B. D1 Lab Work

3.1

I wound 61 turns around the coil.
The value of inductor is 238.48 μH 14.252

The value of output is 1.23 V due to diode and resistor which limit it.

*

*

IL Meta embedded_6st program.

Current limit	Voltage out
0.2 A	~~7~~ 6 V
~~0.330 A~~	10 V
~~0.4 A~~	~~12 V~~
~~0.385 A~~	~~15 V~~

max!

Current	Voltage
0.537 A	15 V
0.358 A	10 V
0.184 A	5 V

Current limit iis 0.537 A.
My voltage do not achieve the 15 V because my inductor value excede 180 μH which mes used in simulations.

max V_{ABE} is 1.4 V. It is save voltage for IL metb.

Signed by Originator _____ Print Name _____ Date _____

Signed by Witness _____ Print Name _____ Date _____

For PWM = 240 $V_{out} = 5\,V$ $V_{ADC} = 0.960\,V$

ADC = 205

ADC = 370 $V_{out} = 10\,V$ $V_{ADC} = 1.78\,V$

ADC = 410 $V_{out} = 15\,V$ ~~$V_{ADC} = 1.350\,V$~~

$V_{ADC} = 2.67\,V$

$V_{out} = k \cdot V_{ADC}$

~~$k = \frac{V_{out}}{V_{ADC}} \approx 7.207$~~

$k = \frac{V_{out}}{V_{ADC}}$

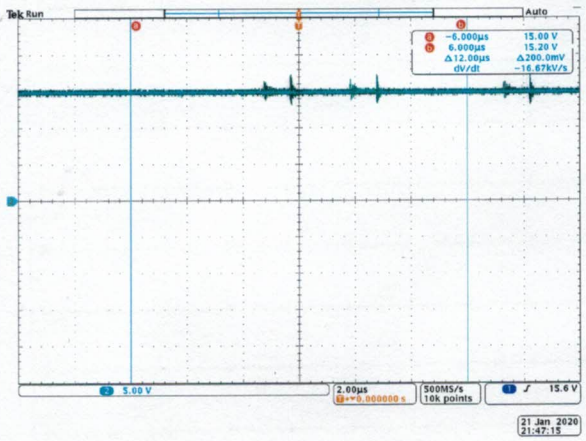
$k_1 = 5.2$

$k_2 = 5.6$ $k \approx 5.46$

$k_3 = 5.6$

```
0350: PWM=240 ==> ADC= 828 --> 14.583 Vload
0351: PWM=240 ==> ADC= 824 --> 14.513 Vload
0352: PWM=240 ==> ADC= 829 --> 14.601 Vload
0353: PWM=240 ==> ADC= 824 --> 14.513 Vload
0354: PWM=240 ==> ADC= 829 --> 14.601 Vload
0355: PWM=240 ==> ADC= 824 --> 14.513 Vload
0356: PWM=240 ==> ADC= 829 --> 14.601 Vload
0357: PWM=240 ==> ADC= 824 --> 14.513 Vload
0358: PWM=240 ==> ADC= 829 --> 14.601 Vload
0359: PWM=240 ==> ADC= 827 --> 14.566 Vload
0360: PWM=240 ==> ADC= 827 --> 14.566 Vload
0361: PWM=240 ==> ADC= 828 --> 14.583 Vload
0362: PWM=240 ==> ADC= 828 --> 14.583 Vload
0363: PWM=240 ==> ADC= 827 --> 14.566 Vload
0364: PWM=240 ==> ADC= 827 --> 14.566 Vload
0365: PWM=240 ==> ADC= 827 --> 14.566 Vload
0366: PWM=240 ==> ADC= 827 --> 14.566 Vload
0367: PWM=240 ==> ADC= 828 --> 14.583 Vload
0368: PWM=240 ==> ADC= 827 --> 14.566 Vload
0369: PWM=240 ==> ADC= 827 --> 14.566 Vload
```

```
uint16_t cnt =0;

init_stdio2uart0();
init_pwm();
init_adc();

for(;;) {
    printf( "%04d:  ", cnt );

    pwm_duty(255);   /* Limited by PWM_DUTY_MAX */

    // printf( "  -->  %5.3f Vadc", v_load() );

    printf( "  -->  %5.3f Vload\r\n", v_load()*K );
    _delay_ms(DELAY_MS);
    cnt++;
}
```



