Embedded Systems Laboratory

Lap7: - มีความรู้ความเข้าใจในรายละเอียดของ ADC ของ ESP32
- การโปรแกรมเพื่อควบคุมการทำงานโดยใช้ ADC
- การโปรแกรมประยุกต์ในการใช้งาน ADC ของ ESP32

อุปกรณ์ Lab7

1. บอร์ดทดลอง Embedded System 1 กล่อง
2. สายไฟสำหรับการต่อวงจร 1 ชุด
3. Adapter แปลงไฟ AC to DC12V 1 อัน
4. Laptop หรือ Notebook 1 เครื่อง
5. LDR, Resistor 1 ชุด

7.1 ข้อมูลเบื้องต้น ADC ของ ESP32

ให้นิสิตหาร่วมกันหาข้อมูลเพื่อนำมาตอบคำถามข้างล่างดังนี้

คำถาม	คำตอบ
ADC คือ	กระบวนการแปลงสัญญาณ analog เป็น digital
ESP32 มี ADC จำนวนเท่าใด? และผู้ใช้สามารถใช้ได้สูงสุดจำนวนกี่ช่อง?	SAR ADC 2 ชุด controller 5 ชุด สามารถใช้ได้สูงสุด 18 ขา
Vref ของ ESP32 มีค่า? ค่าลดทอน default มีค่า?	1.1 V -11dB
ความละเอียดของ ADC ของ ESP32 มีขนาดกี่บิต?	12 bit
ถ้าสัญญาณ Analog แรงดันคงที่ 3.0V ESP32 จะอ่านได้เท่าไหร่? (Dec, Hex) แสดงการคำนวณ และการแปลงค่า	3.0*4095/3.3 = 3723 3723 = 0x E8B

จงตอบคำถามทั่วไปสำหรับ PIN ADC ของ ESP32 จงใส่ GPIO ของ ESP32 ให้ครบถ้วนสมบูรณ์

ADC1 input channels

ADC1 CH	0	1	2	3	4	5	6	7
GPIO	36	37	38	39	32	33	34	35

ADC2 input channels

ADC2 CH	0	1	2	3	4	5	6	7	8	9
GPIO	4	0	2	15	13	12	14	27	25	26

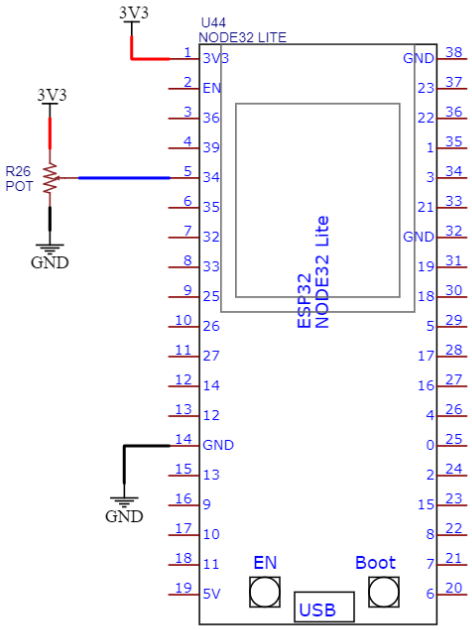
จงอธิบายการทำงาน Function ADC ใน ArduinoIDE

Function	คำตอบ
analogRead()	อ่านค่าสัญญาณอนาล็อกเข้ามาในขาที่ระบุ
analogReadResolution()	กำหนดค่าตวามละเอียดในการแปลงสัญญาณ (1-32)
analogSetWidth()	กำหนดค่าตวามละเอียดในการแปลงสัญญาณ (9-12)
analogSetCycles()	กำหนดค่าความเร็วในการสุ่ม sample
analogSetSamples()	กำหนดจำนวน sample ใน range (default 1)
analogSetClockDiv()	ตั้งค่าตัวหารของ clock (1-255)
analogSetAttenuation()	ตั้งค่าการลดทอน
analogSetPinAttenuation()	ตั้งค่าการลดทอนของช่องสัญญาณนั้น ๆ
adcAttachPin()	ตั้งค่าเป็น pin สำหรับ adc
adcStart()	เริ่มการแปลง adc
adcBusy()	ตรวจสอบการทำงานของ pin นั้น
resultadcEnd()	หยุดการส่งข้อมูล return ค่า เป็น 16-bit interger

