



01076001 INTRODUCTION TO COMPUTER ENGINEERING

OF

FACULTY OF ENGINEERING, COMPUTER ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

ASSIGNMENT 7

-MINI CLOCK-

ARDUINO | EEPROM

GROUP 13

65010179 CHANATHIP YAIYIAM

65010195 CHOLLASAK ANUWAREEPONG

1st YEAR | SEMESTER 1 | 2022

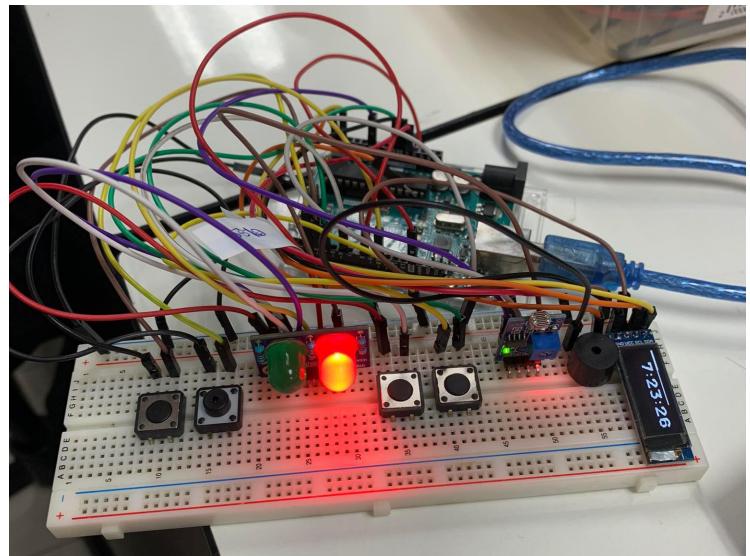
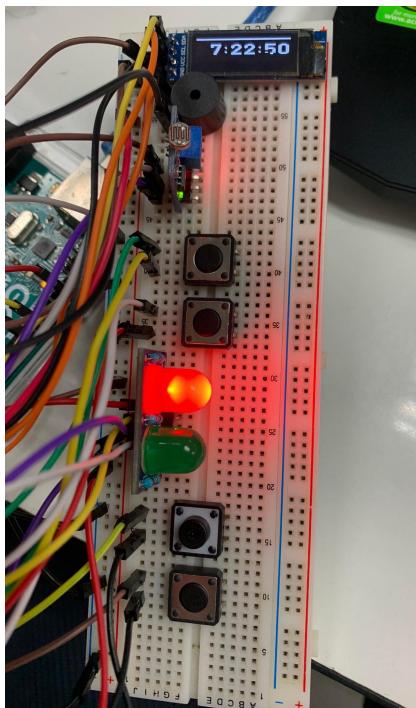
INTRO AND DESIGN (ที่มา)

เนื่องจากปัจจุบัน นาฬิกาส่วนใหญ่นั้นจะมีฟังก์ชั่นการทำงานที่ค่อนข้างคล้ายคลึงกัน ทางผู้พัฒนาจึงต้องการที่จะพัฒนาพิพากที่สามารถบอกว่า ช่วงเช้า หรือ ช่วงเย็น และสามารถบอกผู้ใช้ในบ้านได้ว่า ด้านนอกนั้น มีแดดที่เข้า หรือ ห้องไฟคัลลิ่ม เพื่อบ่งบอกให้ผู้ใช้นั้น ตัดสินใจก่อนการออกจากที่พักอาศัยได้อย่างดี

