# Assignment 7

**ชื่อกลุ่ม :** 9 A.M.

สมาชิกกลุ่ม : 64010761 นายวรพล รังษี

64010757 นายวรโชติ ใจเร็ว

**แนวคิดการออกแบบ :** นอกจากนาฬิกาจะมีหน้าที่หลักในการบอกเวลาให้เราทราบ เรายังทำการออกแบบให้นาฬิกานี้สามารถใช้ในการจับเวลา ตั้งเวลาปลุกและยังสามารถเปลี่ยนโหมดของธีมนาฬิกาจาก Dark mode เป็น Light mode ได้ เมื่อไม่มีแสงสว่าง นาฬิกานี้จะเปลี่ยนเป็น Light Mode เพื่อให้เราสามารถมองเห็นได้ชัดขึ้นอีกด้วย

**การใช้งานโดยย่อ :** การใช้งานโดยหลักจะเป็นการใช้ดูเวลา และฟังก์ชันเสริมจะประกอบไปด้วยการใช้จับเวลา, การตั้งเวลาปลุก และการเปลี่ยนธีม โดยรายละเอียดของปุ่มต่างๆ มีดังนี้

**-ปุ่ม ok**

-กดเพื่อบันทึกเวลาลงหน่วยความจำ EEPROM

-กดหลังจากเลือกตั้งเวลาในปุ่ม alarm ตามที่ต้องการได้แล้ว

-กดหลังจากมีเสียงแจ้งเตือนจากการปลุกเพื่อปิดเสียง

**-ปุ่ม countTime**

-กดเพื่อจับเวลา จะแสดง เวลาเป็น นาที ต่อ วินาที

-กดอีกรอบเพื่อยกเลิกการจับเวลา

**-ปุ่ม alarm**

-กดเพื่อเข้าสู่โหมดตั้งปลุก แสดงเวลา เป็น ชั่วโมง ต่อ นาที

-ตั้งเวลา โดยกดปุ่ม minute หรือ hour

**-ปุ่ม minute**

-กดเพื่อตั้งเวลา นาที หลังจากกด ปุ่ม alarm

**-ปุ่ม hour**

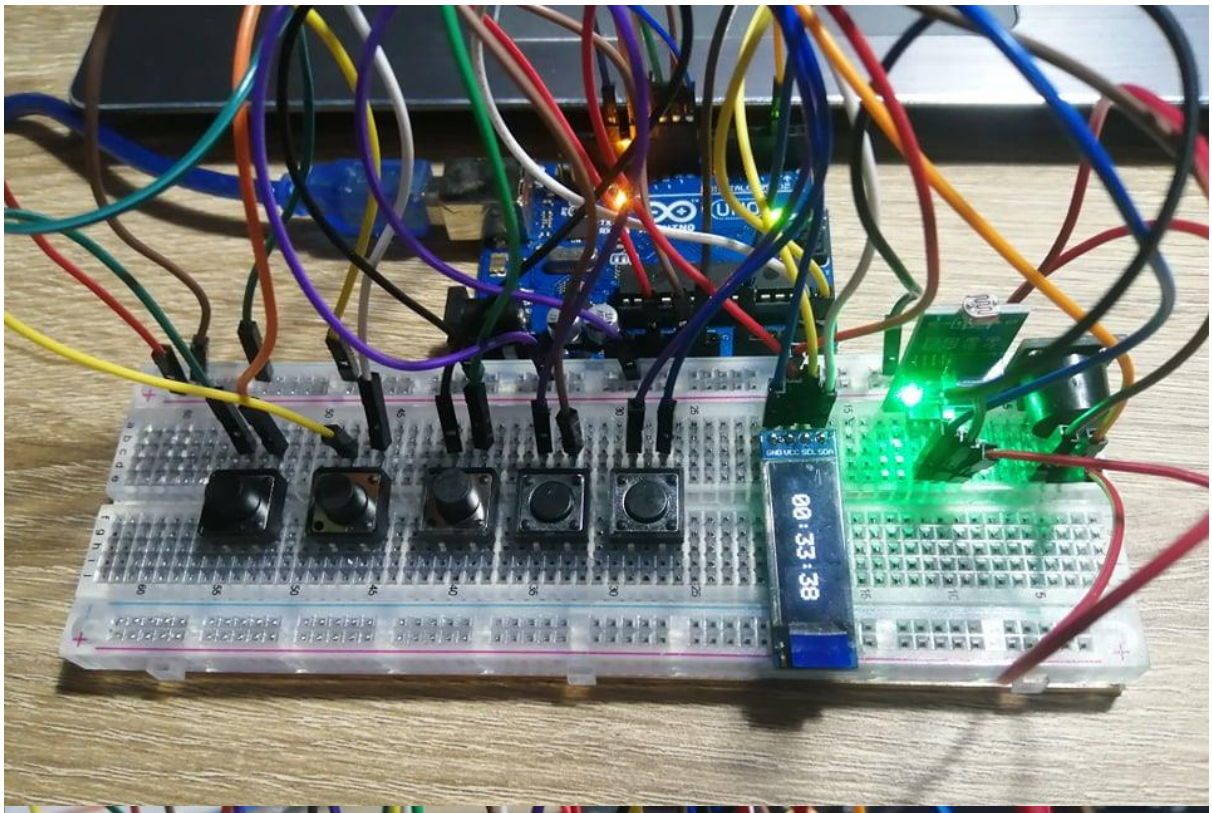
-กดเพื่อตั้งเวลา ชั่วโมง หลังจากกด ปุ่ม alarm

