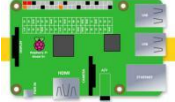


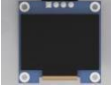


Lesson 16 OLED Display

16.1 Overview

This lesson focuses on teaching you how to use an OLED screen with a Raspberry Pi and an Adeept Robot HAT V3.2. You will learn about the required components, the connection principles, how to wire the components together, and how to run programs to display different content on the OLED screen, such as the current date and time and a snow animation effect.

16.2 Required Components

Components	Quantity	Picture
Raspberry Pi	1	
Adeept Robot HAT V3.2	1	
4 pin wire	1	
OLED Screen	1	

16.3 Principle Introduction

The OLED (Organic Light - Emitting Diode) screen used in this project communicates with the Raspberry Pi through the I2C (Inter - Integrated Circuit) protocol. The I2C protocol is a multi - master, multi - slave, single - ended, serial computer bus standard. It uses two wires: SDA (Serial

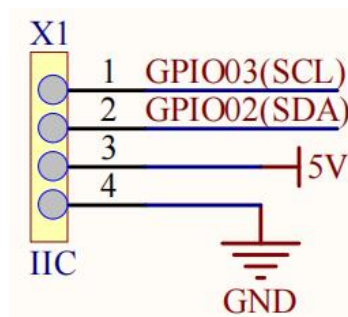
Data Line) for data transfer and SCL (Serial Clock Line) for synchronizing the data transfer. The Adeept Robot HAT V3.2 acts as an interface between the Raspberry Pi and the OLED screen, translating the signals from the Raspberry Pi into a format that the OLED screen can understand.

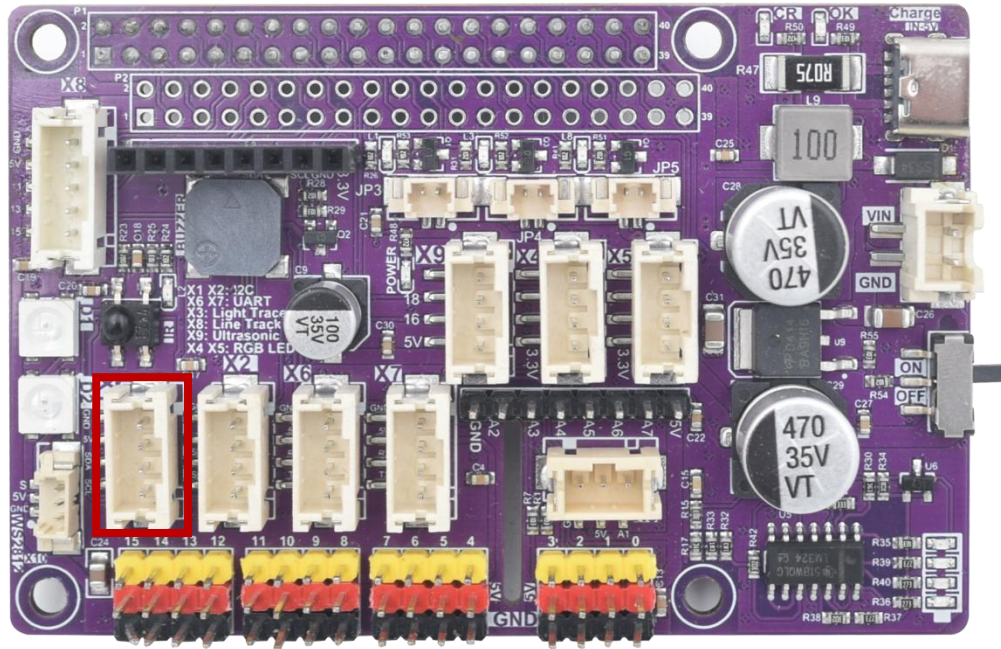
When running programs on the Raspberry Pi, the code sends commands and data to the OLED screen via the I2C interface on the Adeept Robot HAT V3.2. For example, when displaying the date and time, the program retrieves the system date and time information from the Raspberry Pi's operating system and then formats and sends it to the OLED screen for display. When simulating the snow animation, the program generates a sequence of pixel - level data representing the snowflakes' movement and sends it to the screen.

