Electronic Voting Machine-1

What is our GOAL for this CLASS?

In this class, we learned **how to design electronic voting Machine on OLED and how calculations and results can be displayed on OLED by Arduino Programming**

What did we ACHIEVE in the class TODAY?

- We learned how to Design EVM on OLED
- We learned about OLED programming

Which CONCEPTS/ CODING BLOCKS did we cover today?

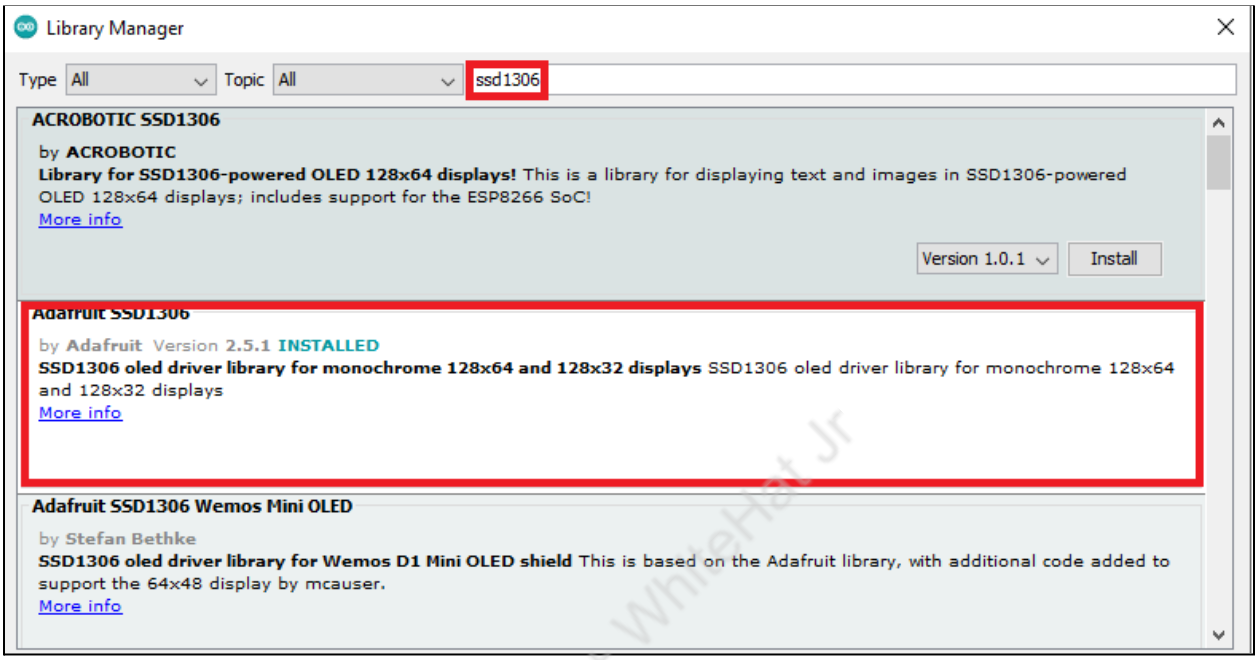
- We learned how to develop our own voting machine.
- We learned how to control OLED.
- We learned how to print data on OLED.

How did we DO the activities?

1. Gather the material from the IoT kit Collect the material
   - 1 x ESP32
   - 1 x USB Cable
   - 4 x Breadboard
   - 2 x Jumper wires
   - 1 x OLED
   - 5 x PushButtons
   - 15 x Jumper Wires -9 Female to Male, 6 Male to Male

2. Do connections:
    - The circuit of this project consists of an ESP32 Controller, pushbuttons, and an OLED screen. Complete processes are controlled by ESP32 Controller, including reading buttons, incrementing vote values, generating results, and sending votes and results to an OLED.
    - Take five buttons in which the first button is for A Party second for B Party third is for C Party fourth is for D Party others and the last button is used for calculating or displaying results.
    - Take **ESP32** from the kit.
    - Take five pushbuttons
    - Insert five push buttons on the breadboard one by one keep spacing between buttons.
    - The five push button's **positive parts** are directly connected with ESP32 **pin no. 13, 33, 14, 27, 26** respectively.
    - Supply negative supply from ESP32 GND to the power rail of the breadboard. Take one **Female to Male** jumper wire, Insert the **Female TERMINAL** into the **ESP32 GND** terminal and drag it to the breadboard and insert **MALE TERMINAL** into the **negative terminal** of the Power rail.
    - The **negative part** of all the **pushbuttons** will be directly connected to the **negative power rail** of the **breadboard.**
    - Take an **OLED** from the kit.
    - Take **Female to Male** jumper wire**.** Use **Female part at ESP32 terminal** and **Male part** on the breadboard
    - Connect OLED PIN VCC to ESP32 PIN 3.3V
    - Connect OLED PIN GND to ESP32 PIN GND
    - Connect OLED PIN SCL to ESP32 PIN GPIO22
    - Connect OLED PIN SDA to ESP32 PIN GPIO21