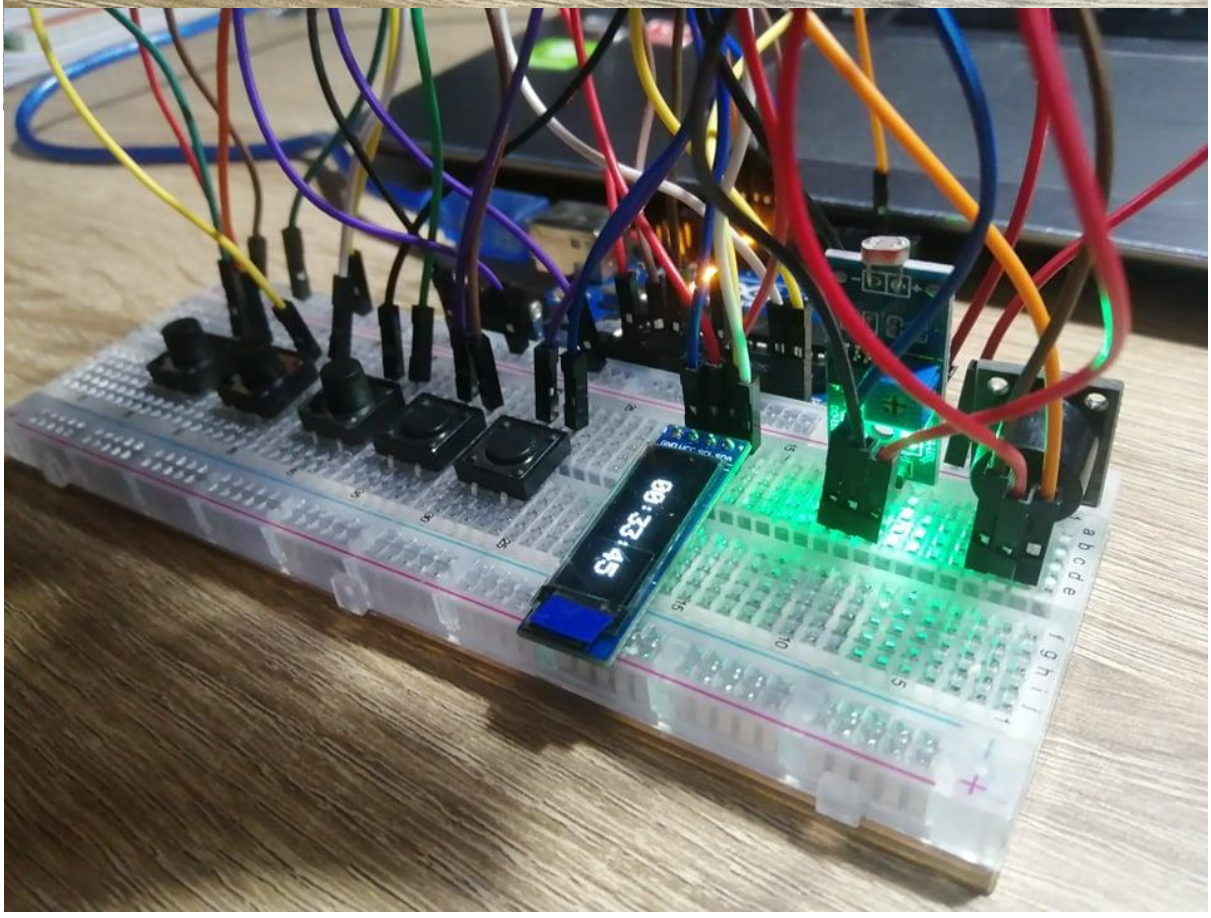
**รายละเอียดโปรแกรมโดยย่อ :** แบ่งออกเป็น 3 ส่วนหลักๆ ประกอบด้วย ส่วนของ setup/ตัวแปร, ส่วนของ loop การทำงาน และ ส่วนของ function การทำงานหลัก

