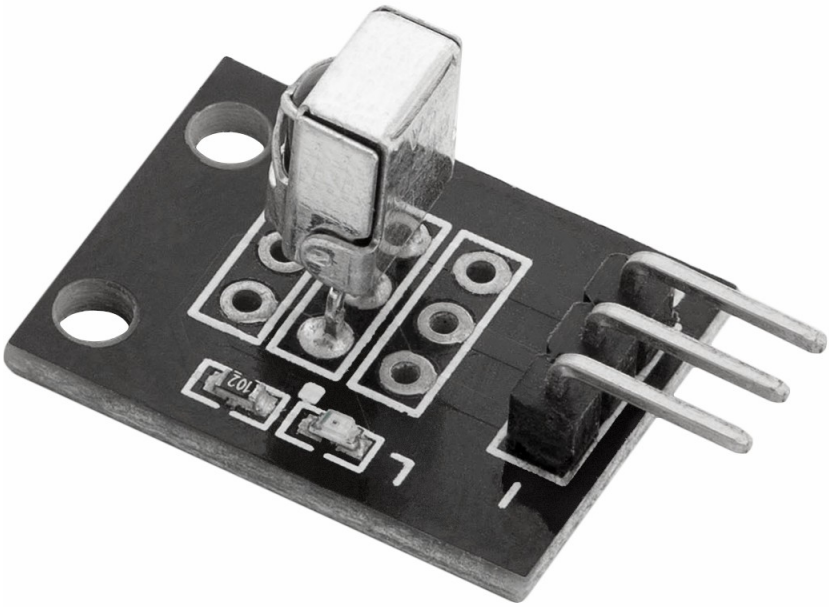
Bienvenido.

Gracias por adquirir nuestro *Módulo Receptor de Infrarrojos AZ-Delivery KY-022.* En las siguientes páginas, se le explicará cómo utilizar y configurar este práctico dispositivo.

¡Que te diviertas!



Areas de aplicación
Educación y enseñanza: uso en escuelas, universidades e instituciones de formación para enseñar los conceptos básicos de electrónica, programación y sistemas integrados. Investigación y desarrollo: Uso en proyectos de investigación y desarrollo para crear prototipos y experimentos en los campos de la electrónica y la informática. Desarrollo de prototipos: Uso en el desarrollo y prueba de nuevos circuitos y dispositivos electrónicos. Proyectos Hobby and Maker: utilizado por entusiastas y aficionados a la electrónica para desarrollar e implementar proyectos de bricolaje.

Conocimientos y habilidades requeridos.
Conocimientos básicos de electrónica e ingeniería eléctrica. Conocimientos de programación, especialmente en el lenguaje de programación C/C++. Capacidad para leer esquemas y diseñar circuitos simples. Experiencia trabajando con componentes electrónicos y soldadura.

Condiciones de operación
El producto sólo puede funcionar con los voltajes especificados en la hoja de datos para evitar daños. Se requiere una fuente de alimentación CC estabilizada para su funcionamiento. Al realizar la conexión a otros componentes y circuitos electrónicos, se deben observar los límites máximos de corriente y voltaje para evitar sobrecargas y daños.

Condiciones ambientales
El producto debe utilizarse en un ambiente limpio y seco para evitar daños causados por la humedad o el polvo. Proteger el producto de la luz solar directa (UV)

Uso previsto
El producto está diseñado para su uso en entornos educativos, de investigación y desarrollo. Se utiliza para desarrollar, programar y crear prototipos de proyectos y aplicaciones electrónicos. El producto Sensor no pretende ser un producto de consumo terminado, sino más bien una herramienta para usuarios con conocimientos técnicos, incluidos ingenieros, desarrolladores, investigadores y estudiantes.

Uso inadecuado previsible
El producto no es adecuado para uso industrial o aplicaciones relevantes para la seguridad. No se permite el uso del producto en dispositivos médicos o para fines de aviación y viajes espaciales.

desecho
¡No lo deseche con la basura doméstica! Su producto es acorde al europeo. Directiva sobre residuos de aparatos eléctricos y electrónicos que deben eliminarse de forma respetuosa con el medio ambiente. Las valiosas materias primas contenidas en ellos se pueden reciclar. convertirse en. La aplicación de esta directiva contribuye a la protección del medio ambiente y la salud. Utilice el punto de recogida habilitado por su municipio para devolver y Reciclaje de aparatos eléctricos y electrónicos viejos. N.º registro RAEE: DE 62624346

descarga electrostática
Atención: Las descargas electrostáticas pueden dañar el producto. Nota: Conéctese a tierra antes de tocar el producto, por ejemplo usando una muñequera antiestática o tocando una superficie metálica conectada a tierra.

instrucciones de seguridad
Aunque nuestro producto cumple con los requisitos de la Directiva RoHS (2011/65/UE) y no contiene sustancias peligrosas en cantidades superiores a los límites permitidos, es posible que aún queden residuos. Observe las siguientes instrucciones de seguridad para evitar riesgos químicos: Precaución: La soldadura puede producir humos que pueden ser perjudiciales para la salud. Nota: Utilice un extractor de humos de soldadura o trabaje en un área bien ventilada. Si es necesario, use una máscara respiratoria. Precaución: algunas personas pueden ser sensibles a ciertos materiales o químicos contenidos en el producto. Nota: Si se produce irritación de la piel o reacciones alérgicas, suspenda su uso y, si es necesario, consulte a un médico. Precaución: Mantenga el producto fuera del alcance de los niños y las mascotas para evitar el contacto accidental y la ingestión de piezas pequeñas. Nota: Guarde el producto en un recipiente cerrado y seguro cuando no esté en uso. Atención: Evite el contacto del producto con alimentos y bebidas. Nota: No almacene ni utilice el producto cerca de alimentos para evitar la contaminación. Aunque nuestro producto cumple con los requisitos de la Directiva RoHS (2011/65/UE) y no contiene sustancias peligrosas en cantidades superiores a los límites permitidos, es posible que aún queden residuos. Observe las siguientes instrucciones de seguridad para evitar riesgos químicos: Precaución: La soldadura puede producir humos que

