



ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์

242-302 ADV COM ENG LAB II

3HB05,3HB06 AVR Microcontroller

เสนอ

อาจารย์ พิชรี เทพนมิตร

จัดทำโดย

นายอรรณพร ศรีปานรอด

ID student 5635512057 section 02

Prince of Songkla University Phuket Campus

Year 2558

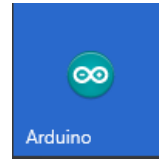
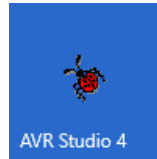
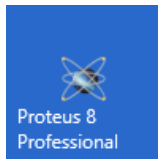
3HB05,3HB06 AVR Microcontroller

► วัตถุประสงค์

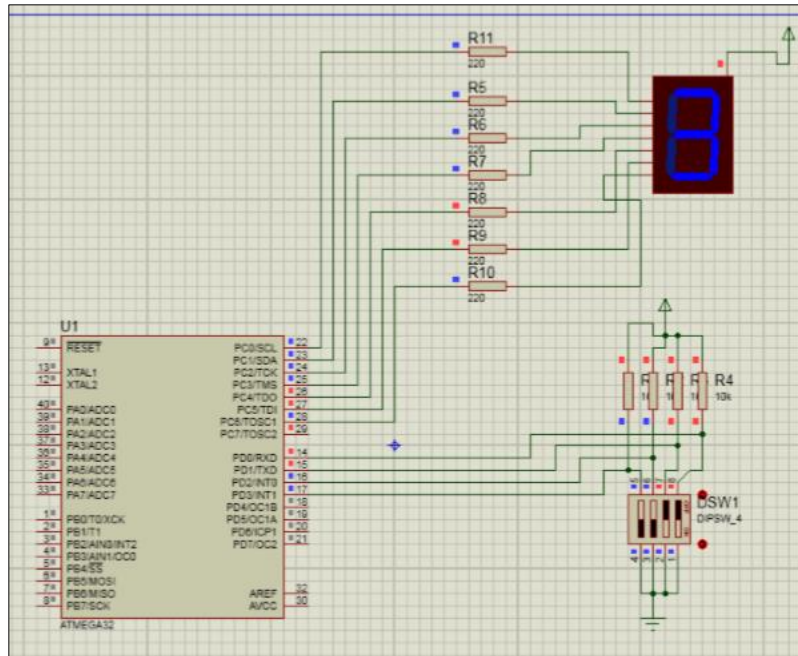
- เพื่อให้นักศึกษาได้เรียนรู้วิธีการเขียนโปรแกรมประยุกต์ใช้งานไมโครคอนโทรลเลอร์ AVR ด้วยภาษา Assembly และ C
- เพื่อให้นักศึกษาได้เรียนรู้เทคนิคการดีบั๊กโปรแกรมภาษา Assembly และ C ก่อนที่จะนำโปรแกรมลงทดสอบบนฮาร์ดแวร์จริง
- เพื่อให้นักศึกษาได้เรียนรู้เทคนิคการออกแบบฮาร์ดแวร์และซอฟต์แวร์ร่วมกัน เพื่อให้ง่ายในการทดสอบความถูกต้อง
- เพื่อให้นักศึกษาได้ฝึกการเขียนซอฟต์แวร์สำหรับการทดสอบความถูกต้องของฮาร์ดแวร์

► อุปกรณ์ที่ใช้ในการทดลอง

- AVR Studio 4
- Proteus 8 professional
- Arduino 1.6.8
- บอร์ดไมโครคอนโทรลเลอร์ AVR รุ่น Diecimila ATmega32
- บอร์ดทดลอง



Proteus จำลองการทำงานของโปรแกรม



▶ CHECKPOINT 2 เขียนโปรแกรม เพื่อนับลอจิกต่ำจากดิปสวิตช์

โปรแกรมจะทำการอ่านค่าจากดิปสวิตช์จำนวน 8 ตัว ที่ต่อกับพอร์ต D จากนั้น ทำการตรวจสอบบิตที่มาค่าลอจิกต่ำ และแสดงผลออก 7-segment ที่ต่อที่พอร์ต C

