[Infrared transmission LED version]

Smart Remote Controller production

- Understanding Infrared Transmitting LEDs and Transistors
- Program the remote control signal (38 kHz modulation)

Table of Contents <<Infrared transmission LED>>

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

2. Infrared transmitting LED and transistor
   - 2-1. Infrared transmission LED
   - 2-2. remote control signal
   - 2-3. Resistance value calculation (infrared transmission LED)
   - 2-4. transistor

3. circuit diagram

4. Wiring diagram

5. Software

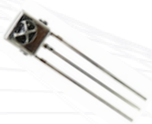
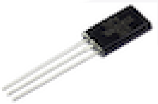
6. operation check

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.	-	-	
2	LED	Learn the basics for beginners. We will make "L blinking" that lights up and blinks the LED.	○	○	Delivered in another video
3	Infrared receiving sensor	Description of infrared receiving sensor Schematic to Wiring, Software	○	○	
4	Infrared transmission LED	Infrared transmission LED description Schematic to Wiring, Software	○	○	this time this 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)	-	○	Delivered in another video
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.	-	○	

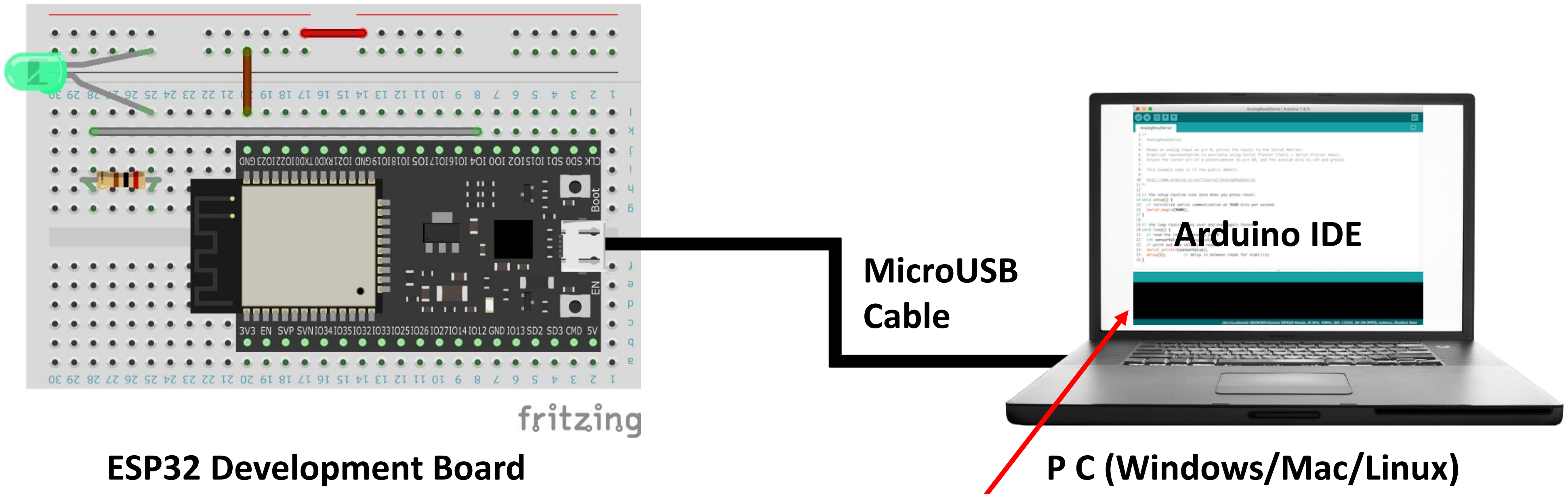
1-2. List of Parts

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

NO	Item	quantity	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.



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

2-1. Infrared transmission LED

The infrared transmission LED "OSI5LA5113A" is available
with a forward voltage of 1.35V and a forward current of 100mA.