pueden ser perjudiciales para la salud. Nota: Utilice un extractor de humos de soldadura o trabaje en un área bien ventilada. Si es necesario, use una máscara respiratoria. Precaución: algunas personas pueden ser sensibles a ciertos materiales o químicos contenidos en el producto. Nota: Si se produce irritación de la piel o reacciones alérgicas, suspenda su uso y, si es necesario, consulte a un médico. Precaución: Mantenga el producto fuera del alcance de los niños y las mascotas para evitar el contacto accidental y la ingestión de piezas pequeñas. Nota: Guarde el producto en un recipiente cerrado y seguro cuando no esté en uso. Atención: Evite el contacto del producto con alimentos y bebidas. Nota: No almacene ni utilice el producto cerca de alimentos para evitar la contaminación. El producto contiene componentes electrónicos sensibles y bordes afilados. Un manejo o montaje inadecuado puede provocar lesiones o daños. Observe las siguientes instrucciones de seguridad para evitar riesgos mecánicos: Atención: La placa de circuito y los conectores del producto pueden tener bordes afilados. Tenga cuidado para evitar cortes. Nota: Utilice guantes protectores adecuados al manipular y montar el producto. Precaución: Evite una presión excesiva o tensión mecánica en la placa y los componentes. Nota: Monte el producto únicamente en superficies estables y planas. Utilice espaciadores y carcasas adecuados para minimizar la tensión mecánica. Atención: Asegúrese de que el producto esté bien sujeto para evitar resbalones o caídas accidentales. Nota: Utilice un soporte adecuado o un montaje seguro en gabinetes o en placas de montaje. Precaución: Asegúrese de que todas las conexiones de los cables estén conectadas de forma segura y correcta para evitar tensiones y desenchufes accidentales. Nota: Tienda los cables de manera que no estén bajo tensión y no representen un peligro de tropiezo. El producto funciona con voltajes y corrientes eléctricas que, si se usan incorrectamente, pueden provocar descargas eléctricas, cortocircuitos u otros peligros. Observe las siguientes instrucciones de seguridad para evitar riesgos eléctricos: Atención: Utilice el producto únicamente con los voltajes especificados. Nota: Los límites de rendimiento del producto se pueden encontrar en la hoja de datos asociada. Precaución: Evite cortocircuitos entre los conectores y componentes del producto. Nota: Asegúrese de que ningún objeto conductor toque o puentee la placa de circuito. Utilice herramientas aisladas y preste atención a la disposición de las conexiones. Precaución: No realice ningún trabajo en el producto cuando esté conectado a una fuente de alimentación. Nota: Desconecte el producto de la alimentación antes de realizar cambios en el circuito o conectar o quitar componentes. Precaución: No exceda las clasificaciones actuales especificadas para las entradas y salidas del producto. Nota: Los límites de rendimiento del producto se pueden encontrar en las especificaciones técnicas o en la ficha técnica. Atención: Asegúrese de que las fuentes de alimentación utilizadas sean estables y del tamaño correcto. Nota: Utilice únicamente fuentes de alimentación probadas y adecuadas para evitar fluctuaciones de voltaje y sobrecargas. Atención: Mantenga una distancia suficiente de las partes vivas para evitar el contacto accidental. Nota: Asegúrese de que el cableado esté dispuesto de forma segura y clara según el voltaje utilizado. Precaución: Utilice carcasas aislantes o cubiertas protectoras para proteger el producto del contacto directo. Nota: Coloque el producto en un estuche no conductor para evitar contactos accidentales y cortocircuitos. El producto y sus componentes pueden calentarse durante el funcionamiento. La manipulación inadecuada o la sobrecarga del producto pueden provocar quemaduras, daños o incendios. Observe las siguientes instrucciones de seguridad para evitar riesgos térmicos: Precaución: Asegúrese de que el producto se utilice dentro de las temperaturas de funcionamiento recomendadas. Nota: El rango de temperatura de funcionamiento recomendado suele estar entre-40 °C y +85 °C. Consulta la información específica en la ficha técnica del producto. Atención: No coloque el producto cerca de fuentes de calor externas como radiadores o luz solar directa. Nota: Asegúrese de que el producto funcione en un área fresca y bien ventilada. Atención: Asegúrese de que el producto esté bien ventilado para evitar el sobrecalentamiento. Nota: Utilice ventiladores o disipadores de calor cuando utilice el producto en un recinto cerrado o en un entorno con circulación de aire limitada. Atención: Monte el producto sobre superficies resistentes al calor y en carcasas resistentes al calor. Nota: Utilice materiales de gabinete que puedan soportar altas temperaturas para evitar daños o riesgos de incendio. Precaución: Implemente un control de temperatura cuando utilice un gabinete y, si es necesario, mecanismos de protección que apaguen el producto si se sobrecalienta. Nota: Nota: Utilice sensores de temperatura y software adecuado para controlar la temperatura del producto y apague el sistema si es necesario. Precaución: Evite sobrecargas que puedan causar un calentamiento excesivo de los componentes. Nota: Para evitar el sobrecalentamiento, no exceda los límites de corriente y voltaje especificados. Precaución: Los cortocircuitos pueden generar mucho calor y provocar incendios. Nota: Asegúrese de que todas las conexiones sean correctas y seguras y que ningún objeto conductor pueda causar cortocircuitos accidentalmente.

Índice

Introducción

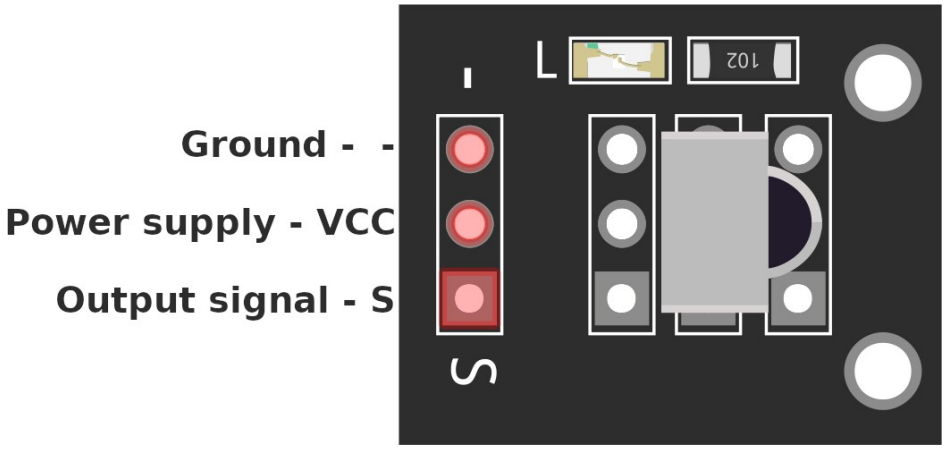
El módulo sensor receptor de infrarrojos KY-022 consta de un receptor de infrarrojos *1838*, una resistencia *de 1kΩ* y un LED rojo. El receptor de infrarrojos reacciona a la luz infrarroja de *38kHz*. El receptor de infrarrojos es un fotodiodo y un preamplificador que convierte la luz infrarroja en una señal eléctrica.

Especificaciones

"	Rango de tensión de funcionamiento:	de 3,3 V a 5 V CC
"	Consumo de corriente:	1,5 mA pico
"	Tipo de chip:	VS1838B
"	Distancia de recepción:	17m
"	Frecuencia portadora:	38 kHz
"	Longitud de onda infrarroja:	940 nm
"	Duración del pulso:	de 400µs a 2000µs
"	Dimensiones:	9 x 22 mm [0,9 X 0,35 pulgadas].

