

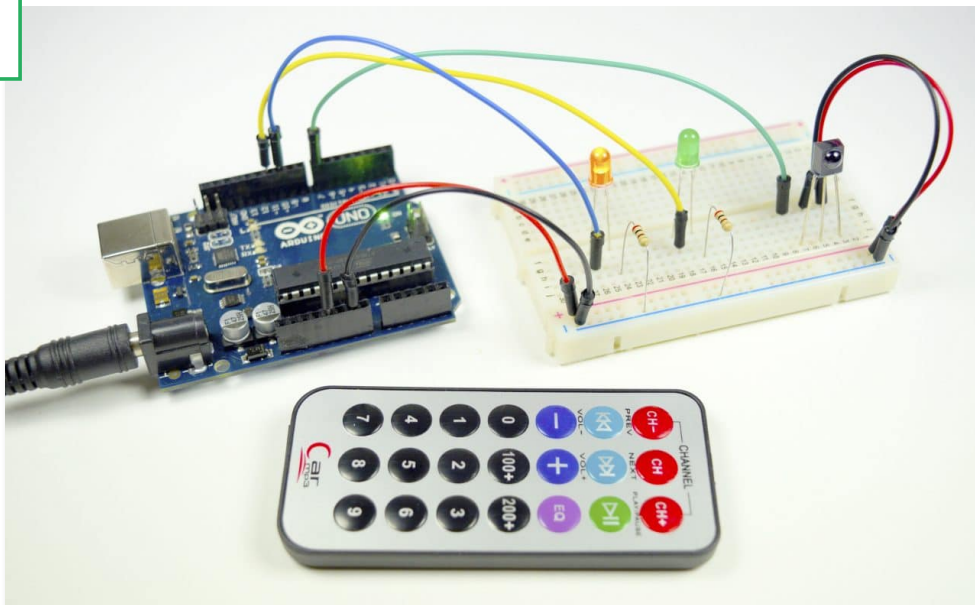


# Circuit Basics

[Raspberry Pi](#)
[Arduino](#)
[DIY Electronics](#)
[Programming](#)
[Videos](#)
[Resources](#)

## HOW TO SET UP AN IR REMOTE AND RECEIVER ON AN ARDUINO

Posted by Krishna Pattabiraman | Arduino | 46



Infrared (IR) communication is a widely used and easy to implement wireless technology that has many useful applications. The most prominent examples in day to

FOLLOW US



SUBSCRIBE

Get new  
tutorials sent to  
your inbox!

SUBSCRIBE

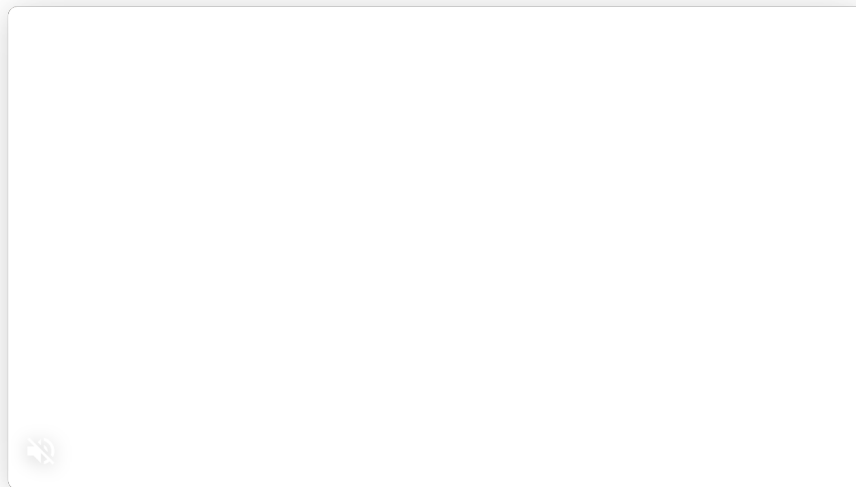
day life are TV/video remote controls, motion sensors, and infrared thermometers.

**PCBWay** HIGH-QUALITY PCB  
**ONLY \$5** FOR 10 PIECES  
• Rogers, HDI, aluminum and rigid-flex PCB are available now  
• Production time 24 hours

PCB ASSEMBLY  
Free shipping + Free stencil  
**ONLY \$30**  
• Component sourcing  
• Quality assurance

There are plenty of interesting Arduino projects that use IR communication too. With a simple IR transmitter and receiver, you can make remote controlled robots, distance sensors, heart rate monitors, DSLR camera remote controls, TV remote controls, and lots more.

In this tutorial I'll first explain what infrared is and how it works. Then I'll show you how to set up an [IR receiver and remote](#) on an [Arduino](#). I'll also show you how to use virtually any IR remote (like the one for your TV) to control things connected to the Arduino.



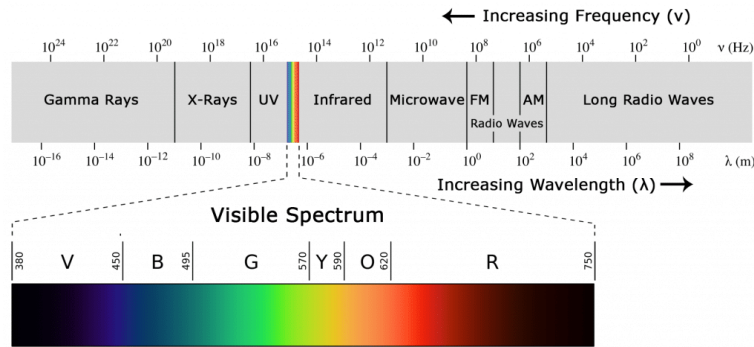
BONUS: I made a quick start guide for this tutorial that you can [download](#) and go back to later if you can't set this up right now. It covers all of the steps, diagrams, and code you need to get started.

