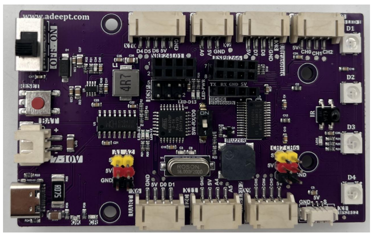
Lesson 13 OLED Display

13.1 Overview

This course mainly introduces the application of OLED display screen on Adeept Robot Control Board (based on Arduino), covering the required components, working principle, hardware connection, code writing and demonstration, to help master the combination of OLED display screen and Arduino, and achieve the display function of information on OLED screen.

13.2 Required Components

Components	Quantity	Picture
Adeept Robot Control Board	1	
Type-C USB Cable	1	
OLED Screen	1	

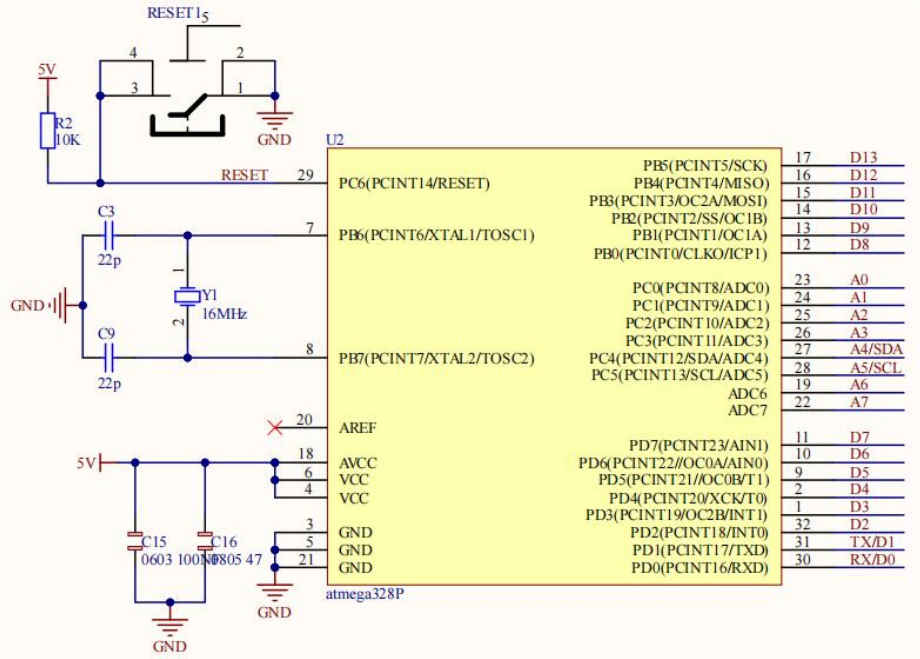
13.3 Principle Introduction

OLED (Organic Light-Emitting Diode), also known as organic electroluminescence display or organic light-emitting semiconductor, belongs to a current-type organic light-emitting device. It emits light through the injection and recombination of carriers, and the luminous intensity is proportional to the injected current, with relatively low power consumption.

Due to their small size, light weight, and low power consumption, OLED screens are increasingly used in embedded electronic devices. There are various types of OLED screens, which display different colors. Commonly used ones include white display, blue display, and yellow-blue two-color display. Additionally, they come in different screen sizes and are equipped with various built-in driver chips. The commonly used driver interfaces are SPI and IIC. This article introduces a 0.96-inch blue IIC-driven screen, and its built-in driver chip is SSD1306.