El pinout

El módulo receptor de infrarrojos KY-022 tiene tres pines. El diagrama de pines se muestra en la siguiente imagen:



Principio de funcionamiento

La luz infrarroja es una luz que emiten el sol, las bombillas o cualquier otra cosa que produzca calor. Eso significa que hay mucho ruido de luz infrarroja a nuestro alrededor. Para evitar que este ruido interfiera en las señales infrarrojas, se desarrolla una técnica de modulación de la señal.

En la modulación de la señal de infrarrojos, un codificador del mando a distancia por infrarrojos convierte una señal binaria en una señal eléctrica modulada. Esta señal eléctrica se envía al LED de infrarrojos. El LED convierte la señal eléctrica modulada en una señal luminosa infrarroja modulada. A continuación, el receptor de infrarrojos o fotodiodo demodula la señal de luz infrarroja y la convierte de nuevo en eléctrica (binaria), antes de pasar la información a un microcontrolador. La señal infrarroja modulada es una serie de impulsos de luz infrarroja que se *encienden* y *apagan* a una frecuencia alta, conocida como frecuencia portadora. La frecuencia portadora utilizada por la mayoría de los transmisores es *de 38 kHz* porque es poco frecuente en la naturaleza y puede distinguirse del ruido ambiente. De este modo, el receptor de infrarrojos sabe que la señal de 38 kHz ha sido enviada desde el transmisor y no captada por el entorno.

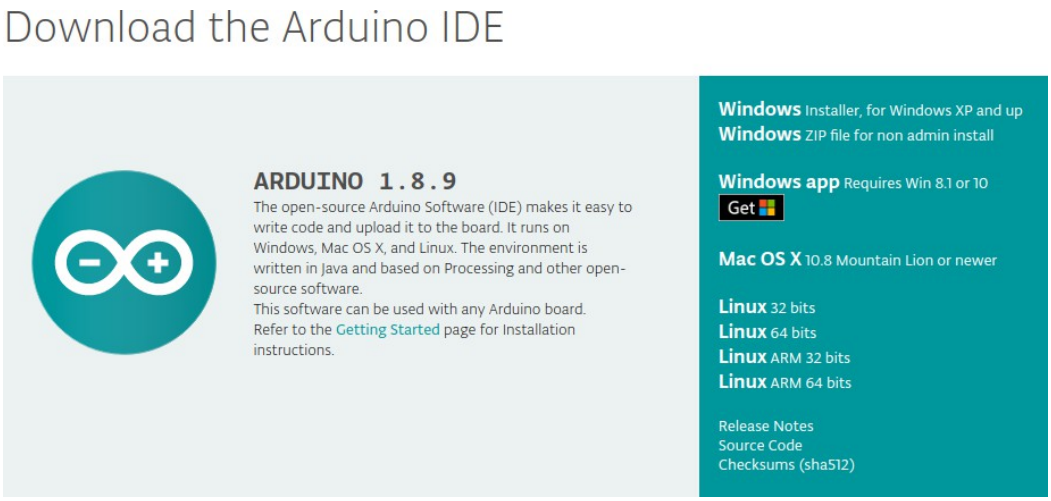
El diodo receptor detecta todas las frecuencias de luz infrarroja, pero tiene un filtro de paso de banda y sólo deja pasar la luz infrarroja a la frecuencia de *38 kHz*. A continuación, amplifica la señal modulada con un preamplificador y la convierte en una señal binaria, antes de enviarla a un microcontrolador. El patrón de conversión en binario de la señal infrarroja modulada se define mediante un protocolo de transmisión.

Existen muchos protocolos de transmisión por infrarrojos: *Sony*, *Matsushita*, *NEC* y *RC5* son algunos de los protocolos más comunes. El *protocolo NEC* es también el tipo más común en proyectos embebidos, por lo que se utiliza en este eBook como ejemplo para mostrar cómo el receptor convierte la señal infrarroja modulada en un uno binario. El 1 lógico comienza con un pulso *HIGH de 562,5µs de* duración a *38kHz de* luz infrarroja seguido de un pulso *LOW de 1687,5µs de* duración. El *0* lógico se transmite con un pulso *HIGH* de 562*,5µs de duración seguido* de un pulso *LOW de 562,5µs de* duración. Otros protocolos sólo difieren en la duración de los pulsos *HIGH* y *LOW* individuales.

Cada vez que se pulsa un botón del mando a distancia, se genera un código hexadecimal único. Esta es la información que se modula y se envía a través de luz infrarroja al receptor. Para descifrar qué tecla se ha pulsado, el microcontrolador receptor necesita saber qué código corresponde a cada tecla del mando. Los distintos mandos envían códigos diferentes para las teclas pulsadas, por lo que hay que determinar el código generado para cada tecla del mando concreto. Se puede encontrar en la hoja de datos (los códigos de las teclas infrarrojas deben estar listados en la hoja de datos). Hay un simple sketch que lee la mayoría de los controles remotos populares y muestra los códigos hexadecimales al Monitor Serial cuando se presiona una tecla (más adelante en el texto).

Cómo configurar Arduino IDE

Si el IDE de Arduino no está instalado, sigue el *enlace* y descarga el archivo de instalación para el sistema operativo que elijas.



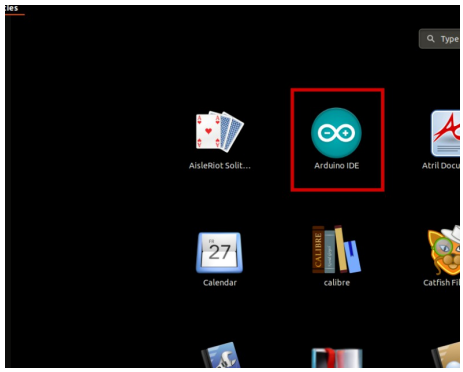
Para los usuarios *de Windows*, haga doble clic en el archivo *.exe* descargado y siga las instrucciones de la ventana de instalación.

Para los usuarios *de Linux*, descarga un archivo con la extensión *.tar.xz*, que hay que extraer. Cuando esté extraído, ve al directorio extraído y abre el terminal en ese directorio. Hay que ejecutar dos scripts *.sh*, el primero llamado *arduino-linux-setup.sh* y el segundo llamado *install.sh*.

Para ejecutar el primer script en el terminal, abra el terminal en el directorio extraído y ejecute el siguiente comando:
sh arduino-linux-setup.sh nombre_usuario
nombre_usuario - es el nombre de un superusuario en el sistema operativo Linux. Se debe introducir una contraseña para el superusuario cuando se inicie el comando. Espera unos minutos a que el script lo complete todo.