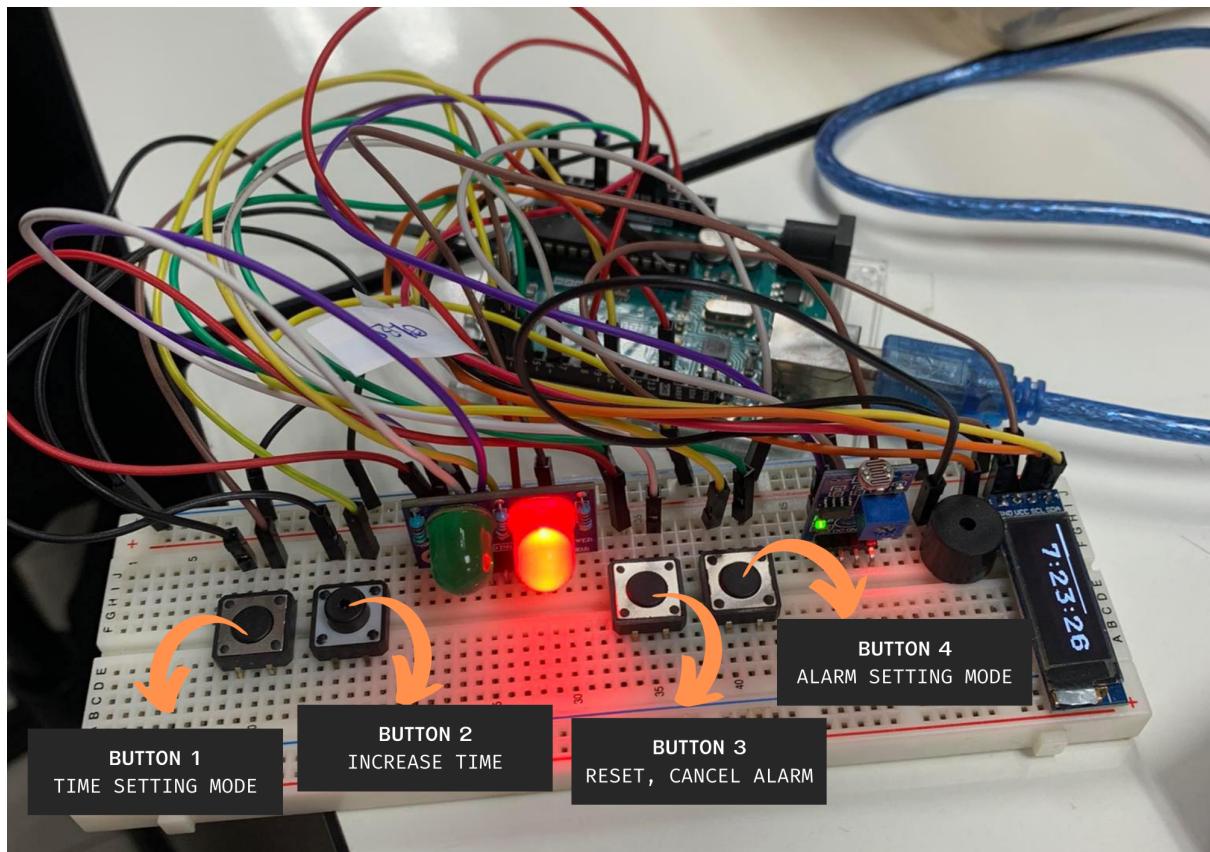
FUNCTIONS (ฟังก์ชั่นการทำงานพื้นฐาน)

- ตั้งเวลา รวมถึง แสดงผลเวลา เป็น วินาที นาที และ ชั่วโมง
- ตั้งเวลาแจ้งเตือน ได้โดยผู้ใช้ต้องกรอกข้อมูลเวลาลง Serial Monitor
- การแจ้งเตือน จะเป็นการใช้เสียงที่ทุกคนคุ้นเคยของการปลุก
- ผู้ใช้สามารถยกเลิกการตั้งปลุก และการแจ้งเตือน ได้
- เวลาจะแสดงบนหน้าจอ OLED 0.96
- จะแสดงผล LED สีแดง เมื่อค้านนอกนั้น มีแดด หรือ แสงไฟที่สูง
- จะแสดงผล LED สีเขียว เมื่อค้านนอกนั้น มีแดดร้อน หรือ แสงไฟที่ต่ำถึงปานกลาง
- มีระบบบันทึกเวลาอัตโนมัติถ้าหากมีการขัดข้องของกระแสไฟฟ้าหรือถ้าหากมีการตัดไฟฟ้า
- เมื่อเปิดหลังจากการปิดหรือถ้าหากมีการขัดข้อง จะเริ่มนับใหม่เวลาล่าสุดการที่จะดับลง

PICTURE OF DEVICE



INSTRUCTION MANUAL (คู่มือการใช้งาน)



