

LAB 5: Advanced Acid-Base Titration with Automate Flow system

Week 1 (ห้อง 1002/3)

Micropython

ไมโครไพทอน (MicroPython) เป็นคอมไพเลอร์ (Compiler) และรันไทม์ (Runtime) ภาษาไพทอนสำหรับทำงานบนไมโครคอนโทรลเลอร์เป็นภาษาที่ง่ายต่อการเรียนรู้ และมีไวยากรณ์ที่ทำความเข้าใจได้ นอกจากนี้ยังเป็นซอฟต์แวร์ประเภท Open Source การเลือกใช้ไมโครไพทอนจะช่วยให้การเรียนรู้เกี่ยวกับการเขียนโปรแกรมและใช้งานไมโครคอนโทรลเลอร์สำหรับผู้เริ่มต้นได้ดียิ่งขึ้นเมื่อเทียบกับภาษาระดับสูงอย่างเช่น ภาษา C/C++, Java เป็นต้น

Thonny

Thonny เป็น IDE (Integrated development environment) ประเภท Open Source ที่มีอินเทอร์เฟซที่ใช้งานง่าย มีฟีเจอร์ที่ช่วยในการเรียนรู้ภาษาไพทอนได้อย่างมีประสิทธิภาพ โดยคุณสมบัติของโปรแกรมนั้นครอบคลุมการเขียนโค้ด ตรวจสอบโค้ด บริหารจัดการไลบรารีของภาษาไพทอน และ รันโปรแกรมภาษาไพทอน ดังนั้นโปรแกรม Thonny จึงเป็นโปรแกรมสำหรับพัฒนาและเขียนโค้ดภาษาไพทอนที่เน้นความง่ายในการใช้งาน โดยเฉพาะสำหรับผู้เริ่มต้นที่มีประสบการณ์น้อยในการเขียนโปรแกรมไพทอน หรือ สำหรับการศึกษาและการเรียนรู้ภาษาไพทอนอย่างง่ายดาย

Virtual COM port ของคอมพิวเตอร์จะเชื่อมต่อกับ USB port ของบอร์ดไมโครคอนโทรลเลอร์ที่ติดตั้งเฟิร์มแวร์สำหรับไมโครไพทอนไว้แล้ว โดยจะเป็นการใช้งานร่วมกับ REPL (Read-Eval-Print Loop) เพื่อให้สามารถเขียนคำสั่งเพื่อรันหรืออัปโหลดโค้ด .py และบันทึกไฟล์ลงใน Flash Storage ของบอร์ดไมโครคอนโทรลเลอร์ผ่านโปรแกรม Thonny เพื่อให้ได้ตอบได้อย่างรวดเร็วและมีประสิทธิภาพ



โปรแกรม Thonny

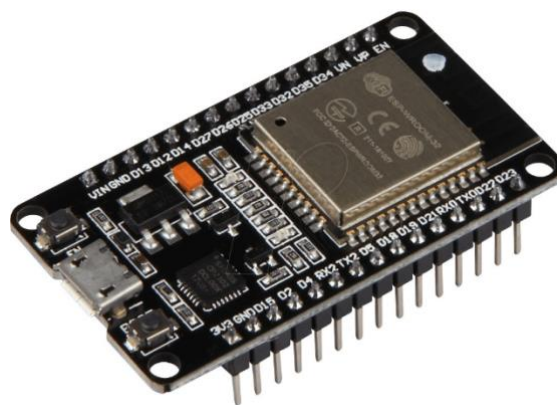
ESP32

ESP32 เป็นชื่อของไอซีไมโครคอนโทรลเลอร์ที่รองรับการเชื่อมต่อ WiFi และมีความสามารถในการเชื่อมต่อ Bluetooth Low-Energy (BLE, BT4.0, Bluetooth Smart) ผลิตโดยบริษัท Espressif Systems โดยจะมีสเปคโดยละเอียด ดังนี้

- ไมโครคอนโทรลเลอร์ Tensilica LX6 แบบ dual core ความเร็ว 240 MHz พร้อม 600 DMIPS
- มี RAM ในตัวที่มีความจุ 512 KB
- มี Wi-Fi transceiver รองรับมาตรฐาน 802.11 b/g/n HT40 ซึ่งรวมถึง baseband, stack และ LwIP