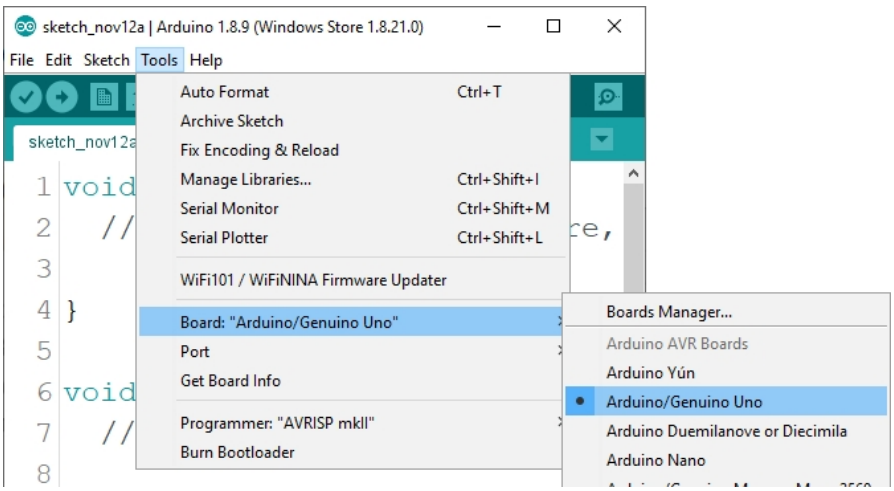
El segundo script llamado *install.sh* script tiene que ser utilizado después de la instalación del primer script. Ejecute el siguiente comando en el terminal (directorio extraído): **sh install.sh**

Después de la instalación de estas secuencias de comandos, vaya a *Todas las aplicaciones*, donde el
Arduino IDE está instalado.



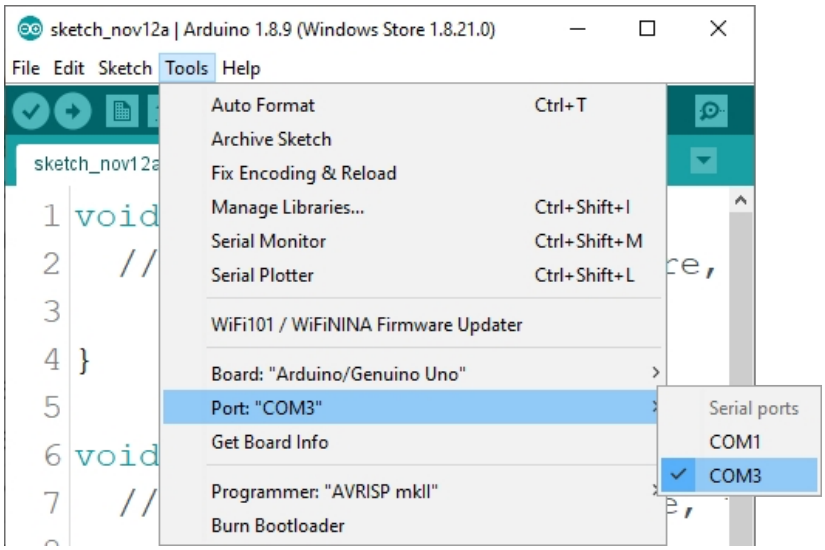
Casi todos los sistemas operativos vienen con un editor de texto preinstalado (por ejemplo, *Windows* viene con *Notepad*, *Linux Ubuntu* viene con *Gedit*, *Linux Raspbian* viene con *Leafpad*, etc.). Todos estos editores de texto son perfectamente adecuados para el propósito del libro electrónico.

Lo siguiente es comprobar si tu PC puede detectar una placa Atmega328p. Abra recién instalado Arduino IDE, y vaya a:
Herramientas > Tablero > {nombre de su tablero aquí}
{el nombre de tu placa aquí} debería ser el *Arduino/Genuino Uno*, como se puede ver en la siguiente imagen:



Hay que seleccionar el puerto al que está conectada la placa Atmega328p. Ir a: *Herramientas > Puerto > {el nombre del puerto va aquí}* y cuando la placa Atmega328p está conectada al puerto USB, se puede ver el nombre del puerto en el menú desplegable de la imagen anterior.

Si se utiliza el IDE Arduino en Windows, los nombres de los puertos son los siguientes:



Para los usuarios *de Linux,* por ejemplo, el nombre del puerto es */dev/ttyUSBx*, donde *x*

representa un número entero entre *0* y *9*.

Cómo configurar la Raspberry Pi y Python

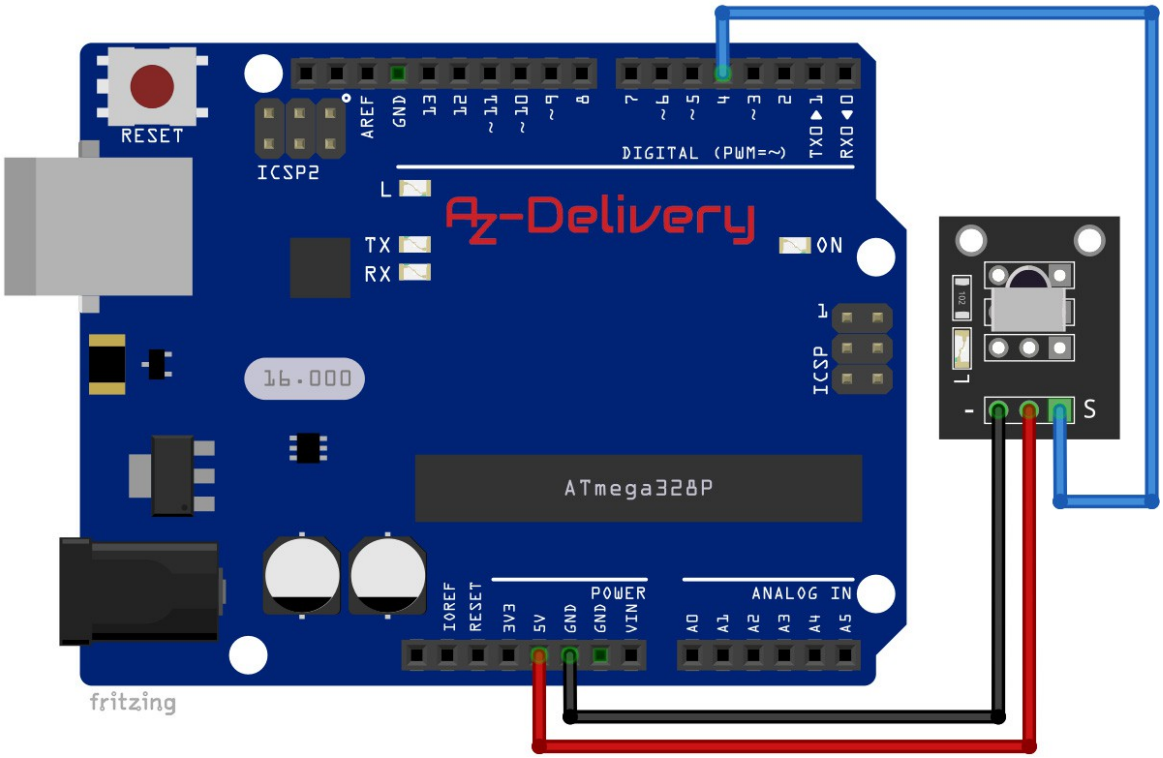
En el caso de la Raspberry Pi, primero hay que instalar el sistema operativo y, a continuación, configurar todo para que pueda utilizarse en el *modo Headless*. El modo *Headless* permite la conexión remota a la Raspberry Pi, sin necesidad de un monitor de pantalla de *PC*, ratón o teclado. Las únicas cosas que se utilizan en este modo son la propia Raspberry Pi, fuente de alimentación y conexión a Internet. Todo esto se explica minuciosamente en el eBook gratuito:

Guía de inicio rápido de Raspberry Pi

El sistema operativo *Raspbian* viene con *Python* preinstalado.