Ready 22/1

Lab Code:	None	Poor	OK	Good	Exceptional	Outstanding
Preparation	0	1	2	3	4	5
Progress	0	1	2	3	(4)	5
Understanding	0	1	2	3	(4)	5
Logbook Use	0	1	2	3	(4)	5

Demonstrators only

Print Full Name	Meijing Liu	Signed	刘美婧

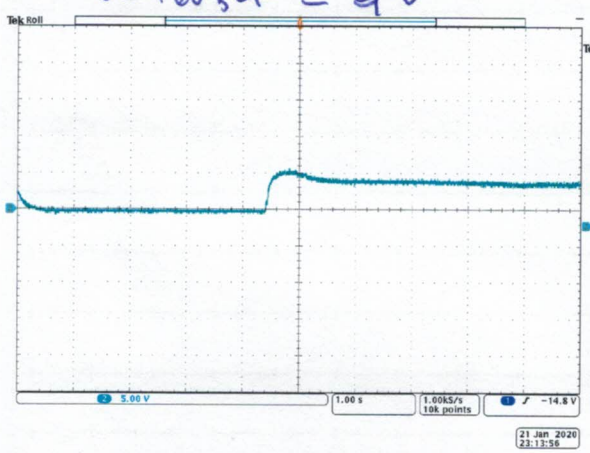
Signed by Originator ..... Print Name ..... Date .....

Signed by Witness ..... Print Name ..... Date .....

4. Programming

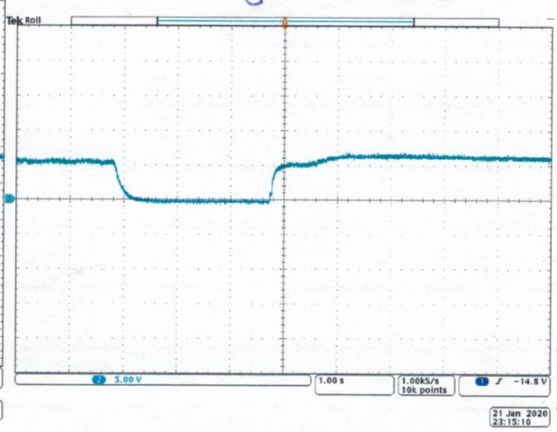
```
for(;;) {
    printf( "%04d:  ", cnt );
    printf( "Error: %5.3f:  ", diferror );
    pwm_duty(dutyCycle);   /* Limited by PWM_DUTY_MAX */

   // printf( "  -->  %5.3f Vadc", v_load() );

    printf( "  -->  %5.3f Vload\r\n", Vout=v_load());
    //PI controller
    //Change duty cycle based on error
    error=Vout-V_target;
    integerror = integerror + error*(DELAY_MS)/1000;     //integral of e(t)
    diferror = (error-errorprev)/DELAY_MS*1000;                 //derivative (slope) of e(t)

    dutyCycle = dutyCycle - error*kP;           // P term
    dutyCycle = dutyCycle - integerror*kI;      // I term
    dutyCycle = dutyCycle - diferror*kD;           // D term

    errorprev = error;
    _delay_ms(DELAY_MS);
    cnt++;

}
}
```

V_target = 4 V



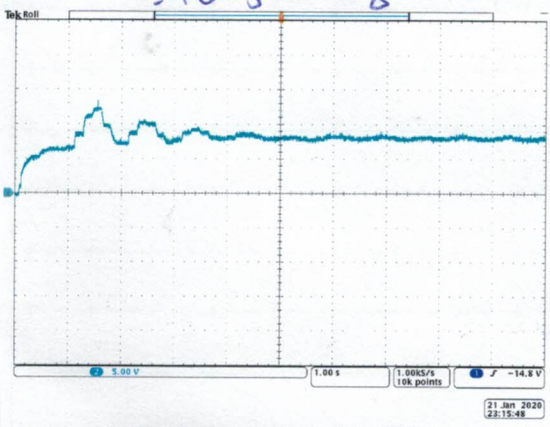
MDO4054B-3 - 16:19:57 21/01/2020

V_target = 6 V



MDO4054B-3 - 16:21:10 21/01/2020

V_target = 8 V



MDO4054B-3 - 16:21:48 21/01/2020

PID controller.