BUTTON 1 : เมื่อกด หน้าจอ OLED จะทำการแสดง สามเหลี่ยมด้านบนตัวเลข เพื่อให้ผู้ใช้ทราบได้ว่า จะตั้งชั่วโมง หรือ นาที การกดอีกครั้งจะเป็นการเลือกหลักไปจนหมดตำแหน่งของตัวเลข

BUTTON 2: ก่อนกด ต้องกดปุ่มที่ 1 เพื่อให้เข้าสู่โหมดการตั้งเวลา และ เมื่อทำการกด จะเป็นการเริ่มเวลา สามารถใช้ร่วมกับปุ่มที่ 1 ในการเปลี่ยนหลักเวลาที่จะทำการตั้งค่า

BUTTON 3: เมื่อกด จะทำการรีเซ็ตค่าที่บันทึกลงในด้าวของนาฬิกา ควรใช้เฉพาะในการยกเลิกการแจ้งเตือนเท่านั้น

BUTTON 4: เมื่อกด จะเข้าสู่โหมดตั้งเวลาแจ้งเตือน ผู้ใช้สามารถกรอกเวลาที่ต้องการผ่าน SERIAL MONITOR หาก SERIAL MONITOR จะแสดงผลให้ผู้ใช้กรอกตัวเลขลงไป ดังภาพด้านล่าง

กรอกชั่วโมง กรอก 2 ลงใน Serial

```
Output  Serial Monitor X

Message (Enter to send message to 'Arduino Uno' on 'COM5')

17:08:46.946 -> Please enter hours:
```

กรอกนาที กรอก 2 ลงใน Serial

```
Output  Serial Monitor X

Message (Enter to send message to 'Arduino Uno' on 'COM5')

17:08:46.946 -> Please enter hours:
17:10:28.011 -> You pressed 2
17:10:28.011 -> Please enter minutes:
```

การแสดงผลเมื่อกรอก นาทีเสร็จ

```
Output  Serial Monitor X

Message (Enter to send message to 'Arduino Uno' on 'COM5')

17:08:46.946 -> Please enter hours:
17:10:28.011 -> You pressed 2
17:10:28.011 -> Please enter minutes:
17:11:51.558 -> You pressed 2
17:11:51.558 -> Alarm is set at 02:02
```

ALARM SYSTEM

เมื่อถึงเวลาที่ผู้ใช้นั้นได้ตั้งค่าไว้ ลำโพง หรือ Buzzer จะทำการดังขึ้น โดยเสียงการดังนั้น จะเป็นเสียงพื้นฐานในการแจ้งเตือนต่างๆ เพื่อให้ผู้ใช้นั้นตื่นตัวหรือรับรู้ได้อย่างทันที

HOW TO STOP ALARM SYSTEM

ถ้าหากผู้ใช้ต้องการหยุดการแจ้งเตือนหรือต้องการรีเซ็ตเวลาแจ้งเตือนที่ตั้งใหม่ ต้องกดปุ่ม 3 จะทำการหยุดการแจ้งเตือน และรีเซ็ตเวลาแจ้งเตือนที่ตั้งไว้ในนาฬิกา

OUTDOOR SUNNY CHECKING

เมื่อค้านอกนั้นมีแสงแวดล้อมที่สูง LED สีแดงจะทำงาน และ เมื่อค้านอกมีแสงแวดล้อมที่ต่ำหรือเหมาะสมต่อการออกจากที่พักอาศัย LED สีเขียวจะทำงาน