อยู่ในตัว

- มีบลูทูธในตัว รองรับการทำงานในโหมด 2.0 และโหมด 4.0 BLE
- รองรับการทำงานแบบ Bluetooth dual mode (classic และ BLE)
- มี Flash Storage ขนาด 16 MB
- รองรับแรงดันไฟฟ้าการทำงานระหว่าง 2.3V ถึง 3.6V
- สามารถทำงานได้ที่อุณหภูมิ -40°C ถึง 125°C
- รองรับการเชื่อมต่อคริสตัล 32.768 kHz สำหรับใช้กับส่วนวงจรนับเวลาโดยเฉพาะ



Microcontroller ESP32

General Purpose Input/Output

GPIO (General Purpose Input/Output) หรือเรียกอีกอย่างว่า “Pins” บนบอร์ด ESP32 สิ่งที่เราต้องให้ความสำคัญคือ คุณสมบัติเบื้องต้นของขาพอร์ต GPIO ของบอร์ด ESP32 เนื่องจากขาพอร์ตแต่ละขามีคุณสมบัติที่ต่างกัน บางขาพอร์ตที่สามารถกำหนดให้ทำหน้าที่เป็น Input เพื่อรับข้อมูล (เช่น อ่านค่าจากเซ็นเซอร์หรือสวิตช์) หรือบางขาพอร์ตกำหนดให้ทำหน้าที่เป็น Output เพื่อส่งสัญญาณไปยังอุปกรณ์อื่น ๆ (เช่น ควบคุม LED หรือมอเตอร์) บางขาพอร์ตสามารถทำงานได้ทั้งการ Input และ Output หรือบางขาพอร์ตในโหมด Input กำหนดให้ Pullup register ภายในได้ บางขาพอร์ตไม่สามารถที่จะกำหนดได้ แต่เมื่อเกิดการรีเซ็ต ไมโครคอนโทรลเลอร์ ESP 32 จะกลับไปมีคุณสมบัติของขาพอร์ตตามคุณสมบัติเดิมของฮาร์ดแวร์

GPIO ของบอร์ด ESP32 มีสิ่งที่ต้องระวังให้มากเป็นพิเศษคือเรื่องในการต่อสายไฟ เนื่องจาก GPIO สามารถรองรับและทนแรงดันไฟสูงสุด 3.3 V เท่านั้น หากได้รับแรงดันไฟที่สูงกว่า 3.3 V ต่อเข้าสู่พอร์ตก็จะเสียหายทันที และไม่สามารถเปลี่ยนหรือซ่อมแซมได้

ในการใช้งาน Pins และ GPIO ในไมโครไพทอนจะต้องใช้คลาสไลบรารี “Pin” ในการอ่านเขียนสัญญาณดิจิทัลกับขาพอร์ต ESP32 โดยสามารถเขียนค่า (output voltage) และอ่านค่า (input voltage) ซึ่งจะใช้คำสั่ง `machine.Pin()` ในการสร้าง Object pin ที่ต้องการใช้งาน GPIO โดย `machine.Pin()` มีรูปแบบการใช้งานดังนี้

`Object = machine.Pin(pin, machine.Pin.mode)`

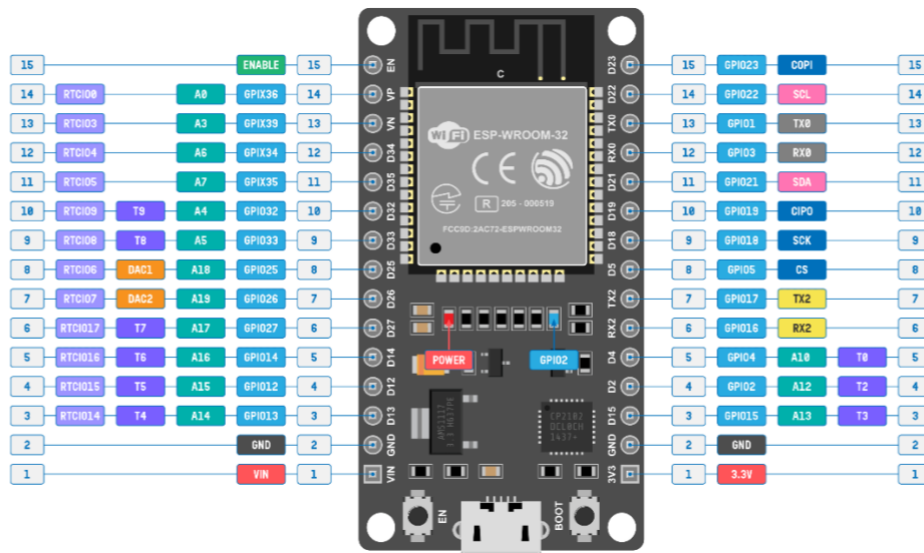
ซึ่งสามารถเห็นได้ว่าคำสั่งมีพารามิเตอร์ 3 ตัว คือ