Conexión del módulo con Atmega328p

Conecte el módulo KY-022 con el Atmega328p como se muestra en el siguiente diagrama de conexión:

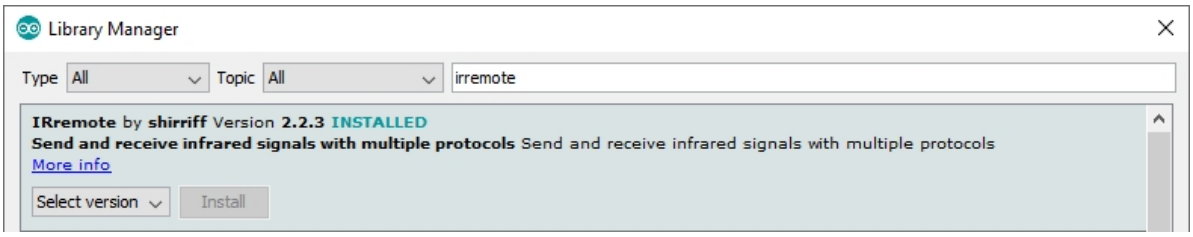


Clavija KY-022	**>**	**Mc pin**	
S	>	D4	**Cable azul**
Pin central (VCC)	>	5V	**Cable rojo**
- (GND)	>	GND	**Cable negro**

Biblioteca para Arduino IDE

Para utilizar el módulo KY-022 con Atmega328p se recomienda instalar una librería externa. Abra Arduino IDE y vaya a *Herramientas > Gestionar Bibliotecas*

Cuando se abra una nueva ventana, escribe *IRremote* (dos R) en la caja de búsqueda e instala una librería llamada *IRremote* hecha por *shirriff*, como se muestra en la siguiente imagen:



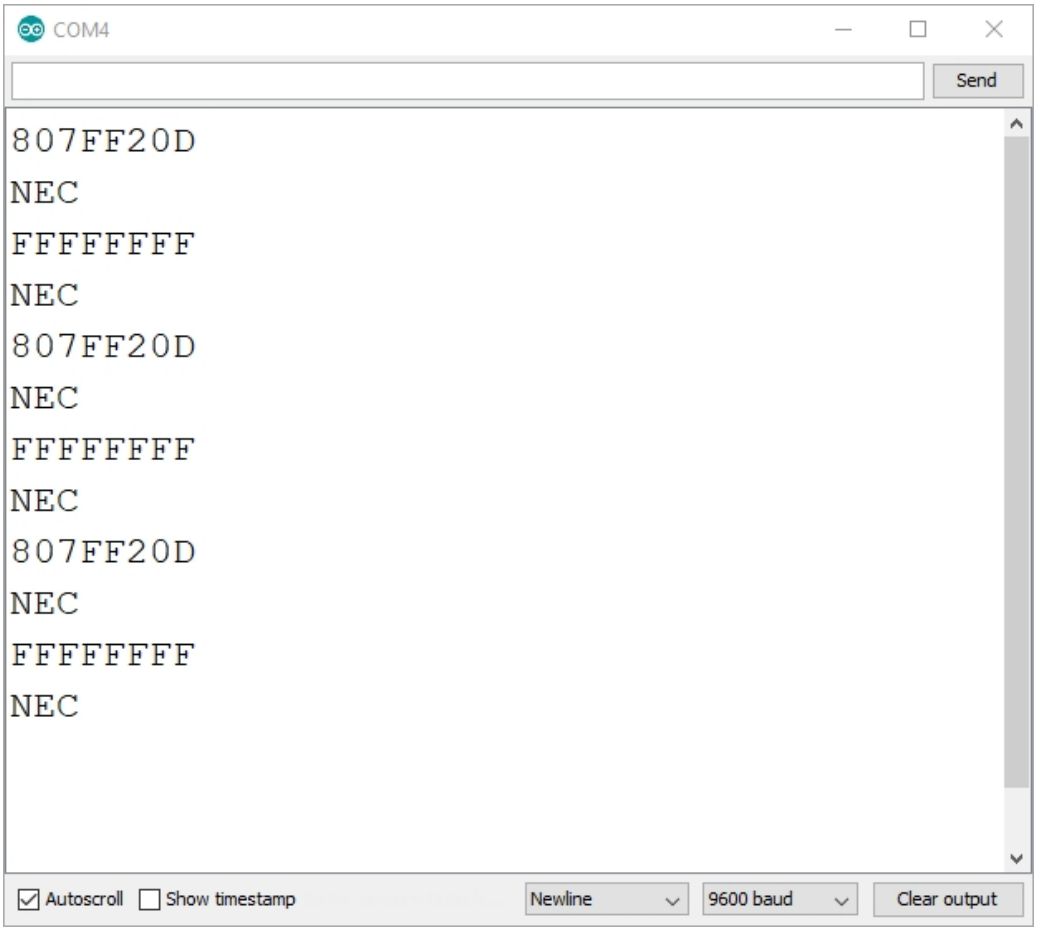
En el siguiente esquema (en la página siguiente) se lee y decodifica la luz infrarroja que proviene del mando a distancia. Un mensaje con la especificación del código del fabricante del control remoto y el código hexadecimal de un botón específico que está siendo presionado es mostrado en el Monitor Serial. De esta forma se puede leer el código hexadecimal específico de la tecla concreta del mando a distancia del televisor y utilizarlo posteriormente (por ejemplo en el eBook para el módulo KY-005).

Código de boceto para detectar el código hexadecimal

```
#include <IRremote.h>
#define SIGNAL_PIN 4
Receptor IRrecv(PIN_SEÑAL);
decode_results output;
void setup(){
  Serial.begin(9600);
  receiver.enableIRIn();
}
void loop() {
  if (receiver.decode(&output)) {
    Serial.println(output.value, HEX);
    switch (output.decode_type) {
      caso 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;
```

```
// dos pestañas
      caso 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;
      por defecto:
      caso UNKNOWN:
        Serial.println("UNKNOWN"); break;
    }
    receiver.resume();
  }
}
```

Sube el sketch al Atmega328p y abre Serial Monitor (*Herramientas > Serial Monitor*). El resultado debería ser como el de la siguiente imagen:



Como puede verse en la imagen superior, se utiliza el mando a distancia con especificación de código *NEC*. Pulse la tecla específica varias veces para obtener su número hexadecimal de código. Este número se utiliza en el siguiente ejemplo.