- "OSI5LA5113A" Data Sheet

▪Absolute Maximum Rating (Ta=25℃)

Item	Symbol	Value	Unit
DC Forward Current	I_F	100	mA
Pulse Forward Current#	I_{FP}	1000	mA
Reverse Voltage	V_R	5	V
Power Dissipation	P_D	180	mW
Operating Temperature	Topr	-30 ~ +85	℃
Storage Temperature	Tstg	-40~ +100	℃
Lead Soldering Temperature	Tsol	260℃/5sec	-

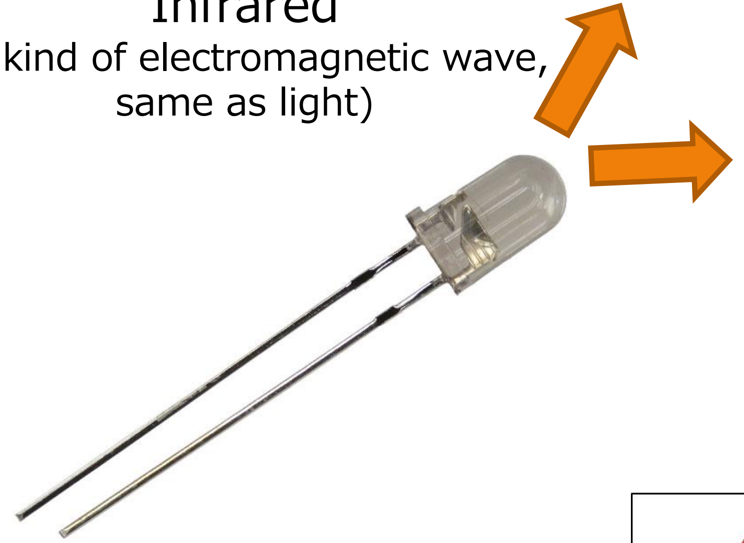
Pulse Width ≤100us, Duty≤1/100

▪Electrical -Optical Characteristics (Ta=25℃)

Item	Symbol	Condition	Min.	Typ.	Max.	Unit
DC Forward Voltage*1	V_F	I_F=100mA	-	1.35	1.6	V
DC Reverse Current	I_R	V_R=5V	-	-	10	µA
Peak Wavelength*2	λ_p	I_F=50mA	-	940	-	nm
Radiant Power*3	Po	I_F=50mA	-	12	-	mW
Radiant Intensity*4	Ie	I_F=50mA	35	55	-	mW/Sr
50% Power Angle	$2\theta_{1/2}$	I_F=50mA	-	15	-	deg

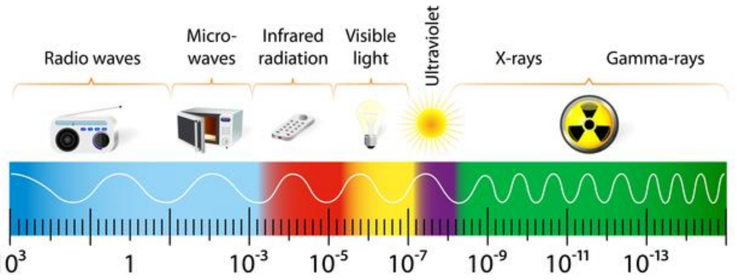
*1 Tolerance of measurements of forward voltage is ±0.1V
*2 Tolerance of measurements of peak wavelength is ±1nm
*3 Tolerance of measurements of Radiant Power is ±15%
*4 Tolerance of measurements of Radiant Intensity is ±15%

