Embedded Systems

IR Receiver Prototype

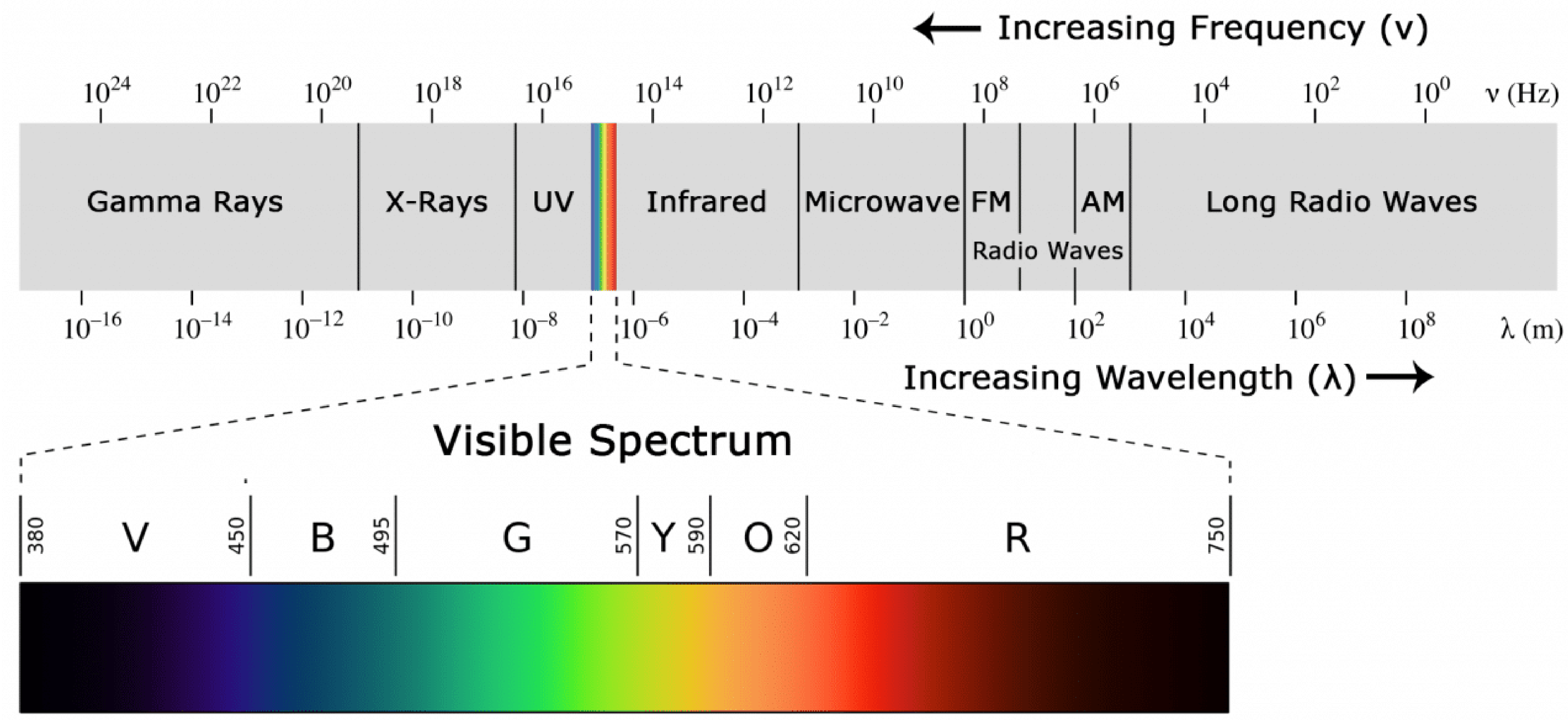
Embedded Team, BFCAI

Infrared (IR) Communication

- Infrared (IR) communication is a widely used and easy to implement wireless technology that has many useful applications.
- The most prominent example in day to day life are TV remote controls.

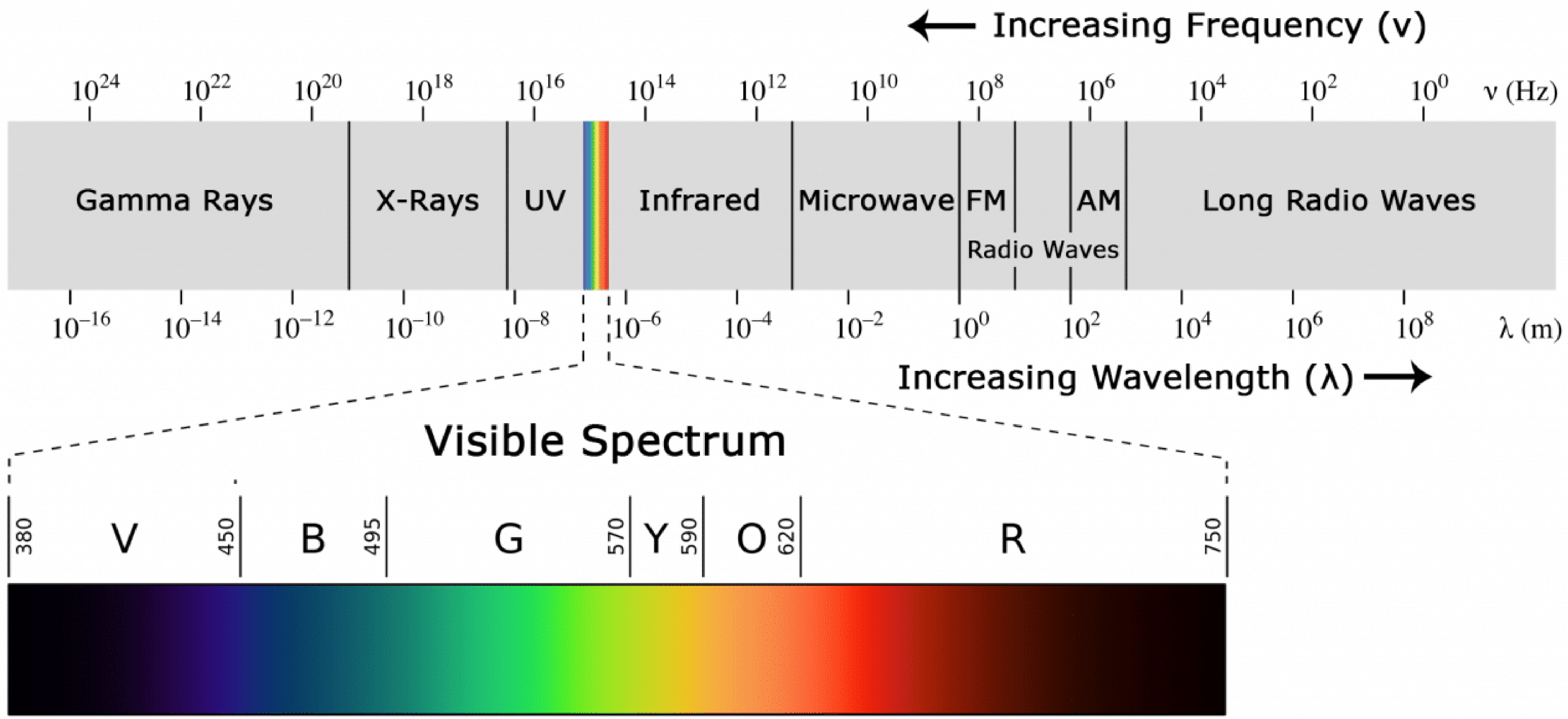
Infrared (IR) Communication

- Infrared is a form of light similar to the light we see all around us.



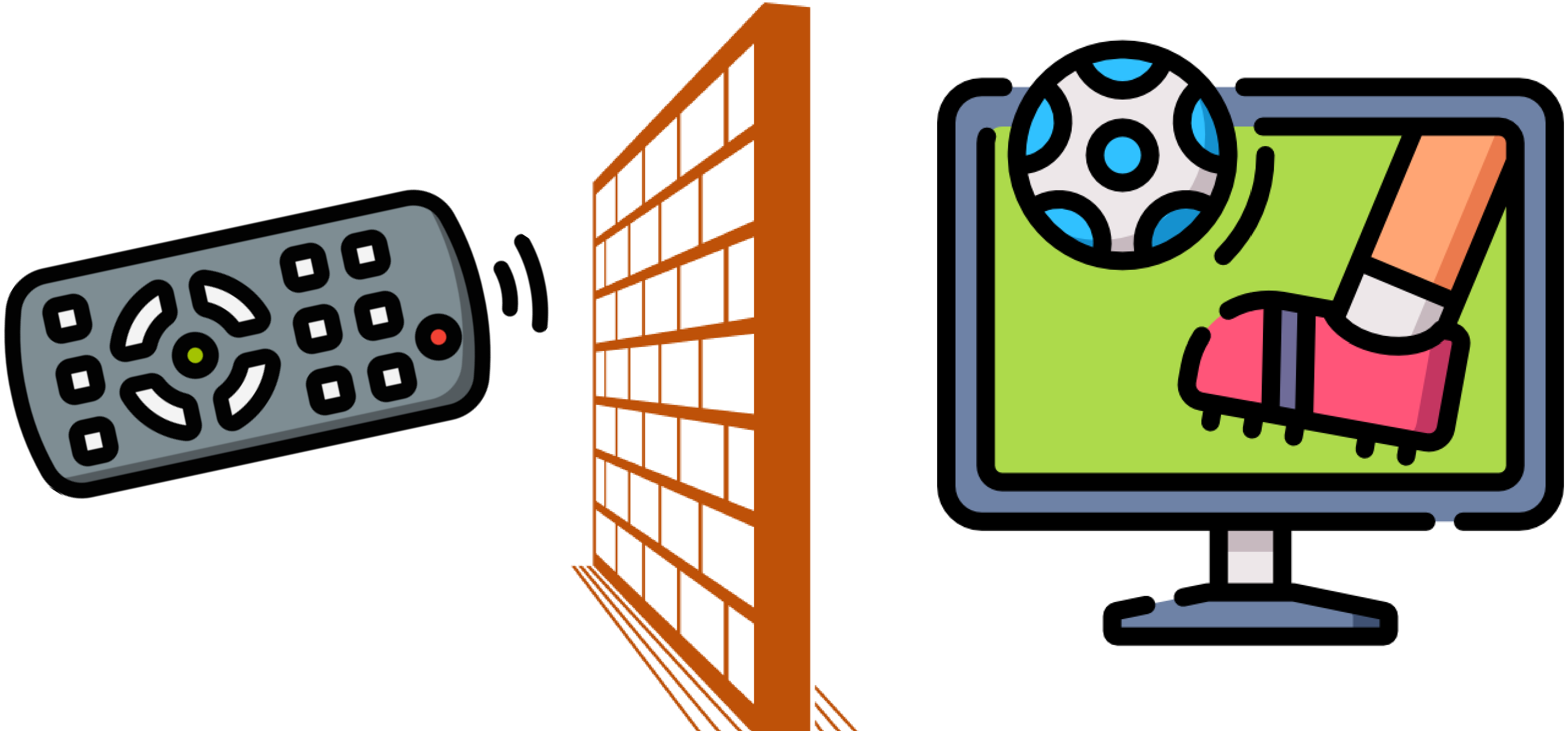
Infrared (IR) Communication

- The difference between IR and visible light is the frequency and wavelength.
- Infrared lies outside the range of visible light, so humans can't see it.



Infrared (IR) Communication

- Because IR is a type of light, IR communication requires a direct line of sight from the receiver to the transmitter.
- It can’t transmit through walls.



Infrared (IR) Communication System

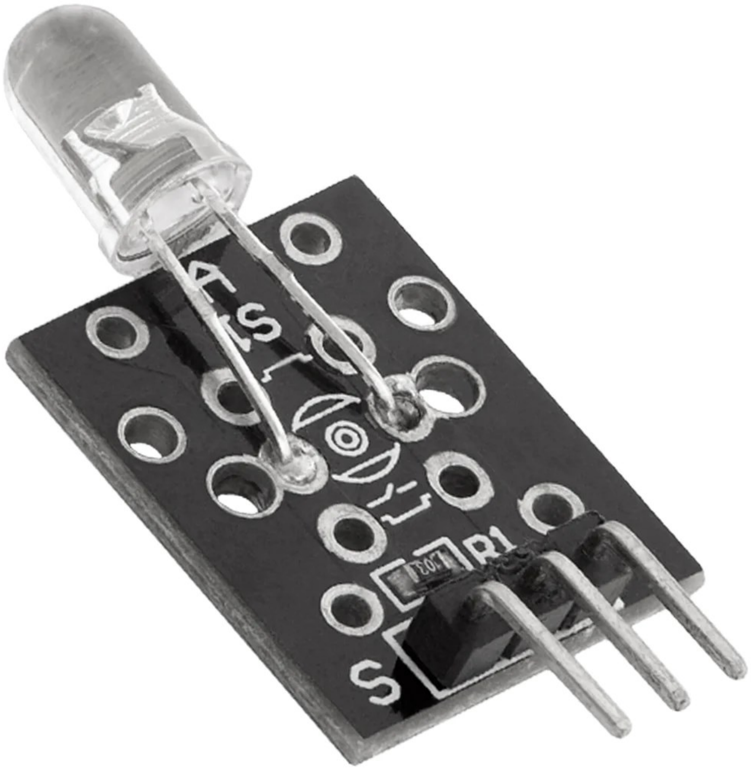
- A typical infrared communication system requires an IR transmitter and an IR receiver.

IR Transmitter

IR Receiver

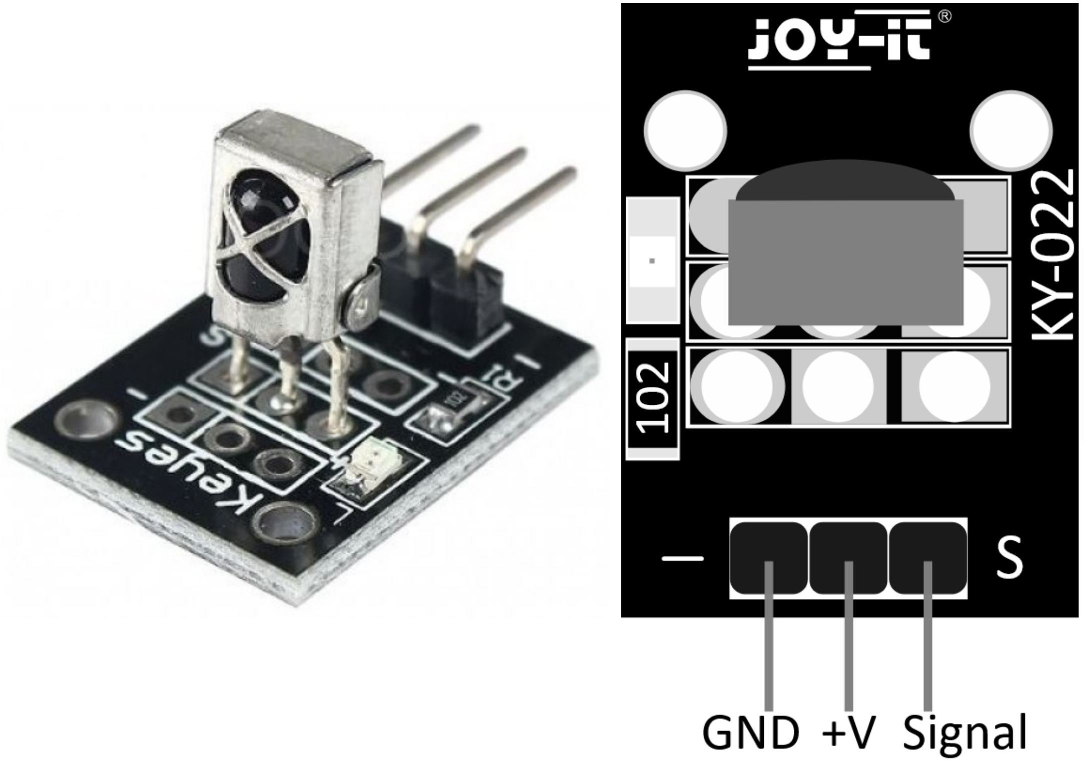
Infrared (IR) Communication System

- The IR transmitter looks just like a standard LED, except it produces light in the IR spectrum instead of the visible spectrum.

