

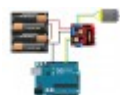


S/. 0,00

CATEGORÍAS

[All Blog News](#)[Tutorial Arduino y control remoto Infrarrojo](#)

ARTÍCULOS POPULARES

[Tutorial transmisor de celda de carga HX711, Balanza Digital ago 23, 2015](#)[Tutorial de Uso del Módulo L298N feb 19, 2015](#)[Tutorial Módulo GPS con Arduino jun 25, 2015](#)

ARTÍCULOS RECIENTES

[Tutorial Sensor de Distancia SHARP sep 28, 2016](#)[Arduino y ESP8266 como cliente web ago 24, 2016](#)[Robot móvil controlado por bluetooth ago 06, 2016](#)

PROMOCIONES ESPECIALES



Módulo Mini Step Down DC-DC
Reduce el voltaje de la forma más...

S/. 6,00 ~~S/. 8,00~~

Todas las promociones especiales >

ETIQUETAS

[sensor](#)[cnc](#)[robot](#)[Servo](#)[LED](#)[LCD](#)[driver](#)[infrarrojo](#)[3d](#)[voltmetro](#)

NOVEDADES



Display touch Raspberry Pi 7" (DSI) - Oficial
Pantalla Oficial LCD Touch de 7 pulgadas

S/. 470,00



Soporte guía lineal circular 8mm SK8
Soporte para montaje en superficies paralelas a las guías de 8mm diámetro

S/. 10,00



Guía Lineal circular 8mm L300mm

S/. 30,00



Rodamiento lineal polimérico RJ4JP-01-08
Rodamiento polimérico para guía de 8mm de diámetro

S/. 15,00

**Cristal oscilador 16 MHz**

Cristal oscilador de 16 MHz de frecuencia. Utilizado en Arduino Uno,...

S/. 2,00

**Accesorio de Ajuste Correa GT2**

Ideal para unir los extremos de nuestra correa GT2.

S/. 10,00

**Módulo Relay 1 canal 5VDC**

El módulo Relay te permite controlar el encendido/apagado de equipos de...

S/. 6,00

**Sensor capacitivo LJC18A3-H-Z/BX**

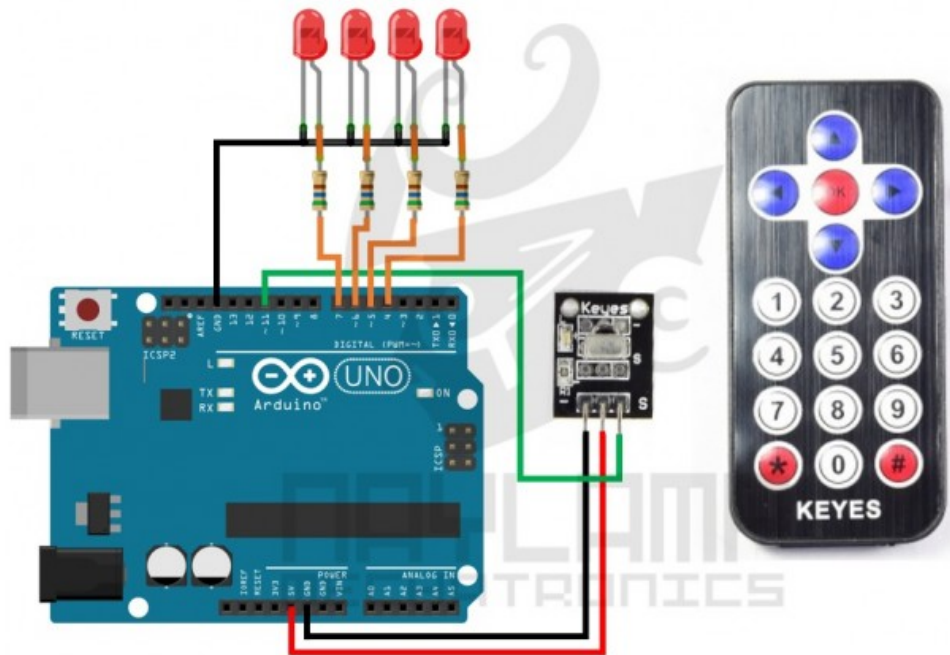
Sensor capacitivo te permite detectar objetos metálicos y no metálicos...

