Crowbits
by Elecrow

Master Kit

A STEAM Kit Suitable for Creating Your Own Products

USER MANUAL V1.1

www.elecrow.com

Directory

Foreword

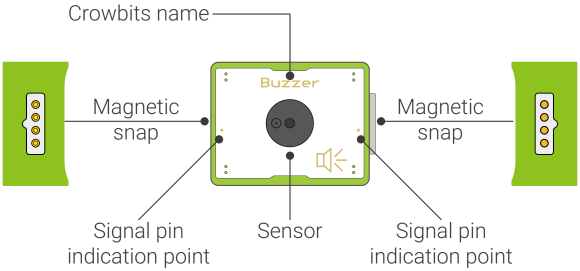
1. About Crowbits

Crowbits is a STEAM educational equipment specially designed for young technology enthusiasts. It is a programmable magnetic electronic module and compatible with Lego. Standard Lego holes are designed on the bottom and sides of the module, which can be perfectly connected with Lego. Modules can realize programming-free and programming functions at the same time. Equipped with three mainboard console and hundreds of electronic modules, it will greatly inspire kids' creative ability. The modules are perfectly packaged and connected by magnet, so that young kids can quickly get started and enjoy much fun with a simple plug.

Get to Know Crowbits modules

Structure:

Front:



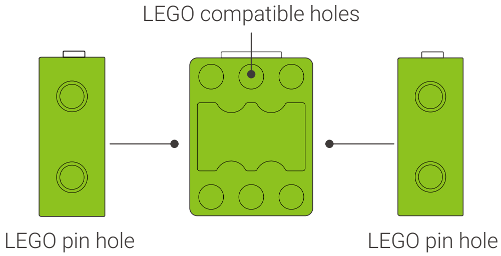
Crowbits name: Crowbits consists many types of modules, so we marked the name for you to recognize easily.

Sensor: The main component of Crowbits is placed there.

Magnetic snap: It's a magical magnetic pogo-pin, and the magnetic snap allows you to connect Crowbits together with magnet.

Signal pin indication point: This point represents the position of the signal input/output pins of the module. When connected to the main console or other modules, only the pins at this point can control the module. The main console uses the pin number to label the corresponding input/output pins.

Back:



A number of Lego holes are reserved on the back and side of Crowbits electronic modules, which are perfectly compatible with Lego rods or pins, and the connection is extremely convenient.

2. Get to know Crowbits Master Kit

The Crowbits Master kit is the ultimate kit of the Crowbits series, which mainly contains three applications: radar, mobile phone and game console. The power of the Crowbits module is that it allows us to make applications in life with the skill of programming we learned. In this kit, we will learn the programming and usage of each functional module in each project according to the needs of these three applications, and finally complete an remarkable project.

Radar project: the radar can scan the obstacle in front of it and show the distance between them, which displays the scanning results on the screen by dots. Before making a complete radar project, you need to understand the mainboard is used to control it and learn how to display on the screen. Then, you need to master the servo control principle, finally learn the use of a laser ranging sensor to make a radar project of your own.

Mobile phone project: a fully functional Crowbits mobile phone. Before making a complete mobile phone project, you need to learn how to use the Keyboard module, get to know the 2G module to implement individual basic functions, and finally make a mobile phone project.

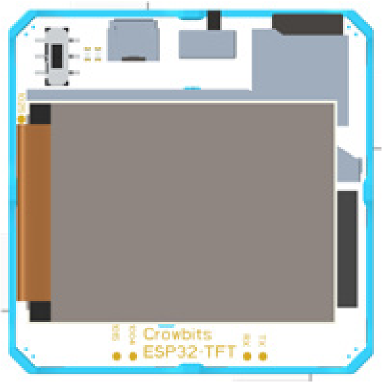
Game console project: a classic game console with dual joysticks. First to learn how to use the joystick, then use the joysticks to play games in the Letscode software, and finally combine the ESP32 module to make a classic game console project.

3. Preparation

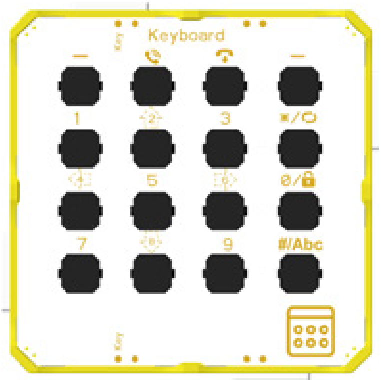
Log in to the official website: www.elecrow.com, then search in the download area to download the Letscode software and all the programs of the Master Kit.

4. Module list of Master Kit

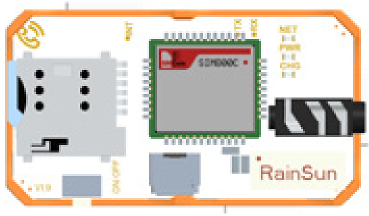
ESP32 Module x1



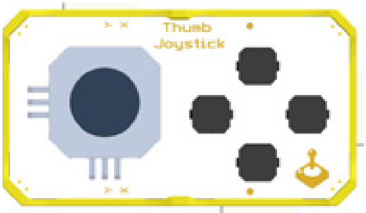
Keyboard Module x1



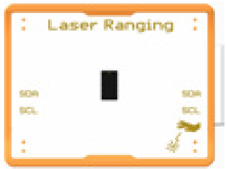
2G Module x1



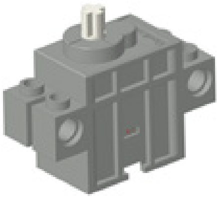
Thumb Joystick Module x2



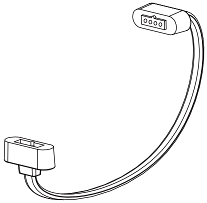
Laser Ranging Sensor x1



Servo Module x1



Magnetic Cable x1



Lesson 1 Know About ESP32 Module

1. Introduction

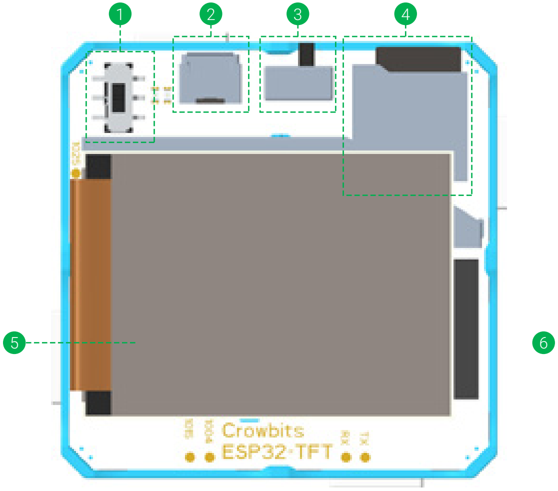
The arrival of the Internet has profoundly affected and enriched our lives, among which communication cables (network cables) and wifi are two important components of the Internet. ESP32 is a main control module with a wifi chip. You can not only use it to control various hardware modules, but also use it as a wifi module that can connect to the Internet. Now, let us learn how to use the wifi function of ESP32 module!

2. Learning Goals

- Know about what the ESP32 module can do.
- Know about the method of displaying content on the TFT display.
- Master the use of ESP32 module to connect to wifi.
- Master programming and uploading with letscode software.
- Master the use of mobile phones or computers to access ESP32 modules.

3. Hardware List

1. ESP32 Module



4. Knowledge Analysis

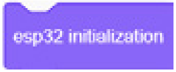
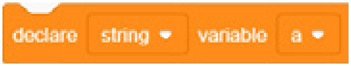
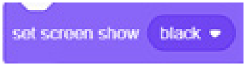
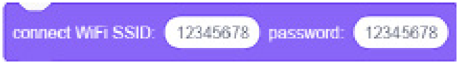
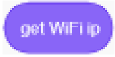
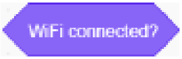
1. Hardware Introduction

The ESP32 module has 11 expansion pins such as IO25, IO24, IO13, RX, TX etc. We can connect the sensor to the corresponding pin and use the program to control it using the ESP32 module.

There are 6 functional areas on the ESP32 module as circled in the picture above:

① Mode switch. It can be switched to 2G mode and normal mode. When we need to call and send messages, we switch to 2G mode.
② Data & Charger port. The port to download the program and charge the ESP32 module.
③ ESP32 power switch. Toggle to the left to turn on, and to the right to turn off.
④ Memory card slot.
⑤ TFT screen. The 320*240 screen can display text, pictures and etc.
⑥ Speaker. Under the TFT display, there is a speaker, which is the sound module of the ESP32 module.

2. Blocks Explanation

Block	Explanation
 esp32 initialization	Initialize ESP32 module, you need to initialize the ESP32 module before using it
 declare string ▾ variable a ▾	Declare the type of the variable. There are integer(int), floating point(float), character(char), string, boolean, unsigned string, unsigned character(unsigned char), and long integer(long int). What needs to be displayed on the TFT screen needs to declared as a string. For arithmetic operations, we generally declare it as an integer
 set screen show black ▾	Set the screen display color, click the inverted triangle to select the color
 connect WiFi SSID: 12345678 password: 12345678	Connect the WiFi by entering name and wifi password
 get WiFi ip	Get the ip address after connecting to wifi
 WiFi connected?	Judge whether the wifi connection is successful, return True if the connection is successful, otherwise return False

Continued

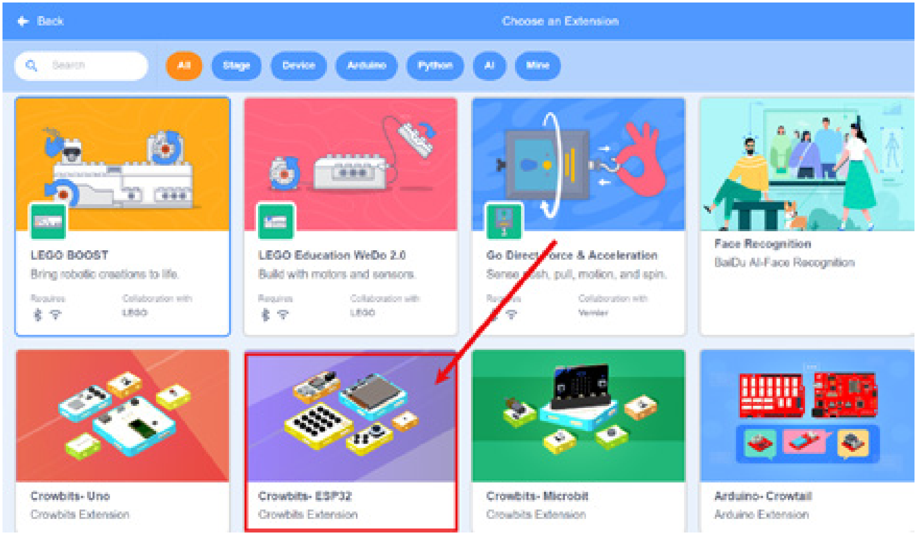
set screen display "Letscode" at x: 10 y: 20	Display the content in the box at the specified coordinates x, y
set webpage display "Letscode"	After successfully connecting to wifi, enter the corresponding IP and the content in the box will be displayed

5. Project Realization

We will program the ESP32 module to connect to wifi through letscode software. First to connect the ESP32 module to wifi and print out the IP address of the ESP32 module when it succeeds, and then connect your mobile phone to the same wifi network, and finally open the browser on your phone and enter the ESP32 IP address to access the webpage of the ESP32 module.

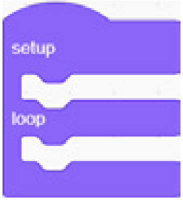
1. Programming

1. Open the letscode software, click the icon in the lower left corner, and select Crowbits - ESP32, as shown in the picture below:



Picture 1-1

Now we have imported the Arduino library of ESP32 into letscode. First we will see the main program, as shown below.



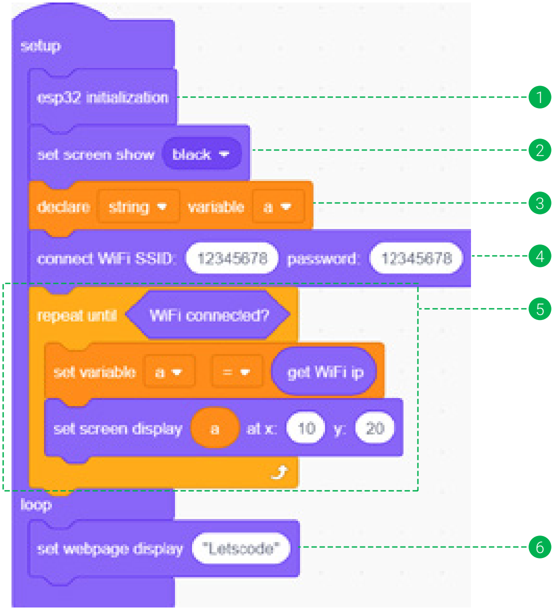
setup

loop

Picture 1-2

The program can only be run after putting inside it. The program inside the “setup” is executed only once, and is generally used to initialize variables and modules. The program inside the “loop” is the main logic program, and generally repeats until the power switch is toggled to off.

2. In the setup area:



setup
esp32 initialization
set screen show black
declare string variable a
connect WiFi SSID: 12345678 password: 12345678
repeat until WiFi connected?
set variable a = get WiFi ip
set screen display a at x: 10 y: 20
loop
set webpage display "Letscode"

Picture 1-3

1 First to initialize the ESP32 module to display the content later.

2 Set the screen display to black to make the content displayed on the screen clearer.

3 Create variable “a” and declare it as a string. To display the content on the screen, it must be a string on the letscode software, therefore you need to declare variable “a” as a string here.

4 Connect the ESP32 module to the to the WiFi(your mobile hotspot or home hotspot).

5 When not connected to the wifi network, it will loop to try to get wifi, and the IP address will be displayed on the screen. We assign the obtained wifi IP address to the string "a" and display it on the screen.

3. In the loop area:

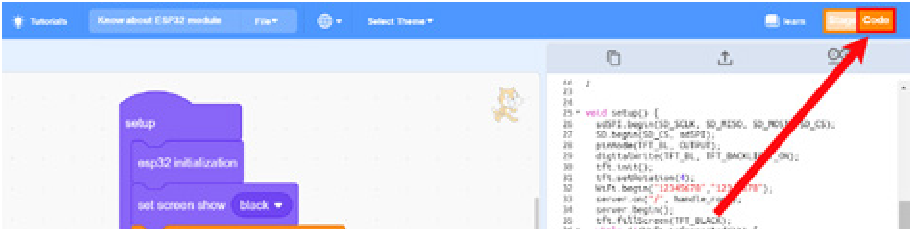
6 When the mobile phone is connected to the wifi and obtains an IP address, the pre-set content will be displayed on the web page of your phone after you access the IP address of the ESP32 module.

The complete code of the project is shown in picture 1-3.

You can find the complete program of this case in the downloaded Master kit folder: Know about ESP32 module.sb3, and open it with letscode.

2. Download

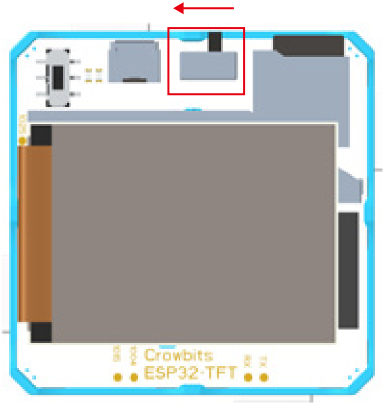
1 Click the “code” icon in the upper right corner of the letscode software to switch to code mode, as shown in picture 1-4 below.



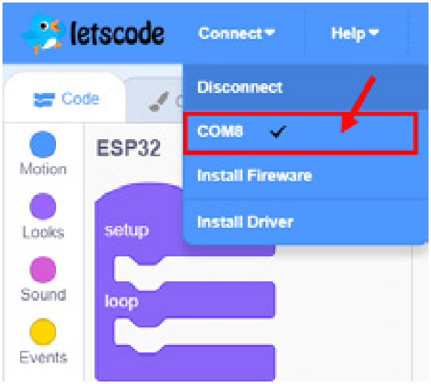
```
}

void setup() {
    sdSPI.begin(SD_SCLK, SD_MISO, SD_MOSI, SD_CS);
    SD.begin(SD_CS, sdSPI);
    pinMode(TFT_BL, OUTPUT);
    digitalWrite(TFT_BL, TFT_BACKLIGHT_ON);
    tft.init();
    tft.setRotation(4);
    WiFi.begin("12345678","12345678");
    server.on("/", handle_root);
    server.begin();
    tft.fillScreen(TFT_BLACK);
```

Picture 1-4

2 Connect the ESP32 module to the computer USB port with the USB cable, and toggle the power switch of the ESP32 module to ON, as shown in picture 1-5 below. Then click the “connect” button in the upper left corner of the software to connect to the corresponding port(check icon is displayed after the port), as shown in picture 1-6.

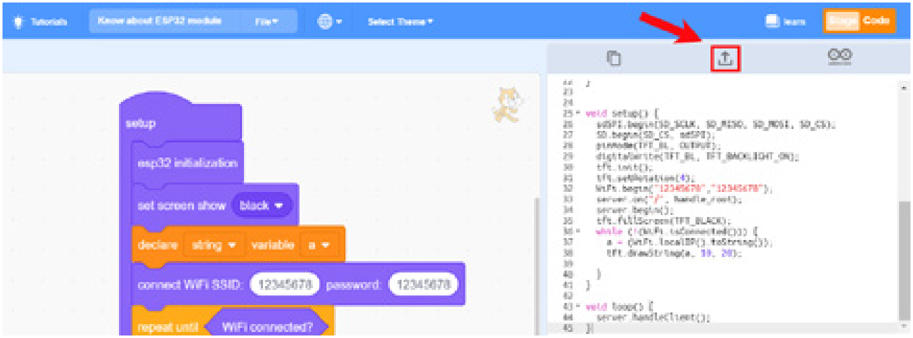


Picture 1-5



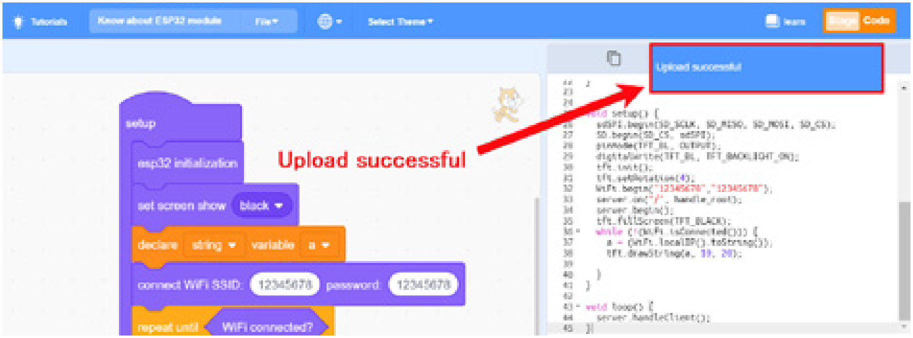
Picture 1-6

Click the icon as shown in picture 1-7 to download the code to the ESP32 module. After clicking download, wait for the download successful prompt as shown in picture 1-8.



```
}


void setup() {
  sdSPI.begin(SD_SCLK, SD_MISO, SD_MOSI, SD_CS);
  SD.begin(SD_CS, sdSPI);
  pinMode(TFT_BL, OUTPUT);
  digitalWrite(TFT_BL, TFT_BACKLIGHT_ON);
  tft.init();
  tft.setRotation(4);
  WiFi.begin("12345678","12345678");
  server.on("/", handle_root);
  server.begin();
  tft.fillScreen(TFT_BLACK);
  while (!(WiFi.isConnected())) {
    a = (WiFi.localIP().toString());
    tft.drawString(a, 10, 20);

  }
}

void loop() {
  server.handleClient();
}
```

Picture 1-7



Upload successful

Upload successful

```
}


void setup() {
  sdSPI.begin(SD_SCLK, SD_MISO, SD_MOSI, SD_CS);
  SD.begin(SD_CS, sdSPI);
  pinMode(TFT_BL, OUTPUT);
  digitalWrite(TFT_BL, TFT_BACKLIGHT_ON);
  tft.init();
  tft.setRotation(4);
  WiFi.begin("12345678","12345678");
  server.on("/", handle_root);
  server.begin();
  tft.fillScreen(TFT_BLACK);
  while (!(WiFi.isConnected())) {
    a = (WiFi.localIP().toString());
    tft.drawString(a, 10, 20);

  }
}

void loop() {
  server.handleClient();
}
```

Picture 1-8

6. Brainstorming

Let’s think about it, is this possible to use this method to send a message to your parents? You can connect the ESP32 module to wifi at home and get the IP address. And then write your SMS on the letscode software. After succeeding upload, share the IP with your parents and let them check what you want to say to them, isn’t it fun? Let’s do it and surprise your parents!

Lesson 2 Learning the ESP32-TFT Module

1. Introduction

Billboards can be seen everywhere in life, displaying pictures, text, videos, etc. But do you know how can we make the billboard to display?

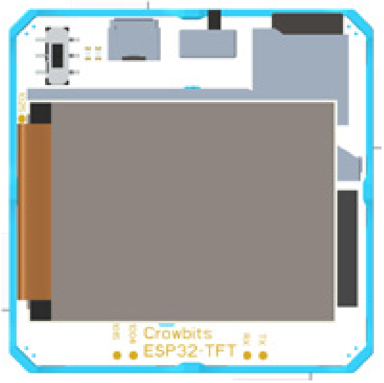
The TFT on the ESP32 module is a 2.0-inch LCD screen, it can display games, animations, text and etc. In this lesson, we will use a program to display pictures and scroll text on the TFT display.

2. Learning Goals

- Master the concept and usage of coordinates.
- Master how to display pictures on TFT.
- Master how to display text on the TFT screen.
- Master how to scroll text on TFT screen.

3. Hardware List

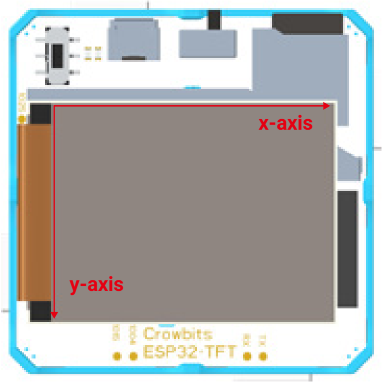
1. ESP32 Module



4. Knowledge Analysis

1. Hardware Introduction

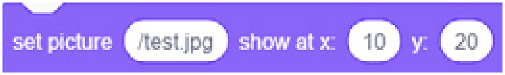
TFT screens are also known as thin-film field-effect transistors, which are a type of active-matrix liquid crystal display. You can control each individual pixel on the screen, which can greatly decrease the response time. The pixel of the TFT screen on the ESP32 module is 320*240. When the ESP32+TFT module is placed as shown in picture 2-1, the coordinates of the upper left corner of the TFT screen are (0, 0), the coordinate of the x-axis increases from left to the right, and the coordinate of the y-axis increases from top to bottom. In this case, we will display pictures, text on the TFT screen in the form of coordinate positioning.



the coordinate value increases along the x-axis and y-axis arrow direction
the maximum value of the x-axis coordinate is 320
the maximum value of the y-axis coordinate is 240

Picture 2-1

2. Blocks Explanation

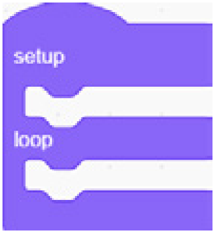
Block	Explanation
 set picture /test.jpg show at x: 10 y: 20	Display picture at the specified coordinates. "/" means the root path of the memory card. If the picture is placed in a folder under the root path, the path format is "/folder name/picture name"

5. Project Realization

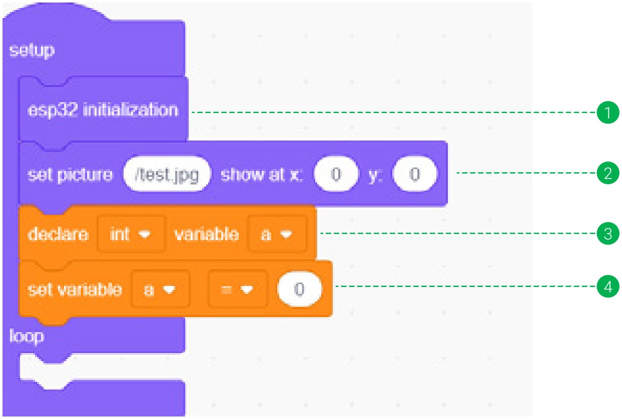
In this case, we will use the TFT screen to display the pictures stored in the memory card and scroll the text from left to right on the picture.

1. Programming

1. Add setup/loop block



2. In the setup area:



Picture 2-2

① Before displaying on the TFT screen, we need to initialize the esp32 module.

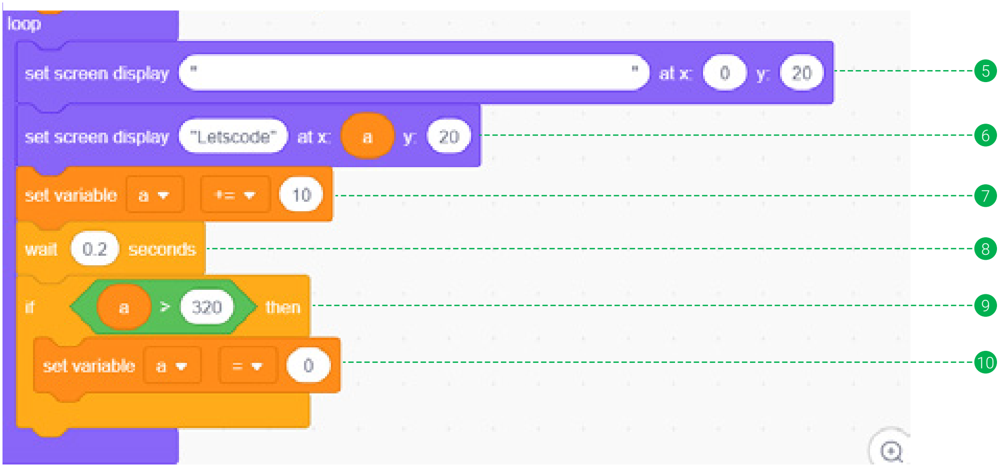
② Display the picture on the TFT screen. When the coordinate you fill in are (0, 0), and the picture size is 320*240 pixels, the entire TFT screen will be covered. If the picture is larger than this pixel, it will not be displayed completely. If it is smaller than this pixel, it cannot cover the entire TFT screen. The picture to be displayed is named test.jpg and it is placed in the root path of the memory card, so here the path we fill in is /test.jpg.

③ Declare the integer variable "a". The function of the integer variable "a" is to change the x coordinate of the displayed content. We change the x coordinate of the displayed content by changing the value of the variable "a" to scroll the content.

④ Initialize the value of variable "a" to 0. The initialization here is to determine where the content to be displayed and start to scroll. If it is initialized to 0, scroll from the leftmost side of the TFT screen.

3. In the loop area:

Scroll the texts from left to right. The scrolling function is to switch between display and emptying. Scrolling from left to right by adding the value of the x-axis.



```
loop
  set screen display " " at x: 0 y: 20
  set screen display "Letscode" at x: a y: 20
  set variable a += 10
  wait 0.2 seconds
  if a > 320 then
    set variable a = 0
```

Picture 2-3

5 In the setup part, we have determined the x coordinate to display the text. Now it's the y coordinate. For example, the y coordinate of this case is 20. Before displaying the text, we need to clear the entire row with the y coordinate is 20.

6 Next display the text "Letscode" (can be modified to the text you want to display), the x coordinate is the variable “a” (already been set in setup area), and the y coordinate is always 20.

7 After displaying it, the value of variable “a” (real-time x coordinate) is increased by 10, and the position displayed next time is increased by 10 on the basis of the previous display. This is to set the amplitude of each scroll. The larger the value, the greater the amplitude of each scroll. Otherwise, the rolling range is smaller.

8 Wait 0.2 seconds is to set the time for each scrolling, the smaller the time is set, the faster the scrolling will be.

9 Finally, when the text scrolls to the far right of the screen, we need to write a judgment program to judge that if the value of the integer variable “a” (real-time x coordinate) is greater than the maximum value of the TFT screen 320, meaning that the displayed text has exceeded the TFT screen Maximum value.

10 At this time, we set the value of the integer variable “a” to 0 (the leftmost side of the TFT screen).

You can find the complete program of this case in the downloaded Master kit folder: Learning the ESP32-TFT module.sb3,and open it with letscode.

2. Download

After finishing the program, download it to the ESP32 module (if you forget the steps to download the program, please refer to it in Lesson 1). After downloading the program, remember to restart the ESP32 module, then you can see the preset picture and scrolling text on the screen.

6. Brainstorming

It is amazing to make the screen into a billboard! If we want the text to scroll from top to bottom, how can we achieve it? Let's try it!

Lesson 3 Drawing on the TFT Screen

1. Introduction

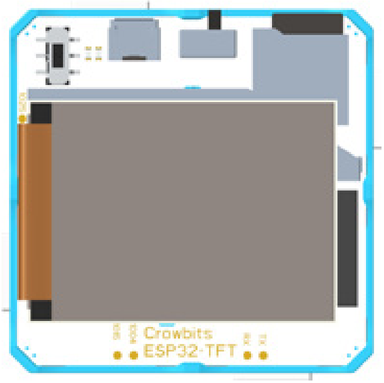
There is a small display screen on the radar, which used to display the detected objects. When the scan line passes, if there is an object, a small dot will appear on the radar screen. If no object is detected, no dots will appear on the radar screen. The TFT screen is like a radar screen, which can display dots, lines and other patterns on it. In this lesson, let us learn how to draw dots and lines of different colors on the TFT screen!

2. Learning Goals

- Master the usage of coordinates.
- Master how to draw points on the TFT screen.
- Master how to draw lines on the TFT screen.
- Master how to draw circles and rectangles on the TFT screen.
- Master how to clear the TFT screen.

