[Infrared Receiving Sensor Edition]

Smart Remote Controller production

- Understanding the mechanism of receiving sensors
- A program that acquires and saves remote control signals

Table of Contents <<Infrared Receiving Sensor Edition>>

1. Overview
   1-1. Overall flow of smart remote controller production
   1-2. Items to use
   1-3. About the development environment Arduino

2. Mechanism of Infrared Receiving Sensor

3. circuit diagram

4. Wiring diagram

5. software

6. operation check

1-1. Overall flow of Smart Remote Controller production

No	Item	Content	Hard	Soft	Note
1	Overview	Overall flow, system configuration, items used, reasons for selection, development environment, etc.	-	-	Delivered in another video
2	LED	Learn the basics for beginners. We will make "L blinking" that lights up and blinks the LED.	○	○	
3	Infrared receiving sensor	Description of infrared receiving sensor Schematic to Wiring, Software	○	○	this time this video
4	Infrared transmission LED	Infrared transmission LED description Schematic to Wiring, Software	○	○	Delivered in another video
5	LED operation with smartphone(at home)	We will create software to operate the LED with smartphone. (Web server function, SPIFFS operation)	-	○	
6	Remote control with smartphone(at home)	We will create software that to operate the remote control with smartphone indoors. (Button name, signal save/read)	-	○	
7	Operate from outside And AI speaker cooperation	We will create software to operate the remote control with smartphone from the outdoors, and AI speaker cooperation.	-	○	

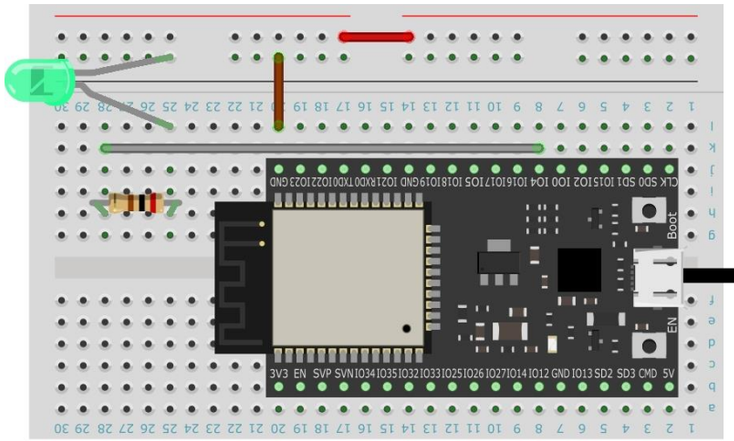
1-2. List of Parts

Can be downloaded from the Hobby-IT site <<Overview>> page

NO	Item	quanti	Image	Item	URL(Japanese Shop)	Price(yen)	Note
1	ESP32 development board	1		ESP32-DevKitC-32E ESP32-WROOM-32E development board 4MB	https://akizukidenshi.com/catalog/g/gM-15673/	1600	19Pin x 2 rows
2	Breadboard 6 hole [EIC-3901]	1		Breadboard 6 hole plate EIC-3901	https://akizukidenshi.com/catalog/g/gP-12366/	460	
3	Resistor 10 Ω	3		Carbon resistor (carbon film resistor) 1/2W 10Ω (100 pieces)	https://akizukidenshi.com/catalog/g/gR-07795/	100	For infrared transmission LED
4	Resistor 200 Ω	2		Carbon resistor (carbon film resistor) 1/2W 200Ω (100 pieces)	https://akizukidenshi.com/catalog/g/gR-07807/	100	For green LED and transistor
5	Green LED	1		3mm yellow-green LED 570nm 70 degrees OSG8HA3Z74A	https://akizukidenshi.com/catalog/g/gI-11637/	10	For status display
6	Infrared receiving sensor	1		Infrared remote control receiver module OSRB38C9AA (2 pieces)	https://akizukidenshi.com/catalog/g/gI-04659/	100	
7	Infrared transmission LED	3		5mm infrared LED 940nm OSI5LA5113A gray (10 pieces)	https://akizukidenshi.com/catalog/g/gI-12612/	100	For infrared transmission LED
8	Transistor	1		Transistor 2SC2655L-Y-T9N-B 50V2A (10 pieces included)	https://akizukidenshi.com/catalog/g/gI-08746/	130	For infrared transmission LED
9	Bread board Jumper	1		Breadboard jumper wire 14 types x 5	https://akizukidenshi.com/catalog/g/gP-02315/	300	
total						**2,900**	Postage +500 yen required

1-3. the development environment “Arduino”

We will use Arduino as the development environment.



