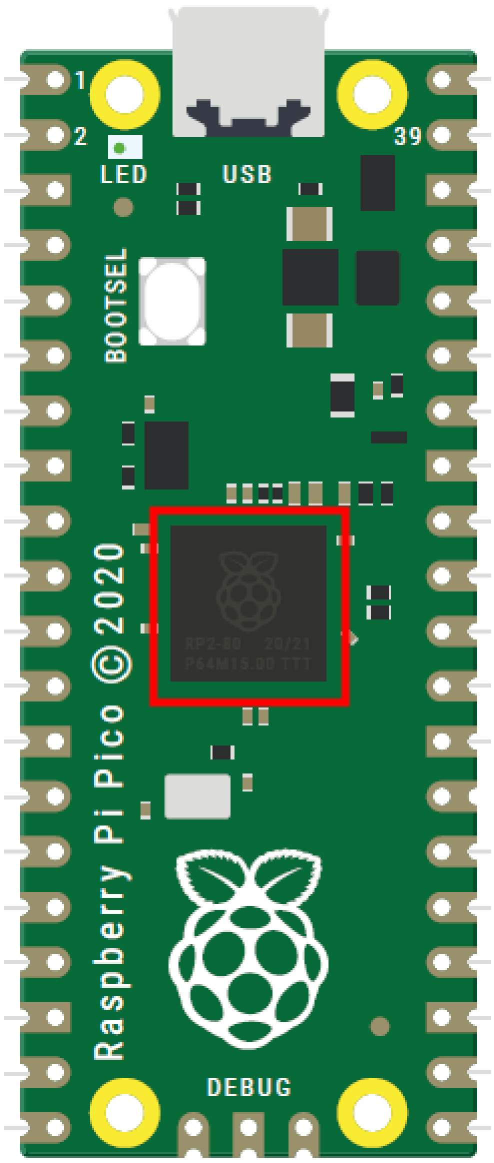
3.2 Onboard temperature sensor

I. Learning objectives

1. Learn the basic use of the ADC pins of the Raspberry Pi Pico 2/Pico board.
2. Learn how to read the temperature through the onboard temperature sensor.

II. Hardware usage

No additional hardware is required for this course, and the temperature sensor on the Raspberry Pi Pico 2/Pico board is directly utilized.



The RP2040 microcontroller of the Raspberry Pi Pico and the RP2350 microcontroller of the Raspberry Pi Pico 2 are digital devices, just like all mainstream microcontrollers: it is composed of thousands of transistors, these tiny switch-like devices are either on or off. Therefore, your Pico cannot really understand an analog signal - it can be anything on the spectrum between fully off and fully on - without relying on additional hardware: an analog-to-digital converter (ADC).

An ADC has two key characteristics: its resolution, measured in digital bits, and its channels, or how many analog signals it can accept and convert at once. The ADC in your PICO has a resolution of 12 bits, which means it can convert analog signals to digital numbers ranging from 0 to 4095 - although this is handled in MicroPython and converted to 16-bit numbers ranging from 0 to 65,535, so it behaves the same as the ADC on other MicroPython microcontrollers. It has three channels brought out to the GPIO pins: GP26, GP27, and GP28, which are also referred to as GP26_ADC0, GP27_ADC1, and GP28_ADC2 for analog channels 0, 1, and 2. There is also a fourth ADC channel, which is connected to a temperature sensor built into the RP2040.

Three, program analysis

Code path: Code -> 1.Basic course -> 2. On board temperature sensor.py

```
import machine
import utime
sensor_temp = machine.ADC(29)
conversion_factor = 3.3 / (4096-1)
while True:
    reading = sensor_temp.read_u16()* conversion_factor
    temperature = reading
    print(temperature)
    utime.sleep(2)
```

import machine

The machine library contains all the instructions MicroPython needs to communicate with Pico and other MicroPython-compatible devices, extending the language of physical computing.

import time

The "time" library. This library handles everything to do with time, from measuring it to inserting delays into your program. The unit is seconds.

sensor_temp = machine.ADC(29)

Initializes an ADC (Analog to Digital Converter) object that will be used to read the analog signal connected to pin 29 of the microcontroller.

conversion_factor = 3.3 / (4096-1)

Calculates the conversion factor used to convert the ADC reading (range 0-4095) to a voltage value. The Raspberry Pi Pico uses 3.3V power, so the conversion factor is 3.3V divided by the maximum value of the ADC minus 1 (i.e. 4095).

reading = sensor_temp.read_u16() * conversion_factor

Reads the 16-bit unsigned integer from the ADC (read_u16()) and multiplies the reading by the conversion factor defined above to get the voltage value.

temperature = reading

The voltage value is assigned to the temperature variable.

print(temperature)

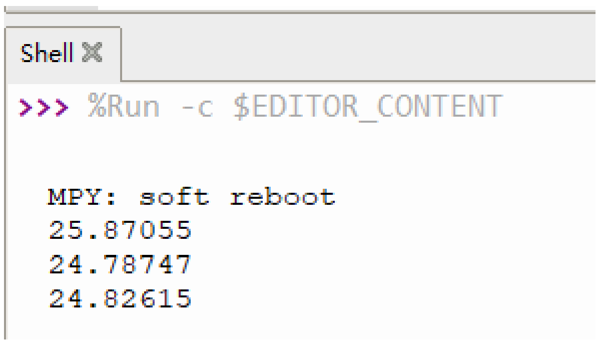
Prints the value.

utime.sleep(2)

This calls the sleep function from the utime library and reenters the loop after 2 seconds.

Fourth, Experimental Phenomenon

After the program is downloaded, we can see that the shell window prints the temperature value every 2 seconds.



```
>>> %Run -c $EDITOR_CONTENT

  MPY: soft reboot
  25.87055
  24.78747
  24.82615
```