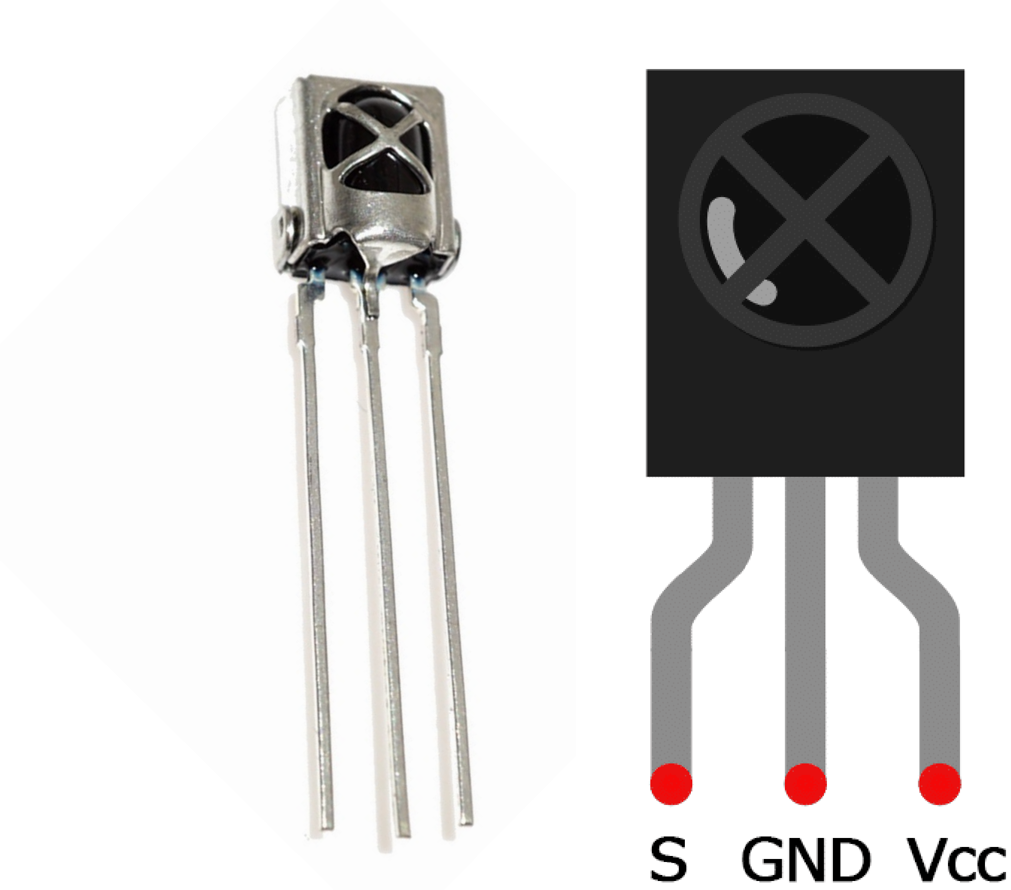


Infrared (IR) Communication System

- The IR receiver converts the IR light into an electrical signal.

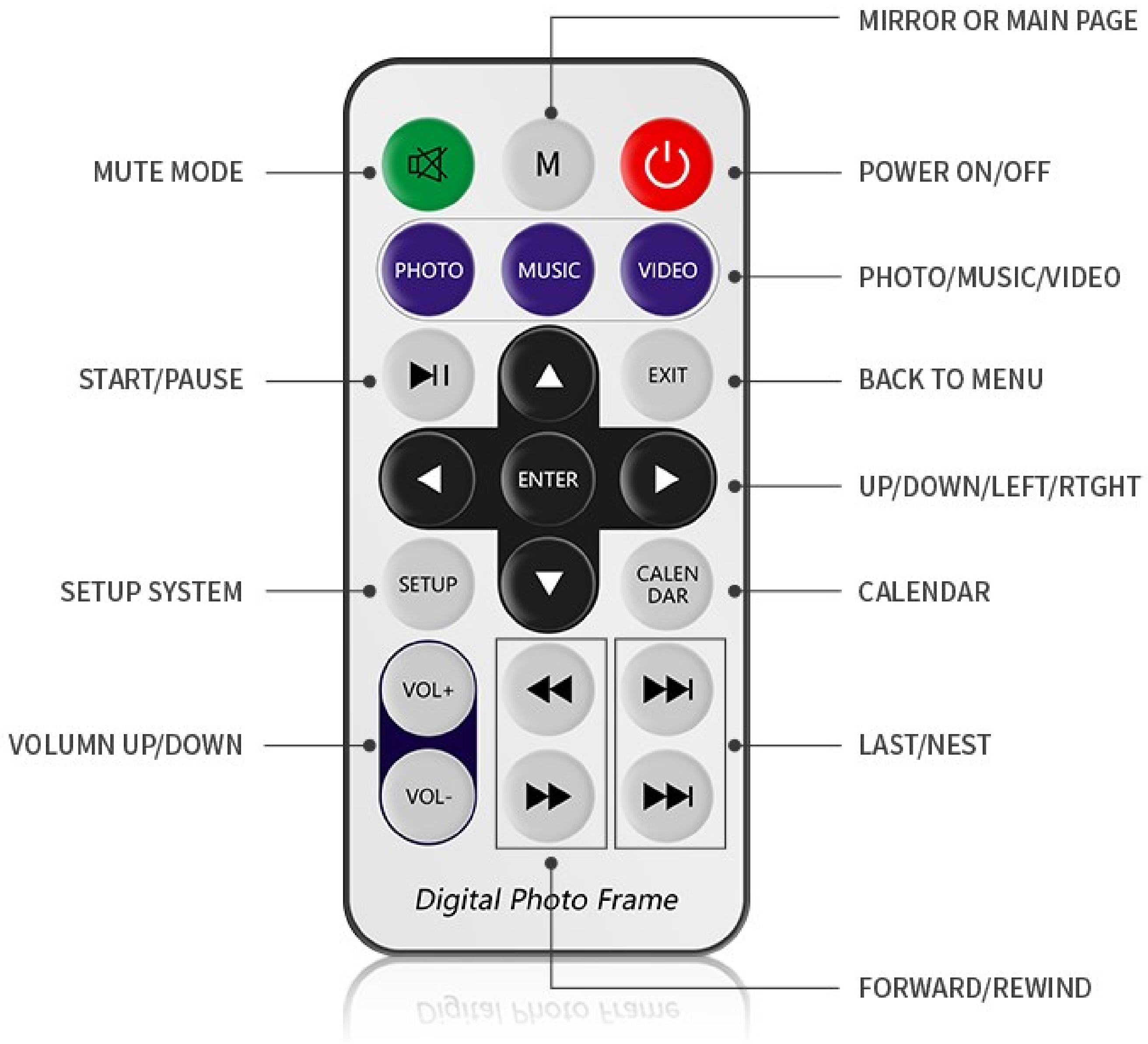


IR Receiver Module



IR Receiver

IR Remote Control



IR Remote Control: Commands

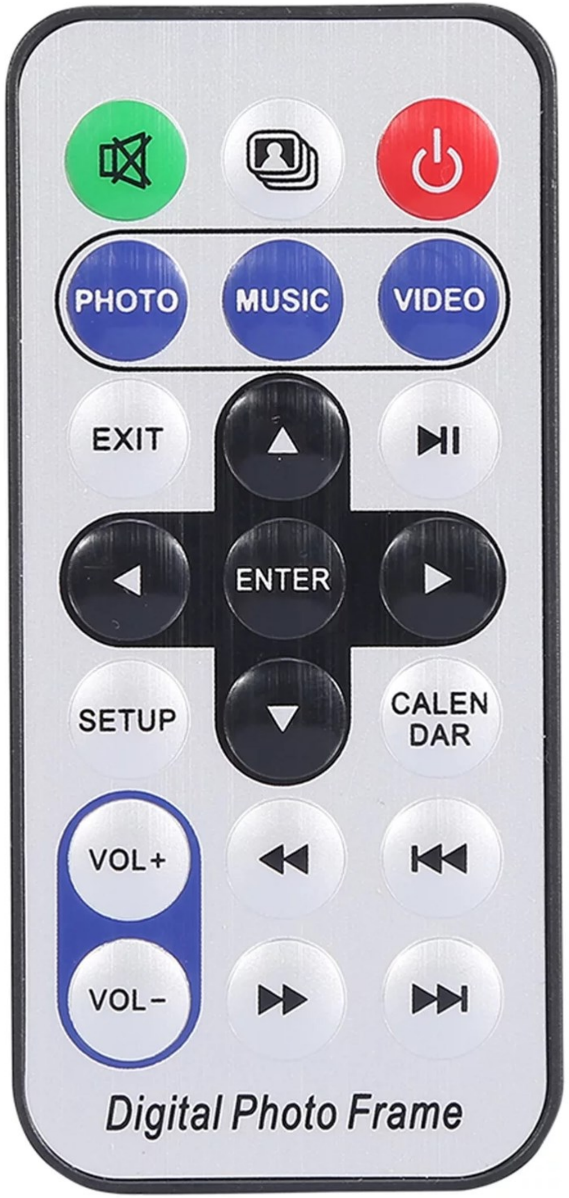
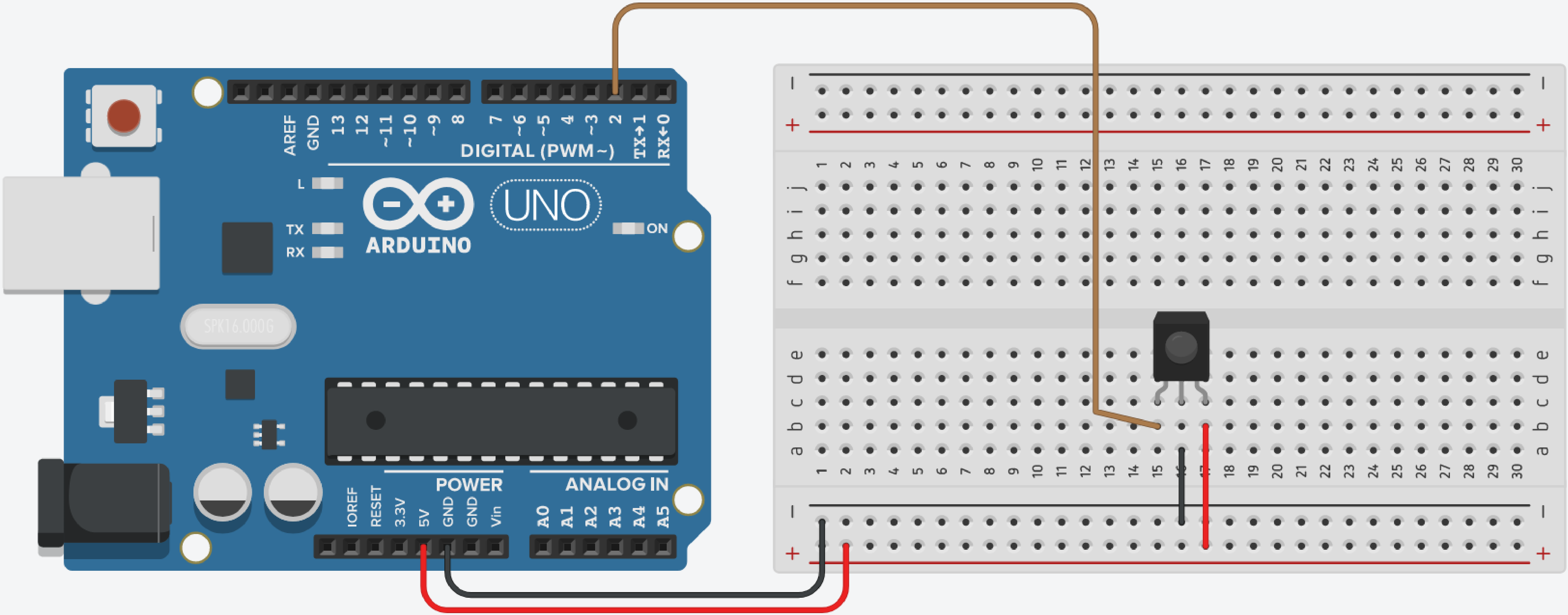