ESP32 Development Board

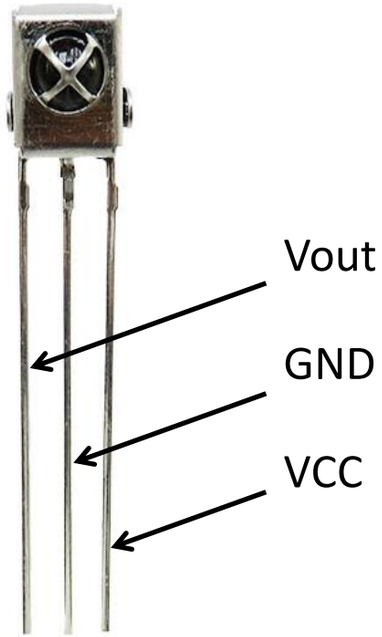
MicroUSB Cable

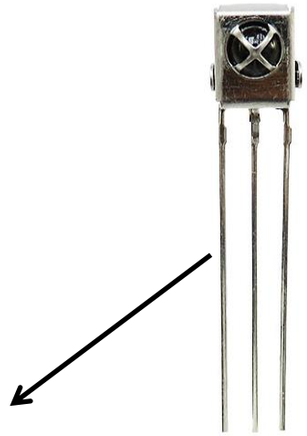
P C (Windows/Mac/Linux)

【Arduino Official site】
https://www.arduino.cc/
Downloadable

2-1. Mechanism of Infrared Receiving Sensor

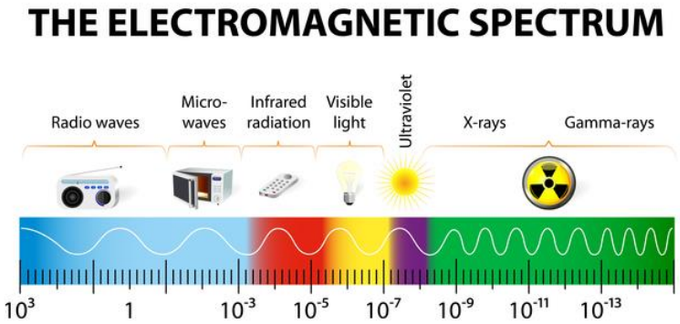
When infrared rays are received, the Vout terminal becomes LOW (voltage). (No signal is HIGH)
These HIGH and LOW signals are recognized by the ESP32 port and acquired.





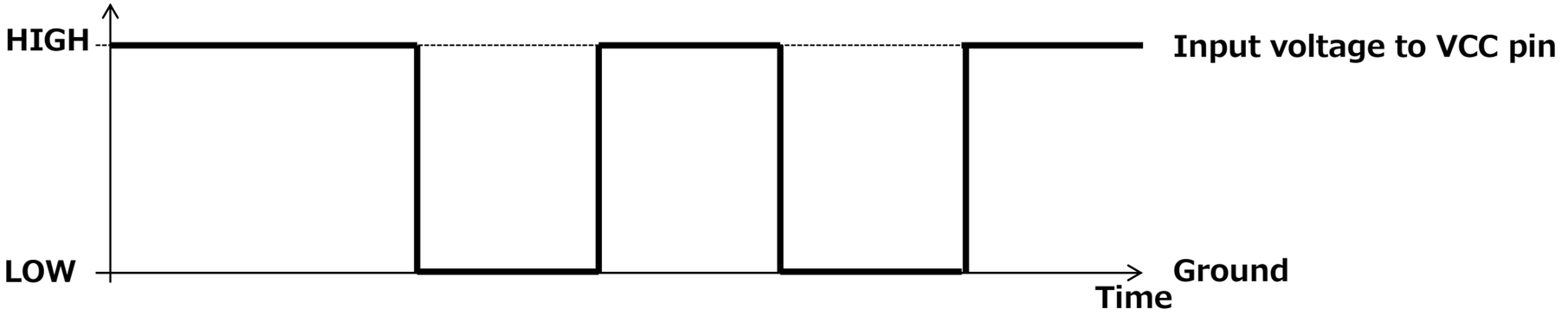
Infrared
(A kind of electromagnetic wave, same as light)

Vout port voltage becomes LOW when infrared is received.
(No signal is HIGH)