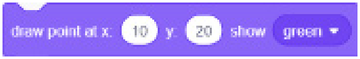
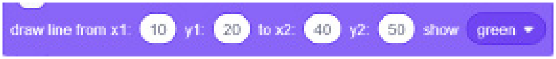
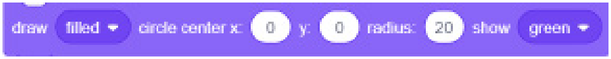
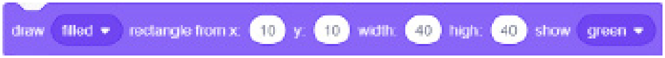
3. Hardware List

1. ESP32 Module



4. Knowledge Analysis

1. Blocks Explanation

Block	Explanation
 draw point at x: 10 y: 20 show green ▾	Draw a point on the coordinates where x is 10 and y is 20, the color of the point is green
 draw line from x1: 10 y1: 20 to x2: 40 y2: 50 show green ▾	The starting point coordinate x is 10 and y is 20. The endpoint coordinate x is 40 and y is 50. Connect two points with a straight line, the color is green
 draw filled ▾ circle center x: 0 y: 0 radius: 20 show green ▾	Draw a circle with coordinates x=0, y=0 as the center, 20 as the radius, and the color is green, filled means that the color fills the entire circle, and not filled means that the circle is not filled with color
 draw filled ▾ rectangle from x: 10 y: 10 width: 40 high: 40 show green ▾	Draw a rectangle with starting point coordinates x=10, y=10, width 40, height 40, and color green, filled means that the color fills the entire rectangle, and not filled means that the color does not fill the rectangle

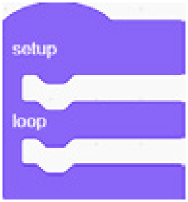
5. Project Realization

The Olympic rings symbolize the gathering of athletes from the five continents and the world at the Olympic Games. It not only represents the unity and friendship of athletes from all over the world on the five continents under the Olympic flag, but also stresses that all athletes should meet each other on the field of competition in a fair and honest spirit.

In this lesson, let's use what we have learned to draw an Olympic rings on the TFT screen!

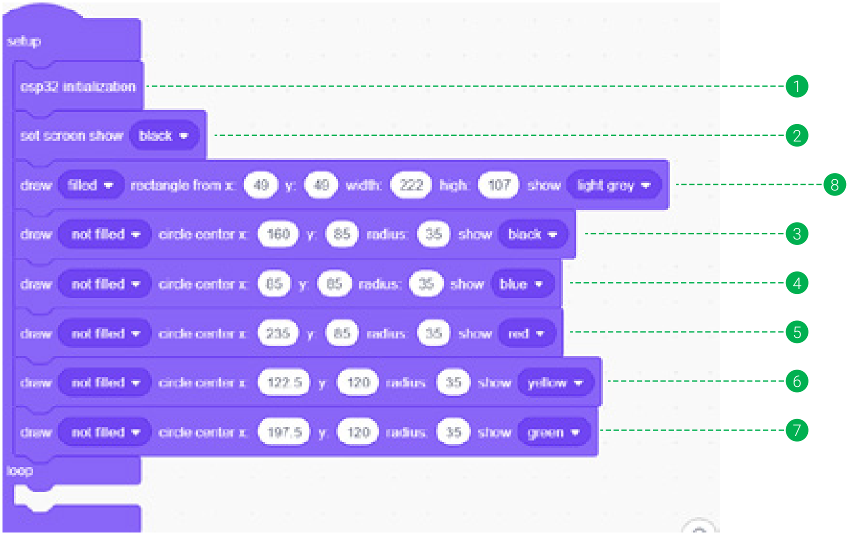
1. Programming

1. Add setup/loop block



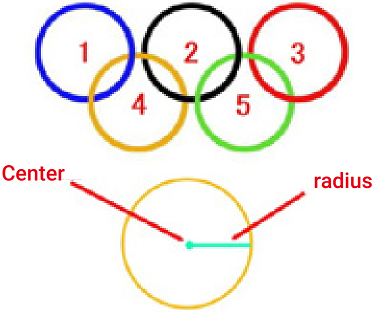
```
setup
loop
```

2. In the setup area:



```
setup
  esp32 initialization                                                                  1
  set screen show black                                                                 2
  draw filled rectangle from x: 49 y: 49 width: 222 high: 107 show light grey           8
  draw not filled circle center x: 160 y: 85 radius: 35 show black                      3
  draw not filled circle center x: 85 y: 85 radius: 35 show blue                        4
  draw not filled circle center x: 235 y: 85 radius: 35 show red                        5
  draw not filled circle center x: 122.5 y: 120 radius: 35 show yellow                  6
  draw not filled circle center x: 197.5 y: 120 radius: 35 show green                   7
loop
```

Picture 3-1



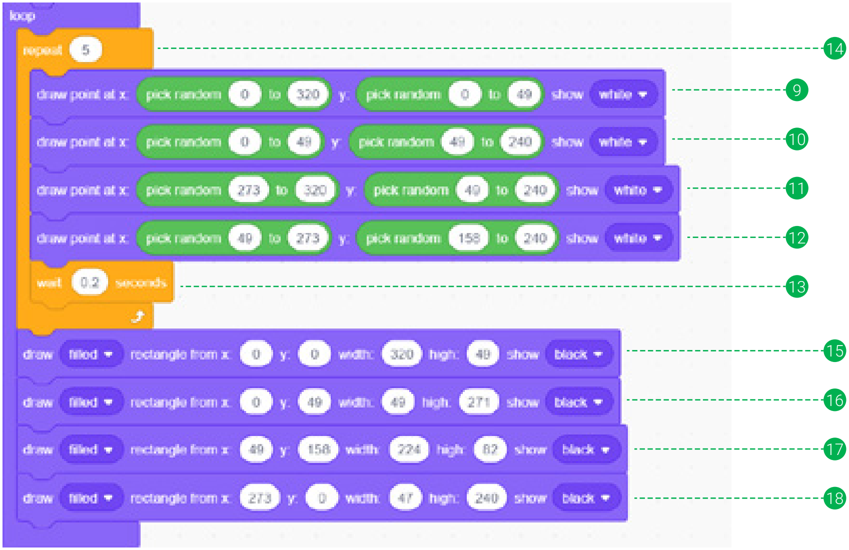
Picture 3-2

1. Because we need to display content on the screen, the initialization of the ESP32 module is an essential step.
2. Set the screen background to black, which allows you see clearly when displaying the content.
3. Now let's draw the Olympic rings. First, we need to determine the center coordinate of the middle circle among the five rings(Circle 2 in Picture 3-2). Our TFT screen has a width of 320 pixels and a height of 240 pixels. In order to align the five rings in the center, we set the x coordinate of Circle 2 at 160. The height is slightly above the screen, so the y coordinate is set at 85, and the radius(as shown in the lower part of the Picture 3-2) of the circle is set to 35. The circle in the middle of the Olympic rings is black, so the color is set to "black".
4. Next we will draw the second circle(Circle 1 in Picture 3-2). The three circles above the Olympic rings have the same height, so the y coordinate of the second circle is set to 85. The x coordinate of the center of Circle 2 is 160. If the center of Circle 1 is 160 minus twice the radius, then Circle 1 and Circle 2 are tangent. We can make the distance between the two circles to 5, so the calculation formula for the x coordinate of Circle 1 is: 160-2*35-5=85, and the color is blue.
5. In the same way, Let's draw the third circle(Circle 3 in Picture 3-2), the y coordinate of Circle 3 is also 85, the calculation formula for the x coordinate is: 160+35*2+5=235, and the color is red.
6. The x coordinate on the left side of the two circles below the five rings(Circle 4 in Picture 3-2) is the x coordinate of Circle 1 plus the radius, and then plus half of the distance between the two circles, that is: 85+35+2.5=122.5. The y coordinate is the y coordinate of Circle 1 plus the radius, that is, 85+35=120, and the color is yellow.
7. The x coordinate on the right side of the two circles below the five rings(Circle 5 in Picture 3-2) is the x coordinate of Circle 4 plus twice the radius, then plus the distance5. between the two circles, that is, 122.5+2*35+5=197.5. The y coordinate same as the y coordinate of Circle 4, the color is green.
8. We add a rectangle to the five rings. The purpose of the rectangle is to enclose the five Olympic rings. The x coordinate of the starting point of the rectangle is the center coordinate of Circle 1 minus the radius, that is, 85-35=50. In order to prevent the rectangle from being tangent to the circle, we subtract 1 from the x coordinate, so the x coordinate of the starting point of the rectangle is 49. In the same way, the y coordinate is also 49. The width of the rectangle is the diameter of the three circles and plus the distance between the two circles, that is, 70*3+5*2=220. In order to prevent the circles on both sides from being tangent to the rectangle, add 1 for each side, and the left and right widths are 222. The height is three radii, plus 1 on each side to prevent tangency, that is, 35*3+2=107, and the height is 107.

Note: Add the rectangle before drawing the Olympic rings, otherwise the Olympic rings will be covered.

3. In the loop area:

For a better effect, we add shiny "little stars" in the loop area.



```
loop
  repeat 5
    draw point at x: pick random 0 to 320 y: pick random 0 to 49 show white
    draw point at x: pick random 0 to 49 y: pick random 49 to 240 show white
    draw point at x: pick random 273 to 320 y: pick random 49 to 240 show white
    draw point at x: pick random 49 to 273 y: pick random 158 to 240 show white
    wait 0.2 seconds
  draw filled rectangle from x: 0 y: 0 width: 320 high: 49 show black
  draw filled rectangle from x: 0 y: 49 width: 49 high: 271 show black
  draw filled rectangle from x: 49 y: 158 width: 224 high: 82 show black
  draw filled rectangle from x: 273 y: 0 width: 47 high: 240 show black
```

Picture 3-3

9 We use random numbers to achieve the effect of "little stars". In order not to affect the effect of the Olympic rings, the "little stars" need to avoid the rectangular area and randomly display in the black area.The x coordinate is random within 0-320, and the y coordinate is random within 0-49, which is the black area above the rectangle. The color of the "little star" is white.

10 The black area on the left side of the rectangle, the x coordinate range is 0-49, and the y coordinate range is 49-240. The color of the "little star" is white.

11 The black area on the right side of the rectangle, the x coordinate range is 273-320, and the y coordinate range is 49-240. The color of the "little star" is white.

12 The black area at the bottom of the rectangle, the x-coordinate range is 49-273, and the y coordinate range is 158-240. The color of the "little star" is white.

13 To achieve the shiny effect, 4 "little stars" appear every 0.2 seconds.

14 Next, we have to clear the screen. If the screen is not cleared, the "little star" will never disappear, and the effect of shinny will not be achieved. Then we make sure that the "little star" appears for 5 rounds, then clear the screen.

15 The principle of clearing the screen is to cover with the screen color. If we want to clear some areas of the screen, we only need to cover the same color as the screen. Here we use a way of covering the same color rectangle to clear the screen. First clear the "little star" above. The rectangle needs to be filled with "filled", the starting point coordinates are 0 on the x axis, 0 on the y axis, 320 width and 49 height.

⑯ Clear the "little star" on the left. Select "filled" for the rectangle, the starting point is 0 on the x axis, 49 on the y axis, the width is 49, and the height is 271.

⑰ Clear the "little star" on the right. Select "filled" for the rectangle, the starting point is 49 on the x axis, 158 on the y axis, the width is 224, and the height is 82.

⑱ Clear the "little star" at the button. Select "filled" for the rectangle, the starting point is 273 on the x axis, 0 on the y axis, the width is 47, and the height is 240.

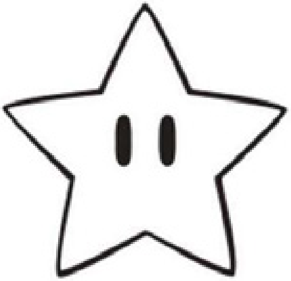
You can find the complete program of this case in the downloaded Master kit folder:
Drawing on the TFT screen.sb3, and open it with letscode.

2. Download

After finishing the program, download it to the ESP32 module (if you forget the steps to download the program, please refer to it in Lesson 1). After downloading the program, remember to restart the ESP32 module to see the effect.

6. Brainstorming

We use small white dots to represent the "little stars" in this lesson, can we draw the "little stars" as shown in the picture below? Let's try it!



Lesson 4 Distance Detector

1. Introduction

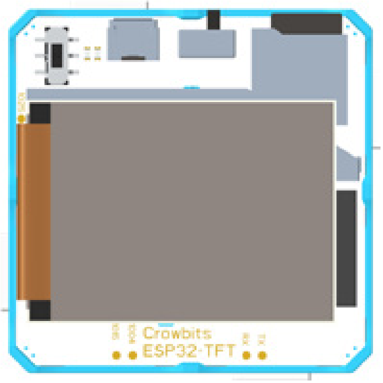
Visual inspection and measurement with a ruler are our daily methods of measuring distance. In this lesson, we will use the crowbits module to make a distance detector for measuring distance, which can measure distance and display it on the TFT screen. If the distance is less than the preset value, the speaker on the ESP32 module will alarm.

2. Learning Goals

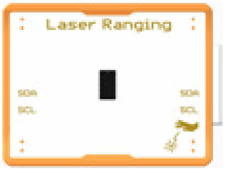
- Get to know the laser ranging sensor.
- Master the usage of laser ranging sensor.
- Master how to use the speaker on the ESP32 module.
- Make a distance detector.

3. Hardware List

1. ESP32 Module



2. Laser Ranging Sensor



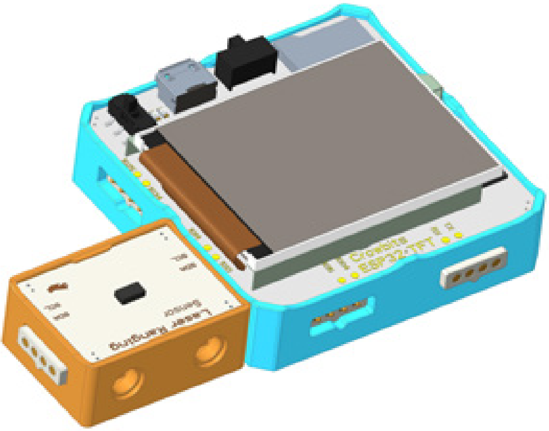
4. Knowledge Analysis

1. Hardware Introduction

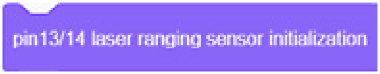
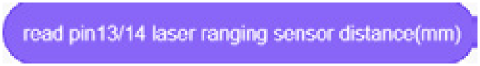
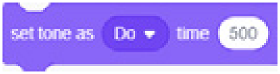
The laser ranging sensor aim at the target and emits laser pulses using a laser diode. The laser is scattered in all directions after being reflected by the target and part of the scattered light returns to the sensor receiver. After being detected, it records and processes the time from when the light pulse is sent to when it returns to be received, and then the target distance can be determined.

The laser ranging sensor of the Crowbits module is a magnetic connection sensor. In the case of this lesson, we can directly connect it to the 13, 14 pin of the ESP32 module.

The hardware connection diagram of this case is as follows:



2. Blocks Explanation

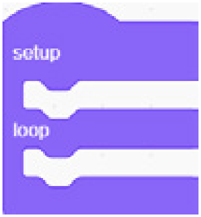
Block	Explanation
 pin13/14 laser ranging sensor initialization	Initialize the module before using
 read pin13/14 laser ranging sensor distance(mm)	After the laser ranging sensor is connected to the 13, 14 pin of the ESP32 module, add this block to obtain the distance value returned by the measurement
 set tone as Do ▾ time 500	Play "Do" tone for 500 milliseconds

5. Project Realization

The distance detector is composed of a laser ranging sensor and an ESP32 module. It can accurately measure the distance within 2 meters. The distance detector will sound an alarm when the distance is less than 10 cm.

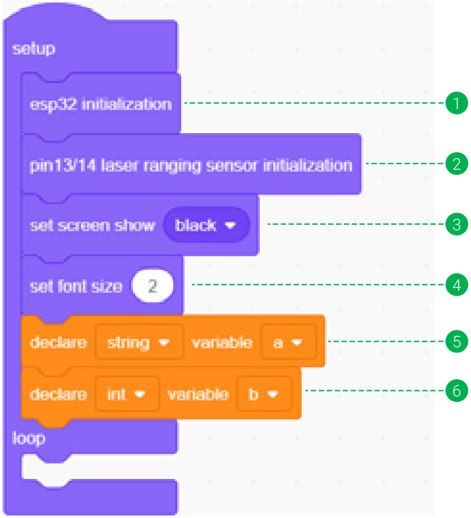
1. Programming

1. Add setup/loop block



```
setup
loop
```

2. In the setup area:



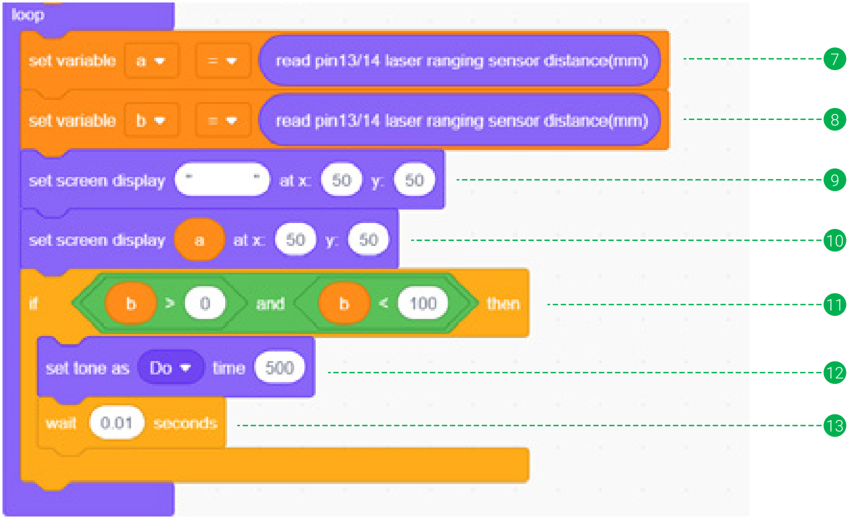
```
setup
  esp32 initialization                              ①
  pin13/14 laser ranging sensor initialization      ②
  set screen show [black ▾]                         ③
  set font size (2)                                 ④
  declare [string ▾] variable [a ▾]                 ⑤
  declare [int ▾] variable [b ▾]                    ⑥
loop
```

Picture 4-1

① Before using the ESP32 module, we need to initialize it.
② For the reason that we need to use the laser ranging sensor, we initialize the laser ranging sensor here.
③ For a better effect to display the text on the TFT screen, set the screen background to black.
④ Set the font size displayed on the TFT screen to 2.
⑤ Declare the string variable "a", which is mainly used to display the real-time distance on the screen.
⑥ Declare the integer variable "b", which is mainly used for data judgment. If the distance is less than the preset value, make the alarm sound.

3. In the loop area:

We need to display the distance on the TFT screen in real-time, and judge whether the currently detected distance is lower than the preset value. If it is, make the alarm sound.



```
loop
  set variable a = read pin13/14 laser ranging sensor distance(mm)
  set variable b = read pin13/14 laser ranging sensor distance(mm)
  set screen display " " at x: 50 y: 50
  set screen display a at x: 50 y: 50
  if b > 0 and b < 100 then
    set tone as Do time 500
    wait 0.01 seconds
```

Picture 4-2

- ⑦ Assign the laser ranging value to the string variable “a” for later display it on the TFT screen.
- ⑧ Assign the laser ranging value to the integer variable “b”, which will be used as a conditional judgment later.
- ⑨ Clear the content of the coordinate before displaying the string variable “a” on the TFT screen, that is, enter the space character to clear the content of the coordinate.
- ⑩ Display the value of the laser ranging sensor on the TFT screen.
- ⑪ Use if/then block to judge whether the integer variable “b”(value detected by laser ranging sensor) is lower than the preset value 100 (can be changed).
- ⑫ If it is less than the preset value(100), a tone will be emitted.
- ⑬ Adding a delay to make the tone sound more like an alarm.

You can find the complete program of this case in the downloaded Master kit folder: Distance detector.sb3, and open it with letscode.

2. Download

After finishing the program, download it to the ESP32 module (if you forget the steps to download the program, please refer to it in Lesson 1). After downloading the program, remember to restart the ESP32 module, now let’s test the distance detector!

6. Brainstorming

In this case, we only set one preset value. Can we set a few more preset values to make the distance detector sound different alarms base on different distances?

Lesson 5 Intrusion Scanner

1. Introduction

Have you ever left food on the table for a while, only to come back and find it had been eaten by your puppy?

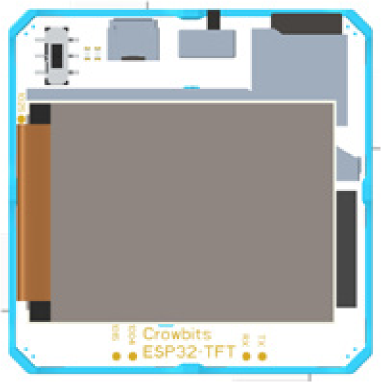
Or sometimes, you don’t want others to approach the boiling hot water for safety. When you want to protect something, making an intrusion scanner will be a good idea. If an object approaches the things you want to protect, the scanner will remind you by playing music.

2. Learning Goals

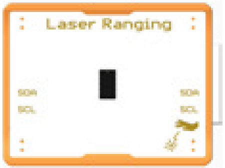
- Master how to use the servo.
- Master how to make music with the speaker.
- Master the structure assembly steps.
- Make an intrusion scanner.

3. Hardware List

1. ESP32 Module



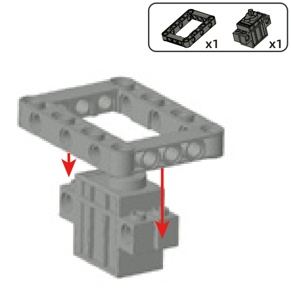
2. Laser Ranging Sensor



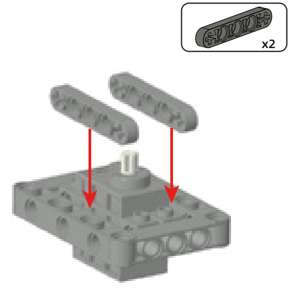
3. Servo Module

Structure building steps:

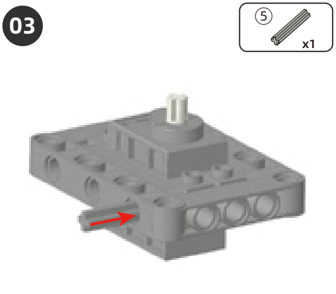
01



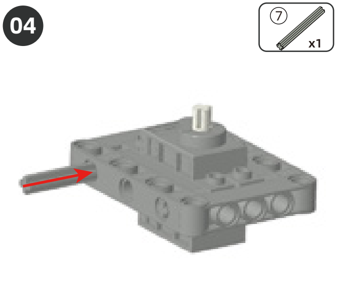
02



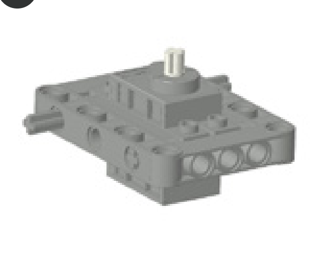
03



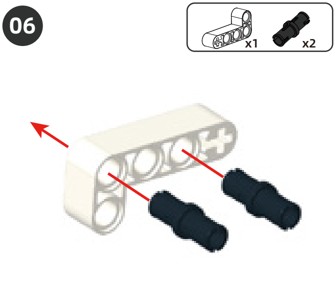
04



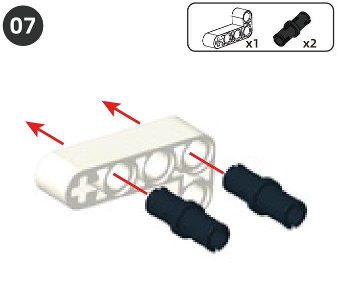
05



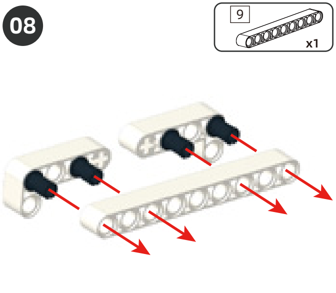
06



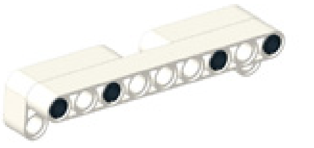
07



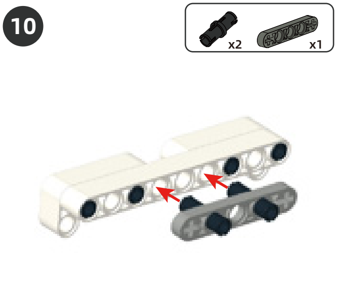
08



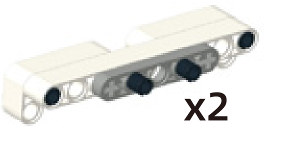
09



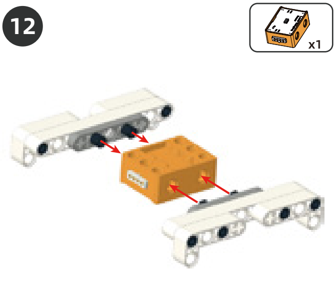
10



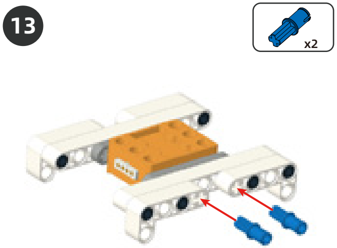
11



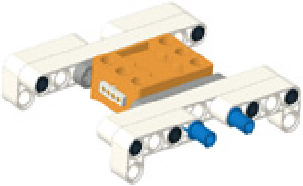
12



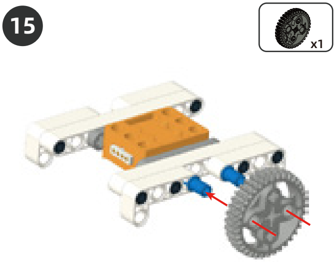
13



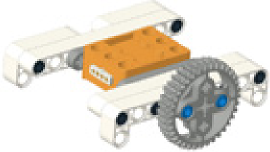
14



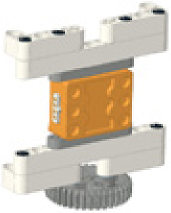
15



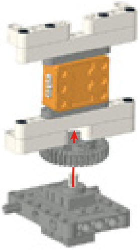
16



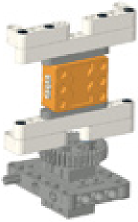
17



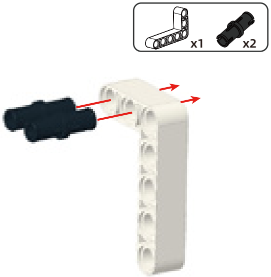
18



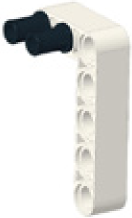
19



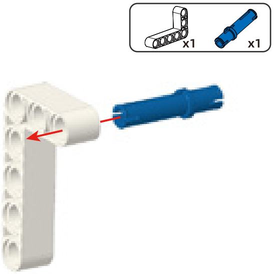
20



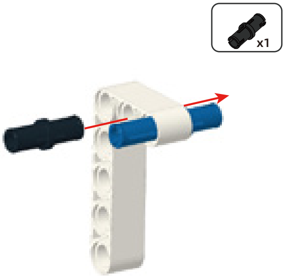
21



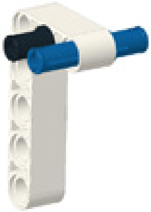
22



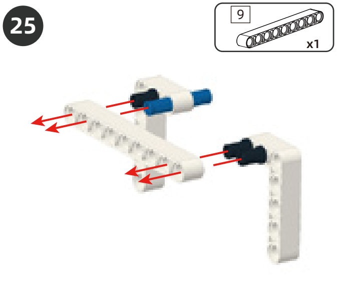
23



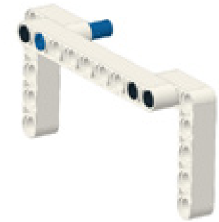
24



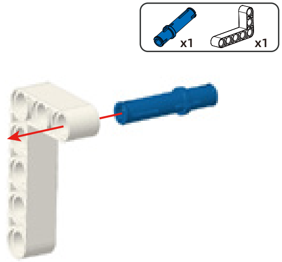
25



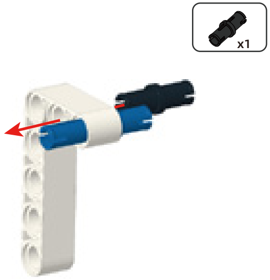
26



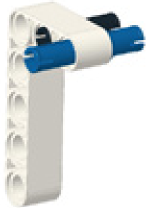
27



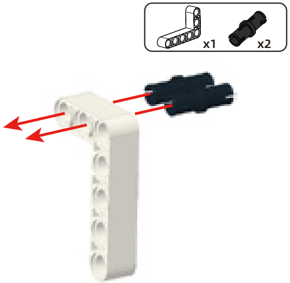
28



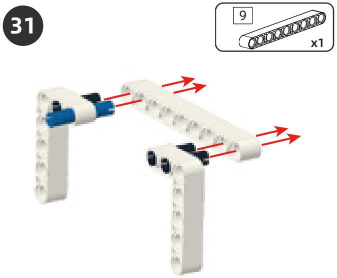
29



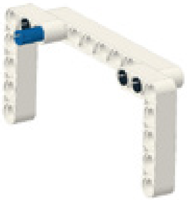
30



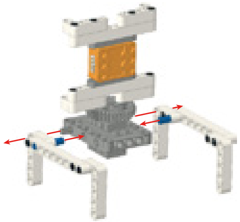
31



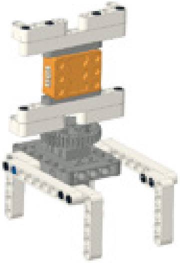
32



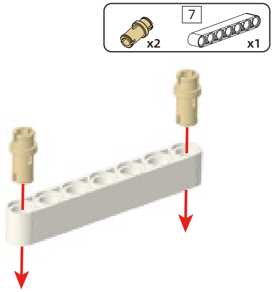
33



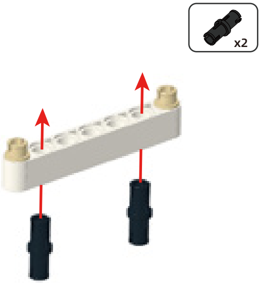
34

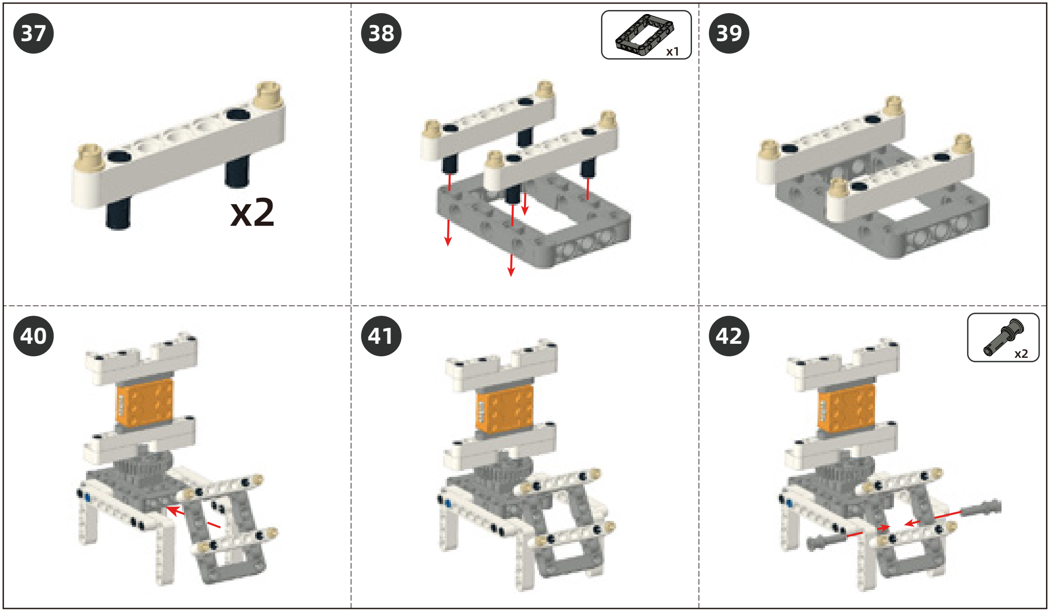


35



36





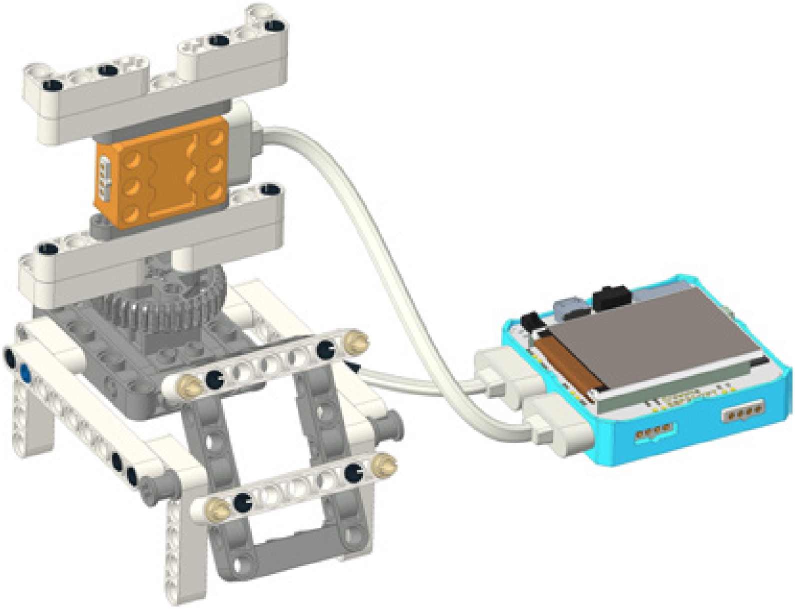
4. Knowledge Analysis

1. Hardware Introduction

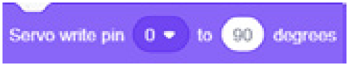
The servo is a kind of angle servo drive, which is suitable for those control systems that need to keep the angle constantly changing. Crowbit's servo can be connected to pin 0 or pin 25 of the ESP32 main control board.