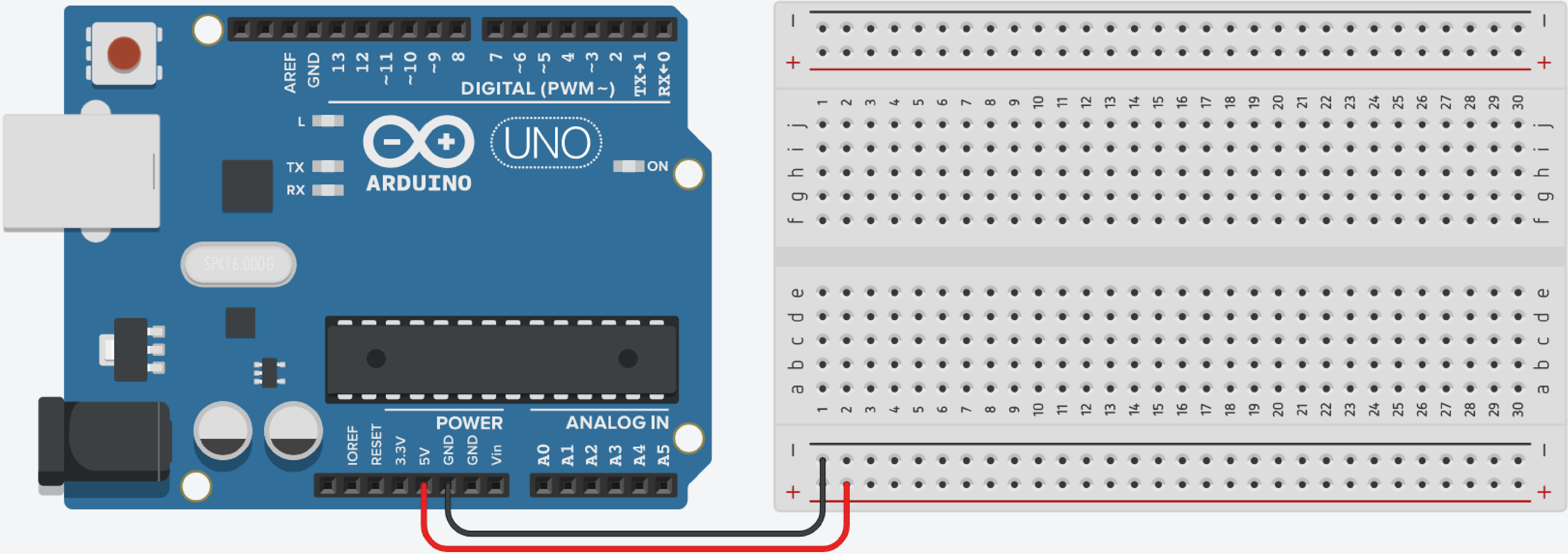
PHOTO	MUSIC	VIDEO	EXIT
0x1	0x2	0x3	0x4

IR Receiver Demo: Circuit



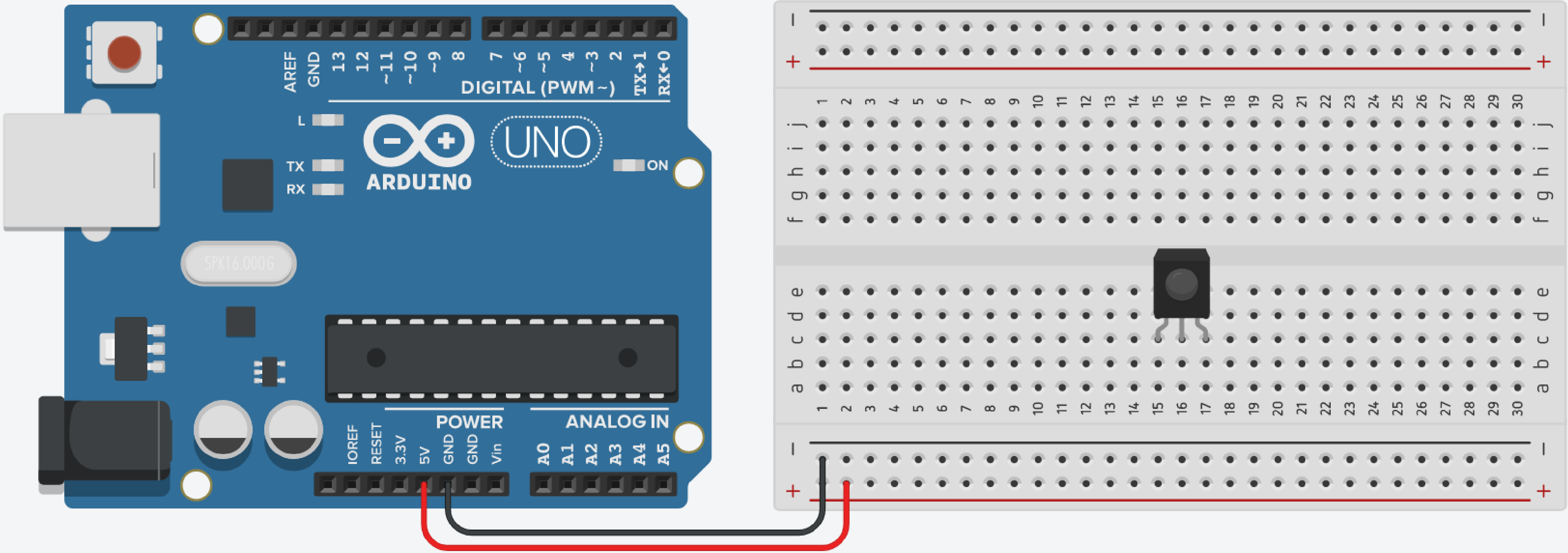
IR Receiver Demo: Steps

1. Connect breadboard power (+) and ground (-) rails to Arduino 5V and ground (GND), respectively.



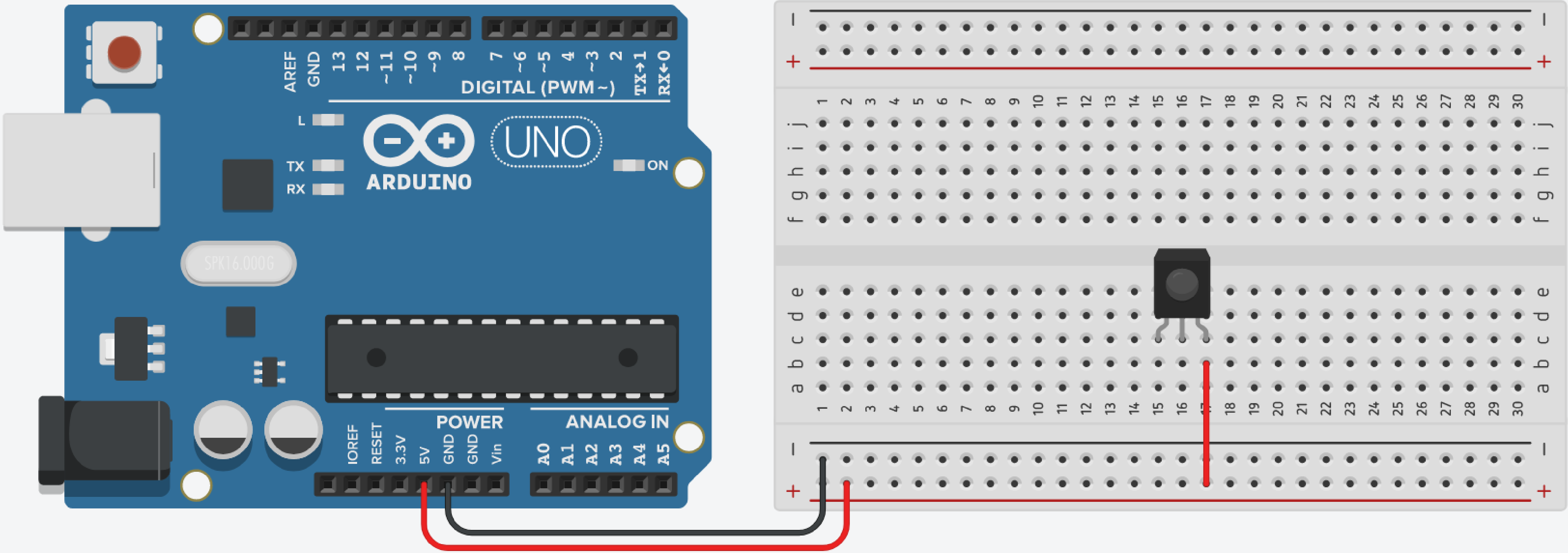
IR Receiver Demo: Steps

2. Plug the IR Receiver into the breadboard.



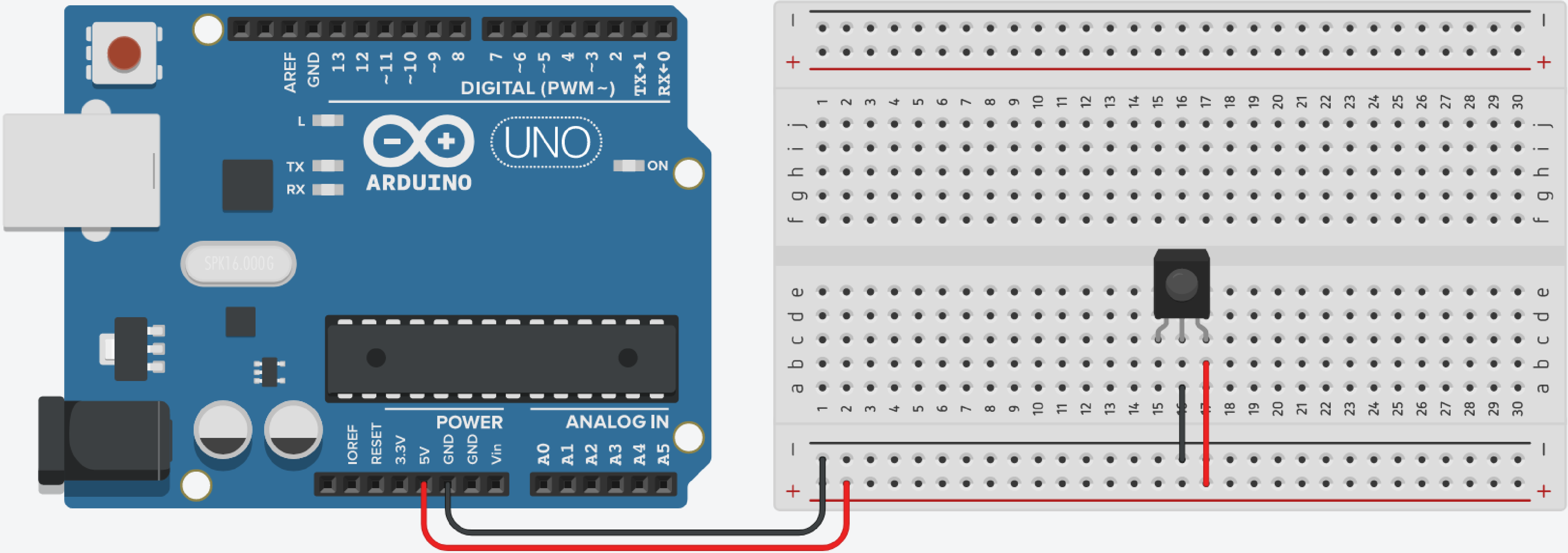
IR Receiver Demo: Steps

3. The VCC pin of the IR Receiver connects to the power.



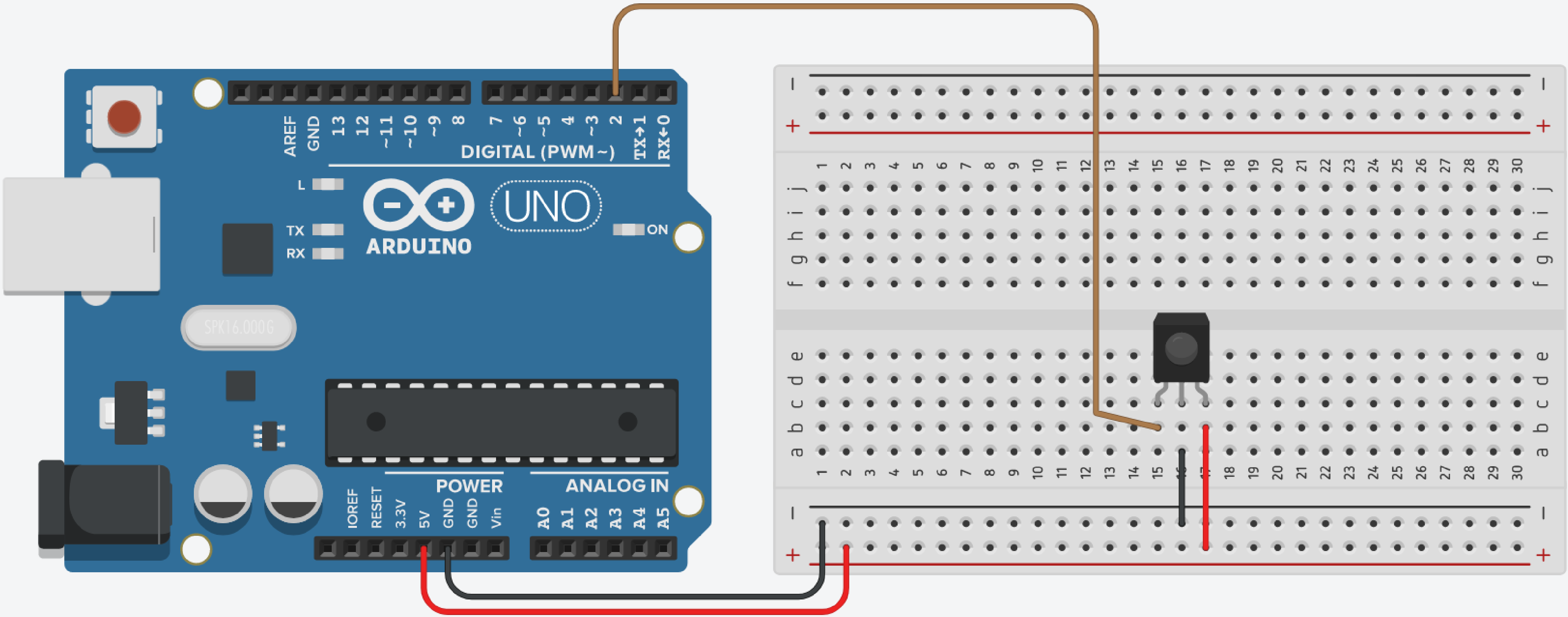
IR Receiver Demo: Steps

4. The GND pin (-) of the IR Receiver connects to the ground.



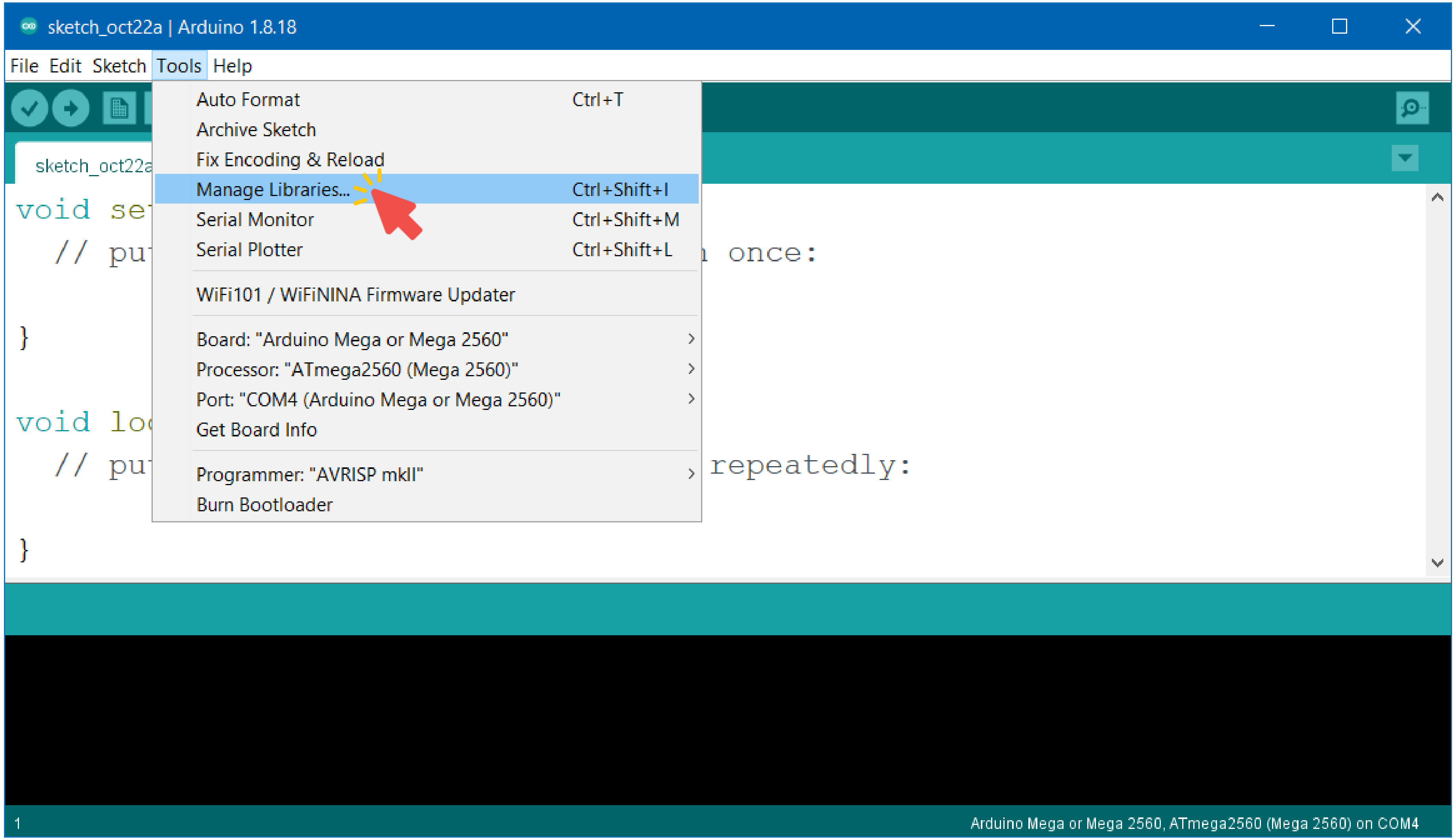
IR Receiver Demo: Steps

5. The signal pin of the IR Receiver connects to pin 2 on Arduino.



IR Receiver Demo: IRremote Library

- To install a library, go to Tools → Manage Libraries.