*1: Relationship between electromagnetic waves and frequencies

Vout
output voltage



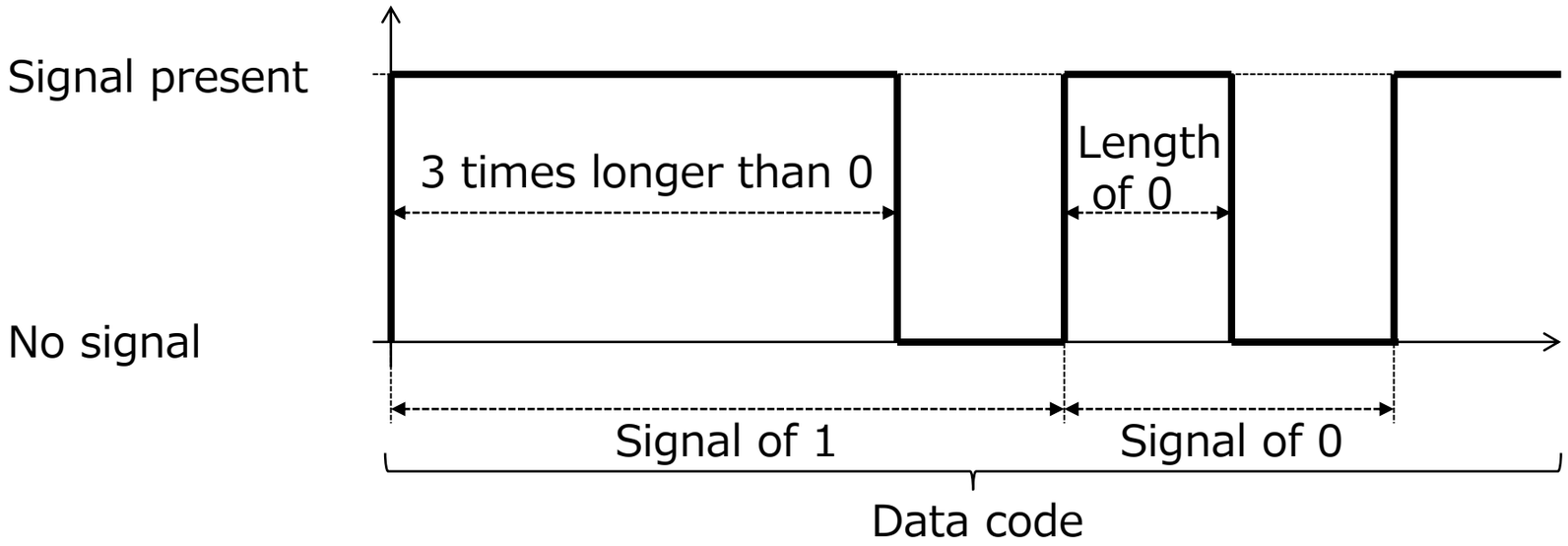
【wiring】
- Vout: ESP32 receiving terminal
- GND: ESP32 ground terminal
- VCC : ESP32 3.3V pin

*1 : https://k-comfort.co.jp/post-knowledge-infrared-1/

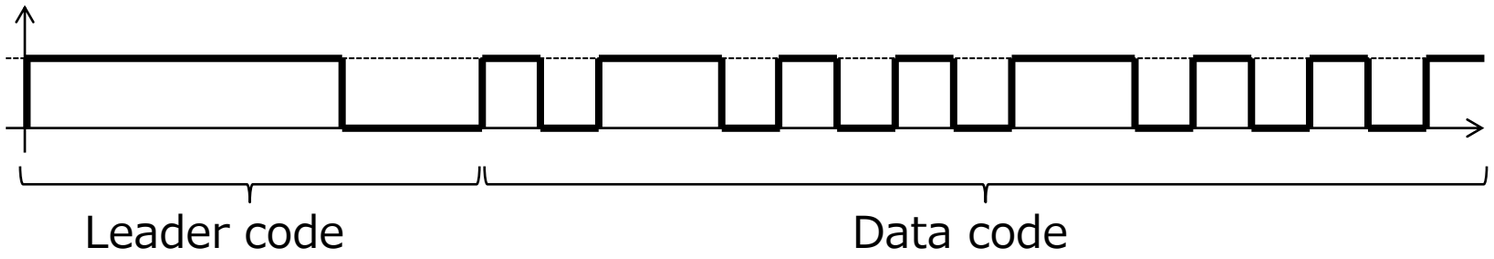
2-2. Remote control signal

The distinction between 0 and 1 is represented by the difference in length of HIGH (or LOW).