PINS of Raspberry Pi	OLED
GPIO03(SCL)	SCL
GPIO02(SDA)	SDA
VCC	VCC
GND	GND

14.4 Wiring Diagram

If you want to use the OLED Screen module, you need to connect the IIC interface on the Adeept Robot HAT V3.2 driver board, as shown in the figure below:





16.5 Demonstration

Run the code

1. **Remotely log:** Remotely log in to the Raspberry Pi terminal.
2. **Navigate to the Program Folder:** Enter the following command in the terminal and press Enter to access the folder where the program is located:

```
cd Adeept_4WD_Smart_Car_for_RPi/Examples/10_OLED/
```

```
pi@raspberrypi:~ $ cd Adeept_4WD_Smart_Car_for_RPi/Examples/10_OLED/  
pi@raspberrypi:~/Adeept_4WD_Smart_Car_for_RPi/Examples/10_OLED $
```

3. **View Directory Contents:** Type "ls" in the terminal and press Enter. This will display all the files in the current directory, ensuring that the "TimeOled.py" and "SnowOled.py" file is present:

```
ls
```

```
pi@raspberrypi:~/Adeept_4WD_Smart_Car_for_RPi/Examples/10_OLED $ ls
SnowOled.py  TimeOled.py
```

4. **Run the Program:**Demonstrate how to use an OLED screen to display the current date and time in real time.

```
sudo python3 TimeOled.py
```

```
pi@raspberrypi:~/Adeept_4WD_Smart_Car_for_RPi/Examples/10_OLED $ sudo python3 TimeOled.py
/home/pi/Adeept_4WD_Smart_Car_for_RPi/Examples/10_OLED/TimeOled.py:39: DeprecationWarning: getsize is deprecated and will be removed in Pillow 10 (2023-07-01). Use getbbox or getlength instead.
```

5. **Observation and Termination:**After successfully running the program, the OLED screen displays the current date and time of the Raspberry Pi in real-time. When you want to terminate a running program, you can press the **Ctrl+C** shortcut key on the keyboard.

6. **Run the Program:**How to simulate the animation effect of snow on an OLED screen.

```
sudo python3 SnowOled.py
```

```
pi@raspberrypi:~/Adeept_4WD_Smart_Car_for_RPi/Examples/10_OLED $ sudo python3 SnowOled.py
```

7. **Observation and Termination:**After successfully running the program, simulate the animation effect of snow on the OLED screen. When you want to terminate a running program, you can press the **Ctrl+C** shortcut key on the keyboard.

16.6 Code

Complete code refer to [TimeOled.py](#)

```
01  #!/usr/bin/env/python
02  # File name   : TimeOled.py
03  # Website    : www.Adeept.com
04  # Author     : Adeept
05  # Date      : 2025/03/7
06  import board
07  import busio
08  import adafruit_ssd1306
09  from PIL import Image, ImageDraw, ImageFont
10  import datetime
11  import time
```

```
12
13 # Create an I2C object
14 i2c = busio.I2C(board.SCL, board.SDA)
15
16 # Create an SSD1306 OLED device object with a screen resolution of 128x64
17 oled = adafruit_ssd1306.SSD1306_I2C(128, 64, i2c)
18
19 # Load the font
20 font = ImageFont.load_default()
21
22 def draw_text_with_wrap(draw, text, x, y, font, fill, max_width):
23     lines = []
24     current_line = ""
25     for word in text.split():
26         test_line = current_line + word + " "
27         # Use the textlength method to get the text width
28         test_width = draw.textlength(test_line, font=font)
29         if test_width <= max_width:
30             current_line = test_line
31         else:
32             lines.append(current_line.rstrip())
33             current_line = word + " "
34     if current_line:
35         lines.append(current_line.rstrip())
36
37     for line in lines:
38         draw.text((x, y), line, font=font, fill=fill)
39         y += font.getsize(line)[1]
40
41 try:
42     while True:
43         # Clear the screen
44         oled.fill(0)
45
46         # Create a blank image
47         image = Image.new('1', (oled.width, oled.height))
48
49         # Create a drawing object
50         draw = ImageDraw.Draw(image)
51
52         # Get the current time
53         now = datetime.datetime.now()
54         current_time = now.strftime("%Y-%m-%d %H:%M:%S")
55         display_text = f"Time: {current_time}"
56
57         # Draw the time on the image with word wrap support
58         draw_text_with_wrap(draw, display_text, 0, 0, font, 255, oled.width)
59
60         # Display the image on the OLED screen
61         oled.image(image)
62         oled.show()
63
64         # Pause for 1 second
65         time.sleep(1)
66
67 except KeyboardInterrupt:
```

```
68 # Clear the screen
69 oled.fill(0)
70 oled.show()
```