Note: if you want to use both the servo and speaker, then you can only connect the servo to pin 25, because the speaker uses 0 pin. Therefore, to prevent conflicts, the servo can only connect to pin 25 in this case.

The hardware connection diagram of this case is as follows:



2. Blocks Explanation

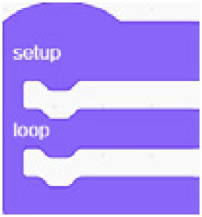
Block	Explanation
 Servo write pin 0 ▾ to 90 degrees	Set pin and turning degrees for the servo. The pin can be set to 0 or 25 according to the hardware connection

5. Project Realization

The intrusion scanner is composed of an ESP32 module, servo, structure, and laser ranging sensor. The structure is built with brick components. The servo can rotates 180 degrees and the laser ranging sensor will scan this area. When there is an object that distance from the laser ranging sensor less than the preset value in the scan area, the speaker on the ESP32 module will make the sound.

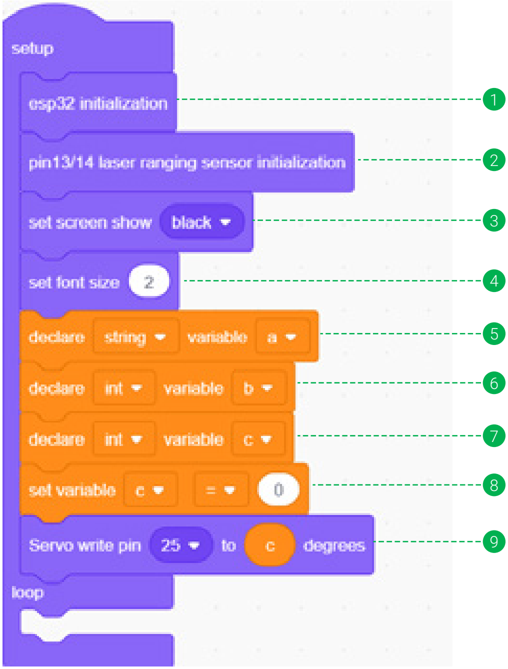
1. Programming

1. Add setup/loop block



```
setup
loop
```

2. In the setup area:



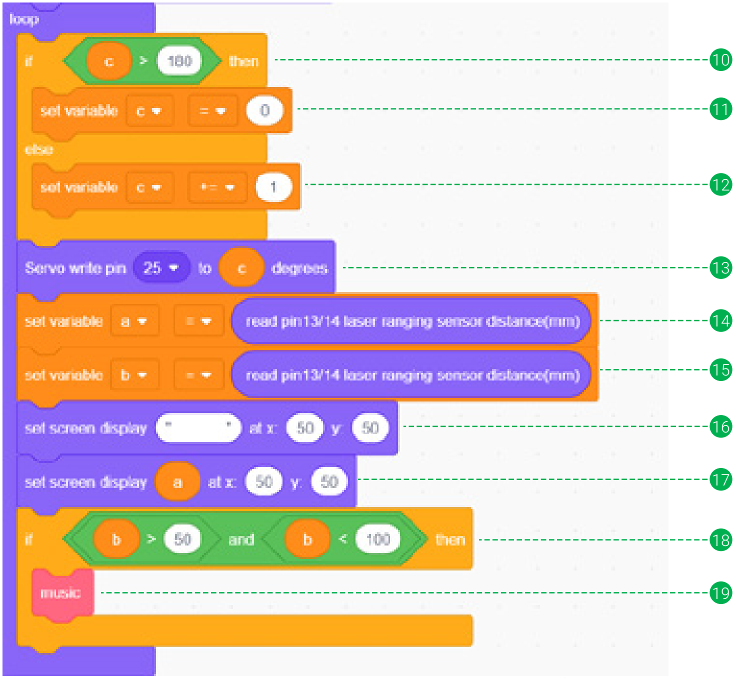
```
setup
  esp32 initialization
  pin13/14 laser ranging sensor initialization
  set screen show black
  set font size 2
  declare string variable a
  declare int variable b
  declare int variable c
  set variable c = 0
  Servo write pin 25 to c degrees
loop
```

Picture 5-1

1. Before using the ESP32 module, we first initialize the ESP32 module.
2. Just like what we learned in the previous lesson, initializing the laser ranging before we use it.
3. For a better effect to display the content on the TFT screen, setting the screen background to black.
4. Set the font size displayed on the screen to 2 for easier to see.
5. In order to display the value detected by the laser ranging sensor on the TFT screen, declaring the variable “a” as a string type, so that the distance detected by the laser ranging sensor can be assigned to the string variable “a” later.
6. Declare the integer variable "b" to be judged as the value of the laser ranging value.
7. Declaring the variable “c” to control the servo rotating between 0-180 degrees.
8. Initialize the value of variable “c” to 0, let variable “c” start from 0 to 180.
9. Set the servo connected to pin 25 and rotate to “c” degrees, in order to initialize the angle of the servo to 0 degrees.

3. In the loop area:

We need to make the servo rotate from 0-180 degrees, and display the distance value detected by the laser ranging sensor on the TFT screen in real-time. When the distance is lower than the preset value, it will emit an alarm.



```
loop
    if c > 180 then
        set variable c = 0
    else
        set variable c += 1
    Servo write pin 25 to c degrees
    set variable a = read pin13/14 laser ranging sensor distance(mm)
    set variable b = read pin13/14 laser ranging sensor distance(mm)
    set screen display " " at x: 50 y: 50
    set screen display a at x: 50 y: 50
    if b > 50 and b < 100 then
        music
```

Picture 5-2

⑩ Before the servo rotates, we need to judge whether the value of “c” is greater than 180.

⑪ If the value of variable “c” is greater than 180, set the value of variable “c” to 0 to make the servo start rotate from 0 degrees.

⑫ Otherwise, add 1 to the value of variable “c” to add 1 degree to the rotation angle of the servo each time.

⑬ After changing the value of the variable “c”, set the servo that connected to pin 25 rotates to “c” degree so that the angle of the servo can change in real-time according to the value of the variable “c”.

⑭ Assign the value of the laser ranging sensor to the string variable “a”, which is used to display on the TFT screen.

⑮ Assign the value of the laser ranging sensor to an integer variable “b”, which is used for arithmetic operations.

⑯ Before displaying the value of the laser ranging sensor, use space characters to clear the content on the coordinates (50, 50) so that the value can be displayed later.

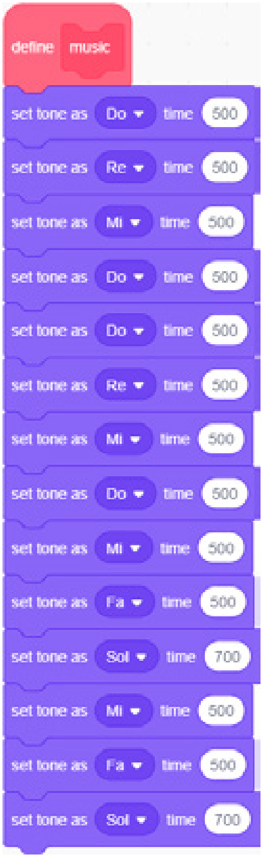
⑰ After clearing, the laser ranging value will be displayed on the coordinates (50, 50).

⑱ What is used for arithmetic judgment must be an integer variable, so here we judge the value of the integer variable “b”.

⑲ If it is within the judgment condition, execute the music module. Note that the music module will no exist if you haven’t created a self-made building block and name it music. Therefore, we need to create a self-made building block and name it music. After completing step 19, we can add the program to the music module.

The music module is a packaged "alarm", the alarm can be various and can be customized with the tone module. A two-tiger tone was made here as an "alarm".

In the music module:



```
define music
set tone as Do time 500
set tone as Re time 500
set tone as Mi time 500
set tone as Do time 500
set tone as Do time 500
set tone as Re time 500
set tone as Mi time 500
set tone as Do time 500
set tone as Mi time 500
set tone as Fa time 500
set tone as Sol time 700
set tone as Mi time 500
set tone as Fa time 500
set tone as Sol time 700
```

Picture 5-3

The program in the music module can be customized. You can create your own music using 7 notes.

You can find the complete program of this case in the downloaded Master kit folder: Intrusion Scanner.sb3, and open it with letscode.

2. Download

After finishing the program, download it to the ESP32 module (if you forget the steps to download the program, please refer to it in Lesson 1). After downloading the program, remember to restart the ESP32 module, now we have finished the intrusion scanner.

6. Brainstorming

After learning how to use the program to play scores, can we change the alarm sound? Try to find some sheet music on the Internet, and then use the program to achieve it!

Lesson 6 Radar

1. Introduction

You may often see radar in military programs and related movies. Do you know what radar is mainly used for? How does it work?

Radar is an electronic device for detecting and ranging by radio. It is widely used in military combat command and navigation guidance of civil aircraft. It is an indispensable detection and monitoring device. The working principle of radar is to illuminate and scan the target object by emitting electromagnetic waves that we can't see and receive its echo, and obtain the distance from the target object to the electromagnetic wave launching point, distance change rate, azimuth and height.

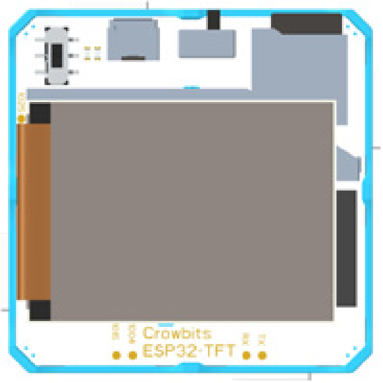
Wow, the radar is so mysterious and indispensable, so let's learn the working principle of simulated radar and make a radar of our own!

2. Learning Goals

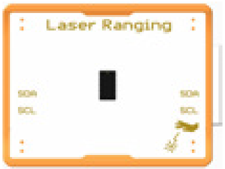
- Know about the working principle of radar.
- Master the steps of uploading radar code.
- Complete a radar project.

3. Hardware List

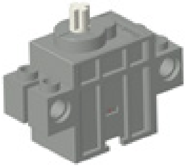
1. ESP32 Module



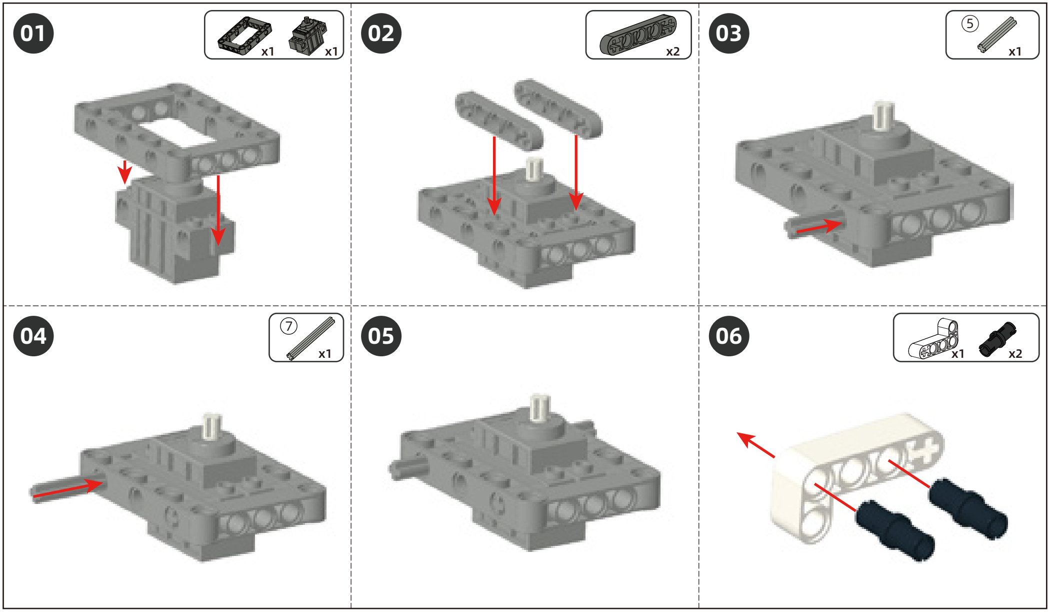
2. Laser Ranging Sensor



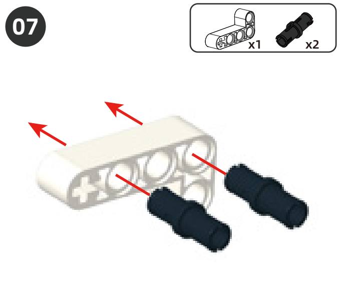
3. Servo Module



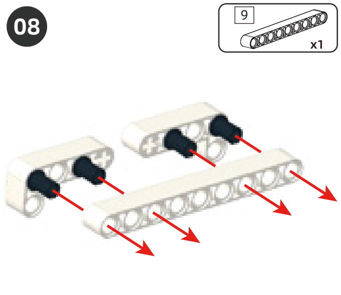
Structure building steps:



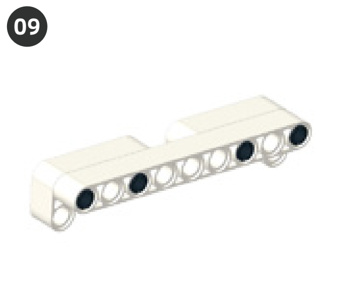
07



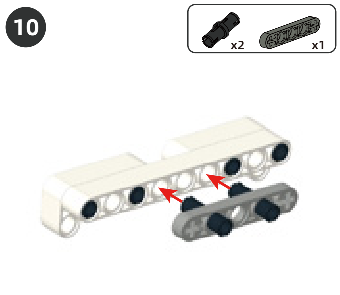
08



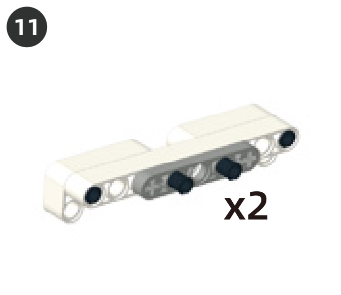
09



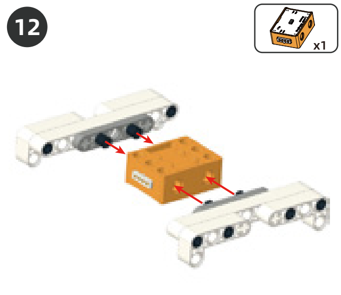
10



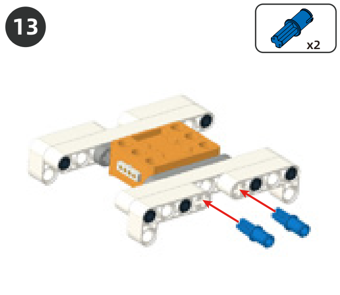
11



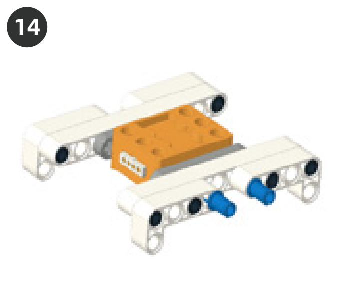
12



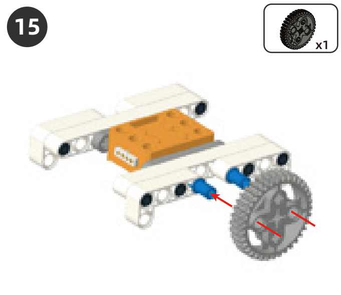
13



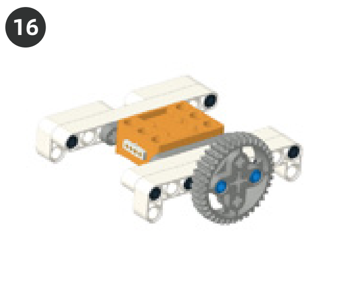
14



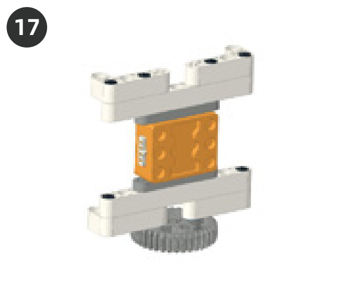
15



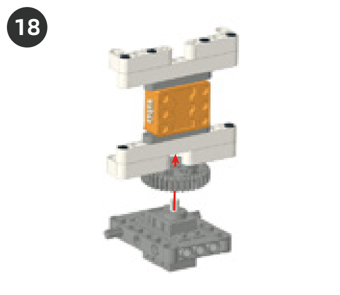
16



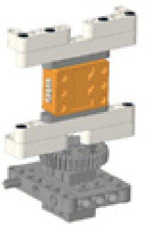
17



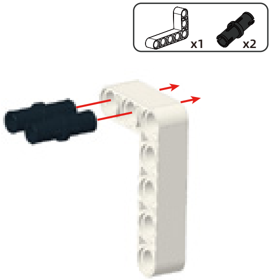
18



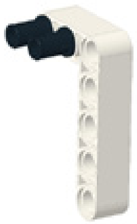
19



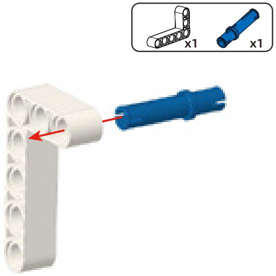
20



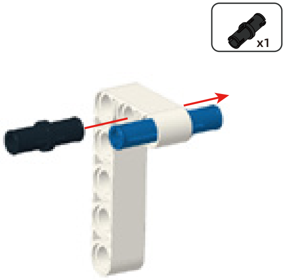
21



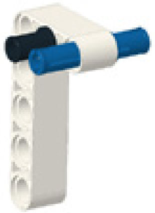
22



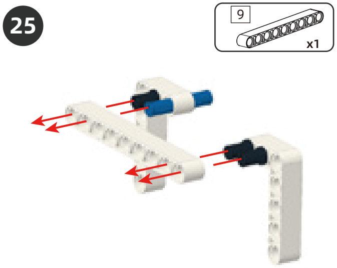
23



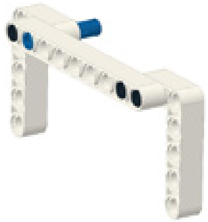
24



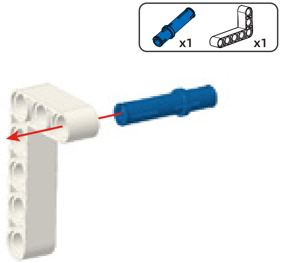
25



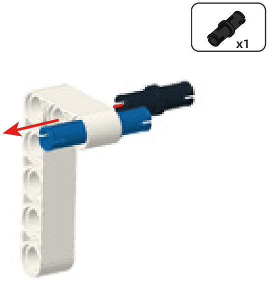
26



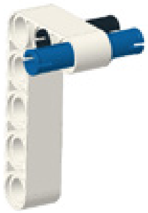
27



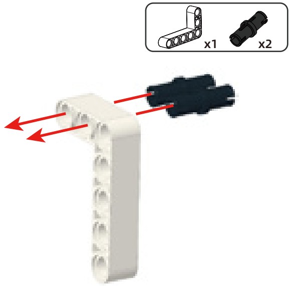
28



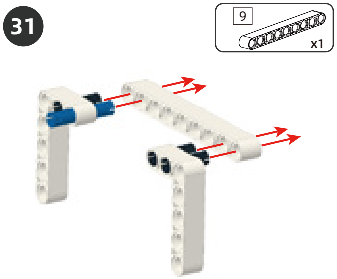
29



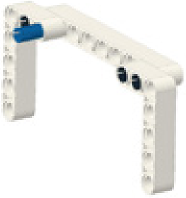
30



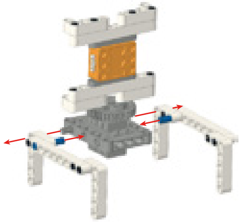
31



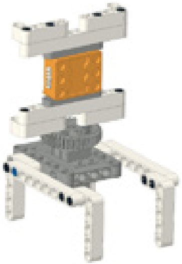
32



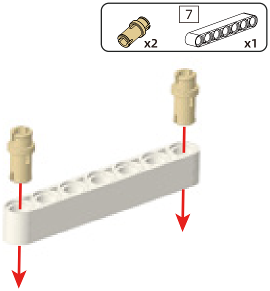
33



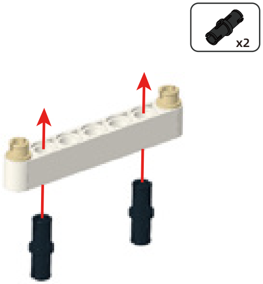
34



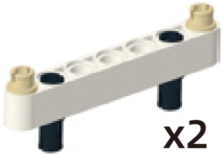
35



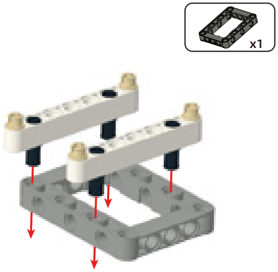
36



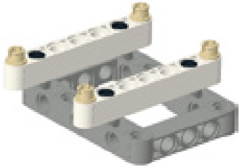
37



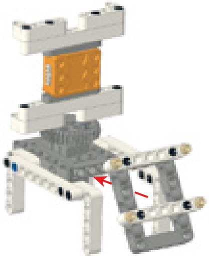
38



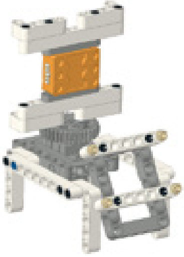
39



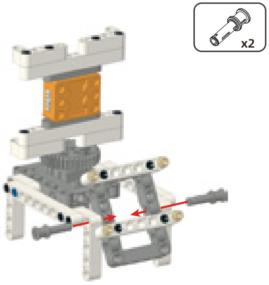
40

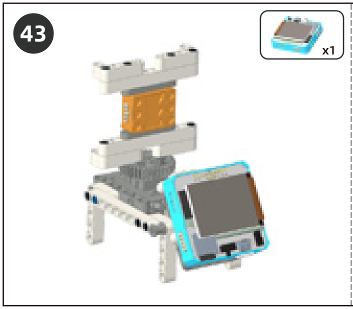


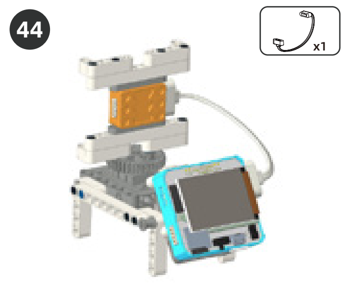
41

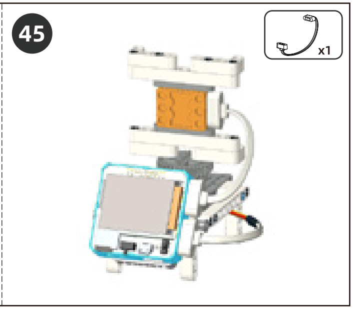


42









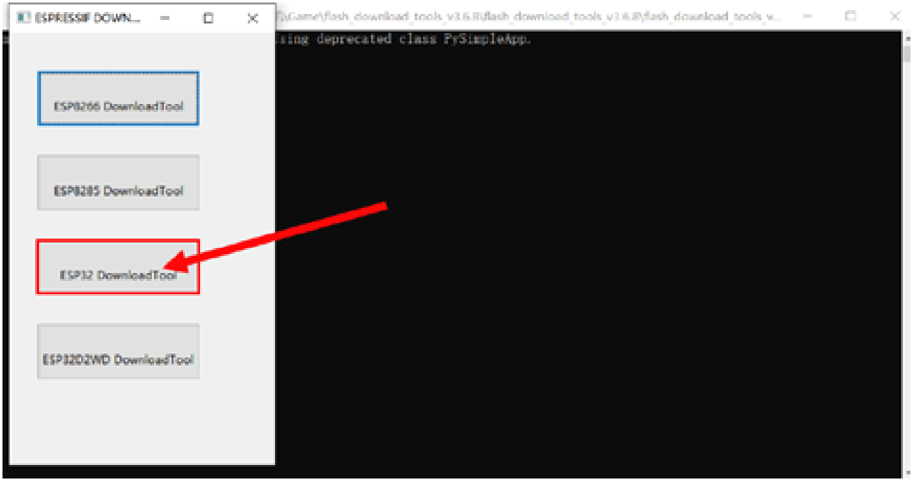
Radar connection instructions: connect the laser ranging sensor to 13, 14 pins of the ESP32 module, and connect the servo to 25, 34 pins of the ESP32 module.

5. Project Realization

1. In the Master kit folder, find the software as shown picture below in the flash_download_tools_v3.6.8 folder and double-click to open it.

flash_download_tools_v3.6.8

2. After opening the software, click the ESP32 DownloadTool as shown in picture 6-1 below.



```
sing deprecated class PySimpleApp.
```

Picture 6-1

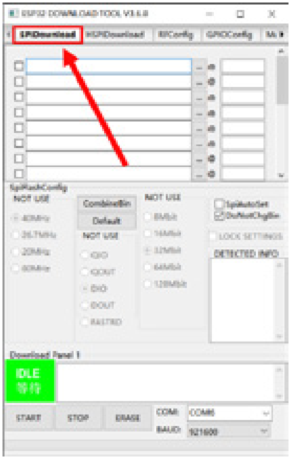
3. Choose to download in SPIDownload mode, as shown in picture 6-2 below.

4. Click the icon ... to select the file to upload, and write the corresponding address, as shown in picture 6-3 below.

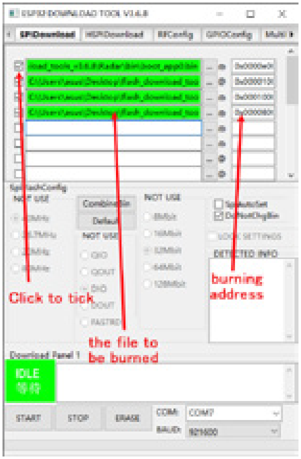
Burn the files and corresponding addresses from top to bottom:

1: File name: boot_app0.bin	address: 0x0000e000
2: File name: bootloader_dio_40m.bin	address: 0x00001000
3: File name: CrowRader.ino.espea32.bin	address: 0x00010000
4: File name: default.bin	address: 0x00008000

Note: Click the box in front of the file name to tick so that the burned file can be downloaded effectively.