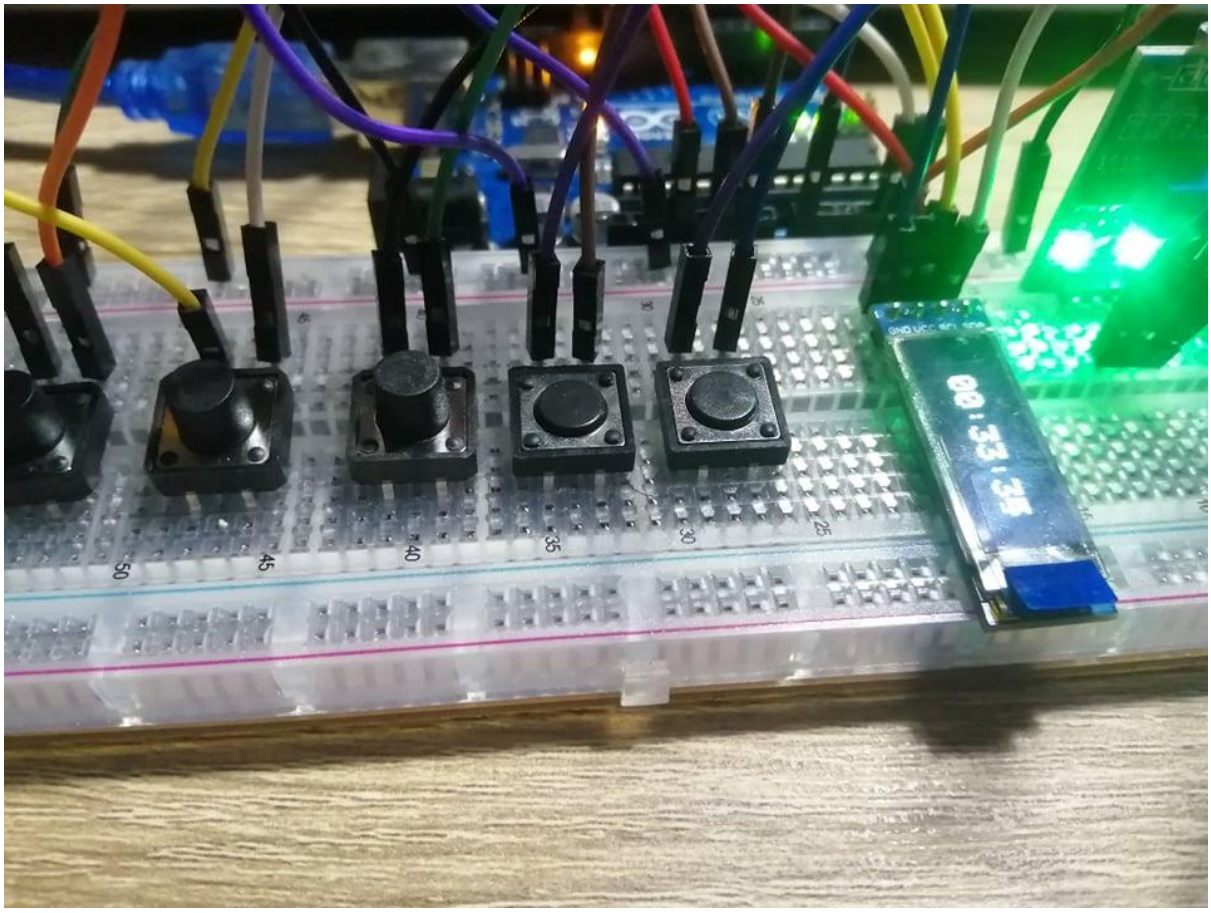
1. ส่วนของ Setup และ ตัวแปร : จะทำการตั้งค่าการแสดงผลของ OLED, ตั้งค่า pinMode ปุ่มต่างๆ, ตั้งค่า pinMode ของลำโพงสำหรับปลุกและ ตั้งค่าตัวแปร h(hour), m(minute), s(second) ให้อ่านค่าที่เก็บไว้ด้วยคำสั่ง EEPROM.read()
2. ส่วนของ loop : เป็นส่วนที่ทำงานตลอดเวลา มีหน้าที่นำ function และการทำงานต่างๆ มาทำงานด้วยกันกลายเป็นระบบการทำงานต่างๆของนาฬิกา ซึ่งจะประกอบไปด้วย สวนที่ใช้เก็บ Input จากปุ่มต่างๆเมื่อถูกกด ส่วนที่เก็บค่าจาก LDR เพื่อตัดสินว่าควรอยู่ในธีม Dark mode หรือ Light mode ส่วนที่ใช้สำหรับตั้งนาฬิกาปลุกโดยปุ่มที่เราใช้ตั้งปลุกจะประกอบปุ่มเพิ่มชม. และเพิ่มนาที เมื่อชม.เพิ่มถึง 24 จะกลับมาเป็น 0 อีกครั้ง ปุ่มนาที่ก็เช่นกันแต่สามารถเพิ่มได้ถึง 60 จึงค่อยกลับมาเป็น 0 ส่วนที่คอยนับเวลาและจับเวลาคอยดูว่าถึงเวลาที่จะส่งเสียนปลุกหรือยัง และสุดท้าย ส่วนที่รับ Input จากปุ่ม OK เพื่อ Save เวลาปลุก
3. ส่วนของ function :