1. **Object** ชื่อของอุปกรณ์ที่ต่อกับขาพอร์ต GPIO ที่ต้องการควบคุม
2. **pin** กำหนดหมายเลขขาพอร์ต GPIO ที่ต้องการควบคุม
3. **Pin.mode** ซึ่งโหมดที่ต้องการให้ขาพอร์ต GPIO ซึ่งสามารถเป็นได้ดังนี้
 - Pin.IN กำหนดให้ขาพอร์ต GPIO เป็นโหมดอ่านข้อมูล (input mode)
 - Pin.OUT กำหนดให้ขาพอร์ต GPIO เป็นโหมดเขียนข้อมูล (output mode)

โดยใช้ตัวเลขหลัง GPIO ในการกำหนดขา ตัวอย่างเช่น

```
led_pin = machine.Pin(21, machine.Pin.OUT)
```

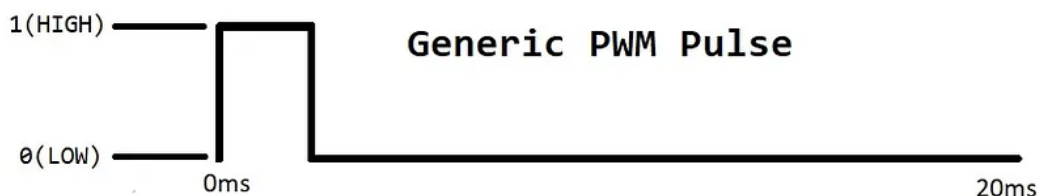
คือการกำหนดให้ LED ที่ต่อกับ GPIO ขาพอร์ตที่ 21 ให้มีสถานะเป็นโหมดเขียนข้อมูล (output mode)



ESP32-DevKit Pinout

Pulse Width Modulation

PWM (Pulse Width Modulation) เป็นการควบคุมการสร้างสัญญาณดิจิทัล (Digital control) จะสร้างสัญญาณคลื่นสี่เหลี่ยมออกมา โดยสัญญาณที่สร้างออกมาจะสลับกันระหว่างค่าสถานะ HIGH (1) ที่มี duty 100% หรือ 1023 และค่าสถานะ LOW (0) จะมี duty 0% หรือ 0



ด้วยหลักการสามารถกำหนดระดับความแรงดันของขาที่ใช้เป็น PWM ได้ด้วยการกำหนดค่า duty เช่น ปกติสามารถส่งแรงดันไฟฟ้า 3.3 V ถ้าต้องการให้เหลือครึ่งหนึ่งหรือ 1.65 V ให้กำหนดค่า duty เป็น 512 ซึ่งสามารถหาค่า duty ได้จาก

$$\text{duty} = 1023 \times \frac{\text{volt}}{3.3}$$

ไมโครโพรเซสเซอร์ใช้คลาส PWM ภายใต้คำสั่ง machine.PWM) โดยทุกขาพอร์ตสามารถเปิดใช้งาน output สัญญาณ PWM ได้ทั้งหมด โดยมีการกำหนดค่า duty อยู่ในค่า 0 ถึง 1023 และช่วงของสัญญาณความถี่อยู่ที่ 1Hz ถึง 40 Hz ซึ่งเมื่อความถี่สูงขึ้นความละเอียดของสัญญาณจะลดลง มีรูปแบบการใช้งานดังนี้

การสร้าง Object pin ที่ต้องการใช้งาน PWM

```
Object = machine.PWM( machine.Pin(หมายเลขขาพอร์ต GPIO) )
```

