as 8MHz

#include <util/delay.h> //Header file for incorporating delay routines.

#include <avr/interrupt.h>

void init\_port()

{

DDRC |= 0x23; // SET PORTC.0 PORTC.5 as output

DDRD &= ~(0x10); // SET PORTD.4 (PIN TO) as input

DDRB = 0xff; // SET PORTB as output

}

void init\_counter0()

{

OCR0A = 0x04; // for Input capture value at 5th pulse

OCR0B = 0x09; // for Int put capture value at 10th pulse

TIFR0 |= (1<<OCF0A)|(1<<OCF0B); // clear the capture flag

//INTRRUPT

TIMSK0 |= (1<<OCIE0B) | (1<<OCIE0A); // enable capture interrupts

TCCR0B |= 0x07; // set clock as external

}

ISR(TIMER0\_COMPA\_vect)

{

/\* TOGGLING THE LED AFTER EVERY 5TH PULSE \*/

PORTC^=(1<<PC4); // XOR operation is used to toggle

OCR0A+= 0x05;

}

ISR(TIMER0\_COMPB\_vect)

{

/\* TOGGLING THE LED AFTER EVERY 10TH PULSE \*/

PORTC^=(1<<PC5);

PORTC^=(1<<PC1);

OCR0B+= 0x0A;

}

int main(void)

{

init\_port();

init\_counter0();

sei();

while(1)

{

/\* GENERATING 50% DUTY CYCLE \*/

\_delay\_ms(250);

PORTC &= ~(0x01);

\_delay\_ms(250);

PORTC |= 0x01;

/\* DISPLAY COUNTER VALUE ON PORTB \*/

PORTB = TCNT0;

}

}

|  |  |  |  |
| --- | --- | --- | --- |
| 4. | ASM CODE WITH COMMENTS FOR TASK 1/A |  |  |

\_\_SP\_H\_\_ = 0x3e

\_\_SP\_L\_\_ = 0x3d

\_\_SREG\_\_ = 0x3f

\_\_tmp\_reg\_\_ = 0

\_\_zero\_reg\_\_ = 1

.text

.global init\_port

.type init\_port, @function

/\* initialisation of ports \*/

init\_port:

/\* prologue: function \*/

/\* frame size = 0 \*/

/\* stack size = 0 \*/

.L\_\_stack\_usage = 0

in r24,0x7 /\*SET PORTC.0 PORTC.5 as output\*/

ori r24,lo8(35)

out 0x7,r24 /\* SET PORTD.4 (PIN TO) as input\*/

cbi 0xa,4

ldi r24,lo8(-1)

out 0x4,r24 /\*SET PORTB as output\*/

ret

.size init\_port, .-init\_port

.section .text.startup,"ax",@progbits

.global main

.type main, @function

main:

/\* prologue: function \*/

/\* frame size = 0 \*/

/\* stack size = 0 \*/

.L\_\_stack\_usage = 0

call init\_port

.L3:

/\* start of a continuos loop while(1)\*/

/\* delay 250 ms\*/

ldi r18,lo8(399999)

ldi r24,hi8(399999)

ldi r25,hlo8(399999)

1: subi r18,1

sbci r24,0

sbci r25,0

brne 1b

rjmp .

nop

/\* send low \*/

cbi 0x8,0

ldi r18,lo8(399999)

ldi r24,hi8(399999)

ldi r25,hlo8(399999)

1: subi r18,1

sbci r24,0

sbci r25,0

brne 1b

rjmp .

nop

/\* send high \*/

sbi 0x8,0

rjmp .L3

.size main, .-main

.ident "GCC: (GNU) 4.9.1"

|  |  |  |  |
| --- | --- | --- | --- |
| 5. | ASM CODE WITH COMMENTS FOR TASK 1/B |  |  |

\_\_SP\_H\_\_ = 0x3e

\_\_SP\_L\_\_ = 0x3d

\_\_SREG\_\_ = 0x3f

\_\_tmp\_reg\_\_ = 0

\_\_zero\_reg\_\_ = 1

.text

.global init\_port

.type init\_port, @function

init\_port:

/\* prologue: function \*/

/\* frame size = 0 \*/

/\* stack size = 0 \*/

.L\_\_stack\_usage = 0

in r24,0x7 /\* SET PORTC.0 PORTC.5 as output\*/

ori r24,lo8(35)

out 0x7,r24 /\*SET PORTD.4 (PIN TO) as input\*/

cbi 0xa,4

ldi r24,lo8(-1)

out 0x4,r24 /\*SET PORTB as output\*/

ret

.size init\_port, .-init\_port

.global init\_counter0

.type init\_counter0, @function

init\_counter0:

/\* prologue: function \*/

/\* frame size = 0 \*/

/\* stack size = 0 \*/

.L\_\_stack\_usage = 0

ldi r24,lo8(4)

out 0x27,r24 /\*for Intput capture value at 5th pulse\*/

ldi r24,lo8(9)

out 0x28,r24 /\*for Intput capture value at 10th pusle\*/

in r24,0x15

ori r24,lo8(6)

out 0x15,r24 /\*clear the capture flag\*/

in r24,0x25

ori r24,lo8(7)

out 0x25,r24 /\*set clock as external\*/

ret

.size init\_counter0, .-init\_counter0

.section .text.startup,"ax",@progbits

.global main

.type main, @function

main:

/\* prologue: function \*/

/\* frame size = 0 \*/

/\* stack size = 0 \*/

.L\_\_stack\_usage = 0

call init\_port

call init\_counter0

ldi r25,lo8(2)

ldi r18,lo8(32)

.L5:

/\* start of while(1)\*/

ldi r19,lo8(399999)

ldi r20,hi8(399999)

ldi r24,hlo8(399999)

1: subi r19,1

sbci r20,0

sbci r24,0

brne 1b

rjmp .

nop

cbi 0x8,0

ldi r19,lo8(399999)

ldi r20,hi8(399999)

ldi r24,hlo8(399999)

1: subi r19,1

sbci r20,0

sbci r24,0

brne 1b

rjmp .

nop

sbi 0x8,0

/\* cheacking capture flags\*/

sbis 0x15,1

rjmp .L4

in r24,0x8

eor r24,r25

out 0x8,r24

in r24,0x27

subi r24,lo8(-(5))

out 0x27,r24

sbi 0x15,1

.L4:

sbis 0x15,2

rjmp .L5

in r24,0x8

eor r24,r18

out 0x8,r24

in r24,0x8

eor r24,r25

out 0x8,r24

in r24,0x28

subi r24,lo8(-(10))

out 0x28,r24

sbi 0x15,2

rjmp .L5

.size main, .-main

.ident "GCC: (GNU) 4.9.1"

|  |  |  |  |
| --- | --- | --- | --- |
| 6. | ASM CODE WITH COMMENTS FOR TASK 1/C |  |  |

\_\_SP\_H\_\_ = 0x3e

\_\_SP\_L\_\_ = 0x3d

\_\_SREG\_\_ = 0x3f

\_\_tmp\_reg\_\_ = 0

\_\_zero\_reg\_\_ = 1

.text

.global init\_port

.type init\_port, @function

init\_port:

/\* prologue: function \*/

/\* frame size = 0 \*/

/\* stack size = 0 \*/

.L\_\_stack\_usage = 0

in r24,0x7

ori r24,lo8(35)

out 0x7,r24

cbi 0xa,4

ldi r24,lo8(-1)

out 0x4,r24

ret

.size init\_port, .-init\_port

.global init\_counter0

.type init\_counter0, @function

init\_counter0:

/\* prologue: function \*/

/\* frame size = 0 \*/

/\* stack size = 0 \*/

.L\_\_stack\_usage = 0

ldi r24,lo8(4)

out 0x27,r24

ldi r24,lo8(9)

out 0x28,r24

in r24,0x15

ori r24,lo8(6)

out 0x15,r24

in r24,0x25

ori r24,lo8(7)

out 0x25,r24

ret

.size init\_counter0, .-init\_counter0

.section .text.startup,"ax",@progbits

.global main

.type main, @function

main:

/\* prologue: function \*/

/\* frame size = 0 \*/

/\* stack size = 0 \*/

.L\_\_stack\_usage = 0

call init\_port

call init\_counter0

ldi r25,lo8(2)

ldi r18,lo8(32)

.L5:

ldi r19,lo8(399999)

ldi r20,hi8(399999)

ldi r24,hlo8(399999)

1: subi r19,1

sbci r20,0

sbci r24,0

brne 1b

rjmp .

nop

cbi 0x8,0

ldi r19,lo8(399999)

ldi r20,hi8(399999)

ldi r24,hlo8(399999)

1: subi r19,1

sbci r20,0

sbci r24,0

brne 1b

rjmp .

nop

sbi 0x8,0

in r24,0x26

out 0x5,r24

sbis 0x15,1

rjmp .L4

in r24,0x8

eor r24,r25

out 0x8,r24

in r24,0x27

subi r24,lo8(-(5))

out 0x27,r24

sbi 0x15,1

.L4:

sbis 0x15,2

rjmp .L5

in r24,0x8

eor r24,r18

out 0x8,r24

in r24,0x8

eor r24,r25

out 0x8,r24

in r24,0x28

subi r24,lo8(-(10))

out 0x28,r24

sbi 0x15,2

rjmp .L5

.size main, .-main

.ident "GCC: (GNU) 4.9.1"

|  |  |  |  |
| --- | --- | --- | --- |
| 7. | ASM CODE WITH COMMENTS FOR TASK 1/D |  |  |

\_\_SP\_H\_\_ = 0x3e

\_\_SP\_L\_\_ = 0x3d

\_\_SREG\_\_ = 0x3f

\_\_tmp\_reg\_\_ = 0

\_\_zero\_reg\_\_ = 1

.text

.global init\_port

.type init\_port, @function

init\_port:

/\* prologue: function \*/

/\* frame size = 0 \*/

/\* stack size = 0 \*/

.L\_\_stack\_usage = 0

in r24,0x7

ori r24,lo8(35)

out 0x7,r24

cbi 0xa,4

ldi r24,lo8(-1)

out 0x4,r24

ret

.size init\_port, .-init\_port

.global init\_counter0

.type init\_counter0, @function

init\_counter0:

/\* prologue: function \*/

/\* frame size = 0 \*/

/\* stack size = 0 \*/

.L\_\_stack\_usage = 0

ldi r24,lo8(4)

out 0x27,r24

ldi r24,lo8(9)

out 0x28,r24

in r24,0x15

ori r24,lo8(6)

out 0x15,r24

ldi r30,lo8(110)

ldi r31,0

ld r24,Z

ori r24,lo8(6)

st Z,r24

in r24,0x25

ori r24,lo8(7)

out 0x25,r24

ret

.size init\_counter0, .-init\_counter0

.global \_\_vector\_14

.type \_\_vector\_14, @function

\_\_vector\_14:/\* COMPA\_Vector\*/

push r1

push r0

in r0,\_\_SREG\_\_

push r0

clr \_\_zero\_reg\_\_

push r24

push r25

/\* prologue: Signal \*/

/\* frame size = 0 \*/

/\* stack size = 5 \*/

.L\_\_stack\_usage = 5

in r25,0x8

ldi r24,lo8(2)

eor r24,r25

out 0x8,r24

in r24,0x27

subi r24,lo8(-(5))

out 0x27,r24

/\* epilogue start \*/

pop r25

pop r24

pop r0

out \_\_SREG\_\_,r0

pop r0

pop r1

reti

.size \_\_vector\_14, .-\_\_vector\_14

.global \_\_vector\_15

.type \_\_vector\_15, @function

\_\_vector\_15: /\* COMPB\_Vector\*/

push r1

push r0

in r0,\_\_SREG\_\_

push r0

clr \_\_zero\_reg\_\_

push r24

push r25

/\* prologue: Signal \*/

/\* frame size = 0 \*/

/\* stack size = 5 \*/

.L\_\_stack\_usage = 5

in r25,0x8

ldi r24,lo8(32)

eor r24,r25 /\*xor operation is used to toggle\*/

out 0x8,r24

in r25,0x8

ldi r24,lo8(2)

eor r24,r25

out 0x8,r24

in r24,0x28

subi r24,lo8(-(10))

out 0x28,r24

/\* epilogue start \*/

pop r25

pop r24

pop r0

out \_\_SREG\_\_,r0

pop r0

pop r1

reti

.size \_\_vector\_15, .-\_\_vector\_15

.section .text.startup,"ax",@progbits

.global main

.type main, @function

main:

/\* prologue: function \*/

/\* frame size = 0 \*/

/\* stack size = 0 \*/

.L\_\_stack\_usage = 0

call init\_port

call init\_counter0

/\* #APP \*/

; 48 "timer.c" 1

sei

; 0 "" 2

/\* #NOAPP \*/

.L6:

ldi r18,lo8(399999)

ldi r24,hi8(399999)

ldi r25,hlo8(399999)

1: subi r18,1

sbci r24,0

sbci r25,0

brne 1b

rjmp .

nop

cbi 0x8,0

ldi r18,lo8(399999)

ldi r24,hi8(399999)

ldi r25,hlo8(399999)

1: subi r18,1

sbci r24,0

sbci r25,0

brne 1b

rjmp .

nop

sbi 0x8,0

in r24,0x26

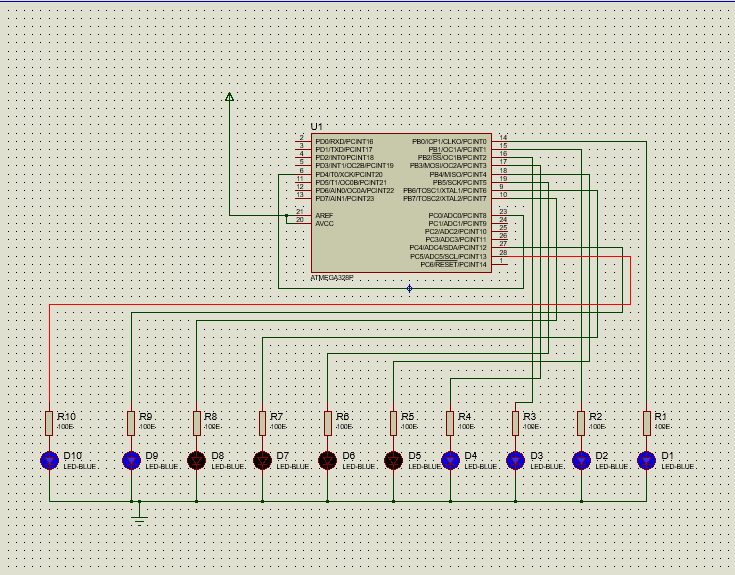
out 0x5,r24

rjmp .L6

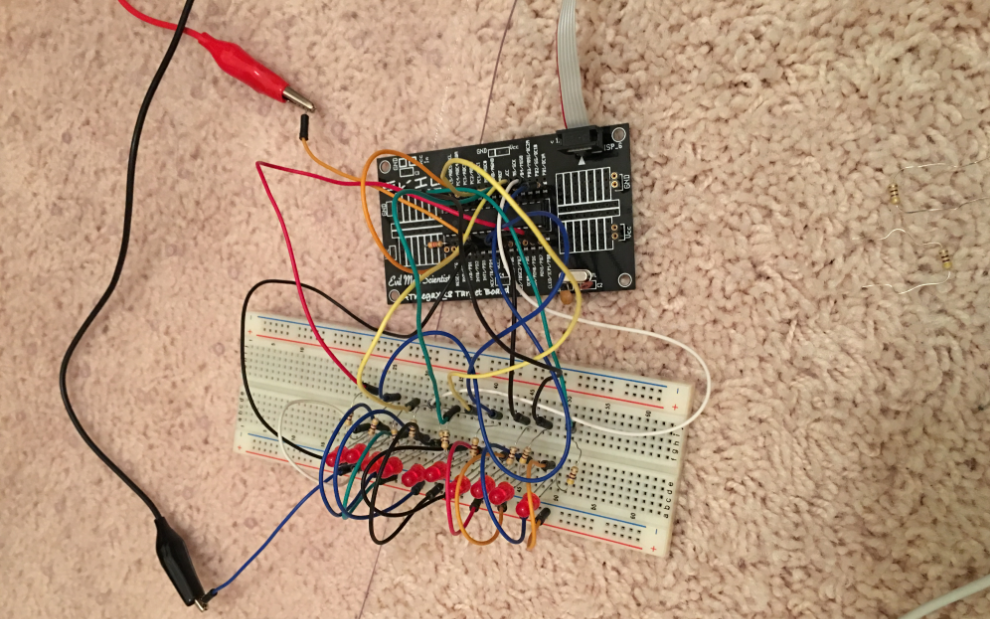
.size main, .-main

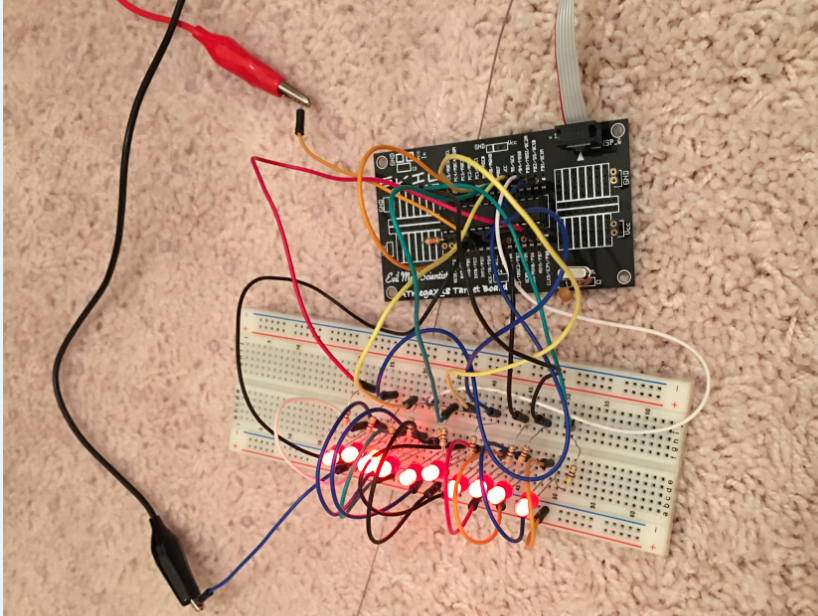
.ident "GCC: (GNU) 4.9.1"

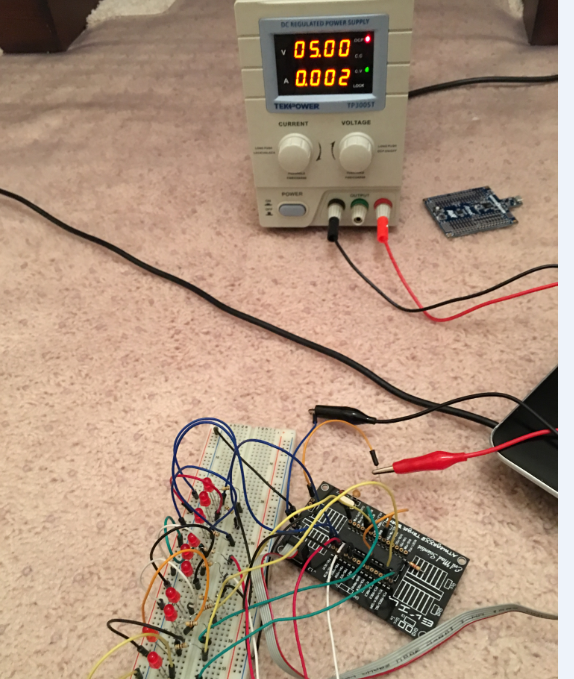
|  |  |  |  |
| --- | --- | --- | --- |
| 8. | SCHEMATIC |  |  |



|  |  |  |  |
| --- | --- | --- | --- |
| 9. | SNAPSHOT OF THE BOARD WITH CONNECTED COMPONENTS |  |  |







**Student Academic Misconduct Policy**

“This assignment submission is my own, original work”.

NAME OF THE STUDENT