El número hexadecimal que es diferente del número *0xFFFFFFFF* es el código del botón específico. El mando a distancia emite el número *0xFFFFFFFF* cuando se pulsa y mantiene pulsado el botón específico (una

indicación de que el botón se mantiene pulsado).

Código de boceto para controlar un LED

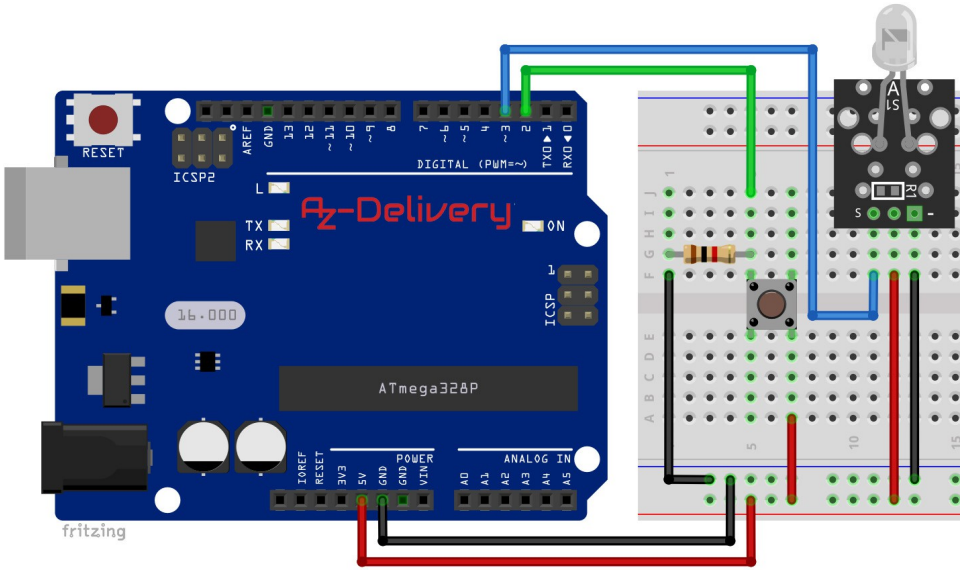
El siguiente sketch se utiliza para alternar el estado del LED (el de a bordo de Atmega328p) cuando se pulsa un botón específico en el mando a distancia.

```
#include <IRremote.h>
#define SIGNAL_PIN = 4;
uint8_t toggle_state = 0;
IRrecv receiver(SIGNAL_PIN);
decode_results output;
void setup(){
  receiver.enableIRIn();
  pinMode(LED_BUILTIN, OUTPUT);
}
void loop(){
  if (receiver.decode(&output)){
    switch(output.value){
      case 0x807FF20D: // código específico de algún botón del mando
        if(toggle_state == 0){
          digitalWrite(LED_BUILTIN, HIGH); toggle_state = 1;
        }
        si no {
          digitalWrite(LED_BUILTIN, LOW); toggle_state = 0;
        }
        romper;
      //caso 0x...: // si desea utilizar más botones
    }
    receiver.resume();
  }
}
```

Carga el sketch en el Atmega328p y pulsa el botón elegido en el mando a distancia. El estado del LED integrado debería cambiar.

Emulación del mando a distancia del televisor con el KY-005

En este capítulo se simula el mando a distancia del televisor utilizando el módulo de LEDs infrarrojos KY-005. Conecte el módulo KY-005 con el Atmega328p como se muestra en el siguiente diagrama de conexión:



Clavija KY-005	>	**Mc pin**	
S	>	D3	**Cable azul**
Pin central (VCC)	>	5V	**Cable rojo**
- (GND)	>	GND	**Cable negro**

Pasador de botón	>	**Mc pin**	
Clavija inferior derecha	>	5V	**Cable rojo**
Pasador superior izquierdo	>	D2	**Cable verde**
Pin superior izquierdo (mediante resistencia*)	>	GND	**Cable negro**

* El valor de la resistencia del resistor es de *1kΩ*

Código Sketch

```
#include <IRremote.h>
#define BUTTON_PIN 2
uint8_t button_state = 0;
uint8_t toggle_state = 0;
IRsend transmisor;
void setup() {
  Serial.begin(9600);
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(BUTTON_PIN, INPUT);
}
void loop() {
  button_state = digitalRead(BUTTON_PIN);
  Serial.println(estado_botón);
  if (estado_botón == HIGH) {
    if (toggle_state == 0){
      transmitter.sendNEC(0x807F00FF, 32);
      digitalWrite(LED_BUILTIN, HIGH);
      toggle_state = 1;
    }
    else { // si mantiene pulsado el botón
      transmitter.sendNEC(0xFFFFFFFF, 32);
    }
  }
  si no {
    digitalWrite(LED_BUILTIN, LOW);
    toggle_state = 0;
  }
  delay(200);
}
```

Al principio del sketch se incluye la librería `IRremote`.

A continuación se crean una macro y dos variables, donde la macro representa el número de pin digital de E/S al que está conectado el pulsador (*2*), y las variables se utilizan para detectar el estado del pulsador.

A continuación, se crea el objeto `transmisor`.

En la función `setup()`, la comunicación serie se inicia con una tasa de baudios de *9600bps*. Los modos de pin para el pin digital *13 se* establece en `OUTPUT` y el modo de pin de botón (*2) se* establece en `INPUT`.

En la función `loop()`, se lee el estado del pulsador y se muestra en el monitor serie. A continuación, se comprueba si el estado del pulsador es *HIGH*. Si lo está, se conmuta el estado de una variable `toggle_state`, se envía el código `NEC` a través de un LED infrarrojo y se *enciende el* LED integrado de la Uno. Si se pulsa el botón durante demasiado tiempo, el estado de la variable `toggle_state` ya no se activa y se envía el código `NEC 0xFFFFFFFF` para simular el comportamiento del mando a distancia del televisor.

Si el estado del pulsador es *BAJO*, el LED integrado de la Uno se enciende. *OFF* y el estado de la variable `toggle_state se pone` a valor cero.