การกำหนดค่าสัญญาณความถี่และค่า duty cycle

```
Object.freq(ค่าความถี่)
```

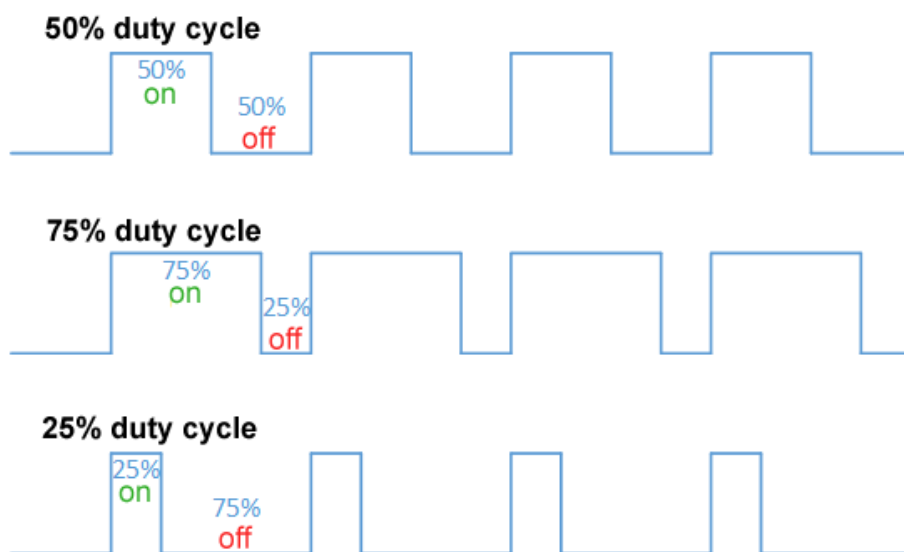
```
Object.duty(ค่า duty)
```

หรือ Object = PWM(Pin(หมายเลขขาพอร์ต GPIO), freq = ค่าความถี่, duty = ค่า duty)

ยกเลิกการทำงานของสัญญาณ PWM ได้โดย

```
Object.deinit()
```

ซึ่งการนำ PWM ไปประยุกต์ใช้ได้หลากหลายอย่าง เช่น การนำไปควบคุมความเร็วของมอเตอร์ไฟฟ้า การควบคุมความสว่างของหลอดไฟ LED การควบคุมการหมุนของเซอร์โวมอเตอร์ เป็นต้น



ILI9341

ILI9341 เป็นชิปควบคุมหน้าจอ LCD TFT (Thin-Film Transistor Liquid Crystal Display) ที่มีความสามารถในการแสดงผลสี 16 บิต สามารถแสดงภาพที่สวยงามและคมชัดได้ มีความละเอียดสูง 320x240 pixels ที่ใช้งานได้ดีกับการแสดงผลกราฟิกและตัวอักษรบนหน้าจอ

ILI9341 มีความสามารถในการควบคุมหน้าจอ TFT LCD ได้อย่างแม่นยำ รองรับการแสดงผลภาพและข้อมูลต่างๆ ได้อย่างรวดเร็ว และมีการเชื่อมต่อผ่านส่วนต่างๆ เช่น SPI (Serial Peripheral Interface) ซึ่งเป็นอินเทอร์เฟซที่ใช้งานได้ดีในการสื่อสารระหว่างไมโครคอนโทรลเลอร์กับจอ LCD TFT ทำให้เป็นที่นิยมสำหรับการใช้งานในโปรเจกต์ที่ต้องการการแสดงผลที่คมชัดและมีสีสันที่สวยงาม



