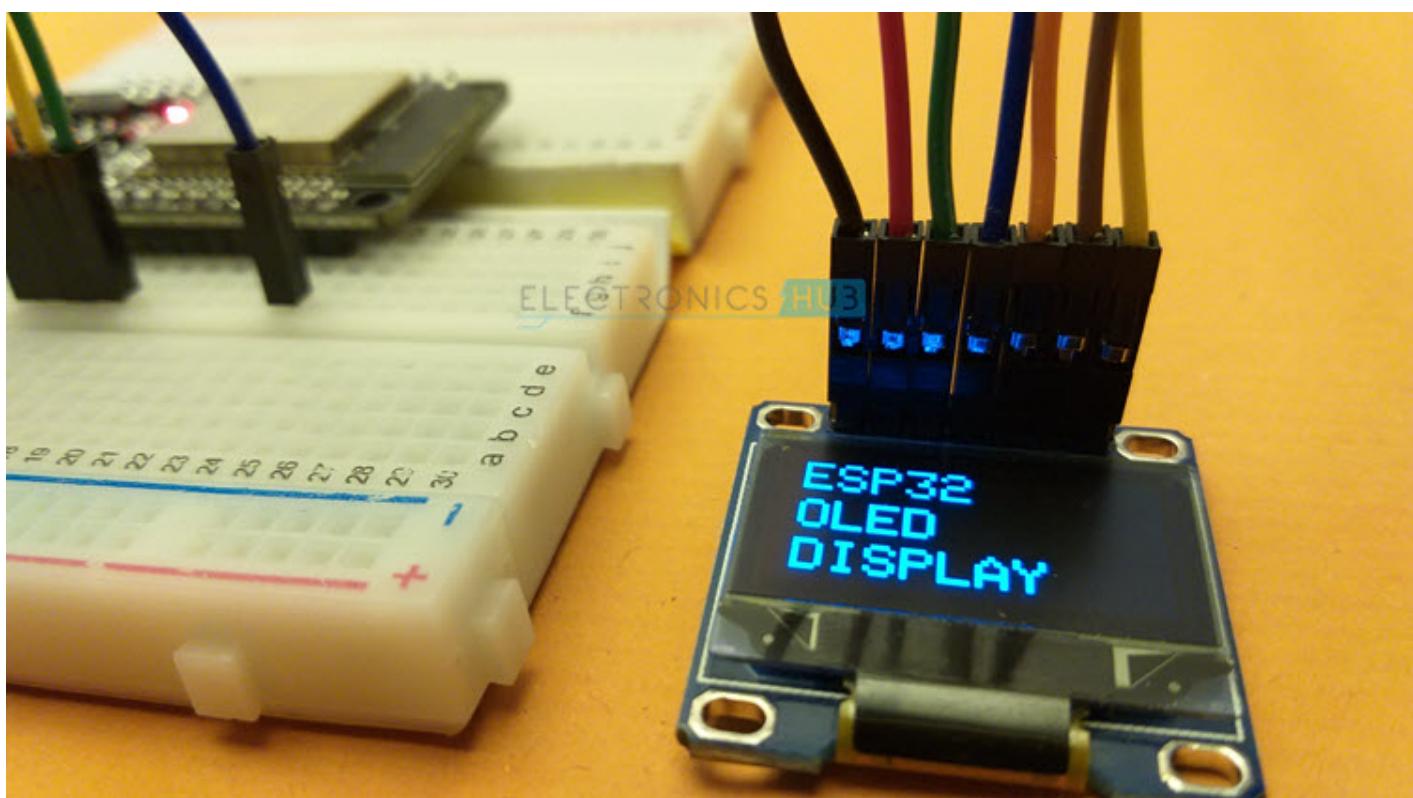


Если вы видите что-то необычное, просто сообщите мне.

Как взаимодействовать с OLED дисплеем через ESP32?

В этой инструкции, мы изучим как взаимодействовать с OLED дисплеем через ESP32 плату разработки. Графический OLED дисплей используемый в проекте основан на SSD1306 OLED драйвере IC и взаимодействует через SPI. Его можно использовать для отображения текста, картинки графики и так далее.



Инструкция

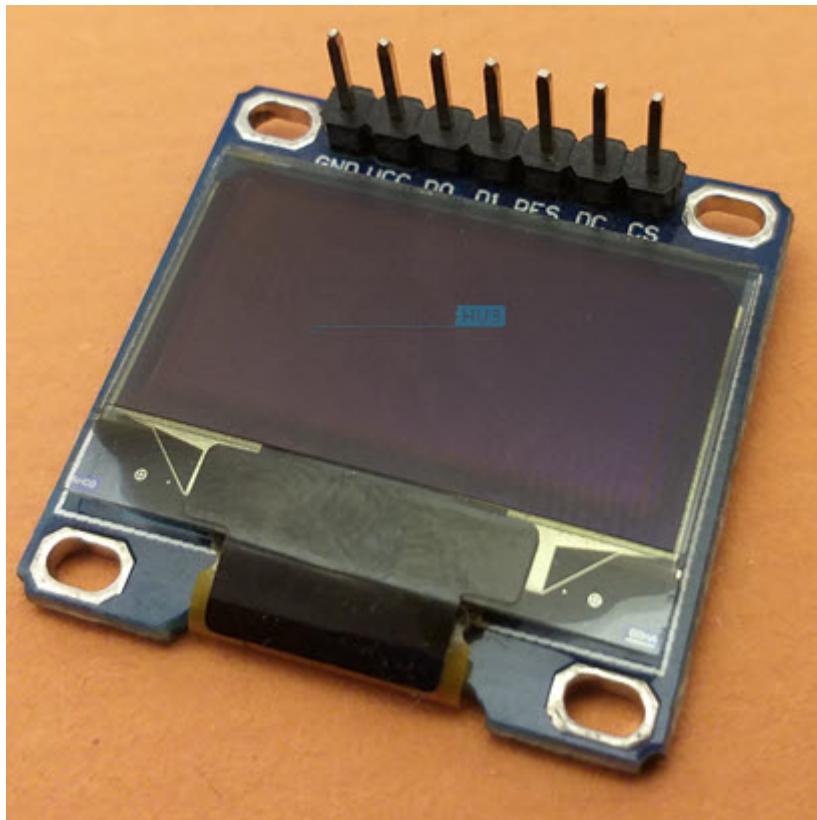
OLED или органический светодиод это улучшенная технология, которая использует пленку с органическим основанием между двумя электродами(анодом и катодом) и когда подается напряжение между электродами, органическая пленка испускает свет.