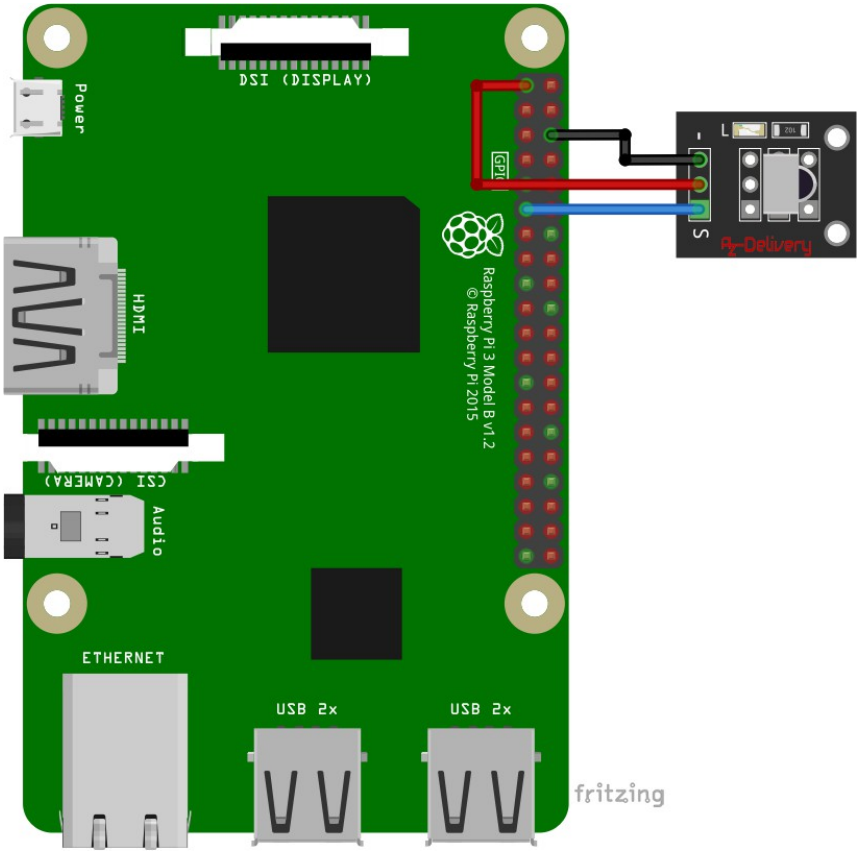
Al final de la función `loop()`, se establece el retardo de *200*

milisegundos.

Con el sketch del capítulo *Emulando el mando a distancia de la TV con el KY-005 se* puede emular el mando a distancia de la TV. Con el sketch del capítulo *Conectando el módulo con Uno se puede emular* el receptor de TV. Combinando estos dos sketches y módulos, se puede crear una interfaz de infrarrojos para enviar y recibir datos en una dirección (del LED de infrarrojos al receptor de infrarrojos).

Conexión del módulo con Raspberry Pi

Conecta el módulo KY-022 con la Raspberry Pi como se muestra en el siguiente diagrama de conexión:



Clavija KY-022	**>**	**Frambuesa Pi pin**		
Pin central (VCC)	>	3V3	[pin 1]	**Cable rojo**
- (GND)	>	GND	[clavija 8]	**Cable negro**
S	>	GPIO17	[clavija 11]	**Cable azul**

Script en Python

```
importar RPi.GPIO como GPIO
from time import time, sleep
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

Pin_señal = 17
GPIO.setup(Pin_señal, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

code = Ninguno
CODES = {
    0xe0e040bf: "ON/OFF",
    0xe0e08877: "0",
    0xe0e020df: "1",
    0xe0e0a05d: "2",
    0xe0e0609f: "3",
    0xe0e010ef: "4",
    0xe0e0906f: "5",
    0xe0e050ad: "6",
    0xe0e030cf: "7",
    0xe0e0b04f: "8",
    0xe0e0708f: "9" # añade más si quieres
}
def binary_aquire(pin, duration):
    # adquiere los datos lo más rápido
    posible t0 = tiempo()
    resultados = []
    while (time() - t0) < duration:
        results.append(GPIO.input(pin))
    devolver resultados
```

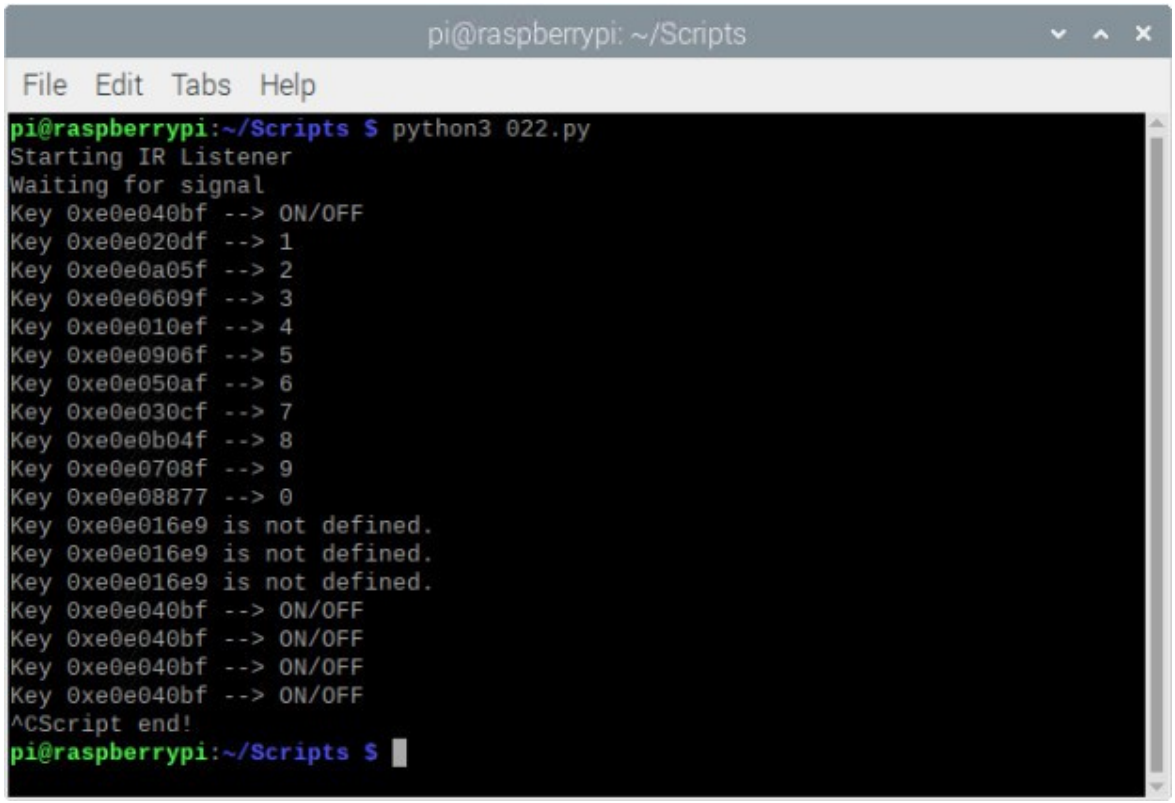
```
def rec(pinNo, bouncetime=150):
    global code, CODES
    data = binary_aquire(pinNo, bouncetime/1000.0) # 150us
    if len(data) < bouncetime:
        devolver
    rate = len(data) / (bouncetime / 1000.0)
    pulses = []
    i_break = 0
    para i en rango(1, len(datos)):
        if (data[i] != data[i-1]) or (i == len(data) - 1):
            pulses.append((data[i-1], int((i-i_break)/rate*1e6)))
            i_break = i

    outbin = ""
    para val, us en pulsos:
        si val != 1:
            continuar
        si outbin y us > 2000: break
        elif us < 1000:
            outbin += "0"
        elif 1000 < us < 2000:
            outbin += "1"
    Inténtalo:
        code = int(outbin, 2)
        print('Clave {} -> {}'.format(str(hex(código)),
        CODES[código]))

    except ValueError:
        code = None

    excepto KeyError:
        print('La clave {} no está
        definida.'.format(str(hex(code))))
```

```
GPIO.add_event_detect(17,GPIO.FALLING,callback=rec,bouncetime=100)

print('[¡Pulsa CTRL + C para finalizar el
script!]') try:
    print('Iniciando IR Listener')
    print('Esperando señal') while
    True:
        sleep(0.000033) # 33 microsegundos

except KeyboardInterrupt: print('\n¡Fin
    del script!')

finalmente:
    GPIO.cleanup()
```