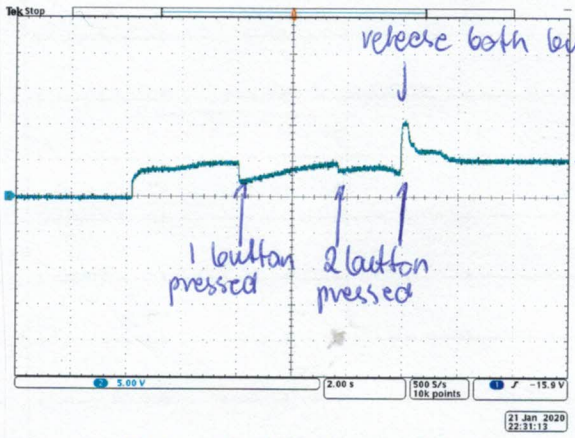
Signed by Originator ..... Print Name ..... Date .....

Signed by Witness ..... Print Name ..... Date .....

```
0001:  PWM=138  ==>  ADC= 270  -->  4.755 Vload
0002:  PWM=138  ==>  ADC= 292  -->  5.143 Vload
0003:  PWM=137  ==>  ADC= 295  -->  5.196 Vload
0004:  PWM=136  ==>  ADC= 293  -->  5.161 Vload
0005:  PWM=135  ==>  ADC= 294  -->  5.178 Vload
0006:  PWM=134  ==>  ADC= 293  -->  5.161 Vload
0007:  PWM=133  ==>  ADC= 292  -->  5.143 Vload
0008:  PWM=132  ==>  ADC= 292  -->  5.143 Vload
0009:  PWM=131  ==>  ADC= 141  -->  2.483 Vload
0010:  PWM=138  ==>  ADC= 141  -->  2.483 Vload
0011:  PWM=145  ==>  ADC= 150  -->  2.642 Vload
0012:  PWM=152  ==>  ADC= 160  -->  2.818 Vload
0013:  PWM=158  ==>  ADC= 171  -->  3.012 Vload
0014:  PWM=163  ==>  ADC= 180  -->  3.170 Vload
0015:  PWM=168  ==>  ADC= 191  -->  3.364 Vload
0016:  PWM=172  ==>  ADC= 201  -->  3.540 Vload
0017:  PWM=176  ==>  ADC= 210  -->  3.699 Vload
0018:  PWM=179  ==>  ADC= 220  -->  3.875 Vload
0019:  PWM=182  ==>  ADC= 228  -->  4.016 Vload
0020:  PWM=184  ==>  ADC= 235  -->  4.139 Vload
0021:  PWM=186  ==>  ADC= 241  -->  4.245 Vload
0022:  PWM=188  ==>  ADC= 248  -->  4.368 Vload
0023:  PWM=189  ==>  ADC= 254  -->  4.474 Vload
0024:  PWM=190  ==>  ADC= 261  -->  4.597 Vload
0025:  PWM=191  ==>  ADC= 260  -->  4.579 Vload
```