C Programming

```

#include <avr/io.h>
int main(void){
    DDRD = 0x00; // set port B as input
    DDRC = 0xFF; // set port C as output
    unsigned char SWITCH_V, DISP, i, count, Bitposition, test_bit;
    unsigned char LOOKUPTB[] = {
        0b00111111, 0b00000110, //0,1
        0b01011011, 0b01001111, //2,3
        0b01100110, 0b01101101, //4,5
        0b01111101, 0b00000111, //6,7
        0b01111111, 0b01101111, //8,9
        0b01110111, 0b01111100, //10,11
        0b00111001, 0b01011110, //12,13
        0b01111001, 0b01110001 //14,15
    };

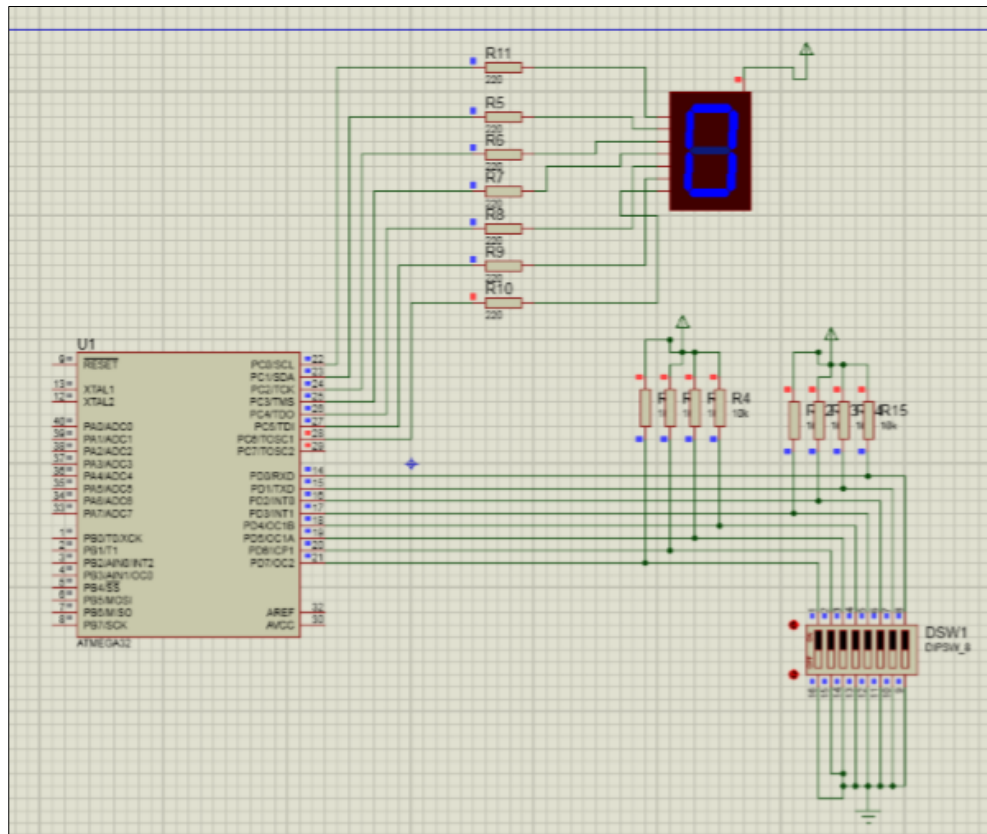
    while (1){
        SWITCH_V = PIND;
        count = 0;
        Bitposition = 0x01;
        for (i=0;i<8;i++){
            test_bit = SWITCH_V & (Bitposition << i);
            if (!test_bit)
                count++;
        }
        DISP = LOOKUPTB[count];
        PORTC = DISP ;
    }
}
  
```

ให้ PORTD เป็น Input (รับข้อมูลจาก dipswitch) และ PORTC เป็น Output

Count เป็นตัวนับ ใน loop while

While loop ใช้วนนับโดย จะทำการชิป (Bitposition << 1) แล้วเช็คว่าเป็น High หรือไม่
*หากเป็น High ให้เพิ่มค่า Count++ แล้ววนเช็คบิตถัดไปจนครบ 8 บิต ก็จะได้จำนวนของ logic low แล้วนำค่าที่ได้ไปชี้ใน LOOKUPTB[] และให้แสดงผลออกมาทาง 7-segment

Proteus จำลองการทำงานของโปรแกรม



▶ CHECKPOINT 3 เขียนโปรแกรม เพื่ออ่านค่าจากสวิตช์ โดยข้อมูลเป็นเลข signed number

จงเขียนโปรแกรมเพื่อแปลงค่าจากสวิตช์เป็นเลขฐานสิบแบบมีเครื่องหมาย ซึ่งมีค่าอยู่ระหว่าง -8...+7
 ออกแสดงผลทางแอลอีดี แบบ 7 เซกเมนต์ 2 ตัว

```

check3 | Arduino 1.6.8
File Edit Sketch Tools Help

check3

#include <avr/io.h>
int main (void)
{
    unsigned char LOOKUPTB[] = { 0b00111111, //0
                                0b00000110,0b01011011, //1,2
                                0b01001111,0b01100110, //3,4
                                0b01101101,0b01111101, //5,6
                                0b00000111,0b11111111, //7,8
                                0b10000111,0b11111101, //-7,-6
                                0b11101101,0b11100110, //-5,-4
                                0b11001111,0b11011011, //-3,-2
                                0b10000110 }; // -1

    unsigned char DISPLY, SWITCH_V;

    DDRC = 0xFF;
    DDRD = 0x00;
    while(1)
    {
        SWITCH_V = PIND;
        SWITCH_V &= 0xF0;
        SWITCH_V = SWITCH_V >> 4;
        DISPLY = LOOKUPTB[SWITCH_V];
        PORTC = DISPLY;
    }
}