Главное преимущество OLED дисплея, в том, что она испускает свой собственный свет, и не требует другого источника подсветки. По этой причине OLED дисплеи часто имеют лучший контраст, яркость и углы обзора при сравнении с LCD дисплеями.

Другое важное свойство OLED дисплея это уровень черного цвета. Так как каждый пиксель испускает свой свет в OLED дисплее, чтобы получить черный свет, достаточно отключить пиксель.

Краткое описание SSD1306 OLED дисплея

Хотите использовать OLED дисплеи в ваших DIY проектах? Хотите отображать важную информацию, вроде IP адреса, Адреса Web сервера? Тогда модуль SSD1306 отличный выбор!



Этот модуль состоит из монохромного OLED дисплея с разрешением 128 на 64 пикселя.

Диагональ такого дисплея 0.96. Этот дисплей использует SSD1306 OLED драйвер.

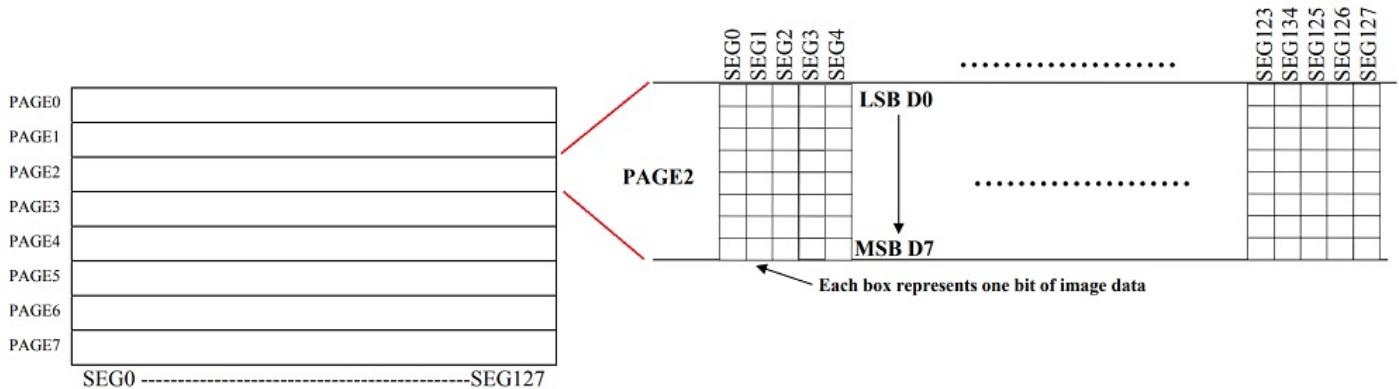
SSD1306 OLED Displays have three types of communication interfaces:

- -bit 6800 Parallel Interface
- 3 or 4 wire SPI
- I2C Of these, the I2C and SPI type OLEDs are very common. It is possible of change the configuration from SPI to I2C and vice-versa (you have solder / de-solder some SMD resistors). The model that I have is using 4-wire SPI Communication.



The SSD1306 OLED Driver IC has 128 x 64 bits Graphic Display Data RAM (GDDRAM). It is divided into eight pages (PAGE 0 to PAGE 7) and each page has 128 Segments. Again, each segment consists of 8-bits and each bit represents one pixel of the display.

So, 8 Pages * 128 Segments * 8 Bits = 8192 Bits (1024 Bytes).

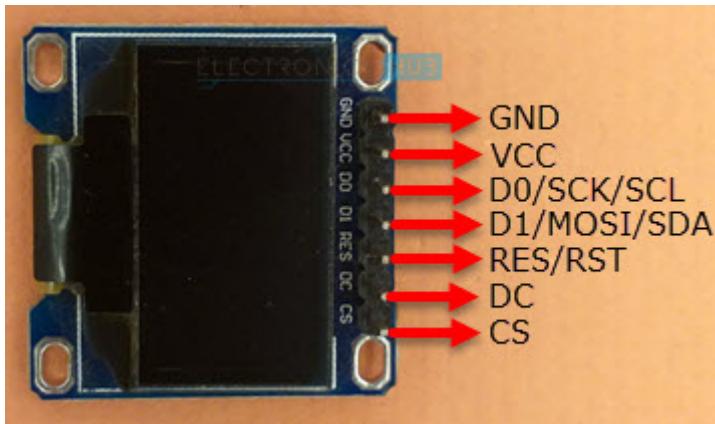


Pinout of OLED Display Module

The following table shows the Pinout of 7-pin SPI based OLED Display Module.

Pin (Alt ern ativ e Na mes)	Des crip tion
GND	Ground
VCC	Power Supply
D0 (SC K, SCL, CLK)	Clock
D1 (MO SI, SDA)	Data
RES (RS T)	Reset

Pin (Alt ern ativ eNa mes)	Des crip tion
DC (A0)	Data / Command Selection
CS	Chip Select