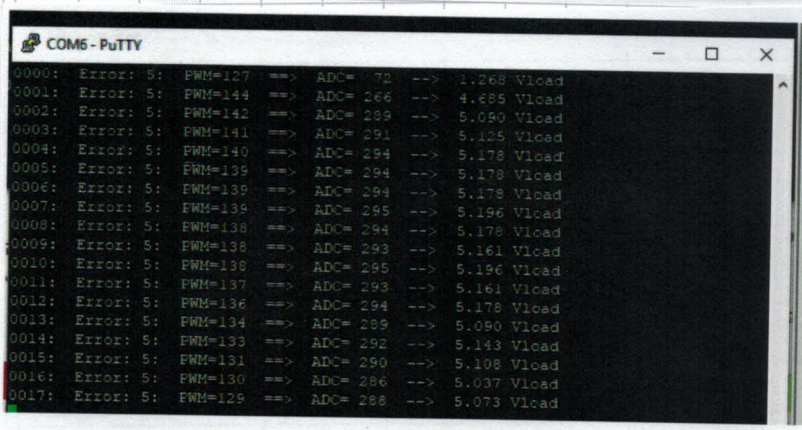
```
59      double kP=3;
60      double Vout;
61      double error;
62
63      for(;;) {
64          printf( "%04d:  ", cnt );
65          //printf( "Error: %5.3f:  ", error );
66          pwm_duty(dutyCycle);   /* Limited by PWM_DUTY_MAX */
67
68          // printf( "  -->  %5.3f Vadc", v_load() );
69
70          printf( "  -->  %5.3f Vload\r\n", Vout=v_load());
71          //Change duty cycle based on error
72      error=Vout-V_target;
73      dutyCycle=dutyCycle-error*kP;
74
75          _delay_ms(DELAY_MS);
76          cnt++;
77
78      }
79  }
80
81  int uputchar0(char c, FILE *stream)
```

C source file — length : 4,085 lines : 175



P controller.

Lab Code: PI week	None	Poor	OK	Good	Exceptional	Outstanding
Preparation	0	1	2	3	4	5
Progress	0	1	2	3	4	5
Understanding	0	1	2	3	4	5
Logbook Use	0	1	2	3	4	5
Demonstrators only						
Print Full Name					Signed	




```
68
69      V_targetc = ugetchar0(stdin);
70      V_target = V_targetc - '0';
71
72
73
74      for(;;) {
75          printf( "%04d:  ", cnt );
76          printf( "Error: %d:  ", V_t
77          pwm_duty(dutyCycle);   /* L
78
79      // printf( "  -->  %5.3f Vad
80
81          printf( "  -->  %5.3f Vload
82          //PI controller
83          //Change duty cycle based o
84          error=Vout-V_target;
85          integerror = integerror + e
86          diferror = (error-errorprev
87
88          dutyCycle = dutyCycle - err
```

Keyboard controll

The keyboard control, char is getting from keyboard

Signed by Originator ..... Print Name ..... Date .....

Signed by Witness ..... Print Name ..... Date .....

end program set V_target;

```
ISR(INT0_vect)
{

V_targetc = ugetchar0(stdin);
V_target = V_targetc - '0';
if(!V_target)V_target = 10;
//if(V_target > 10 && V_target < 2) PORTD |= _BV(PB7); //turn on led
//else PORTD &= ~_BV(PB7);
diferror = 0;
integerror = 0;
error = 0;
}
```

ISR: Externel interupt is tuggered by button. While button is pressed, V_terget can be intered from keyboard.

Drawing functions

```
void draw_rectangle(int x,int y, uint16_t c){

    rectangle r = {x,x+100,y,y+20};
    fill_rectangle(r, c);
    rectangle r2 = {x+1,x+99,y+1,y+19};
    fill_rectangle(r2,BLACK);

}

void init_pwm1(int x,int y, uint16_t c){


    rectangle r1 = {x,x+200,y,y+10};
    fill_rectangle(r1,c);

    rectangle r2 = {x+1,x+199,y+1,y+9};
    fill_rectangle(r2,BLACK);

}
```

