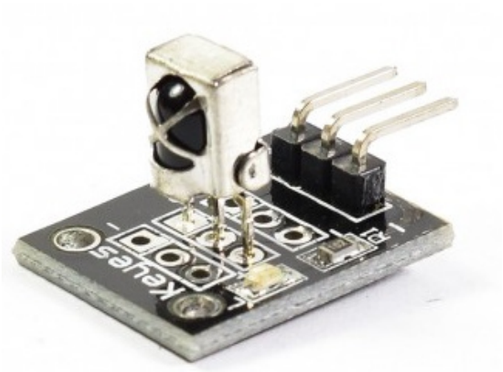
Tutorial Arduino y control remoto Infrarrojo

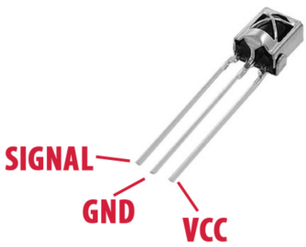
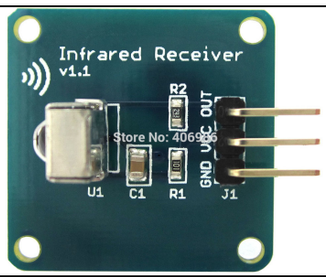
(versión 1-9-17)

En este tutorial usaremos un módulo sensor infrarrojo para recibir la señal de controles remotos IR que usan muchos de los equipos domésticos como TVs, equipos de sonidos, etc. A través de estos controlaremos las salidas de nuestro Arduino.
Para este tutorial usaremos el siguiente sensor de Infrarrojos:

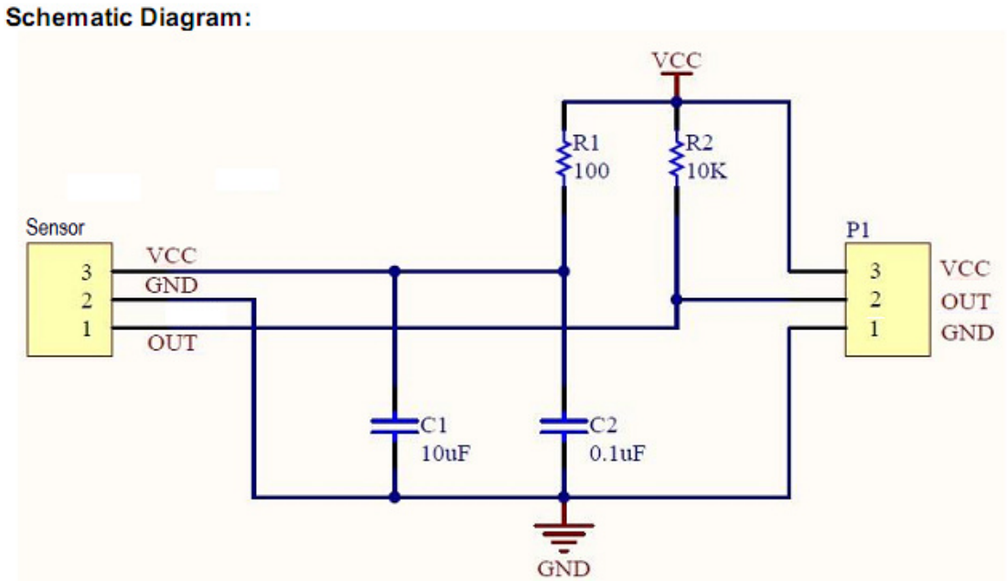


Este sensor tiene un filtro interno para detectar solo frecuencias infrarrojos cercanas a 38KHz, lo que lo hace compatible con la mayoría de mandos infrarrojos, posee 3 pines de conexión GND, VCC y DATA , el cual nos permite conectar directamente a un pin digital de nuestro Arduino o cualquier microcontrolador que deseemos usar.

NOTA:

Es posible que en el KIT que se tiene disponible, no venga el modulo sino solo el sensor. En este caso el usuario deberá armar la interfase.		

La interfase es la siguiente:



Ensayaremos el de la figura, el cual viene en algunos kit de Arduino.

Este mando (*según fuente consultada – de no ser así, la idea es similar*) usa el protocolo NEC que trabaja a 38KHz de frecuencia, el formato del tren de pulsos que envía al presionar una tecla se muestra en la siguiente gráfica:

Lo particular de este protocolo es que envía doble vez tanto la dirección como el comando, de forma normal y negado, con esto posteriormente se puede validar los datos.

La dirección está asociada a un dispositivo, por ejemplo un TV, una equipo de sonido, un VCR, etc. Y el comando está asociado a la acción o función del botón. Por ejemplo subir el volumen, apagar, el número 1 o 2, etc.