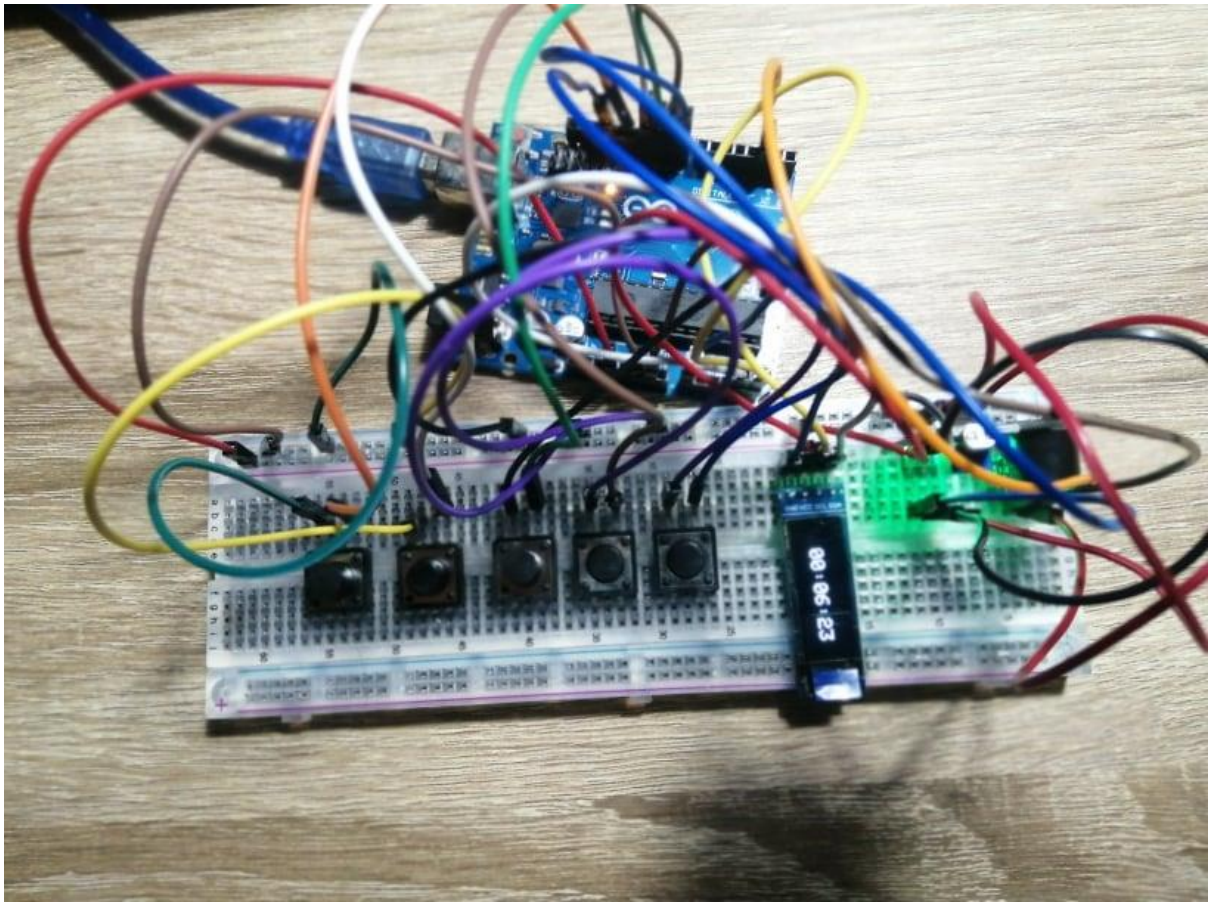
| **ชื่อ function** | **การทำงาน** |
|---|---|
| ok_saveTime() | Save เวลาปลุกที่เราตั้งไว้ลงใน EEPROM เมื่อเรากดปุ่ม ok |
| display_dark_mode() | เปลี่ยนเป็น Dark mode |
| display_light_mode() | เปลี่ยนเป็น Light mode |
| display_normal() | เข้าสู่โหมดปกติ แสดงเวลาปกติ |
| display_countTime() | เข้าสู่โหมดจับเวลา |
| display_alarm() | เข้าสู่โหมดตั้งนาฬิกาปลุก |
| sound_alarm() | เล่นเสียงเมื่อถึงเวลาปลุก |
| sound_add_time() | เล่นเสียงเมื่อปุ่มถูกกด |
| timePass() | เพิ่มค่าตัวแปร hour ขึ้นมา 1 เมื่อ minute เท่ากับ 60 เพิ่มค่าตัวแปร minute ขึ้นมา 1 เมื่อ second เท่ากับ 60 และเมื่อถึงเวลา 23 : 59 : 59 เวลาก็จะกลับไปเป็น 00 : 00 : 00 อีกครั้ง |