[0, 1 signal example]

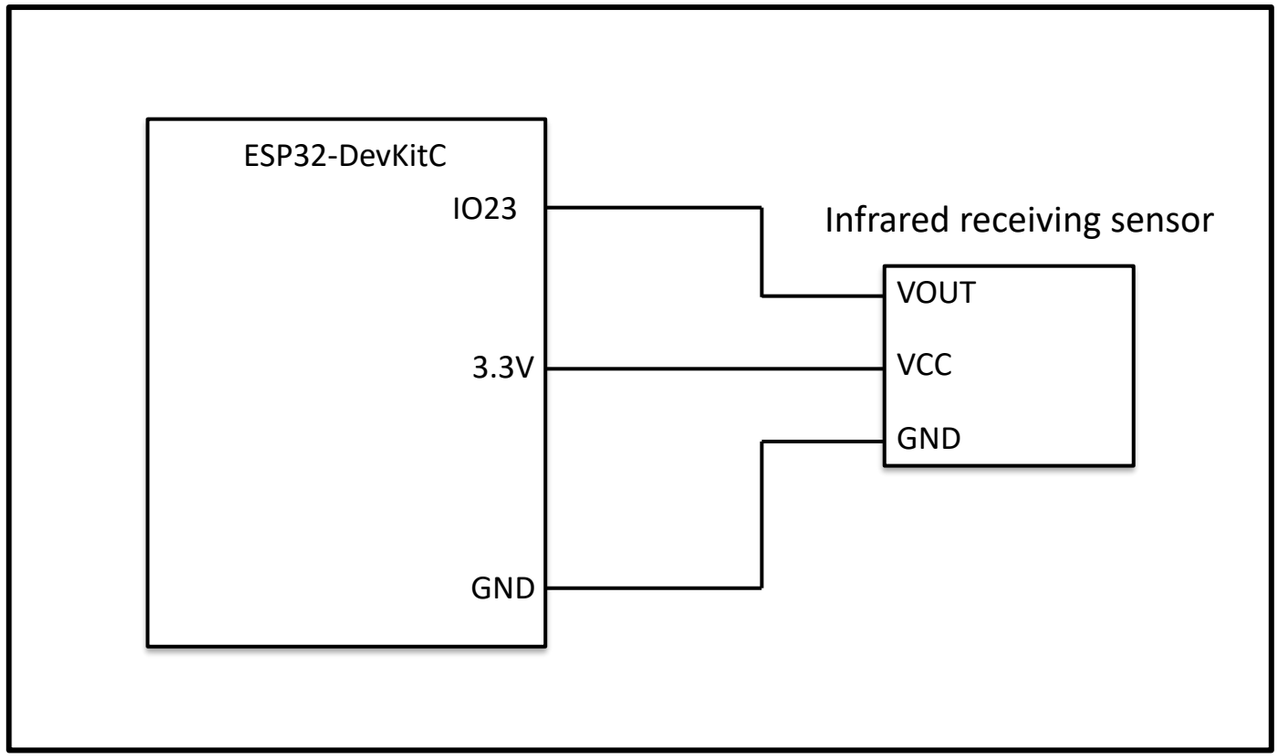


[Reference: General remote control signal]