Picture 6-2



Click to tick

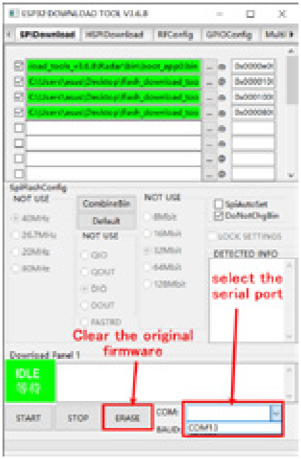
the file to be burned

burning address

Picture 6-3

5. Select the correct serial port and click the ERASE button to clear the original firmware, as shown in picture 6-4.

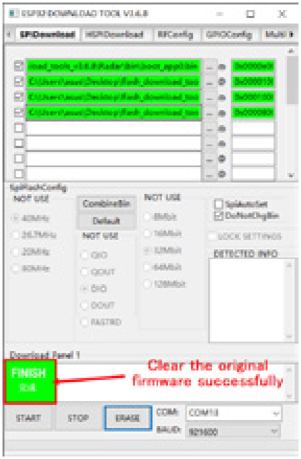
6. The original firmware is successfully cleared, as shown in picture 6-5 below.



select the serial port

Clear the original firmware

Picture 6-4

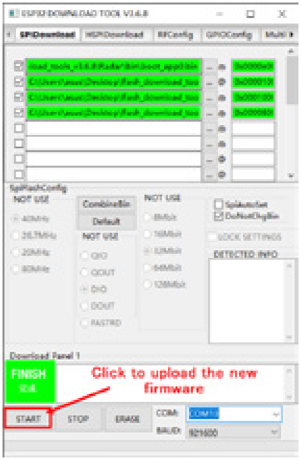


Clear the original firmware successfully

Picture 6-5

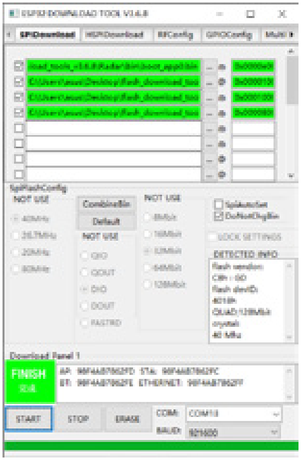
7. After clearing the original firmware, click the START button to upload the new firmware, as shown in picture 6-6 below.

8. Uploading the new firmware successfully, as shown in picture 6-7 below.



Click to upload the new firmware

Picture 6-6



Picture 6-7

9. Restart the ESP32 module to make the firmware work.

6. How to use

1. Toggle the mode switch of the ESP32 module to normal mode(upper) and place the radar on the floor.
2. Place obstacles at a distance of 50-100 cm in front of the radar
3. Toggle the power switch of the ESP32 module to ON to start the radar.

All the obstacles in front of the radar will display on the screen, which will be shown as dots in different positions of the screen base on the distance from the radar. When the radar scan angle reaches 180 degrees, you will see all the obstacles in front of the radar. How interesting it is! Let's make it!

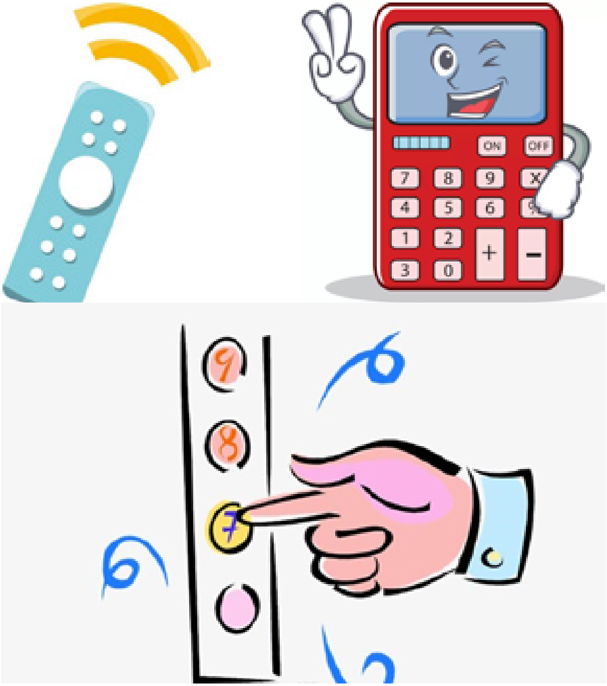
7. Brainstorming

How can we make the radar more accurate? The dots displayed on the radar screen can be connected in a line? What will the effect be if the laser distance sensor is replaced by an ultrasonic sensor?

Lesson 7 Keyboard

1. Introduction

The button is a commonly used control electrical component, which is widely used in various applications in life. For example, TV remote controls, calculators, elevator buttons, etc.



The button is a simple switch structure. When it is connected to the circuit, it can control the on and off of the circuit, send signals, set the working mode of the machine, etc. The button can simplify some complicated and difficult to operate and explain processes. Just press a button to complete.

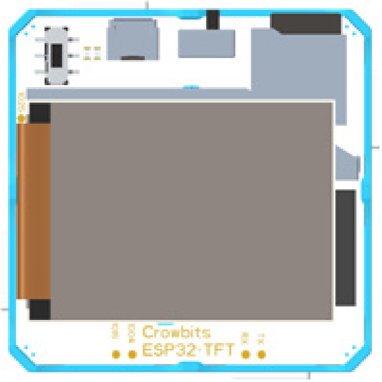
In this lesson, we will learn about Crowbit's Keyboard module, including the principle and use of buttons on it.

2. Learning Goals

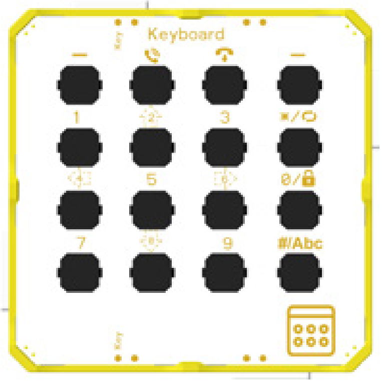
- Master the working principle of the Keyboard module.
- Master how to read the analog value of the Keyboard module.
- Master how to use the Keyboard to switching pictures on ESP32.

3. Hardware List

1. ESP32 Module



2. Keyboard Module



4. Knowledge Analysis

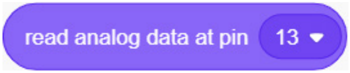
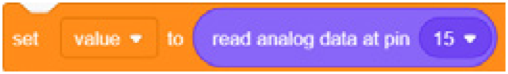
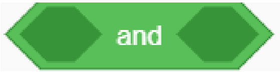
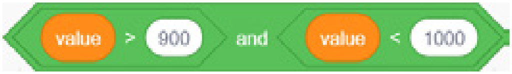
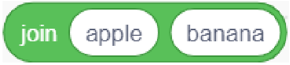
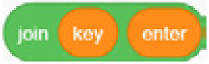
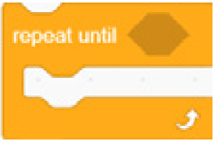
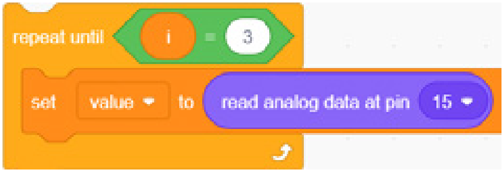
Let's learn about the Keyboard module and new code blocks that will be used in this lesson!

1. Hardware Introduction

Keyboard module: It is a 4x4 input Crowbit's module with 16 keys in total. The resistance of each button on the module is different, and when different buttons are pressed, the output voltage value will be in a different range. In the program, it can be judged which button is pressed by comparing the voltage value in which range. There are a total of 16 buttons on the Keyboard module, which means there will be 16 different analog value ranges.

2. Blocks Explanation

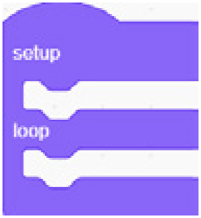
Table 7-1 Blocks explanation

Block	Example
 Explanation: Read the analog value of analog pin 13; click the inverted triangle to select the pin	 Read the analog value of pin 15 and assign it to the variable “value”
 Explanation: Logical AND operation; return true only if the operation result of both sides are true, return false if the operation result of any side is false	 Judge whether the variable “value” is greater than 900 and less than 1000; returns true if it is, otherwise, returns false
 Explanation: Connect two strings; fill in the string to be connected in the white circle, you can fill in any variable type in the circle, and the variable type will be converted to a string type and then connected	 Connect variable “key” and variable “enter”; no matter what type of “key” and “enter” are, they will be converted to strings and then connected together
 Explanation: Control module; repeat the code in the block until the condition is met, and then jump out of the loop	 repeat until i = 3 set value to read analog data at pin 15 Repeat to read the analog value of pin 15 and assign it to the variable “value”, until i=3, jump out the repeat execution

3. Determine the key analog value range

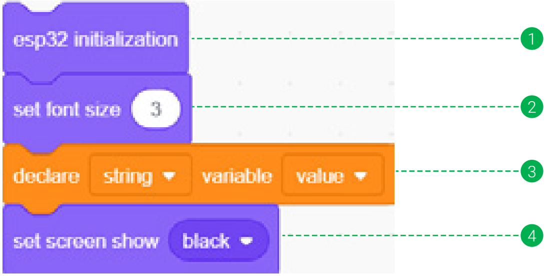
How to determine the range of the analog value of each pressed key? The following will explain how to read and determine the analog value range when the button is pressed.

1. Add setup/loop block



```
setup

loop
```

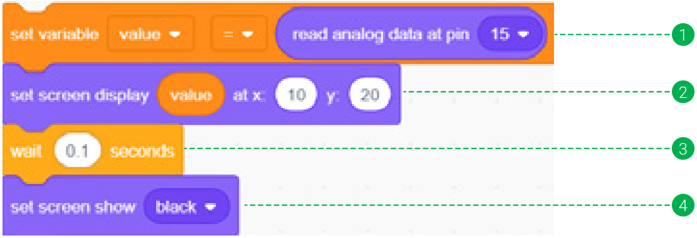
2. In the setup area:



```
esp32 initialization
set font size (3)
declare [string ▾] variable [value ▾]
set screen show [black ▾]
```

① Initialize the esp32 module to use the screen display text later.
② Set the font size displayed on the screen to 3.
③ Create the variable “value” and declare it as a string type.
④ Set the screen background to black.

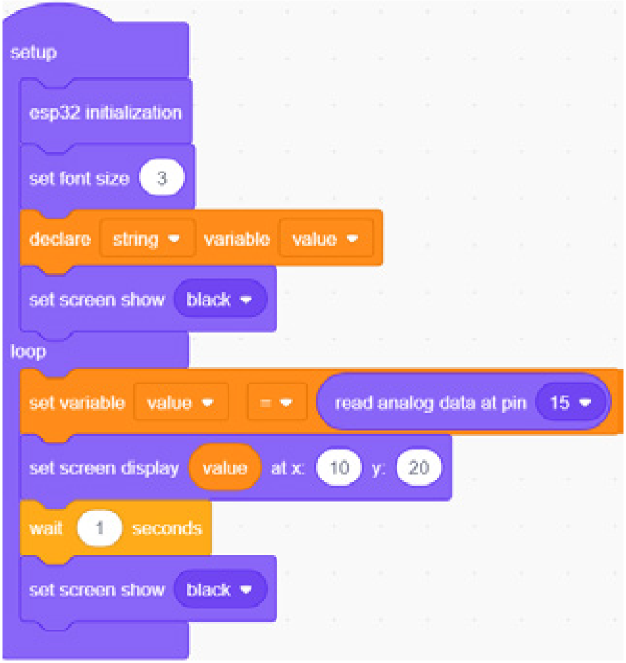
3. In the loop area:



```
set variable [value ▾] [= ▾] (read analog data at pin [15 ▾])
set screen display (value) at x: (10) y: (20)
wait (0.1) seconds
set screen show [black ▾]
```

① Read the analog value of the Keyboard from pin 15 and assign it to the variable “value”.
② Display the variable “value” on the screen coordinates X=10, Y=20.
③ Wait for 0.1 second so that we can see the displayed value clearly.
④ Set the screen background to black to refresh the screen display.

The complete code to read the key analog value range is as follows:



```
setup
    esp32 initialization
    set font size (3)
    declare [string ▾] variable [value ▾]
    set screen show (black ▾)
loop
    set variable [value ▾] [= ▾] (read analog data at pin (15 ▾))
    set screen display (value) at x: (10) y: (20)
    wait (1) seconds
    set screen show (black ▾)
```

The read analog value is not a fixed value, but a range, so you should use the displayed value as an intermediate value and decrease it to the left and expand it to the right to determine the analog value range when a button is pressed.

For example, the value displayed on the screen is 935 when button 1 is pressed; the value displayed on the screen is 1190 when button 2 is pressed; the value displayed on the screen is 1425 when button 3 is pressed.

Then the value range when button 1 is pressed can be set to 820-1050; the value range when button 2 is pressed is 1100-1270; the value range when button 3 is pressed is 1320-1530. Do it yourself to record the analog value range of each button.

Note: The value range of the keys cannot overlap, otherwise the keys will be misjudged to be pressed.

The following table lists the analog value range of each button being pressed:

Table 7-2 Analog value of keys

Key symbol	Analog value range	Key symbol	Analog value range
-(left)	0-19	4	1850-2030
📞	70-180	5	2080-2240
☎	300-500	6	2300-2510
-(right)	550-750	0	2560-2780
1	820-1050	7	2830-3050
2	1100-1270	8	3100-3330
3	1320-1530	9	3400-3630
✳	1600-1800	#/Abc	3700-3900

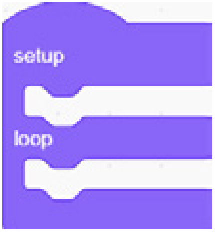
Note: The above correspondence table is the keys and the analog values read when the ESP32 module fully charged. The analog value range of different modules and battery level will be different, the specific analog value range is subject to your measurement.

5. Project Realization

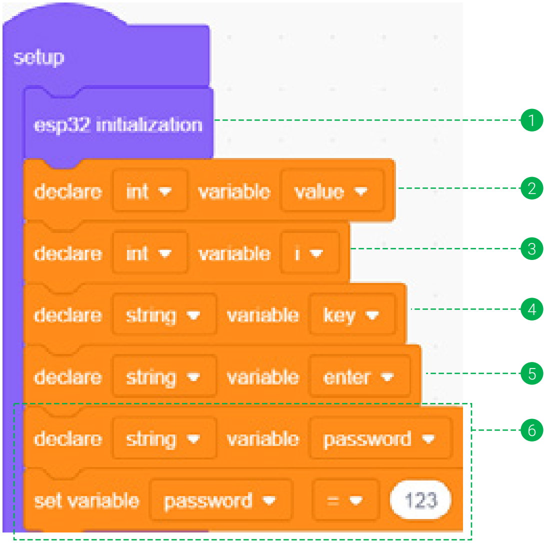
In this lesson, we will use the ESP32 module and the Keyboard module to make a secure photo viewer prevent strangers from viewing our private photos: the screen of the ESP32 module will display a public picture first, and only when you press the button to enter the preset password, the screen will switch to personal photos for viewing.

1. Programming

1. Add setup/loop block



2. In the setup area:

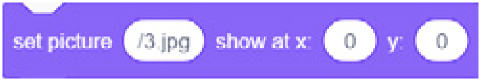


```
setup
  esp32 initialization
  declare int variable value
  declare int variable i
  declare string variable key
  declare string variable enter
  declare string variable password
  set variable password = 123
```

1. Initialize the ESP32 Module so that we can use the screen to display pictures later.
2. Declare the integer variable “value” to store the analog value when the key is pressed.
3. Declare an integer variable “i” to store the number of digits input by keys.
4. Declare the string variable “key” to store the entered password.
5. Declare the string variable “enter” to store the input that is pressed each time.
6. Declare the string variable “password” and assign it to 123 as the password to switch pictures.

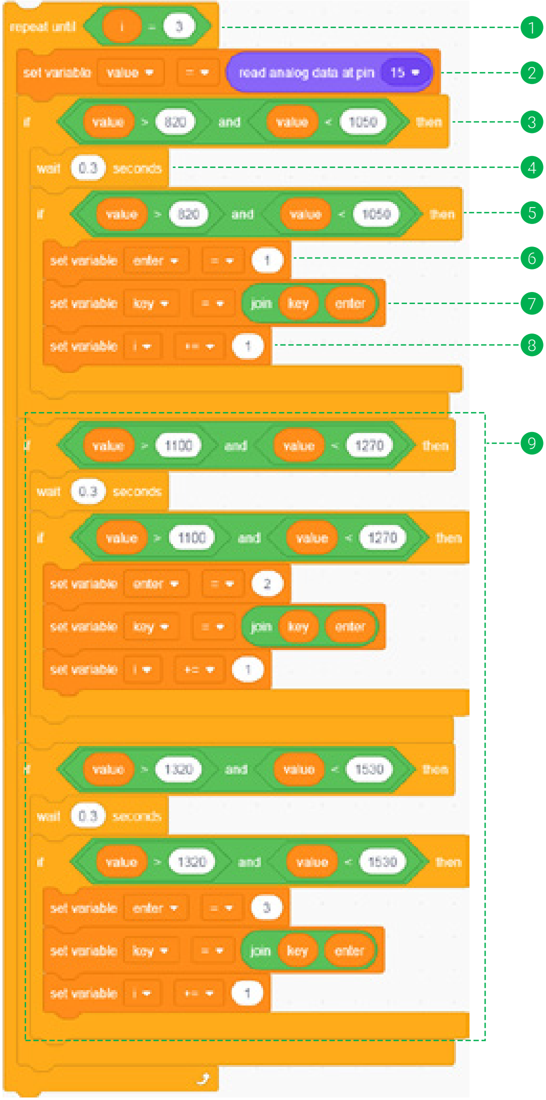
3. In the loop area, it is mainly divided into 3 steps to build blocks:

Step 1: Let the screen display the picture named 3. jpg in the memory card at the coordinates X=0, Y=0 as the starting point.



```
set picture /3.jpg show at x: 0 y: 0
```

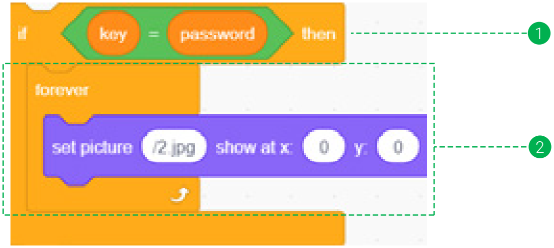
Step 2: Receive the three-digit password entered through the keys



```
repeat until  i = 3
  set variable  value ▾  = ▾  read analog data at pin 15 ▾
  if  value > 820  and  value < 1050  then
    wait 0.3 seconds
    if  value > 820  and  value < 1050  then
      set variable  enter ▾  = ▾  1
      set variable  key ▾  = ▾  join key enter
      set variable  i ▾  += ▾  1
  if  value > 1100  and  value < 1270  then
    wait 0.3 seconds
    if  value > 1100  and  value < 1270  then
      set variable  enter ▾  = ▾  2
      set variable  key ▾  = ▾  join key enter
      set variable  i ▾  += ▾  1
  if  value > 1320  and  value < 1530  then
    wait 0.3 seconds
    if  value > 1320  and  value < 1530  then
      set variable  enter ▾  = ▾  3
      set variable  key ▾  = ▾  join key enter
      set variable  i ▾  += ▾  1
```

1. Repeat the following code blocks until the variable “i” that stores the number of digits input by keys is equal to 3.
2. Read the analog value of Keyboard at pin 15 and assign it to the variable “value”.
3. Use the "if/then" code block to judge whether the value of variable “value” is greater than 820 and less than 1050 to determine whether button 1 is pressed.
4. Add a delay of 0.3 seconds to prevent repeated reading of the pressed button.
5. Use the "if/then" code block again to judge whether the value of variable “value” is greater than 820 and less than 1050 to determine whether button 1 is pressed.
6. After button 1 is confirmed to be pressed, 1 is assigned to the string “enter”.
7. Connect the variable “key” that stores the password with the variable “enter” that stores each time key input and re-assign the value to the variable “key”.
8. Increase the value of “i” by 1
9. According to the steps 3 - 8, judge whether button 2 and button 3 are pressed.

Step 3: Determine if the password entered is the same as the password set



```
if key = password then
    forever
        set picture /2.jpg show at x: 0 y: 0
```

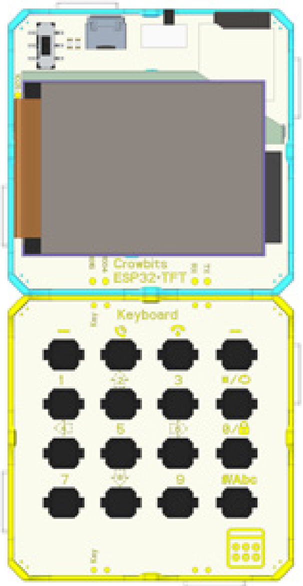
① Use the "if/then" block to determine whether the variable “key” that stored password input by key is the same as the preset password variable “password”.

② If yes, use the "forever" block to display the picture named 2. jpg in the memory card at coordinates X=0, Y=0 as the starting point.

You can find the complete program of this case in the downloaded Master kit folder: Keyboard.sb3, and open it with letscode.

2. Download

Follow the steps below to upload the code to the ESP32 module, and make your safe photo viewer!



① Name the two pictures 2. jpg and 3. jpg respectively and put them into the memory card. (3. jpg is a public picture, and 2. jpg is a private picture)

② Upload the program to the ESP32 module.

③ Connect the Keyboard module to pin 15 of the ESP32 module. (as shown in picture)

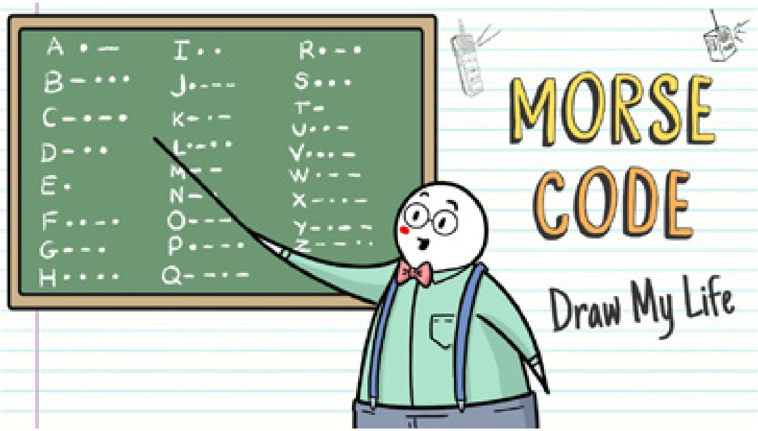
④ Restart the ESP32 module.

⑤ Use the Keyboard module to enter the password 123 to see what happens!

6. Brainstorming

In this lesson, we mainly learned the principle and usage of the keyboard module. But the function and usage of the button are not just to judge whether it is pressed or used as a switch, it can also be used in many usage scenarios and fields, such as the world's earliest communication method that uses electricity to transmit information-telegraph.

Telegraph transmits information (Morse code) through the combination of the length of the key press. You can add multiple pictures first, and then create different functions for the long press and short press of the key to realize fast and slow browsing of pictures; that is, long press to switch pictures constantly, while short press is to switch pictures one by one. Go ahead and implement it!



Lesson 8 Send Messages

1. Introduction

The mobile phone is the most common electronic communication tool in our daily life. It has a variety of functions, such as sending text messages and making phone calls, making people who are thousands of miles away can communicate with each other. So, do you know why mobile phones can achieve this function? It turns out that mobile phone communication relies on the wireless communication module inside, which enables the mobile phone to have the ability to transmit information on the Internet.

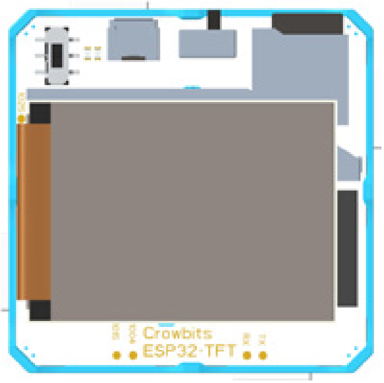
Next, what we are going to learn is the 2G module that enables mobile phones to have communication capabilities. Let's learn how it works and how to use it to send text messages!

2. Learning Goals

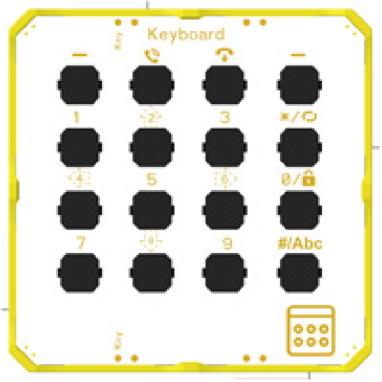
- Know about the communication principle of mobile phones.
- Know about the working principle of 2G module.
- Master how to use the program to send SMS to a specific mobile phone number.
- Master how to use the Keyboard to enter the phone number and send text messages.

3. Hardware List

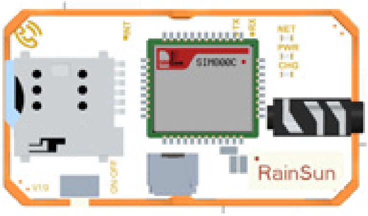
1. ESP32 Module



2. Keyboard Module



3. 2G Module



4. Knowledge Analysis

Principle of mobile phone communication

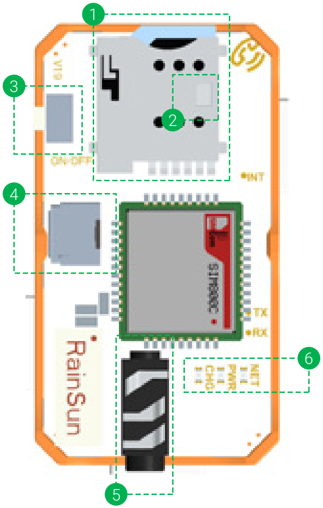
Communication refers to the exchange and transmission of information between people through a certain behavior or medium. Communication requires accurate and safe transmission of information from one party to the other. At present, the most common communication tool in our lives is the mobile phone.

Communication modules are used for communication between mobile phones, which mainly include 2G, 3G, 4G and 5G modules. The communication module uses radio to exchange information with nearby base stations, and then communicates with other mobile phones within the communication range through the base station's high-power system. So no matter how far apart, as long as they are within the coverage of the base station, mobile phones can communicate.

Let us continue to learn the 2G module and new code blocks that will be used in this lesson!

1. Hardware Introduction

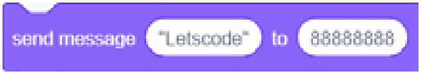
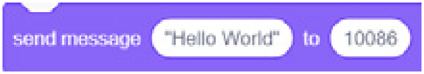
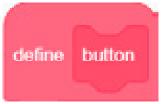
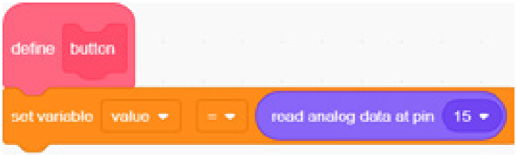
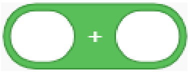
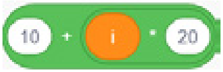
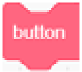
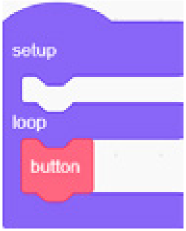
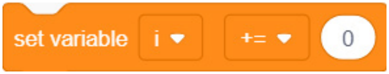
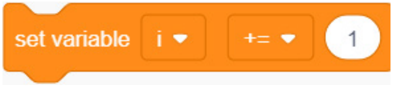
2G module: It is a serial module. This module can perform 2G signal communication to the input number, including functions such as sending text messages and making calls. The 2G module has a phone card slot, a power switch button, a module charging interface, a headphone jack and three status indicators. Their functions are as follows:



1. Phone card slot; before using the 2G module for communication, you need to insert the phone card into this card slot.
2. Phone card insertion reminder icon; the direction of the phone card inserted into the card slot must be consistent with the reminder icon. Pay attention to the direction of the phone card notch.
3. Power switch button; 2G module uses an independent power supply, you can turn on or off the 2G module by pressing and holding this button for 5 seconds.
4. Charging interface; when the battery is low, charge the 2G module through this Micro-USB interface.
5. Headphone jack; when using the 2G module to make a call, insert a headset with a microphone into this interface to make a call.
6. Three indicator lights; from left to right are 2G module charging indicator, 2G module power indicator, 2G module network indicator.

2. Blocks Explanation

Table 8-1 Blocks explanation

Block	Example
 send message "Letscode" to 88888888 Explanation: Send the text messages to the specific phone number; fill in the text message to be sent in the first white circle, and fill in the phone number in the second white circle. Strings (must be enclosed in double quotation marks) or string variables must be filled in the first white circle	 send message "Hello World" to 10086 Send SMS "Hello World" to phone number 10086
 define button Explanation: Define a self-made building block; under the self-made building block, you can join many blocks according to the function you want to achieve. "button" is the name of the self-made building block, which can be modified yourself	 define button set variable value = read analog data at pin 15 Define the function of the self-made building block as reading the analog value of pin 15 and assign it to the variable "value"
 + Explanation: Multiplication operation; fill in the number or variable you want to operate in the circle	 10 + i * 20 The result of multiplying variable "i" and 20 is added to 10
 button Explanation: Call the self-made building block; after the self-made building block is defined, drag this block into the program to call the function defined by the self-made building block	 setup loop button Call the self-made building block function named button in the loop
 set variable i += 0 Explanation: Set the variable i=i+0; drop down the inverted triangle to select the operation method, fill in the number of superimpositions in the circle	 set variable i += 1 Set the value of variable "i" to i+1, that is, i=i+1

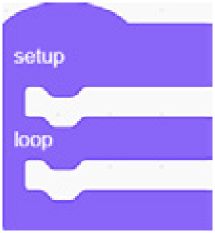
5. Project Realization

There will be two cases in this lesson. Case 1 needs to use ESP32 module, and 2G module to send SMS directly to a fixed phone number through blocks; Case 2 needs to use ESP32 module, Keyboard module and 2G module. Enter the phone number by pressing the button on Keyboard module, the message will be sent to the entered phone number after pressing the send button.