หน้าจอ ILI9341

pH sensor

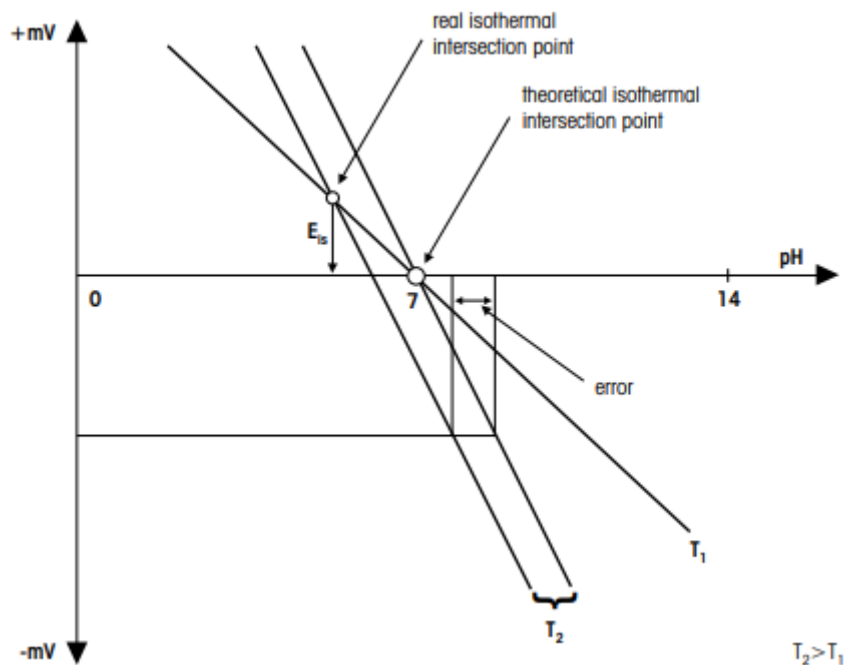
pH sensor เป็นอุปกรณ์ที่ใช้วัดค่า H^+ ในสารละลาย โดยจะระบุความเข้มข้นของ H^+ เป็นค่าที่แสดงความเป็นกรด-ด่าง ดังนั้นค่า pH ของน้ำบริสุทธิ์ที่อุณหภูมิห้องอยู่ที่ประมาณ 7 ซึ่งถือว่าเป็น "เป็นกลาง" เนื่องจากความเข้มข้นของ H^+ เท่ากับความเข้มข้นของ OH^- ที่เกิดจากการแตกตัวของน้ำ ความเข้มข้นที่เพิ่มขึ้นของ H^+ ที่สัมพันธ์กับ OH^- ที่ลดลงจะสร้างสารละลายที่มีค่า pH น้อยกว่า 7 สารละลายนั้นถือว่าเป็น "acidic" และถ้าความเข้มข้นที่ลดลงของ H^+ ที่สัมพันธ์กับ OH^- ที่เพิ่มขึ้นจะสร้างสารละลายที่มีค่า pH มากกว่า 7 สารละลายนั้นถือว่าเป็น "alkaline" หรือ "basic"

ดังนั้นตามระบบการวัดค่า pH เมื่อสารละลายสองชนิดที่มีความเข้มข้นของไอออน H^+ ต่างกันถูกแยกออกจากกันโดย glass membrane ศักย์ไฟฟ้าจะกระจายไปทั่วเมมเบรน (Sensing electrode) และยังมีการสร้างศักย์ไฟฟ้าจาก reference electrode อีกด้วย ด้วยเหตุนี้ pH meter จึงวัดความต่างศักย์ของแรงดันไฟฟ้า (mV) ระหว่าง sensing electrode และ reference electrode และแสดงค่า pH ผ่านอัลกอริทึม

pH sensor เป็น voltmeter ที่มี sensitivity สูง เมื่อนำ pH sensor จุ่มลงในสารละลาย ศักย์ไฟฟ้า mV จะถูกสร้างขึ้นเพื่อตอบสนองต่อความเข้มข้นของ H^+ ดังนั้น voltage ทางทฤษฎีที่สร้างขึ้นสามารถกำหนดได้จากสมการ Nernst

$$E = E_0 - \frac{RT}{nF} \log a_H \quad (a_H = \text{Hydrogen ion activity})$$

ในทางทฤษฎีที่อุณหภูมิ 25 °C สารละลาย pH 7.0 จะสร้าง 0 mV และจะมีการเปลี่ยนแปลง 59.16 mV สำหรับแต่ละหน่วย pH ดังนั้น ที่ pH 4.0 จะมีการสร้าง +177.48 mV ขณะที่ pH 10.0 โดยที่ความเข้มข้น H^+ ต่ำกว่าเมื่อเทียบกับ pH 7.0 จะมีการสร้างศักย์ไฟฟ้าที่ -177.48 mV ในการที่ Electrode วัดค่า pH ใหม่จะสร้างค่าระหว่าง +/- 10 mV ในค่า pH 7.0 และจะมีเปอร์เซ็นต์ความชันระหว่าง 95 ถึง 105% เปอร์เซ็นต์ความชันถูกกำหนดโดยการหาร voltage ที่เกิดขึ้นจริงตามทฤษฎีแล้วคูณด้วย 100



ขั้นตอนการทดลอง Week 1

1. ดาวน์โหลดและติดตั้งโปรแกรม Thonny จาก <https://thonny.org/> (เลือก version ล่าสุด)