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

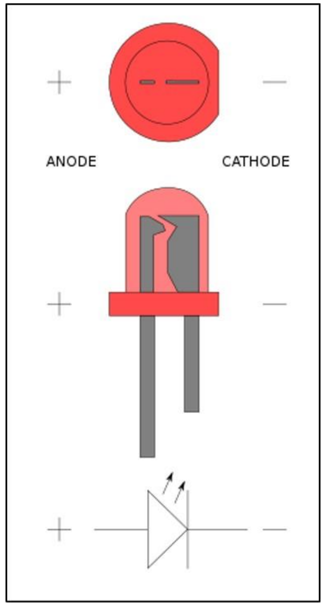


Forward voltage: 1.35
VForward current: 100mA

THE ELECTROMAGNETIC SPECTRUM *1



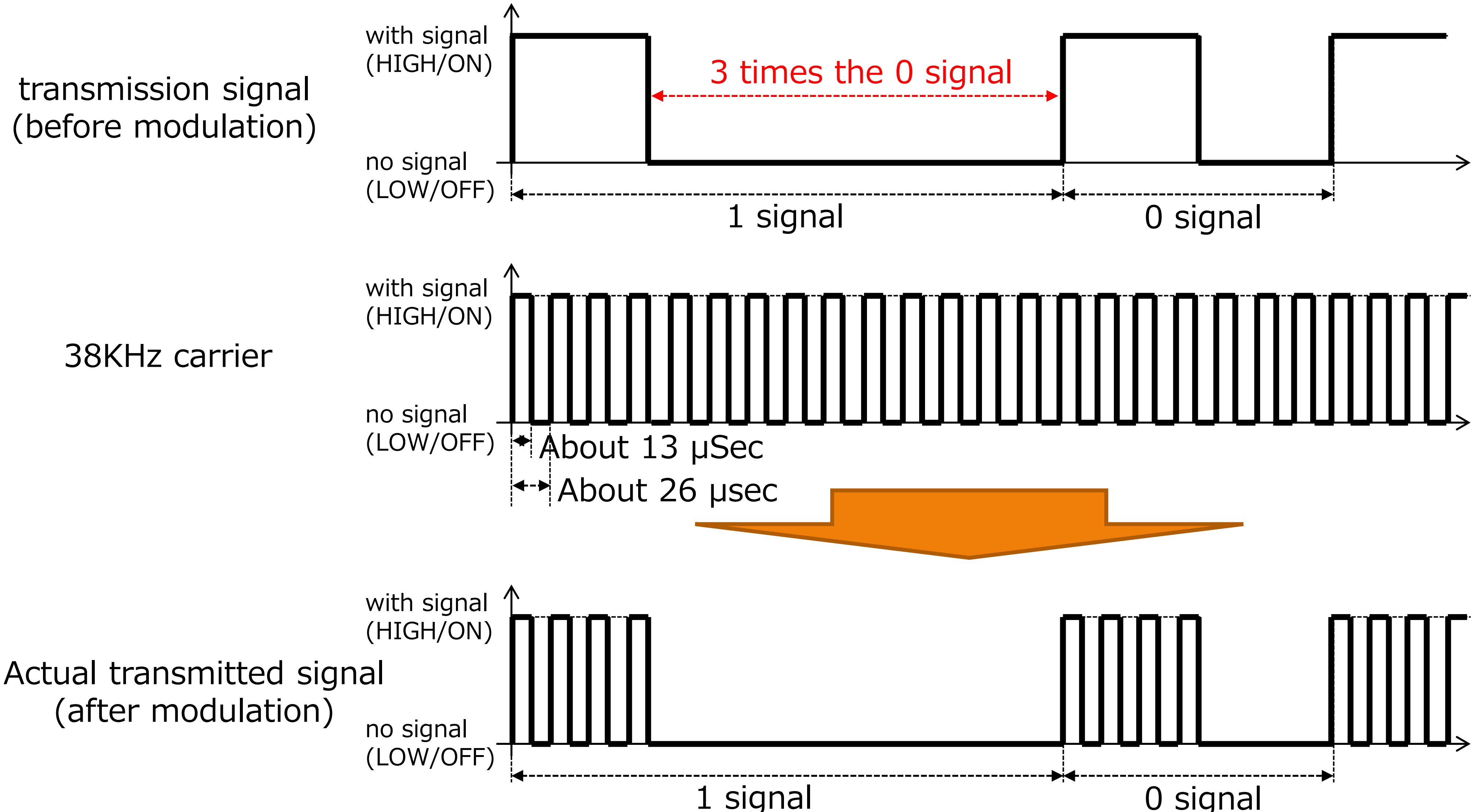
*1：https://k-comfort.co.jp/post-knowledge-infrared-1/



【caution】
The longer leg is the anode and the positive is wired.

2-2. Remote control signal

0 and 1 of the remote control signal are represented by lengthening the time of HIGH or LOW.
The actual transmission signal is modulated with a 38KHz signal.



[Infrared frequency]
Frequency: 310-320 THz
Wavelength: 940-950 nm
(modulate this infrared with a carrier wave)

[Modulation reason] (assumed)
- Increase the maximum output and increase the distance
- Reduce the influence of noise
- Lower power consumption
- Prolong the life of the LED

2-3. Resistance value calculation (for Ir LED)

under consideration

If the resistance value is small, the LED will be damaged, and if it is large, the remote control will not be able to reach the distance.
Since the signal is turned ON/OFF, a resistance value of 10 Ω is used as the forward current that is constantly applied.