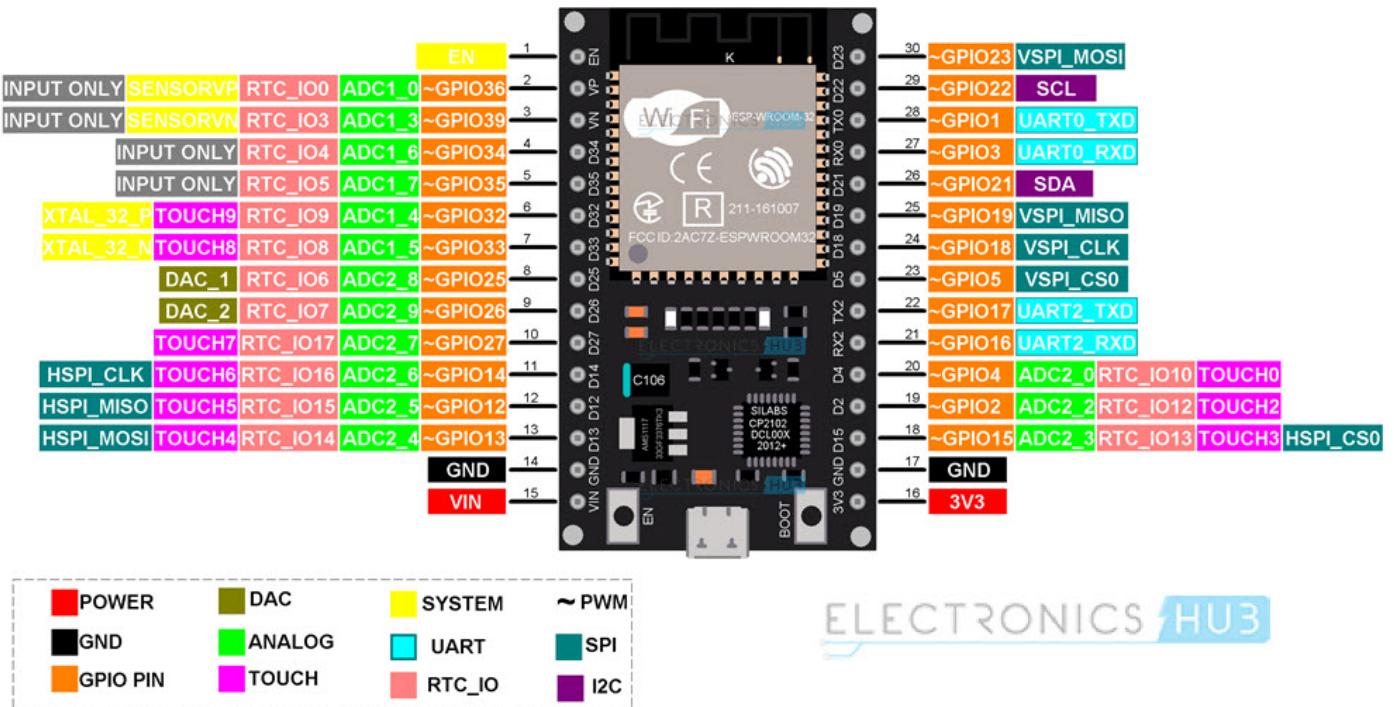
Power Supply

The SSD1306 OLED Driver IC runs on $V_{DD} = 1.65V$ to $3.3V$ and the actual OLED Panel runs on $V_{CC} = 7V$ to $15V$. The OLED Display Module takes care of these wide ranges of voltage requirements with a charge pump circuit (for Panel) and regulator (for Driver IC) from a single power supply (usually between $3V$ and $5V$).

This makes the OLED Display Module to be connected to different boards like Arduino (with $5V$ logic) and ESP32 (with $3.3V$ logic).

ESP32 OLED Display Interface

Let us now see how to interface an OLED Display with ESP32. First thing to understand is that the communication interface is SPI. So, look at the Pinout of ESP32 and identify the SPI Pins.



From the above image, HSPI and VSPI are available on ESP32 Development Board for SPI Interface.

Let us use the VSPI peripheral. The pins for VSPI in ESP32 are:

VSP	GPI
I	O
Pin	Pin
VSPI_MOSI	GPIO23
VSPI_SCLK	GPIO22
VSPI_CS0	GPIO19
VSPI_MISO	GPIO18
VSPI_CLK	GPIO5
VSPI_CS1	GPIO17
VSPI_CS2	GPIO16
VSPI_CS3	GPIO4
VSPI_CS4	GPIO2
VSPI_CS5	GPIO15
VSPI_CS6	GPIO13
VSPI_CS7	GPIO10
VSPI_CS8	GPIO12
VSPI_CS9	GPIO11
VSPI_CS10	GPIO14
VSPI_CS11	GPIO16
VSPI_CS12	GPIO18
VSPI_CS13	GPIO5
VSPI_CS14	GPIO17
VSPI_CS15	GPIO16
VSPI_CS16	GPIO4
VSPI_CS17	GPIO2
VSPI_CS18	GPIO15
VSPI_CS19	GPIO13
VSPI_CS20	GPIO10
VSPI_CS21	GPIO12
VSPI_CS22	GPIO11
VSPI_CS23	GPIO14
VSPI_CS24	GPIO16
VSPI_CS25	GPIO5
VSPI_CS26	GPIO17
VSPI_CS27	GPIO16
VSPI_CS28	GPIO4
VSPI_CS29	GPIO2
VSPI_CS30	GPIO15
VSPI_CS31	GPIO13
VSPI_CS32	GPIO10
VSPI_CS33	GPIO12
VSPI_CS34	GPIO11
VSPI_CS35	GPIO14
VSPI_CS36	GPIO16
VSPI_CS37	GPIO5
VSPI_CS38	GPIO17
VSPI_CS39	GPIO16
VSPI_CS40	GPIO4
VSPI_CS41	GPIO2
VSPI_CS42	GPIO15
VSPI_CS43	GPIO13
VSPI_CS44	GPIO10
VSPI_CS45	GPIO12
VSPI_CS46	GPIO11
VSPI_CS47	GPIO14
VSPI_CS48	GPIO16
VSPI_CS49	GPIO5
VSPI_CS50	GPIO17
VSPI_CS51	GPIO16
VSPI_CS52	GPIO4
VSPI_CS53	GPIO2
VSPI_CS54	GPIO15
VSPI_CS55	GPIO13
VSPI_CS56	GPIO10
VSPI_CS57	GPIO12
VSPI_CS58	GPIO11
VSPI_CS59	GPIO14
VSPI_CS60	GPIO16
VSPI_CS61	GPIO5
VSPI_CS62	GPIO17
VSPI_CS63	GPIO16
VSPI_CS64	GPIO4
VSPI_CS65	GPIO2
VSPI_CS66	GPIO15
VSPI_CS67	GPIO13
VSPI_CS68	GPIO10
VSPI_CS69	GPIO12
VSPI_CS70	GPIO11
VSPI_CS71	GPIO14
VSPI_CS72	GPIO16
VSPI_CS73	GPIO5
VSPI_CS74	GPIO17
VSPI_CS75	GPIO16
VSPI_CS76	GPIO4
VSPI_CS77	GPIO2
VSPI_CS78	GPIO15
VSPI_CS79	GPIO13
VSPI_CS80	GPIO10
VSPI_CS81	GPIO12
VSPI_CS82	GPIO11
VSPI_CS83	GPIO14
VSPI_CS84	GPIO16
VSPI_CS85	GPIO5
VSPI_CS86	GPIO17
VSPI_CS87	GPIO16
VSPI_CS88	GPIO4
VSPI_CS89	GPIO2
VSPI_CS90	GPIO15
VSPI_CS91	GPIO13
VSPI_CS92	GPIO10
VSPI_CS93	GPIO12
VSPI_CS94	GPIO11
VSPI_CS95	GPIO14
VSPI_CS96	GPIO16
VSPI_CS97	GPIO5
VSPI_CS98	GPIO17
VSPI_CS99	GPIO16
VSPI_CS100	GPIO4
VSPI_CS101	GPIO2
VSPI_CS102	GPIO15
VSPI_CS103	GPIO13
VSPI_CS104	GPIO10
VSPI_CS105	GPIO12
VSPI_CS106	GPIO11
VSPI_CS107	GPIO14
VSPI_CS108	GPIO16
VSPI_CS109	GPIO5
VSPI_CS110	GPIO17
VSPI_CS111	GPIO16
VSPI_CS112	GPIO4
VSPI_CS113	GPIO2
VSPI_CS114	GPIO15
VSPI_CS115	GPIO13
VSPI_CS116	GPIO10
VSPI_CS117	GPIO12
VSPI_CS118	GPIO11
VSPI_CS119	GPIO14
VSPI_CS120	GPIO16
VSPI_CS121	GPIO5
VSPI_CS122	GPIO17
VSPI_CS123	GPIO16
VSPI_CS124	GPIO4
VSPI_CS125	GPIO2
VSPI_CS126	GPIO15
VSPI_CS127	GPIO13
VSPI_CS128	GPIO10
VSPI_CS129	GPIO12
VSPI_CS130	GPIO11
VSPI_CS131	GPIO14
VSPI_CS132	GPIO16
VSPI_CS133	GPIO5
VSPI_CS134	GPIO17
VSPI_CS135	GPIO16
VSPI_CS136	GPIO4
VSPI_CS137	GPIO2
VSPI_CS138	GPIO15
VSPI_CS139	GPIO13
VSPI_CS140	GPIO10
VSPI_CS141	GPIO12
VSPI_CS142	GPIO11
VSPI_CS143	GPIO14
VSPI_CS144	GPIO16
VSPI_CS145	GPIO5
VSPI_CS146	GPIO17
VSPI_CS147	GPIO16
VSPI_CS148	GPIO4
VSPI_CS149	GPIO2
VSPI_CS150	GPIO15
VSPI_CS151	GPIO13
VSPI_CS152	GPIO10
VSPI_CS153	GPIO12
VSPI_CS154	GPIO11
VSPI_CS155	GPIO14
VSPI_CS156	GPIO16
VSPI_CS157	GPIO5
VSPI_CS158	GPIO17
VSPI_CS159	GPIO16
VSPI_CS160	GPIO4
VSPI_CS161	GPIO2
VSPI_CS162	GPIO15
VSPI_CS163	GPIO13
VSPI_CS164	GPIO10
VSPI_CS165	GPIO12
VSPI_CS166	GPIO11
VSPI_CS167	GPIO14
VSPI_CS168	GPIO16
VSPI_CS169	GPIO5
VSPI_CS170	GPIO17
VSPI_CS171	GPIO16
VSPI_CS172	GPIO4
VSPI_CS173	GPIO2
VSPI_CS174	GPIO15
VSPI_CS175	GPIO13
VSPI_CS176	GPIO10
VSPI_CS177	GPIO12
VSPI_CS178	GPIO11
VSPI_CS179	GPIO14
VSPI_CS180	GPIO16
VSPI_CS181	GPIO5
VSPI_CS182	GPIO17
VSPI_CS183	GPIO16
VSPI_CS184	GPIO4
VSPI_CS185	GPIO2
VSPI_CS186	GPIO15
VSPI_CS187	GPIO13
VSPI_CS188	GPIO10
VSPI_CS189	GPIO12
VSPI_CS190	GPIO11
VSPI_CS191	GPIO14
VSPI_CS192	GPIO16
VSPI_CS193	GPIO5
VSPI_CS194	GPIO17
VSPI_CS195	GPIO16
VSPI_CS196	GPIO4
VSPI_CS197	GPIO2
VSPI_CS198	GPIO15
VSPI_CS199	GPIO13
VSPI_CS200	GPIO10
VSPI_CS201	GPIO12
VSPI_CS202	GPIO11
VSPI_CS203	GPIO14
VSPI_CS204	GPIO16
VSPI_CS205	GPIO5
VSPI_CS206	GPIO17
VSPI_CS207	GPIO16
VSPI_CS208	GPIO4
VSPI_CS209	GPIO2
VSPI_CS210	GPIO15
VSPI_CS211	GPIO13