7.2 Potentiometer

ให้นิสิต นำ ESP32 ต่อวงจรร่วมกับ Rปรับค่าได้(Potentiometer) ดังรูป

วงจรทดสอบ



Firmware

```
// Potentiometer is connected to GPIO
34 (Analog ADC1_CH6)
const int potPin = 34;

int potValue = 0;

void setup() {
  Serial.begin(115200);
  delay(1000);
}

void loop() {

  potValue = analogRead(potPin);

  Serial.println(potValue);

  delay(500);
}
```

เมื่อปรับแรงดันเป็น 0V สังเกตผลลัพท์ จากนั้นให้เขียนผลลัพท์ที่ได้จาก Serial Monitor

0

เมื่อปรับแรงดันเป็น 3.3V สังเกตผลลัพท์ จากนั้นให้เขียนผลลัพท์ที่ได้จาก Serial Monitor

4095

ให้ทดลองปรับ Potentiometer ให้มีค่า 3000 จะสามารถวัดแรงดันได้เท่าใด

ให้นิสิตเขียนวิธีการคำนวณ

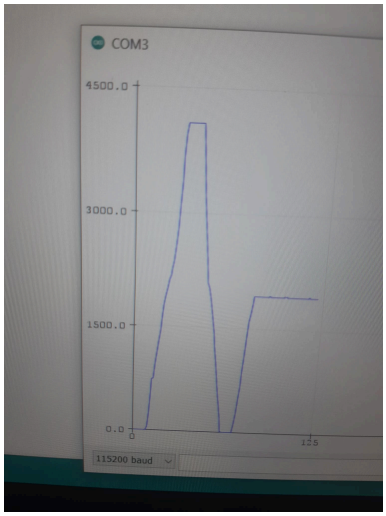
3000*3.3/4095 = 2.42 V

Potentiometer ให้มีค่า 3000 วัดแรงดันโดยใช้ Multimeter วัดแรงดันจริงได้เท่าใด (ถ้ามี)

ทดลอง Plot Graph ดังนี้

- เลื่อน POT ไปทาง 0V - เลื่อน POT ไปทาง 3.3V

สังเกตความผิดปกติ และถ่ายรูปกราฟที่แสดงความผิดปกติในทั้ง 2 เหตุการณ์

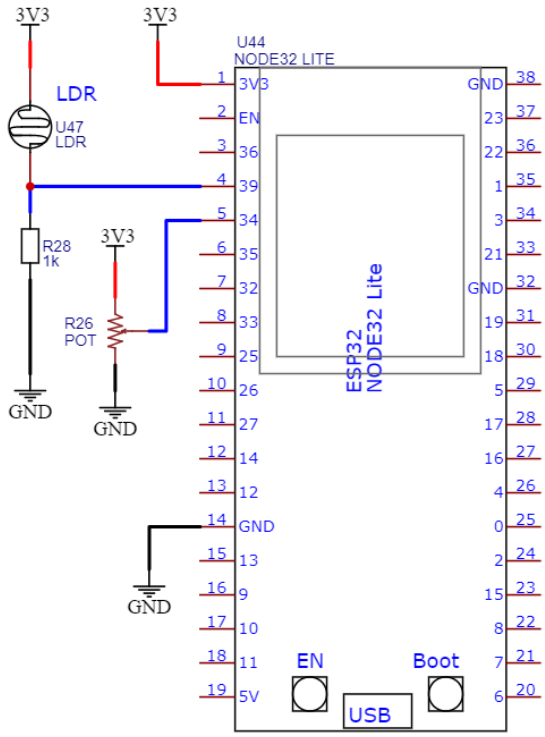


ปัญหา ADC ของ ESP32 คืออะไร?