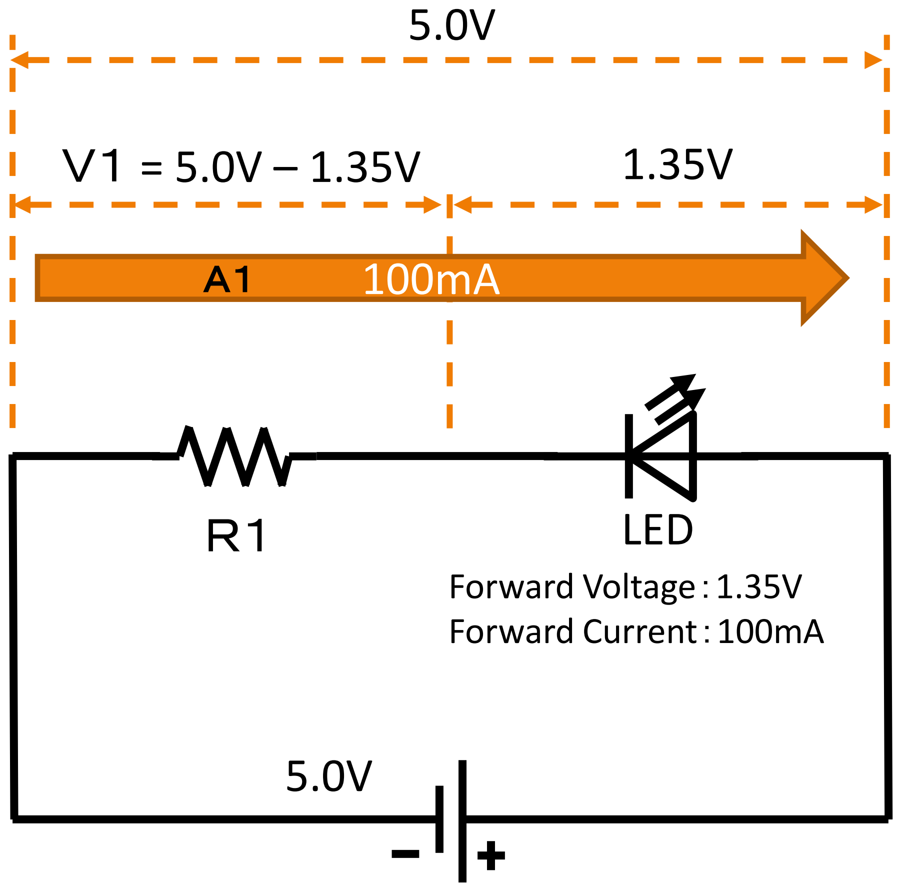
● Resistance value calculation

Forward voltage: 1.35 V
Forward current: 100mA

$$R1 = \frac{V1}{A1}$$

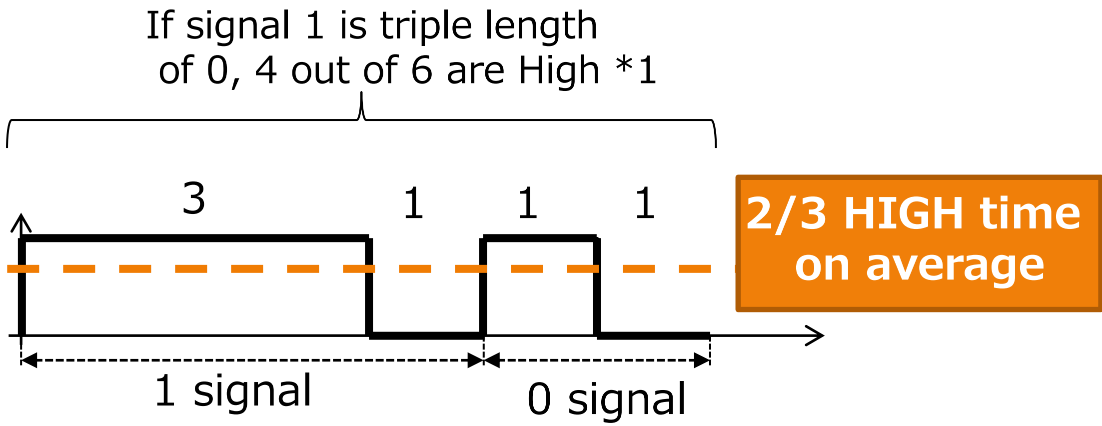
$$= \frac{5.0\,V - 1.35\,V}{100mA}$$

$$= 36.5\,\Omega$$



● Consider signal ON/OFF

*1: Assuming that the numbers of 0 and 1 bits are the same



Duty [Overall percentage of HIGH(ON)]