S/. 40,00

[Todas los nuevos productos >](#)

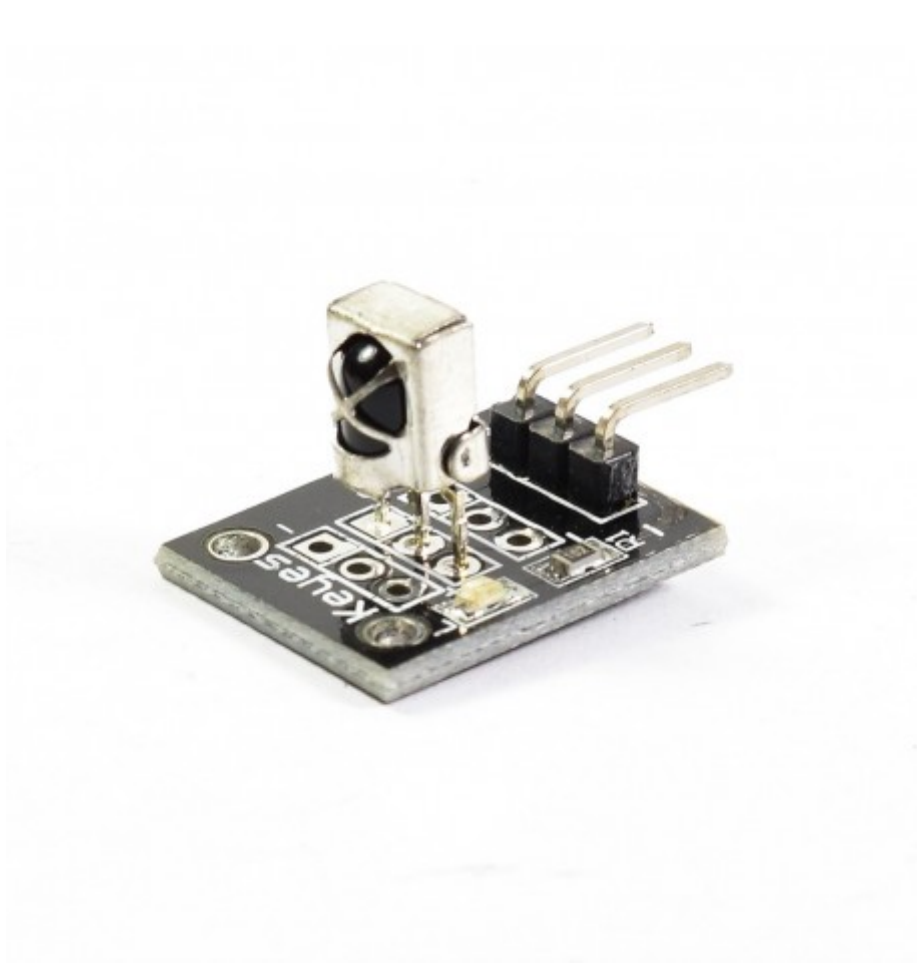
Tutorial Arduino y control remoto Infrarrojo

Posted by Tutoriales 2 Comments



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 infrarrojas 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.

Como mando usaremos uno que viene junto al sensor en el kit del mando infrarrojo.

Botón

Dirección (HEX)

comando (HEX)

Dato 32 bits (HEX)



Este Mando 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

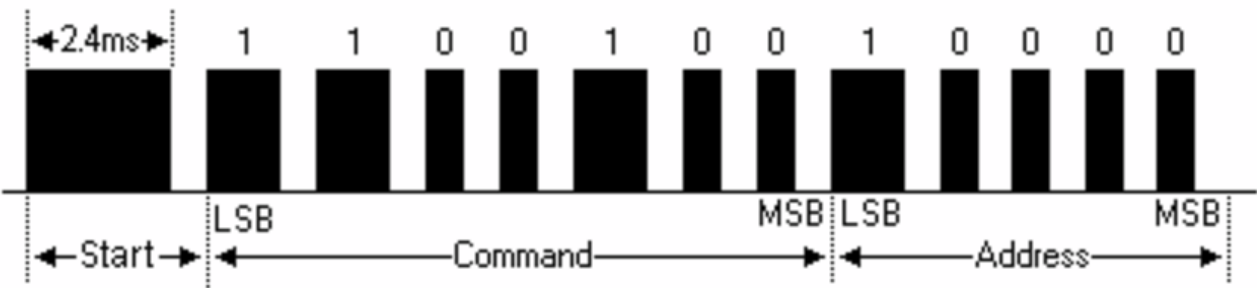
Botón	Dirección (HEX)	comando (HEX)	Dato 32 bits (HEX)
Arriba (↑)	0x00	0x62	0x00FF629D
Abajo (↓)	0x00	0xA8	0x00FFA857
Izquierda (←)	0x00	0x22	0x00FF22DD
Derecha (→)	0x00	0xC2	0x00FFC23D
1	0x00	0x68	0x00FF6897
2	0x00	0x98	0x00FF9867
3	0x00	0xB0	0x00FFB04F
4	0x00	0x30	0x00FF30CF
5	0x00	0x18	0x00FF18E7
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