OLED Screen	Arduino(X11)
SDA	A4
SCL	A5
GND	GND
VCC	5V

13.4 Wiring Diagram



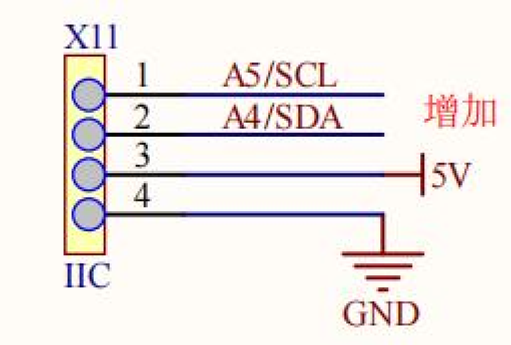
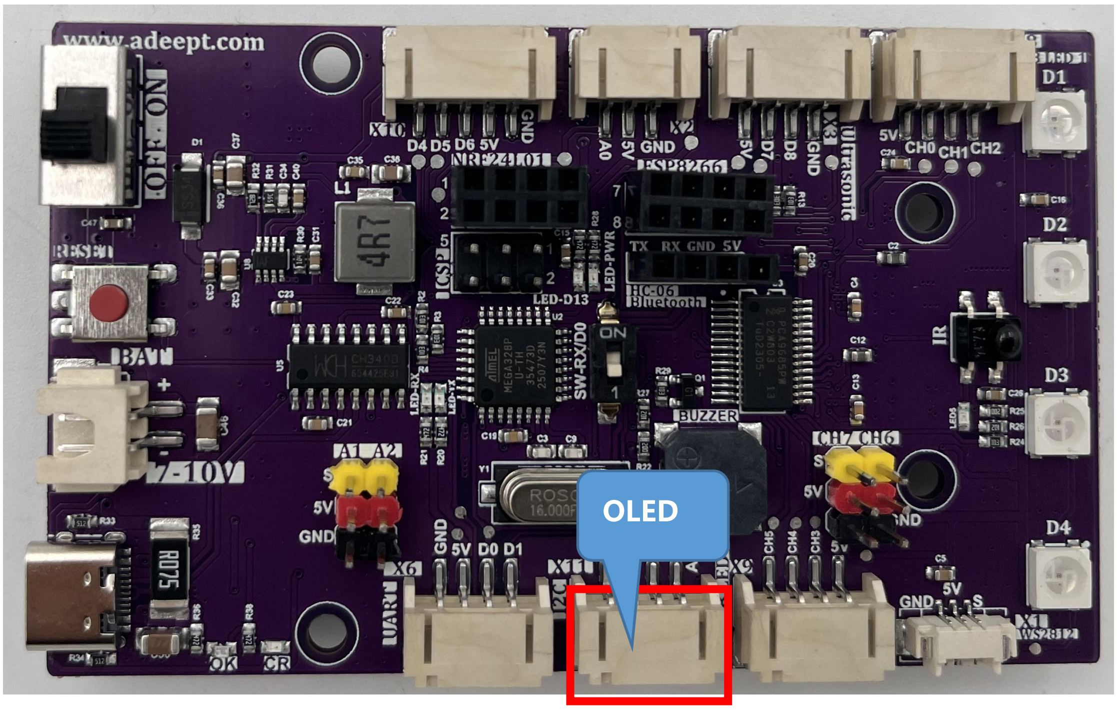
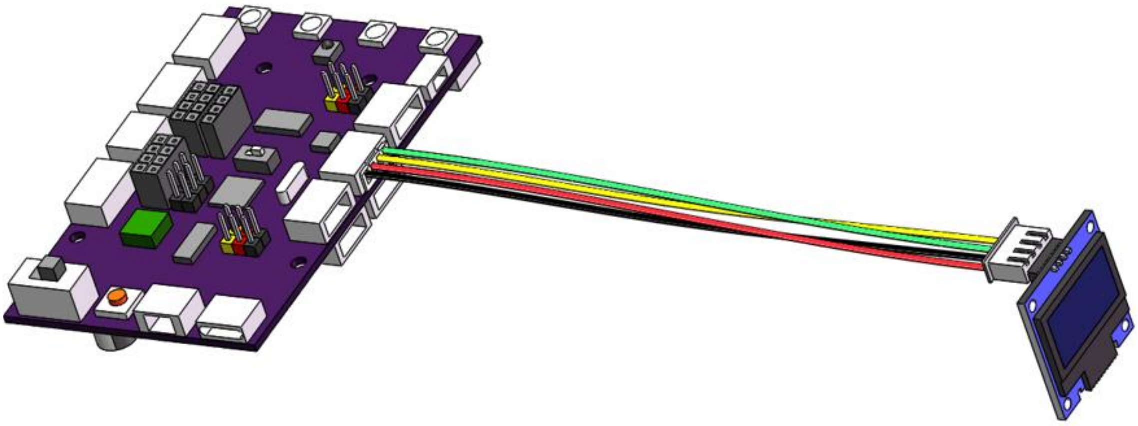