1. Programming

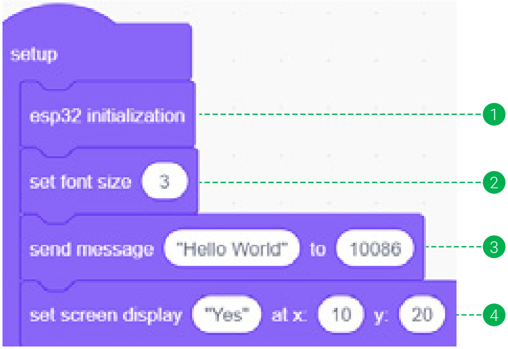
Case 1

1. Add setup/loop block



```
setup
loop
```

2. In the setup area:



```
setup
  esp32 initialization
  set font size 3
  send message "Hello World" to 10086
  set screen display "Yes" at x: 10 y: 20
```

1 Initialize the ESP32 module for display text on the screen later.

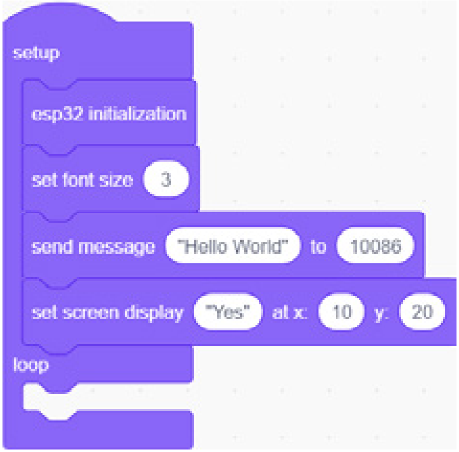
2 Set the font size displayed on the screen to 3.

3 Send the text message "Hello World" to the phone number 10086.

4 Display the text "Yes" on the screen at coordinates X=10, Y=20.

3. In the loop area: no need to add blocks

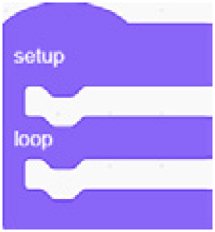
The complete code of the case 1 is as follows:



```
setup
  esp32 initialization
  set font size 3
  send message "Hello World" to 10086
  set screen display "Yes" at x: 10 y: 20
loop
```

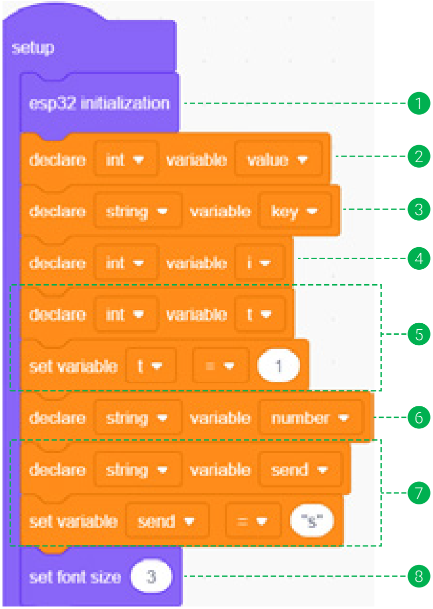
Case 2

1. Add setup/loop block



```
setup
loop
```

2. In the setup area:



```
setup
  esp32 initialization
  declare int variable value
  declare string variable key
  declare int variable i
  declare int variable t
  set variable t = 1
  declare string variable number
  declare string variable send
  set variable send = "s"
  set font size 3
```

1 Initialize the esp32 module to display the entered phone number later.

2 Declare the integer variable “value” to store the analog value of the pressed button in the self-made building block “button”.

3 Declare the string variable “key” to store the pressed key.

4 Declare the integer variable “i” to store the sequence number of the pressed key to determine the coordinate position of the key displayed on the screen.

5 Declare the integer variable “t” as 1 so that the key can be added to the number only when the key is pressed.

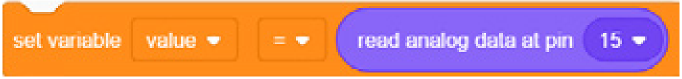
6 Declare the string variable “number” to store the phone number.

7 Declare the string variable “send” to "s" as the SMS sender, that is, send the SMS when the pressed key is "s".

8 Set the font size displayed on the screen to 3.

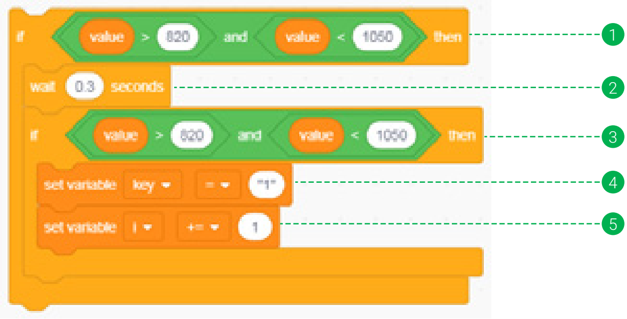
3. Self-made building block “button", mainly used to determine which button is pressed. It can be divided into 3 steps:

Step 1: Read the analog value of Keyboard at pin 15 and assign it to the variable “value”.

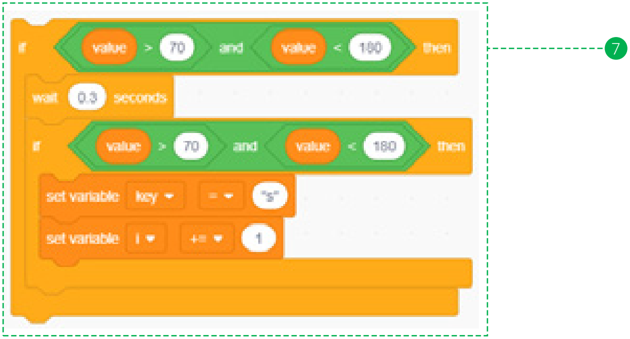


```
set variable value = read analog data at pin 15
```

Step 2: Determine which button is pressed according to the value of variable “value”



```
if <(value) > (820) and (value) < (1050)> then
    wait (0.3) seconds
    if <(value) > (820) and (value) < (1050)> then
        set variable [key v] [= v] ("1")
        set variable [i v] [+= v] (1)
```



```
if <(value) > (70) and (value) < (180)> then
    wait (0.3) seconds
    if <(value) > (70) and (value) < (180)> then
        set variable [key v] [= v] ("s")
        set variable [i v] [+= v] (1)
```

1 Use the "if/then" block to judge whether the variable “value” is greater than 820 and less than 1050 to determine whether button 1 is pressed.

2 Add a delay of 0.3 seconds to prevent repeated reading of the pressed button.

3 Use the "if/then" code block again to judge whether the variable “value” is greater than 820 and less than 1050 to determine whether button 1 is pressed.

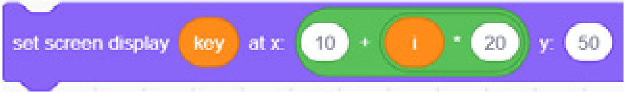
4 After the button 1 is confirmed to be pressed, 1 is assigned to the string variable “key” that stores the key pressed each time.

5 Increase the value of “i” by 1 to display the key pressed this time in the corresponding coordinate on the screen.

6 According to the previous 1 - 5 above, judge whether buttons 2-9 and button 0 are pressed.

7 According to the previous 1 - 5 sequence above, judge whether the button “s” is pressed.

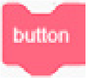
Step 3: Display the pressed key on the screen coordinates X=10+i*20, Y=50.



```
set screen display (key) at x: ((10) + ((i) * (20))) y: (50)
```

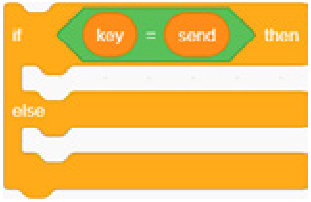
4. In the loop area, it is mainly divided into 4 steps to build code blocks:

Step 1: Call the self-made building block “button” to determine which button is pressed.



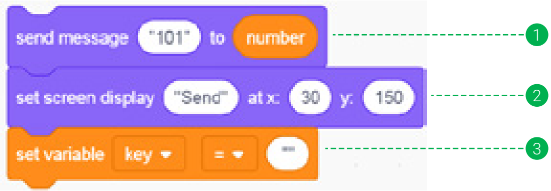
```
button
```

Step 2: Use the "if/then...else" code block to judge whether the variable “key” is the same as the string variable “send”.



```
if key = send then
else
```

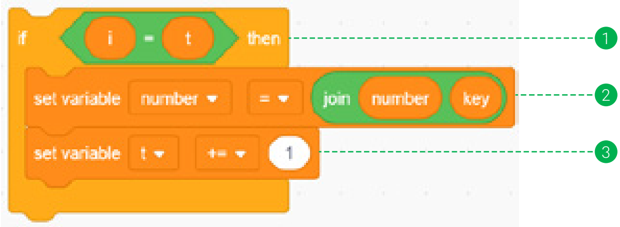
Step 3: If the “key” pressed is the same as the string variable “send”, press the SMS send key.



```
send message "101" to number
set screen display "Send" at x: 30 y: 150
set variable key = ""
```

① Send the message "101" to the phone “number”.
② The text "Send" is displayed at the coordinates X=30, Y=150 on the screen, which means the message has been sent.
③ Set the “key” to be empty to prevent repeated SMS.

Step 4: If the “key” pressed is not equal to the string variable “send”, it means that you are entering a phone number



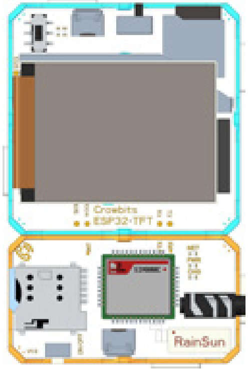
```
if i = t then
    set variable number = join number key
    set variable t += 1
```

① Use the "if/then" code block to judge whether “i” is equal to “t”, that is, whether the button has just been pressed.
② If yes, connect the “number” that storing the phone number with the key that storing the pressed button and re-store it as the phone “number”.
③ Increase the value of “t” to prevent the button from being connected to the variable “number” when the button is not pressed.

You can find the complete program of this case in the downloaded Master kit folder: Send messages.sb3, and open it with letscode.

2. Download

Follow the steps below to upload the code to the ESP32 module, and send a text message to a friend with your mobile phone!



Module Connection Diagram

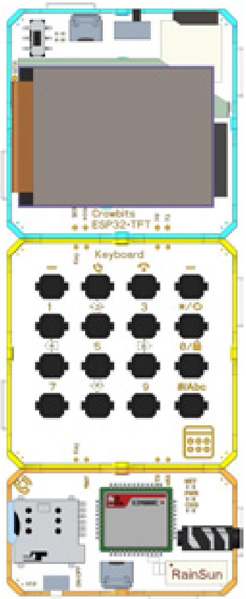
Case 1

① Build the program according to Case 1 and change the phone number to your friend's, then upload the program to the ESP32 module.

② Insert the phone card into the 2G module card slot, press and hold the power switch button of the 2G module to turn on the 2G module.

③ Connect the 2G module to the TX and RX pins of the ESP32 module.

④ Toggle the DIP switch on the upper left corner of the ESP32 to the 2G mode and restart the ESP32 module to check whether your friend has received the text message later.



Module Connection Diagram

Case 2

① Build the program according to Case 2, and upload the program to the ESP32 module.

② Connect the Keyboard module to pin 15 of the ESP32 module.

③ Insert the phone card into the 2G module card slot, press and hold the power switch button of the 2G module to turn on the 2G module.

④ Connect the 2G module to the Keyboard module.

⑤ Toggle the DIP switch on the upper left corner of the ESP32 to the 2G mode and restart the ESP32 module, use the Keyboard to enter the friend's phone number and send a text message!

6. Brainstorming

In this lesson, we mainly learned the principle of mobile phone communication, 2G module and how to use it to send SMS.

When using the Keyboard to enter the number, we may accidentally enter the wrong number. In order to solve this problem, you can add a deletion function, that is, when the wrong number is entered, press a key on the Keyboard (such as key “-”) to delete the previous number and re-enter the correct number.

Lesson 9 Dial

1. Introduction

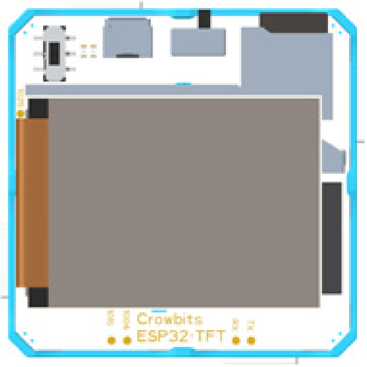
In the previous lesson, we have learned how to use the 2G module to send SMS. In addition to communicating via SMS, making phone calls is currently one of the most popular communication methods. Compared with text messages that require input word by word and convey information in the form of text, making a call allows people who are thousands of miles away to communicate through voice and in a faster, more efficient, and more vivid way to transfer information. In this lesson, we will learn how to use the 2G module to make calls through code blocks and Keyboard input.

2. Learning Goals

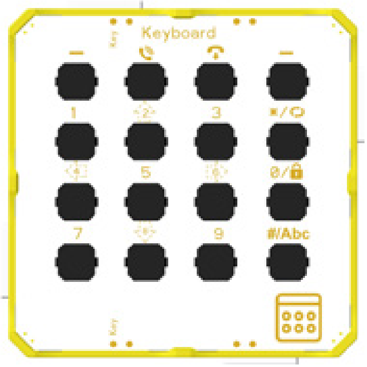
- Master how to use code blocks to make calls.
- Master how to use the Keyboard module to enter numbers and make calls.

3. Hardware List

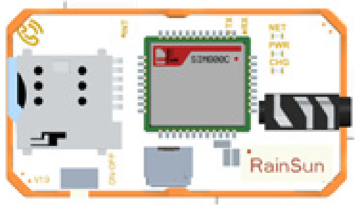
1. ESP32 Module



2. Keyboard Module



3. 2G Module

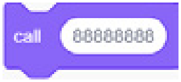
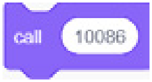
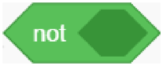
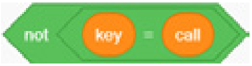


4. Knowledge Analysis

Let us learn the new code blocks that will be used in this lesson!

1. Blocks Explanation

Table 9-1 Blocks explanation

Block	Example
 call 88888888 Explanation: Make a call; fill in the phone number in the white circle, string and string variables can be filled in	 call 10086 Call to number 10086
 not Explanation: The condition is not established; fill in the judgment condition in the box, and return true when the judgment condition is not established, otherwise, return false	 not key = call Determine whether the variable “key” is equal to the variable “call”; if not, return true, otherwise, return false

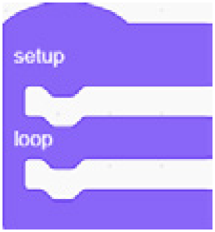
5. Project Realization

There will be two cases in this lesson. Case 1 needs to use ESP32 module and 2G module, and dail a fixed phone number through blocks; Case 2 needs to use ESP32 module, Keyboard and 2G module, enter the phone number by pressing the button on Keyboard module, After pressing the dial key, it will make a call to the entered phone number.

1. Programming

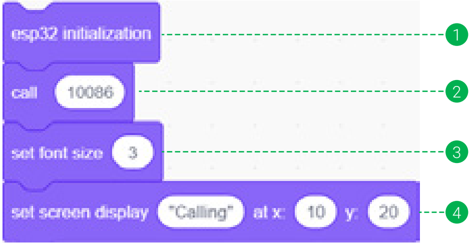
Case 1

1. Add setup/loop block



```
setup
loop
```

2. In the setup area:

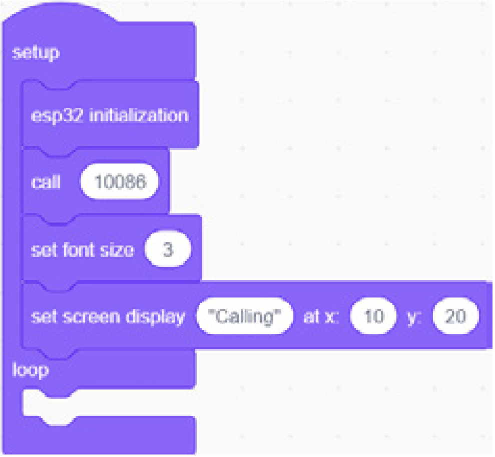


```
esp32 initialization
call 10086
set font size 3
set screen display "Calling" at x: 10 y: 20
```

1. Initialize the ESP32 module for display text on the screen later.
2. Dial the phone number 10086.
3. Set the font size displayed on the screen to 3.
4. Display the text "Calling" on the screen at coordinates X=10, Y=20.

3. In the loop area: no need to add blocks

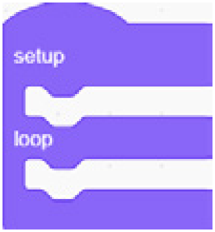
The complete code of the case 1 is as follows:



```
setup
    esp32 initialization
    call 10086
    set font size 3
    set screen display "Calling" at x: 10 y: 20
loop
```

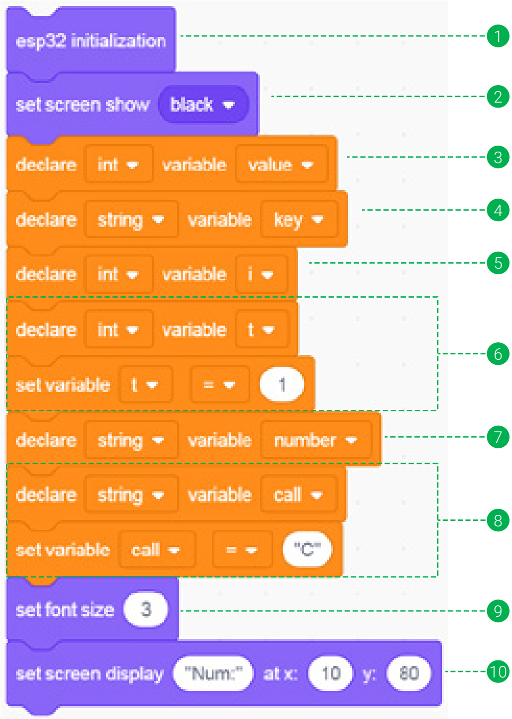
Case 2

1. Add setup/loop block



```
setup
loop
```

2. In the setup area:



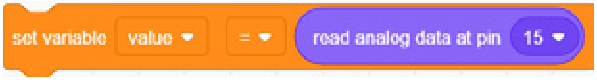
```
esp32 initialization
set screen show black
declare int variable value
declare string variable key
declare int variable i
declare int variable t
set variable t = 1
declare string variable number
declare string variable call
set variable call = "C"
set font size 3
set screen display "Num:" at x: 10 y: 80
```

1. Initialize the esp32 module to display the entered phone number later.
2. Set the screen background to black.
3. Declare the integer variable “value” to store the analog value of the pressed button in the self-made building block “button”.
4. Declare the string variable “key” to store the pressed key.
5. Declare the integer variable “i” to store the sequence number of the pressed key to determine the coordinate position of the key displayed on the screen.
6. Declare the integer variable “t” as 1 so that the key can be added to the number only when the key is pressed.
7. Declare the string variable “number” to store the phone number.
8. Declare the string variable “call” to "C" as the dial key, that is, dial the number when the pressed key is "C".
9. Set the font size displayed on the screen to 3.
10. Set the screen displays the text “Num” at coordinates X=10, Y=80, and the number to be dialed will display behind.

You can find the complete program of this case in the downloaded Master kit folder:Intrusion dialer.sb3, and open it with letscode.

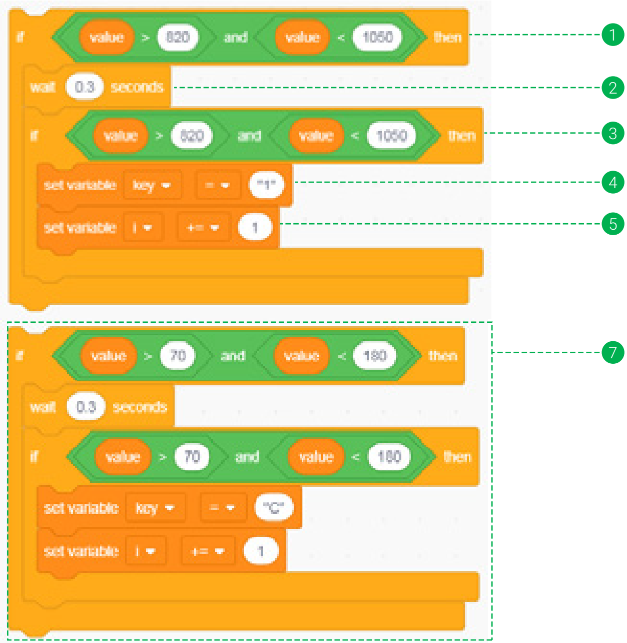
3. Self-made building block “button", mainly used to determine which button is pressed. It can be divided into 3 steps:

Step 1: Read the analog value of Keyboard at pin 15 and assign it to the variable “value”.



```
set variable value = read analog data at pin 15
```

Step 2: Determine which button is pressed according to the value of variable “value”



```
if value > 820 and value < 1050 then
    wait 0.3 seconds
    if value > 820 and value < 1050 then
        set variable key = "1"
        set variable i += 1

if value > 70 and value < 180 then
    wait 0.3 seconds
    if value > 70 and value < 180 then
        set variable key = "C"
        set variable i += 1
```

❶ Use the "if/then" block to judge whether the variable “value” is greater than 820 and less than 1050 to determine whether button 1 is pressed.

❷ Add a delay of 0.3 seconds to prevent repeated reading of the pressed button.

❸ Use the "if/then" code block again to judge whether the variable “value” is greater than 820 and less than 1050 to determine whether button 1 is pressed.

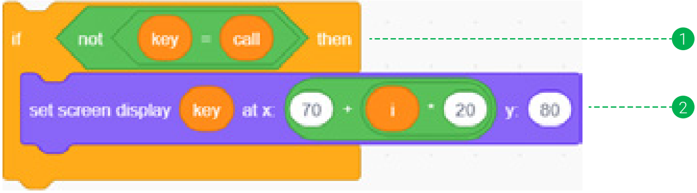
❹ After the button 1 is confirmed to be pressed, 1 is assigned to the string variable “key” that stores the key pressed each time.

❺ Increase the value of “i” by 1 to display the key pressed this time in the corresponding coordinate on the screen.

❻ According to the previous ❶ - ❺ sequence above, judge whether buttons 2-9 and button 0 are pressed.

❼ According to the previous ❶ - ❺ sequence above, judge whether the button “C” is pressed.

Step 3: Decide whether to display the key on the screen according to whether the key is a dial key

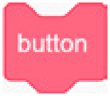


```
if not key = call then
    set screen display key at x: 70 + i * 20 y: 80
```

1. Use the "if/then" block to judge whether the pressed key is key “C”.
2. If it is, displays the pressed key on the screen coordinates X=70+i*20, Y=80.

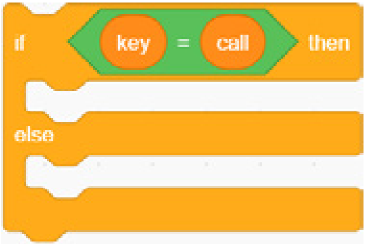
4. In the loop area, it is mainly divided into 4 steps to build code blocks:

Step 1: Call the self-made building block “button” to determine which button is pressed.



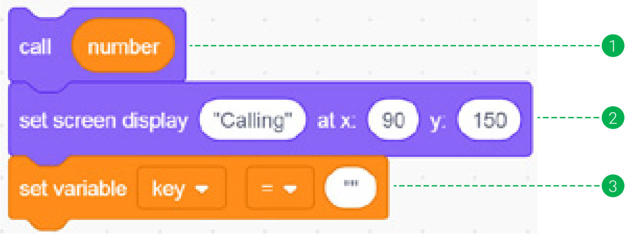
```
button
```

Step 2: Use the "if/then...else" code block to judge whether the variable “key” is the same as the string variable “call”.



```
if key = call then

else

```

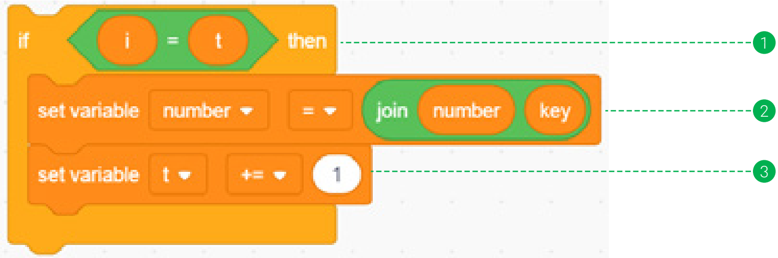
Step 3: If the pressed key is equal to the string variable “call”, it means that the dial key is pressed



```
call number
set screen display "Calling" at x: 90 y: 150
set variable key = ""
```

1. Dial the “number”.
2. Displays the text “Calling” on the screen coordinates X=90, Y=150 to show is dialing.
3. Set the “key” to be empty to prevent repeated dial.

Step 4: If the “key” pressed is not equal to the string variable “call”, it means that you are entering a phone number



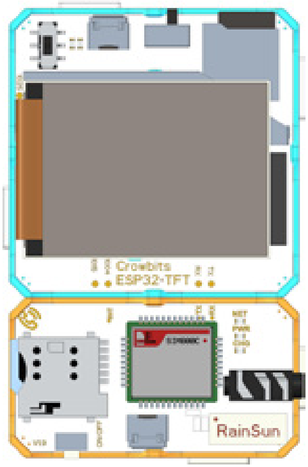
```
if i = t then
    set variable number = join number key
    set variable t += 1
```

1. Use the "if/then" code block to judge whether “i” is equal to “t”, that is, whether the button has just been pressed.
2. If yes, connect the “number” that storing the phone number with the key that storing the pressed button and re-store it as the phone “number”.
3. Increase the value of “t” to prevent the button from being connected to the variable “number” when the button is not pressed.

You can find the complete program of this case in the downloaded Master kit folder:Dial.sb3,and open it with letscode.

2. Download

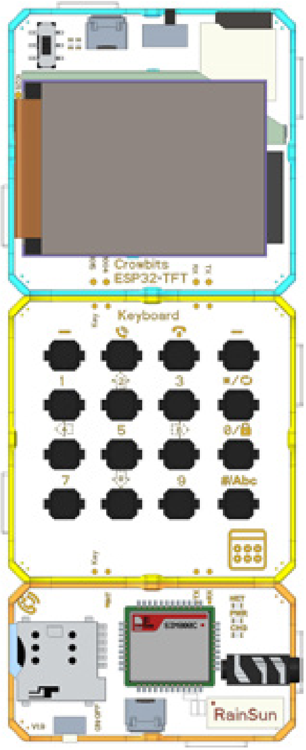
Follow the steps below to upload the code to the ESP32 module, and use it to make a call and chat with your friends!



Module Connection Diagram

Case 1

1. Build the program according to Case 1 and change the phone number to your friend's, and then upload the program to the ESP32 module.
2. Insert the phone card into the 2G module card slot, press and hold the power switch button of the 2G module to turn on the 2G module.
3. Connect the 2G module to the TX and RX pins of the ESP32 module.
4. Toggle the DIP switch on the upper left corner of the ESP32 to the 2G mode, restart the ESP32 module, and wait for your friend to pick up.



Module Connection Diagram

Case 2

① Build the program according to Case 2, and upload the program to the ESP32 module.

② Connect the Keyboard module to pin 15 of the ESP32 module.

① Insert the phone card into the 2G module card slot, press and hold the power switch button of the 2G module to turn on the 2G module.

④ Connect the 2G module to the Keyboard module.

⑤ Toggle the DIP switch on the upper left corner of the ESP32 to the 2G mode and restart the ESP32 module, use the Keyboard to enter your friend's phone number and make a call to chat!

6. Brainstorming

In this lesson, we mainly learned how to use the dial code block and dial it to your friends. We often dial the wrong number because we forget the phone numbers of family and friends. There are many ways to solve this problem, such as creating an address book, adding people you frequently contact to the address book, and recording their names and mobile numbers separately. After creating the address book, check the name of the person who owns the number, and you won't dial the wrong number anymore. Go ahead and implement this function!

Lesson 10 Speed dial

1. Introduction

For the family and friends who often contact, it is very helpless to waste time opening the address book and searching for contacts, and then typing the number, so is there a way that we can call them with just one key?

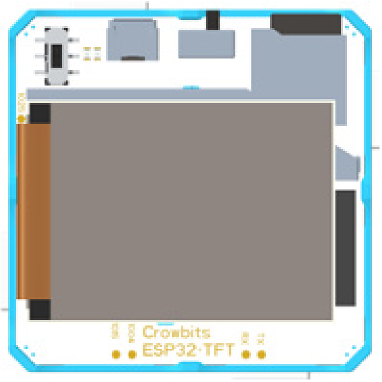
The following lesson will solve this problem. In the lesson, we will learn how to make a call with only one key, save the numbers of your family and best friends, and call to them by pressing a key. In addition, you just need to press a hang up key when you want to end the call.

2. Learning Goals

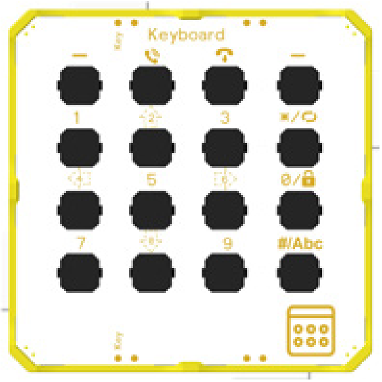
- Create a list of phone numbers for your family and friends.
- Realize one-key dialing function.
- Master the use of the "hang up" block.
- Realize the hang up function.