```

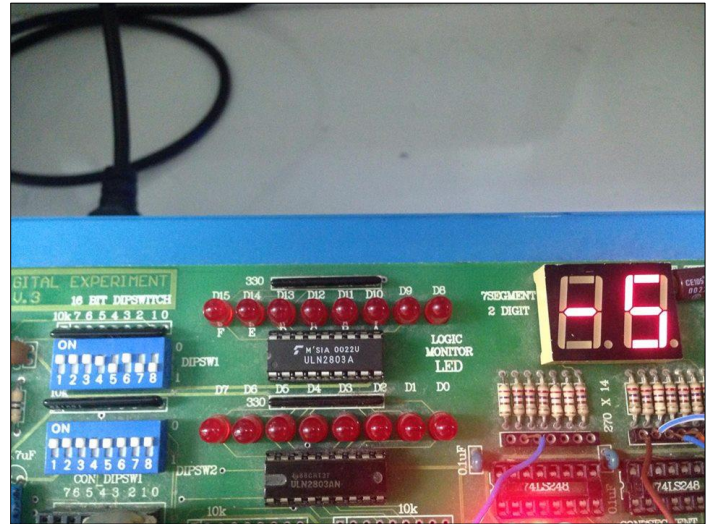
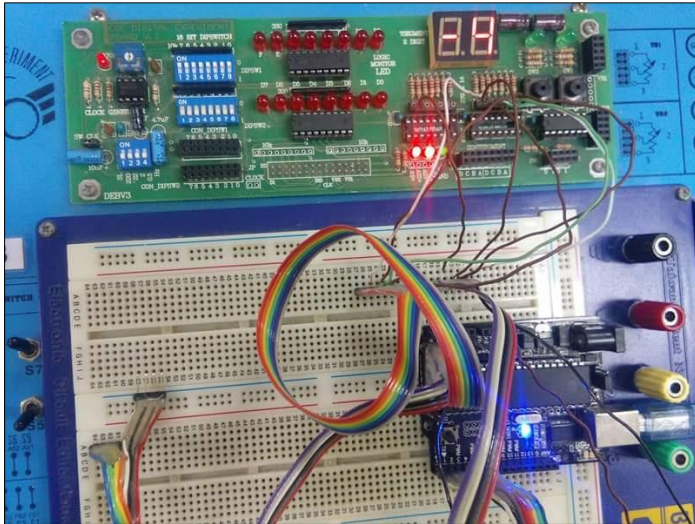
LOOKUPTB[] ให้แสดงผลออกมาทาง 7-segment โดยกำหนด ให้มีค่าเลข ติดลบ โดยในวงจรจะใช้ 7-segment 2 ตัว

ให้ DDRC เป็น output
 และ DDRD เป็น input ใช้ 4 บิต

Shift 4 bit และให้ชี้ตำแหน่งไปที่ LOOKUPTABLE[] โดยในตารางจะมีค่าระหว่าง -8 ถึง +7 ในส่วนที่เป็น ลบ จะกำหนด sign bit = 1 หากไม่มีก็กำหนดให้เป็น 0

Done compiling.

เมื่อนำโปรแกรมอัลดงบอร์ด



▶ CHECKPOINT 4 เขียนโปรแกรม เพื่อแสดงค่าจากสวิตช์ โดยข้อมูลเป็นเลขฐานสิบ

จงเขียนโปรแกรมเพื่อแปลงค่าจากสวิตช์เป็นเลขฐานสิบแบบไม่มีเครื่องหมาย ซึ่งมีค่าอยู่ระหว่าง 0-15 ออกแสดงผลทางแอลอีดี แบบ 7 เซกเมนต์ 2 ตัว

The screenshot shows the Arduino IDE interface with the following code:

```

check4 | Arduino 1.6.8
File Edit Sketch Tools Help

check4 §
#include <avr/io.h>
int main (void)
{
    unsigned char LOOKUPTB[] = { 0b00111111, //0
                                0b00000110, 0b01011011, //1,2
                                0b01001111, 0b01100110, //3,4
                                0b01101101, 0b01111101, //5,6
                                0b00000111, 0b01111111, //7,8
                                0b01100111, 0b10111111, //9,10
                                0b10000110, 0b11011011, //11,12
                                0b11001111, 0b11100110, //13,14
                                0b11101101 //15
    };

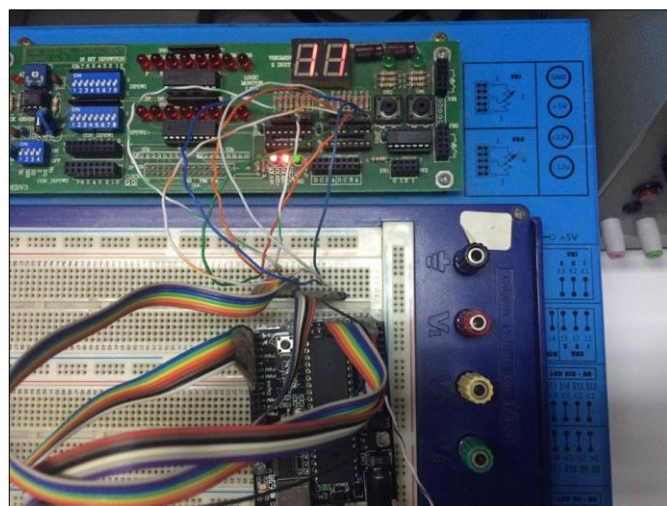
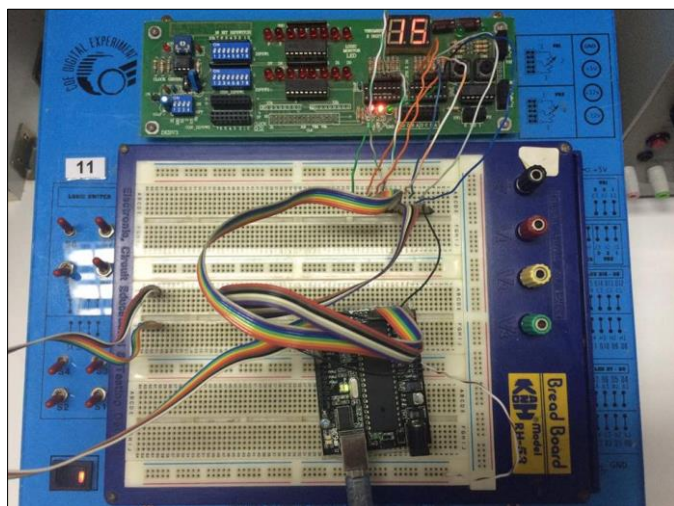
    unsigned char DISPLY, SWITCH_V;

    DDRC = 0xFF;
    DDRD = 0x00;
    while(1)
    {
        SWITCH_V = PIND;
        SWITCH_V &= 0x0F;
        SWITCH_V = SWITCH_V >> 4;
        DISPLY = LOOKUPTB[SWITCH_V];
        PORTC = DISPLY;
    }
}
  