Now let's get into the details...

## WHAT IS INFRARED?

Infrared radiation is a form of light similar to the light we see all around us. The only difference between IR light and visible light is the frequency and

wavelength. Infrared radiation lies outside the range of visible light, so humans can't see it:



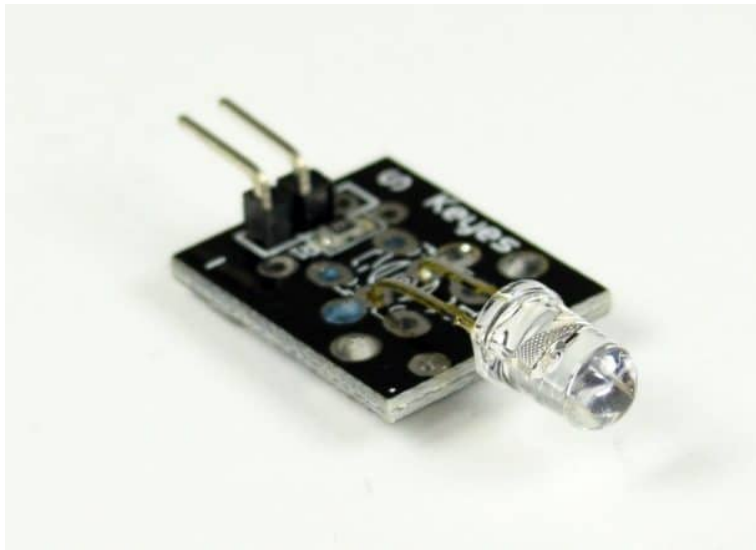
Because IR is a type of light, IR communication requires a direct line of sight from the receiver to the transmitter. It can't transmit through walls or other materials like WiFi or Bluetooth.

## HOW IR REMOTES AND RECEIVERS WORK

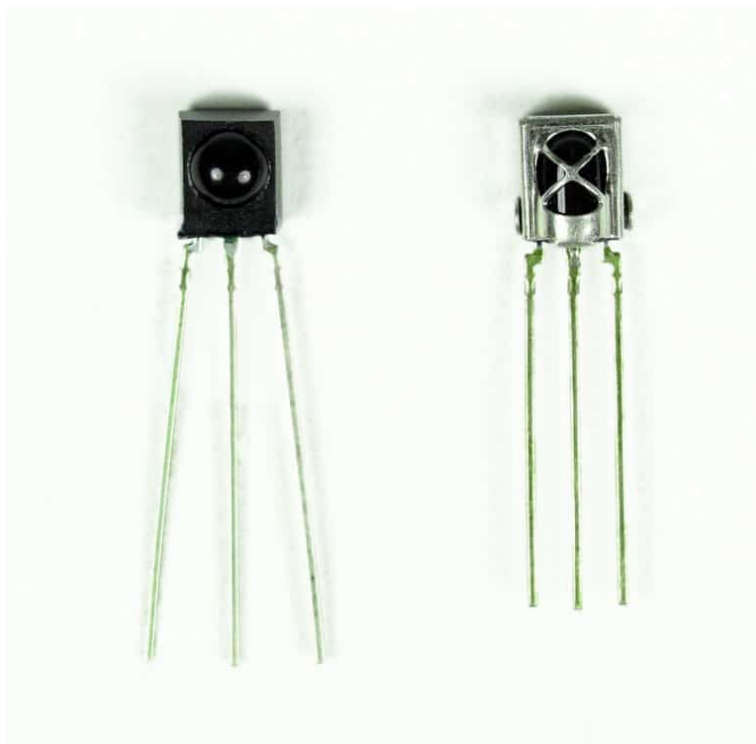
A typical infrared communication system requires an IR transmitter and an IR receiver. The transmitter looks just like a standard LED, except it produces light in the IR spectrum instead of the visible spectrum. If you have a look at the front of a TV remote, you'll see the IR transmitter LED:



The same type of LED is used in IR transmitter breakout boards for the Arduino. You can see it at the front of this Keyes IR transmitter:



The IR receiver is a [photodiode](#) and pre-amplifier that converts the IR light into an electrical signal. [IR receiver diodes](#) typically look like this:



Some may come on a breakout board like this:





## IR SIGNAL MODULATION

IR light is emitted by the sun, light bulbs, and anything else that produces heat. That means there is a lot of IR light noise all around us. To prevent this noise from interfering with the IR signal, a signal modulation technique is used.

In IR signal modulation, an encoder on the IR remote converts a binary signal into a modulated electrical signal. This electrical signal is sent to the transmitting LED. The transmitting LED converts the modulated electrical signal into a modulated IR light signal. The IR receiver then demodulates the IR light signal and converts it back to binary before passing on the information to a microcontroller:



The modulated IR signal is a series of IR light pulses switched on and off at a high frequency known as the carrier frequency. The carrier frequency used by most transmitters is 38 kHz, because it is rare in nature and thus can be distinguished from ambient noise. This way the IR receiver will know that the 38 kHz signal was sent from the transmitter and not picked up from the surrounding environment.