Para este tutorial vamos a trabajar como si se tratase de un solo bloque de datos 32 bits.
A continuación se muestra los valores de los datos correspondientes a los botones del control en mención:

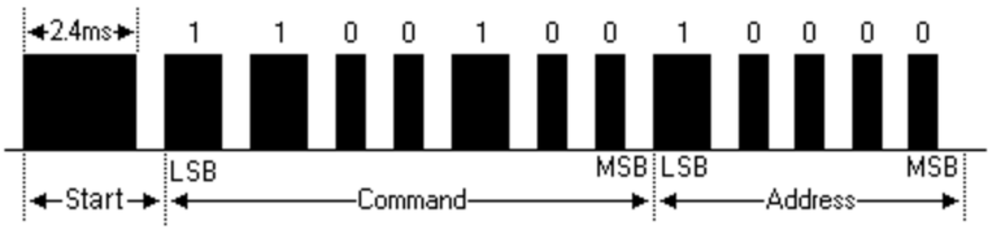
Botón	Dirección (HEX)	comando (HEX)	Dato 32 bits (HEX)
OK	**0x00**	**0x02**	**0x00FF02FD**
Arriba (↑)	**0x00**	**0x62**	**0x00FF629D**
Abajo (↓)	**0x00**	**0XA8**	**0x00FFA857**
Izquierda (←)	**0x00**	**0x22**	**0x00FF22DD**
Derecha (→)	**0x00**	**0xC2**	**0x00FFC23D**
1	**0x00**	**0x68**	**0x00FF6897**
2	**0x00**	**0x98**	**0x00FF9867**
3	**0x00**	**0xB0**	**0x00FFB04F**
4	**0x00**	**0x30**	**0x00FF30CF**
5	**0x00**	**0x18**	**0x00FF18E7**
6	**0x00**	**0x7A**	**0x00FF7A85**
7	**0x00**	**0x10**	**0x00FF10EF**
8	**0x00**	**0x38**	**0x00FF38C7**
9	**0x00**	**0x5A**	**0x00FF5AA5**
0	**0x00**	**0x4A**	**0x00FF4AB5**
*****	**0x00**	**0x42**	**0x00FF42BD**
#	**0x00**	**0x52**	**0x00FF52AD**

**Como se observa el dato está formado por la dirección, comando y sus negaciones, por ejemplo para la tecla OK: el dato de 32bits es 0x00FF02FD, donde la dirección es 00 y su negación FF, y el comando o función es 02 y su negación FD.*

Control Remoto de TV

Tambien podemos realizar ensayos con otros mandos de TV, por ejemplo un control remoto de un TV SONY

EL protocolo SONY trabaja con una frecuencia de 40KHz, en la siguiente figura se muestra los pulsos que se envían cuando se presiona una botón
Trabaja con 12 bits de datos , de los cuales 5bits son para la direccion y 7 bits para comando o función.
Existen variaciones de 15 bits y 20 bit pero en todos los bits de comando son de 7 bits.



A continuación se muestra algunos de los datos correspondientes al protocolo SONY (*según fuente consultada*):

Botón	**Dirección (DEC)**	**Comando (DEC)**	**Dato 12bits (Hex)**
Power	1	21	0xA90
Menú	1	96	0X070
Arriba	1	116	0x2F0
Abajo	1	117	0xAF0
izquierda	1	52	0x2D0
Derecha	1	51	0xCD0
1	1	0	0x010
2	1	1	0x810
3	1	2	0x410
4	1	3	0xC10
5	1	4	0x210
6	1	5	0xA10
7	1	6	0x610
8	1	7	0xE10
9	1	8	0x110
0	1	9	0x910

**Para convertir el dato de 12 bits en su dirección y comando se toma empezando desde el bit menos significativo considerándolo de mayor peso.*

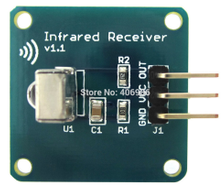
Explicado lo anterior Implementemos ambos controles en Arduino.

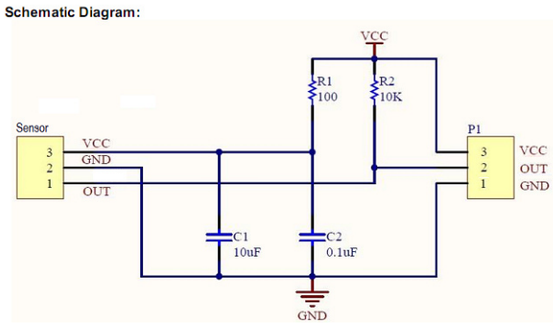
Librería de control infrarrojo para Arduino

Hay distintas versiones de librerías Arduino para trabajar con mando infrarrojo, incluso tiene implementado varios protocolos de las marcas más conocidas como Sony, LG, Samsung, Sanyo, etc, es decir los reconocen de acuerdo a su función.
Es necesario descargar e importarla a nuestro IDE Arduino para poder trabajar los ejemplos de este tutorial. Usaremos la **Arduino-IRremote-master.zip** se incluye una copia en el Tutor de Arduino, Carpeta *Bibliotecas.*