3. Hardware List

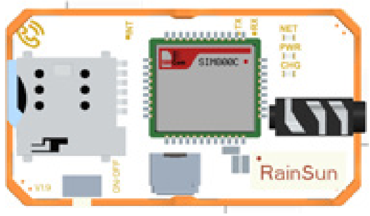
1. ESP32 Module



2. Keyboard Module



3. 2G Module

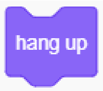
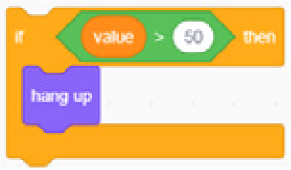


4. Knowledge Analysis

Let us learn the new code blocks that will be used in this lesson!

1. Blocks Explanation

Table 10-1 Blocks Explanation

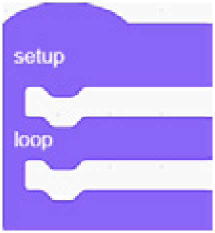
Block	Example
 hang up Explanation: Hang up the call	 if value > 50 then hang up If the value of the variable "value" is greater than 50, hang up the call

5. Project Realization

In this lesson, we will make a dialer that allows us to dial or hang up the call with only one key. Save the number of the person you often contact and match the number of each person with a key on the Keyboard module. When you press a key on the Keyboard module, you can dial the phone to the corresponding contact. Besides, you will use the hang up block to achieve the hang up function. When you need to end the call, just press the hang up key to end the current call!

1. Programming

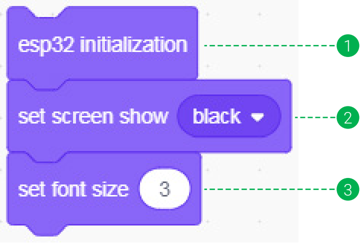
1. Add setup/loop block



```
setup
loop
```

2.In the setup area: it can be divided into 3 steps to build code blocks

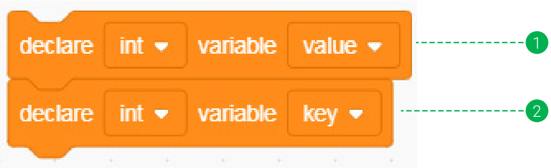
Step 1： Initialize ESP32 module and set the display effect of the screen:



```
esp32 initialization
set screen show black
set font size 3
```

① Initialize the ESP32 module to display the text on the screen later.
② Set the screen background to black.
③ Set the font size displayed on the screen to 3.

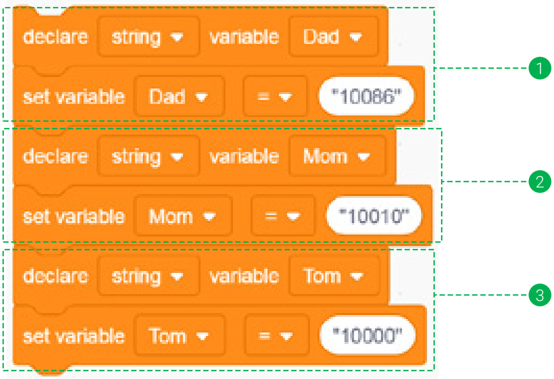
Step 2: Declare variables to store key-related variables:



```
declare int variable value
declare int variable key
```

① Declare the integer variable “value” to store the analog value of the pressed button in the self-made building block “button”.
② Declare the integer variable “key” to store the pressed key.

Step 3: Declare 3 string variables to store the contact phone number: (note that the number needs to be modified by yourself)

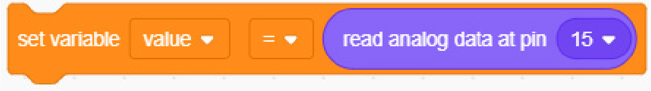


```
declare string variable Dad
set variable Dad = "10086"
declare string variable Mom
set variable Mom = "10010"
declare string variable Tom
set variable Tom = "10000"
```

1. Declare the string variable "Dad" to store Dad's phone number, and set the number to "10086".
2. Declare the string variable "Mom" to store the mother's phone number, and set the number to "10010".
3. Declare the string variable "Tom" to store the phone number of your friend Tom, and set the number to "10000".

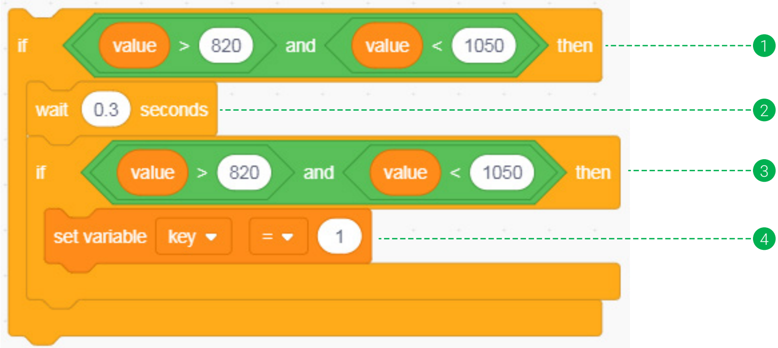
3. Self-made building block “button", mainly used to determine which button is pressed. It can be divided into 3 steps:

Step 1: Read the analog value of Keyboard at pin 15 and assign it to the variable “value”.



```
set variable value = read analog data at pin 15
```

Step 2: Determine which button is pressed according to the value of variable “value”



```
if value > 820 and value < 1050 then
    wait 0.3 seconds
    if value > 820 and value < 1050 then
        set variable key = 1
```

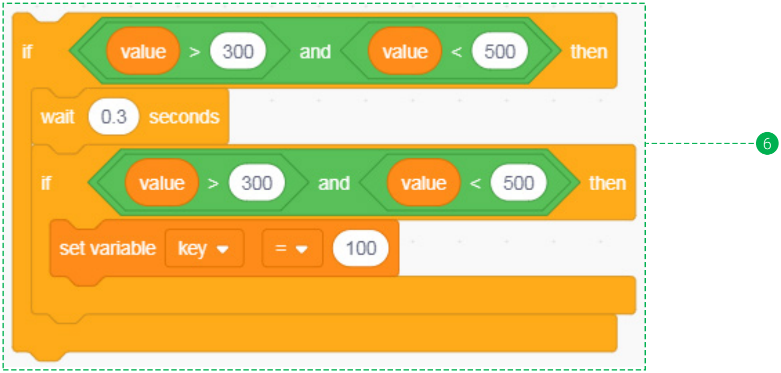
① Using the "if/then" code block to judge whether the variable “value” is greater than 820 and less than 1050 to determine whether button 1 is pressed.

② Add a delay of 0.3 seconds to prevent reading until the button is pressed.

③ Using the "if/then" code block again to judge whether the variable “value” is greater than 820 and less than 1050 to determine whether button 1 is pressed.

④ After button 1 is confirmed to be pressed, 1 is assigned to the integer variable “key” that stores the key pressed each time.

⑤ According to the previous ① - ④ sequence above, judge whether button 2 and button 3 are pressed.

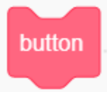


```
if value > 300 and value < 500 then
    wait 0.3 seconds
    if value > 300 and value < 500 then
        set variable key = 100
```

⑥ According to the previous ① - ④ sequence above, judge whether button ☎ is pressed.

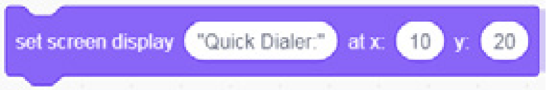
4.In the loop area, it is mainly divided into 4 steps to build code blocks:

Step 1: Call the self-made building block “button” to determine which button is pressed.



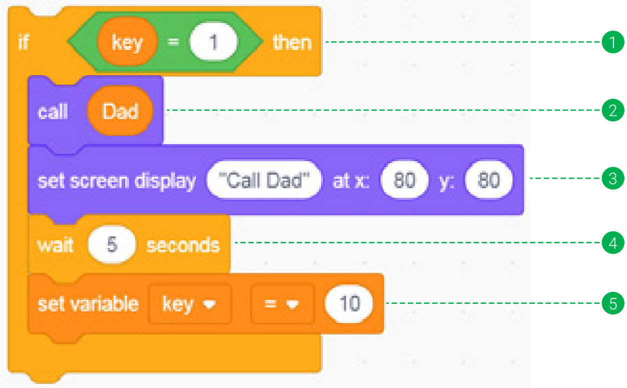
```
button
```

Step 2: Display the prompt "Quick Dialer:" at the screen coordinates X=10, Y=20.



```
set screen display "Quick Dialer:" at x: 10 y: 20
```

Step 3: Judged the value of variable “key” of the pressed button to dial the corresponding phone number.



```
if  key = 1  then
    call  Dad
    set screen display  "Call Dad"  at x: 80  y: 80
    wait  5  seconds
    set variable  key  =  10
```

1 Use the "if/then" code block to determine whether button 1 is pressed.

2 When button 1 is pressed, dial the phone number stored in variable “Dad”.

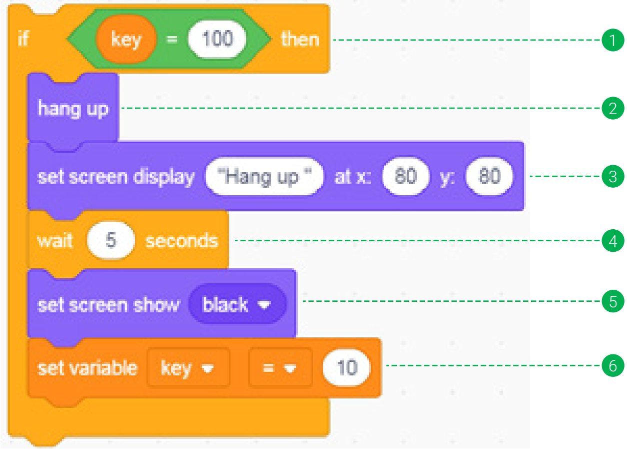
3 Displays the name of the contact to which the number belongs.

4 Wait five seconds to show whose number you dialing.

5 In order to prevent repeated calls, set the variable "key" to a number that is not equal to the number we use to dial or hang up.

6 According to the sequence of 1 - 5 above, respectively determine the number to be dialed when button 2 and button 3 are pressed.

Step 4: Judge whether the value of the variable “key” is 100. If it is, the hang up key ☎ is pressed, and hand up the call:



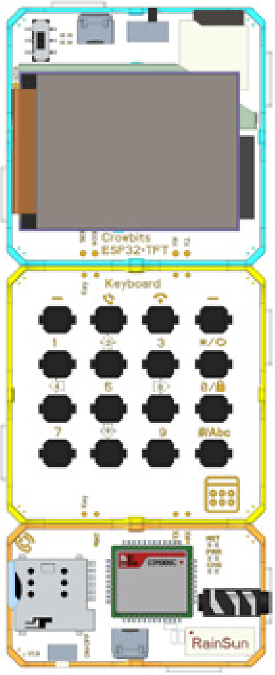
```
if  key = 100  then
    hang up
    set screen display  "Hang up "  at x: 80  y: 80
    wait  5  seconds
    set screen show  black
    set variable  key  =  10
```

1. Use the "if/then" code block to judge whether the variable “key” is equal to 100, that is, whether the hang up key ☎ is pressed.
2. When the hang up key is pressed, hang up the call.
3. Display a hang up prompt on the screen.
4. Wait for five seconds to show the hang up prompt.
5. Set the screen background to black to clear the screen display.
6. In order to prevent repeated send hang up command, set the variable "key" to a number that is not equal to the number we use to dial or hang up.

You can find the complete program of this case in the downloaded Master kit folder: Speed dial.sb3, and open it with letscode.

2. Download

Follow the steps below to upload the code to the ESP32 module, and start using this one-key dialer to talk to family and friends!



Module Connection Diagram

1. Build the program according to the case and change the phone number, and then upload the program to the ESP32 module.
2. Connect the Keyboard module to pin 15 of the ESP32 module.
3. Insert the phone card into the 2G module card slot, press and hold the power switch button of the 2G module to turn on the 2G module.
4. Connect the 2G module to the Keyboard module.
5. Toggle the DIP switch on the upper left corner of the ESP32 to the 2G mode, restart the ESP32 module, let’s dial to your family and friends with this speed dialer!

6. Brainstorming

In this lesson, we achieved how to dial numbers with one key and hang up the call. When there are too many contacts, we may forget the contacts represented by each key! To get around this, we can add a contact viewer function to display the contact's photo and name before dialing. When a key is pressed, the contact's photo and name are displayed on the ESP32 screen, and the number will be dialed only when the button is pressed again.

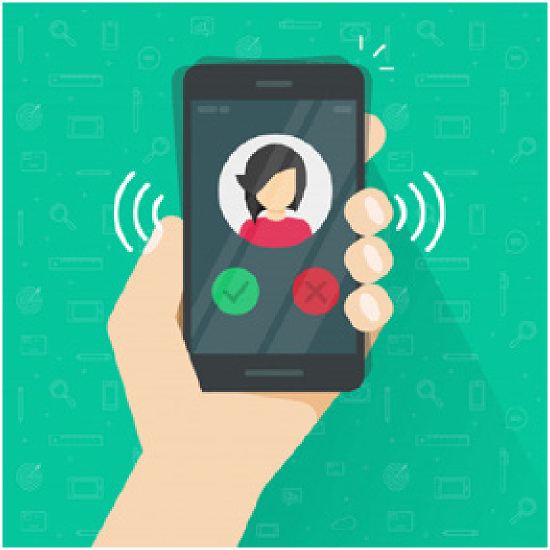
Lesson 11 Phone

1. Introduction

Mobile phone is one of the indispensable communication tools in our daily life. It brings great convenience to our lives and changes the way people communicate. But in most cases, We are only mobile phone users rather than mobile phone manufacturers. Through the previous study, we have a preliminary understanding of the structure and working principle of mobile phones.

To get a better understanding of how phones are made and how they work, you need to switch roles from user to maker, making a phone with basic functions.

Next, let's use the Crowbits module to make a mobile phone!

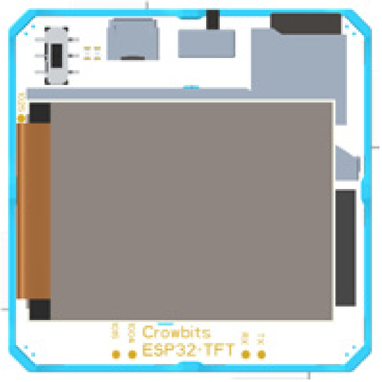


2. Learning Goals

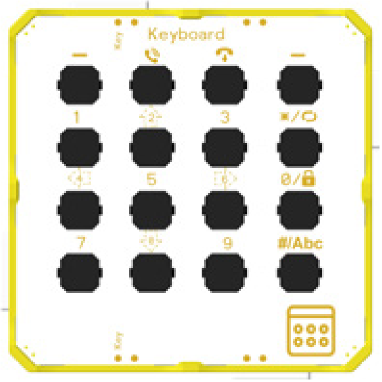
- Use the modules and the phone case to make a complete mobile phone.
- Master the procedure of uploading phone programs.
- Master the use of Crowbits mobile phone.

3. Hardware List

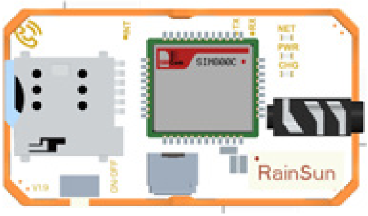
1. ESP32 Module



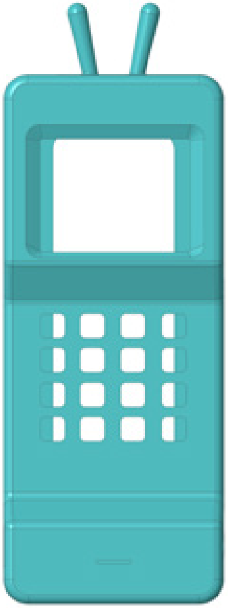
2. Keyboard Module



3. 2G Module



4. Phone Case



4. Build Steps

1. Install the ESP32 module to the phone case. Note that the power switch of the module should be installed on the upper part of the phone case, that is, the side close to the antenna of the phone case. (Pay attention to protect the screen of ESP32 to prevent crushing)

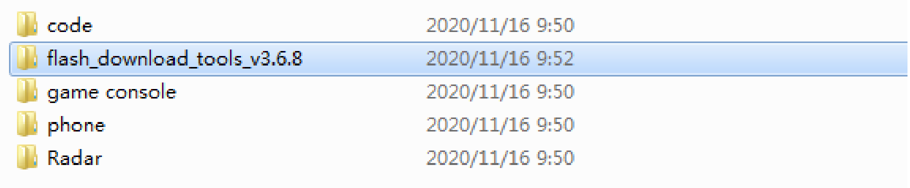
2. Install the 2G module to the phone case, and note that the power switch of the module should be installed in the lower part of the phone case, with the module headphone jack facing right.

3. Install the Keyboard module to the phone case. The module name "Keyboard" should be connected to the ESP32 at the top; the bottom of the module should be connected to the 2G module.

5. Program and Firmware Download

You need to follow the steps below to upload the complete phone program to the ESP32 module.

1. Open and unzip the downloaded compressed package, which contains the folders as shown in picture 11-1 below:

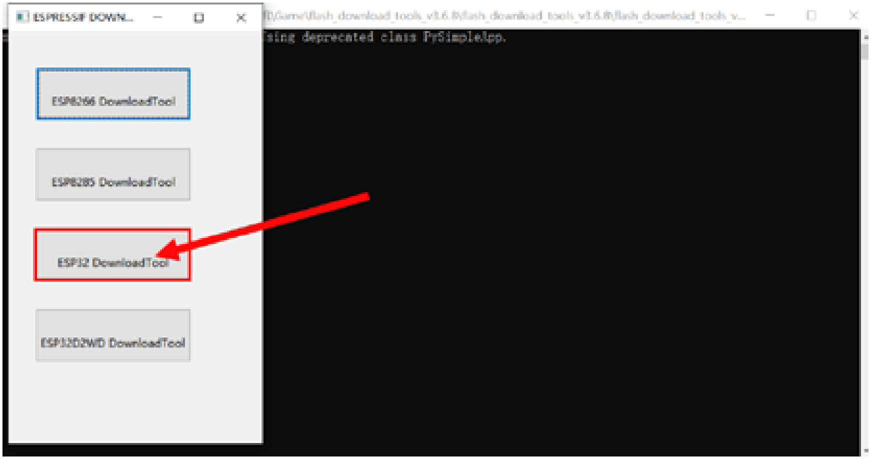


code	2020/11/16 9:50
flash_download_tools_v3.6.8	2020/11/16 9:52
game console	2020/11/16 9:50
phone	2020/11/16 9:50
Radar	2020/11/16 9:50

Picture 11-1

2. Enter the flash_download_tools_v3.6.8 folder and find the software named flash_download_tools_v3.6.8 and double-click to open it, as shown below:

3. After opening the software, click the ESP32 DownloadTool as shown in picture 11-2 below.



```
sing deprecated class PySimpleApp.
```

Picture 11-2

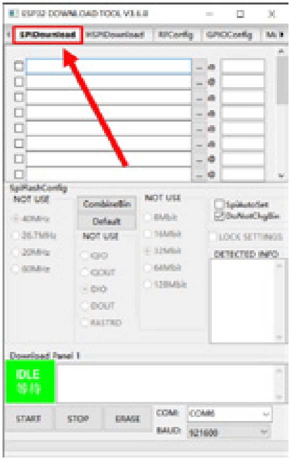
4. Choose to download in SPIDownload mode, as shown in picture 11-3 below.

5. Click the icon ... to select the file to upload, and write the corresponding address, as shown in picture 11-4 below.

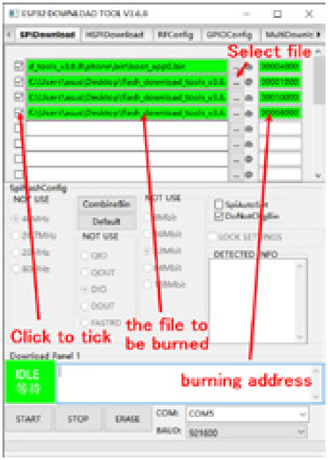
Burn the files and corresponding addresses from top to bottom:

Order	File	Address
①	boot_app0.bin	0x0000e000
②	bootloader_dio_40m.bin	0x00001000
③	firmware.bin	0x00010000
④	partitions.bin	0x00008000

Note: Click the box in front of the file name to tick so that the burned file can be downloaded effectively.



Picture 11-3



Select file

Click to tick

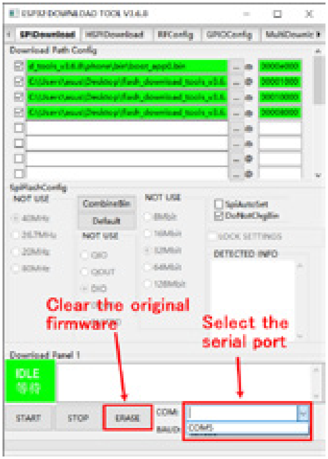
the file to be burned

burning address

Picture 11-4

6. Select the correct serial port and click the ERASE button to clear the original firmware, as shown in picture 11-5 below.

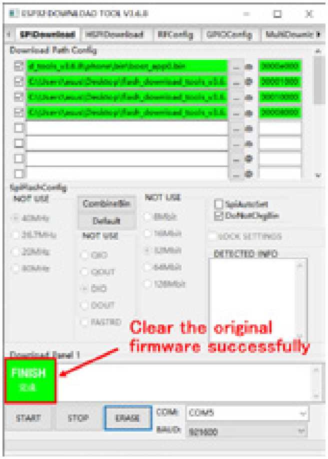
7. The original firmware is successfully cleared, as shown in picture 11-6 below.



Clear the original firmware

Select the serial port

Picture 11-5



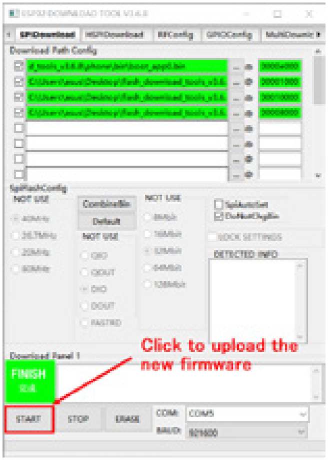
Clear the original firmware successfully

Picture 11-6

8. After clearing the original firmware, click the START button to upload the new firmware, as shown in picture 11-7 below.

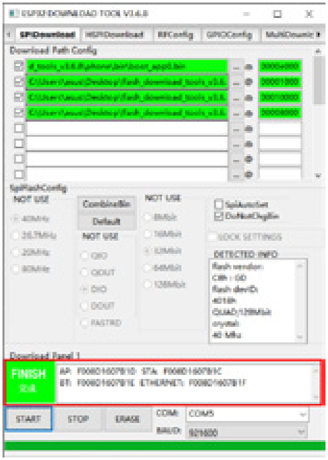
9. Uploading the new firmware successfully, as shown in picture 11-8 below.

10. Restart the ESP32 module to make the firmware work.



Click to upload the new firmware

Picture 11-7



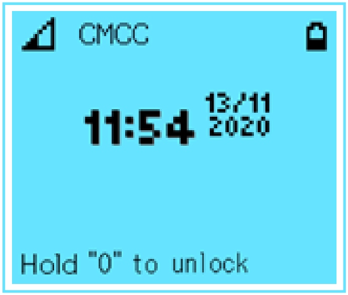
Picture 11-8

6. How to use

Insert the phone card into the 2G module, press and hold the 2G module power button to turn on the 2G module; insert the memory card into the ESP32 module and restart the ESP32 module. Then, start using your phone!

The following describes the functions of Crowbits mobile phone:

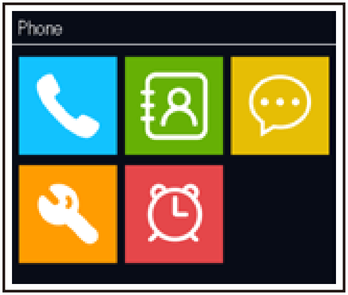
1. Lock screen interface: the phone will be in the lock screen interface at the beginning(as shown below). The current phone card signal, date and time, and ESP32 battery power will be displayed on this interface. You only need to press and hold button 0 to unlock and enter the main interface of the phone. When the phone is in the off-screen state, press any button on the top row of the Keyboard module to wake up the phone to the lock screen interface.


CMCC

11:54 13/11 2020

Hold "0" to unlock

2. Main interface: The main interface includes 5 functional interfaces, including Phone interface, Contacts interface, Messages interface, Settings interface, and Clock interface.



3. Use of the Keyboard: The Keyboard has multiplexing functions. Press the ✳ button to switch the mode to direction key mode or key input mode. In the direction key mode, the keys can be used to control direction, such as left, right, up, down; in the key input mode, the keys can be used to input characters, such as numbers, letters and etc. The functions of the buttons in the direction key mode and key input mode are as follows:

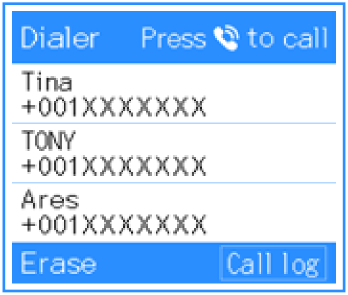
— LFN	☎ Select	☎ Back	— RFN
1	⇧2	3	✳ ✥ ⟳ A
⇦4	5	6⇨	0
7	⇩8	9	#

Direction key mode

— LFN	☎ Select	☎ Back	— RFN
1 +-*/':	2 ABC!;<	3 DEF%[=	✳ ✥ ⟳ A
4 GHI(\>	5 JKL)]^	6 MNO@_`	0 #$.?"&
7 PQRS{|	8 TUV,}~	9 WXYZ␣	# CAPS

Key input mode

4. Phone interface: You can dial numbers here. Enter a phone number using the Keyboard. Press the call button(Select button) to make a call, and press the left "-" button(LFN button) to erase the entered number. Press the right "-" button(RFN button)to view the call log, and you can dial the phone number in the call log or delete the call log. When making a call, you need to plug the headphone with a microphone into the headphone jack of the 2G module to make a call. Press the ✳ button to switch mode to the key input mode to enter the number (in the Phone interface, key input mode only can enter numbers); press the ✳ button again to switch the mode back to the direction key mode to control up and down.



Dialer Press ☎ to call

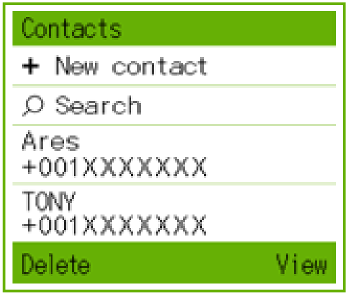
Tina
+001XXXXXXX

TONY
+001XXXXXXX

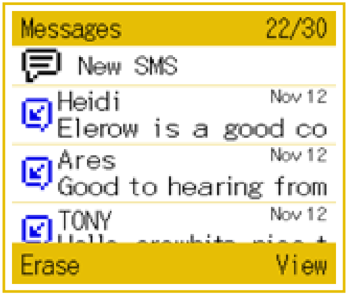
Ares
+001XXXXXXX

Erase Call log

5. Contacts interface: This interface saves contact information, including phone numbers and names. You can add new contacts, search for contacts, view contact information, modify contact information, delete contacts, and dial contact numbers on this interface. When searching for contacts, press the ✳ key to switch mode to key input mode to enter characters; press the ✳ key again to switch the mode back to the direction key mode to control up or down.



6. Messages interface: You can edit a new message, and view the sent and received text messages. Besides, you can delete the received or sent SMS. Press the ✳ key to switch mode to key input mode to enter the text; press the ✳ key again to switch the mode back to the direction key mode to control up or down.

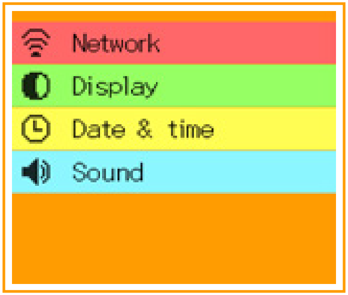


Heidi Nov 12
Elerow is a good co

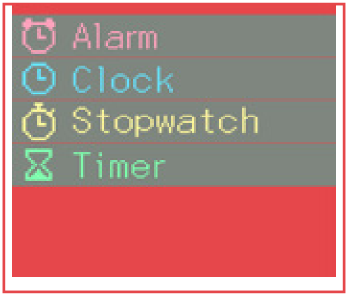
Ares Nov 12
Good to hearing from

TONY Nov 12

7. Settings interface: You can set the phone's mode, display, date and time, and sound volume on this interface.



8. Clock interface: You can set the alarm clock, check the time, use the stopwatch and timer functions on this interface.



7. Brainstorming

In this lesson, we learned how to burn the mobile phone program into the Crowbits module to make it a full-featured phone.

There are many classifications of mobile phones according to different age, including children's mobile phones, elderly mobile phones, etc. The corresponding functions of different types of mobile phones will be slightly different. Based on the Crowbits mobile phone, you can transform the Crowbits mobile phone into an elderly phone suitable for the elderly, adding the following functions:

1. Add a button tone; when the button is pressed, the speaker can emit a button tone to prompt the key is being pressed;
2. Add a one-key emergency call; in an emergency, the elderly can press the emergency button to ask for help immediately.

Lesson 12 Joystick Servo

1. Introduction

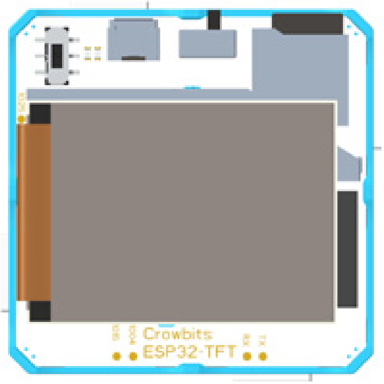
In childhood arcade games, most of the joysticks were designed using airplane flight control sticks as prototypes. Buttons were also designed near the joystick, which can be used for bullet launching and missile launching operations in the game. On the Crowbits thumb joystick sensor, you can see a joystick and several buttons on it. In this lesson, we will learn how to use the joystick to control the hardware module, such as controlling the rotation of the servo.

2. Learning Goals

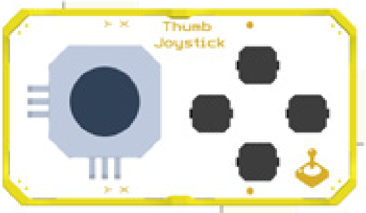
- Know about the working principle of thumb joystick sensor.
- Use the thumb joystick sensor to control the servo.