The receiver diode detects all frequencies of IR light, but it has a band-pass filter and only lets through IR at 38 kHz. It then amplifies the modulated signal with a pre-amplifier and converts it to a binary signal before sending it to a microcontroller.

## IR TRANSMISSION PROTOCOLS

The pattern in which the modulated IR signal is converted to binary is defined by a transmission protocol. There are many IR transmission protocols. Sony, Matsushita, NEC, and RC5 are some of the more common protocols.

<a href="#">1. New Hi-Tech</a>	<a href="#">6. IR Remote Sensor</a>
<a href="#">2. Remote Control</a>	<a href="#">7. Remote Control</a>
<a href="#">3. Ir Remote Control</a>	<a href="#">8. Process Flow</a>
<a href="#">4. IR Receiver</a>	<a href="#">9. Infrared Remote</a>
<a href="#">5. Motor Control</a>	<a href="#">10. PCB Layout</a>

The NEC protocol is also the most common type in Arduino projects, so I'll use it as an example to show you how the receiver converts the modulated IR signal to a binary one.

Logical '1' starts with a 562.5  $\mu$ s long HIGH pulse of 38 kHz IR followed by a 1,687.5  $\mu$ s long LOW pulse. Logical '0' is transmitted with a 562.5  $\mu$ s long HIGH pulse followed by a 562.5  $\mu$ s long LOW pulse:



This is how the NEC protocol encodes and decodes the binary data into a modulated signal. Other protocols differ only in the duration of the individual HIGH and LOW pulses.

## IR CODES

Each time you press a button on the remote control, a unique hexadecimal code is generated. This is the information that is modulated and sent over IR to the receiver. In order to decipher which key is pressed, the receiving microcontroller needs to know which code corresponds to each key on the remote.

Different remotes send different codes for the keypresses, so you'll need to determine the code generated for each key on your particular remote. If you can find the datasheet, the IR key codes should be listed. If not though, there is a simple Arduino sketch that will read most of the popular remote controls and print the hexadecimal codes to the serial monitor when you press a key. I'll show you how to set that up in a minute, but first we need to connect the receiver to the Arduino...

## HOW TO CONNECT AN IR RECEIVER TO THE ARDUINO

There are several different types of IR receivers, some are stand-alone, and some are mounted on a breakout board. Check the datasheet for your particular IR receiver since the pins might be arranged differently than the [HX1838 IR receiver and remote set](#) I am using here. However, all IR receivers will have three pins: signal, ground, and Vcc.

Lets get started with the hardware connections. The pin layout on most breakout boards looks like this:

The pinout of most stand-alone diodes is like this:



To connect a breakout board mounted IR receiver, hook it up to the Arduino like this:



To connect a stand-alone receiver diode, wire it like this:



## PROGRAMMING THE IR RECEIVER

Once you have the receiver connected, we can install the Arduino library and start programming. In the examples below, I'll show you how to find the codes sent by your remote, how to find the IR protocol used by your remote, how to print key presses to the serial monitor or an LCD, and finally, how to control the Arduino's output pins with a remote.

## INSTALL THE IRREMOTE LIBRARY

We'll be using the IRremote library for all of the code examples below. You can download a ZIP file of the library from [here](#).

To install the library from the ZIP file, open up the Arduino IDE, then go to Sketch > Include Library > Add .ZIP Library, then select the IRremote ZIP file that you downloaded from the link above.

## FIND THE CODES FOR YOUR REMOTE

To find the key codes for your remote control, upload this code to your Arduino and open the serial monitor:

```
#include <IRremote.h>

const int RECV_PIN = 7;
IRrecv irrecv(RECV_PIN);
decode_results results;

void setup(){
  Serial.begin(9600);
  irrecv.enableIRIn();
  irrecv.blink13(true);
}

void loop(){
  if (irrecv.decode(&results)){
    Serial.println(results.value, HEX);
    irrecv.resume();
  }
}
```



Now press each key on your remote and record the hexadecimal code printed for each key press.



Using the program above, I derived a table of keys and their corresponding codes from the remote that came with my [HX1838 IR receiver and remote set](#). Note that you will receive a 0xFFFFFFFF code when you press a key continuously.

Key	Code
CH-	0xFFA25D
CH	0xFF629D
CH+	0xFFE21D
<<	0xFF22DD
>>	0xFF02FD
>	0xFFC23D
-	0xFFE01F
+	0xFFA857
EQ	0xFF906F

100+	0xFF9867
200+	0xFFB04F
0	0xFF6897
1	0xFF30CF
2	0xFF18E7
3	0xFF7A85
4	0xFF10EF
5	0xFF38C7
6	0xFF5AA5
7	0xFF42BD
8	0xFF4AB5
9	0xFF52AD