VSPI_CS212	GPIO10
VSPI_CS213	GPIO12
VSPI_CS214	GPIO11
VSPI_CS215	GPIO14
VSPI_CS216	GPIO16
VSPI_CS217	GPIO5
VSPI_CS218	GPIO17
VSPI_CS219	GPIO16
VSPI_CS220	GPIO4
VSPI_CS221	GPIO2
VSPI_CS222	GPIO15
VSPI_CS223	GPIO13
VSPI_CS224	GPIO10
VSPI_CS225	GPIO12
VSPI_CS226	GPIO11
VSPI_CS227	GPIO14
VSPI_CS228	GPIO16
VSPI_CS229	GPIO5
VSPI_CS230	GPIO17
VSPI_CS231	GPIO16
VSPI_CS232	GPIO4
VSPI_CS233	GPIO2
VSPI_CS234	GPIO15
VSPI_CS235	GPIO13
VSPI_CS236	GPIO10
VSPI_CS237	GPIO12
VSPI_CS238	GPIO11
VSPI_CS239	GPIO14
VSPI_CS240	GPIO16
VSPI_CS241	GPIO5
VSPI_CS242	GPIO17
VSPI_CS243	GPIO16
VSPI_CS244	GPIO4
VSPI_CS245	GPIO2
VSPI_CS246	GPIO15
VSPI_CS247	GPIO13
VSPI_CS248	GPIO10
VSPI_CS249	GPIO12
VSPI_CS250	GPIO11
VSPI_CS251	GPIO14
VSPI_CS252	GPIO16
VSPI_CS253	GPIO5
VSPI_CS254	GPIO17
VSPI_CS255	GPIO16
VSPI_CS256	GPIO4
VSPI_CS257	GPIO2
VSPI_CS258	GPIO15
VSPI_CS259	GPIO13
VSPI_CS260	GPIO10
VSPI_CS261	GPIO12
VSPI_CS262	GPIO11
VSPI_CS263	GPIO14
VSPI_CS264	GPIO16
VSPI_CS265	GPIO5
VSPI_CS266	GPIO17
VSPI_CS267	GPIO16
VSPI_CS268	GPIO4
VSPI_CS269	GPIO2
VSPI_CS270	GPIO15
VSPI_CS271	GPIO13
VSPI_CS272	GPIO10
VSPI_CS273	GPIO12
VSPI_CS274	GPIO11
VSPI_CS275	GPIO14
VSPI_CS276	GPIO16
VSPI_CS277	GPIO5
VSPI_CS278	GPIO17
VSPI_CS279	GPIO16
VSPI_CS280	GPIO4
VSPI_CS281	GPIO2
VSPI_CS282	GPIO15
VSPI_CS283	GPIO13
VSPI_CS284	GPIO10
VSPI_CS285	GPIO12
VSPI_CS286	GPIO11
VSPI_CS287	GPIO14
VSPI_CS288	GPIO16
VSPI_CS289	GPIO5
VSPI_CS290	GPIO17
VSPI_CS291	GPIO16
VSPI_CS292	GPIO4
VSPI_CS293	GPIO2
VSPI_CS294	GPIO15
VSPI_CS295	GPIO13
VSPI_CS296	GPIO10
VSPI_CS297	GPIO12
VSPI_CS298	GPIO11
VSPI_CS299	GPIO14
VSPI_CS300	GPIO16
VSPI_CS301	GPIO5
VSPI_CS302	GPIO17
VSPI_CS303	GPIO16
VSPI_CS304	GPIO4
VSPI_CS305	GPIO2
VSPI_CS306	GPIO15
VSPI_CS307	GPIO13
VSPI_CS308	GPIO10
VSPI_CS309	GPIO12
VSPI_CS310	GPIO11
VSPI_CS311	GPIO14
VSPI_CS312	GPIO16
VSPI_CS313	GPIO5
VSPI_CS314	GPIO17
VSPI_CS315	GPIO16
VSPI_CS316	GPIO4
VSPI_CS317	GPIO2
VSPI_CS318	GPIO15
VSPI_CS319	GPIO13
VSPI_CS320	GPIO10
VSPI_CS321	GPIO12
VSPI_CS322	GPIO11
VSPI_CS323	GPIO14
VSPI_CS324	GPIO16
VSPI_CS325	GPIO5
VSPI_CS326	GPIO17
VSPI_CS327	GPIO16
VSPI_CS328	GPIO4
VSPI_CS329	GPIO2
VSPI_CS330	GPIO15
VSPI_CS331	GPIO13
VSPI_CS332	GPIO10
VSPI_CS333	GPIO12
VSPI_CS334	GPIO11
VSPI_CS335	GPIO14
VSPI_CS336	GPIO16
VSPI_CS337	GPIO5
VSPI_CS338	GPIO17
VSPI_CS339	GPIO16
VSPI_CS340	GPIO4
VSPI_CS341	GPIO2
VSPI_CS342	GPIO15
VSPI_CS343	GPIO13
VSPI_CS344	GPIO10
VSPI_CS345	GPIO12
VSPI_CS346	GPIO11
VSPI_CS347	GPIO14
VSPI_CS348	GPIO16
VSPI_CS349	GPIO5
VSPI_CS350	GPIO17
VSPI_CS351	GPIO16
VSPI_CS352	GPIO4
VSPI_CS353	GPIO2
VSPI_CS354	GPIO15
VSPI_CS355	GPIO13
VSPI_CS356	GPIO10
VSPI_CS357	GPIO12
VSPI_CS358	GPIO11
VSPI_CS359	GPIO14
VSPI_CS360	GPIO16
VSPI_CS361	GPIO5
VSPI_CS362	GPIO17
VSPI_CS363	GPIO16
VSPI_CS364	GPIO4
VSPI_CS365	GPIO2
VSPI_CS366	GPIO15
VSPI_CS367	GPIO13
VSPI_CS368	GPIO10
VSPI_CS369	GPIO12
VSPI_CS370	GPIO11
VSPI_CS371	GPIO14
VSPI_CS372	GPIO16
VSPI_CS373	GPIO5
VSPI_CS374	GPIO17
VSPI_CS375	GPIO16
VSPI_CS376	GPIO4
VSPI_CS377	GPIO2
VSPI_CS378	GPIO15
VSPI_CS379	GPIO13
VSPI_CS380	GPIO10
VSPI_CS381	GPIO12
VSPI_CS382	GPIO11
VSPI_CS383	GPIO14
VSPI_CS384	GPIO16
VSPI_CS385	GPIO5
VSPI_CS386	GPIO17
VSPI_CS387	GPIO16
VSPI_CS388	GPIO4
VSPI_CS389	GPIO2
VSPI_CS390	GPIO15
VSPI_CS391	GPIO13
VSPI_CS392	GPIO10
VSPI_CS393	GPIO12
VSPI_CS394	GPIO11
VSPI_CS395	GPIO14
VSPI_CS396	GPIO16
VSPI_CS397	GPIO5
VSPI_CS398	GPIO17
VSPI_CS399	GPIO16
VSPI_CS400	GPIO4
VSPI_CS401	GPIO2
VSPI_CS402	GPIO15
VSPI_CS403	GPIO13
VSPI_CS404	GPIO10
VSPI_CS405	GPIO12
VSPI_CS406	GPIO11
VSPI_CS407	GPIO14
VSPI_CS408	GPIO16
VSPI_CS409	GPIO5
VSPI_CS410	GPIO17
VSPI_CS411	GPIO16
VSPI_CS412	GPIO4
VSPI_CS413	GPIO2
VSPI_CS414	GPIO15
VSPI_CS415	GPIO13
VSPI_CS416	GPIO10
VSPI_CS417	GPIO12
VSPI_CS418	GPIO11
VSPI_CS419	GPIO14
VSPI_CS420	GPIO16
VSPI_CS421	GPIO5
VSPI_CS422	GPIO17