Conexiones entre Arduino y modulo receptor IR

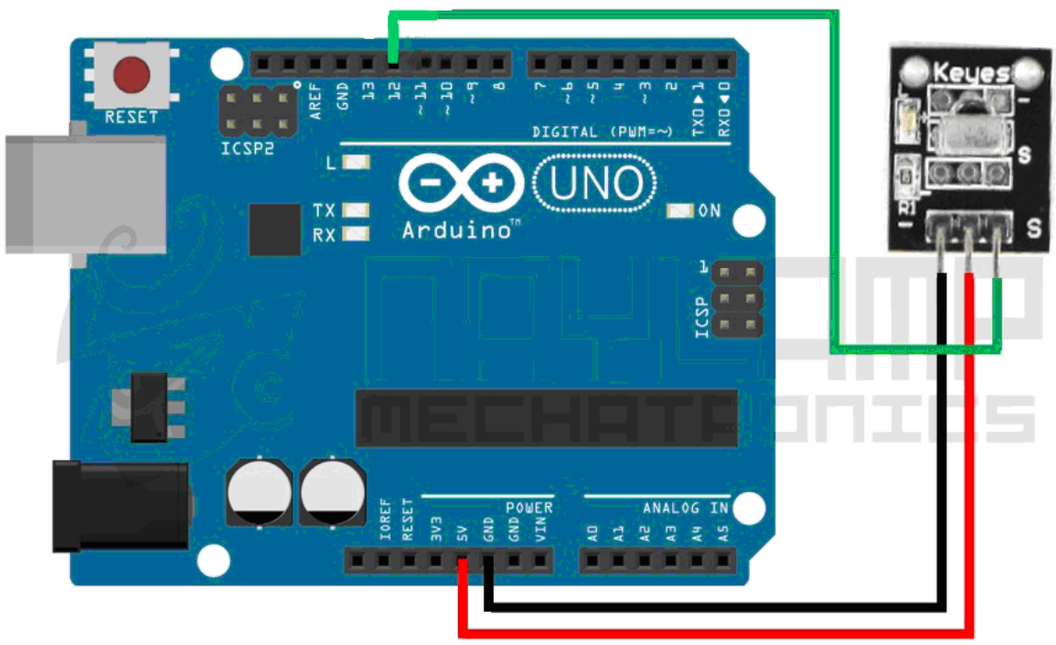
Las conexiones son simples el sensor tiene un pin VCC el cual se alimenta con 5V un pin GND y un pin de DATA, que es una salida digital el cual conectaremos al pin 11 del Arduino.





Recuerde que si el sensor que cuenta en su kit no incluye la interfaz en el modulo Keyes, entonces deberá implementarla como mencionamos antes.

En la siguiente figura se muestra el conexionado del caso de tener modulo Keyes.
En particular la señal que proviene del sensor que es del tipo digital, se toma por el PIN12, sin embargo se le puede puede setear otro, por ejemplo.



Empecemos con algunos ejemplos:

Encendiendo un led con nuestro control Remoto

En este ejemplo se encenderá y apagará el led del pin 5. Con la tecla “0” del control remoto elegido se apagará el LED con la tecla “1” se encenderá.

El código es el siguiente:

```
//Encendido de un LED
//Tecla 0 Apaga - Tecla 1 lo enciende
//Para infrarrojos-----------
#include <boarddefs.h>
#include <IRremote.h>
#include <IRremoteInt.h>
#include <ir_Lego_PF_BitStreamEncoder.h>
//Fin lista bibliotecas------

int receptor = 12;//Define PIN recepcion de infrarrojo

int led = 5; //Define el PIN de LED testigo

IRrecv irrecv(receptor);
decode_results codigo; //OBJETO CODIGO DE CLASE decode_result, oriundo de IRremote.h

void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // INICIA LA RECEPCIÓN
  pinMode(led, OUTPUT);
}

void loop()
{
 if (irrecv.decode(&codigo))
 {
Serial.println(codigo.value, DEC);//Muestra por MONITOR SERIAL lo que recibe
                                  //en sistema DECIMAL

      if (codigo.value==3238126971) //CÓDIGO DEL NÚMERO CERO PARA ACTIVAR LED
      {
         digitalWrite(led,LOW);//Apaga LED
      }

      if (codigo.value==2534850111)//CÓDIGO DEL NÚMERO UNO PARA DESACTIVAR LED
      {
         digitalWrite(led,HIGH); //Apaga LED
      }

delay(500);
irrecv.resume();

 }
}
```

Expliquemos un poco el código:

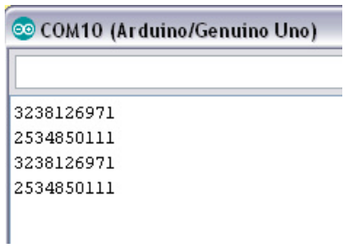
Con **IRrecv irrecv(receptor)** informamos que el receptor IR envía a Arduino por el PIN especificado por la variable entera **receptor** (en este caso 12) , luego creamos la variable **codigo**, que es una estructura en donde se guardaran todos los datos relacionados cuando se recibe un dato por sensor.

En Setup() inicializamos la recepción de datos con **irrecv.enableIRIn()** y configuramos el PIN 5 como salida para el LED testigo **pinMode(led, OUTPUT)** , ya que la variable entera **led** la cargamos con 5.

Esto es trabajar con variables para definir los PINES es útil cuando deseamos luego cambiar los pines utilizados en futuros programas.

En el void loop() simplemente comprobamos si le llega un dato al receptor, esto lo hacemos con if(irrecv.decode(&results)), si hay un dato y es **3238126971** o **2534850111** apagamos o encendemos el LED.