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 funcion. 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

Botón	Dirección (DEC)	Comando (DEC)	Dato 12bits (Hex)
Power	1	21	0xA90
Menú	1	96	0X070

Botón	Dirección (DEC)	Comando (DEC)	Dato 12bits (Hex)
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 IR remote para Arduino

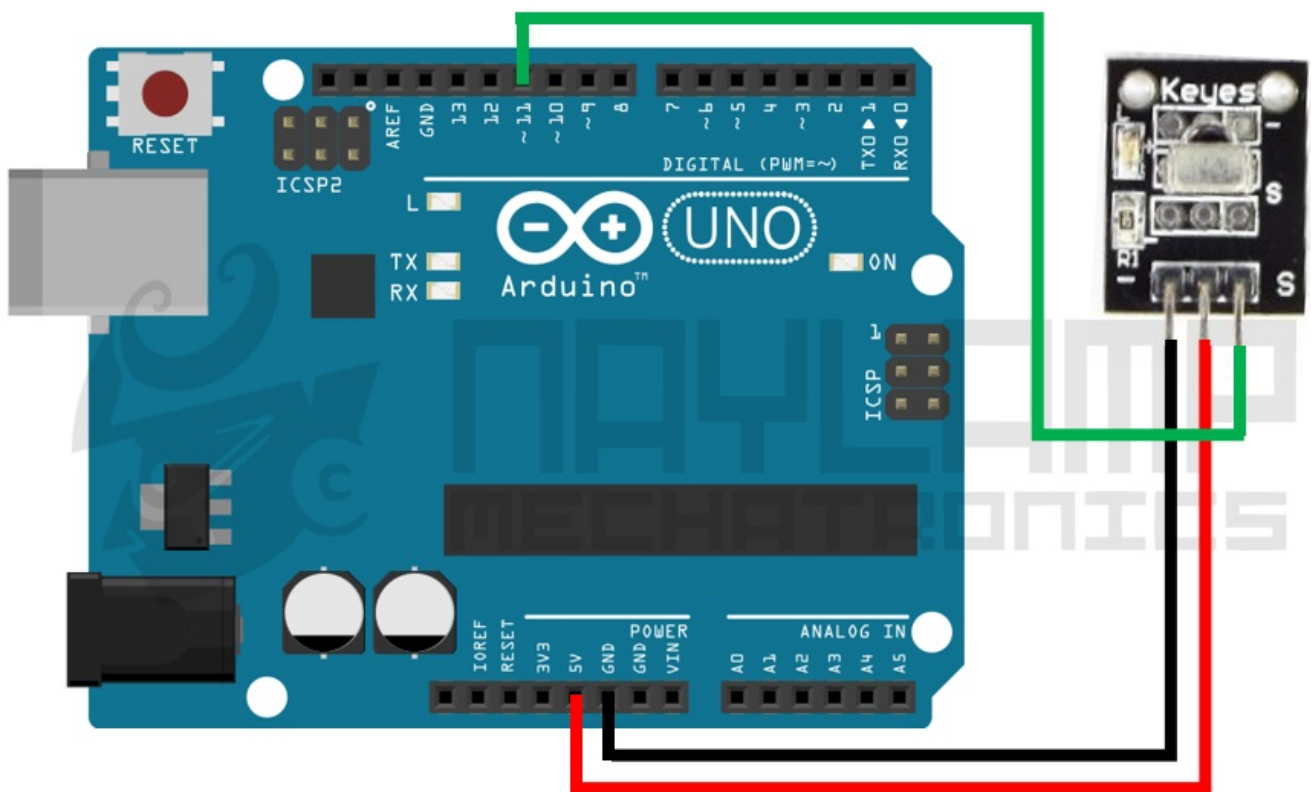
IRremote es una de las librerías más usadas y completas para trabajar con protocolos de controles infrarrojos, tiene implementado varios protocolos de las marcas más conocidas como Sony, LG, Samsung, Sanyo, etc