Drawing table and values on LCD screen

```
    init_lcd();
display.background = BLACK;
display.foreground = WHITE;
clear_screen();
set_orientation(North);

    clear_screen();

    char buf [20];

    draw_rectangle(5,5,WHITE);

    display.x = 18; display.y = 13; display_string("VOLTAGE LOAD");

    draw_rectangle(5,25,WHITE);

    display.x = 18; display.y = 33; display_string("VOLTAGE ADC");

    draw_rectangle(5,45,WHITE);

    display.x = 18; display.y = 53; display_string("VOLTAGE TARGET");

    draw_rectangle(5,65,WHITE);

    display.x = 18; display.y = 73; display_string("ERROR");

    draw_rectangle(5,85,WHITE);

    display.x = 18; display.y = 93; display_string("DUTY CYCLE");

    draw_rectangle(5,105,WHITE);

    init_pwm1(20,200,WHITE);

    display.x = 110;
    display.y = 180;
    display_string("PWM %");

    display.x = 20;
    display.y = 180;
    display_string("0 %");

    display.x = 200;
    display.y = 180;
    display_string("100 %");

    //LCD
            draw_pwm(20,200,dutyCycle,WHITE);

            dtostrf(Vout, 6, 2, buf);
            display.x = 120; display.y = 13; display_string(buf); //DISPLAY V LOAD
            dtostrf(Vadc, 6, 2, buf);
            display.x = 120; display.y = 33; display_string(buf); //DISPLAY V ADC

            display.x = 120; display.y = 53; display_string(Vchar); //DISPLAY V TARGET
            dtostrf(error, 6, 2, buf);
            display.x = 120; display.y = 73; display_string(buf); //DISPLAY Error


            //dtostrf(dutyCycle, 6, 2, buf);
            //display.x = 120; display.y = 93; display_string(buf); //DISPLAY Duty Cycle
            _delay_ms(DELAY_MS);
```

e _____ Date

Signed by Witness _____ Print Name _____ Date

Thu.

Lab Code 21.	None	Poor	OK	Good	Exceptional	Outstandin
Preparation	0	1	2	3	4	5
Progress	0	1	2	3	4	5
Understanding	0	1	2	3	4	5
Logbook Use	0	1	2	3	4	5
Demonstrators only						
Print Full Name	Colin Zhang.				Signed	

Tuning

P slow 7 0 0

P1 6.5 0.5 0

PID 5 0.5 0.005

```
inp:

printf("=====MENU=====\n");
printf("1. Change target load voltage\n");
printf("2. Change control system\n");
printf("3. Resume\n");
fail = scanf("%d",&input);

if(fail != 1 || (input != 1 && input != 2 && input != 3)){
    if(fail != 1){
        scanf("%c",&bill);
        printf("Input: %c\n", bill);
    }
    else printf("Input: %d\n", input);

    goto erroruser;
}
printf("Input: %d\n", input);
```

The program wait for input and checks if output is correct.

```
switch (input)
{
    case 1: goto volchange;
            break;
    case 2: goto ctrchange;
            break;
    case 3: goto start;
            break;
    default: goto erroruser;
```

Signed by Originator ..... Print Name ..... Date .....

Signed by Witness ..... Print Name ..... Date .....

```
//CHANGE TARGET VOLTAGE MENU
volchange:
printf("=====TARGET VOLTAGE CHANGE=====\n");
printf("Type desire target load voltage: [range 2-10]\n");
fail = scanf("%d",&V_target);
Vchar = V_target + '0';
if(V_target > 10 || V_target < 2 || fail != 1){

    if(fail != 1){scanf("%c",&bill);
        printf("Input: %c\n", bill);
    }
    else
        printf("Input: %d\n", V_target);

        printf("Error, input not valid.\n");
    goto erroruser;

}
printf("Input: %d\n", V_target);
goto start;
```

```
        fgets(string1,5,stdin);
        kP = (string1[0] - '0') + ((double) (string1[2] - '0'))*0.1 + ((double) (string1[3] - '0'))*0.01;
        goto ctrchange;
        break;
    case 6:
        fgets(string1,5,stdin);
        kI = (string1[0] - '0') + ((double) (string1[2] - '0'))*0.1 + ((double) (string1[3] - '0'))*0.01;
        goto ctrchange;
        break;
    case 7:
        fgets(string1,5,stdin);
        kD = (string1[0] - '0') + ((double) (string1[2] - '0'))*0.1 + ((double) (string1[3] - '0'))*0.01;
        goto ctrchange;
        break;
```

fgets() function get input from keyboard. Then, the value is converted into float by algorithm.

System error detector.

```
//SYSTEM ERROR RECEIVER
if(cnt > 40 && (error > 4 || error < -4)){
    pwm_duty(5);
    break;
}
```

The program checks if the boost converter is in steady state mode. ~~and then can~~ If its not, and error keeps high, system error is detected.

Users error:

Program blink diode for 0.1 sec if user input error is detected.

Signed by Originator _____ Print Name _____ Date _____

Signed by Witness _____ Print Name _____ Date _____