VSP IPin	GPI O Pin
VSPI _CL K	GPI O 18
VSPI _CS	GPI O 5

NOTE: ESP32 has totally 4 SPI Peripherals. (SPI0, SPI1, HSPI and VSPI). SPI0 is dedicated to SPI Flash IC. SPI1 shares the hardware with SPI0. This leaves HSPI and VSPI for interfacing SPI Devices.

The following table shows the connections between ESP32 and OLED Display Module. In total, we have to make seven connections as this is an SPI OLED Display.

OLE D Dis pla y	ESP 32
GND	GND
VCC	3.3V
D0 (SC K)	GPI O 18
D1 (MO SI)	GPI O 23
RES	GPI O 17
DC	GPI O 16

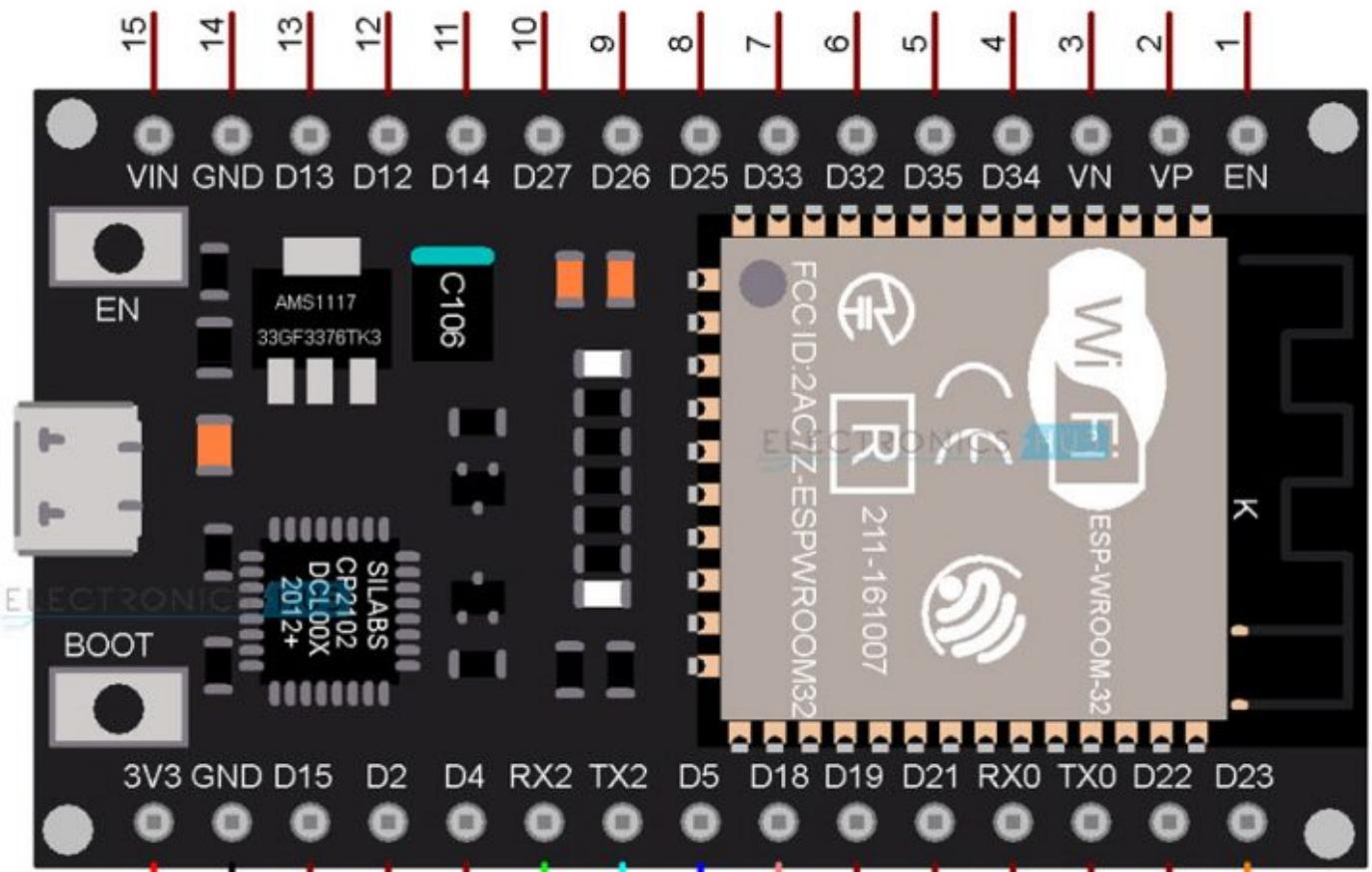
OLE D Dis pla y	ESP 32
CS	GPI O 5

Components Required

- ESP32 DevKit Development Board
- OLED Display Module
- Breadboard
- Connecting Wires
- Micro USB Cable

Circuit Diagram

The following image shows the circuit diagram for Interfacing SPI OLED Display with ESP32.