3. install libraries
    - Open **Arduino IDE**
    - Go to **Tools**
    - Go to **Manage Libraries**
    - Type "**SSD1306"** in the search box and install the **SSD1306** library from Adafruit.
    - After installing the SSD1306 library from Adafruit, type **"GFX"** in the search box and install library
    - After installing the libraries, restart your Arduino IDE.



4. Import Libraries
   - **SPI.h Serial Peripheral Interface (SPI)** is a synchronous serial communication protocol used by microcontrollers for communicating with one or more peripheral devices quickly over short distances.When using SPI, there is always one master device (usually a microcontroller) that controls all peripheral devices.
   - **Wire.h** This library allows you to communicate with I2C / devices. I2C is **a serial communication protocol**, so data is transferred bit by bit along a single wire.
   - **Adafruit_GFX.h:** This library offers a common graphical syntax and set of functions for all LCD displays, OLED displays, and LED matrices.
   - **Adafruit_SSD1306** : This library takes care of low-level communication with the hardware.

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
```

5. Write the program:
   - Define **SCREEN_WIDTH & SCREEN_HEIGHT** for OLED

```
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
```

   - Declaration of an **SSD1306** display that connects to **I2C** communication using

Wire Library

- Initialize a **display** object with the **SCREEN_WIDTH & SCREEN_HEIGHT** defined earlier with the I2C communication protocol.
- A value of **(-1)** indicates that our OLED display does not have a **RESET** pin. Sometimes OLED displays have a RESET pin on the OLED, in that case, we should connect it to a GPIO and should include the GPIO number as a parameter.

```
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
```

- Initialize using **void setup()** function

```
void setup() {
  Serial.begin(115200);

