4. Buzzer

1. Learning objectives

1. Learn to connect an external buzzer to the Raspberry Pi 2 motherboard for experiments.
2. Understand the use of active buzzers.

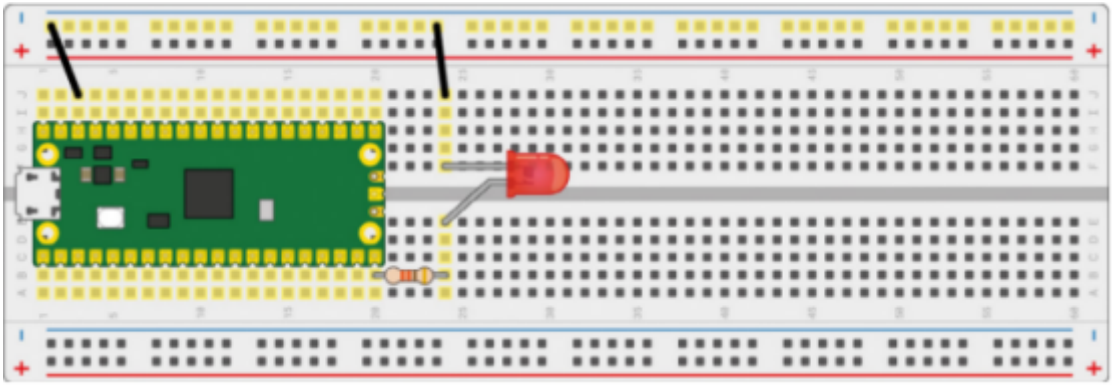
2. Hardware construction

This course.

Active buzzer*1

220Ω resistor*1

The circuit wiring diagram is shown below: (Replace the LED light with a buzzer, and connect the + sign of the buzzer to the resistor end)



3. Program analysis

Thonny programming

For the use of ThonnyIDE, please refer to the previous environment construction related courses.

```
#include <stdio.h>
#include "pico/stdlib.h"

int main()
{
    const uint BUZZER_PIN = 15;

    gpio_init(BUZZER_PIN);
    gpio_set_dir(BUZZER_PIN, GPIO_OUT);

    while (1) {
        for(int i = 0;i<80;i++)
        {
          gpio_put(BUZZER_PIN, 1);
          sleep_ms(1);
          gpio_put(BUZZER_PIN, 0);
          sleep_ms(1);
        }
        for(int i = 0;i<100;i++)
```

```
        {
          gpio_put(BUZZER_PIN, 1);
          sleep_ms(2);
          gpio_put(BUZZER_PIN, 0);
          sleep_ms(2);
        }
    }
}
```

#include "pico/stdlib.h"

This library includes common hardware and advanced libraries such as hardware_gpio and pico_time. It also introduces components such as pico_standard_link.

gpio_init(BUZZER_PIN);

Initialize the buzzer pin, clear its output enable, and clear any output value.

gpio_set_dir(BUZZER_PIN,GPIO_OUT);

Set the buzzer pin to output mode.

Through the While loop, two different for loops make the buzzer emit different frequencies of sound

4. Program creation and burning

Please refer to the previous tutorial on creating a project to create a project. I will not repeat it here.

After writing the program, before compiling, you need to add the standard library to the build according to the program you wrote. Open the CMakeLists.txt in the project and check target_link_libraries (buzzer pico_stdlib). You can see that since the previous program only uses the pico_stdlib library, only pico_stdlib is added here. If the function you use adds other libraries, you need to add the corresponding libraries. If you are not sure which library to add, you can check the official pico-examples case downloaded earlier, which contains examples of various functions and corresponding CMakeLists.txt files for reference.

After the modification is completed, save and exit, and enter the bulid path of the project.

Enter the following command to compile

```
cmake .. -G "NMake Makefiles"
nmake
```

After the compilation is completed, files in formats such as .bin .hex .elf .uf2 can be generated in the build directory.

Drag u2f in the above file into the disk recognized by Pico 2 for burning. (Note: When burning for the first time, the code is empty. Pico 2 can directly recognize the disk when connected to USB. When there is an executable program in it, you need to press and hold the BOOTSEL button and then connect the USB)

5. Experimental phenomenon

After the program is downloaded, we can hear the buzzer playing sounds of different frequencies.