```

Annotation 1: LOOKUPTB[] ให้แสดงผลออกมาทาง 7-segment โดยกำหนด ให้แสดงเลขตั้งแต่ 0-15 โดยในวงจรจะใช้ 7-segment 2 ตัว




Annotation 2: เลข 1 ข้างหน้า (10-15) กำหนดให้ signed bit เป็น 1 แต่จะมีการต่อของสายจุด b กับ C ใน 7-segment ของบอร์ดทดลอง ดยหลักการจะ คล้าย check 3

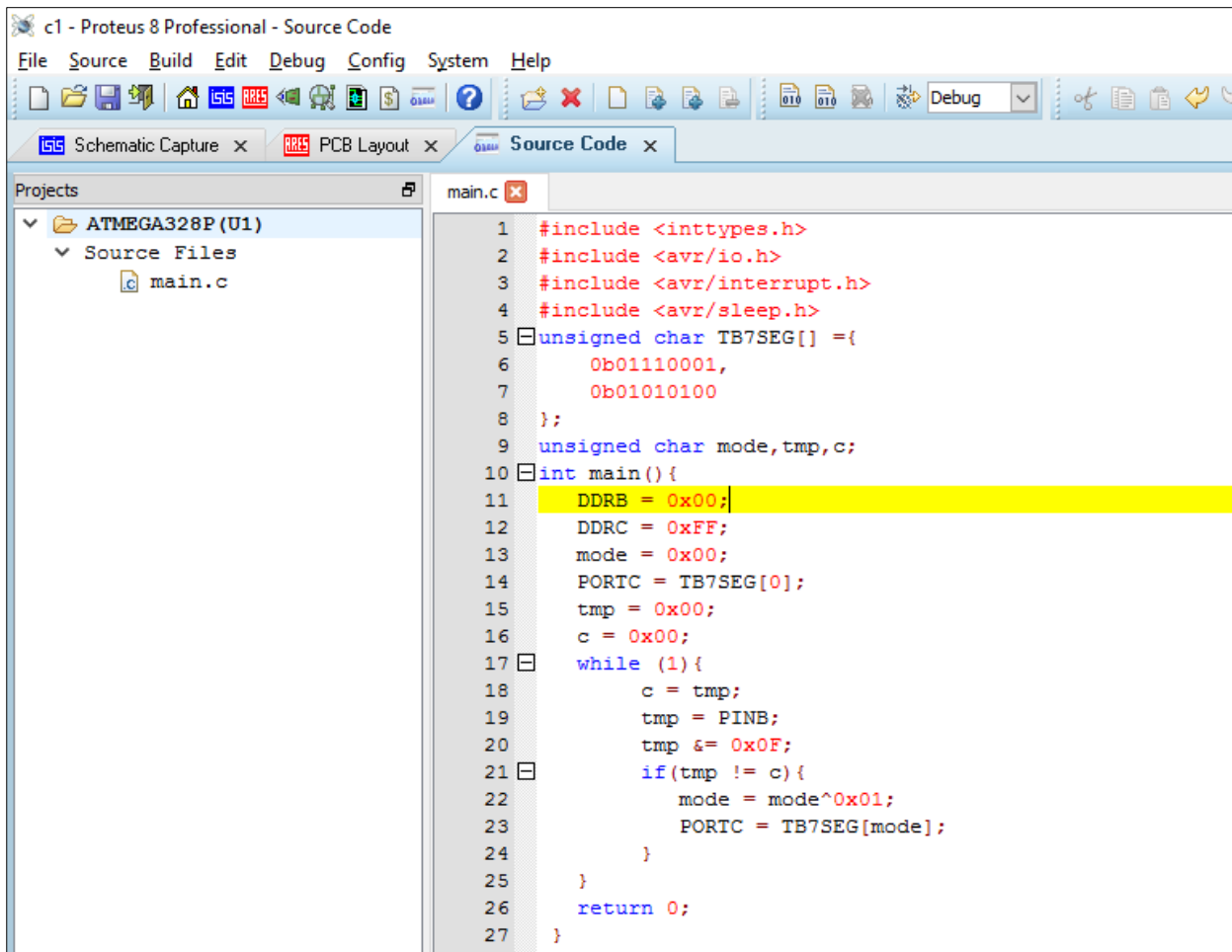
เมื่อนำโปรแกรมอัลดาวน์โหลด



การทดลองที่ 5 ออกแบบและทดสอบระบบสวิทช์สัมผัสสำหรับเปิดปิดเครื่องใช้ไฟฟ้า

▶ CHECKPOINT 5.1 ออกแบบและทดสอบระบบสวิทช์สัมผัสสำหรับเปิดปิดเครื่องใช้ไฟฟ้า

- เมื่อเริ่มจ่ายไฟให้กับโปรแกรม หลอด L1 จะดับอยู่ และ 7-segment แสดงคำว่า 
- เมื่อกดสวิทช์ หลอด L1 จะติด และ 7-segment แสดงคำว่า 
- เมื่อกดสวิทช์อีกครั้ง หลอด L1 จะดับ และ 7-segment แสดงคำว่า 