$$\frac{4}{6} = \frac{2}{3}$$ → 38KHz modulation, so 1/2 → $$\frac{1}{3}$$

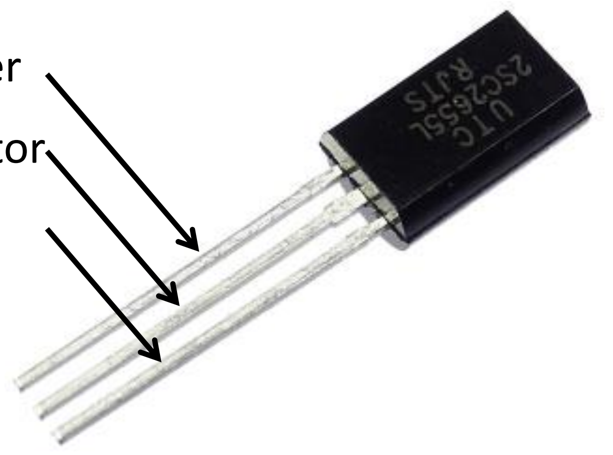
Calculated as an average of 100mA, considering signal ON/OFF

$$36.5\,\Omega \times \frac{1}{3} \fallingdotseq 10\,\Omega$$

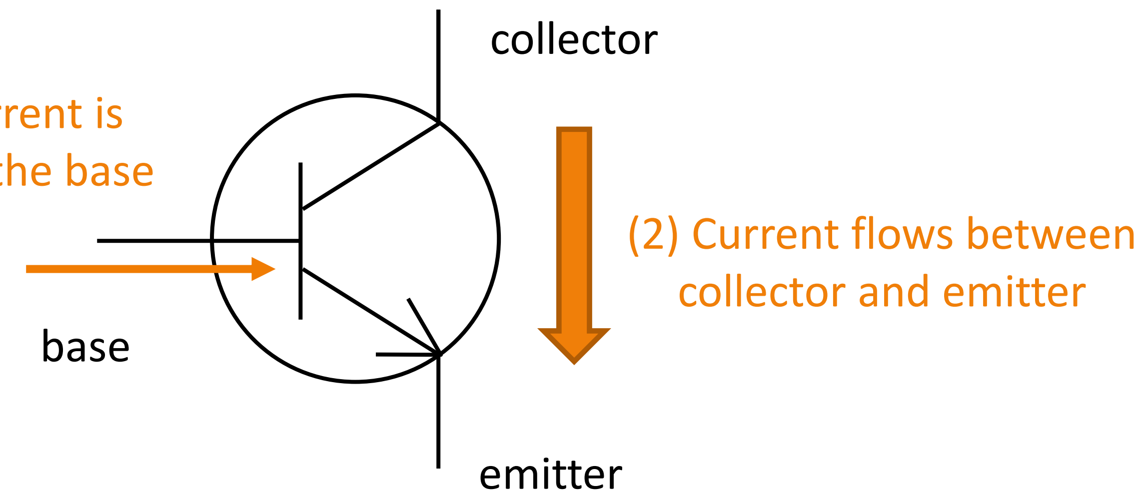
2-4. Transistor

- Transistor

emitter
collector
base

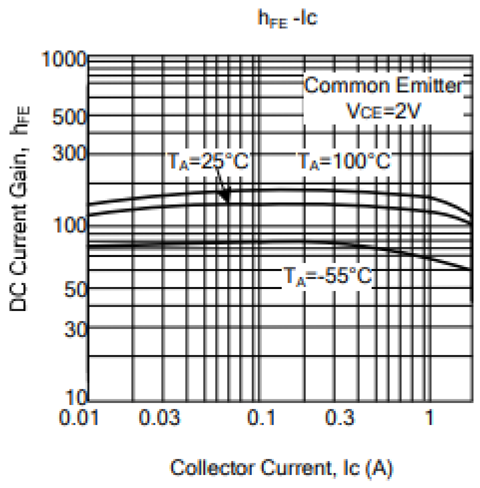


- Transistor operation



- Transistor base resistance calculation

“2SC2655L” Data Sheet



Amplification factor hFE is about 100 times