VIDEO OF THE SYSTEM



<https://www.youtube.com/watch?v=MGbv28kQ0KU&t=2s>

SOURCE CODE

```
#include <Arduino.h>
#include <U8g2lib.h>
#include <EEPROM.h>

#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

#ifndef U8X8_HAVE_HW_SPI
#include <SPI.h>
#endif

#ifndef U8X8_HAVE_HW_I2C
#include <Wire.h>
#endif

// Variables to store the time
int hours ;
int minutes ;
int seconds ;

int analogPin = A0;
int val = 0;

int ledred = 8;
int ledgreen = 10;

// Constants for the button pins
int countMinutes = 0;
int countHours = 0;

int timer_round = 0;

int buzzer = 12;

int alarmstate = 0;

int eepromshowminutes ;
int eepromshowhours ;

int hoursfromeprom ;
```

```
int minutesfromeprom ;  
  
const int PIN_BUTTON_MODE = 3;  
const int PIN_BUTTON_SET = 2;  
const int PIN_BUTTON_ALARMSET = 4;  
  
const int PIN_BUTTON_ALARMSTOP = 6;  
  
const int BUTTON_MODE_DEBOUNCE_TIME = 250;  
const int BUTTON_SET_DEBOUNCE_TIME = 100;  
  
const int MODE_SHOW_TIME = 0;  
const int MODE_SET_SECONDS = 3;  
const int MODE_SET_MINUTES = 2;  
const int MODE_SET_HOURS = 1;  
  
int timeformuser;  
  
// Variables for the button state  
// We are using the internal pull-up resistors via INPUT_PULLUP, so  
// press is LOW and not pressed is HIGH  
unsigned long elapsedButtonModeMillis = 0;  
unsigned long previousButtonModeMillis = 0;  
  
unsigned long elapsedButtonSetMillis1 = 0;  
unsigned long previousButtonModeMillis1 = 0;  
  
unsigned long elapsedButtonSetMillis = 0;  
unsigned long previousButtonSetMillis = 0;  
  
// Char array for the time being showed on the display  
char timeString[9];  
  
// Variables to store the time  
unsigned long currentMillis = 0;  
  
// Int is enough to store the elapsed time  
int elapsedTimeUpdateMillis = 0;  
unsigned long previousTimeUpdateMillis = 0;  
  
float percentageOfSecondElapsed = 0;
```

```
byte currentMode = MODE_SHOW_TIME;

int alarm_hours_set = -1;
int alarm_minutes_set = -1;

unsigned long previousMillis = 0;
const long interval = 500;
int buzzerstate = LOW;

bool checkbuzzer;

int showhours;
int showminutes;

// A complete list of all displays is available at:
// https://github.com/olikraus/u8g2/wiki/u8g2setupcpp
U8G2_SSD1306_128X64_NONAME_1_HW_I2C u8g2(U8G2_R0, /* 
reset=*/U8X8_PIN_NONE);

void setup(void) {

    Serial.begin(9600);
    u8g2.setFont(u8g2_font_logisoso28_tf);
    u8g2.begin();

    // Configure the pins of the buttons with the internal PULLUP
    resistor
    pinMode(PIN_BUTTON_MODE, INPUT_PULLUP);
    pinMode(PIN_BUTTON_SET, INPUT_PULLUP);
    pinMode(PIN_BUTTON_ALARMSET, INPUT_PULLUP);
    pinMode(PIN_BUTTON_ALARMSTOP, INPUT_PULLUP);
    pinMode(buzzer, OUTPUT);
    pinMode(ledred, OUTPUT);
    pinMode(ledgreen, OUTPUT);
    pinMode(analogPin, INPUT);

    // EEPROM.write(0, 0);

    // EEPROM.write(2, 0);

    // EEPROM.write(4, 0);
    // EEPROM.write(6, 0);
```

```

// EEPROM.write(8, 0);

hours = EEPROM.read(0);
minutes = EEPROM.read(2);

checkbuzzer = EEPROM.read(4);

}

void loop(void) {

int sensorValue = analogRead(A0);
val = analogRead(analogPin); //อ่านค่าเซ็นเซอร์


if(val < 500){
    digitalWrite(ledgreen, HIGH); //แสดงสีเขียว
}else{
    digitalWrite(ledgreen, LOW);
}

if(val > 500){
    digitalWrite(ledred, HIGH); //แสดงสีแดง
}else{
    digitalWrite(ledred, LOW);
}

// millis() itself takes 1.812 micro seconds that is 0.001812 milli
seconds
//
https://arduino.stackexchange.com/questions/113/is-it-possible-to-find-the-time-taken-by-millis
currentMillis = millis();
if (digitalRead(PIN_BUTTON_ALARMSTOP) == LOW) {
    stopalarm();
    alarm_hours_set = -1;
    alarm_minutes_set = -1; //เมื่อกดปุ่มที่ 3 ให้เข้าฟังกชันหยุดการแจ้งเตือนทำงาน
}