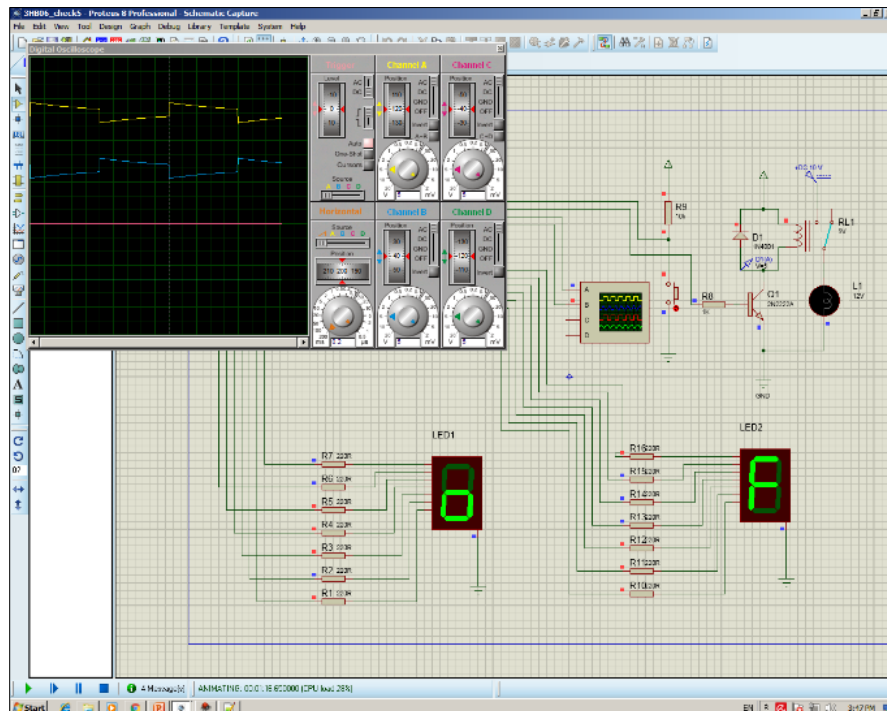
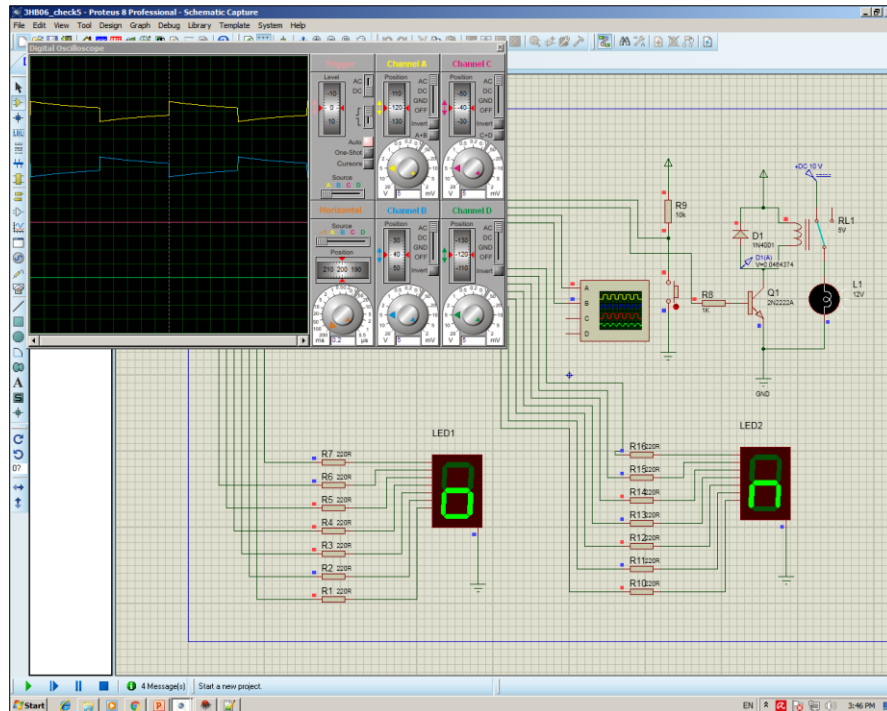


```

1  #include <inttypes.h>
2  #include <avr/io.h>
3  #include <avr/interrupt.h>
4  #include <avr/sleep.h>
5  unsigned char TB7SEG[] = {
6      0b01110001,
7      0b01010100
8  };
9  unsigned char mode, tmp, c;
10 int main() {
11     DDRB = 0x00;
12     DDRC = 0xFF;
13     mode = 0x00;
14     PORTC = TB7SEG[0];
15     tmp = 0x00;
16     c = 0x00;
17     while (1) {
18         c = tmp;
19         tmp = PINB;
20         tmp &= 0x0F;
21         if (tmp != c) {
22             mode = mode ^ 0x01;
23             PORTC = TB7SEG[mode];
24         }
25     }
26     return 0;
27 }

```

ผลลัพธ์ของวงจร ที่ได้ใน program proteus



▶ CHECKPOINT 5.2 เขียนโปรแกรมสร้างพัลส์ ออก Oscilloscope

- เมื่อเริ่มจ่ายไฟให้กับโปรแกรม สัญญาณพัลส์ที่วัดได้ที่ Oscilloscope จะมีความถี่ 0.5 Hz (คาบ 2 sec)

The screenshot shows the Proteus 8 Professional Source Code window for a project named '3HB06_check5'. The code is written in C for an ATmega328P microcontroller. The code includes headers for `<inttypes.h>`, `<avr/io.h>`, `<avr/interrupt.h>`, and `<avr/sleep.h>`. It defines a constant `VALUE_T1` as 34286. The `main` function initializes the DDR registers (DDRB, DDRC, DDRD) to 0xFF, sets the TCCR1A and TCCR1B registers, and starts the timer. It then enters a `while (1)` loop. The `ISR(TIMER1_OVF_vect)` function is called when the timer overflows, and it updates the `TCNT1` register to `VALUE_T1` and toggles the `PORTB` register.