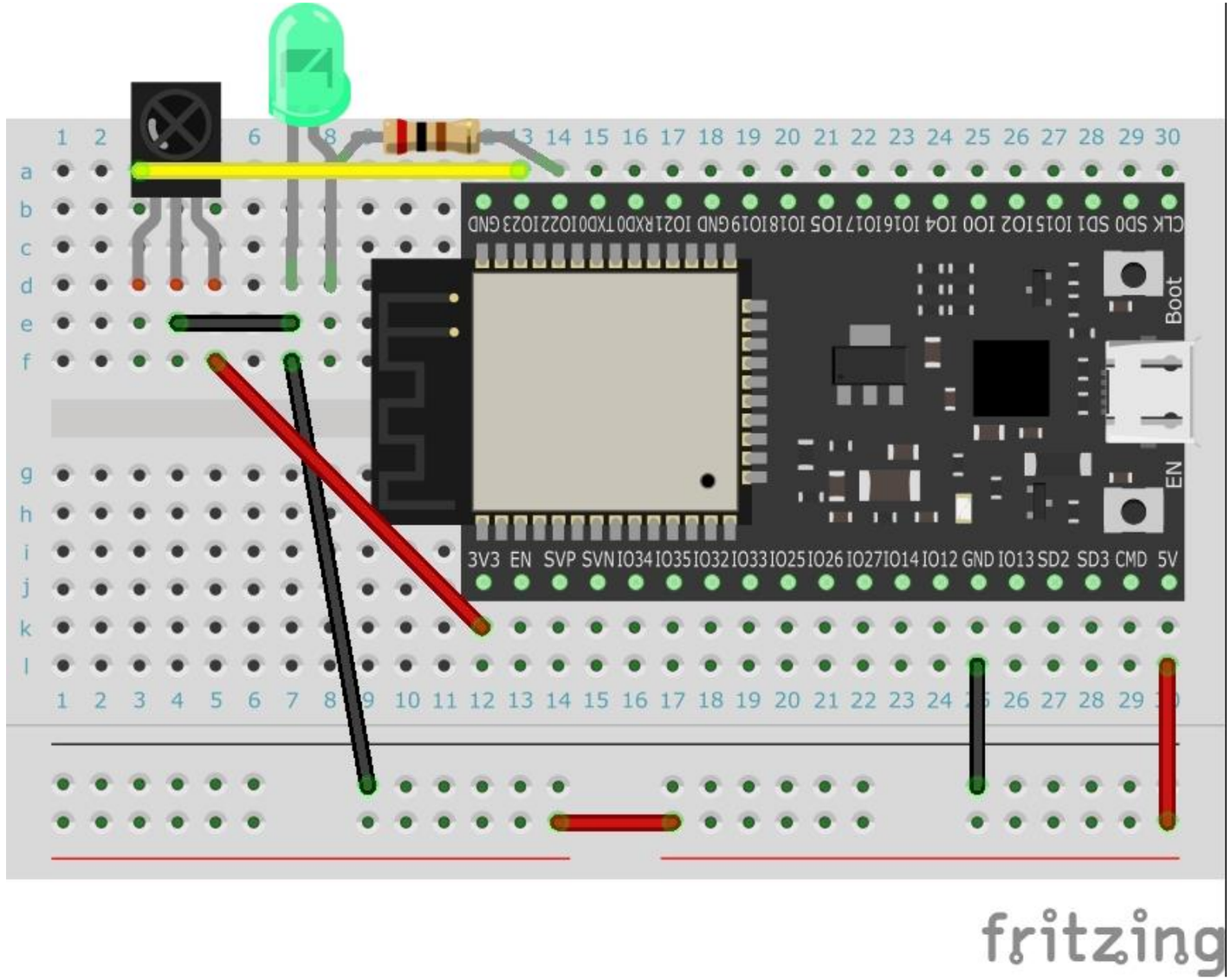
3. Circuit diagram

Set IO23 of ESP32 to receive and acquire the signal
from Vout of the infrared receiving sensor