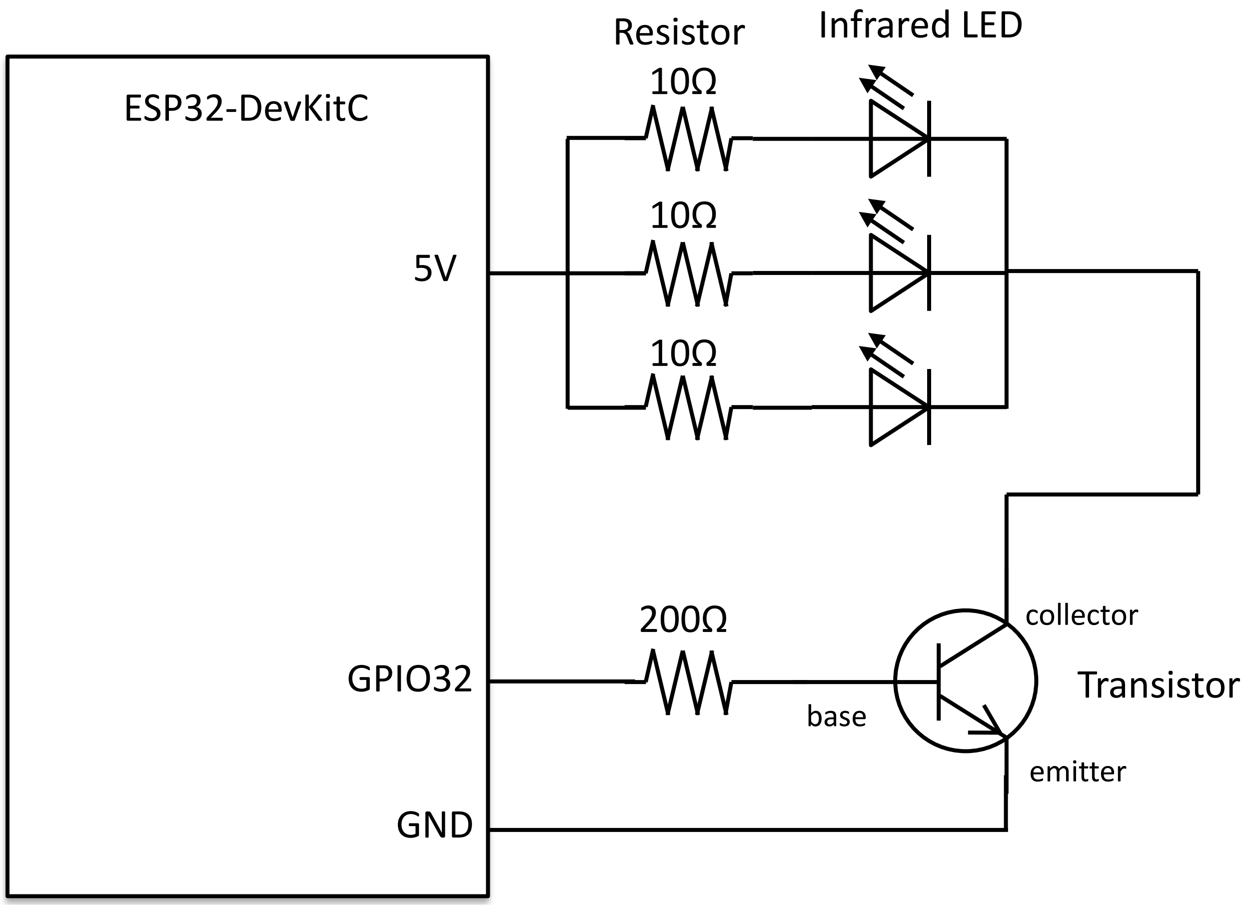
<Transistor specifications>
Base- Emitter Saturation Voltage (VBE) : 1.2 V
DC Current Gain (hFE) : 100

[Resistance connected to the base]
(1) Current you want to pass through the infrared transmission LED
[current between collector and emitter]
⇨ 100mA (forward current) x 3LEDs x 3 *1 = 900mA
***1: In consideration of PWM, it flows three times instantaneously.**
(2) Current you want to flow to the base
900mA / 100(hFE)⇨ 9 mA
(3) Resistor installed on the base
(3.3V – 1.2V) / 9mA = 0.23KΩ ⇨ about 200Ω