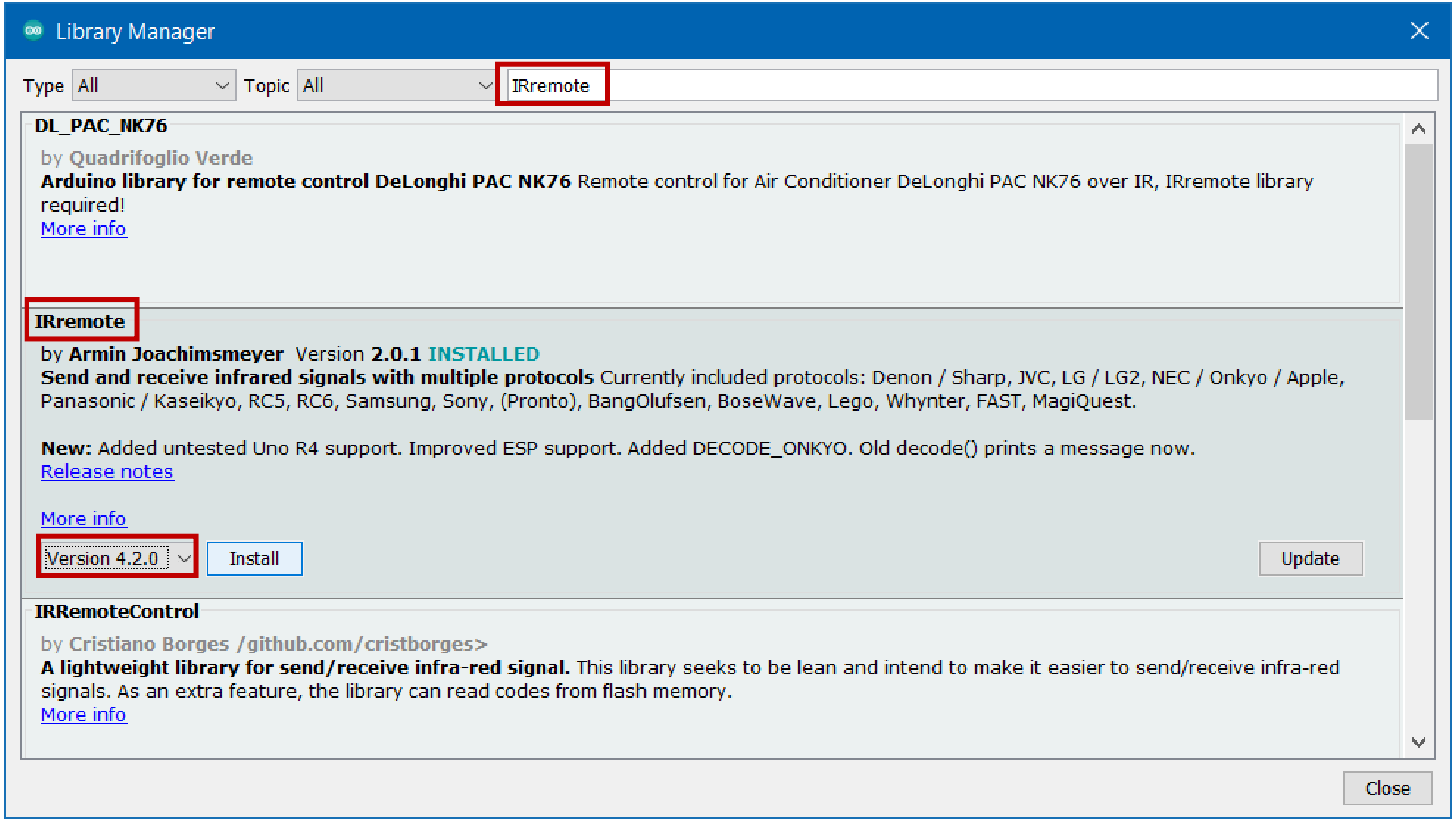
sketch_oct22a | Arduino 1.8.18
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Mega or Mega 2560"
Processor: "ATmega2560 (Mega 2560)"
Port: "COM4 (Arduino Mega or Mega 2560)"
Get Board Info
Programmer: "AVRISP mkII"
Burn Bootloader
sketch_oct22a

```
void se
  // pu                            n once:

}

void lo
  // pu                            repeatedly:

}
```

1
Arduino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM4

IR Receiver Demo: IRremote Library

- Search for "IRremote", and select Version 4.2.0.



Type All Topic All IRremote

DL_PAC_NK76

by Quadrifoglio Verde

Arduino library for remote control DeLonghi PAC NK76 Remote control for Air Conditioner DeLonghi PAC NK76 over IR, IRremote library required!

More info

IRremote

by **Armin Joachimsmeyer** Version **2.0.1** **INSTALLED**

Send and receive infrared signals with multiple protocols Currently included protocols: Denon / Sharp, JVC, LG / LG2, NEC / Onkyo / Apple, Panasonic / Kaseikyo, RC5, RC6, Samsung, Sony, (Pronto), BangOlufsen, BoseWave, Lego, Whynter, FAST, MagiQuest.

New: Added untested Uno R4 support. Improved ESP support. Added DECODE_ONKYO. Old decode() prints a message now.

Release notes

More info

Version 4.2.0

IRRemoteControl

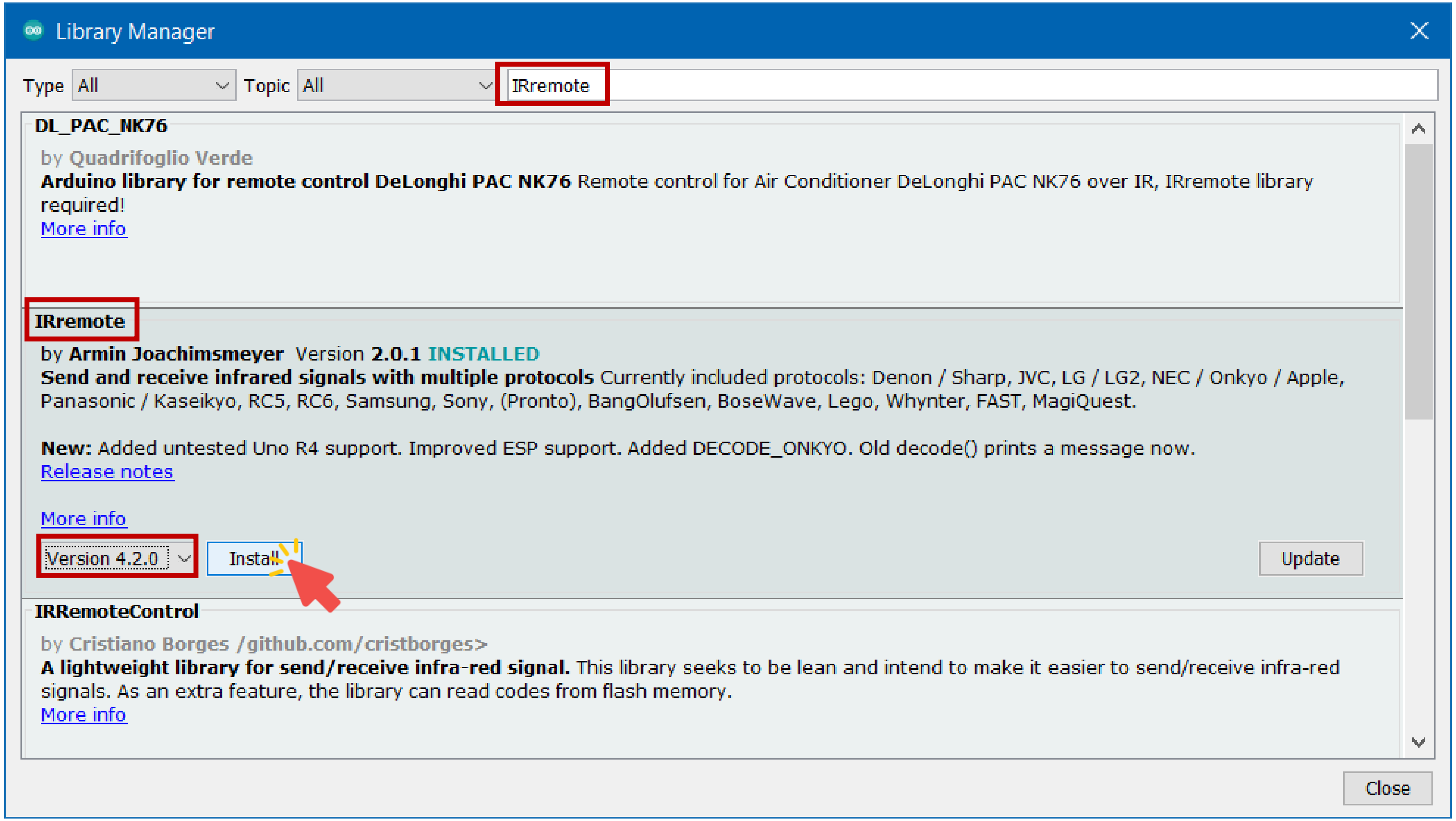
by **Cristiano Borges /github.com/cristborges>**

A lightweight library for send/receive infra-red signal. This library seeks to be lean and intend to make it easier to send/receive infra-red signals. As an extra feature, the library can read codes from flash memory.

More info

IR Receiver Demo: IRremote Library

- Click Install.



DL_PAC_NK76

by Quadrifoglio Verde

Arduino library for remote control DeLonghi PAC NK76 Remote control for Air Conditioner DeLonghi PAC NK76 over IR, IRremote library required!

More info

IRremote

by **Armin Joachimsmeyer** Version **2.0.1** INSTALLED

Send and receive infrared signals with multiple protocols Currently included protocols: Denon / Sharp, JVC, LG / LG2, NEC / Onkyo / Apple, Panasonic / Kaseikyo, RC5, RC6, Samsung, Sony, (Pronto), BangOlufsen, BoseWave, Lego, Whynter, FAST, MagiQuest.

New: Added untested Uno R4 support. Improved ESP support. Added DECODE_ONKYO. Old decode() prints a message now.

Release notes

More info

IRRemoteControl

by Cristiano Borges /github.com/cristborges>

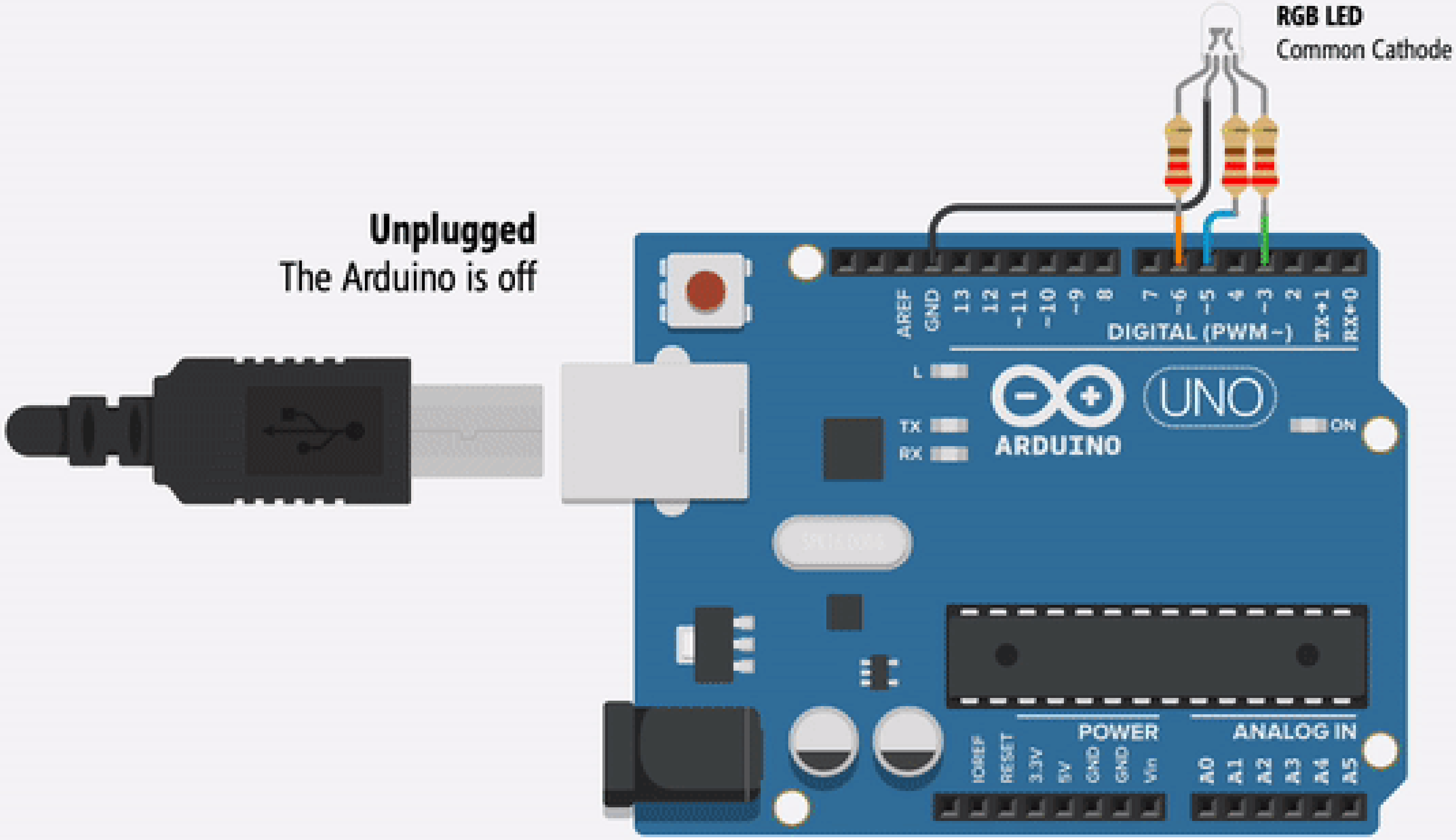
A lightweight library for send/receive infra-red signal. This library seeks to be lean and intend to make it easier to send/receive infra-red signals. As an extra feature, the library can read codes from flash memory.

More info

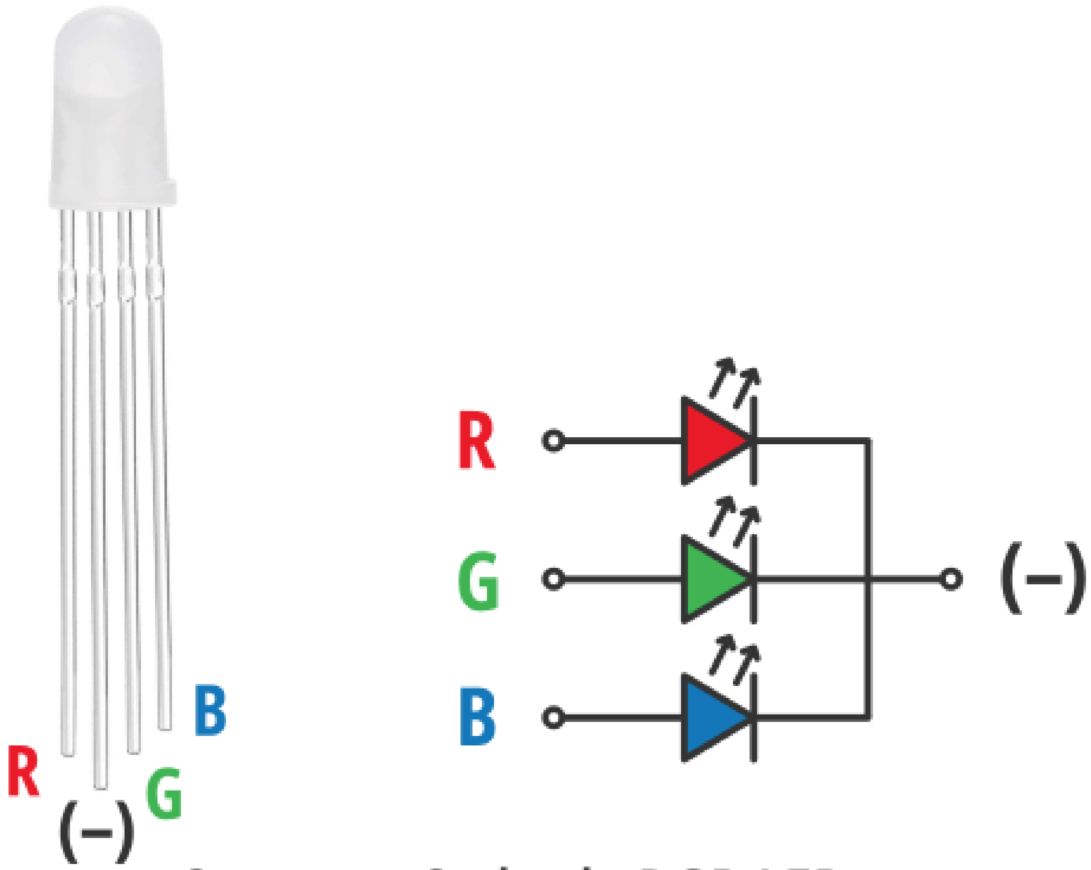
IR Receiver Demo: Code

```
#include <IRremote.h>                              // Import IRremote library
#define RECV_PIN 2                                 // Receiver pin
unsigned int command;                              // Variable to store the infrared command

void setup()
{
  Serial.begin(9600);                              // Begin serial communication at 9600 baud rate
  IrReceiver.begin(RECV_PIN);                      // Start the receiver
}

void loop() {
  if (IrReceiver.decode()) {                       // If a button is pressed
    command = IrReceiver.decodedIRData.command;    // Get the infrared command
    Serial.println(command, HEX);                  // Print button command in hexadecimal
    IrReceiver.resume();                           // Receive the next value
  }

  delay(50);                                       // Short delay to improve performance
}
```