```

```

if  (digitalRead(PIN_BUTTON_MODE)  ==  LOW)  {

    buttonModeHandler(); //กดปุ่มที่ 1 ให้ฟังก์ชันปรับหลักทำงาน
}

if  (digitalRead(PIN_BUTTON_SET)  ==  LOW)  {

    buttonSetHandler(); //กดปุ่มที่ 2 ให้ฟังก์ชันปรับเลขทำงาน
}

if  (digitalRead(PIN_BUTTON_ALARMSET)  ==  LOW)  {
    alarmSetHoursAndMinutes(); //กดปุ่มที่4 ให้เข้าสู่โหมด ตั้งเวลา
}

}

alert();
drawTime();
checkTime();

if  (currentMode == MODE_SHOW_TIME)  {
    increaseSeconds(); // ถ้ากดปุ่ม ให้เพิ่มตัวเลข
} else {
    previousTimeUpdateMillis = currentMillis;
}

drawScreen(); //โชว์เลข
}

void checkTime() {
// Check if a minutes has been elapsed
if  (seconds > 59) {
    seconds = 0;
    minutes++;
    EEPROM.update(2, minutes);
    //ถ้าวินาทีมากกว่า 59 ให้ นาทีบวกขึ้น และรีเซ็ตวินาที
}
}

```

```

// Check if an hour has been elapsed
if (minutes > 59) {
    minutes = 0;
    EEPROM.update(2, minutes);
    hours++;
    EEPROM.update(0, hours);
//ถ้านาทีมากกว่า 59 ให้ ข้ามไปขั้น และรีเซ็ตนาที
}

// Check if a day has been elapsed
if (hours > 23) {
    hours = 0;
    EEPROM.update(0, hours);
//ถ้าชั่วโมงมากกว่า 59 ให้ รีเซ็ตชั่วโมง
}
}

void buttonModeHandler() {
    elapsedButtonModeMillis = currentMillis - previousButtonModeMillis;
    if (elapsedButtonModeMillis > BUTTON_MODE_DEBOUNCE_TIME) {
        //Serial.println("Mode Handler");
        previousButtonModeMillis = currentMillis;
        currentMode++;

        if (currentMode > 3) {
            currentMode = 0;
        }
    }
} //โหลดเข้าสู่การตั้งหลัก

void buttonSetHandler() { //ฟังก์ชันการเพิ่มเวลา
    elapsedButtonSetMillis = currentMillis - previousButtonSetMillis;
    //Serial.println(elapsedButtonSetMillis);
    if (elapsedButtonSetMillis > BUTTON_SET_DEBOUNCE_TIME) {
        //Serial.println("Set Handler");
        previousButtonSetMillis = currentMillis;

        if (currentMode == MODE_SET_SECONDS) {
            seconds = 0; //ถ้ากด ให้ วินาทีเท่ากับ 0
        }
        if (currentMode == MODE_SET_MINUTES) {
            minutes++; //ถ้ากดเพิ่มวินาที
            EEPROM.update(2, minutes);
        }
    }
}

```

```

        }

        if (currentMode == MODE_SET_HOURS) {
            hours++;
            EEPROM.update(0, hours);

        }
    }

}

void increaseSeconds() { // ដំឡើងការលាតវេលា
    elapsedTimeUpdateMillis = currentMillis - previousTimeUpdateMillis;

    // Check if 1000ms, 1 second, has been elapsed
    if (elapsedTimeUpdateMillis > 1000) {
        seconds++;

        // It might be possible that more than 1000ms has been elapsed e.g.
        1200ms
        // Then there are already 200ms elapsed of the next second. We need
        to
        // subtract these on the "last time". So the next second will be
        updated 200ms earlier.
        // This reduces the amount of time drift.
        previousTimeUpdateMillis = currentMillis - (elapsedTimeUpdateMillis
        - 1000);
    }
}

void drawScreen() { // ដំឡើងនិងតាមលេខបន្ថែម
    u8g2.firstPage();

    do {

        if (currentMode != MODE_SHOW_TIME) {
            u8g2.drawTriangle((currentMode - 1) * 43 + 5, 0, currentMode * 43
            - 5, 0, (currentMode - 1) * 43 + 21, 5);
        }

        drawAnimation();
        drawTime();

    } while (u8g2.nextPage());
}