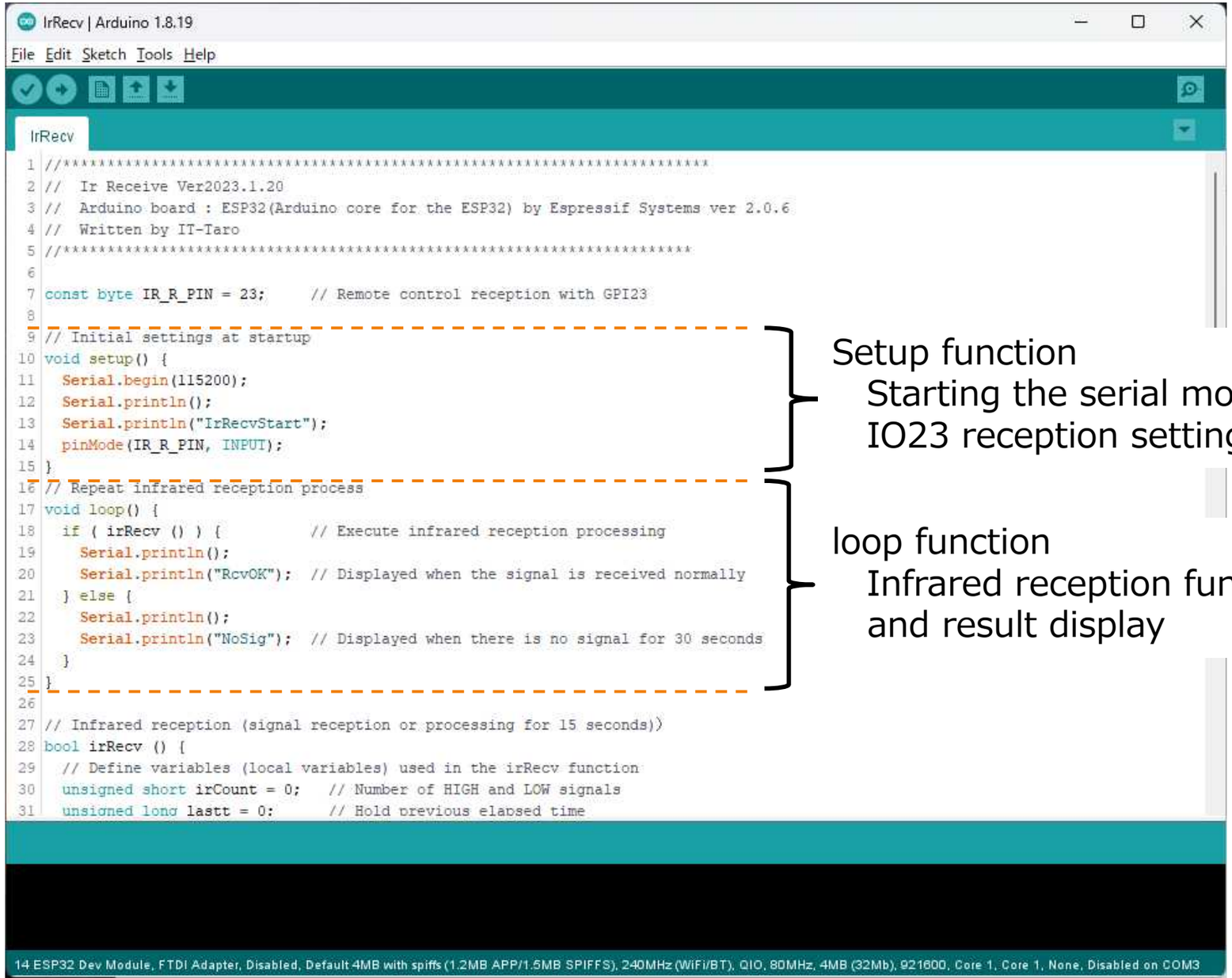
4. Wiring diagram

Note) LED wiring was performed in <<LED Edition>>.



5-1. Software

Note) Since we have learned the basics in the LED edition, we will understand the software part related to the infrared receiving sensor.