Estos son los códigos en decimal que envía el control utilizado en este ejemplo. Como podemos observar en el Monitor Serie de Arduino.
3238126971 apaga LED
2534850111 enciende LED

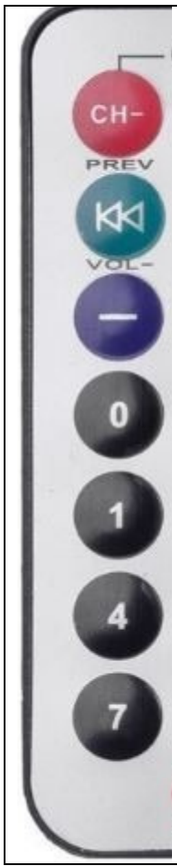


CONTROL UTILIZADO

Decodificando datos de los controles infrarrojos

Podríamos usar cualquier control remoto para nuestros ejemplos, pero debemos decodificarlos. El programa anterior permite ver por el MONITOR SERIAL lo que llega al sensor, entonces podemos utilizarlo para obtener los codigos asociados a cada tecla.

A continuación se muestra los datos recibidos al presionar las teclas del control remoto que viene con el kit del sensor con el que se hicieron los ensayos, tanto en HEXA como en DECIMAL.

	**E318261B** **3810010651** **52A3D41F** **1386468383** **F076C13B** **4034314555** **C101E57B** **3238126971** **716BE3F** **2534850111** **8C22657B** **2351064443** **32C6FDF7** **851901943**		**511DBB** **5316027** **D7E84B1B** **3622325019** **A3C8EDDB** **2747854299** **97483BFB** **2538093563** **3D9AE3F7** **1033561079** **488F3CBB** **1217346747** **1BC0157B** **465573243**		**EE886D7F** **4001918335** **20FE4DBB** **553536955** **E5CFBD7F** **3855596927** **F0C41643** **4039382595** **6182021B** **1635910171** **449E79F** **71952287** **3EC3FC1B** **1053031451**

También podríamos utilizar le siguiente programa que dado que la biblioteca empleada es capaz de reconocer algunos códigos de algunas marcas, nos podría ser útil.

Biblioteca necesaria: (Buscar en Internet o dentro de carpeta Bibliotecas del Tutor de Arduino.