Pueden descargarlo y encontrar más información en: <https://github.com/z3t0/Arduino-IRremote>

Es necesario descargar e importarla a nuestro IDE Arduino para poder trabajar los ejemplos de este tutorial

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



Empecemos con nuestros ejemplos:

Encendiendo un led con nuestro control Remoto.

En este ejemplo se encenderá y apagará el led del pin 13 con cualquier tecla de nuestro control remoto, incluso con cualquier control.

El código es el siguiente:

```
#include <IRremote.h>
```

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

void setup()
{
  irrecv.enableIRIn(); // Empezamos la recepción por IR
  pinMode(13, OUTPUT);
}

boolean on = LOW;

void loop() {
  if (irrecv.decode(&results)) {
    // Dato recibido, conmutamos el LED
    on = !on;
    digitalWrite(13, on? HIGH : LOW);
    irrecv.resume(); // empezamos una nueva recepción
  }
  delay(300);
}
```

Explicuemos un poco el código:

Con `IRrecv irrecv(RECV_PIN)` creamos la variable u objeto para el receptor IR, en el pin especificado, luego creamos la variable `result`, 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 13 como salida.

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, encendemos o apagamos el LED.

Después de cargar el programa, al presionar cualquier tecla de cualquier control remoto, deberá encender o apagar el LED.

Decodificando datos de los controles infrarrojos.

En este ejemplo obtendremos los datos correspondientes a las teclas de los diferentes mandos infrarrojos.

El código es el siguiente:

```
#include <IRremote.h>
int RECV_PIN = 11;

IRrecv irrecv(RECV_PIN);

decode_results results;

void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Empezamos la recepción por IR
}
```