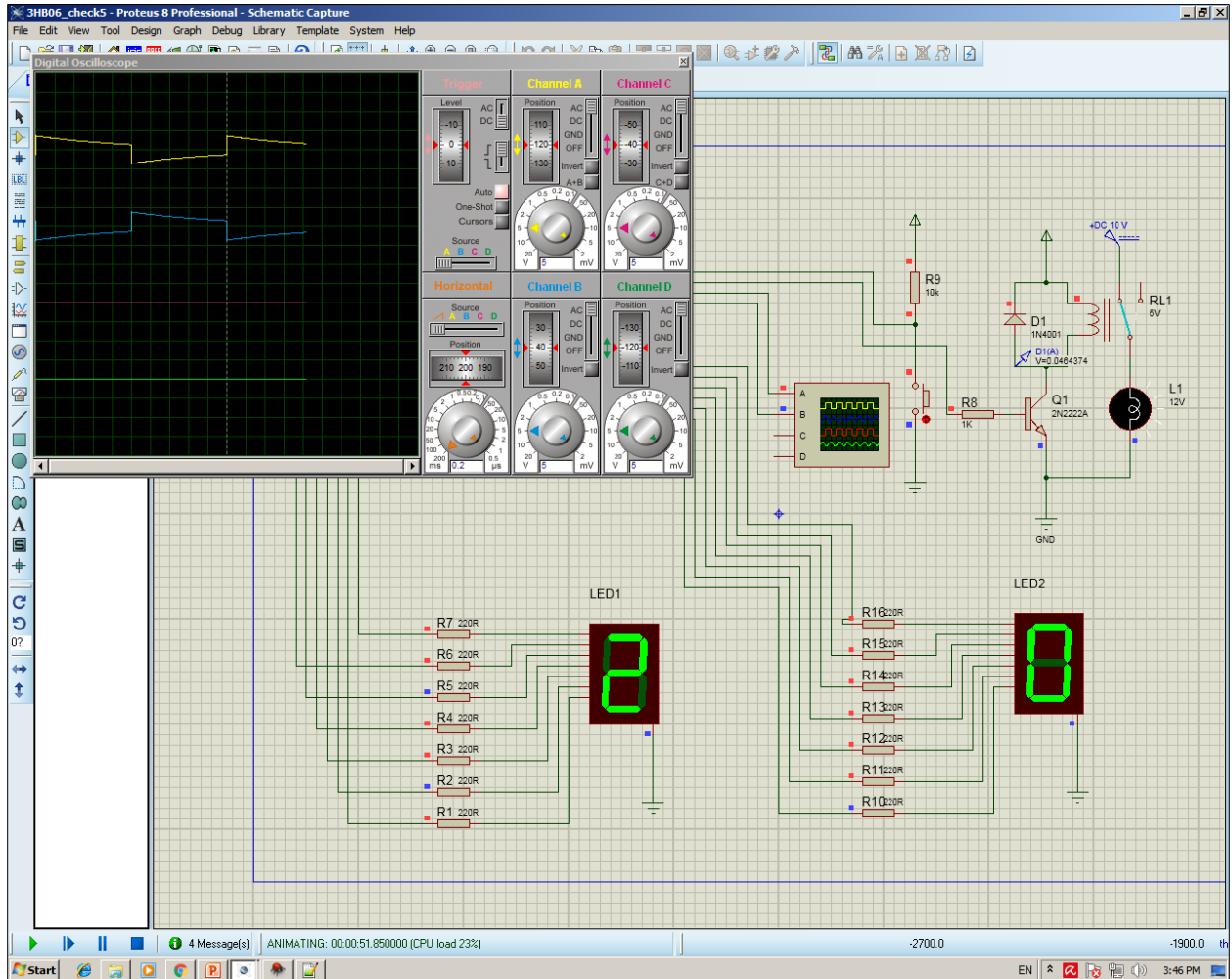
Callout box text: `#define VALUE_T1 34286` เป็นตัวกำหนดความเร็วในการนับถอยหลัง

```

1  #include <inttypes.h>
2  #include <avr/io.h>
3  #include <avr/interrupt.h>
4  #include <avr/sleep.h>
5  #define VALUE_T1 34286
6
7  int main()
8  {
9      DDRB = 0xFE;
10     DDRC = 0xFF;
11     DDRD = 0xFF;
12
13
14     TCCR1A = 0x00;
15     TCCR1B = 0x04;
16     TIMSK1 = 0x01;
17
18     cli();
19     TCNT1 = VALUE_T1;
20     sei();
21     // Write your code here
22     while (1)
23     ;
24     return 0;
25 }
26
27 ISR(TIMER1_OVF_vect){
28     TCNT1 = VALUE_T1;
29     PORTB = PORTB ^ 0x0C;
30 }