วัดค่าได้ไม่แม่นยำ output เป็น 0 ก่อนที่จะปรับค่า R เป็นค่าต่ำที่สุด และ เป็น 4095 ก่อนที่ R ที่ปรับจะเป็นค่าสูงสุด

7.3 LDR (Light Dependent Resistor)

ให้นิสิต นำ ESP32 ต่อวงจรร่วมกับ Rปรับค่าได้(Potentiometer) และ LDR ดังรูป



การทำงานของ Firmware ของโจทย์ข้อนี้ รายละเอียดดังต่อไปนี้

1. เมื่อเริ่มทำงาน ESP32 จะอ่านค่า Analog ช่อง GPIO39 และ GPIO34 ออกทาง Serial Monitor แสดงผลดังนี้

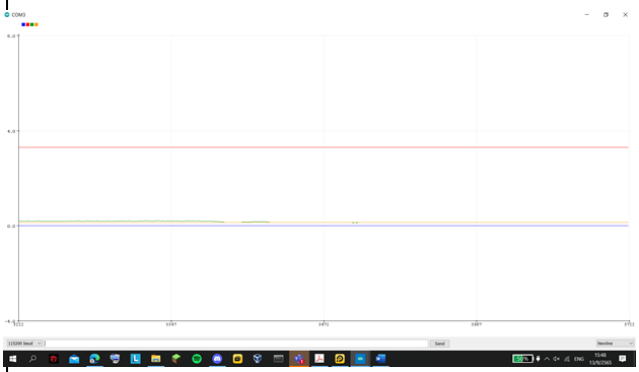
   Lower limit Volt, Upper limit Volt, LDR Volt, POT Volt

   *** แสดงเป็นค่าแรงดันทศนิยม 2หน่วย เช่น 0.00, 3.30, 1.23, 3.25 เป็นต้น

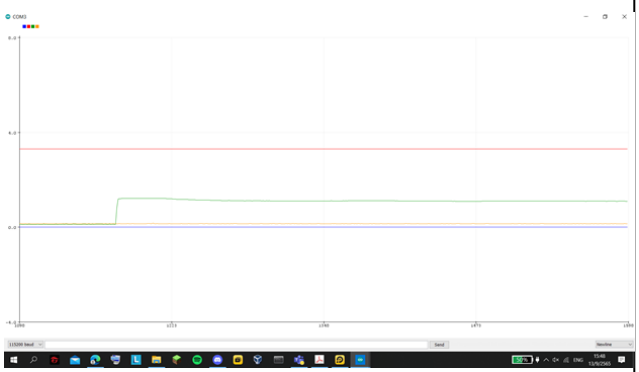
2. ตั้งค่า Sampling rate ที่ 10ms โดยใช้ Timer Interrupt หรือ millis()

3. ทำการทดลอง โดยการส่องไฟฉาย (ใช้จากมือถือได้) ไปที่ LDR ถ่ายรูปเปรียบเทียบระว่างส่องไฟและไม่ส่องไฟ

เมื่อ ส่อง/ไม่ส่องไฟไปยัง LDR สังเกตผลลัพท์ จากนั้นให้ถ่ายรูปที่ได้จาก Serial Plotter ลงกล่องคำตอบ