**รูปถ่ายชิ้นงาน :**







Code : https://github.com/worachote1/itc/blob/main/ITC_Assignment_07.ino

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 32
#define OLED_RESET -1
Adafruit_SSD1306 OLED(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);

#include <EEPROM.h>
#include <TimerOne.h>

//for sound
#define buzzer 2

#define debounce 50

#define ok_button 12

#define countTime_button 11
#define alarm_button 10

#define add_minute_button 9
#define add_hour_button 8

//for working with Dark Mode , Light Mode using LDR module
#define ldr A0


int h = 1 ;
int m  = 0;
int s = 4;
void setup() {
  OLED.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  OLED.clearDisplay();
  //drawArea();
  Serial.begin(9600);

  //No matter which part of the program's running , if reach to 1 second
excute timePass().
  Timer1.initialize(1000000);
  Timer1.attachInterrupt(timePass);


  //
  pinMode(ok_button, INPUT_PULLUP);
```

```
  pinMode(countTime_button, INPUT_PULLUP);
  pinMode(alarm_button, INPUT_PULLUP);
  pinMode(add_minute_button, INPUT_PULLUP);
  pinMode(add_hour_button, INPUT_PULLUP);

  //sound
  pinMode(buzzer,OUTPUT);

  h = EEPROM.read(0);
  m = EEPROM.read(sizeof(h));
  s = EEPROM.read(sizeof(m) * 2);


  OLED.display();
}

int ok, countTime, alarm, minute, hour;//button variable
int lastPress = 0;

int alarm_h = 0     , alarm_m = 0;
int countTime_m = 0 , countTime_s = 0;

String alarm_state = "OFF" , countTime_state = "OFF";
String sound_state = "OFF";

//resistor value inverse with lux
//more light less R , less light more R
int resistor_value ;

void loop() {
  OLED.clearDisplay();

  // button variable
  ok = digitalRead(ok_button);
  countTime = digitalRead(countTime_button);
  alarm = digitalRead(alarm_button);
  minute = digitalRead(add_minute_button);
  hour = digitalRead(add_hour_button);

  // LDR variable
  resistor_value = analogRead(ldr);
  Serial.println(resistor_value);

  //select Dark Mode or Light Mode depend on lux_value
  if(resistor_value >= 440)
  {
    display_light_mode();
  }
  else
```

```
{
  display_dark_mode();
}

if (millis() - lastPress >= debounce )
{
  lastPress = millis();
  if (hour == 0)
  {
    Serial.println("hour");
    if (alarm_state == "ON")
    {
      alarm_h += 1 ;
      if(alarm_h > 23)
      {
        alarm_h = 0;
      }

      sound_add_time();
    }
  }
  else if (minute == 0)
  {
    Serial.println("minute");
    if (alarm_state == "ON")
    {
      alarm_m += 1 ;
      if(alarm_m >= 60)
      {
        alarm_h += 1;
        alarm_m = 0;
      }

      sound_add_time();
    }
  }

  else if (countTime == 0)
  {
    if (countTime_state == "OFF")
    {
      Serial.println("CountTime now Turn ON ");
      countTime_state = "ON";
    }
    else  // countTime_state == "ON"
    {
      Serial.println("CountTime now Turn OFF ");
      countTime_m = 0 ;
```

```
        countTime_s = 0;
        countTime_state = "OFF";
      }
    }
    else if (alarm == 0)
    {
      if (alarm_state == "OFF")
      {
        Serial.println("Alarm now Turn ON ");
        alarm_state = "ON";
      }
//      else // alarm_state == "ON"
//      {
//        Serial.println("alarm now Turn OFF ");
//      }
    }

    else if (ok == 0)
    {
      ok_saveTime();
      Serial.println("ok");
      Serial.println("Alarm now Turn OFF ");
      alarm_state = "OFF";
      sound_state = "OFF";
    }

  }

  // Display Section

  // display normal mode ,if not press countTime_button or alarm_button
  if (alarm_state == "OFF" && countTime_state == "OFF")
  {
    display_normal();
  }

  //display countTime mode , if countTime_button has been pressed
  else if (countTime_state == "ON")
  {
    Serial.println("run display_countTime");
    display_countTime();
  }