```

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

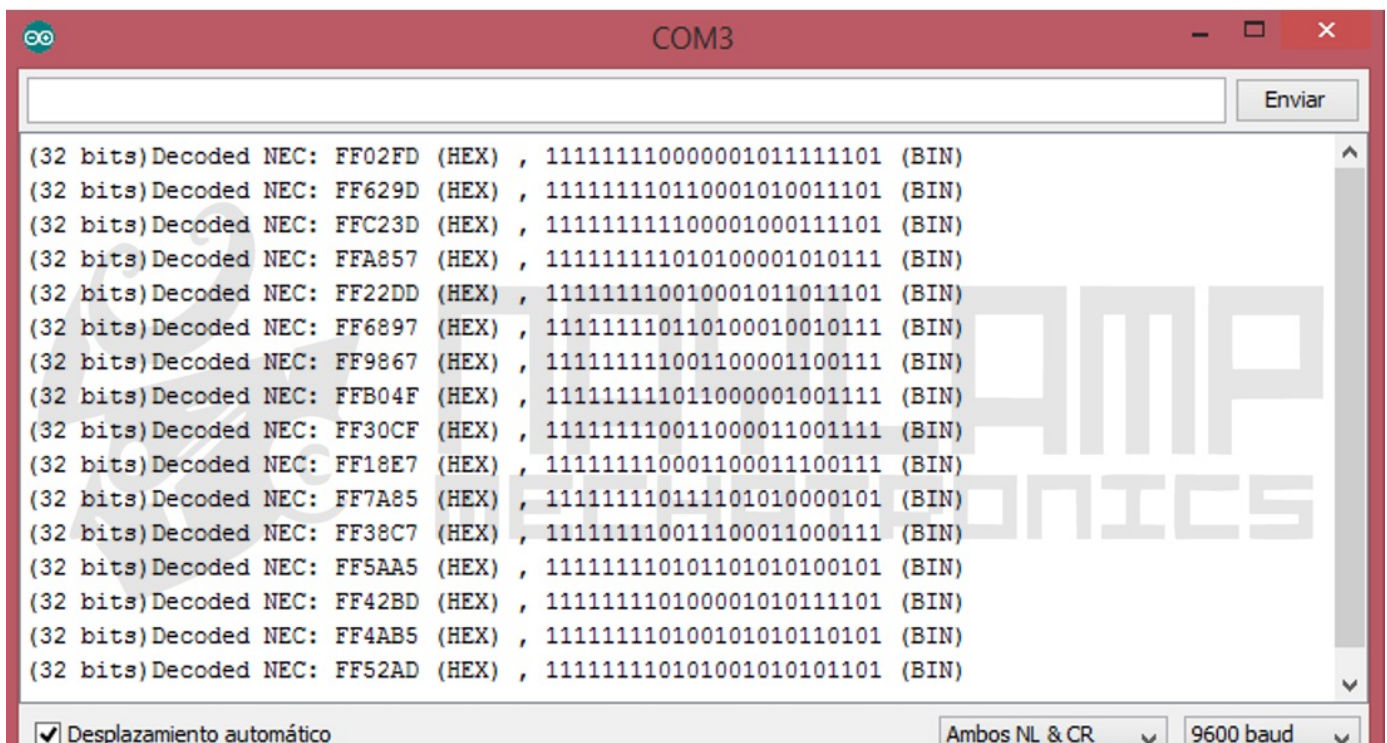
    else if (results->decode_type == SONY) {
        Serial.print("Decoded SONY: ");
    }
    else if (results->decode_type == RC5) {
        Serial.print("Decoded RC5: ");
    }
    else if (results->decode_type == RC6) {
        Serial.print("Decoded RC6: ");
    }
    else if (results->decode_type == 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);
}

```

El código anterior envía por el puerto serial los datos correspondientes a la tecla presionada.

A continuación se muestra los datos recibidos al presionar las teclas del control remoto que viene con el kit del sensor.

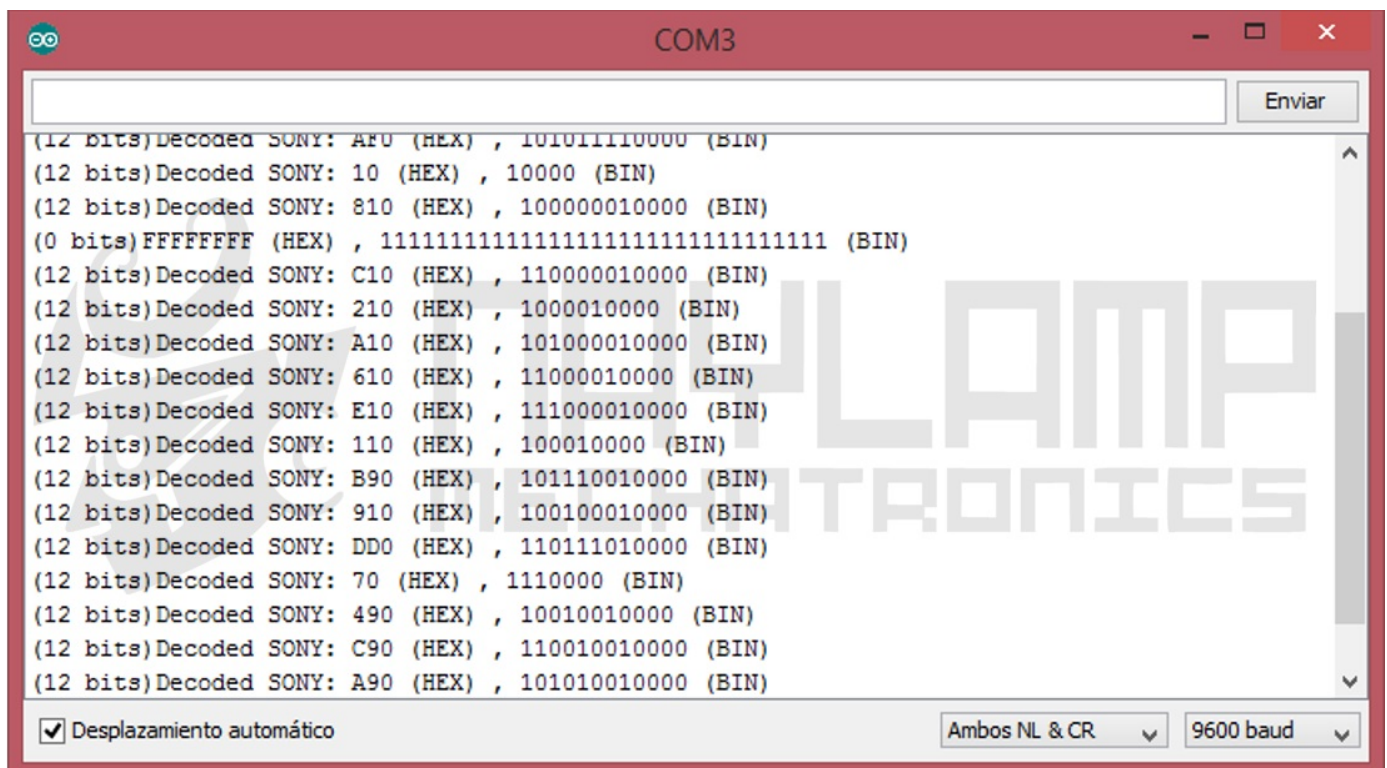