3. Hardware List

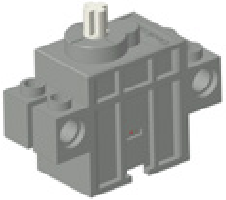
1. ESP32 Module



2. Thumb Joystick Module



3. Servo Module



4. Knowledge Analysis

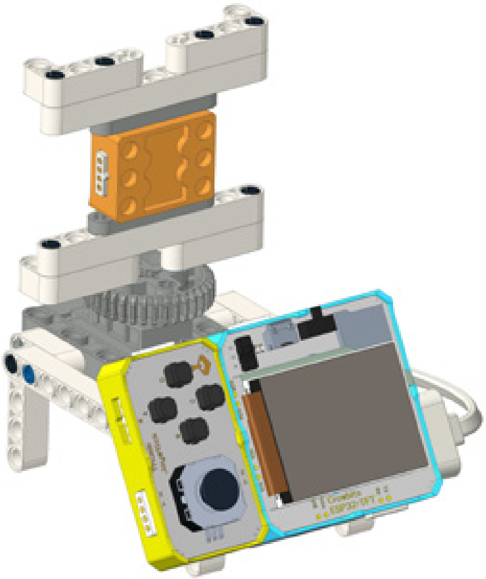
1. Hardware Introduction

There are a joystick and four buttons on the thumb joystick sensor. The joystick is controlled by two pins, and the four buttons are controlled by one pin. After connecting it to the ESP32 module, by reading the analog values of the pins, you can determine the range of analog values and program to use the thumb joystick.

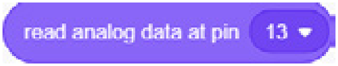
Table 12-1 Analog value table

Symbol	Range	Symbol	Range
A	2000-2200	Up	1600-1800
B	3400-3600	Down	3300-3500
C	1300-1500	Left	3200-3400
D	0-200	Right	1600-1800
No button press	4000-4100	No joystick move	2600-2800

In this case, we need to connect the thumb joystick sensor to pin 13 (shaking left and right) and the servo to pin 0 of the ESP32 module. The connection is shown in the picture below: (please refer to lesson 6 radar for structure)



2. Blocks Explanation

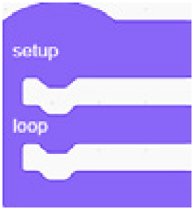
Block	Explanation
 read analog data at pin 13 ▾	Read the analog value of the pin, you can choose to read different pins

5. Project Realization

The joystick servo is composed of a thumb joystick sensor and a servo. The analog value of the pin connected to the thumb joystick sensor can be read by the program to control the left and right rotation of the servo.

1. Programming

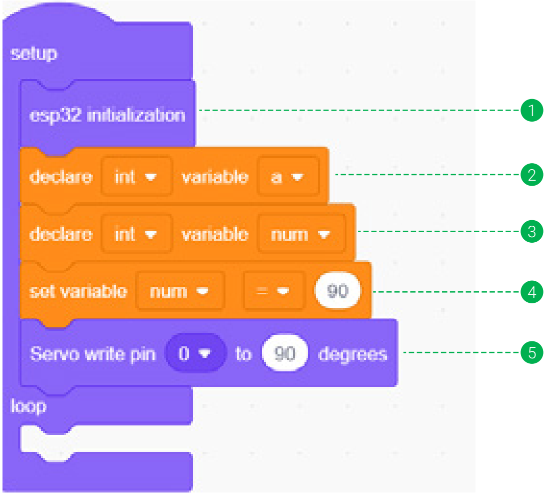
1. Add setup/loop block



setup

loop

2. In the setup area:



setup

esp32 initialization

declare int ▾ variable a ▾

declare int ▾ variable num ▾

set variable num ▾ = ▾ 90

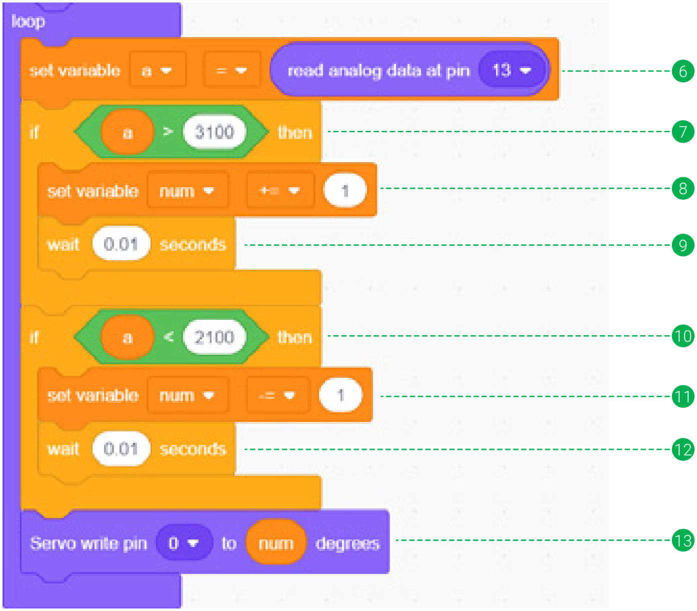
Servo write pin 0 ▾ to 90 degrees

loop

1. Before using the ESP32 module, we first initialize the ESP32 module.
2. Declare an integer variable “a” to store the analog value of the joystick.
3. Declare an integer variable “num” to control the rotation of the servo.
4. Initialize the integer variable “num” to 90, and let the servo turn left and right from 90 degrees.
5. Set the initial angle of the servo to 90 degrees.

3. In the loop area:

In the loop area, repeatedly read the analog value of the joystick. When the analog value is greater than 3100, it means that the joystick is moved to the left, and increase the value of the variable “num” by 1, and the delay time is set so that the value does not increase too fast. When the analog value is less than 2100, it means that the joystick is moved to the right, and decrease the value of the variable “num” by 1, and the delay time is set so that the value does not increase too fast. Finally, make the servo rotate to the angle of the variable “num” value to achieve this case.



```
loop
  set variable a = read analog data at pin 13
  if a > 3100 then
    set variable num += 1
    wait 0.01 seconds
  if a < 2100 then
    set variable num -= 1
    wait 0.01 seconds
  Servo write pin 0 to num degrees
```

6 Set the value of variable “a” to the analog value read from pin 13. We connect the thumb joystick sensor to pin 13 of the ESP32 module, and then repeatedly read the analog value from this pin to determine which direction the joystick is heading.

7 After assigning the analog value returned by the current joystick sensor to variable “a”, the next step is to judge the range of the value of variable “a” and determine the joystick heading side. If the value of variable “a” is greater than 3100, it means that the joystick is heading to the left.

8 When the joystick sensor is moved to the left, the value of the variable “num” is increased by 1. Then the servo can turn 1 degree to the left.

9 Wait for 0.01 second is to make the servo rotate slowly. If the 0.01 second delay is not added and the joystick is moved to the left for 1 second, the num value may have increased to several hundred. Therefore, adding the delay time can avoid the problem that the value increases too fast, and can also make the servo rotate slowly.

10 If the value of variable “a” is less than 2100, it means that the joystick is moved to the right.

⑪ When the joystick is moved to the right, the value of the variable "num" is reduced by 1. Then the servo can turn 1 degree to the right.

⑫ Wait for 0.01 seconds.

⑬ Finally set the rotation angle of the servo at pin 0 to the value of the variable "num". Here we connect the servo to the pin 0 of the ESP32 module.

2. Download

After finishing the program, download it to the ESP32 module (if you forget the steps to download the program, please refer to it in Lesson 1). After downloading the program, remember to restart the ESP32 module, now we have finished the joystick servo.

6. Brainstorming

In this case, if you move the joystick to the right for 3 seconds after the servo has turned to the far right. Then you want to move the joystick to the left to make the servo turn left, but you will find that the servo can only turn left after 3 seconds. Do you know why? It turns out that when the servo rotates to the maximum angle on the right, the value of the servo rotation angle stored in the program continues to increase, that is, it increases to a value greater than 180. Is there any way we can prevent this problem? Fix the variable which stored the rotation angle of servo to 0-180 to see what happens!

Lesson 13 Control Cursor

1. Introduction

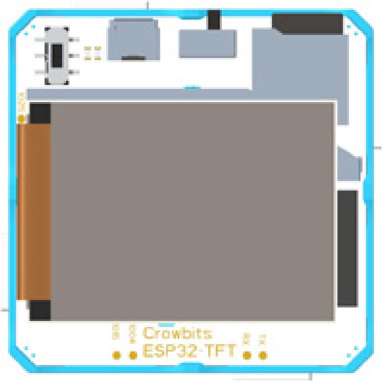
The drawing board is a tool that can draw the patterns you want. If you are not satisfied with your painting works, you can wipe it off and redraw it. In this lesson, we are going to make a "drawing board" that can create your own ideal works. Draw points or clear the screen by controlling the cursor movement on the "drawing board".

2. Learning Goals

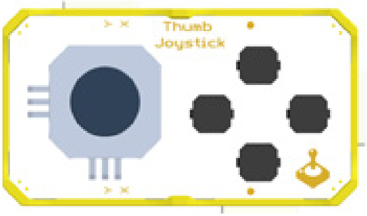
- Master how to read analog values on multiple pins.
- Master the method of obtaining the real-time position on the TFT screen.
- Make a TFT screen drawing board.

3. Hardware List

1. ESP32 Module



2. Thumb Joystick Module



In this case, we connect the joystick part to pin 13, 14 of the ESP32 module, and the button part to pin 25 of the ESP32 module. The connection diagram of this case is shown below:

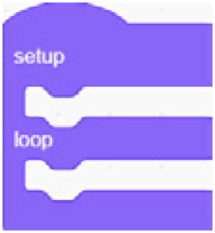
Joystick
Crowbits
ESP32-TFT

4. Project Realization

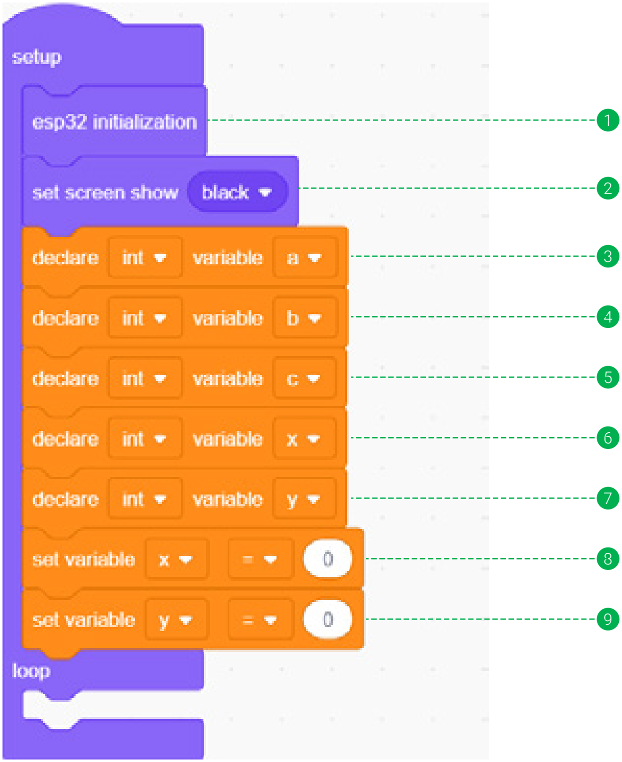
Record the real-time coordinates with variable "x" and variable "y". Read the analog value returned by the sensor. When there is a movement of the sensor, determine which side it moves to, thereby changing the value of variable "x" and variable "y" to change the real-time position of the cursor. When it is judged that the button is pressed, the cursor will leave a trace when it moving. If the button is not pressed, the cursor movement will not leave a trace.

1. Programming

1. Add setup/loop block



2. In the setup area:



```
setup
    esp32 initialization
    set screen show black
    declare int variable a
    declare int variable b
    declare int variable c
    declare int variable x
    declare int variable y
    set variable x = 0
    set variable y = 0
loop
```

Picture 13-1

① Before using the screen of the ESP32 module, we first initialize the ESP32 module.

② Set the screen background to black, which can make the screen have a better effect to display.

③ Declare an integer variable “a”, which is used to store the analog value of the buttons on the thumb joystick sensor.

④ Declare an integer variable “b”, which is used to store the analog value of the x-axis of the joystick that on the thumb joystick sensor.

⑤ Declare an integer variable “c”, which is used to store the analog value of the y-axis of the joystick that on the thumb joystick sensor.

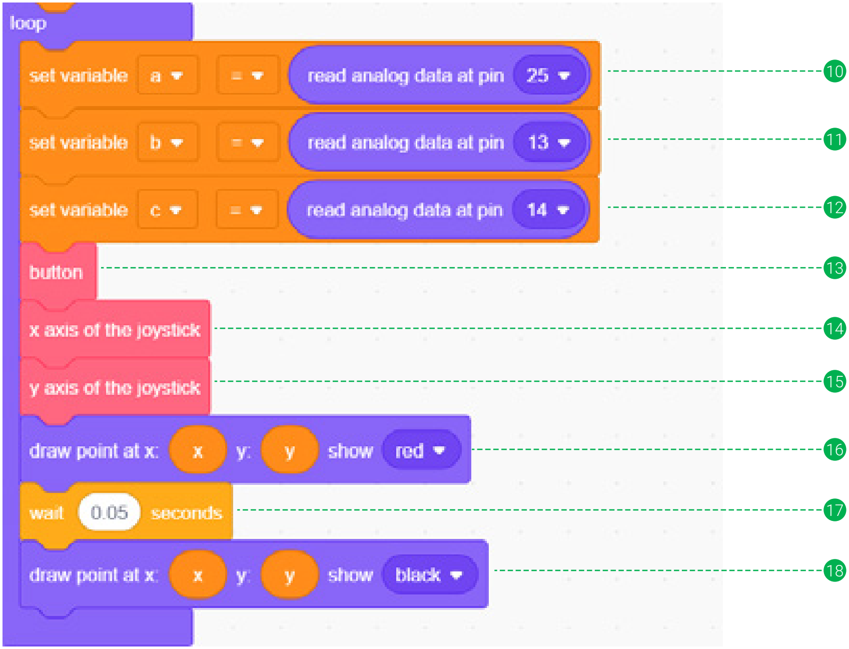
⑥ Declare an integer variable “x”, which is used to record the real-time x coordinate of the cursor.

⑦ Declare the integer variable “y”, which is used to record the real-time y coordinate of the cursor.

⑧ Set the initial value of the variable “x” to 0.

⑨ Set the initial value of the variable “y” to 0, and the coordinates of the starting cursor are (0, 0).

2. In the loop area:



```
loop
  set variable a ▾ = ▾ read analog data at pin 25 ▾
  set variable b ▾ = ▾ read analog data at pin 13 ▾
  set variable c ▾ = ▾ read analog data at pin 14 ▾
  button
  x axis of the joystick
  y axis of the joystick
  draw point at x: x y: y show red ▾
  wait 0.05 seconds
  draw point at x: x y: y show black ▾
```

Picture 13-2

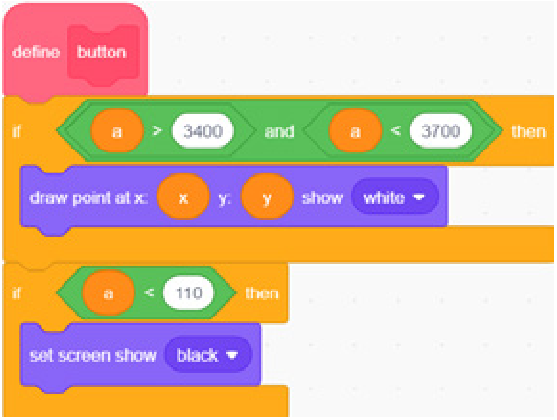
⑩ Read the analog value of pin 25 and store it in the variable “a”.

⑪ Read the analog value of pin 13 and store it in the variable “b”.

⑫ Read the analog value of pin 14 and and store it in the variable “c”.

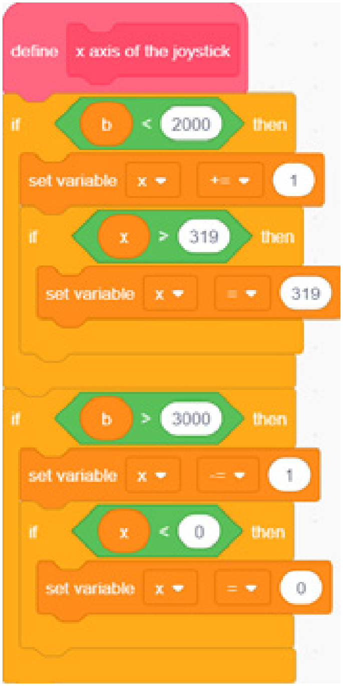
⑬ Define self-made building block named button to judge whether the button on the thumb joystick sensor is pressed, if the button B is pressed, a white line will be drawn on the TFT screen, and if the button D is pressed, the screen will be cleared.

Self-made building block “button”:



```
define button
if a > 3400 and a < 3700 then
    draw point at x: x y: y show white
if a < 110 then
    set screen show black
```

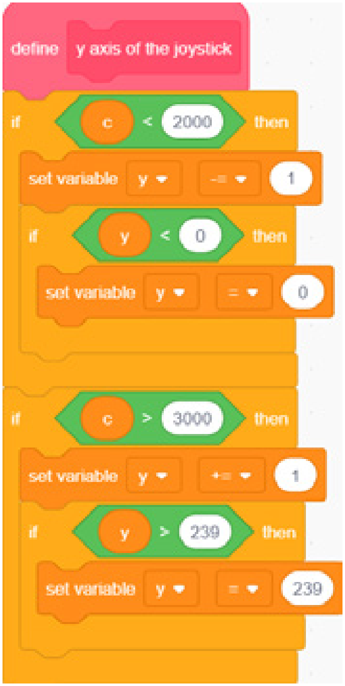
Self-made building block “x axis of the joystick”:



```
define x axis of the joystick
if b < 2000 then
    set variable x += 1
    if x > 319 then
        set variable x = 319
if b > 3000 then
    set variable x -= 1
    if x < 0 then
        set variable x = 0
```

14 Define a self-made building block “x axis of the joystick” to determine the direction of movement of the joystick x-axis. If you move the joystick left, the cursor will move left; if you move the joystick right, the cursor will move right.

Self-made building block “y axis of the joystick”



```
define y axis of the joystick
if c < 2000 then
    set variable y -= 1
    if y < 0 then
        set variable y = 0
if c > 3000 then
    set variable y += 1
    if y > 239 then
        set variable y = 239
```

15 Define a self-made building block “y axis of the joystick” to determine the direction of movement of the joystick y-axis. If you move the joystick up, the cursor will move up; if you move the joystick down, the cursor will move down.

16 Display the real-time coordinates of the cursor, in order to highlight the cursor, we set the cursor color to red.

17 Wait for 0.05 seconds, and place it here to control the speed of cursor blinking.

18 Hide the real-time coordinates of the cursor, we achieve it by setting the cursor color to black that same as the background color. Before the cursor is moved, clear the position of the previous cursor so that no trace will be displayed when the cursor moves. Only when the button is pressed, the trace is displayed during cursor movement.

2. Download

After finishing the program, download it to the ESP32 module (if you forget the steps to download the program, please refer to it in Lesson 1). After downloading the program, remember to restart the ESP32 module, now you have made a TFT screen drawing board.

5. Brainstorming

A perfect drawing board should be able to change the color of the font. Switch the background color, use your imagination, and add the ability to draw various colors to the drawing board!

Lesson 14 Aircraft war

1. Introduction

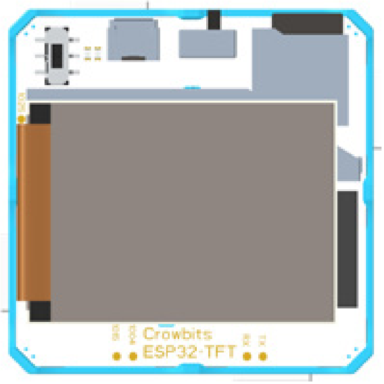
Aircraft War is a classic air combat game! The player can fly an aircraft to fight an enemy's entire air force. To shoot down enemy aircraft by dodging missile attacks. Collecting crystals can increase the firepower of the guns on the aircraft. In this lesson, we will use letscode software to make the scene of the aircraft war, use the thumb joystick sensor to control the flight, shoot bullets to shoot down the enemy! Let's review the classic aircraft war game!

2. Learning Goals

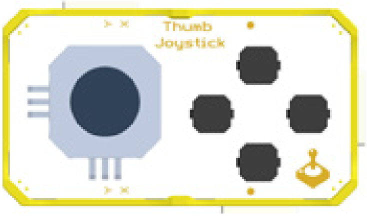
- Master the use of letscode software and hardware interaction.
- Use the thumb joystick sensor to control the aircraft to achieve the aircraft war game.

3. Hardware List

1. ESP32 Module



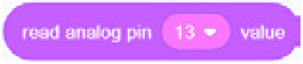
2. Thumb Joystick Module



In this case, we connect the joystick part to pin 13, 14 of the ESP32 module, and the buttons part to pin 25 of the ESP32 module. The connection diagram of the case is as shown:

4. Project Realization

1. Hardware Introduction

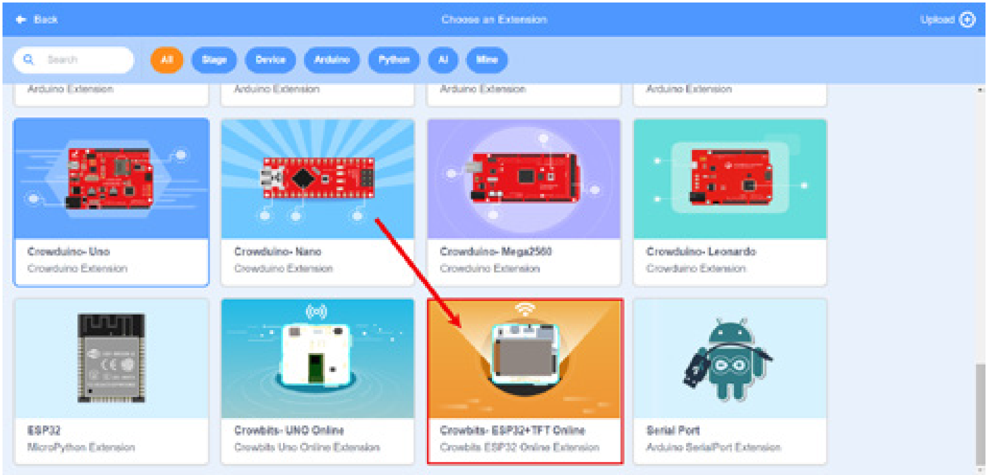
Block	Explanation
 read analog pin 13 value	The function of ESP32 module to interact with software, read the analog value of pin 13

5. Project Realization

Move the joystick up, down, left, and right to control the aircraft's flight and avoid the enemy. The button D of the thumb joystick controls the aircraft to fire bullets and destroy the enemy. When the bullet hits the enemy, the enemy disappears and one point is added. When the enemy hits your aircraft or your aircraft is shot down, lose the game.

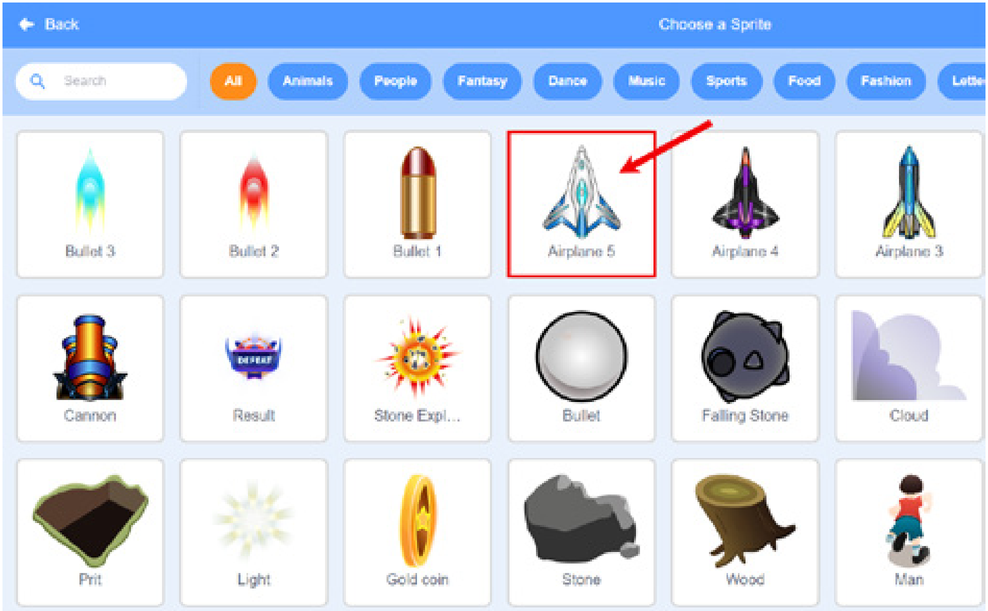
1. Programming

1. In order to use the software and hardware interaction function in the case, we first import the ESP32+TFT Online extension library, as shown in picture 14-1 below.



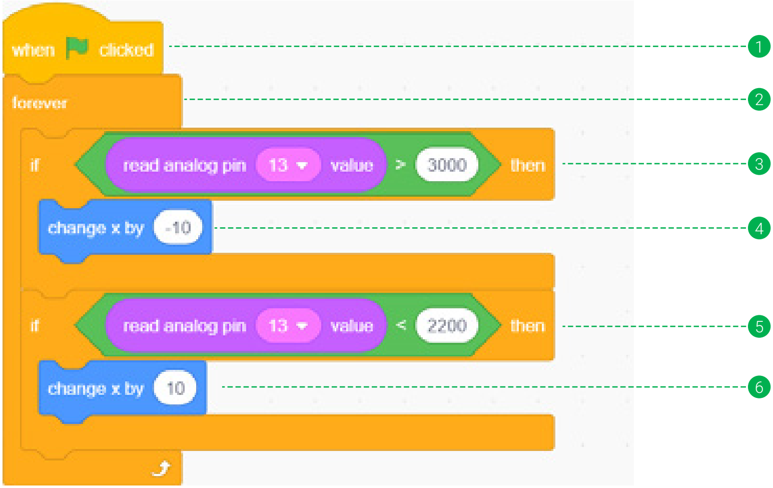
Picture 14-1

2. Add the aircraft sprite, here we add the sprite Airplane5.



Picture 14-2

Program of aircraft:

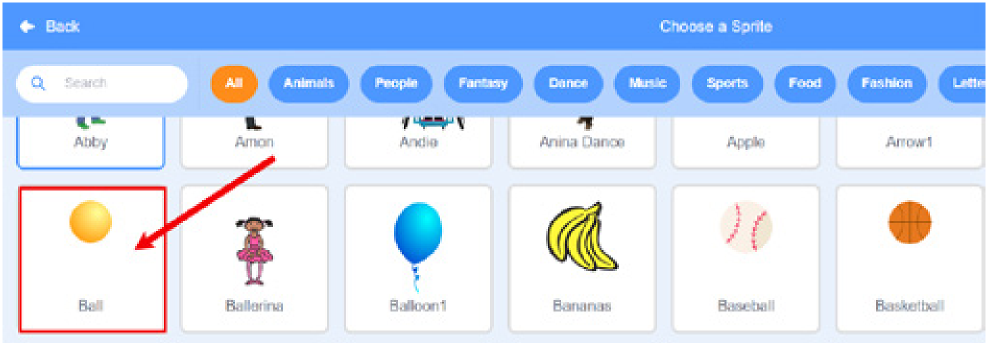


```
when ⚑ clicked
forever
    if <(read analog pin (13 ▾) value) > (3000)> then
        change x by (-10)
    if <(read analog pin (13 ▾) value) < (2200)> then
        change x by (10)
```

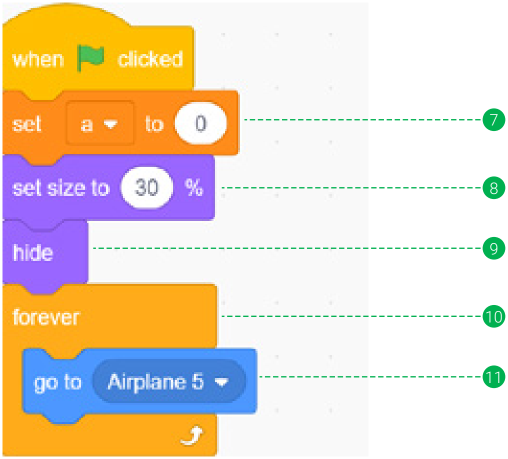
Picture 14-3

1. The software and hardware interactive program uses the small green flag as the start of the program.
2. Repeat to judge the analog value of pin 13. When the analog value of pin 13 has changed, the program can immediately determine it and run corresponding blocks.
3. Conditional judgment, if the analog value read from pin 13 is greater than 3000, it means that the joystick has been moved to the left.
4. If the joystick is moved to the left, aircraft move to the left. Here we can achieve it by reducing 10 to the x coordinate of the aircraft.
5. Conditional judgment, if the analog value read from pin 13 is less than 2200, it means that the joystick has been moved to the right.
6. If the joystick is moved to the right, aircraft move to the right, here we can achieve it by increasing 10 from the x coordinate of the aircraft.