Thonny
Python IDE for beginners








Download version **4.1.4** for
Windows • Mac • Linux

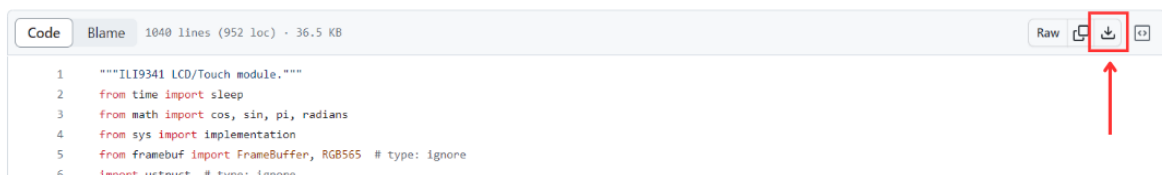
2. ตรวจสอบอุปกรณ์ที่ได้รับ และต่อสายไฟที่เชื่อมต่อกับขาพอร์ต GPIO ดังต่อไปนี้
 - GPIO2 เชื่อมต่อกับ GREEN
 - GPIO4 เชื่อมต่อกับ RED
 - GPIO16 เชื่อมต่อกับ DS18B20 (Temperature Sensor)
 - GPIO21 เชื่อมต่อกับ CONTROL_1
 - GPIO22 เชื่อมต่อกับ CONTROL_2
 - GPIO25 เชื่อมต่อกับ pH_PROBE
 - GPIO26 เชื่อมต่อกับ BUZZER
 - GPIO32 เชื่อมต่อกับ POT_1
 - GPIO33 เชื่อมต่อกับ POT_2
 - GPIO34 เชื่อมต่อกับ BUTTON_1
 - GPIO35 เชื่อมต่อกับ BUTTON_2
 - GPIO39 เชื่อมต่อกับ BUTTON_3
3. ดาวน์โหลดโค้ดสำหรับการแสดงผลหน้าจอ ILI9341 จาก <https://github.com/yyods/TitraLab> ดังนี้
 - ArcadePix9x11.c
 - EspressoDolce18x24.c
 - ili9341.py
 - xglcd_font.py

วิธีการดาวน์โหลดโค้ดมี ดังนี้

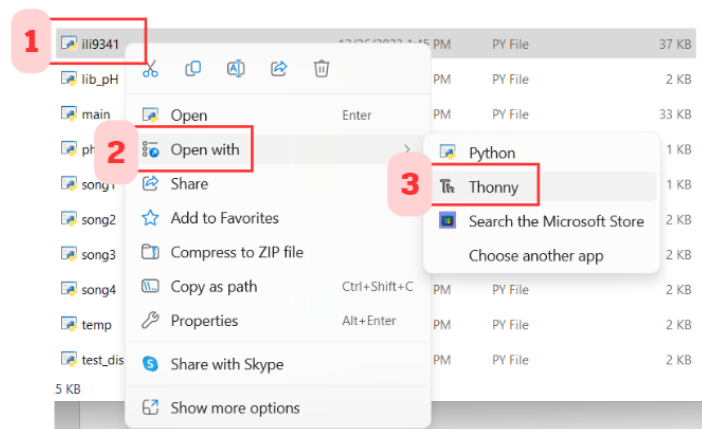
TitraLab / MicroPython / Show / 

 yyods Please enter the commit message for your changes. Lines starting 	
Name	Last commit message
..	
fonts	Please enter the commit message for your changes. Lines starting
 ili9341.py	Please enter the commit message for your changes. Lines starting
 main.py	Please enter the commit message for your changes. Lines starting
 xgld_font.py	Please enter the commit message for your changes. Lines starting

3.1 เลือกโค้ดที่ต้องการดาวน์โหลด



3.2 กดดาวน์โหลดโค้ด



3.3 เปิดโค้ดในโปรแกรม Thonny

Assignment Week 1

(กำหนด Pin โดยใช้ขาพอร์ต GPIO ดังที่แสดงในข้อ 2 นิสิตสามารถใช้ Generative AI ในการเรียนรู้ได้)

1. กดปุ่มเพื่อเปิด-ปิดไฟ*
2. แสดงผลหน้าจอโดยมีข้อความ 2 บรรทัด โดยให้ใช้ตัวอักษรแตกต่างกันและอยู่กึ่งกลางหน้าจอ*
3. แสดงผลหน้าจอโดยแสดงค่าที่อ่านได้จาก pH probe ทุก ๆ 3 วินาที ในหน่วย mV*
4. ใช้ PWM เพื่อหรี่หลอดไฟ LED ที่ CONTROL_1
5. แสดงผลหน้าจอโดยแสดงค่าที่อ่านได้จาก DS18B20 ทุก ๆ 3 วินาที
6. ใช้ BUZZER เพื่อเล่นเพลง 1 เพลง

*ข้อที่ 1-3 เป็นข้อที่จำเป็นต้องทำ ข้อที่ 4-6 หากทำจะได้คะแนนพิเศษ