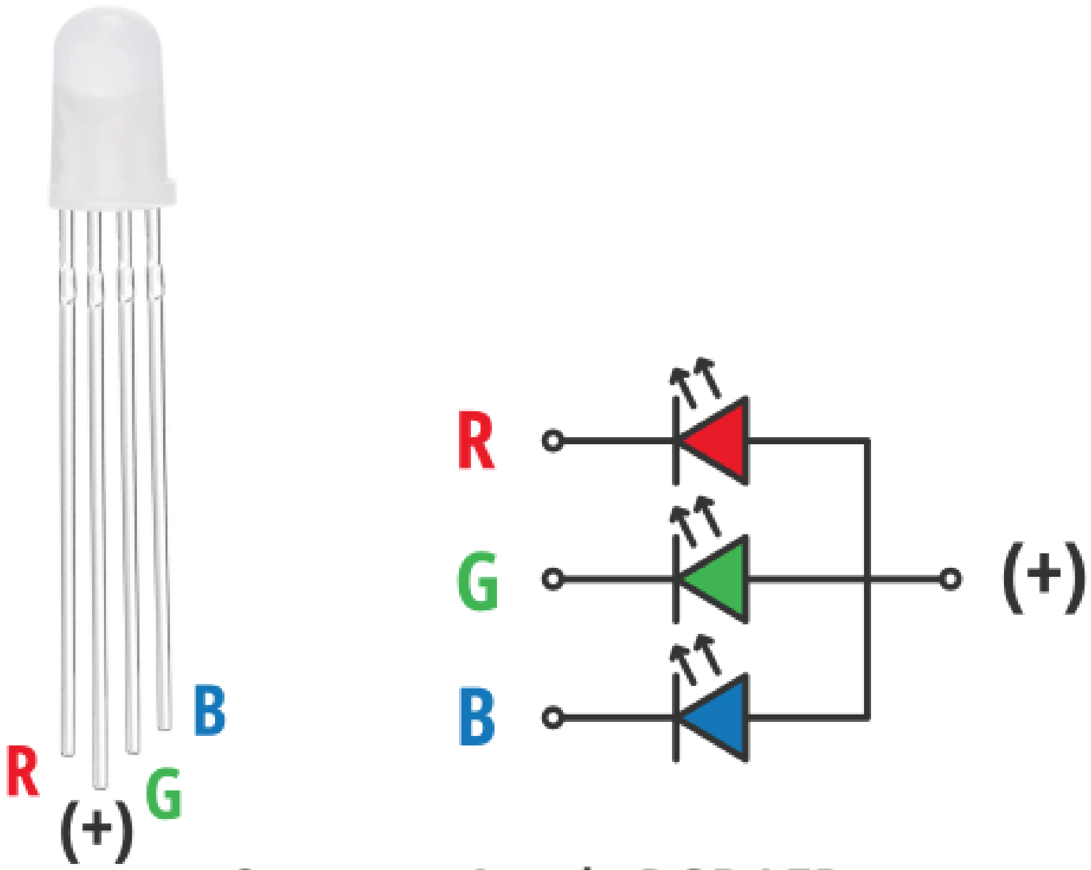
RGB LEDs


RGB LED
Common Cathode
Unplugged
The Arduino is off
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
ARDUINO
UNO
ON
IOREF
RESET
3.3V
5V
GND
GND
Vin
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5

RGB LEDs

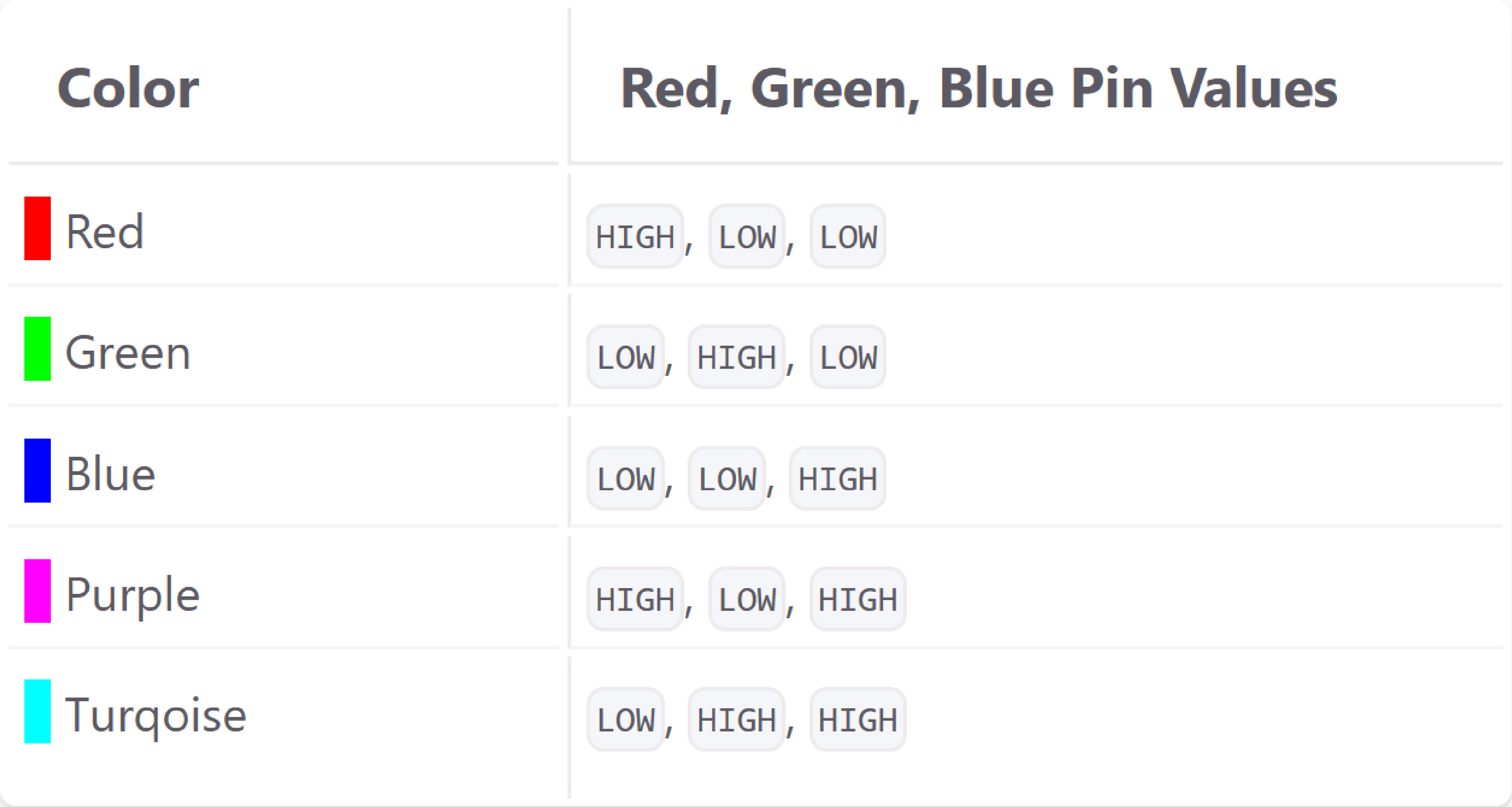


Common Cathode RGB LED



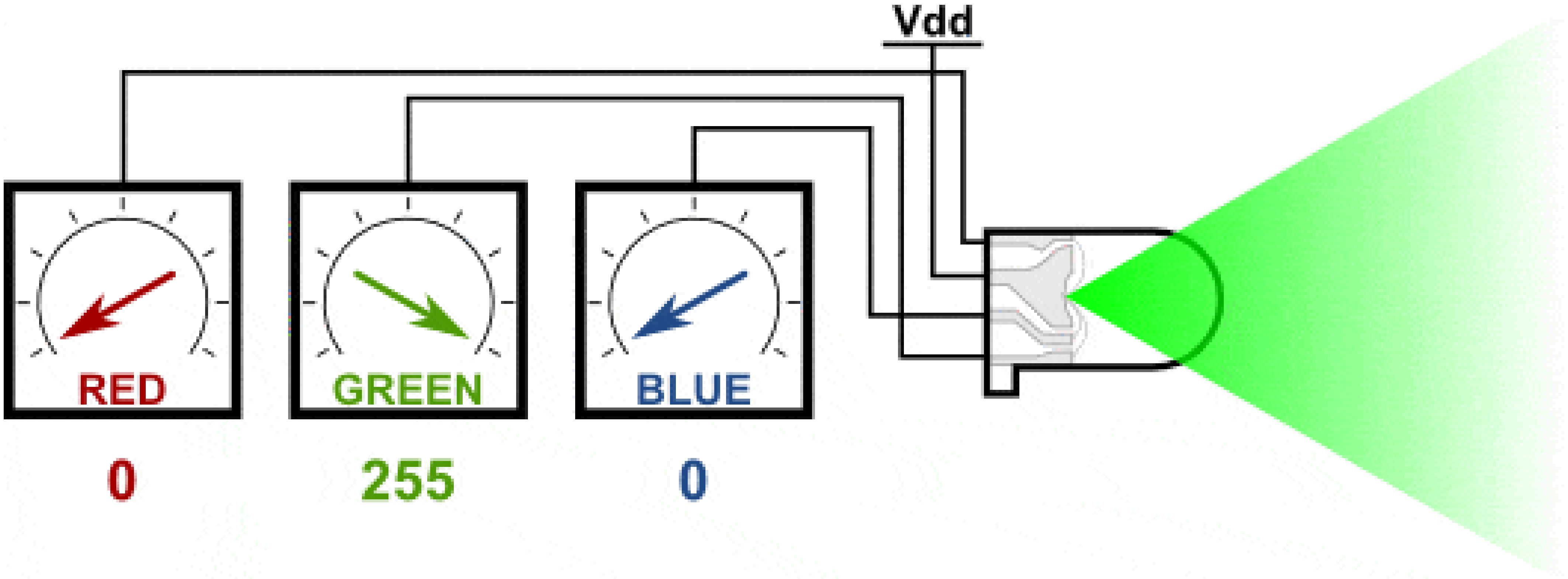
Common Anode RGB LED

RGB LEDs



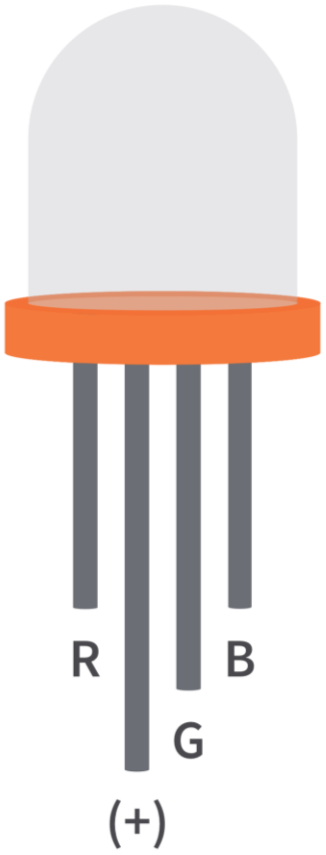
Color	Red, Green, Blue Pin Values
🟥 Red	`HIGH`, `LOW`, `LOW`
🟩 Green	`LOW`, `HIGH`, `LOW`
🟦 Blue	`LOW`, `LOW`, `HIGH`
🟪 Purple	`HIGH`, `LOW`, `HIGH`
🟦 Turqoise	`LOW`, `HIGH`, `HIGH`

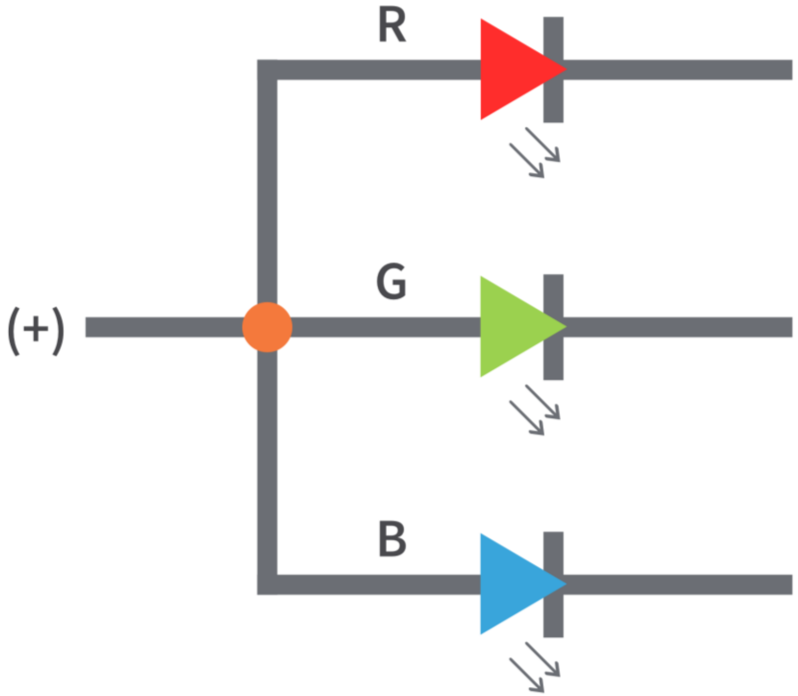
RGB LEDs



Common Anode RGB LED

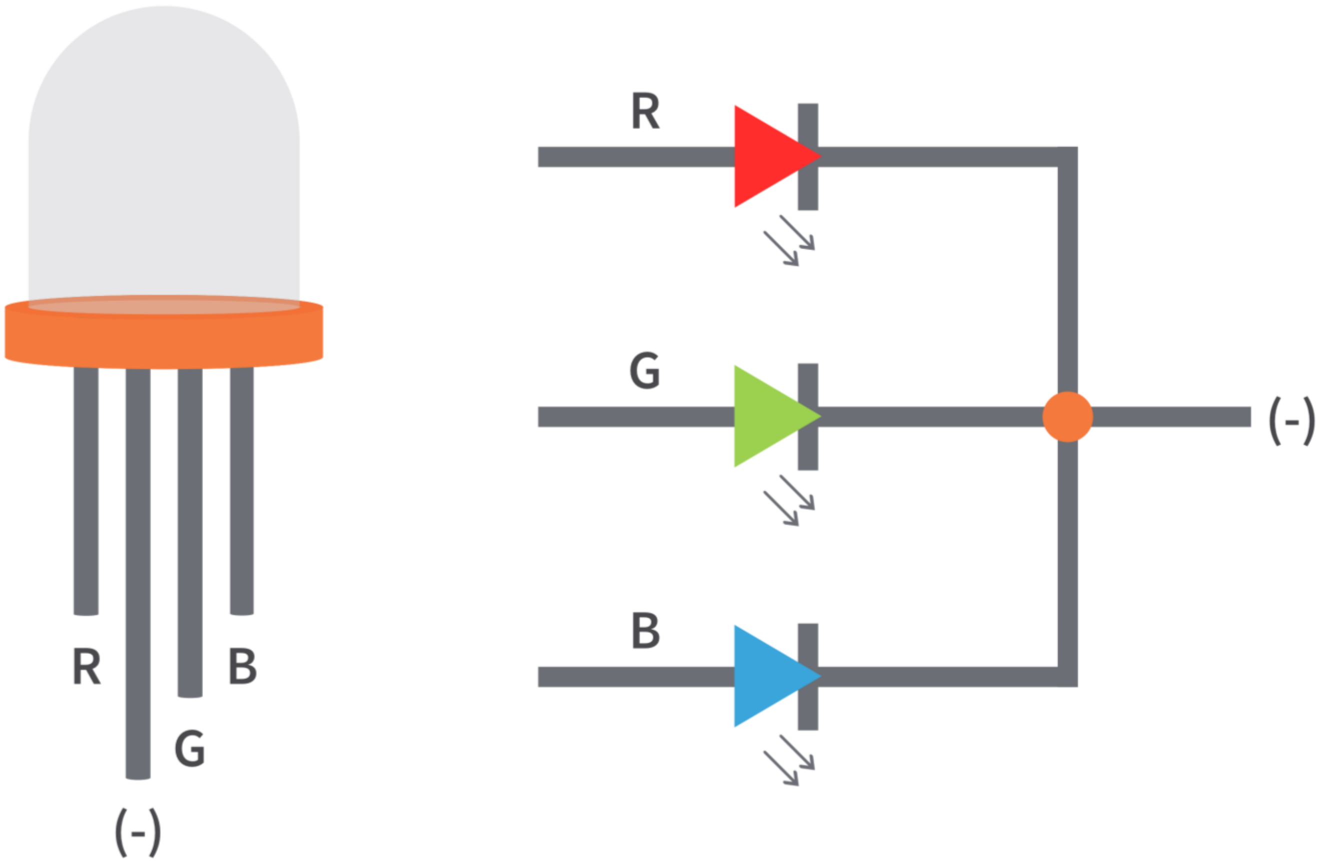
- To control each color, you need to apply a LOW signal to the leads and connect the common to the VCC.