Figure as below：

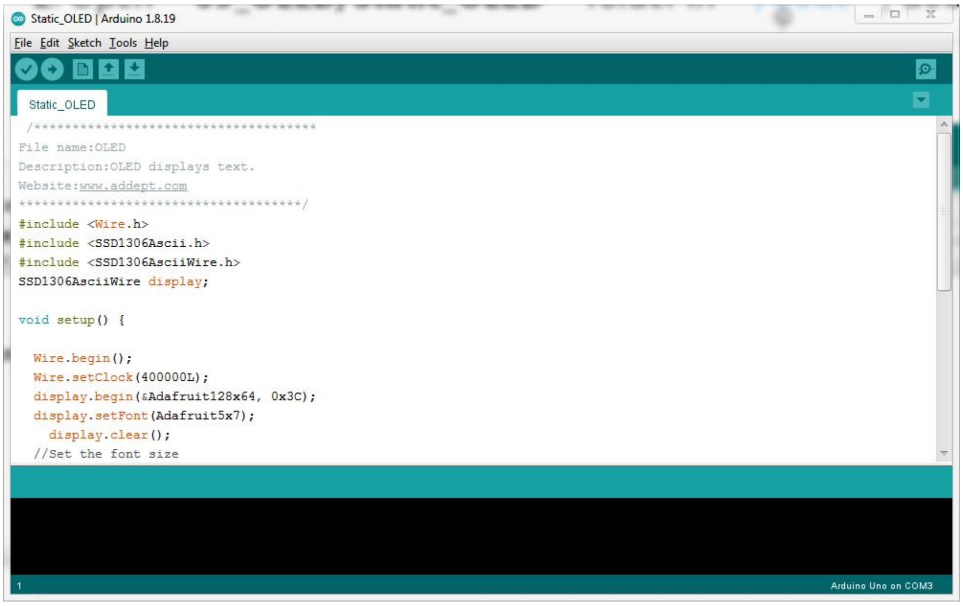




13.5 Demonstration

1. Connect your computer and Adeept Robot Control Board (Arduino Board) with a USB cable.

2. Open “**09_OLED/Static_OLED**” folder in “/Code”, double-click “Static_OLED.ino”.



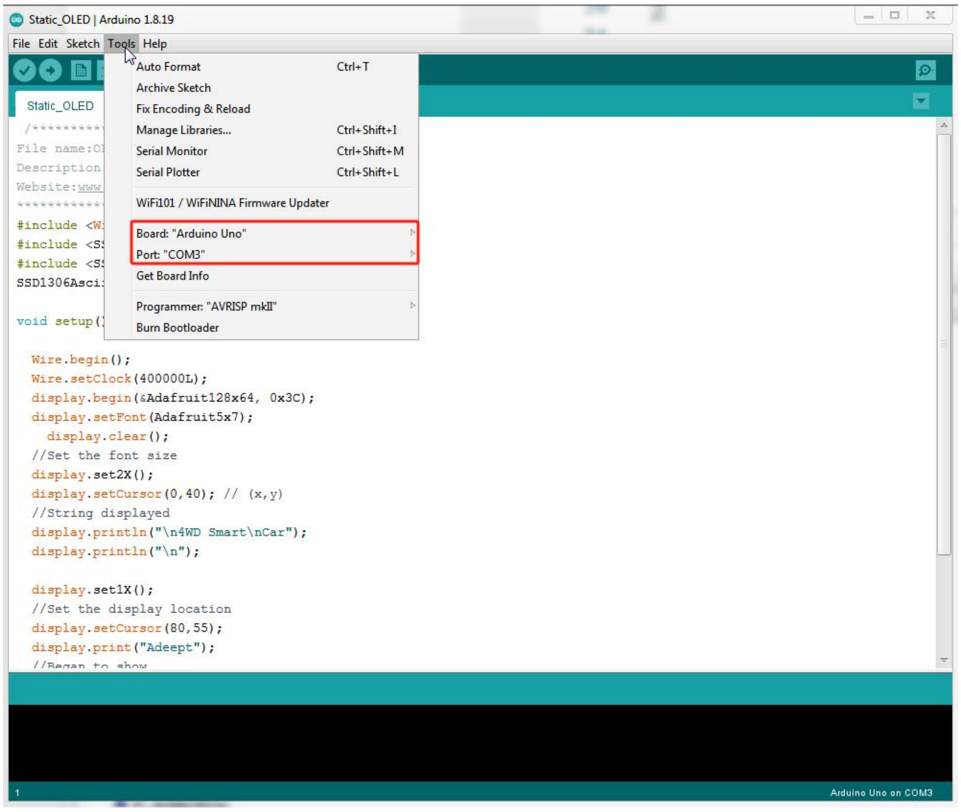
```
 /****************************************
File name:OLED
Description:OLED displays text.
Website:www.addept.com
****************************************/
#include <Wire.h>
#include <SSD1306Ascii.h>
#include <SSD1306AsciiWire.h>
SSD1306AsciiWire display;

void setup() {

  Wire.begin();
  Wire.setClock(400000L);
  display.begin(&Adafruit128x64, 0x3C);
  display.setFont(Adafruit5x7);
    display.clear();
  //Set the font size
```