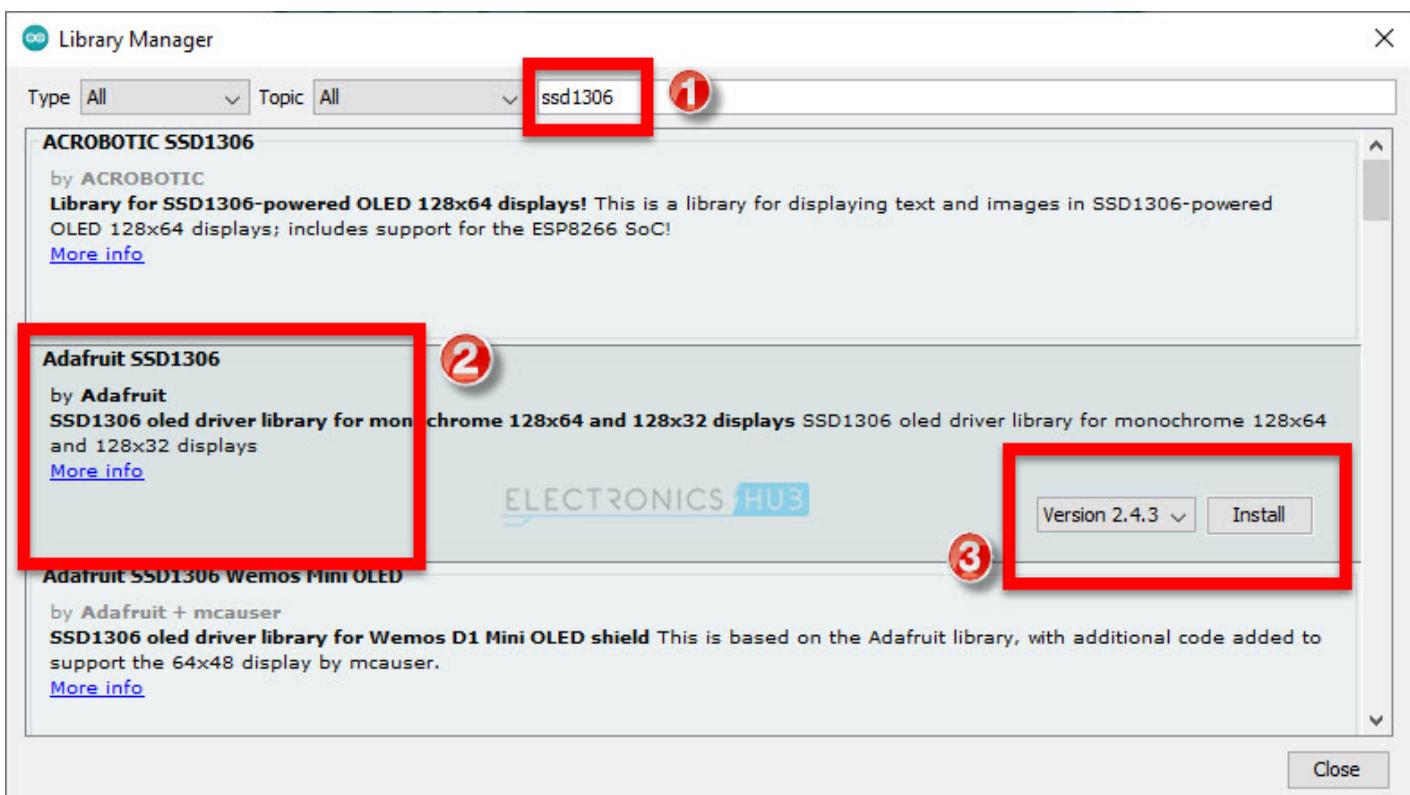
Preparing Arduino IDE

Before writing the code, you need to download some libraries for Arduino IDE related to SSD1306 OLED Display.

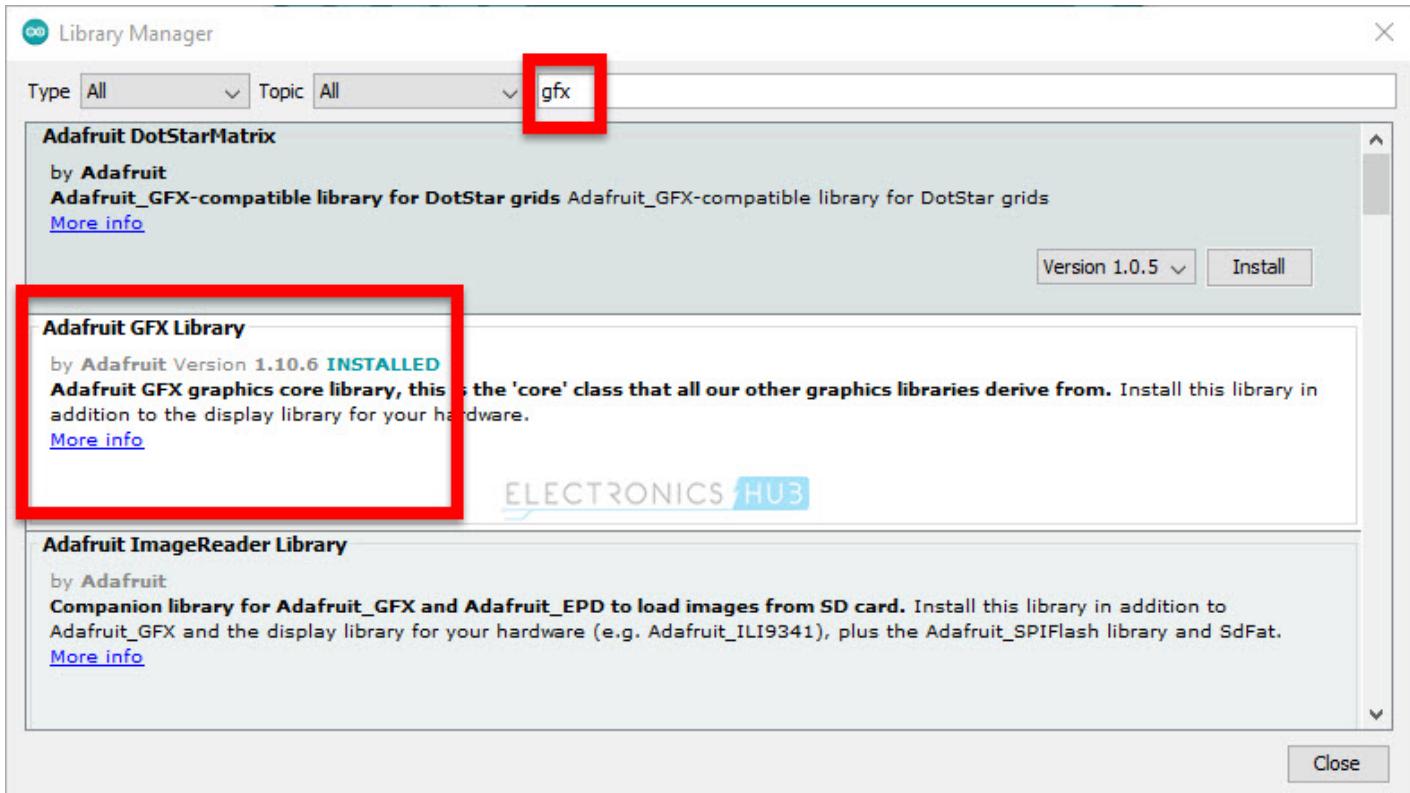
I made a dedicated tutorial on how to install ESP32 Board in Arduino IDE. You can check out that tutorial first. Now, open the Arduino IDE and go to Tools -> Manage Libraries. . .

Arduino-IDE-OLED-Library-1

A Library Manager window will pop-up. In the search bar, type “ssd1306” and from the results select the “Adafruit SSD1306” option and click on install. This library is written specifically for monochrome OLED Displays based on SSD1306 Driver IC. The supported resolutions are 128 x 32 and 128 x 64.



After installing SSD1306 Library, search for “gfx” and install “Adafruit GFX Library”. This is a graphics library by Adafruit for displaying basic graphics like lines, circles, rectangles etc.



Close the library manager window after downloading all the necessary libraries. Now, make sure that ESP32 Board is selected in Arduino IDE (Tools -> Board -> ESP32 Arduino -> ESP32 Dev Module).

Testing ESP32 OLED Display

After making all the necessary connections, we will now proceed to write a test code for ESP32 to display some text and graphics on the OLED Display. In this code, I am testing various features of the OLED Display like displaying normal text, inverted text, scrolling text, displaying ASCII Characters, setting font size.

I also added the code for displaying graphics like rectangle, filled rectangle, rounded rectangle, filled rounded rectangle, circle, filled circle, triangle and filled triangle.

Finally, I took the “Electronics Hub” logo and converted it into a bitmap and displayed it on the OLED Display.