```

ผลลัพธ์ของวงจร ที่ได้ใน program proteus



▶ CHECKPOINT 5.3 เขียนโปรแกรมตั้งเวลา

- เมื่อเริ่มจ่ายไฟให้กับโปรแกรม 7-segment แสดงเลข 20 เมื่อกดสวิตช์ 7-segment จะเริ่มนับถอยหลังจากทุกๆ 1 วินาที จะมีค่าลดลงทีละหนึ่ง จนเหลือ 00
- ถ้ากดสวิตช์อีกครั้ง 7-segment จะรีเซ็ตจากค่า 00 มาเป็น 20
- ถ้ากดสวิตช์อีกครั้ง 7-segment จะนับถอยหลังไปที่เลข 00 การทำงานจะเป็นสลับกันไปมา

```
main.c
1 #include <avr/io.h>
2 #include <util/delay.h>
3 #define F_CPU 16000000UL
4 int main (void){
5     unsigned char switch_v ,count,i;
6     unsigned char LOOKUPTB[]={ 0b00111111, 0b00000110,
7                                0b01011011, 0b01001111,
8                                0b01100110, 0b01101101,
9                                0b01111101, 0b00000111,
10                               0b01111111, 0b01101111,
11                               0b01110111, 0b01111100,
12                               0b00111001, 0b01011110,
13                               0b01111001, 0b01110001 };
14     DDRC = 0xFF;
15     DDRB = 0xFF;
16     DDRD = 0xFE;
17
18     count = 0;
19     PORTB = 0b01011100; // o
20     PORTC = 0b01110001; // F
21
22     while(1){
23         switch_v = PIND; // input from PIND
24         switch_v &= 0x01; // Check Bit equal 0
25         if(switch_v == 0x00){
26             if(count == 0){
27                 PORTB = 0b01011100; // o
28                 PORTC = 0b01010100; // n
29                 PORTD = 0b00000010; //Lamp on
30                 count = 1;
31                 _delay_ms(8000); // delay 1 sec
32             }
33
34             else if(count == 1){
35                 for(i=0;i<21;i++){
36                     _delay_ms(8000); // delay 1 sec
37                     PORTB = LOOKUPTB[(20-i)/10]; // Tens
```

ให้ PORTC, D, B เป็น input

เริ่มต้นให้ count = 20 และเริ่มนับถอยหลัง โดยให้
 *port c เป็นหลักหน่วย (count mod 10)
 *port d เป็นหลักสิบ (count ทหาร 10)

**TCCR1A= 0x00เป็นการบอกว่า เป็น
 timer ขนาด 16-bit แบบ normal
 **TCCR1B = 0x04 ใช้ prescaler = 256

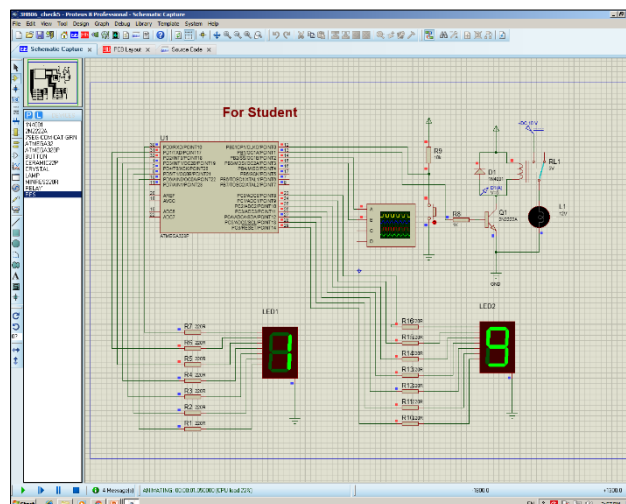
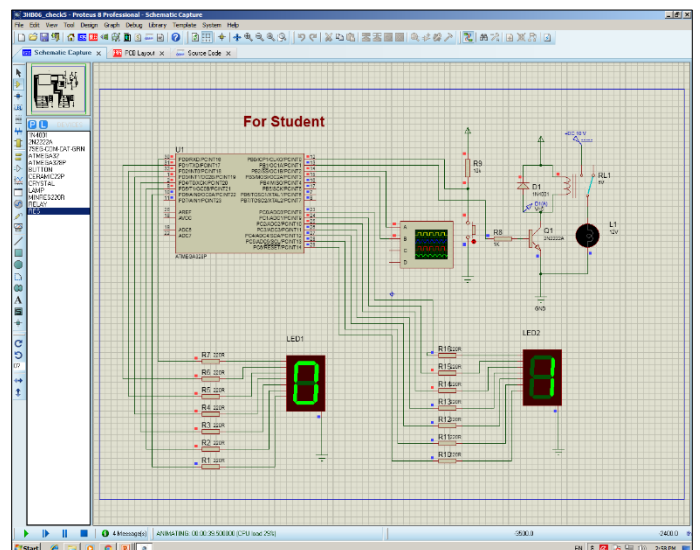
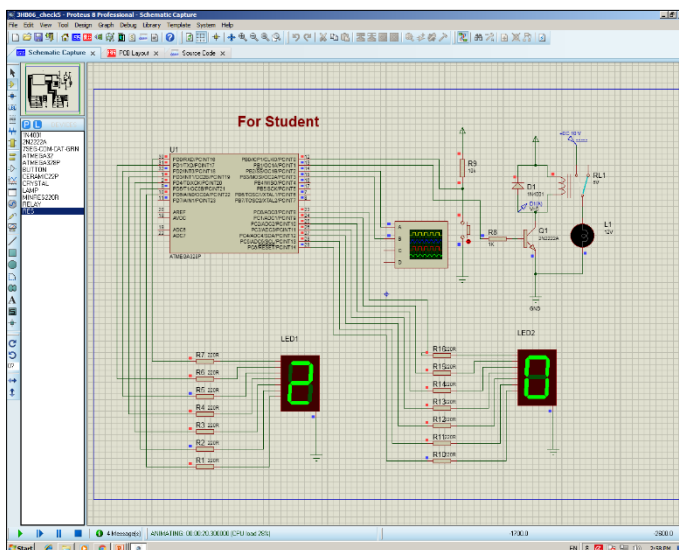

```

38     PORTC = LOOKUPTB[(20-i)%10]; // unit
39     PORTD = ~PORTD;
40     PORTD |= 0b00000010;
41     PORTD &= 0b00000100;
42 }
43 PORTB = 0b01011100; // o
44 PORTC = 0b01110001; // F
45 PORTD = 0b00000000; //Lamp off
46 count = 0;
47 }
48 }
49 }
50 }