200Ω

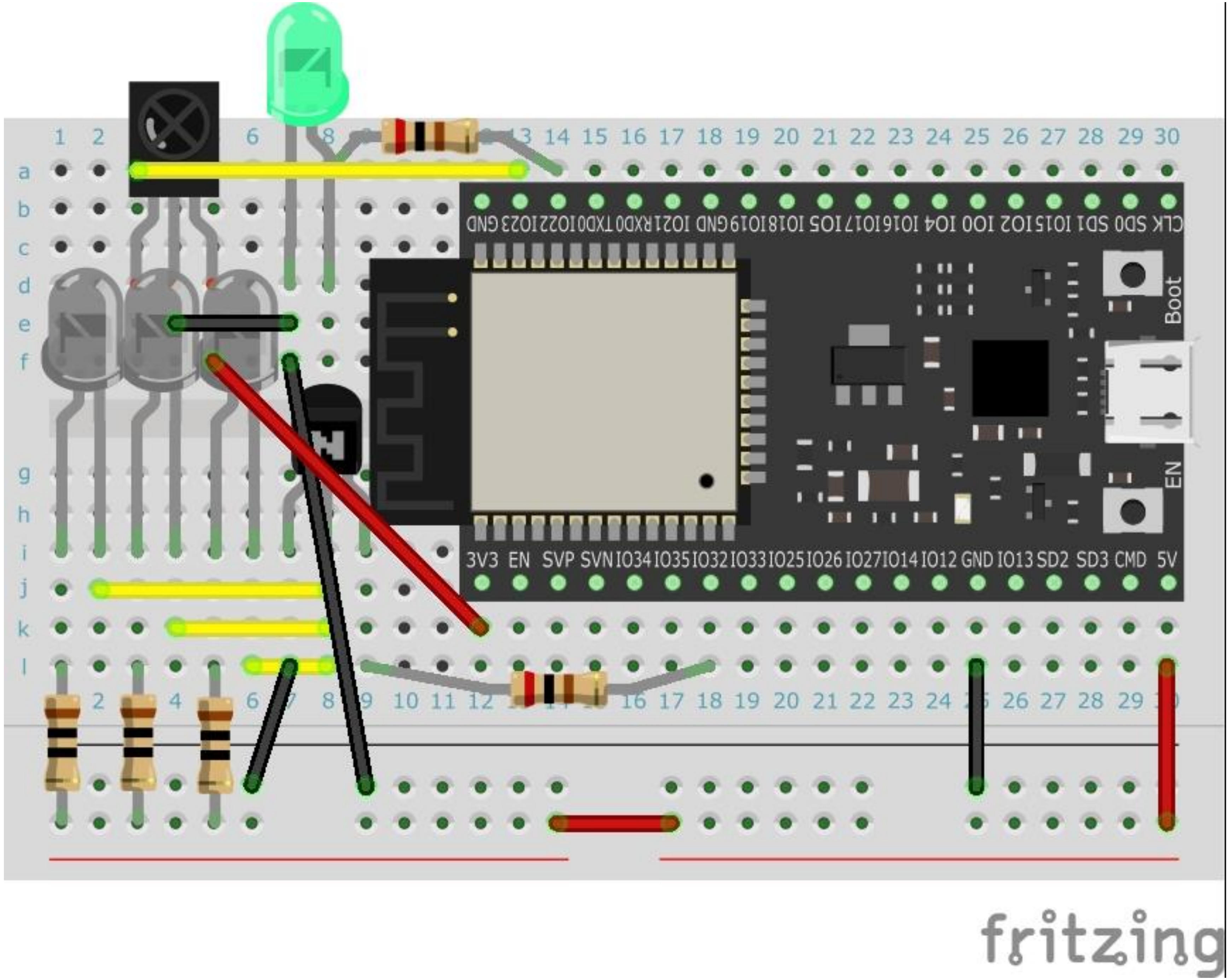
3. Circuit diagram

By setting IO23 of ESP32 to OUTPUT and passing current through the base of the transistor, the large current flowing between the collector and emitter is controlled to turn ON/OFF the infrared transmission LED.



4. Wiring diagram

Note) The LED and infrared reception sensor have already been wired in the previous video, so we will wire them additionally.



5-1. Software

Note) Since we have learned the basics in the LED section, we will understand the software part related to the infrared transmission LED.

IrSend | Arduino 1.8.19

File Edit Sketch Tools Help

IrSend §

```
//**************************************************************************
//  Ir Send Ver2023.1.22
//  Arduino board : ESP32(Arduino core for the ESP32) by Espressif Systems ver 2.0.6
//  Written by IT-Taro
//************************************************************************

const byte IR_S_PIN = 32;       // Transmission control with GPIO32
// Set transmission data
unsigned short irData[] = { 303,171,...(omission)...,43,44 };  // ### [Data rewrite required] ###

// Initial settings at startup
void setup() {
  Serial.begin(115200);
  Serial.println();
  pinMode(IR_S_PIN, OUTPUT);

  // Execute function to transmit infrared
  irSend ();
  Serial.println("SndOK");
}

void loop() {

}

// Infrared transmission processing
void irSend () {
  // Local variable definition
  unsigned short irCount = 0; // Number of HIGH and LOW signals
  unsigned long l_now = 0;    // Hold transmission start time
  unsigned long sndt = 0;     // Elapsed time from transmission start
  // Calculate the number of HIGH and LOW signals
  irCount = sizeof(irData) / sizeof(irData[0]);
```

Set here the remote control signal acquired by the infrared receiving sensor during electronic work.