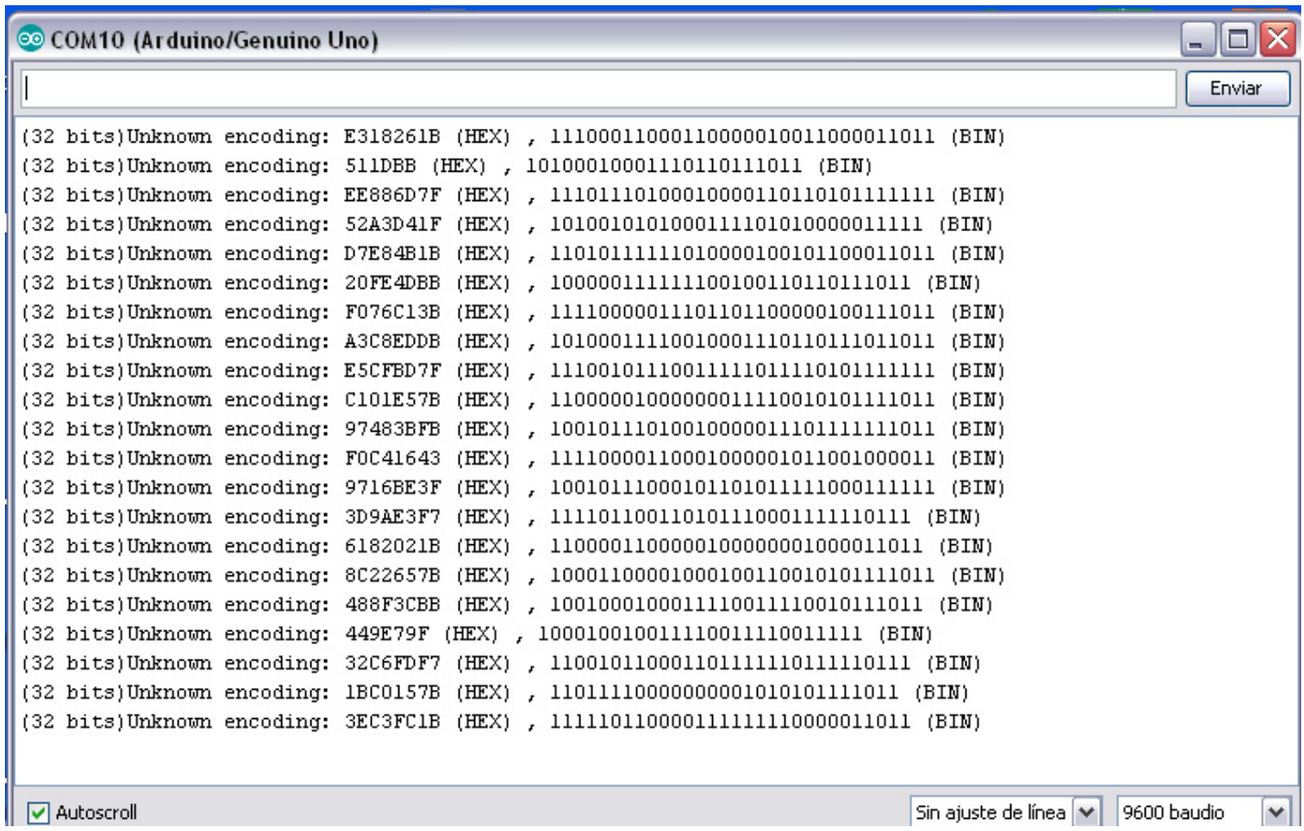
Arduino-IRremote-master.zip

Como ya mencionamos en esta biblioteca figuran algunos codigos conocidos de ciertas marcas, el programa es capas de detectarlos y lo informa NEC, SONY, RC5, RC6, RPANASONIC etc.

```
#include <boarddefs.h>
#include <IRremote.h>
#include <IRremoteInt.h>
#include <ir_Lego_PF_BitStreamEncoder.h>

int RECV_PIN = 11;
IRrecv irrecv(RECV_PIN);
 decode_results results;

void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Empezamos la recepción
  pinMode(5, OUTPUT); // configura ‘pin’ como salida, para visualizar
                      //llegada de marca desconocida
 }

void dump(decode_results *results) {
  // Dumps out the decode_results structure.
  // Call this after IRrecv::decode()

  Serial.print("(");
  Serial.print(results->bits, DEC);
  Serial.print(" bits)");

  if (results->decode_type == UNKNOWN) {
    Serial.print("Unknown encoding: ");

     digitalWrite(5,HIGH );//Pulso de llegada desconocido
     delay(100);
     digitalWrite(5,LOW );//Termina pulso

  }
  else if (results->decode_type == NEC) {
    Serial.print("Decoded NEC: ");

  }
  else if (results->decode_type == SONY) {
    Serial.print("Decoded SONY: ");
  }
  else if (results->decode_type == RC5) {

    Serial.print("Decoded RC5: ");
  }
  else if (results->decode_type == RC6) {
    Serial.print("Decoded RC6: ");
  }
  else if (results->decode_type == PANASONIC) {
    Serial.print("Decoded PANASONIC - Address: ");
    Serial.print(results->address, HEX);
    Serial.print(" Value: ");
  }
  else if (results->decode_type == LG) {
    Serial.print("Decoded LG ");
  }
  else if (results->decode_type == JVC) {
    Serial.print("Decoded JVC ");
  }
  else if (results->decode_type == AIWA_RC_T501) {
    Serial.print("Decoded AIWA RC T501 ");
```

```
  }
  else if (results->decode_type == WHYNTER) {
    Serial.print("Decoded Whynter ");
  }
  Serial.print(results->value, HEX);
  Serial.print(" (HEX) , ");
  Serial.print(results->value, BIN);
  Serial.println(" (BIN)");

}

void loop() {
  if (irrecv.decode(&results)) {
    dump(&results);
    irrecv.resume(); // empezamos una nueva recepción
  }
  delay(300);
}
```

Este es el Programa Remoto5 (del Tutor de Arduino)

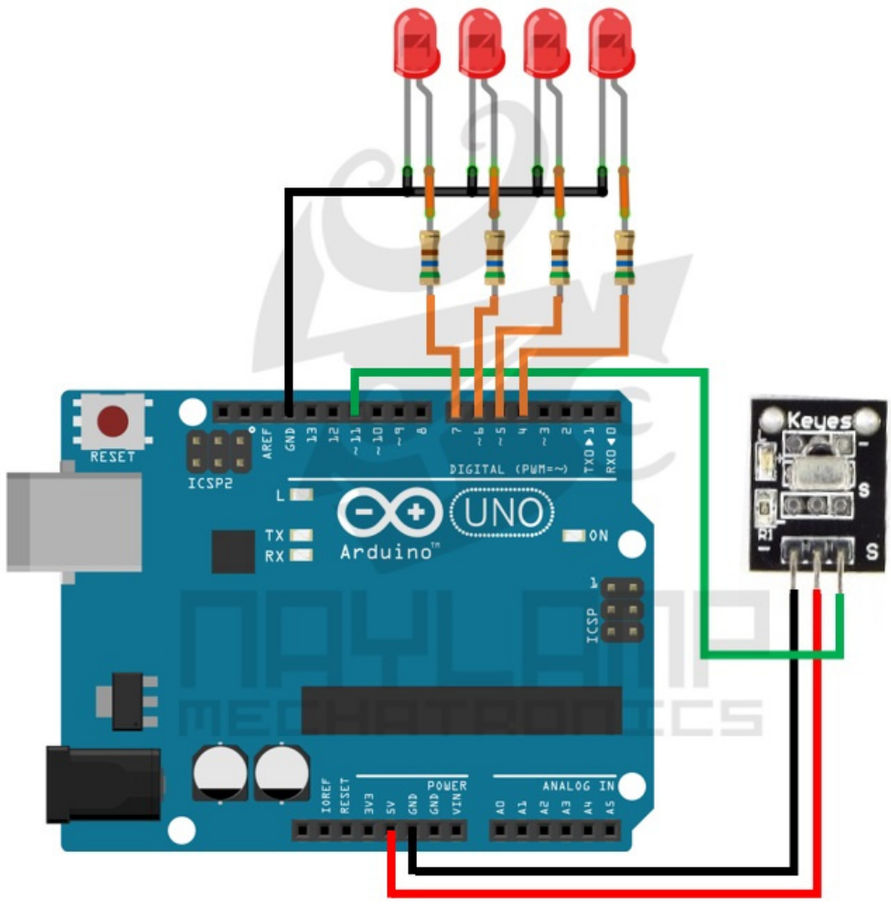


Controlar varios Pines digitales con control remoto por infrarrojos

Como caso relacionado, se deja para el lector hacer una adaptación de este programa y que se se controlen 3 LEDs.