Common Cathode RGB LED

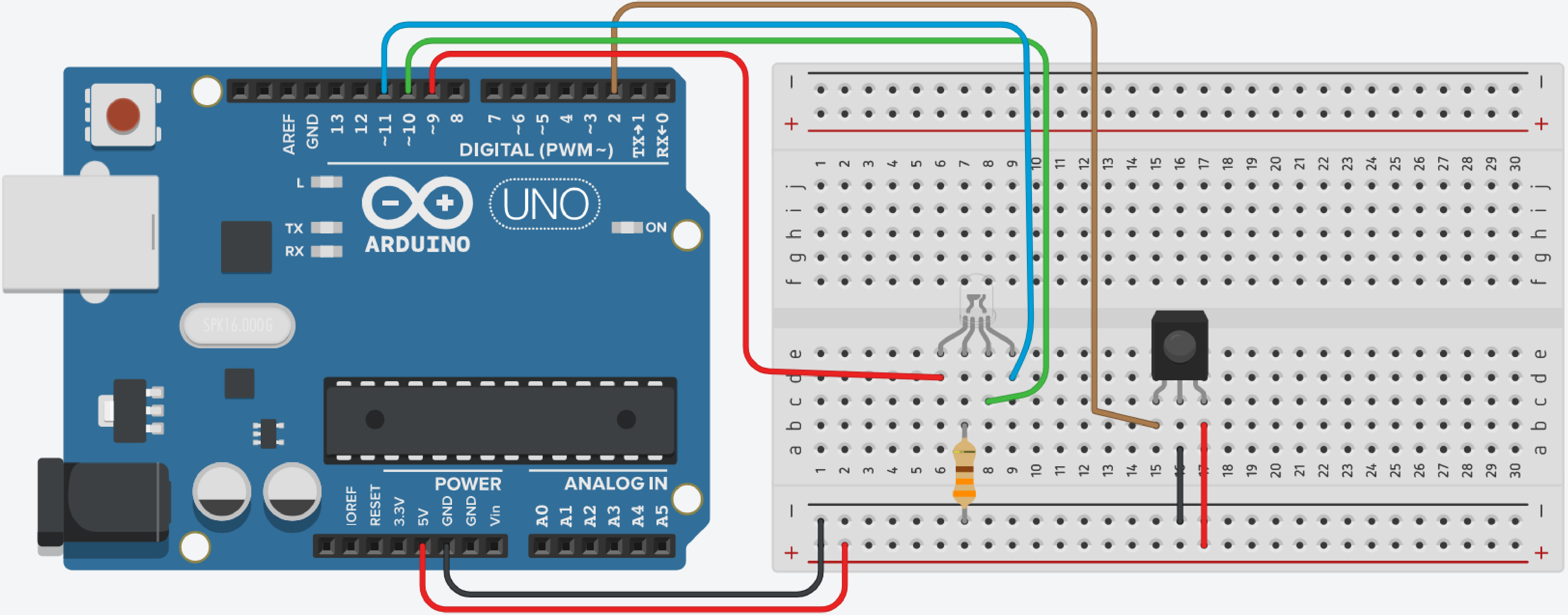
- To control each color, you need to apply a HIGH signal to the leads and connect the common to the GND.