3. Select development board and serial port.

Board: Tools--->Board--->Arduino AVR Boards--->Arduino Uno

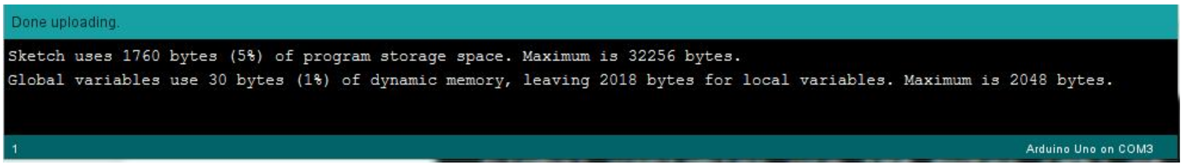
Port: Tools --->Port--->COMx

Note: The port number will be different in different computers.


Static_OLED | Arduino 1.8.19
File Edit Sketch Tools Help
Static_OLED
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Uno"
Port: "COM3"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
1
Arduino Uno on COM3

```
  Wire.begin();
  Wire.setClock(400000L);
  display.begin(&Adafruit128x64, 0x3C);
  display.setFont(Adafruit5x7);
    display.clear();
  //Set the font size
  display.set2X();
  display.setCursor(0,40); // (x,y)
  //String displayed
  display.println("\n4WD Smart\nCar");
  display.println("\n");

  display.set1X();
  //Set the display location
  display.setCursor(80,55);
  display.print("Adeept");
  //Began to show
```

4. After opening, click → to upload the code program to the Arduino. If there is no error warning in the console below, it means that the Upload is successful.



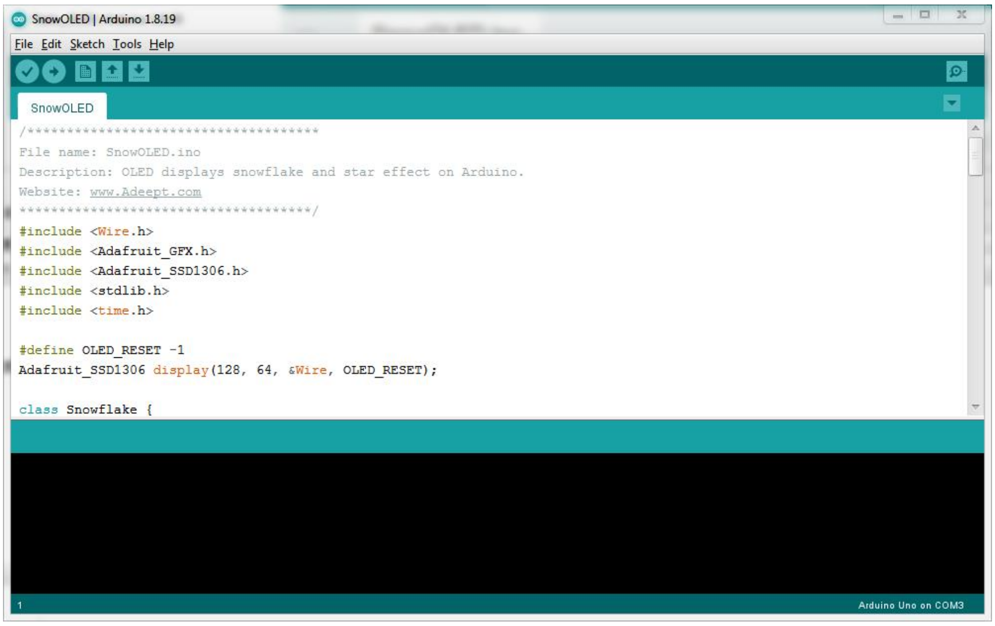
Done uploading.

```
Sketch uses 1760 bytes (5%) of program storage space. Maximum is 32256 bytes.
Global variables use 30 bytes (1%) of dynamic memory, leaving 2018 bytes for local variables. Maximum is 2048 bytes.
```

1 Arduino Uno on COM3

5. When the program runs successfully, you will clearly see the words "4WD Smart Car" and "Adeept" on the OLED screen..

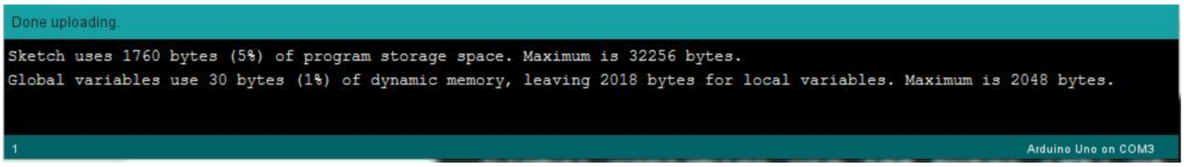
6. Open “**09_OLED/SnowOLED**” folder in “/Code”, double-click “SnowOLED.ino”.



```
/***************************************
File name: SnowOLED.ino
Description: OLED displays snowflake and star effect on Arduino.
Website: www.Adeept.com
***************************************/
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <stdlib.h>
#include <time.h>

#define OLED_RESET -1
Adafruit_SSD1306 display(128, 64, &Wire, OLED_RESET);

class Snowflake {
```