O sea ahora que ya sabemos el valor del dato que corresponde a cada tecla, vamos a asociar una tecla a un pin digital y prender o pagar leds, que se podrían remplazar por cualquier otro actuador.

Por ejemplo en la figura siguiente se pretende actuar sobre 3 LEDs.



Comando Infrarrojo
Características Técnicas:

Contro Remoto:	
Modelo	I0526017
Material	Plástico
Características:	
	21 botones de función
Control Rango:	por encima de los 8M;
Longitud de onda infrarroja:	940nm;
Oscilador de cristal:	455KHz,
Frecuencia portadora:	38 kHz
Ángulo de uso:	60 grados
Adherencia Material:	0.125mmPET,
Tiempo de Vida:	20.000 veces.
Corriente estática:	3 ~ 5uA,
Corriente dinámica:	3 ~ 5 mA

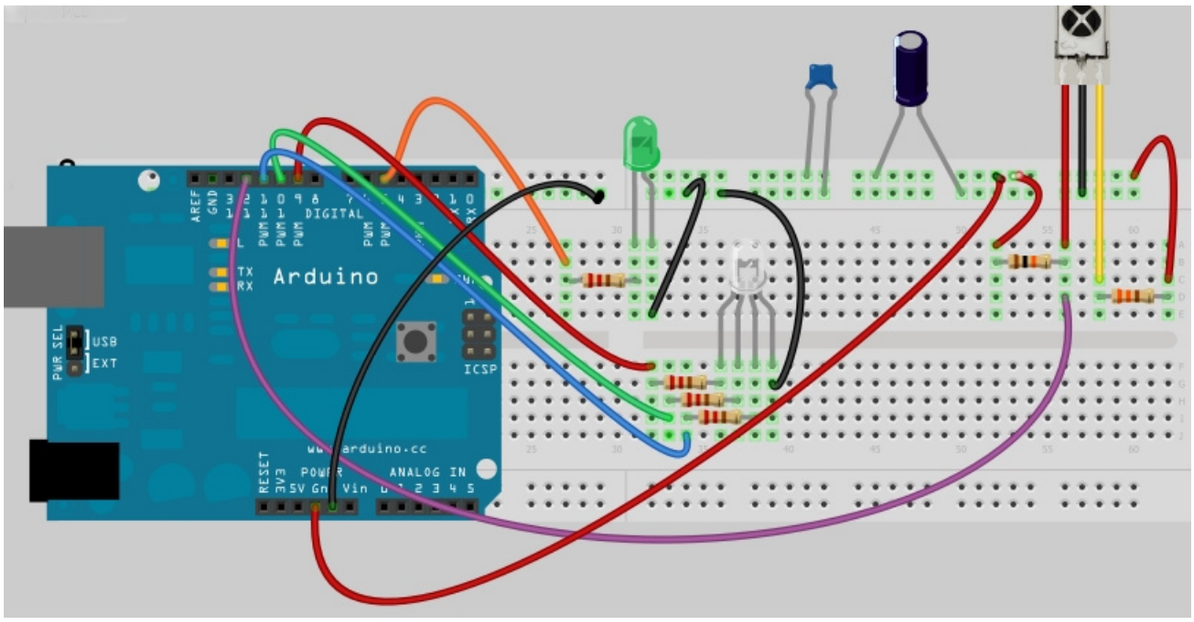
Receptor Infrarrojo:	
Modelo:	Vs1838B
Working Voltage:	2.7V to 5.5V
Reception Distance:	18M
Reception Angle:	45 Degree
Low Level Voltage:	0.4V
High Level Voltage:	4.5V
Body Size:	7 x 7 x 5mm / 0.27" x 0.27" x 0.2"(L*W*T)
Pin Length:	22.5mm / 0.88"
Pitch:	2mm / 0.08"
Material:	Plastic, Alloy

Controlar Pines digitales y analogicos (PWM) con control remoto por infrarrojos TACHOS LEDs

Este ejemplo de programa se hace uso del control infrarrojo para ejecutar ciertas acciones en un LED RGB cátodo comun(que representa un TACHO LED).

Perol no solamente se puede comandar por mandos infrarrojo sino además por media de la PC conectada a Arduino y por Bluetooth. En un apunte aparte se podrá encontrar el desarrollo completo.

