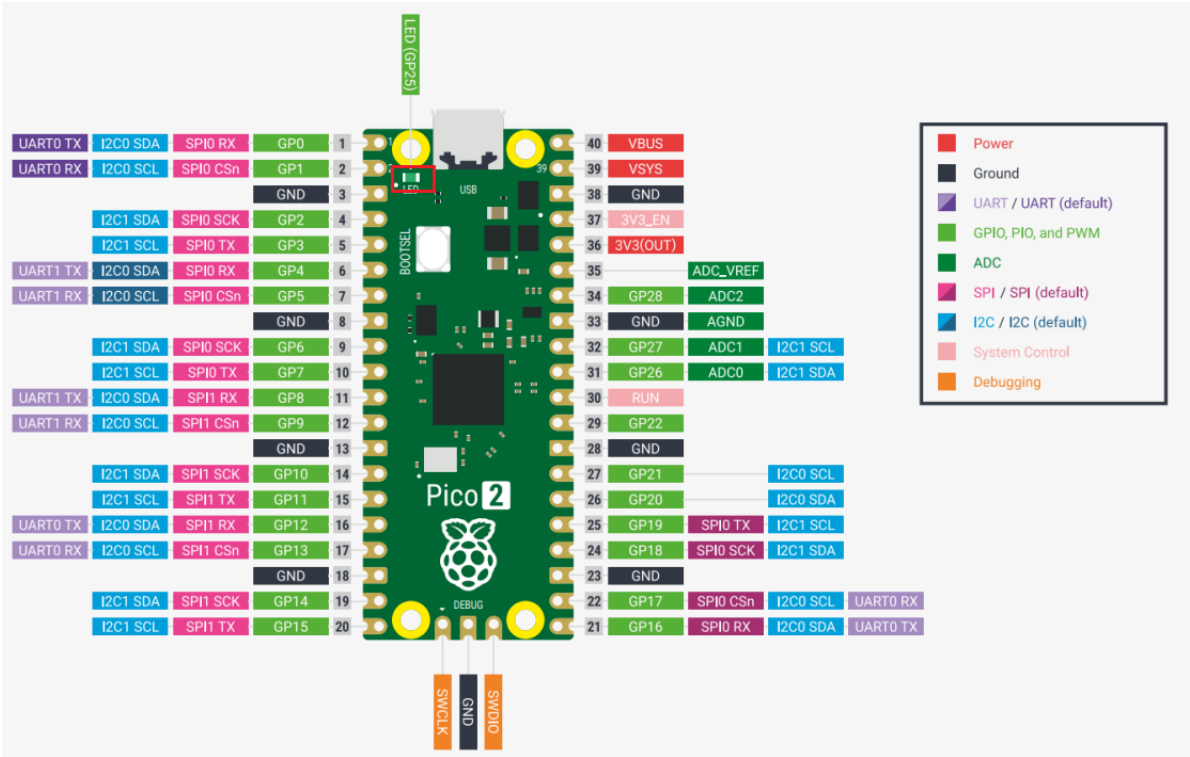
1. Flashing the onboard LED light

1. Learning objectives

1. Learn the basic use of the pins of the Raspberry Pi Pico 2 motherboard.
2. Learn how to control the onboard LED light.

2. Hardware construction

This course does not require additional hardware equipment and can directly use the onboard LED light on the Raspberry Pi Pico 2 motherboard.



3. Program Analysis

```
#include "pico/stdlib.h"
#define   LED_PIN   25

int main()
{
    gpio_init(LED_PIN);
    gpio_set_dir(LED_PIN, GPIO_OUT);

    while(1)
```

```
    {
        gpio_put(LED_PIN, 1);
        sleep_ms(250);
        gpio_put(LED_PIN, 0);
        sleep_ms(250);
    }
}
```

#include "pico/stdlib.h"

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

gpio_init(LED_PIN);

Initialize IO25, clear its output enable, and clear any output value.

gpio_set_dir(LED_PIN, GPIO_OUT);

Set IO25 to output mode.

sleep_ms(250);

Delay 250 milliseconds.

gpio_put(LED_PIN, 1);

Set a GPIO high/low, the first parameter is the GPIO number, and the second parameter input value is 0 or 1.

4. Program creation and burning

To create a project, please refer to the previous tutorial on creating a project, which will not be repeated here.

After writing the program, you need to add the standard library to the build according to the program you wrote before compiling. Open the CMakeLists.txt in the project and check target_link_libraries(blink 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 used 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 to burn it. (Note: When burning for the first time, it is an empty code. 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 USB)

5. Experimental phenomenon

After the program is downloaded, we can see that the LED light on the Raspberry Pi Pico 2 development board flashes every 5 seconds.