## FIND THE PROTOCOL USED BY YOUR REMOTE

Knowing which protocol your remote uses can be useful if you want to work on some more advanced projects. Or you might just be curious. The program below will identify the protocol used by your remote. It should even work on most of the remote controls around your house.

```
#include <IRremote.h>

const int RECV_PIN = 7;
IRrecv irrecv(RECV_PIN);
decode_results results;

void setup(){
  Serial.begin(9600);
  irrecv.enableIRIn();
  irrecv.blink13(true);
}
```

```

void loop(){
  if (irrecv.decode(&results)){
    Serial.println(results.value, HEX);
    switch (results.decode_type){
      case NEC: Serial.println("NEC"); break
      ;
      case SONY: Serial.println("SONY"); break
      ;
      case RC5: Serial.println("RC5"); break
      ;
      case RC6: Serial.println("RC6"); break
      ;
      case DISH: Serial.println("DISH"); break
      ;
      case SHARP: Serial.println("SHARP"); break
      ;
      case JVC: Serial.println("JVC"); break
      ;
      case SANYO: Serial.println("SANYO"); break
      ;
      case MITSUBISHI: Serial.println("MITSUBISHI"); break
      ;
      case SAMSUNG: Serial.println("SAMSUNG"); break
      ;
      case LG: Serial.println("LG"); break
      ;
      case WHYNTER: Serial.println("WHYNTER"); break
      ;
      case AIWA_RC_T501: Serial.println("AIWA_RC_T501"); break
      ;
      case PANASONIC: Serial.println("PANASONIC"); break
      ;
      case DENON: Serial.println("DENON"); break
      ;
      default:
        case UNKNOWN: Serial.println("UNKNOWN"); break
        ;
    }
    irrecv.resume();
  }
}

```

## PRINT KEYS TO THE SERIAL MONITOR

I extended the code above to print the key value instead of the hexadecimal code:

```

#include <IRremote.h>

const int RECV_PIN = 7;
IRrecv irrecv(RECV_PIN);
decode_results results;
unsigned long key_value = 0;

void setup(){
  Serial.begin(9600);
  irrecv.enableIRIn();
  irrecv.blink13(true);
}

```

```
void loop(){
  if (irrecv.decode(&results)){

    if (results.value == 0xFFFFFFFF)
      results.value = key_value;

    switch(results.value){
      case 0xFFA25D:
        Serial.println("CH-");
        break;
      case 0xFF629D:
        Serial.println("CH");
        break;
      case 0xFFE21D:
        Serial.println("CH+");
        break;
      case 0xFF22DD:
        Serial.println("|<<");
        break;
      case 0xFF02FD:
        Serial.println(">>|");
        break ;
      case 0xFFC23D:
        Serial.println(">|");
        break ;
      case 0xFFE01F:
        Serial.println("-");
        break ;
      case 0xFFA857:
        Serial.println("+");
        break ;
      case 0xFF906F:
        Serial.println("EQ");
        break ;
      case 0xFF6897:
        Serial.println("0");
        break ;
      case 0xFF9867:
        Serial.println("100+");
        break ;
      case 0xFFB04F:
        Serial.println("200+");
        break ;
      case 0xFF30CF:
        Serial.println("1");
        break ;
      case 0xFF18E7:
        Serial.println("2");
        break ;
      case 0xFF7A85:
        Serial.println("3");
        break ;
      case 0xFF10EF:
        Serial.println("4");
        break ;
      case 0xFF38C7:
        Serial.println("5");
        break ;
      case 0xFF5AA5:
```



```
Serial.println("6");
break ;
case 0xFF42BD:
Serial.println("7");
break ;
case 0xFF4AB5:
Serial.println("8");
break ;
case 0xFF52AD:
Serial.println("9");
break ;
}
key_value = results.value;
irrecv.resume();
}
}
```



If your remote sends different codes than the ones in the table above, just replace the hex code in each line where it says:

```
case 0xFFA25D:
Serial.println("CH-");
```


In these lines, when the hex code `0xFFA25D` is received, the Arduino prints "CH-".

## HOW THE CODE WORKS

For any IR communication using the `IRremote` library, first we need to create an object called `irrecv` and specify the pin number where the IR receiver is connected (line 3). This object will take care of the protocol and processing of the information from the receiver.



The next step is to create an object called **results**, from the **decode\_results** class, which will be used by the **irrecv** object to share the decoded information with our application (line 5).



In the **void setup()** block, first we configure the serial monitor baud rate. Next we start the IR receiver by calling the **IRrecv** member function **enableIRIn()** (line 10).

The **irrecv.blink13(true)** function on line 11 will blink the Arduino's on board LED every time the receiver gets a signal from the remote control, which is useful for debugging.

In the **void loop()** block, the function **irrecv.decode** will return true if a code is received and the program will execute the code in the if statement. The received code is stored in **results.value**. Then I used a switch to handle each IR code and print the corresponding key value.

[1. New Hi-Tech](#)
[6. Process Flow](#)
[2. Electrical](#)
[7. TV Remote Codes](#)
[3. Cheap Industrial](#)
[8. Remote Control](#)
[4. Ir Remote Control](#)
[9. PCB Layout](#)
[5. Wireless](#)
[10. Program My](#)

Before the switch block starts there is a conditional block:

```
if (results.value == 0xFFFFFFFF)
  results.value = key_value;
```



If we receive 0xFFFFFFFF from the remote, it means a repetition of the previous key. So in order to handle the repeat key pattern, I am storing the hex code in a global variable `key_value` every time a code is received:

```
key_value = results.value;
```

When you receive a repeat pattern, then the previously stored value is used as the current key press.

At the end of the `void loop()` section, we call `irrecv.resume()` to reset the receiver and prepare it to receive the next code.

## PRINT KEYS TO AN LCD

Instead of printing the key values to the serial monitor, you can also display the information on an LCD. Check out our article on [setting up and programming an LCD on the Arduino](#) for more information on programming the LCD, but the basic setup looks like this:



The resistor sets the LCD's backlight brightness. It can be anything from 200 ohms to about 2K ohms. The potentiometer sets the character contrast. I normally use a 10K ohm potentiometer for this one.