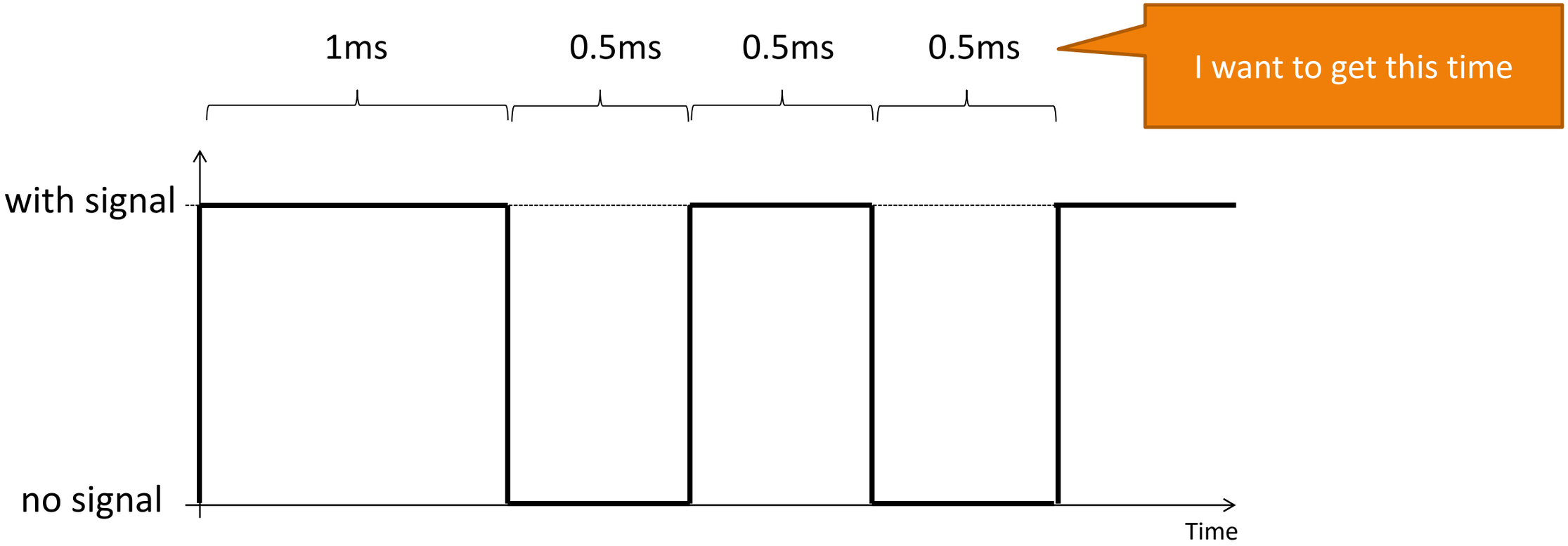


Setup function
- Starting the serial monitor and IO23 reception settings

loop function
- Infrared reception function execution and result display

5-2. What you want to achieve with software

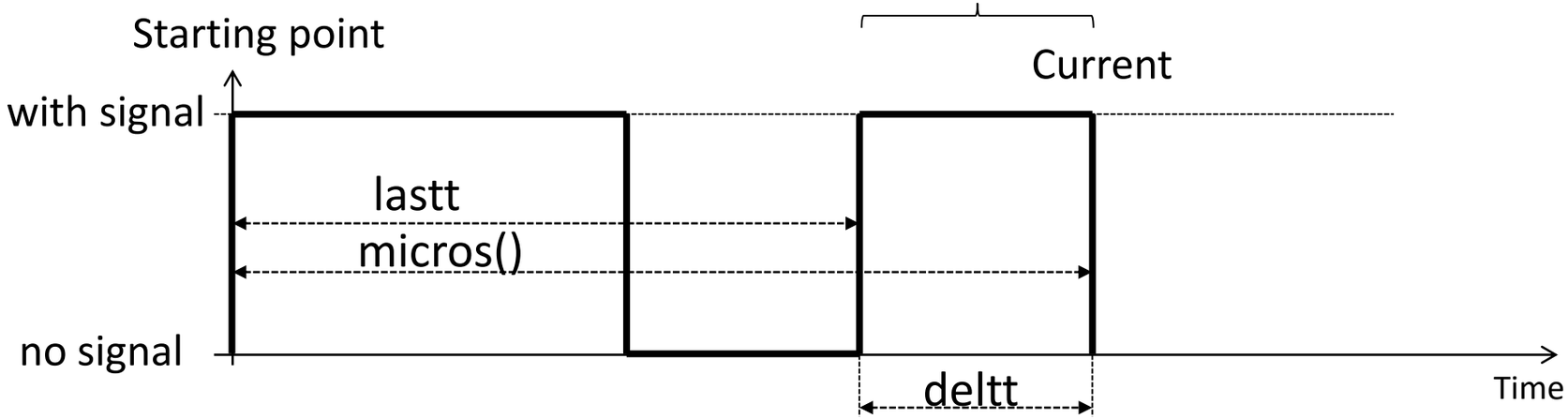
Get the length of time between "present" and "not present" for the remote control signal