Guarde el script con el nombre *ky022.py*. Para ejecutar el script abre el terminal en el directorio donde está guardado el script y ejecuta el siguiente comando:

python3 ky022.py

El resultado debería parecerse al de la imagen siguiente:



Para detener el script pulse *CTRL + C* en el teclado.

El script comienza con la importación de dos librerías, *RPi.GPIO* y *time*, donde la primera se utiliza para los nombres de GPIO, y la segunda para las funciones de temporización.

A continuación, la denominación GPIO se establece en *BCM* y se desactivan todas las advertencias relacionadas con las interfaces GPIO.

Después de esom el modo del pin GPIO *17* se establece en *INPUT* con resistencia interna PULL DOWN, con la siguiente línea de código:

```
GPIO.setup(Pin_señal, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
```

Se crea la variable llamada *código* donde se almacena el valor del código descodificado recibido del mando a distancia del televisor.

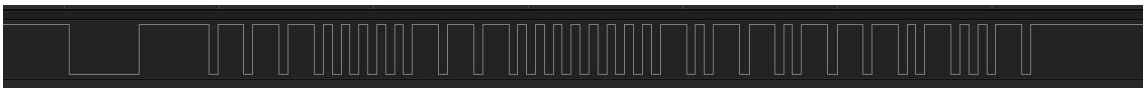
Además, *se* crea el *diccionario CODES*, que es un par clave-valor para códigos específicos. En el *diccionario CÓDIGOS*, las *claves* son nombres para los *valores de* códigos hexadecimales específicos del mando a distancia del televisor

A continuación se crean dos funciones. La primera se llama *binary_aquire()* que tiene dos argumentos, nombre del pin y duración de la grabación. Esta función devuelve una lista llamada *resultados*. La función *binary_aquire() se* encarga de leer los datos del módulo receptor de infrarrojos lo más rápido posible.

La segunda función se llama *rec()* tiene dos argumentos, número de pin y tiempo de rebote (el argumento de tiempo de rebote es opcional). Esta

función no devuelve ningún valor. La función `rec()` es responsable de leer, decodificar y mostrar los datos recibidos.

Dentro de la función *rec()*, se ejecuta la función *binary_aqurie() para* obtener datos del sensor de infrarrojos. La duración de la lectura es en microsegundos porque las señales infrarrojas del transmisor infrarrojo están en el rango de *kHz* (*38kHz*, mencionado al principio de este capítulo). A continuación, los datos recibidos se dividen en impulsos. Para hacerse una idea, mire la siguiente imagen:



La señal recibida tiene el aspecto de varios impulsos con diferentes duraciones (mencionadas al principio de este capítulo). Algunos pulsos representan el "*1" lógico* y otros el "*0"* lógico.

Para obtener lo que es un "*1"* lógico y lo que es un "*0"* lógico, después de la división de los datos en pulsos, se decodifica el código de pulsos. Hacemos esto con las siguientes líneas de código:

```
outbin = ""
para val, us en pulsos:
    si val != 1:
        continuar
    si outbin y us > 2000:
        break
    elif us < 1000:
        outbin += "0"
    elif 1000 < us < 2000:
        outbin += "1"
```

La *outbin es* la variable temporal utilizada para construir un código. La variable *val* representa el estado del pulso, *ALTO* o *BAJO*, y el *us* representa microsegundos. Si el valor de la *variable val* es distinto de *HIGH* (lógico "*1*"), lo que significa que el pulso está en estado *LOW*, no es un punto de interés para el algoritmo de descodificación, por lo que se descarta. A continuación, se comprueba la duración del pulso en estado *ALTO*. Si la duración del pulso es superior a *2000* microsegundos, significa que el pulso no es "*1*" o "*0*" lógico, por lo que se descarta. Si el pulso es más corto que *1000 microsegundos ese pulso* es "*0*" *lógico*. Si el pulso dura de *1000* a *2000 microsegundos ese pulso es* "*1*" *lógico*.

Al final de la función *rec()*, el código se convierte en un número hexadecimal y se muestra en el terminal. La visualización está en el *bloque try-except* porque el código se lee del diccionario *CODES*, lo que a veces no puede hacerse (cuando no hay una *clave* específica en el *diccionario*). Para evitar este error, esta parte del código tiene que estar en el *bloque* de código try-except. Si no hay un código definido en el diccionario, Python genera un *KeyError*.

Después de crear las funciones se establece la rutina de interrupción. Cada vez que hay un flanco descendente de la señal digital en el pin GPIO *17*, se ejecuta la función *rec()*.

Al final del script se ejecuta el bloque de código try-except-finally. En el bloque de código try se crea el bucle indefinido (*while True:*) para mantener el script en ejecución.

Para finalizar el script pulsa *CTRL + C* en el teclado. Esto se llama interrupción de teclado, y cuando esta interrupción ocurre, el bloque de código *except* es ejecutado (except KeyboardInterrupt:) mostrando el mensaje *Script end!* en la terminal.

El bloque de código *finally* se ejecuta al finalizar el script. Se utiliza para deshabilitar cualquier modo e interfaz de pin GPIO utilizado previamente ejecutando la función llamada cleanup().

Ahora es el momento de aprender y hacer tus propios proyectos. Puedes hacerlo con la ayuda de muchos scripts de ejemplo y otros tutoriales, que puedes encontrar en internet.

Si busca microelectrónica y accesorios de alta calidad, AZ-Delivery Vertriebs GmbH es la empresa adecuada. Dispondrá de numerosos ejemplos de aplicación, guías de instalación completas, libros electrónicos, bibliotecas y asistencia de nuestros expertos técnicos.

https://az-delivery.de

¡Diviértete!

Impresionante

https://az-delivery.de/pages/about-us