Complete code refer to [SnowOled.py](#)

```
01 #!/usr/bin/env/python
02 # File name : SnowOled.py
03 # Website : www.Adeept.com
04 # Author : Adeept
05 # Date : 2025/03/7
06 import board
07 import busio
08 import adafruit_ssd1306
09 from PIL import Image, ImageDraw
10 import random
11 import time
12
13 # Create an I2C object
14 i2c = busio.I2C(board.SCL, board.SDA)
15
16 # Create an SSD1306 OLED device object with a screen resolution of 128x64
17 oled = adafruit_ssd1306.SSD1306_I2C(128, 64, i2c)
18
19 # Define the Snowflake class
20 class Snowflake:
21     def __init__(self, x, y, speed):
22         self.x = x
23         self.y = y
24         self.speed = speed
25
26     def fall(self):
27         self.y += self.speed
28         if self.y > oled.height:
29             self.y = 0
30             self.x = random.randint(0, oled.width)
31
32     def draw(self, draw):
33         draw.point((self.x, self.y), fill=255)
34
35 # Define the Star class
36 class Star:
37     def __init__(self, x, y):
38         self.x = x
39         self.y = y
40         self.brightness = random.choice([0, 255])
41
42     def twinkle(self):
43         if random.random() < 0.1: # 10% chance to change brightness
44             self.brightness = 255 if self.brightness == 0 else 0
45
46     def draw(self, draw):
47         draw.point((self.x, self.y), fill=self.brightness)
48
49 # Create lists of snowflakes and stars
```

```
50 snowflakes = [Snowflake(random.randint(0, oled.width), random.randint(0, oled.height),
51 random.randint(1, 3)) for _ in range(20)]
52 stars = [Star(random.randint(0, oled.width), random.randint(0, oled.height)) for _ in range(10)]
53
54 try:
55     while True:
56         # Clear the screen
57         oled.fill(0)
58
59         # Create a blank image
60         image = Image.new('1', (oled.width, oled.height))
61
62         # Create a drawing object
63         draw = ImageDraw.Draw(image)
64
65         # Update and draw the snowflakes
66         for snowflake in snowflakes:
67             snowflake.fall()
68             snowflake.draw(draw)
69
70         # Update and draw the stars
71         for star in stars:
72             star.twinkle()
73             star.draw(draw)
74
75         # Display the image on the OLED screen
76         oled.image(image)
77         oled.show()
78
79         # Pause for a while
80         time.sleep(0.1)
81
82 except KeyboardInterrupt:
83     # Clear the screen
84     oled.fill(0)
85     oled.show()
```

Code explanation

TimeOled.py

Initialization Stage:

Create an I2C object for communication with OLED screens. Create an SSD1306 OLED device object with a screen resolution of 128x64. Load default font.

Loop Control Process:

After entering an infinite loop, execute the following steps in sequence:

Stage 1: At the beginning of each cycle, clear the screen.

Stage 2: Create a blank image and drawing object.

Stage 3: Retrieve the current date and time and format it as a string of YYYY-MM-DD HH: MM: SS.

Stage 4: Use the draw_text-with_rap function to draw the time text onto the image.

Stage 5: After pausing for 1 second, cycle to update the time again.

Enter an infinite loop, constantly updating and displaying the time.

[SnowOled.py](#)

Initialization Stage:

Create an I2C object for communication with OLED screens. Create an SSD1306 OLED device object with a screen resolution of 128x64.

Loop Control Process:

After entering an infinite loop, execute the following steps in sequence:

Stage 1: At the beginning of each cycle, clear the screen.

Stage 2: Create a blank image and drawing object.

Stage 3: Traverse the snowflake list, update the position of each snowflake, and draw it onto the image.

Stage 4: Traverse the star list, update the brightness of each star, and draw it onto the image.

Stage 5: After pausing for 0.1 seconds, update the animation in a loop again.

Enter an infinite loop, constantly updating and displaying animations.