  //display alarm mode , if alarm_button has been pressed
  else if (alarm_state == "ON")
  {
    Serial.println("run display_alarm ");
    display_alarm();
```

```
  }

  //check if sound should be played ?
  if(sound_state == "ON")
  {
    sound_alarm();                                    //----------------------
-------------
  }

//  //save to EEPROM
//  EEPROM.put(0, h);
//  EEPROM.get(0, h);
//
//  EEPROM.put(sizeof(h), m);
//  EEPROM.get(sizeof(h), m);
//
//  EEPROM.put(sizeof(m) * 2, s);
//  EEPROM.get(sizeof(m) * 2, s);


  OLED.display();
}

//save time , if ok button pressed
void ok_saveTime()
{
  //save to EEPROM
  EEPROM.put(0, h);
  EEPROM.get(0, h);

  EEPROM.put(sizeof(h), m);
  EEPROM.get(sizeof(h), m);

  EEPROM.put(sizeof(m) * 2, s);
  EEPROM.get(sizeof(m) * 2, s);
}

//Dark Mode and Light Mode
void display_dark_mode()
{
  OLED.setTextColor(WHITE);
  OLED.setTextSize(2);
}
void display_light_mode()
{
  OLED.fillScreen(WHITE);
  OLED.setTextColor(BLACK);
  OLED.setTextSize(2);
```

```
}

void display_normal()  //normal mode function
{
  //Normal Mode display section
  OLED.setCursor(24, 14);
//  OLED.setTextColor(WHITE);
//  OLED.setTextSize(2);
  if (h < 10)
  {
    OLED.print("0");
  }
  OLED.print(h);
  OLED.print(":");

  if (m < 10)
  {
    OLED.print("0");
  }

  OLED.print(m);
  OLED.print(":");

  if (s < 10)
  {
    OLED.print("0");
  }
  OLED.print(s);
}

int countTime_pass = 0;
void display_countTime() // countTime mode function
{
  //CountTime Mode display section
  OLED.setCursor(28, 14);
//  OLED.setTextColor(WHITE);
//  OLED.setTextSize(2);

  if (countTime_m < 10)
  {
    OLED.print("0");
  }

  OLED.print(countTime_m);
  OLED.print(":");

  if (countTime_s < 10)
  {
```

```
    OLED.print("0");
  }
  if (countTime_s >= 60)
  {
    countTime_m += 1;
    countTime_s = 0;
  }
  OLED.print(countTime_s);

  // countTime_s += 1; put this line of code in timePass instead , because of
Timer1.attachInterrupt(timePass);
}

void display_alarm()      // Alarm mode function
{
  //Alarm Mode display section
  OLED.setCursor(28, 14);
//  OLED.setTextColor(WHITE);
//  OLED.setTextSize(2);
  if (alarm_h < 10)
  {
    OLED.print("0");
  }

  OLED.print(alarm_h);
  OLED.print(":");

  if (alarm_m < 10)
  {
    OLED.print("0");
  }

  OLED.print(alarm_m);
}

//play sound function

// play sound when alarm end (will be use when sound_state == "ON")
void sound_alarm(){
  tone(buzzer, 500, 50);
  Serial.println("play sound alarm");
 }
//play sound when minute or hour button pressed
void sound_add_time()
{
  tone( buzzer, 250, 50);
  Serial.println("play sound add time");
}
```

```
void timePass()
{
  //OLED.clearDisplay();
  s += 1;

  if (h == 24)
  {
    h = 0;
  }
  if (m >= 60)
  {
    h++;
    m = 0;
  }
  if (s >= 60)
  {
    m++;
    s = 0;
  }

  if (countTime_state == "ON") //add each 1 second in CountTime Mode
  {
    countTime_s += 1;
  }

  if ( h == alarm_h && m == alarm_m ) // wake up , get to work !!!
  {
    alarm_h = 0;
    alarm_m = 0;
    sound_state = "ON";
    Serial.println("Alarm sucess !!!");
  }
}
```