เมื่อไม่ส่องไฟ



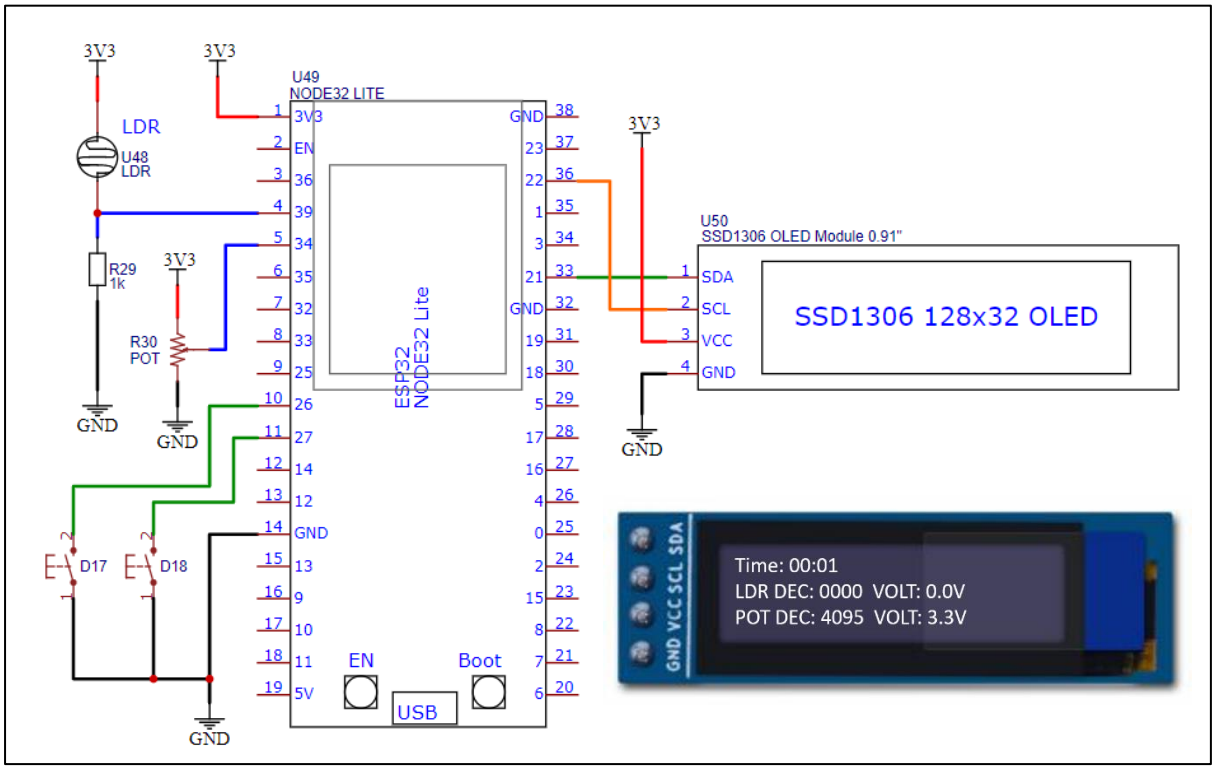
เมื่อส่องไฟ

เขียนโปรแกรมลงในกล่องคำตอบด้านล่าง ในกรณีตัวหน้ากระดาษไม่พอให้เพิ่มหน้าแทรกในไฟล์แทน

```
#define LDR 39
#define POT 34

long lasttime10 = 0;

void setup()
{
  Serial.begin(112500);
}

void loop()
{ //10
  if (millis() - lasttime10 >= 10)
  { lasttime10 = millis();
    int ldr_value = analogRead(LDR);
    int pot_value = analogRead(POT);
    float ldr_volt = ldr_value * 3.3 / 4095;
    float pot_volt = pot_value * 3.3 / 4095;
    Serial.printf("0.00,3.30,%.2f,%.2f\n", ldr_volt, pot_volt);
  }
}
```