IR Remote Control: Commands

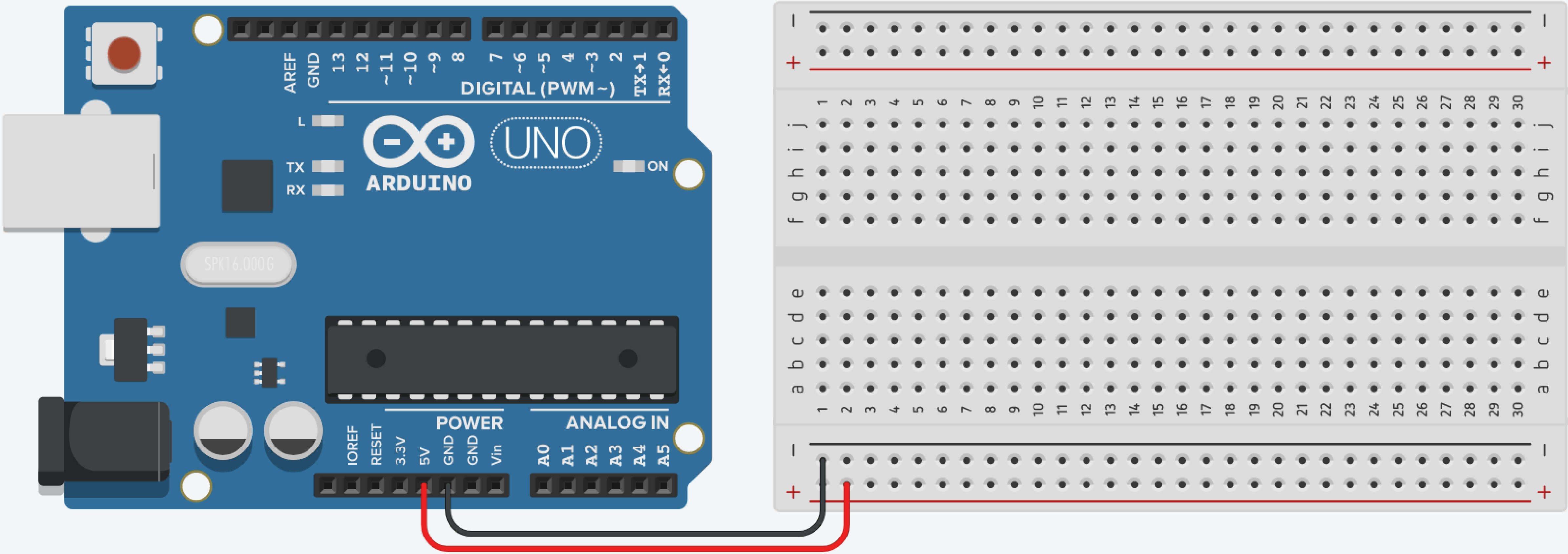
PHOTO	MUSIC	VIDEO	EXIT
0x1	0x2	0x3	0x4

IR-Controlled RGB LED: Circuit



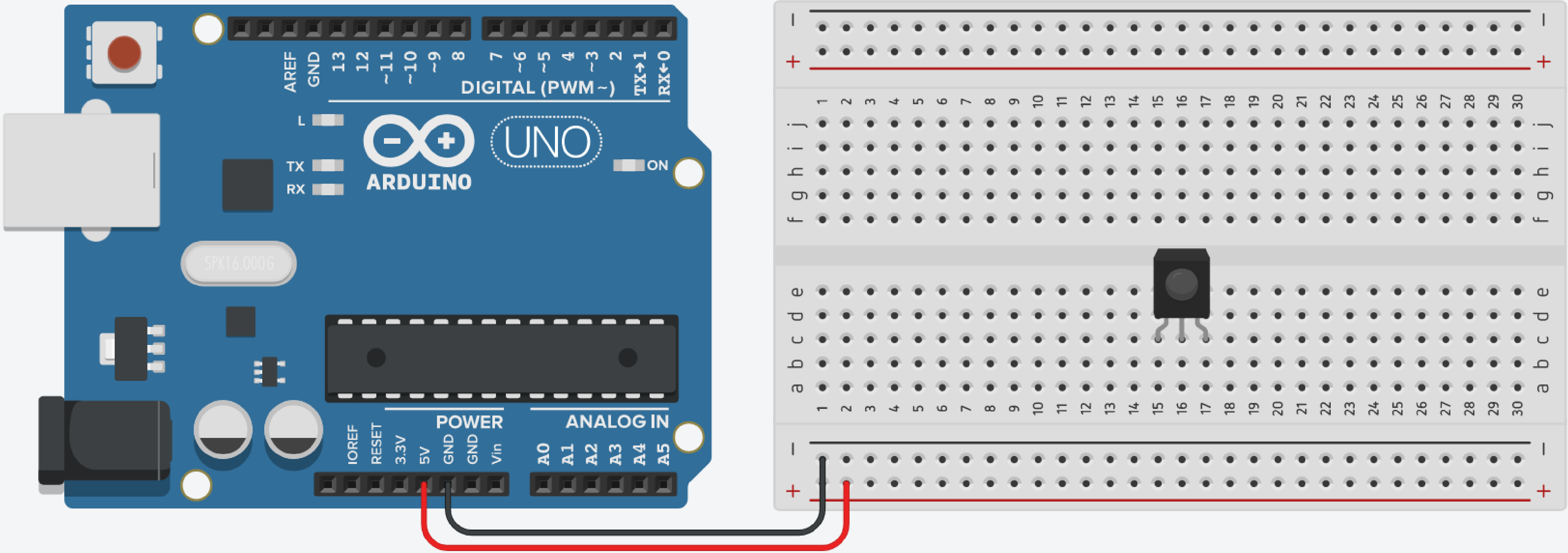
IR Receiver Demo: Steps

1. Connect breadboard power (+) and ground (-) rails to Arduino 5V and ground (GND), respectively.



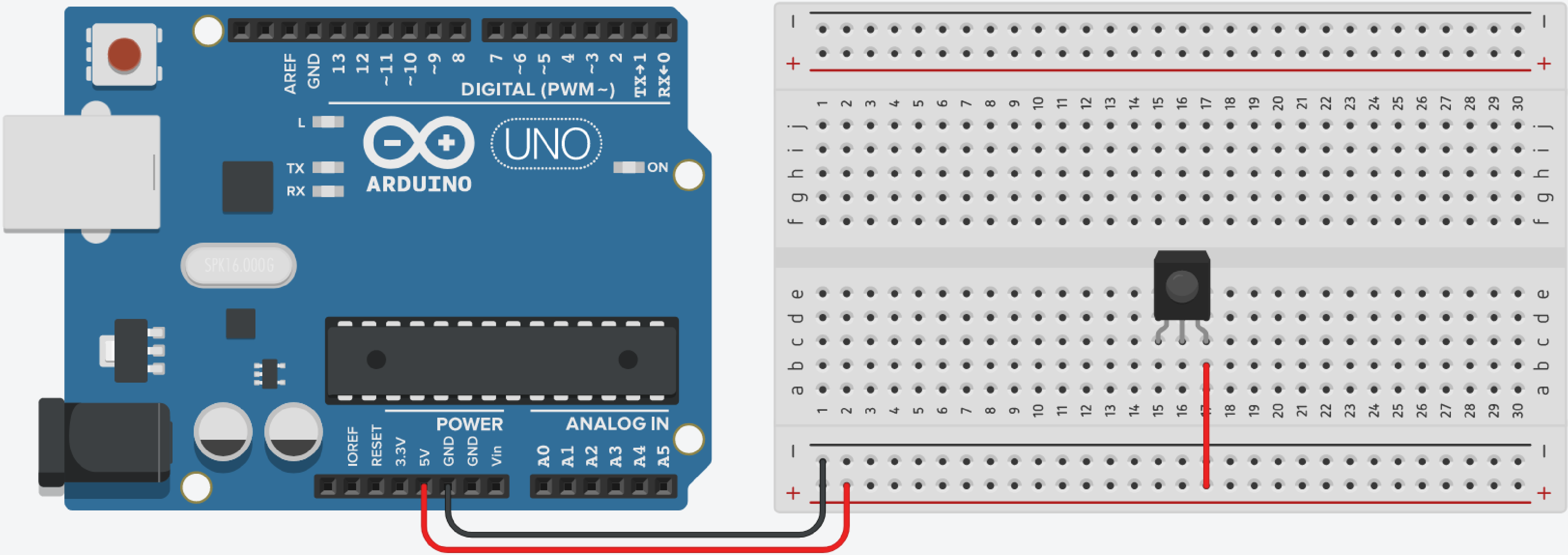
IR Receiver Demo: Steps

2. Plug the IR Receiver into the breadboard.



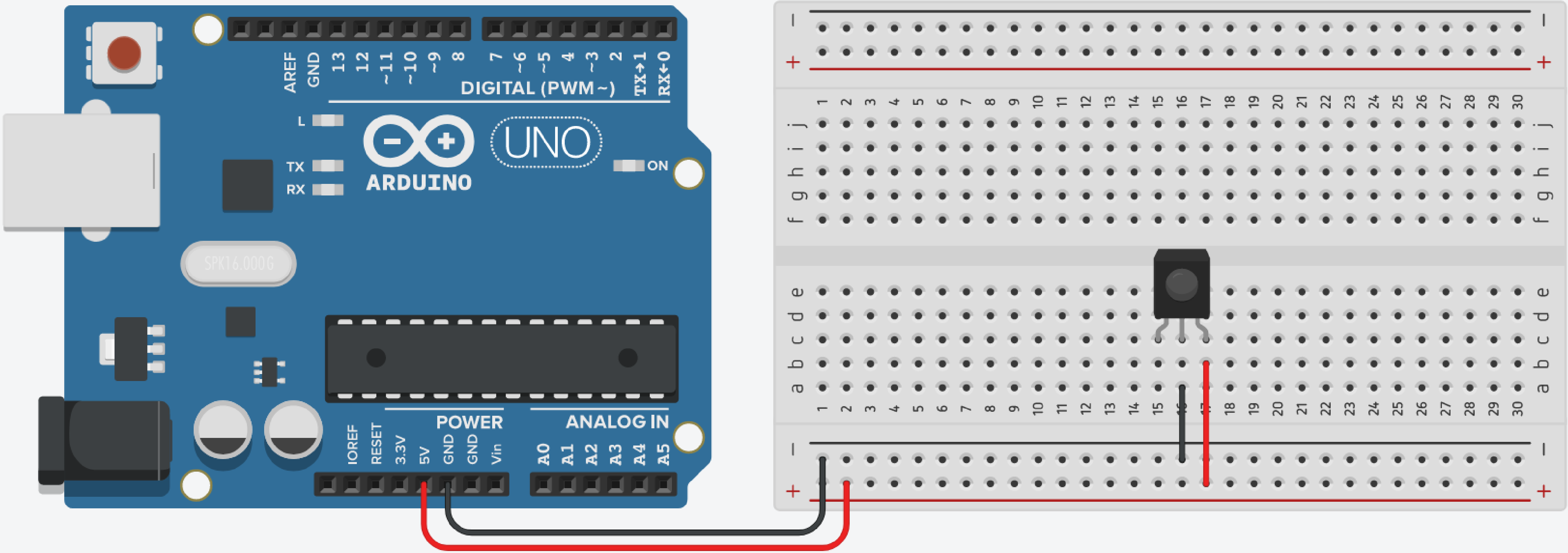
IR Receiver Demo: Steps

3. The VCC pin of the IR Receiver connects to the power.



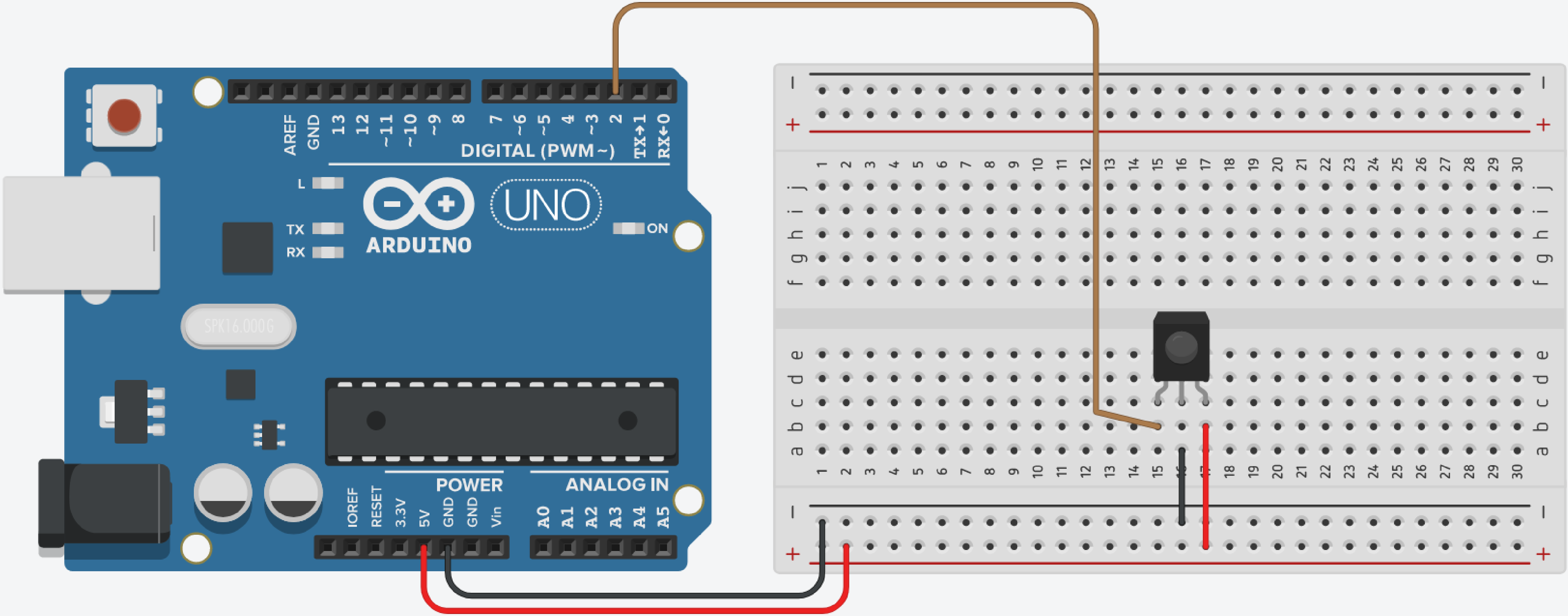
IR Receiver Demo: Steps

4. The GND pin (-) of the IR Receiver connects to the ground.



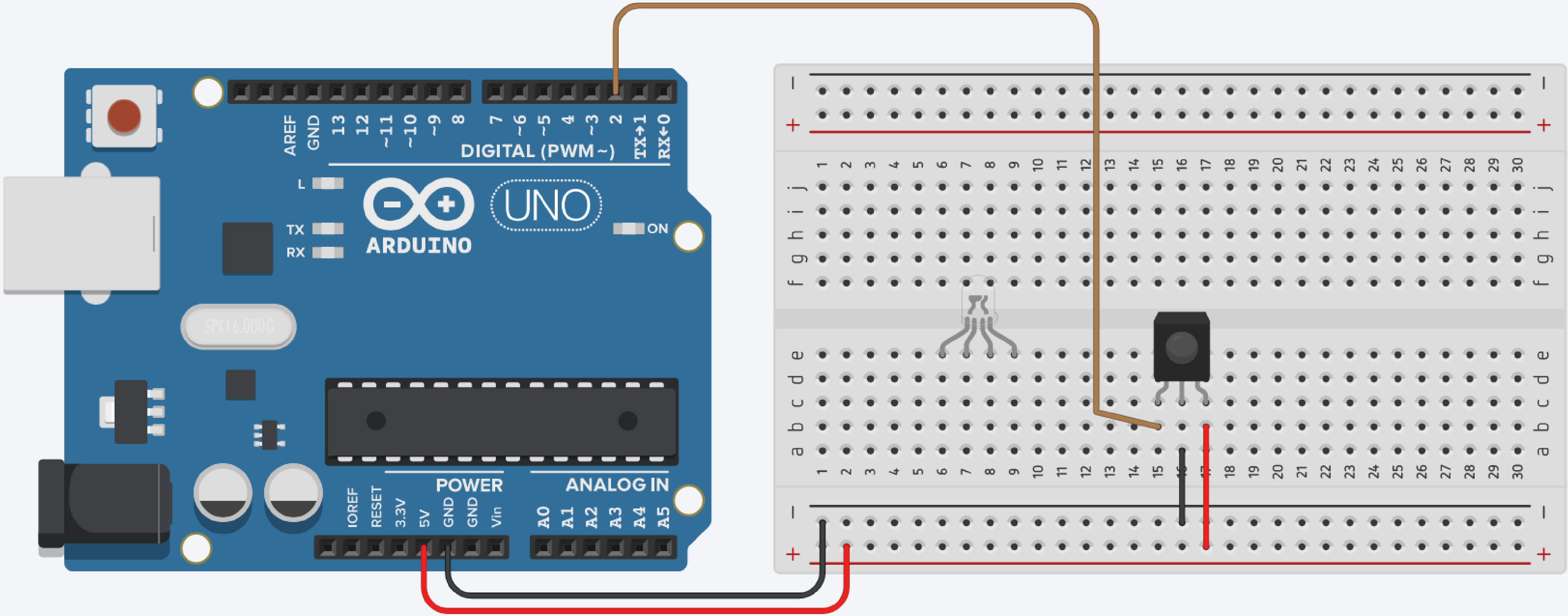
IR Receiver Demo: Steps

5. The signal pin of the IR Receiver connects to pin 2 on Arduino.



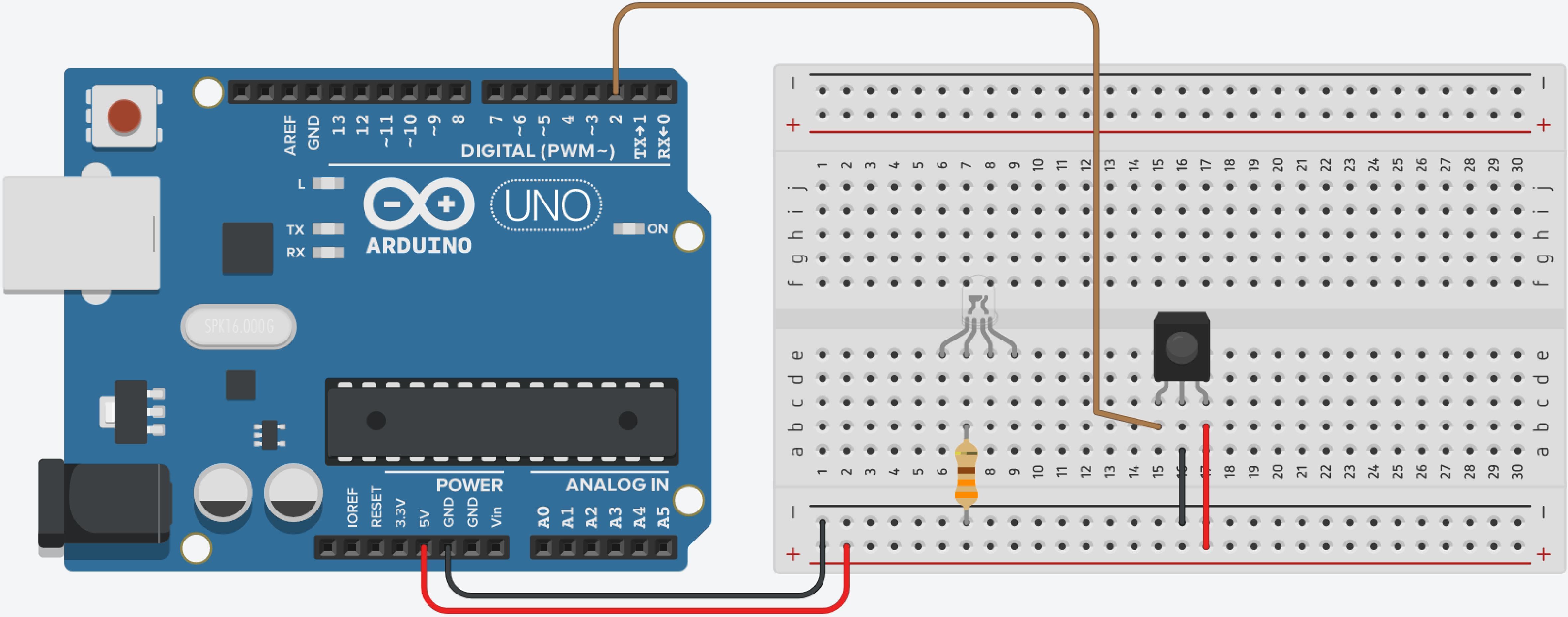
IR-Controlled RGB LED: Steps

6. Plug the RGB LED into the breadboard.



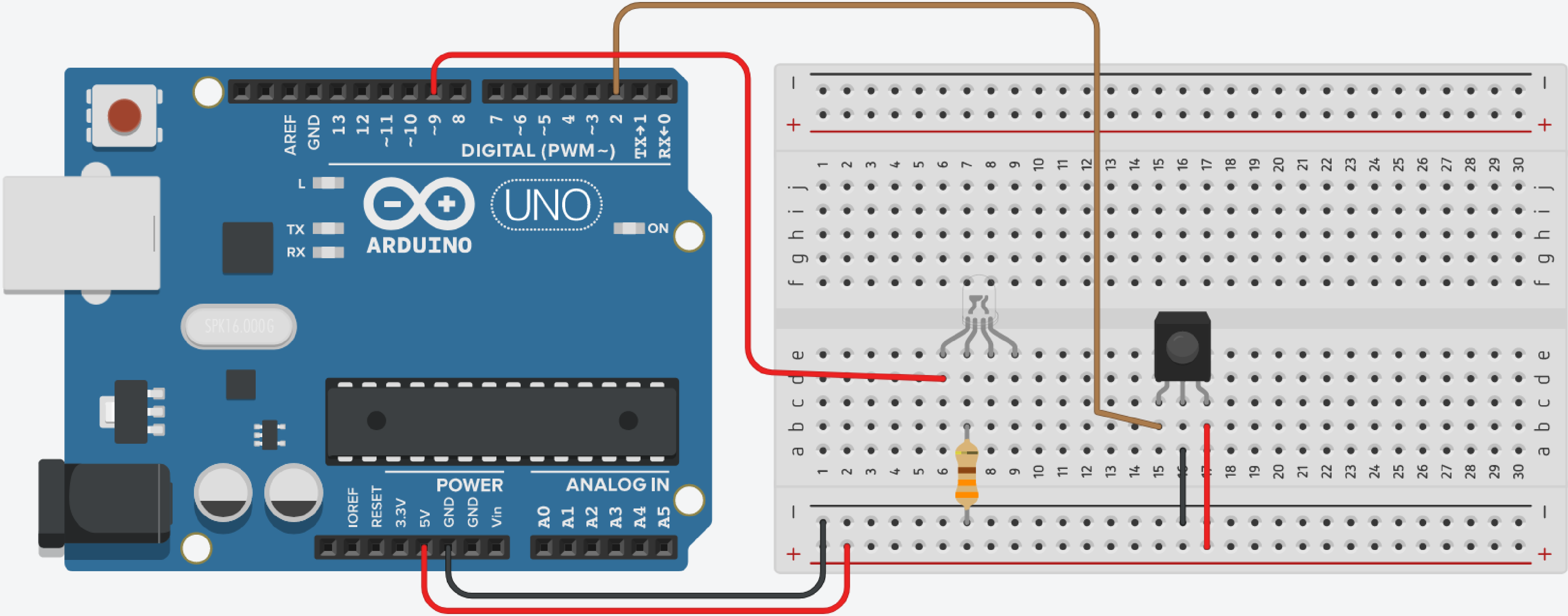
IR-Controlled RGB LED: Steps

7. Connect the LED Common pin the ground using a 330Ω resistor.



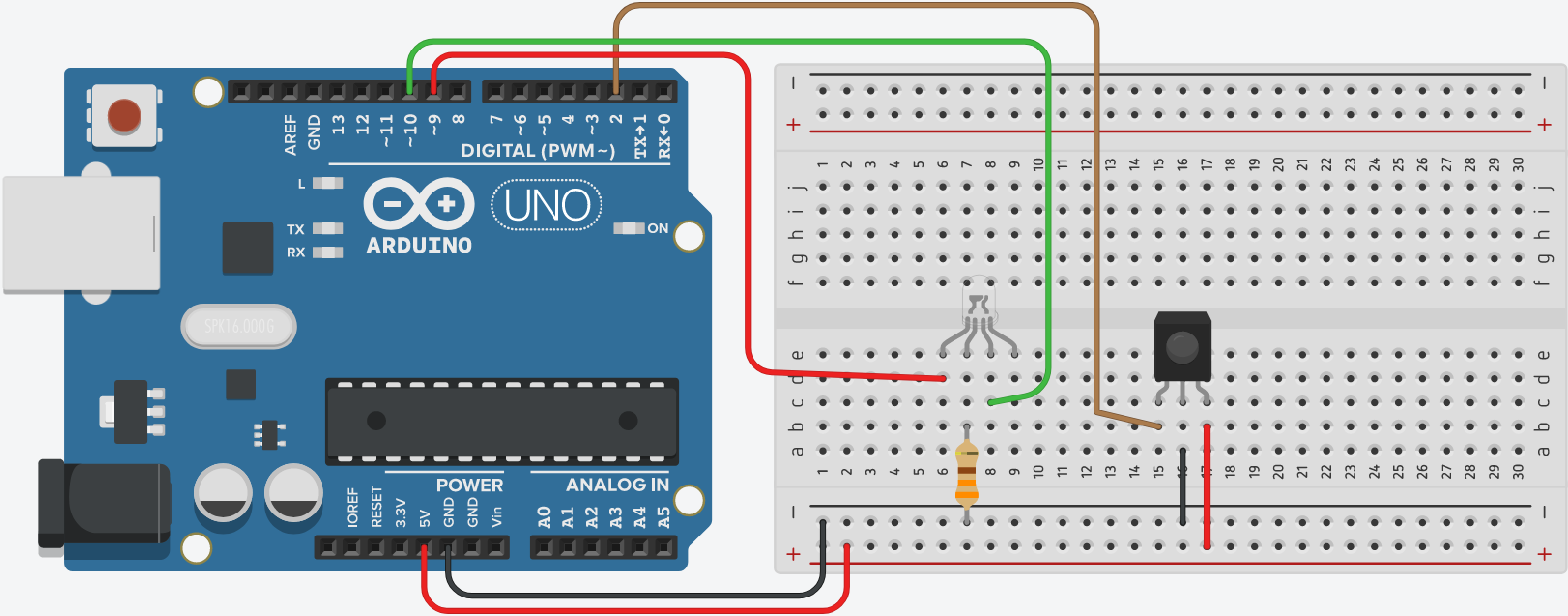
IR-Controlled RGB LED: Steps

8. Connect the Red pin to pin 9 on Arduino.



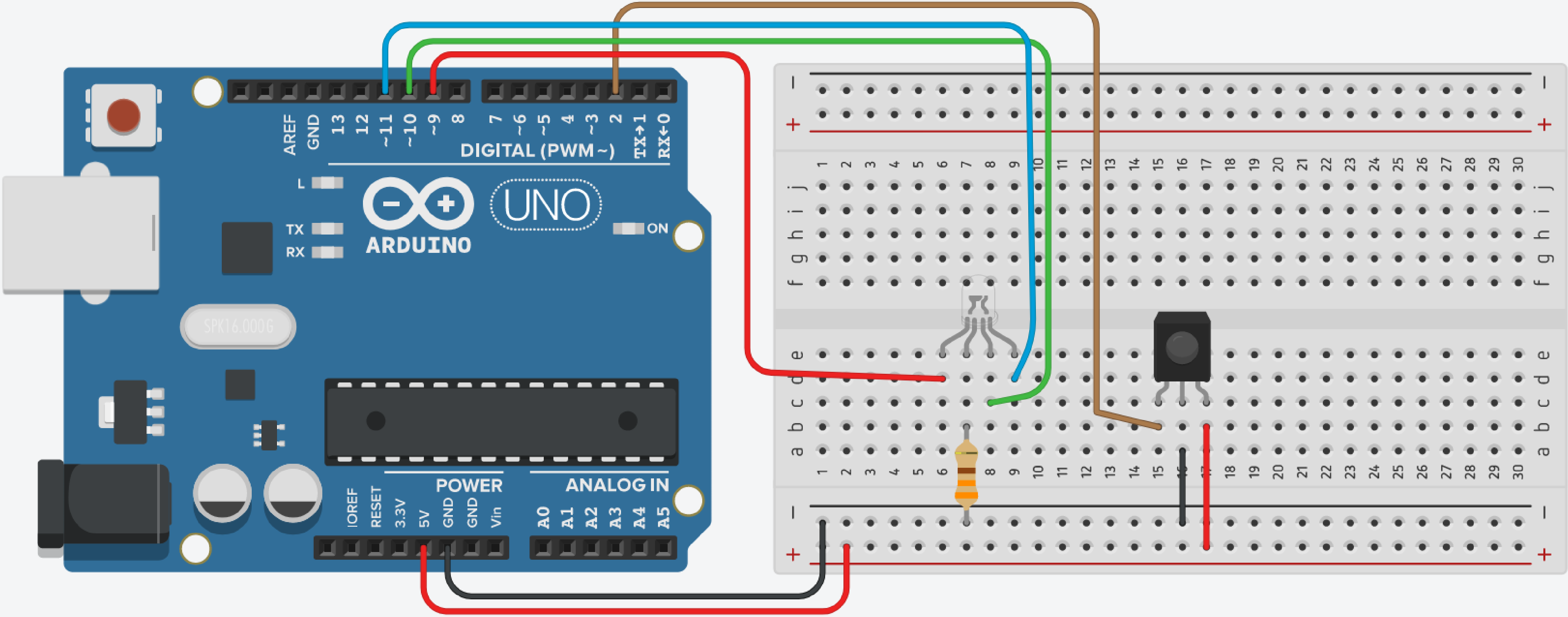
IR-Controlled RGB LED: Steps

9. Connect the Green pin to pin 10 on Arduino.



IR-Controlled RGB LED: Steps

10. Connect the Blue pin to pin 11 on Arduino.



IR-Controlled RGB LED: Code

```
#include <IRremote.h>                              // Import IRremote library
#define RED_PIN 9                                  // Red LED pin
#define GREEN_PIN 10                               // Green LED pin
#define BLUE_PIN 11                                // Blue LED pin
#define RECV_PIN 2                                 // Receiver pin
unsigned int command;                              // Variable to store the infrared command

// Turn off all LEDs
void turn_off(){
  digitalWrite(RED_PIN, LOW);
  digitalWrite(GREEN_PIN, LOW);
  digitalWrite(BLUE_PIN, LOW);
}

// Turn on the red LED
void red_light(){
  digitalWrite(RED_PIN, HIGH);
  digitalWrite(GREEN_PIN, LOW);
  digitalWrite(BLUE_PIN, LOW);
}
```

IR-Controlled RGB LED: Code

```
// Turn on the green LED
void green_light(){
  digitalWrite(RED_PIN, LOW);
  digitalWrite(GREEN_PIN, HIGH);
  digitalWrite(BLUE_PIN, LOW);
}

// Turn on the blue LED
void blue_light(){
  digitalWrite(RED_PIN, LOW);
  digitalWrite(GREEN_PIN, LOW);
  digitalWrite(BLUE_PIN, HIGH);
}

void setup()
{
  Serial.begin(9600);                              // Begin serial communication at 9600 baud rate
  IrReceiver.begin(RECV_PIN);                      // Start the receiver
  pinMode(RED_PIN, OUTPUT);                        // Set red LED pin as output
  pinMode(GREEN_PIN, OUTPUT);                      // Set green LED pin as output
  pinMode(BLUE_PIN, OUTPUT);                       // Set blue LED pin as output
  turn_off();                                      // Reset
}
```

IR-Controlled RGB LED: Code

```
void loop() {
  if (IrReceiver.decode()) {                       // If a button is pressed
    command = IrReceiver.decodedIRData.command;    // Get the infrared command
    Serial.println(command, HEX);                  // Print button command in hexadecimal

    switch(command)                                // Check the infrared command
    {
      case 0x1:                                    // Photo button
        red_light();    break;                     // Turn on the red LED
      case 0x2:                                    // Music button
        green_light();  break;                     // Turn on the green LED