Once everything is connected, upload this code to the Arduino:

```

#include <IRremote.h>
#include <LiquidCrystal.h>

const int RECV_PIN = 7;
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
IRrecv irrecv(RECV_PIN);
decode_results results;
unsigned long key_value = 0;

void setup(){
  Serial.begin(9600);
  irrecv.enableIRIn();
  irrecv.blink13(true);
  lcd.begin(16, 2);
}

void loop(){
  if (irrecv.decode(&results)){

    if (results.value == 0xFFFFFFFF)
      results.value = key_value;
    lcd.setCursor(0, 0);
    lcd.clear();

    switch(results.value){
      case 0xFFA25D:
        lcd.print("CH-");
        break;
      case 0xFF629D:
        lcd.print("CH");
        break;
      case 0xFFE21D:
        lcd.print("CH+");
        break;
      case 0xFF22DD:
        lcd.print("|<<");
        break;
      case 0xFF02FD:
        lcd.print(">>|");
        break ;
      case 0xFFC23D:
        lcd.print(">|");
        break ;
      case 0xFFE01F:
        lcd.print("-");
        break ;
      case 0xFFA857:
        lcd.print("+");
        break ;
      case 0xFF906F:
        lcd.print("EQ");
        break ;
      case 0xFF6897:
        lcd.print("0");
        break ;
      case 0xFF9867:
        lcd.print("100+");
        break ;
      case 0xFFB04F:

```



```
lcd.print("200+");  
break ;  
case 0xFF30CF:  
lcd.print("1");  
break ;  
case 0xFF18E7:  
lcd.print("2");  
break ;  
case 0xFF7A85:  
lcd.print("3");  
break ;  
case 0xFF10EF:  
lcd.print("4");  
break ;  
case 0xFF38C7:  
lcd.print("5");  
break ;  
case 0xFF5AA5:  
lcd.print("6");  
break ;  
case 0xFF42BD:  
lcd.print("7");  
break ;  
case 0xFF4AB5:  
lcd.print("8");  
break ;  
case 0xFF52AD:  
lcd.print("9");  
break ;  
}  
key_value = results.value;  
irrecv.resume();  
}  
}
```



Again, if the hex codes don't match the codes output by your remote, just replace them for each character where it says `case 0XXXXXXXX;`.

## USING THE IR REMOTE TO CONTROL THINGS

Now I'll show you a simple demonstration of how you can use the IR remote to control the Arduino's output pins. In this example, we will light up an LED when a particular button is pressed. You can easily modify the code to do things like control servo motors, or activate relays with any button press from the remote.

The example circuit has the IR receiver connected to the Arduino, with a red LED connected to pin 10 and a green LED connected to pin 11:



The code below will write digital pin 10 HIGH for 2 seconds when the "5" button is pressed, and write digital pin 11 HIGH for 2 seconds when the "2" button is pressed:

```
#include <IRremote.h>

const int RECV_PIN = 7;
IRrecv irrecv(RECV_PIN);
decode_results results;
const int redPin = 10;
const int greenPin = 11;

void setup(){
  irrecv.enableIRIn();
  irrecv.blink13(true);
```

```
pinMode(redPin, OUTPUT);
pinMode(greenPin, OUTPUT);
}

void loop(){
  if (irrecv.decode(&results)){

    switch(results.value){
      case 0xFF38C7: //Keypad button "5"
        digitalWrite(redPin, HIGH);
        delay(2000);
        digitalWrite(redPin, LOW);
      }

      switch(results.value){
        case 0xFF18E7: //Keypad button "2"
          digitalWrite(greenPin, HIGH);
          delay(2000);
          digitalWrite(greenPin, LOW);
        }

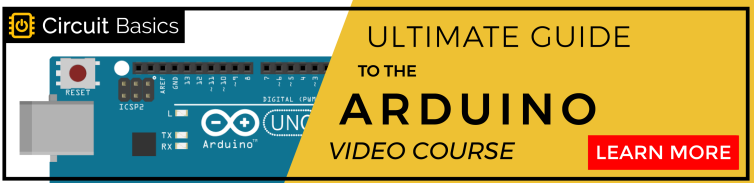
      irrecv.resume();
    }
  }
```



So far we have covered the properties of infrared radiation and how communication happens between the transmitter and receiver. We saw how to identify the IR key codes for a given remote control. We learned how to display key presses on serial monitor and on an LCD screen. Finally I showed you how to control the Arduino's output with the remote. Have fun playing with this and be sure to let us know in the comments if you have any questions or trouble setting this up!

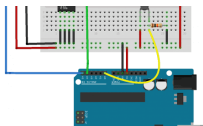
*Krishna Pattabiraman is a frequent guest writer on Circuit Basics and the founder of [www.codeelectron.com](http://www.codeelectron.com).*

JLCPCB - Only \$2 for PCB Prototype (Any Color)  
Great Quality Approved by 600,000+ Customers, 10,000+ PCB Orders Per Day.  
Sign Up & Get at Least Two \$5 Coupons  
Now: <https://jlcpcb.com/quote>

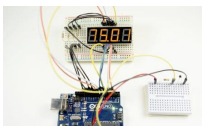


SHARE:     

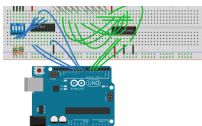
RELATED POSTS



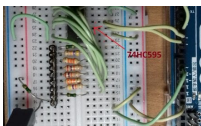
How to Use  
EEPROM on  
the Arduino



How to Set up  
Seven  
Segment  
Displays on  
the Arduino



How to Use  
Shift  
Registers on  
the Arduino



How Shift  
Registers  
Work

1. Ir Remote	6. PCB
2. TV Remote	7. Electronic
3. Remote	8. RCA
4. Program	9. Under



[5. Process](#)[10. New Hi-](#)

## 46 COMMENTS

**StEw Lindenberger** on November 2, 2017 at 4:41 pm

Thank you.