7.4 Assigment ADC

วงจรทดสอบ นำ ESP32 ต่อวงจรร่วมกับ LDR, POT, OLED 0.91” และ Tact Switch ดังรูป



การทำงานของ Firmware ของโจทย์ข้อนี้ รายละเอียดดังต่อไปนี้

1. โปรแกรมอ่านค่า ADC จำนวน 2ช่องแสดงผลบน OLED
   - บรรทัดแรก ให้เขียน Code นาฬิกาแสดงบน OLED
   - บรรทัดสอง ให้เขียน ADC แสดง Data12bit และ Volt ของ LDR
   - บรรทัดสอง ให้เขียน ADC แสดง Data12bit และ Volt ของ POT
2. การแสดงเวลาที่ OLED Display อยู่ในรูปนาฬิกามาตรฐาน 00:00
   - เวลาเดินขึ้นที่ละ 1วินาที
   - เครื่องหมาย : กระพริบติดดับทุก 500ms กรณีสั่งหยุด : จะกระพริบต่อไป
3. สามารถควบคุมการทำงานโดยใช้ Serial Monitor
   - ป้อน 1 เวลาจะเดินขึ้นทุกๆ 1 วินาที
   - ป้อน 2 เวลาจะหยุดเดิน
4. สามารถควบคุมการทำงานโดยใช้ Switch D1/D2
   - กด D17 เวลาจะเดินขึ้นทุกๆ 1 วินาที
   - กด D18 เวลาจะหยุดเดิน
5. ในกรณี Clock Stop ยังสามารถอ่านค่า ADC ออกทาง OLED ได้ตามปกติ และเครื่องหมาย : ยังกระพริบปกติ

เขียนโปรแกรมลงในกล่องคำตอบด้านล่าง **และถ่ายวีดีโอผลลัพท์ของโจทย์นี้ Upload ไฟล์ตามหมู่เรียน**

ในกรณีตัวหน้ากระดาษไม่พอให้เพิ่มหน้าแทรกในไฟล์แทน

```
#define LDR 39
#define POT 34
#define D17 26
#define D18 27
#define LED 2
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
Adafruit_SSD1306 display(128, 32, &Wire);

bool sToggle = true;
bool tS = false;
long lasttime10 = 0, lasttime500 = 0, lasttime1000 = 0;
int mintime = 0, sectime = 0;
int ldr_value;
int pot_value;
float ldr_volt;
float pot_volt;

void IRAM_ATTR isrSW1() {
  tS = false;
}
void IRAM_ATTR isrSW2() {
  tS = true;
}

void setup()
{
  Serial.begin(112500);
  pinMode(D17, INPUT_PULLUP);
  pinMode(D18, INPUT_PULLUP);
  pinMode(LED, OUTPUT);
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  attachInterrupt(digitalPinToInterrupt(D18),
isrSW2,CHANGE);
  attachInterrupt(digitalPinToInterrupt(D17),
isrSW1,CHANGE);
}
```

เขียนโปรแกรมลงในกล่องคำตอบด้านล่าง **<u>และถ่ายวีดีโอผลลัพท์ของโจทย์นี้ Upload ไฟล์ตามหมู่เรียน</u>**

ในกรณีตัวหน้ากระดาษไม่พอให้เพิ่มหน้าแทรกในไฟล์แทน

```
void loop()
{ //10
  if (millis() - lasttime10 >= 10)
  { lasttime10 = millis();
    ldr_value = analogRead(LDR);
    pot_value = analogRead(POT);
    ldr_volt = ldr_value * 3.3 / 4095;
    pot_volt = pot_value * 3.3 / 4095;
    Serial.printf("0.00,3.30,%.2f,%.2f\n", ldr_volt, pot_volt);
  }
  //500
  if (millis() - lasttime500 >= 500)
  { lasttime500 = millis();
    sToggle = !sToggle;
    display.clearDisplay();
    display.setTextSize(1);
    display.setTextColor(SSD1306_WHITE);
    display.setCursor(0, 0);
    digitalWrite(LED, sToggle);
    if (sToggle || tS)
    {
      display.printf("Time: %02d:%02d", mintime, sectime);
    } else
    {
      display.printf("Time: %02d %02d", mintime, sectime);
    }
    display.setCursor(0, 12);
    display.printf("LDRDEC:%04d VOLT:%.1fV", ldr_value, ldr_volt);
    display.setCursor(0, 24);
    display.printf("POTDEC:%04d VOLT:%.1fV", pot_value, pot_volt);
    display.display();
  }
  //1000
  if (millis() - lasttime1000 >= 1000 && tS == false)
  { lasttime1000 = millis();
    sectime++;
    if (sectime >= 60)
    { sectime = 0;
      mintime++;
    } if (mintime >= 60)
    { mintime = 0;
    }
  }
}
```