  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for(;;);
  }
```

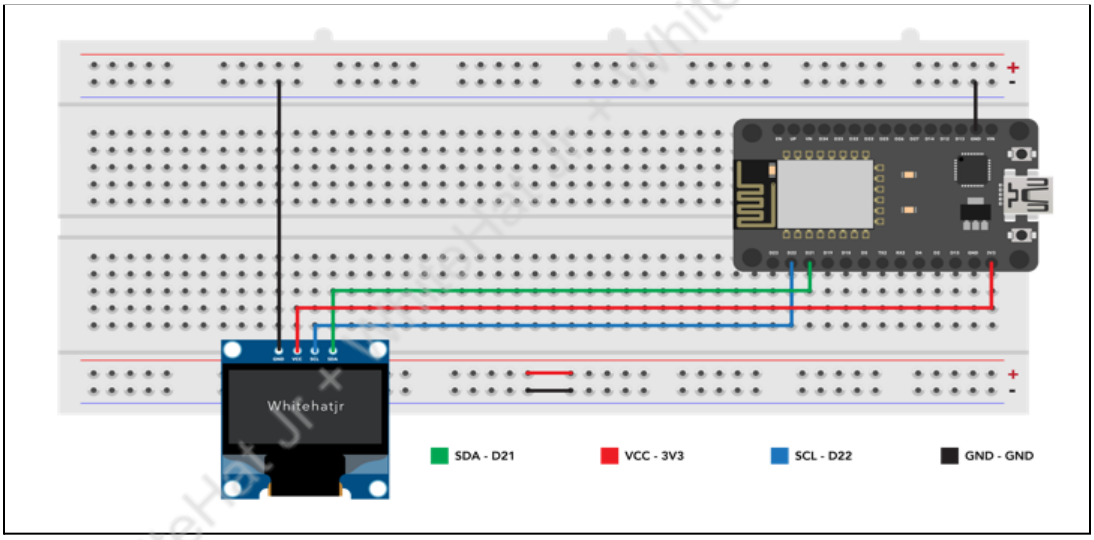
6. Set up for Voting Machine.

7. **Gather the material from the IoT kit:**
   - 1 x ESP32
   - 1 x USB Cable
   - 4 x Breadboard
   - 1 x OLED
   - 5 x Push Buttons

8. **Do connections:**
   - The circuit of this project consists of an ESP32 Controller, pushbuttons, and an OLED screen. Complete processes are controlled by ESP32 Controller, including reading buttons, incrementing vote values, generating results, and sending votes and results to an OLED.
   - Take five buttons in which the first button is for A Party second for B Party third is for C Party fourth is for D Party others and the last button is used for calculating or displaying results.
   - Take **ESP32** from the kit.
   - Take five pushbuttons
   - Insert five push buttons on the breadboard one by one keep spacing between buttons.
   - The five push button's **positive parts** are directly connected with ESP32 **pin no.**

13, 33, 14, 27, 26 respectively.

- Supply negative supply from ESP32 GND to the power rail of the breadboard. Take one **Female to Male** jumper wire, Insert the **Female TERMINAL** into the **ESP32 GND** terminal and drag it to the breadboard and insert **MALE TERMINAL** into the **negative terminal** of the Power rail.
- The **negative part** of all the **pushbuttons** will be directly connected to the **negative power rail** of the **breadboard.**
- Take an **OLED** from the kit.
- Take **Female to Male** jumper wire**.** Use **Female part at ESP32 terminal** and **Male part** on the breadboard
- Connect OLED PIN VCC to ESP32 PIN 3.3V
- Connect OLED PIN GND to ESP32 PIN GND
- Connect OLED PIN SCL to ESP32 PIN GPIO22
- Connect OLED PIN SDA to ESP32 PIN GPIO21



9. Initialize the **setup()**
    - Define the **GPIO Pins** for all five **PushButtons 13, 33, 14, 27, 26**

```
#define S1 13
#define S2 33
#define S3 14
#define S4 27
#define S5 26
```

10. Declare the variables for **vote1, vote2, vote3, vote4, vote5**
    - int is used to define the datatype
    - vote1, vote2, vote3, vote4
    - Vote1 is used for A party
    - Vote 2 is used for B party
    - Vote 3 is used for C party
    - Vote 4 is used for D party

```
int vote1 = 0;
int vote2 = 0;
int vote3 = 0;
int vote4 = 0;
```

- Variable **button state** will store the current value of **button State**

```
int buttonState = 0;
```

11. Initialize the setup()

- **Serial. begin(9600)** is used for data exchange speed. speed parameters. This tells the Arduino to get ready to exchange messages with the Serial Monitor at a data rate of 9600 bits per second. That's 9600 binary ones or zeros per second and is commonly called a baud rate.
- **pinMode()** configures the specified pin to behave either as input or output. Since we want this pin for output, we are writing **OUTPUT** here.
- **Syntax**: *pinMode(pin, mode)*
- **pin**: The pin do we need to set

- **mode**: Set the mode **INPUT, OUTPUT**
- Use all five push_buttons as **OUTPUT**

```
void setup()
{

  Serial.begin(9600);

  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(S3, OUTPUT);
  pinMode(S4, OUTPUT);
  pinMode(S5, OUTPUT);
```

12. The pin needs to be programmed to be either ON or OFF, that is, we can command it to be ON (output 5 volts), or OFF (output 0 volts).
    - To make it **HIGH**, use a function called **digitalWrite().**
    - Make **HIGH** for all pushbuttons i.e. **S1,S2,S3,S4,S5**

```
digitalWrite(S1, HIGH);
digitalWrite(S2, HIGH);
digitalWrite(S3, HIGH);
digitalWrite(S4, HIGH);
digitalWrite(S5, HIGH);
```

- Initialize the OLED display with the **begin()** method.
- If the OLED displays nothing, check the OLED address at **0x3C**. the address is 0x3C.
- If you are not able to connect to the display, it prints a message on the Serial Monitor. If something fails, don't proceed further, try to repeat the process using **for() l**oop

```
// initialize OLED display with I2C address 0x3C
if (!oled.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
  Serial.println(F("failed to start SSD1306 OLED"));
  while (1);
}
```

13. Print data on OLED

- In order to allow the OLED to initialize, add a **two-second delay** before writing text
- Clear the display buffer with the **clearDisplay()** method after initializing the

display

- To **write** text, you must first set the font size, color, and location where the text will be displayed in the OLED and data which need to be printed.
- Set the font size using the **setTextSize()** method
- Set the font color using the **setTextColor()** method.**WHITE** sets white font and black background.
- Using the **setCursor(x,y)** method, specify the starting point of the text. In this case, the text will be started at **(2, 5).**
- As a final step, display data using the **println()** method. Print **("Voting Machine")**
- At last, we need to call the **display()** method to actually display the text on the screen.