      case 0x3:                                    // Movie/Video button
        blue_light();   break;                     // Turn on the blue LED
      case 0x4:                                    // Exit/Stop button
        turn_off();     break;                     // Turn off all LEDs
    }

    IrReceiver.resume();                           // Receive the next value
  }

  delay(50);                                       // Short delay to improve performance
}
```

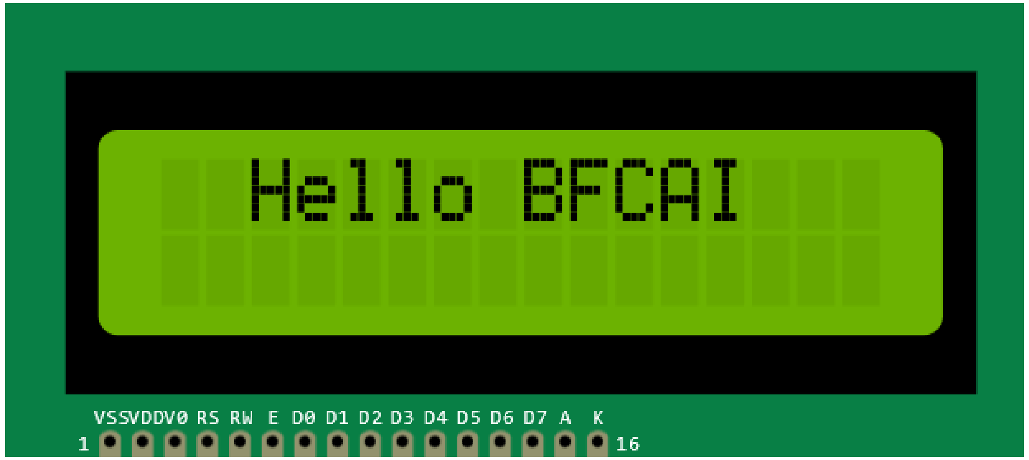
IR-Controlled LCD

◀

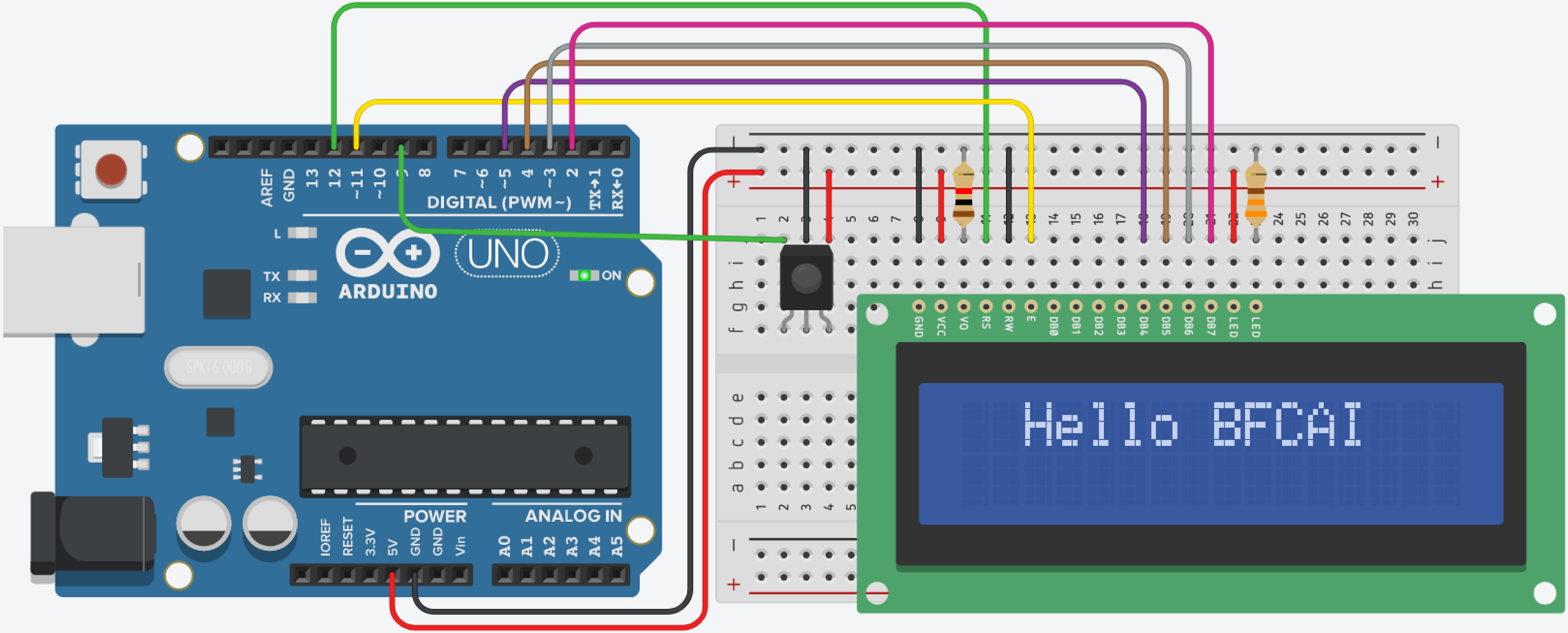
0x7

▶

0x9

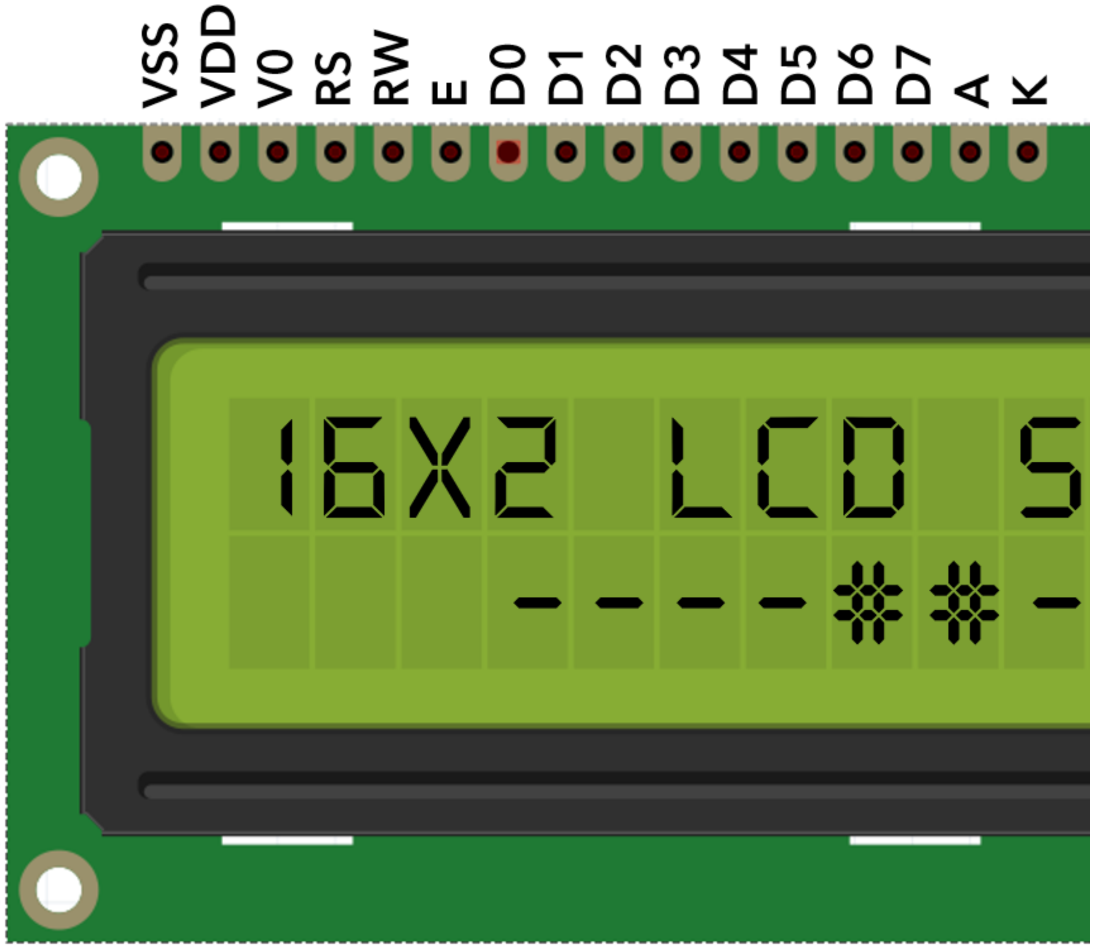


IR-Controlled LCD: Circuit



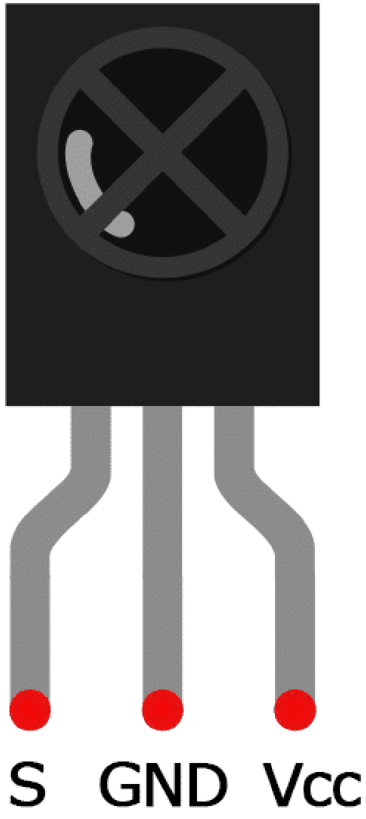
IR-Controlled LCD: LCD Connections

- LCD VSS pin to ground
- LCD VCC pin to 5V
- LCD VO pin to POT wiper
- LCD RS pin to digital pin 12
- LCD R/W pin to ground (write mode)
- LCD Enable pin to digital pin 11
- LCD D4 pin to digital pin 5
- LCD D5 pin to digital pin 4
- LCD D6 pin to digital pin 3
- LCD D7 pin to digital pin 2
- LCD A pin to 5V
- LCD K pin to ground



IR-Controlled LCD: IR Receiver Connections

- The GND pin (-) of IR Receiver connects to the ground.
- The VCC pin of IR Receiver connects to the power.
- The Signal pin (S) of IR Receiver connects to pin 9.



IR-Controlled LCD: Code

```
#include <LiquidCrystal.h>                         // Import LiquidCrystal library
#include <IRremote.h>                              // Import IRremote library
#define RECV_PIN 9                                 // Receiver pin
unsigned int command;                              // Variable to store the infrared command
unsigned int col = 0;                              // Variable to store column position
unsigned int row = 0;                              // Variable to store row position

// Initialize the LiquidCrystal library
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

// Print a message
void printMessage(int col, int row){
    lcd.clear();                                   // Clear LCD
    lcd.setCursor(col, row);                       // Set cursor at position (col, row)
    lcd.print("Hello BFCAI");                      // Display text
}

void setup() {
  Serial.begin(9600);                              // Begin serial communication at 9600 baud rate
  IrReceiver.begin(RECV_PIN);                      // Start the receiver
  lcd.begin(16, 2);                                // Set up the LCD's number of columns and rows
  printMessage(0, 0);                              // Print a message at position (0, 0)
}
```

IR-Controlled LCD: Code

```
void loop() {
  if (IrReceiver.decode()) {                       // If a button is pressed
    command = IrReceiver.decodedIRData.command;    // Get the infrared command
    Serial.println(command, HEX);                  // Print button command in hexadecimal

    switch(command)                                // Check the infrared command
    {
      case 0x9:                                    // Right arrow button
        if(col < 15)                               // Check max position
          col++;                                   // Move right
        break;

      case 0x7:                                    // Left arrow button
        if(col > 0)                                // Check min position
          col--;                                   // Move left
        break;
    }

    printMessage(col, row);                        // Print a message at position (col, row)
    IrReceiver.resume();                           // Receive the next value
  }

  delay(50);                                       // Short delay to improve performance
}
```