A well presented, informative and useful overview including specific examples for implementation. Bravo.

REPLY

**Jeremy Borg** on November 3, 2017 at 10:18 am

A very well written and informative article. One thing I would have liked to learn more about is how to choose the IR emitter and receiver. My local store stocks several options of each, does it matter which one I choose?

<https://www.fabian.com.mt/en/products/webshop/bycategory/843/name/asc/18/1/infrared-uv-emitters-and-receivers.htm>

REPLY

**Stan** on November 3, 2017 at 3:26 pm

Well written, clear and concise. Keep it up.

REPLY

**Shekhar** on November 17, 2017 at 11:03 am

Are the codes complete?

REPLY

**Saiid** on December 13, 2017 at 6:06 pm

**Sajjad** on December 12, 2017 at 6:06 pm

Hello hope that you all are fine. my Ir reciever giving me continous values on serial moniter although i am sending no signals to it. kindly reply.  
Thanks

REPLY

**anonymous** on March 22, 2019 at 5:37 pm

i'm 2 years too late, but you don't need to type anything in the monitor

REPLY

**Rasmus Agerbo** on January 17, 2021 at 4:13 pm

hi my receiver gets as hot as the sun and it does not give me any values can you help me

REPLY

**peter obasa** on January 6, 2018 at 1:41 pm

Hi, please i try loading the code on uno and nano board this is the error message (

Build options changed, rebuilding all  
C:\Program Files  
(x86)\Arduino\libraries\RobotIRremote\src\IRremoteTools.c  
pp:5:16: error: 'TKD2' was not declared in this scope

int RECV\_PIN = TKD2; // the pin the IR receiver is  
connected to

^

exit status 1  
Error compiling for board Arduino/Genuino Uno.  
)

please what should i do , thanks in advance

**REPLY****Tom** on October 25, 2018 at 4:28 pm

Hi, late but hopefully still helpful, if not for you maybe for somebody else.

This error-message occurs when you're using the "Robot IR Remote" library instead of the "IRremote" library, which you would first have to import, either by using the buildt-in feature of the Arduino-IDE, or by downloading a ZIP-archive.

**REPLY****Saviour** on December 26, 2019 at 1:29 am

Hi I'm Savvy I faced the same error too so I installed the IRremote library folder into my Arduino library and it worked the link is above

**REPLY****Sunil** on August 31, 2018 at 8:28 pm

How to remove receiving NEC repeat code. From my remote control it always display 0xFFFFFFFF but when i presses key fast at once it display correct value like 18E7E817 so how to turn off receiving repeat code. Please help.

**REPLY****Luca Segalla** on September 28, 2018 at 6:49 pm

If you have problems like "error: 'TKD2' was not declared in this scope – int RECV\_PIN = TKD2; // the pin the IR receiver is connected to" just remove the "RobotIRremote" default library and install the "Arduino-IRremote-master". Then rename the folder "Arduino-IRremote-master" in "IRremote". That's all.

**REPLY**

**medo** on October 20, 2018 at 7:08 pm

hi,

i connected atmega 328 ic with 4 relays. really, i made that using two cycles one for the ic and the ir receiver. the cycle was supported using 5v (2 amp) adapter.

The other cycle contained 4 relays, each one have 1 diodes and 574 transistor. this cycle was supported using 5v (1 amp) adapter.

those cycles were connected from (a)- cathode (b)- the ic's output pins to the transistors.

the target was to open/close each relay by lg-tv remote control. the cycle work very well through 1 hour from starting point, but after that it hang and not receive the signals.

um looking forward to hearing from you, why this problem is happened.

REPLY

**trinity fogarty** on November 11, 2018 at 10:15 am

hi there,

for some reason the program never finishes uploading onto my uno. The program verifies properly and I see some on the memory usage figures but it just never finishes. Any ideas?

REPLY

**Colin** on December 17, 2018 at 9:36 pm

Excellent article thank you:)

I'm stuck at the LCD part. I am relatively confident I have connected everything properly, as I have checked and rechecked. However nothing displays on the screen. Is there a way to trouble shoot this?

Thank you,  
Colin

REPLY

**Colin** on December 17, 2018 at 9:38 pm

Its working now! Sorry, I turned on serial monitor in arduino and it started to work. Coincidence?  
Thanks:)

REPLY

**Andreas** on December 22, 2018 at 9:09 pm

Thx helped me out a lot with my project. Clearly structured and nice to read. Worked like a charm

REPLY

**Serge Nadeau** on December 27, 2018 at 1:05 am

Hi,  
I have been looking for an understandable explanation how to use a IR receiver with Arduino for a while. Your explanation is the first that is simple and understandable for a beginner.  
Thanks to put such good quality information on this site.