```
delay(2000);          // wait two seconds for initializing
oled.clearDisplay(); // clear display

oled.setTextSize(1);          // set text size
oled.setTextColor(WHITE);     // set text color
oled.setCursor(2, 5);        // set position to display
oled.println("Voting Machine"); // set text
oled.display();               // display on OLED

delay(2000);

oled.clearDisplay();
```

14. Call the main loop()

- To **write** text, you must first set the font size, color, and location where the text will be displayed in the OLED and data which need to be printed.
- Set the font size using the **setTextSize()** method.
- Set the font color using the **setTextColor()** method.**WHITE** sets white font and black background.
- Using the **setCursor(x,y)** method, specify the starting point of the text. In this case, the text will be started at **(2,5).**
- As a final step, display data using the **println()** method.Print **("A").** Here **A** refers to **Party A**
- At last, we need to call the **display()** method to actually display the text on the screen.

```
void loop()
{
  oled.setTextSize(2);
  oled.setTextColor(WHITE);
  oled.setCursor(1, 0);
  oled.print("A");
  oled.setCursor(1, 50);
  oled.print(votel);
  oled.display();
```

- To **write** text, first set the font size, color, and location where the text will be displayed in the OLED and data which need to be printed.
- Set the font size using the **setTextSize()** method.
- Set the font color using the **setTextColor()** method.**WHITE** sets white font and black background.
- Using the **setCursor(x,y)** method, specify the starting point of the text. In this case, the text will be started at **(2, 5).**
- As a final step, display data using the **println()** method.Print **("B").** Here **B** refers to **Party B** and it will print **vote 2** for the party.
- Call the **display()** method to actually display the text on the screen.

```
  oled.setTextSize(2);
  oled.setTextColor(WHITE);
  oled.setCursor(35, 0);
  oled.print("B");
  oled.setCursor(35, 50);
  oled.print(vote2);
  oled.display();
```

- To **write** text, you must first set the font size, color, and location where the text will be displayed in the OLED and the data which need to be printed.
- Set the font size using the **setTextSize()** method.
- Set the font color using the **setTextColor()** method.**WHITE** sets white font and black background.
- Using the **setCursor(x,y)** method, specify the starting point of the text. In this case, the text will be started at **(65,0)**.vote will display just before the C i.e coordinates will be (65,50)
- As a final step, display data using the **println()** method. Print **("B").** Here **B** refers to **Party C** and it will print vote 3 for the party.
- Call the **display()** method to actually display the text on the screen.

```
oled.setTextSize(2);
oled.setTextColor(WHITE);
oled.setCursor(65, 0);
oled.print("C");
oled.setCursor(65, 50);
oled.print(vote3);
oled.display();
```

- To **write** text, you must first set the font size, color, and location where the text will be displayed in the OLED and data which need to be printed.
- Set the font size using the **setTextSize()** method.
- Set the font color using the **setTextColor()** method.**WHITE** sets white font and black background.
- Using the **setCursor(x,y)** method, specify the starting point of the text.
- As a final step, display data using the **println()** method. Print **("D").** Here **D** refers to **Party D** and it will print vote 4 for the party.
- Call the **display()** method to actually display the text on the screen.

```
oled.setTextSize(2);
oled.setTextColor(WHITE);
oled.setCursor(100, 0);
oled.print("D");
oled.setCursor(100, 50);
oled.print(vote4);
oled.display();
```

15. **Output:**
    - Compile and upload the program to the ESP32 board using Arduino IDE
    - Verify the program by clicking the Tick option.
    - Upload the program by clicking the arrow option.

What's NEXT?

In the **next class**, we will learn about **Electronic Voting Machine2.**

Expand Your Knowledge

To know more about **different categories of OLED** click here.