3. Program of bullet



Picture 14-4



```
when [flag] clicked
set a to 0
set size to 30 %
hide
forever
    go to Airplane 5
```

Picture 14-5

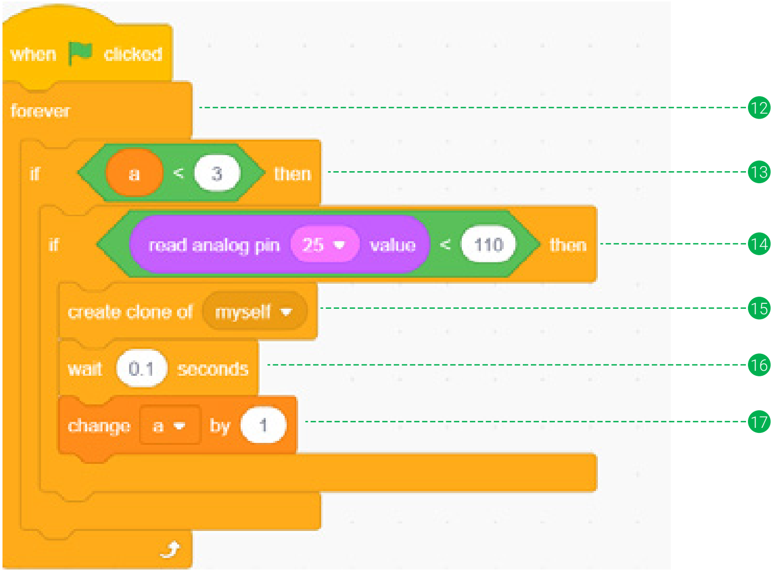
7 First, add the small green flag as the start of the program. In order to limit the number of bullets launched per round, we create a new variable “a” and initialize it to 0.

8 Set the size of the bullet to an appropriate size, here we set it to 30.

9 In order to be able to launch multiple bullets without adding multiple bullet sprites, we need to use the clone block. Hide the bullet first, and the subsequent bullets are cloned from the hidden bullet.

10 The bullet is hidden in the aircraft, in this way, the bullet that is launched from the aircraft, so here we need to repeat the execution to move the hidden bullet to the aircraft.

11 Move the hidden bullet to the aircraft so that the bullet can be launched from the aircraft.



```
when 🏴 clicked
forever
  if a < 3 then
    if read analog pin 25 ▾ value < 110 then
      create clone of myself ▾
      wait 0.1 seconds
      change a ▾ by 1
```

Picture 14-6

12 First, add a small green flag to start another task in a sprite at the same time. This task is to make the aircraft fire bullets, and it is necessary to repeatedly judge whether the button D is pressed.

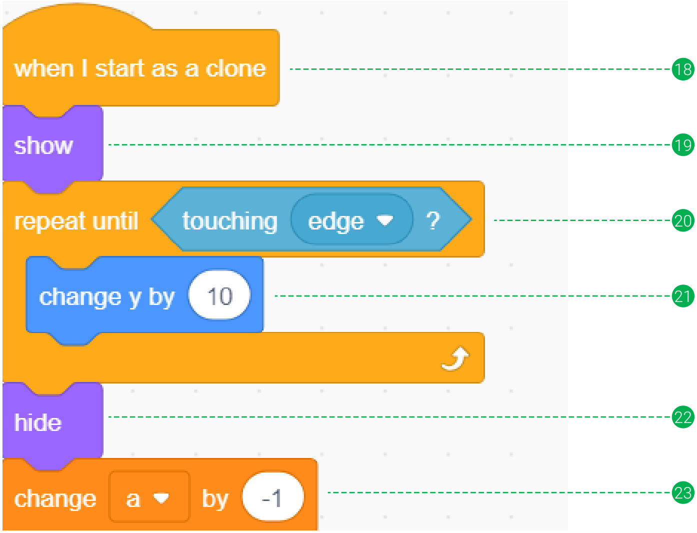
13 Limit only 3 bullets can be fired in each round, and add condition judgment. It can go into the next condition judgment only when the value of variable “a” is less than 3.

14 Judge whether the button D of the thumb joystick sensor is pressed, so here you only need to read the analog value of pin 25 to determine whether the button D is pressed. When the analog value is less than 110, it means the button D of the thumb joystick sensor is pressed.

15 When the button D is pressed, the cloned bullet is fired.

16 In order to prevent too many bullets from being fired at once, wait for 0.1 seconds. When the button D is kept pressed, the bullets will be fired every 0.1 seconds.

17 For each launch, the variable “a” is increased by 1, which means that the total number of bullets that have been launched is increased by 1.



```
when I start as a clone
show
repeat until <touching (edge ▾) ?>
    change y by (10)
end
hide
change (a ▾) by (-1)
```

Picture 14-7

18 You need to use the block “when I start as a clone” to start the program.

19 The bullet is hidden, and the cloned bullet is displayed first.

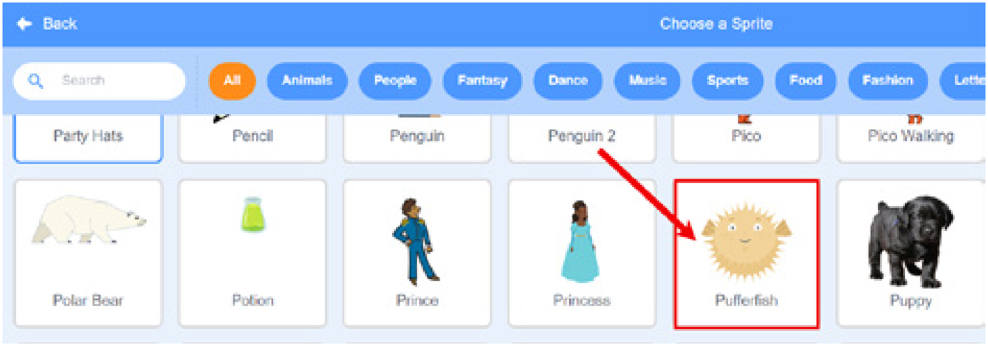
20 After the cloned bullet is displayed, it needs to move repeatedly until it touches the upper edge of the stage, and then to jump out of the loop.

21 Move up all the time, increase the y coordinate by 10, and change the number to change the speed of the bullet. The higher the number, the faster the bullet.

22 When it hit the edge of the stage, the bullet disappeared.

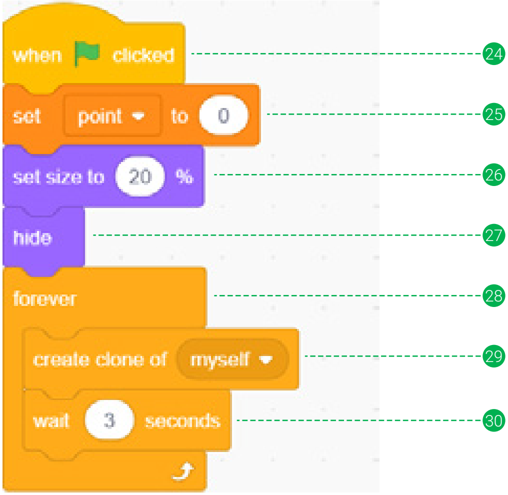
23 After the bullet disappears, the number of bullets on the stage needs to be reduced by 1, so that only 3 bullets can be fired per round. When the bullets on the stage are greater than or equal to 3, no more bullets can be fired. When the bullets on the stage are less than 3, you can continue to fire bullets.

4. Add the enemy sprite, here we add the sprite Pufferfish.



Picture 14-8

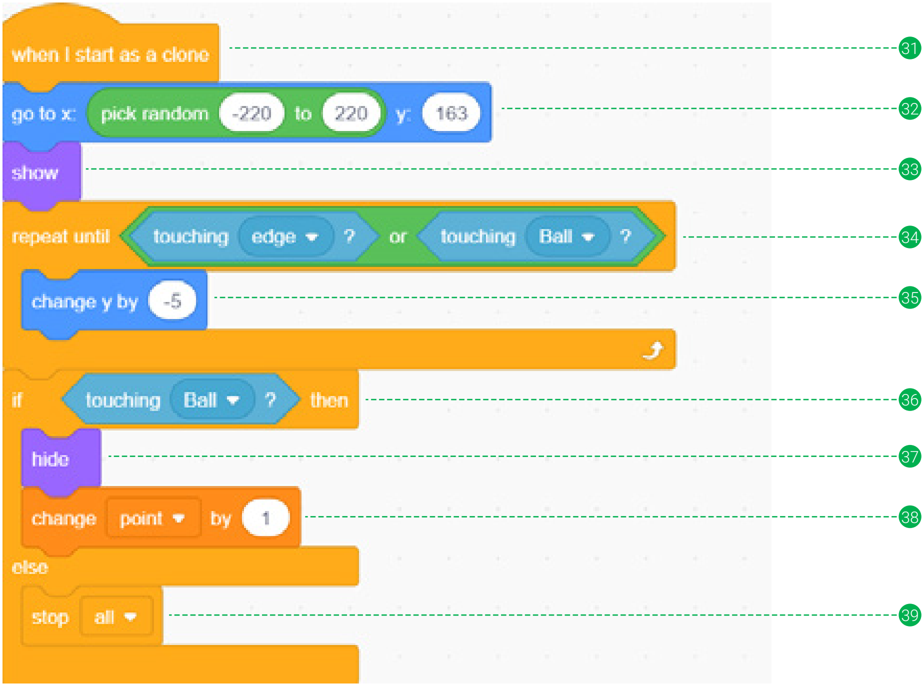
Program of enemy



```
when 🏴 clicked
set point ▾ to 0
set size to 20 %
hide
forever
    create clone of myself ▾
    wait 3 seconds
```

Picture 14-9

24 Similarly, we first use the small green flag to start the program.
25 Create a score variable “point” and initialize it to 0. Add one point when hitting an enemy.
26 Initialize the size of the enemy to 20.
27 Hide the enemy, what appears is a clone of the enemy.
28 The follow-up requires enemies to appear all the time, so we need to use repeated execution to achieve it.
29 Clone the enemy in repeated execution.
30 In order to prevent too many enemies from appearing at once, the enemy itself is cloned every 3 seconds.



```
when I start as a clone
go to x: (pick random (-220) to (220)) y: (163)
show
repeat until <<touching (edge ▾) ?> or <touching (Ball ▾) ?>>
    change y by (-5)
end
if <touching (Ball ▾) ?> then
    hide
    change [point ▾] by (1)
else
    stop [all ▾]
end
```

Picture 14-10

31 When the enemy is cloned, you need to use the block “when I start as a clone” to start the program.

32 When the enemy is cloned, move the clone to a random position on the edge of the stage, with a range of the stage is -240 to 240. Here we set the x coordinate to a random number(from -220 to 220). Let the clones of the enemy appears randomly on the edge of the stage without touching the edges on the left and right sides. The y coordinate is set to: 163.

33 Displays the enemy clone after setting the position of the enemy clone.

34 Set the conditions for the disappearance of the enemy clone, that is repeat until it hits the edge of the stage or hits a bullet.

35 When an enemy clone appears, repeat to reducing the y coordinate by 5 to make it drop.

36 There are two possibilities when the enemy clone jumps out of the loop. The first is that it hit a bullet, the second is that it touches the edge of the stage.

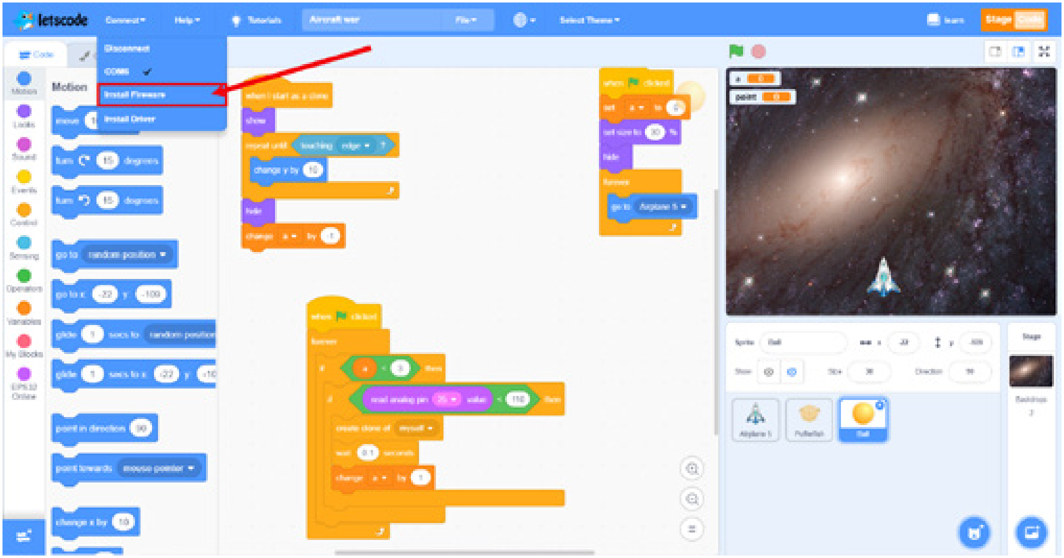
37 If the enemy clone hits a bullet, make the enemy clone disappears.

38 If the bullet hits the enemy clone, the score variable “point” increases by 1 point.

39 Otherwise, it will hit the edge of the stage, indicating that the bullet did not hit the enemy clone, and the game will fail and all scripts will stop.

2. Download

Connect the ESP32 module to a PC with a data cable, then select the connected port in the letscode software. After the port is successfully connected, click the “Install Firmware” button to download the program, as shown in picture 14-8 below. After successful download, click on the green flag on the stage to start playing your aircraft war!



```
when I start as a clone
show
repeat until <touching edge ?>
    change y by 10
hide
change a by -1
```

```
when 🏴 clicked
set a to 5
set size to 30 %
hide
forever
    go to Airplane 1
```

```
when 🏴 clicked
forever
    if <a < 3> then
        if <read analog pin 26 value < 150> then
            create clone of myself
            wait 0.1 seconds
            change a by 1
```

Picture 14-11

6. Brainstorming

There are many different intensifiers in aircraft war games, such as missiles, cannons, lasers, etc. You can launch it by pressing other buttons. Let's do this using the remaining buttons on the thumb joystick sensor.

Lesson 15 Two-Player Aircraft War

1. Introduction

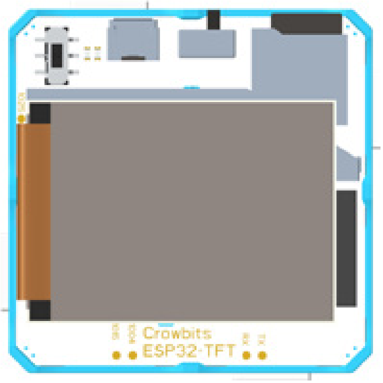
It would be more interesting if you could play aircraft war with friends! But how do we get our friends to join us? Don't worry, in this lesson, we will add another thumb joystick sensor to control another aircraft, and win the game with your friends!

2. Learning Goals

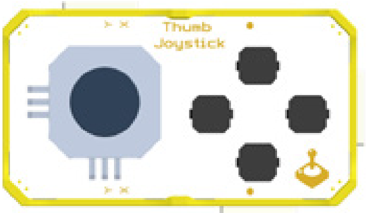
- Master the use of dual thumb joystick sensor.

3. Hardware List

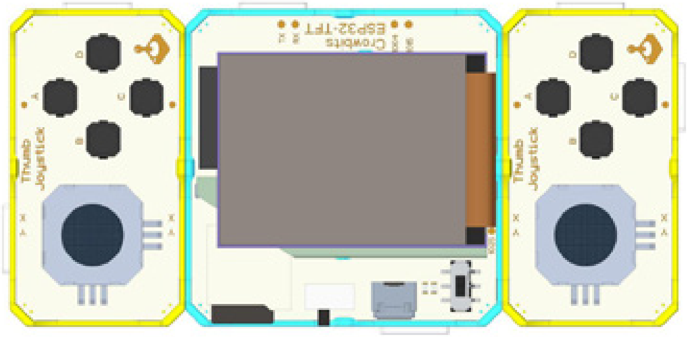
1. ESP32 Module



2. Thumb Joystick Module



In this case, we connect the left thumb joystick sensor to the IO39, IO36, and IO0 pins of the ESP32 module. Connect the right thumb joystick sensor to the IO25, IO34, IO13, and IO14 pins of the ESP32 module. As shown below:



4. Project Realization

Use the left joystick to control aircraft 1 to fly left and right, and its bullets will be launched automatically. Use the right joystick to control aircraft 2 to fly left and right, and its bullets will be launched automatically. If the bullet shoots the enemy aircraft and it disappears. Enemy aircraft also can fire bullets and you need to dodge bullets. When the enemy aircraft's bullets shoot your aircraft, your aircraft will blow up. During the game, you can upgrade the bullets of your aircraft through the "heart" icon that appears randomly in the game. if you collect a “heart” icon, your aircraft bullets will upgrade once. Ordinary bullets have a low speed and disappear after destroying an enemy aircraft. By getting the "heart" icon to upgrade the bullets, it will get a faster speed and the bullets can continue to attack other aircraft after shotting the enemy aircraft.

1. Download

As the code is too complex, you can use the program we edited. Open the downloaded file, find the two-player aircraft war.sb3, open it and upload it to the ESP32 module.

Connect the ESP32 module to a PC with a data cable, then select the connected port in the letscode software. After the port is successfully connected, click the upload button to download the program (refer to the download steps in Lesson 14). After successful download, start to play the aircraft war with your friend!

5. Brainstorming

In the game of two-player aircraft war, we can upgrade bullets by getting "heart". Then can we add weapons by getting some other icons, such as adding a missile to the aircraft if getting a yellow "heart" icon. Then, press the button B of the thumb joystick sensor to launch the missile.

Lesson 16 Game Console

1. Introduction

Playing games can bring people excellent leisure and entertainment experience, and game consoles have become one of the most popular electronic products. So how is a game console made? Game consoles need to go through cumbersome electronic design and production processes to produce, and it is too difficult to make a DIY game console of your own! But is there a simpler way to make a handheld game console?

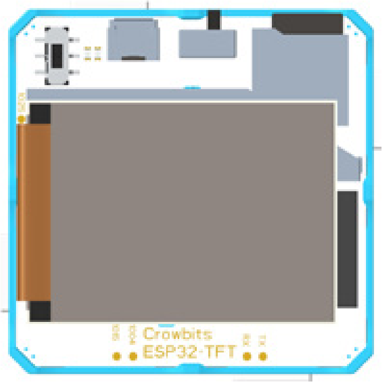
The answer is yes. Just magnet the modules together to make a complete game console! How easy it is! Let's start it.

2. Learning Goals

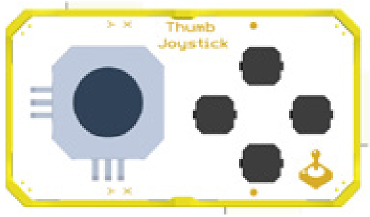
- Master the steps of building a game console.
- Master the burning steps of game console code.
- Master the operation of the game consoles.
- Master the steps of finding and downloading games.

3. Hardware List

1. ESP32 Module

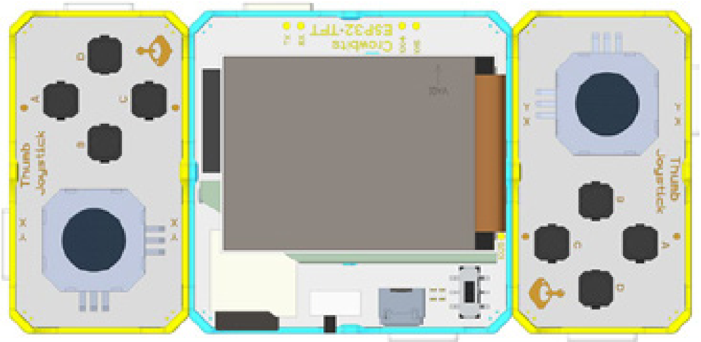


2. Thumb Joystick Module



4. Connection of Game console

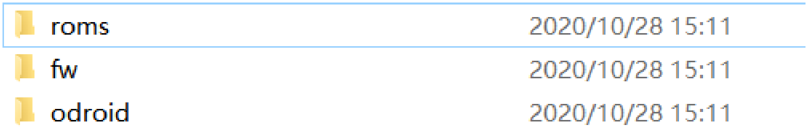
Connect the two thumb joystick sensors with the ESP32 module, as shown in the picture below.



5. SD card preparation

1. Before downloading the game console program and firmware, we can prepare the files for the SD card. First, download the required files, plug the SD card into a card reader and connect it to a laptop to format your SD card.

2. Open the game console folder and copy all the folder inside to the SD card, as shown in Picture 16-1 below.



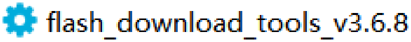
roms	2020/10/28 15:11
fw	2020/10/28 15:11
odroid	2020/10/28 15:11

Picture 16-1

3. The roms folder is used to store the downloaded simulator and games.

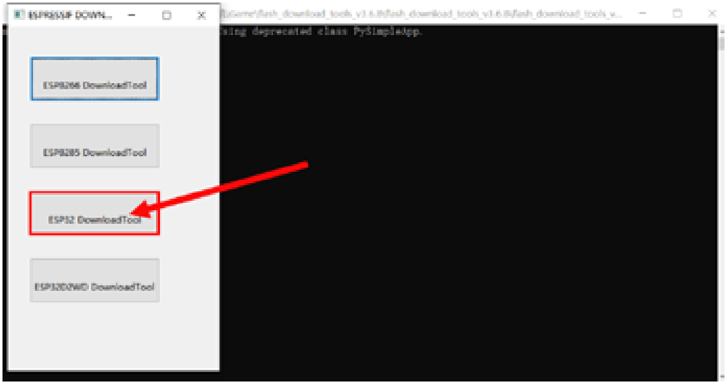
6. Program and firmware download

1. Find the software as shown picture below in the flash_download_tools_v3.6.8 folder and double-click to open it.



flash_download_tools_v3.6.8

2. After opening the software, click the ESP32 DownloadTool as shown in picture 16-2 below.



```
xing deprecated class PySimpleApp.
```

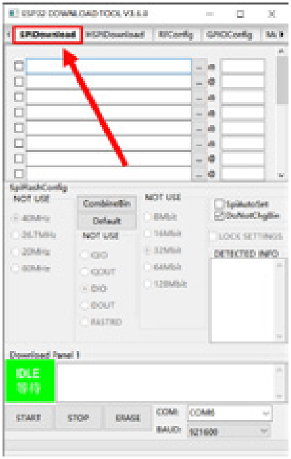
Picture 16-2

3. Choose to download in SPIDownload mode, as shown in picture 16-3 below.

4. Click the icon **...** to select the file to upload, and write the corresponding address, as shown in picture 16-4 below.

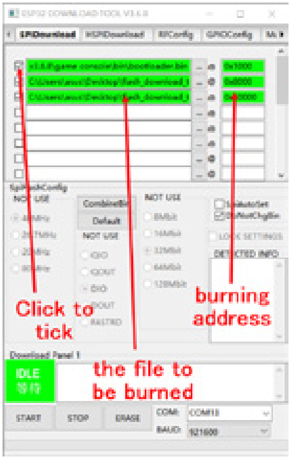
Burn the files(in /game_console/bin folder) and corresponding addresses from top to bottom:

1: File name: bootloader.bin	address: 0x1000
2: File name: partitions.bin	address: 0x8000
3: File name: odroid-go-firmware.bin	address: 0x10000



COM: COM6
BAUD: 921600

Picture 16-3



☑ ...\1.0.f\game console\bin\bootloader.bin @ 0x1000
☑ C:\Users\asus\Desktop\flash_download_t @ 0x8000
☑ C:\Users\asus\Desktop\flash_download_t @ 0x10000

COM: COM13
BAUD: 921600

Click to tick

the file to be burned

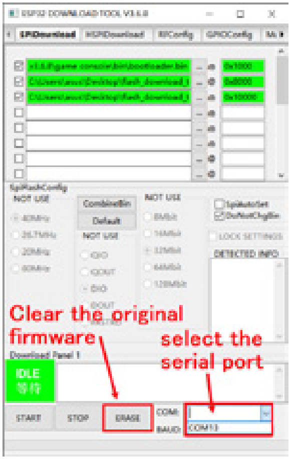
burning address

Picture 16-4

Note: Click the box in front of the file name to tick so that the burned file can be downloaded effectively.

5. Select the correct serial port and click the ERASE button to clear the original firmware, as shown in picture 16-5.

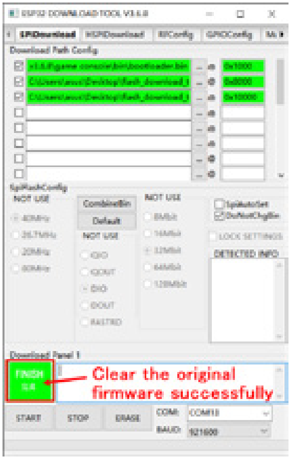
6. The original firmware is successfully cleared, as shown in picture 16-6 below.



Clear the original firmware

select the serial port

Picture 16-5

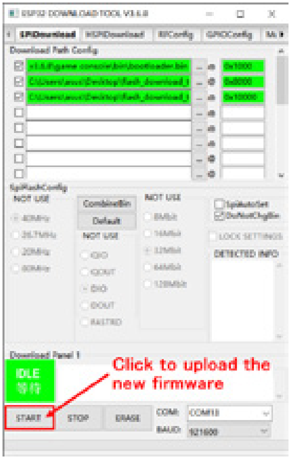


Clear the original firmware successfully

Picture 16-6

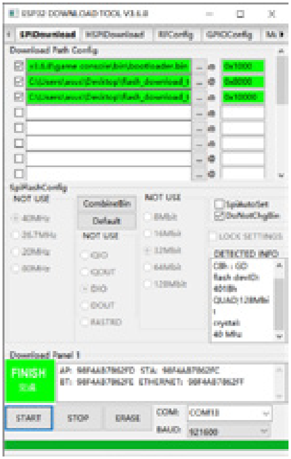
7. After clearing the original firmware, click the START button to upload the new firmware, as shown in picture 16-7 below.

8. Uploading the new firmware successfully, as shown in picture 16-8 below.



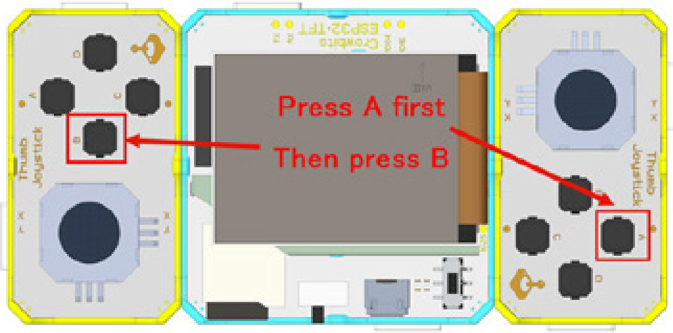
Click to upload the new firmware

Picture 16-7



Picture 16-8

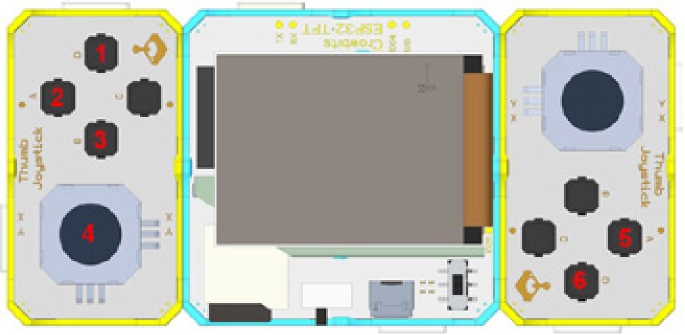
9. Restart the ESP32 module to make the firmware work. After that, click button A on the right thumb joystick sensor, and then click button B on the left thumb joystick sensor to install the firmware required for the game console, as shown in picture 16-9.



Picture 16-9

7. How to use

1. The function of the buttons on the game console is shown in picture 16-10 below.



Picture 16-10

① Adjust the game volume; click to increase the volume, when the maximum volume is reached, click again to return to the minimum volume.

② Return to the main menu, press it to return to the main menu interface.

③ Start button; after entering the emulator, press this button to enter the game.

④ Joystick part; used to control up, down, left, and right.

⑤ Confirm button; generally used in the game as an attack button and confirm button.

⑥ Back button; return to the previous level directory, generally used for jumping in the game.

After downloading the firmware, select the Emulators directory and click the confirm button to enter. There are various simulators in the Emulators directory, such as nes, gb, gbc, sms, gg, etc. Click to enter an emulator, select a game and press the confirm button to play.

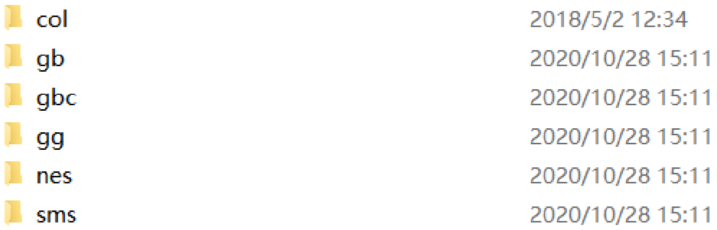
The game console is shown in picture 16-11:

Picture 16-11

8. Brainstorming

How exciting to complete a handheld game console of your own! In the roms folder, we have prepared 300 games for playing. But how to add other games into the game console?

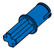
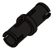
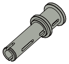
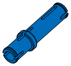
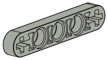
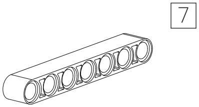
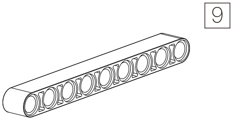
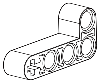
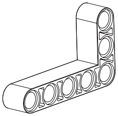
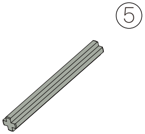
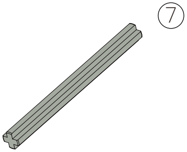
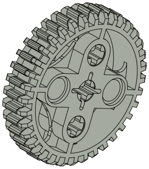
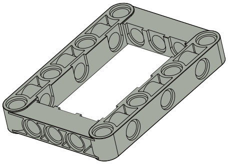
Don’t worry, you can download games from this link: https://www.downloadroms.io/
Just put the corresponding games under the corresponding emulator folder inside the roms directory, for example, put gb game inside the gb folder as shown in picture below.



col	2018/5/2 12:34
gb	2020/10/28 15:11
gbc	2020/10/28 15:11
gg	2020/10/28 15:11
nes	2020/10/28 15:11
sms	2020/10/28 15:11

Note: If there are too many games in a folder, it may get stuck after clicking. If it gets stuck, you need to reduce the number of game files in the SD card. The SD card can be unplugged only after the ESP32 module is powered off.

Parts List

 x4	 x2	 x25
 x2	 x2	 x4
 x2	 x4	 x4
 x4	 x1	 x1
 x1	 x2	

Crowbits

by Elecrow