REPLY

**Hilton Bennett** on January 14, 2019 at 12:21 am

Very well written tutorial. Thanks!  
Is there a way to speed up the response when the remote button is pushed? There seems to be about a 3-4 second delay between button push and LED response in most cases.

REPLY

**Jim DiGriz** on January 16, 2019 at 3:52 am

Wow, this was actually exciting and fun.

Each piece of code worked. I could read the codes. It told me the manufacturer.

Now I'm ready to buy a used/discarded remote from a thrift store, map its keys, and use it to drive relays.

Thanks very much for short clear instructions.

REPLY

**ioan** on February 2, 2019 at 12:55 pm

My 3 IR receivers are always blinking even before I add the code, and aren't receiving any data sent from a functional RGB remote controller.

Please, can someone help me?

REPLY

**ジロラッタ バッグ** on February 24, 2019 at 6:38 am

I have read so many articles or reviews about the blogger lovers except this post is in fact a nice paragraph, keep it up. <http://www.cardtricksdesigns.com/lva.php>

REPLY

**saket** on February 25, 2019 at 5:40 pm

when i try to get the codes for my remote after clicking the serial monitor it automatically starts giving the values why??

REPLY

**Dominic shaji** on April 2, 2019 at 5:29 am

I am building a hand gesture based remote using the ultrasonic sensor. I want to control a music player which already has a remote can anybody help with writing the code. My project is the modification of

<https://www.instructables.com/id/Ultrasonic-gesture-based-TV-remote-control/>

**S.Keerthy** on April 4, 2019 at 6:25 am

I am keerthy, a student of mechanical engineering. I followed the above said steps everything worked properly except one. In the part of printing the keys to the serial monitor, i had a difficulty. While compiling the codes after changing it according to my remote's hexadecimal codes, it throws an error saying that the variable is not declared in the scope.

Here is my changed code:

```
#include
```

```
const int RECV_PIN = 7;  
IRrecv irrecv(RECV_PIN);  
decode_results results;  
unsigned long key_value = 0;
```

```
void setup(){  
  Serial.begin(9600);  
  irrecv.enableIRIn();  
  irrecv.blink13(true);  
}
```

```
void loop(){  
  if (irrecv.decode(&results)){
```

```
    if (results.value == 0xFFFFFFFF)  
      results.value = key_value;
```

```
    switch(results.value){  
      case 1FE48B7:  
        Serial.println("Switch ON/OFF");  
        break;  
      case 1FE58A7:  
        Serial.println("Mode");  
        break;  
      case 1FE7887:  
        Serial.println("MUTE");  
        break;  
      case 1FE807F:
```



```

Serial.println(">||");
break;
case 1FE40BF:
Serial.println("<|");
break;
case 1FE20DF:
Serial.println("EQ");
break;
case 1FEA05F:
Serial.println("VOL-");
break;
case 1FE609F:
Serial.println("VOL+");
break;
case 1FEE01F:
Serial.println("0");
break;
case 1FE10EF:
Serial.println("RPT");
break;
case 1FE906F:
Serial.println("U/SD");
break;
case 1FE50AF:
Serial.println("1");
break;
case 1FED827:
Serial.println("2");
break;
case 1FEF807:
Serial.println("3");
break;
case 1FE30CF:
Serial.println("4");
break;
case 1FEB04F:
Serial.println("5");
break;
case 1FE708F:
Serial.println("6");
break;
case 1FE00FF:
Serial.println("7");

```





```

Serial.println( / ),
break ;
case 1FEF00F:
Serial.println("8");
break ;
case 1FE9867:
Serial.println("9");
break ;
}
key_value = results.value;
irrecv.resume();
}
}

```

And the error says that:

exit status 1  
 'IFEEEEFFFFFF' was not declared in this scope

Help me to rectify my mistakes as fast as you can!!!

REPLY

**DIPIN V SIDHARTH** on April 18, 2019 at 8:14 pm

hi Keerthi , i think the code in your serial monitor is hexa decimal it should change to decimal (hexadecimal to decimal conversion.online converter is available on google.just copy and paste the hexa decimal code in converter and then convert it.copy the converted code and replace that code in your arduino program)...100% working. i already made this..it is useful simple program

REPLY

**sarath** on August 3, 2019 at 12:22 pm

Thanks for the tip.Me also faced same issue but now rectified.

REPLY



**Sergei** on April 22, 2019 at 7:58 pm

Very good explanation and work!

REPLY

**Joseph T** on June 3, 2019 at 2:04 am

Oops! There is a way using send.Raw.

REPLY

**Daniel Fernandes** on June 14, 2019 at 4:59 pm

Very cool!

I would suggest a tutorial to control a led matrix, 8×32, for example, (4 in 1) with this same procedure;

Thank you

REPLY

**Semiconductor Design** on June 28, 2019 at 7:14 am

Thanks for Sharing this is really informative!!

REPLY

**Paul R** on August 23, 2019 at 3:20 am

I find that I often see "FFFFFFFF" when I use some of your code. I added this IF statement around the print statements to omit that:

```
if(results.value != 4294967295){ // decimal equivalent to  
OXFFFFFFFF  
ORIGINAL Serial.println OR lcd.print LINE  
}
```

REPLY

**Joel** on September 15, 2019 at 10:33 pm

my code error says IRremote.h: No such file or directory  
can you help me out?