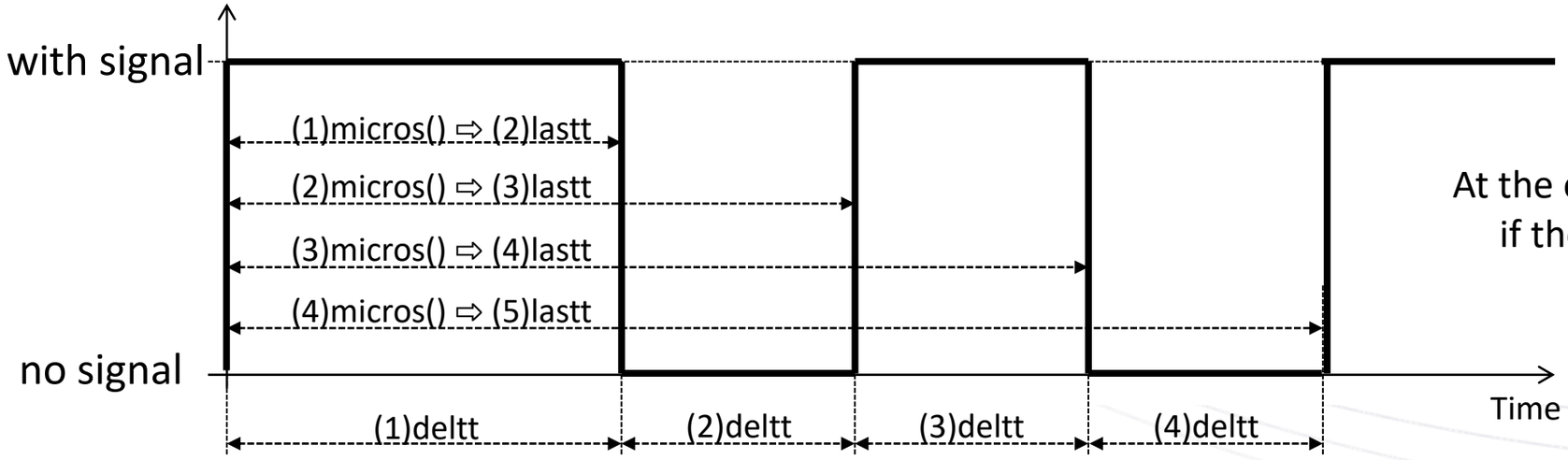
●The diff from the current change to the previous change

current time　　last time

Get this time ＝ micros() － lastt



●Calculating by replacing the current time with the previous time



At the end, it is judged to be finished
if there is no change for 0.5 sec

5-3. Software understanding

```
27 // Infrared reception (signal reception or processing for 15 seconds))
28 bool irRecv () {
29   // Define variables (local variables) used in the irRecv function
30   unsigned short irCount = 0;   // Number of HIGH and LOW signals
31   unsigned long lastt = 0;      // Hold previous elapsed time
32   unsigned long deltt = 0;      // Difference time from previous one
33   unsigned long sMilli;         // Start time of this process
34   unsigned long sMicro;         // Processing start time
35   unsigned long wMicro;         // wait start time
36   bool rState = 1;              // Infrared receiver module status 0: LOW, 1: HIGH
37   sMilli = millis();            // Get the current system time (get in milliseconds)
38   // Infinite loop until specific condition (signal received or 15 seconds elapsed)
39   while(1) {
40     // Get start time to wait for Ir reception
41     wMicro = micros();          // Get current system time (get in microseconds)
42     // Waiting for reception of inverted signal
43     while (digitalRead(IR_R_PIN) == rState) {
44       // When 0.5 seconds or more have passed after starting to wait
45       if (micros() - wMicro > 500000) {
46         // After waiting for 0.5 seconds or more
47         if ( irCount > 10 ) {
48           return true;    // Successfully completed
49         }
50         // If there are not more than 10 0,1 signals, receive again from zero due to noise
51         irCount = 0;
52         delay(1); // For watchdog timer (must be reset within 3 seconds)
53       }
54       // After 15 seconds or more of processing, T.O.
55       if ( millis() - sMilli > 15000 ) {
56         return false;    // Ends after 15 seconds (no reception)
57       }
58     }
59     // Get the current time and elapsed time at the start of signal reception
60     if ( irCount == 0 ) {
61       sMicro = micros();
62       lastt = 0;
63       irCount++;
64       Serial.println("ir:");
65     // Processing after starting signal reception processing (irCount is 1 or more)
66     } else {
67       // Calculate the elapsed time from the time when the status change of the sig
68       deltt = ( (micros() - sMicro)/ 10 ) - lastt;
69       // Save the last changed elapsed time for the next elapsed time calculation
70       lastt = lastt + deltt;
71       irCount++;
72       Serial.print(deltt);
73       Serial.print(",");
74     }
75     // Change the value to detect state change in While next time
76     rState = !rState;
77   }
78 }
```

Declaration of variables to use

① Repeat while processing

No change over 0.5 seconds

10 or more signals are terminated normally as success

Signals less than 10 times clear as possible noise

Failed after 15 seconds

② Repeat processing until there is a change in HIGH and LOW

First change is StartTime retention etc.

From the second time onwards, get the time difference from the previous time and display it on the serial monitor

Process when there is a change in HIGH, LOW. It is processed through while of ②.

[Main variables and system functions]

Starting time : sMicro (Start Time)
Last change : last (Last Time)
H, L change count : irCount

millis() : System elapsed time milliseconds ← for 0.5/15 sec
maicros() : System elapsed time microseconds ← for signals