Circuito empleado para control remoto infrarrojo



Programa (Remoto13)

```
//FUNCIONO

//En esta version TARRGB6. Se combinara con Remoto Infrarrojo

//Avances de Remoto10

//SE CAMBIO EL PIN 11 POR EL 6 YA QUE TENIA PROBLEMAS

//PARA EL CONTROL DEL RGB

//Para tachos----------

//Declaro variables globales

int red=0,green=0,blue=0;

int num;

//Fin de tachos----------

//Para infrarrojos----------

#include <boarddefs.h>

#include <IRremote.h>
```

```
#include <IRremoteInt.h>

#include <ir_Lego_PF_BitStreamEncoder.h>

int RECV_PIN = 12; //Pin que recibe el envio infrarrojo

 IRrecv irrecv(RECV_PIN);

 decode_results results; //results contiene el valor llegado

//Fin para infrarrojos-------

void setup()

{

   pinMode(5, OUTPUT); // configura ‘pin’ como salida, para visualizar

                       //llegada de desconocido

  pinMode(12, INPUT);//Configura PIN12 como entrada

  Serial.begin(9600);   //Iniciamos comunicación con el puerto serie

//Para tachos-------

//Paneo de canales---

  analogWrite(9, 255);

  delay(1000);

  analogWrite(9, 0);

  analogWrite(10, 255);

  delay(1000);

  analogWrite(10, 0);

  analogWrite(6, 255);

  delay(1000);

  analogWrite(6, 0);

//Fin paneo de colores---

//Fin para tachos-----

//Para infrarrojos---

  irrecv.enableIRIn(); // Empezamos la recepción

 //Fin para infrarrojos-----

 }

//Funcion  que muestra resultado en el Serie Monitor

void dump(decode_results *results) {

  // Dumps out the decode_results structure.
```

```
  // Call this after IRrecv::decode()


 // Serial.print("(");

  //Serial.print(results->bits, DEC);

  //Serial.print(" bits)");


  if (results->decode_type == UNKNOWN) {

    //Serial.print("Unknown encoding: ");


      digitalWrite(5,HIGH );//Pulso de llegada desconocido

      delay(100);

      digitalWrite(5,LOW );//Termina pulso

  }


  //Serial.print(results->value, HEX);

  //Serial.print(" (HEX) , ");

  //Serial.print(results->value, BIN);

 // Serial.println(" (BIN)");

  }

//Fin de funcion que muestra resultado-----------

 void loop() {

  //Para infrarrojos----------

  if (irrecv.decode(&results)) {

   dump(&results);

    //irrecv.resume(); // empezamos una nueva recepción

    //Serial.print("LEIDO");

    //Serial.print(results.value, HEX);

   Serial.println(results.value);

   switch(results.value)

      {

        case 3238126971://Tecla 0
```

```
      //Serial.print(results.value);
       SecuenciaA1();
       break;
        case 3810010651://Tecla ch-  Rojo
      //Serial.print(results.value);

       Color(255,0,0);
       break;

      case 5316027://Tecla ch  -Verde

      //Serial.print(results.value);

       Color(0,255,0);
       break;
      case 4001918335://Tecla ch+ Azul
      //Serial.print(results.value);
      Color(0,0,255);
       break;
   case 3622325019://Tecla NEXT-Negro
      //Serial.print(results.value);
     Color(0,0,0);
       break;

  case 553536955://Tecla Play Pause - Lila
      //Serial.print(results.value);
      Color(255,0,255);
       break;

    case 1386468383://Tecla Play Pause -Amarillo
      //Serial.print(results.value);
      Color(255,255,0);
```

```
        break;
     case 3855596927://Tecla EQ -Blanco
       //Serial.print(results.value);
       Color(255,255,255);
        break;
   case 2534850111://Tecla 0
       //Serial.print(results.value);
        SecuenciaA2();
        break;
      }
  irrecv.resume(); // empezamos una nueva recepción
      }
  //delay(300); //retardo de infrarrojo
//Fin para infrarrojos----------
//Relacion infrarrojos tachos---
//Fin relacion infrarrojos tachos---
//Para tachos--------------------
 /*
   * Evaluamos el momento en el cual recibimos un caracter
   * a través del puerto serie
   */
  if (Serial.available()>0) {


  //Delay para favorecer la lectura de caracteres
    delay(22);
    //Se crea una variable que servirá como buffer
    String bufferString = "";
    /*
     * Se le indica a Arduino que mientras haya datos
     * disponibles para ser leídos en el puerto serie
     * se mantenga concatenando los caracteres en la
     * variable bufferString
```

```
    */
   while (Serial.available()>0) {
     bufferString += (char)Serial.read();
   }
   //Se transforma el buffer a un número entero
   //int num = bufferString.toInt();
   num = bufferString.toInt();
   //int red=0,green=0,blue=0;
//Colores fijos
    if (num==700)
          {
           analogWrite(9, 0) ;   // Rojo  PIN 9
           analogWrite(10, 0) ;   // Green - Verde  PIN 10
           analogWrite(6, 0) ;   // blue - blue  PIN 11
          red= 0;
          green= 0;
          blue= 0;
          }

   if (num==750)  //Rojo
          {

           analogWrite(9, 255) ;   // Rojo  PIN 9
           analogWrite(10, 0) ;   // Green - Verde  PIN 10
           analogWrite(6, 0) ;   // blue - blue  PIN 11
          red= 255;
          green= 0;
          blue= 0;
          }

   if (num==800) //Verde
          {
```

```
            analogWrite(9, 0) ;   // Rojo  PIN 9