```

ISR จะเคลียร์บริตที่ 0, 1 โดยการ OR กัน
และลดค่า count 1 จนมีค่า เท่ากับ 0 ก็จจะ
รีเซตให้เป็น20อีกครั้ง แสดงทาง 7-
segment

ผลลัพธ์ของวงจร ที่ได้ใน program proteus



▶ CHECKPOINT 5.4

- เมื่อเริ่มจ่ายไฟให้กับเครื่อง เครื่องใช้ไฟฟ้าจะดับอยู่ และแอลอีดีแสดงสถานะ 
- เมื่อกดสวิทช์ 1 ครั้งจะเป็นการกดเปิดเครื่องใช้ไฟฟ้าโดยมีแอลอีดี 7 เซกเมนต์ 2 ตัวแสดงสถานะของวงจรเป็น 
- เมื่อกดสวิทช์อีก 1 ครั้งจะเป็นการสั่งปิดเครื่องใช้ไฟฟ้า แต่เครื่องใช้ไฟฟ้าจะไม่ถูกตัดไฟในทันที แต่จะมีการหน่วงเวลาออกไป 20 วินาที พร้อมทั้งแสดงการนับลงที่แอลอีดีทั้งสอง เมื่อครบ 20 วินาทีจึงดับเครื่องใช้ไฟฟ้าแล้วแสดงสถานะที่แอลอีดีเป็น 

```

1  #include <inttypes.h>
2  #include <avr/io.h>
3  #include <avr/interrupt.h>
4  #include <avr/sleep.h>
5  #define VALUE 34286 //กำหนดความเร็วในการนับถอยหลัง
6  #define COUNT 20 //กำหนดค่า count
7  unsigned char count = COUNT, mode = 0,
8  led0 = 0b01011100, //O
9  ledf = 0b01110001, //F
10 ledn = 0b01010100, //N
11 input ,
12 status = 0x01,
13 flag = 0;
14 unsigned char LOOKUPTB[] = {
15     0b00111111,
16     0b00000110,
17     0b01011011,
18     0b01001111,
19     0b01100110,
20     0b01101101,
21     0b01111101,
22     0b00000111,
23     0b01111111,
24     0b01101111,
25     0b01110111,
26     0b01111100,
27     0b00111001,
28     0b01011110,
29     0b01111001,
30     0b01110001 };
31 int main()
32 {
33     DDRD = 0xFF;
34     DDRC = 0xFF;
35     DDRB = 0xFE;
36
37
38     PORTB = PORTB & 0xF3; //1111 0011
39
40     PORTC = 0b00000000; //led1
41     PORTD = 0b00000000; //led0

```

ในข้อนี้จะนำ โค้ดข้อ 5.1 มาผสมกับ 5.3 รวมกัน ที่ต่างกันคือ เริ่มต้นให้ 7-seg แสดงผลเป็น of เมื่อกด button Register ก็ทำงาน (TIMER1_OVF_vect) **เมื่อ 7-seg แสดง คำว่า on ค่า count ก็จะเริ่มนับที่ 20 และลดค่าลงเรื่อยๆ **เมื่อค่า count = 0 7-seg ก็จะแสดงคำว่า of หากกด button อีก ก็จะนับอีกครั้ง (วนลูป)

```

42
43     TCCR1A = 0x00;
44     TCCR1B = 0x04;
45     TIMSK1 = 0x00;
46
47     cli();
48     TCNT1 = VALUE;
49     sei();
50     // 0x00 for disable time / 0x01 for enable time
51
52
53     while (1){
54         input = PINB & 0b00000001; //pin B รับ Input เข้ามา
55         if(mode==0 && !input && flag == 0 && TIMSK1 == 0x00){
56             if(status == 0x00){
57                 PORTC = 0b01010100; //N
58                 PORTD = 0b01011100; //O
59                 PORTB = 0b00000010; //oscilloscope
60             }else {
61                 TIMSK1= 0x01;
62                 flag = 1;
63             }
64         }
65
66         if(flag == 1 && count == COUNT){
67             flag =0;
68         }
69
70         if(input == 0 && mode == 1){
71             status ^= 0x01;
72             mode = 0;
73
74         }else if(input == 1 && mode == 0){ //กำหนดนับค่าดยหลัง
75             mode = 1;
76         }
77     }
78     return 0;
79 }
80 ISR(TIMER1_OVF_vect)
81 {
82     TCNT1 = VALUE;
83     if(count == 0 && flag == 0) //ถ้าเซคเท่ากับ 0 ให้ขึ้น OF
84     {
85         count = COUNT;
86         PORTC = 0b01110001; //F
87         PORTD = 0b01011100; //O
88         PORTB = 0b00000010; //oscilloscope
89         TIMSK1 = 0x00;
90     }else{ //ถ้าไม่ใช่ให้แสดงค่าจากตัวเลข และ count --
91         PORTB = PORTB ^ 0x0C;
92         PORTC = LOOKUPTB[count%10]; //หลักหน่วย
93         PORTD = LOOKUPTB[count/10]; //หลักสิบ
94         count--; //สั่ง count --
95     }
96 }

```

ผลลัพธ์ของวงจร ที่ได้ใน program proteus