```

```

}

void drawTime() {

    // Found at https://forum.arduino.cc/index.php?topic=371117.0
    // sprintf_P uses the Program Memory instead of RAM, more info at
    http://gammon.com.au/progmem
    // Here we format the minutes and seconds with a leading zero: e.g.
    01, 02, 03 etc.
    sprintf_P(timeString, PSTR("%2d:%02d:%02d"), hours, minutes,
    seconds);

    // Draw the timeString
    u8g2.drawStr(0, 45, timeString); //ฟังก์ชันแสดงเวลา
}

void drawAnimation() { //ฟังก์ชันอนิเมชันขีดด้านบน
    // Calculate the percentage elapsed of a second
    percentageOfSecondElapsed = elapsedTimeUpdateMillis / 1000.0;

    if (currentMode == MODE_SHOW_TIME) {
        // Draw the yellow lines
        u8g2.drawBox(0, 0, 127 - (127 * percentageOfSecondElapsed), 2);
        u8g2.drawBox(0, 3, (127 * percentageOfSecondElapsed), 2);
    }
}

void alarmSetHoursAndMinutes() { //ฟังก์ชันตั้งเวลา
    Serial.flush();
    Serial.println("Please enter hours: ");

    while (alarm_hours_set == -1) {
        if (Serial.available() > 0) {
            alarm_hours_set = Serial.readString().toInt();
            if (alarm_hours_set <= 23 && alarm_hours_set >= 0) {
                Serial.print("You pressed ");
                Serial.println(alarm_hours_set);
                EEPROM.put(6, alarm_hours_set);
                EEPROM.get(6, hoursfromeprom);
                EEPROM.put(4, true);
            }
            //เข็ค่าว่าตัวเลขเข้ามากกกว่า 23 หรือไม่ ถ้าไม่ให้ทำงานต่อไป
        } else {
    }
}

```

```
        Serial.println("Error, Please press the button again.");
    }
}
}

Serial.println("Please enter minutes: ");

while (alarm_minutes_set == -1) {
    if (Serial.available() > 0) {
        alarm_minutes_set = Serial.readString().toInt();
        if (alarm_minutes_set <= 59 && alarm_minutes_set >= 0) {
            Serial.print("You pressed ");
            Serial.println(alarm_minutes_set);
            EEPROM.put(8, alarm_minutes_set);
            EEPROM.get(8, minutesfromeprom);
            Serial.print("Alarm is set at ");
            if(alarm_hours_set < 10 && alarm_hours_set >= 0) {
                Serial.print("0");
                Serial.print(alarm_hours_set);
                Serial.print(":");
            }else{
                Serial.print(alarm_hours_set);
                Serial.print(":");
            }

            if(alarm_minutes_set < 10 && alarm_minutes_set >= 0) {
                Serial.print("0");
                Serial.println(alarm_minutes_set);
            }else{
                Serial.print(alarm_minutes_set);
            }
        }
    }
}

} else {
```

```
        Serial.println("Error, Please press the button again.");
    }
}
}

alarm_hours_set = -1;
alarm_minutes_set = -1;
}

void alert() { //ฟังก์ชันแจ้งเตือน
EEPROM.get(6, hoursfromeprom);
EEPROM.get(8, minutesfromeprom);

EEPROM.get(4, checkbuzzer);

unsigned long currentMillis = millis();

if (hours == hoursfromeprom && minutes == minutesfromeprom &&
checkbuzzer) {
    if (timer_round < 100) {

        if (currentMillis - previousMillis >= interval) {
            // save the last time you blinked the LED
            previousMillis = currentMillis;

            // if the LED is off turn it on and vice-versa:
            if (buzzerstate == LOW) {
                buzzerstate = HIGH;

            } else {

                buzzerstate = LOW;
                timer_round++;

            }

            // set the LED with the ledState of the variable:
            digitalWrite(buzzer, buzzerstate);
        }
    }
}
```

```
    } else {
        timer_round = 0;
        digitalWrite(buzzer, LOW);

    }

}

void stopalarm() {
    alarm_hours_set = 3;
    alarm_minutes_set = -3;
    EEPROM.put(4, false);
    EEPROM.put(6, alarm_hours_set);
    EEPROM.put(8, alarm_minutes_set);
    EEPROM.get(6, hoursfromeprom);
    EEPROM.get(8, minutesfromeprom);
    Serial.println("Reset.....");
}
```