            analogWrite(10, 255) ;   // Green - Verde  PIN 10
            analogWrite(6, 0) ;   // blue - blue  PIN 11
           red= 0;
           green= 255;
           blue= 0;
           }
  if (num==850)  //blue
           {
            analogWrite(9, 0) ;   // Rojo  PIN 9
            analogWrite(10, 0) ;   // Green - Verde  PIN 10
            analogWrite(6, 255) ;   // blue - blue  PIN 11
           red= 0;
           green= 0;
           blue= 255;
           }
 if (num==900)  //Blanco
           {

            analogWrite(9, 255) ;   // Rojo  PIN 9
            analogWrite(10, 255) ;   // Green - Verde  PIN 10
            analogWrite(6, 255) ;   // blue - blue  PIN 11
           red= 255;
           green= 255;
           blue= 255;
           }

//Secuencias en programa del Arduino---
 if (num==130)
           {
            SecuenciaA1();
```

```
            }

//--------------------------------------------

//Se ordena el encendido de los LEDs por pasos

//Va de 0 a 255 con pasos de 5

// Subiendo intensidad----------

           if (num==111)

            {

             Mas_red();

            }


         if (num ==222)

            {

             Mas_green();

            }

          if (num ==333)

            {

            //Serial.print(num);

            //Serial.print(blue);

            Mas_blue();

            }

//Fin subiendo intensidad----------


//---   Bajando intensidad de a pasos----------

         if (num==444)

            {

            Menos_red();

            }


         if (num ==555)

            {

            Menos_green();

            }
```

```
        if (num ==666)
          {
          Menos_blue();
          }
//Fin bajando intensidad de a pasos----------------
//-------------------------------------------------
  }
 }
//*************************************************
//Funciones creadas----------------------
void Menos_blue()
     {
          blue = blue - 5;
          if (blue == -5)
           {
            blue = blue + 5;
           }
          analogWrite(6, blue) ;// blue -blue  PIN 11
      }
//--------------------
void Menos_green()
     {
          green= green - 5;
          if (green == -5)
           {
           green = green + 5;
           }
           analogWrite(10, green) ;   // Green - Verde  PIN 10
      }
//--------------------
void Menos_red()
     {
```

```
            red= red - 5;
            if (red == -5)
            {
            red= red +5;
            }
            analogWrite(9, red) ;   // Rojo  PIN 9
     }
//-----------------------
void Mas_blue()
    {
     //Serial.print(num);

     blue = blue + 5;
     Serial.print(blue);
     if (blue > 255)
      {
     blue = blue -5;
      }
     analogWrite(6, blue);
     }

//--------------------
void Mas_green()
      {
            green= green + 5;
            if (green > 255)
            {
            green = green - 5;
            }
            analogWrite(10, green) ;   // Green - Verde  PIN 10
     }
//--------------------
```

```
void Mas_red()
    {
          red= red +5;
          if (red > 255)
          {
          red= red-5;
          }
          analogWrite(9, red) ;   // Rojo  PIN 9
    }
//------------------------------
void Color(int R, int G, int B)
   {
        analogWrite(9 , R) ;   // Rojo
        analogWrite(10, G) ;   // Green - Verde
        analogWrite(6, B) ;   // blue - blue
   }
//-------------------------
void SecuenciaA1()
  {
    for (int i =0 ; i<255 ; i++)
         {
          Mas_red();//Sube a maximo rojo
          delay(20);
          Color(0, 0, 0); //negro
          delay(25);
          }
     Color(0, 0, 0); //negro

    for (int i =0 ; i<255 ; i++)
         {
          Mas_green();//Sube a maximo verde
          delay(20);
```

```
          Color(0, 0, 0); //negro
          delay(25);
          }
        Color(0, 0, 0); //negro

     for (int i =0 ; i<255 ; i++)
         {

          Mas_blue();//Sube a maximo blue
          delay(20);
          Color(0, 0, 0); //negro
          delay(25);
          }
for (int i =0 ; i<25 ; i++)
         {
        Color(0, 0, 0); //negro
          delay (100);
        Color(255, 0, 0);
          delay (100);
        Color(0, 0, 255);
          delay (300);
         Color(0, 255, 0);
          delay (100);
         Color(0, 0, 0); //negro
          delay (200);
         Color(0, 255, 0);
          delay (100);
         Color(0, 0, 0); //negro
        }
   }

void SecuenciaA2()
```

```
  {

for (int i =0 ; i<25 ; i++)

         {

       Color(255, 0, 255); //negro

          delay (100);

       Color(0, 255, 0);

          delay (100);

       Color(255, 0, 0);

          delay (100);

        Color(255, 255, 0);

          delay (100);

        Color(255, 255, 255); //negro

         delay (200);

        Color(255, 0, 0);

          delay (100);

        Color(0, 0, 0); //negro

       }

  }
```

FUENTE consultada:

http://www.naylampmechatronics.com/blog/36_Tutorial-Arduino-y-control-remoto-Infrarrojo.html