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



```
};

void setup()
{
    Serial.begin(115200);
    if(!display.begin(SSD1306_SWITCHCAPVCC))
    {
        Serial.println(F("SSD1306 allocation failed"));
        for(;;);
    }

    display.clearDisplay();
    display.display();
    delay(1000);

    display.clearDisplay();
    display.drawBitmap(0, 0, electronicshub_logo, SCREEN_WIDTH, SCREEN_HEIGHT, SSD1306_WHITE);
    display.display();
    delay(1000);

}

void loop()
{
    AllPixels();
    TextDisplay();
    InvertedTextDisplay();
    ScrollText();
    DisplayChars();
    TextSize();
    DrawRectangle();
    DrawFilledRectangle();
    DrawRoundRectangle();
    DrawFilledRoundRectangle();
    DrawCircle();
    DrawFilledCircle();
    DrawTriangle();
    DrawFilledTriangle();
}
```

```
void AllPixels()
{
    int i;
    int j;
    display.clearDisplay();
    for(i=0;i<64;i++)
    {
        for(j=0;j<128;j++)
        {
            display.drawPixel(j, i, SSD1306_WHITE);
        }
        display.display();
        delay(30);
    }

    for(i=0;i<64;i++)
    {
        for(j=0;j<128;j++)
        {
            display.drawPixel(j, i, SSD1306_BLACK);
        }
        display.display();
        delay(30);
    }
}

void TextDisplay()
{
    display.clearDisplay();
    display.setTextSize(1);
    display.setTextColor(SSD1306_WHITE);
    display.setCursor(5,28);
    display.println("Electronics Hub");
    display.display();
    delay(3000);
}
```

```
void InvertedTextDisplay()
{
    display.clearDisplay();
    display.setTextColor(SSD1306_BLACK, SSD1306_WHITE);
    display.setCursor(5,28);
    display.println("Electronics Hub");
    display.display();
    delay(3000);
}
```

```
void ScrollText()
{
    display.clearDisplay();
    display.setCursor(0,0);
    display.setTextSize(1);
    display.setTextColor(SSD1306_WHITE);
    display.println("This is a");
    display.println("Scrolling");
    display.println("Text!");
    display.display();
    delay(100);
    display.startscrollright(0x00, 0x0F);
    delay(2000);
    //display.stopscroll();
    //delay(1000);
    display.startscrollleft(0x00, 0x0F);
    delay(2000);
    //display.stopscroll();
    //delay(1000);
    display.startscrolldiagright(0x00, 0x0F);
    delay(2000);
    display.startscrolldiagleft(0x00, 0x0F);
    delay(2000);
    display.stopscroll();
}
```

```
void DisplayChars()
{
    display.clearDisplay();
```

```
display.setTextSize(1);
display.setTextColor(SSD1306_WHITE);
display.setCursor(0, 0);
display.cp437(true);

for(int16_t i=0; i<256; i++)
{
    if(i == '\n')
    {
        display.write(' ');
    }
    else
    {
        display.write(i);
    }
}

display.display();
delay(4000);
}

void TextSize()
{
    display.clearDisplay();

    display.setTextSize(1);
    display.setTextColor(SSD1306_WHITE);
    display.setCursor(0,0);
    display.println(F("Size: 1"));
    display.println(F("ABC"));

    display.setTextSize(2);
    display.setTextColor(SSD1306_WHITE);
    display.println("Size: 2");
    display.println(F("ABC"));

    display.display();
    delay(3000);
}

void DrawRectangle()
```

```
{  
    display.clearDisplay();  
    display.setTextSize(1);  
    display.setTextColor(WHITE);  
    display.setCursor(0,0);  
    display.println("Rectangle");  
    display.drawRect(0, 15, 90, 45, SSD1306_WHITE);  
    display.display();  
    delay(2000);  
}  
  
void DrawFilledRectangle()  
{  
    display.clearDisplay();  
    display.setTextSize(1);  
    display.setTextColor(WHITE);  
    display.setCursor(0,0);  
    display.println("Filled Rectangle");  
    display.fillRect(0, 15, 90, 45, SSD1306_WHITE);  
    display.display();  
    delay(2000);  
}
```

```
void DrawRoundRectangle()  
{  
    display.clearDisplay();  
    display.setTextSize(1);  
    display.setTextColor(SSD1306_WHITE);  
    display.setCursor(0,0);  
    display.println("Round Rectangle");  
    display.drawRoundRect(0, 15, 90, 45, 10, SSD1306_WHITE);  
    display.display();  
    delay(2000);  
}
```

```
void DrawFilledRoundRectangle()  
{  
    display.clearDisplay();  
    display.setTextSize(1);
```

```
display.setTextColor(SSD1306_WHITE);
display.setCursor(0,0);
display.println("Filled Round Rect");
display.fillRoundRect(0, 15, 90, 45, 10, SSD1306_WHITE);
display.display();
delay(2000);

}
```

```
void DrawCircle()
{
    display.clearDisplay();
    display.setTextSize(1);
    display.setTextColor(SSD1306_WHITE);
    display.setCursor(0,0);
    display.println("Circle");
    display.drawCircle(30, 36, 25, SSD1306_WHITE);
    display.display();
    delay(2000);
}
```

```
void DrawFilledCircle()
{
    display.clearDisplay();
    display.setTextSize(1);
    display.setTextColor(SSD1306_WHITE);
    display.setCursor(0,0);
    display.println("Filled Circle");
    display.fillCircle(30, 36, 25, SSD1306_WHITE);
    display.display();
    delay(2000);
}
```

```
}
```

```
void DrawTriangle()
{
    display.clearDisplay();
    display.setTextSize(1);
    display.setTextColor(SSD1306_WHITE);
    display.setCursor(0,0);
    display.println("Triangle");
}
```

```
display.drawTriangle(30, 15, 0, 60, 60, 60, SSD1306_WHITE);
display.display();
delay(2000);
}

void DrawFilledTriangle()
{
display.clearDisplay();
display.setTextSize(1);
display.setTextColor(SSD1306_WHITE);
display.setCursor(0,0);
display.println("Filled Triangle");
display.fillTriangle(30, 15, 0, 60, 60, 60, SSD1306_WHITE);
display.display();
delay(2000);
}
```

Conclusion

A simple tutorial on how to interface SPI OLED Display Module with ESP32 DevKit Board. You learned the pinout of SSD1306 OLED Display, necessary connections for SPI Interface with ESP32, download libraries for Arduino IDE and display some text, graphics and image on the OLED Display using ESP32.

Revision #3

Created 7 April 2023 18:35:45 by gasick

Updated 1 September 2023 06:49:40 by gasick