Setup function
Starting the serial monitor and IO32 output settings
Execution of the infrared transmission function “irSend”

Loop function
No processing

35 ESP32 Dev Module, FTDI Adapter, Disabled, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, Core 1, Core 1, None, Disabled on COM3

5-2. Software

IrSend | Arduino 1.8.19

File Edit Sketch Tools Help

IrSend §

```
// Infrared transmission processing
void irSend () {
  // Local variable definition
  unsigned short irCount = 0; // Number of HIGH and LOW signals
  unsigned long l_now = 0;    // Hold transmission start time
  unsigned long sndt = 0;     // Elapsed time from transmission start
  // Calculate the number of HIGH and LOW signals
  irCount = sizeof(irData) / sizeof(irData[0]);
  // Acquire transmission start time
  l_now = micros();
  // Loop for the number of signals 0 and 1 with a For statement
  for (int i = 0; i < irCount; i++) {
    // Calculate signal end time from transmi
    sndt += irData[i];
    do {
      // If "i" is an even number, the inf
      // Transmitting with ON time at carr
      digitalWrite(IR_S_PIN, !(i&1));
      microWait(13);
      // Transmit at carrier frequency 38
      digitalWrite(IR_S_PIN, 0);
      microWait(13);
    // Loop from transmission start to sig
    } while (long(l_now + (sndt * 10) - micros()) > 0);
  }
}

// Wait in microseconds
void microWait(signed long waitTime) {
  unsigned long waitStartMicros = micros();
  // Loop processing (wait) with While until the specified microseconds have passed
  while (micros() - waitStartMicros < waitTime) {};
}
```

Declaration of variables to use

get duration

Transmit infrared signal when HIGH
(Transmission stops when LOW)
Continues for 13 μsec

Stop sending both HIGH and LOW
(for 38KHz modulation)
Continues for 13 μsec

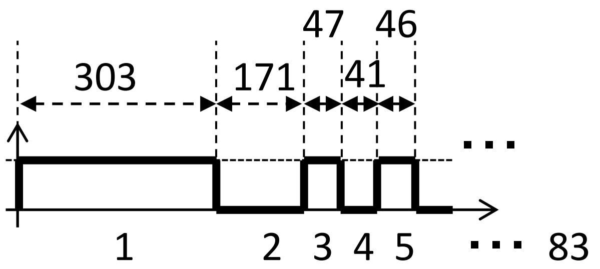
Do and while statements repeat for the duration of HIGH and LOW.

Repeat the For statement as many times as HIGH and LOW.

Wait processing in μsec units

52 ESP32 Dev Module, FTDI Adapter, Disabled, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600, Core 1, Core 1, None, Disabled on COM3

5-3. Software



- Data acquired by the infrared receiving sensor (time interval between HIGH and LOW)

unsigned short irData[] = { 303,171,47,41,46,41,46,128,46,128, ・ ・ ・ （omit） ・ ・ ・,44,43,44 };

1 2 3 4 5 6 7 8 9 10 ・・・ （omit） ・・・ 81, 82,83 ⇒ 83 HIGHs and LOWs

- number of HIGH and LOW

```
irCount = sizeof(irData) / sizeof(irData[0]);
```

Get numbers separated by commas. In other words, the number of HIGH and LOW ⇒ 83

- Infrared transmission LED control

```
digitalWrite(IR_S_PIN, !(i&1));
```

0: no infrared transmission
1: Infrared is transmitted

"i" is processed in order from 0 to (irCount-1).
“i&1” is a bitwise operation, and since it is an AND operation, even numbers are 0 and odd numbers are 1.

even number

binary of i	x	x	x	x	x	x	0	and ⇒	0
binary of 1	0	0	0	0	0	0	1		

odd number

binary of i	x	x	x	x	x	x	1	and ⇒	1
binary of 1	0	0	0	0	0	0	1		

“!(i&1)” inverts the above, so even numbers are 1 and odd numbers are 0.
Therefore, it will be sent when the number is even and not when the number is odd.