REPLY

**TITO** on October 25, 2020 at 2:59 am

iINSTALL LIBRRARY

<https://www.arduino-libraries.info/libraries/i-r-remote>

REPLY

**Mr Wolf** on October 16, 2019 at 8:27 pm

Hi, just want to make you my compliments: great tutorial, very well explained. Thanks!

REPLY

**Gary** on December 6, 2019 at 2:57 am

Thank you so much for this tutorial. It was clear, concise, and the examples worked.

I had loaded IRrecvDumpV2 into my project, got all needed codes, but could not figure out how to use them to control a homemade arduino robot. Nothing I tried on my own worked.

The IRrecvDumpV2 instructions were saved into the example directories, and was 123 pages long. The doc seemed to be musings of folks who truly live in an embedded world, and went deep into the artistry and wonderment of their cleverness.

Thank you for helping me see the light. These 2 lines are what significantly helped me move forward with my project.

```
if (results.value == 0xFFFFFFFF)
  results.value = key_value;
```

Thank you, thank you, thank you

REPLY

**Jean** on December 27, 2019 at 8:18 pm

Very good job but there s somethng I can't understand. When I read the variable 'results.decode\_type' I get a number from 1 to 7 and not a string like those you use in

```
the program (NEC, SONY ...)  
switch (results.decode_type){  
case NEC: Serial.println("NEC"); break ;  
case SONY: Serial.println("SONY"); break ;
```

By the way I use the same Library as you,, results.value  
codes and the number of bits are perfect  
Can you help me?  
Thanks

REPLY

**Avanish** on July 15, 2020 at 4:12 pm

it should be printing unknown if any one of the  
above is not printing,

Anyway it depends upon which protocol your  
remote is using like sony,  
lg they have their unique protocols . May be your  
remote is not matching  
with the listed strings

REPLY

**Jan Speyer** on February 16, 2020 at 12:09 pm

Dear sir,

In the scheme 'Using the IR Remote to Control Things' the  
two resistors are connected to the anode. They should be  
connected to the kathode-side(ground), as I found out.

Thanks for this great tutorial!

Kind regards,  
Jan Speyer, the Netherlands

REPLY

**Jan Speyer** on February 16, 2020 at 12:18 pm

I also found out that on my breadboard the '+' is on the left  
and the '-' on the right. Sorry for my comment:-)

REPLY

**alfredo** on February 18, 2020 at 10:50 pm

Grazie mille, complimenti!!!!!!!

REPLY

**Harold** on September 25, 2020 at 5:05 pm

I am getting this error Message ? This report would have more information with

“Show verbose output during compilation”

enabled in File > Preferences.

Arduino: 1.0.6 (Windows 2000), Board: “Arduino Uno”

In file included from sketch\_sep25j.ino:1:

C:\Documents and Settings\HAC\My

Documents\Arduino\libraries\Arduino-IRremote-

2.6.1\src\IRremote.h:486: error: ISO C++ forbids initialization of member ‘sendPin’

C:\Documents and Settings\HAC\My

Documents\Arduino\libraries\Arduino-IRremote-

2.6.1\src\IRremote.h:486: error: making ‘sendPin’ static

REPLY

**Harold** on September 25, 2020 at 5:06 pm

I am Getting this error ? This report would have more information with

“Show verbose output during compilation”

enabled in File > Preferences.

Arduino: 1.0.6 (Windows 2000), Board: “Arduino Uno”

In file included from sketch\_sep25j.ino:1:

C:\Documents and Settings\HAC\My

Documents\Arduino\libraries\Arduino-IRremote-

2.6.1\src\IRremote.h:486: error: ISO C++ forbids initialization of member ‘sendPin’

C:\Documents and Settings\HAC\My

Documents\Arduino\libraries\Arduino-IRremote-

2.6.1\src\IRremote.h:486: error: making ‘sendPin’ static

REPLY

**Volker** on September 29, 2020 at 12:56 pm

:-) Like it.

REPLY

**Alex** on November 14, 2020 at 12:37 pm

Hello,

I'm trying to copy a remote controller of which I have the schematics, but not the parts.

It seems that it's using the Toshiba protocol, but I can't find information about it... The part I'm trying to simulate is the PT2248

REPLY

**Daniel John Ohm** on January 25, 2021 at 12:33 pm

I had a whole bunch of problems getting the codes to work for my arduino.

the first code, instead of giving me IR codes, it would just print a 0 to the serial port every time i pressed a button on the controller.

the second code didnt work at all...

I tried a number of things, including testing all the different examples in the IRremote library....

after i failed to get it to work seamlessly, i discovered that the IRremote library has been updated in the last few months.

I managed to solve the problem by reverting to an earlier update for the IRremote library (2.7.0), and now everything is working perfectly!

thanks very much!

REPLY

## LEAVE A REPLY

Your email address will not be published. Required fields are marked \*

COMMENT

NAME \*

EMAIL \*

WEBSITE

☐ Save my name, email, and website in this browser for the next time I comment.

☐ Notify me of follow-up comments by email.

☐ Notify me of new posts by email.

For security, use of Google's reCAPTCHA service is required which is subject to the Google [Privacy Policy](#) and [Terms of Use](#).

☐ I agree to these terms.

POST COMMENT

Copyright **Circuit Basics**

[Raspberry Pi](#) [Arduino](#) [DIY Electronics](#) [Programming](#) [Videos](#) [Resources](#) [About](#) [Contact Us](#)

[Privacy Policy](#)