1 Arduino Uno on COM3

7. After opening, click → to upload the code program to the Arduino. If there is no error warning in the console below, it means that the Upload is successful.



Done uploading.

```
Sketch uses 1760 bytes (5%) of program storage space. Maximum is 32256 bytes.
Global variables use 30 bytes (1%) of dynamic memory, leaving 2018 bytes for local variables. Maximum is 2048 bytes.
```

1 Arduino Uno on COM3

8. After successfully running the program, simulate the animation effect of snow on the OLED screen.

13.6 Code

Complete code refer to Static_OLED.ino

```
/*************************************
File name:OLED
Description:OLED displays text.
Website:www.addept.com
*************************************/
#include <Wire.h>
#include <SSD1306Ascii.h>
#include <SSD1306AsciiWire.h>
SSD1306AsciiWire display;

void setup() {

  Wire.begin();
  Wire.setClock(400000L);
  display.begin(&Adafruit128x64, 0x3C);
  display.setFont(Adafruit5x7);
    display.clear();
  //Set the font size
  display.set2X();
  display.setCursor(0,40); // (x,y)
  //String displayed
  display.println("\n4WD Smart\nCar");
  display.println("\n");

  display.set1X();
  //Set the display location
  display.setCursor(80,55);
  display.print("Adeept");
  //Began to show
}

void loop() {

}
```

Complete code refer to SnowOLED.ino

```
/*************************************
File name: SnowOLED.ino
Description: OLED displays snowflake and star effect on Arduino.
Website: www.Adeept.com
*************************************/
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <stdlib.h>
#include <time.h>
```

```

#define OLED_RESET -1
Adafruit_SSD1306 display(128, 64, &Wire, OLED_RESET);

class Snowflake {
public:
    int x;
    int y;
    int speed;

    Snowflake(int xPos, int yPos, int s) {
        x = xPos;
        y = yPos;
        speed = s;
    }

    void fall() {
        y += speed;
        if (y > 64) {
            y = 0;
            x = random(0, 128);
        }
    }

    void draw() {
        display.drawPixel(x, y, WHITE);
    }
};

class Star {
public:
    int x;
    int y;
    bool isBright;

    Star(int xPos, int yPos) {
        x = xPos;
        y = yPos;
        isBright = random(0, 2);
    }

    void twinkle() {
        if (random(0, 10) == 0) {
            isBright =!isBright;
        }
    }

    void draw() {
        display.drawPixel(x, y, isBright? WHITE : BLACK);
    }
};

Snowflake* snowflakes[20];
Star* stars[10];

void setup() {
    Wire.begin();
    display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
    display.clearDisplay();
    display.display();
    randomSeed(millis());
    for (int i = 0; i < 20; i++) {
        snowflakes[i] = new Snowflake(random(0, 128), random(0, 64), random(1, 3));
    }

    for (int i = 0; i < 10; i++) {
        stars[i] = new Star(random(0, 128), random(0, 64));
    }
}

void loop() {
    display.clearDisplay();

    for (int i = 0; i < 20; i++) {
        snowflakes[i]->fall();
        snowflakes[i]->draw();
    }

    for (int i = 0; i < 10; i++) {
        stars[i]->twinkle();
        stars[i]->draw();
    }

    display.display();
    delay(100);
}
```

Code explanation

Static_OLED.ino

Initialization Stage:

In the setup () function, first initialize I2C communication with Wire.setLock (400000L) and stabilize the frequency, then initialize the Adafruit 128x64 type OLED screen with address 0x3C, set the font to Adafruit 5x7 and clear the screen; Next, use display. set2X() to enlarge the font, locate it with display. setCursor (0,40), and finally use display. println to display "4WD Smart Car".

Loop Control Process:

Since the displayed content is static, no additional operation is required.

SnowOLED.ino

Initialization Stage:

In the setup () function, initialize I2C communication, then initialize the OLED screen and set the power mode and I2C address, followed by screen clearing update, and finally set a random number seed for subsequent generation of random positions and speeds.

Loop Control Process:
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.