```

(32 bits)Decoded NEC: FF02FD (HEX) , 1111111100000010111111101 (BIN)
(32 bits)Decoded NEC: FF629D (HEX) , 111111110110001010011101 (BIN)
(32 bits)Decoded NEC: FFC23D (HEX) , 111111111100001000111101 (BIN)
(32 bits)Decoded NEC: FFA857 (HEX) , 111111111010100001010111 (BIN)
(32 bits)Decoded NEC: FF22DD (HEX) , 111111110010001011011101 (BIN)
(32 bits)Decoded NEC: FF6897 (HEX) , 111111110110100010010111 (BIN)
(32 bits)Decoded NEC: FF9867 (HEX) , 111111111001100001100111 (BIN)
(32 bits)Decoded NEC: FFB04F (HEX) , 111111111011000001001111 (BIN)
(32 bits)Decoded NEC: FF30CF (HEX) , 111111110011000011001111 (BIN)
(32 bits)Decoded NEC: FF18E7 (HEX) , 111111110001100011100111 (BIN)
(32 bits)Decoded NEC: FF7A85 (HEX) , 111111110111101010000101 (BIN)
(32 bits)Decoded NEC: FF38C7 (HEX) , 111111110011100011000111 (BIN)
(32 bits)Decoded NEC: FF5AA5 (HEX) , 111111110101101010100101 (BIN)
(32 bits)Decoded NEC: FF42BD (HEX) , 111111110100001010111101 (BIN)
(32 bits)Decoded NEC: FF4AB5 (HEX) , 111111110100101010110101 (BIN)
(32 bits)Decoded NEC: FF52AD (HEX) , 111111110101001010101101 (BIN)
  
```

☒ Desplazamiento automático Ambos NL & CR 9600 baud

Y si utilizamos el control remoto de SONY les debe mostrar valores similares al de la siguiente imagen:



```

(12 bits)Decoded SONY: AF0 (HEX) , 101011110000 (BIN)
(12 bits)Decoded SONY: 10 (HEX) , 10000 (BIN)
(12 bits)Decoded SONY: 810 (HEX) , 100000010000 (BIN)
(0 bits)FFFFFFF (HEX) , 11111111111111111111111111111111 (BIN)
(12 bits)Decoded SONY: C10 (HEX) , 110000010000 (BIN)
(12 bits)Decoded SONY: 210 (HEX) , 1000010000 (BIN)
(12 bits)Decoded SONY: A10 (HEX) , 101000010000 (BIN)
(12 bits)Decoded SONY: 610 (HEX) , 11000010000 (BIN)
(12 bits)Decoded SONY: E10 (HEX) , 111000010000 (BIN)
(12 bits)Decoded SONY: 110 (HEX) , 100010000 (BIN)
(12 bits)Decoded SONY: B90 (HEX) , 101110010000 (BIN)
(12 bits)Decoded SONY: 910 (HEX) , 100100010000 (BIN)
(12 bits)Decoded SONY: DD0 (HEX) , 110111010000 (BIN)
(12 bits)Decoded SONY: 70 (HEX) , 1110000 (BIN)
(12 bits)Decoded SONY: 490 (HEX) , 10010010000 (BIN)
(12 bits)Decoded SONY: C90 (HEX) , 110010010000 (BIN)
(12 bits)Decoded SONY: A90 (HEX) , 101010010000 (BIN)
  
```

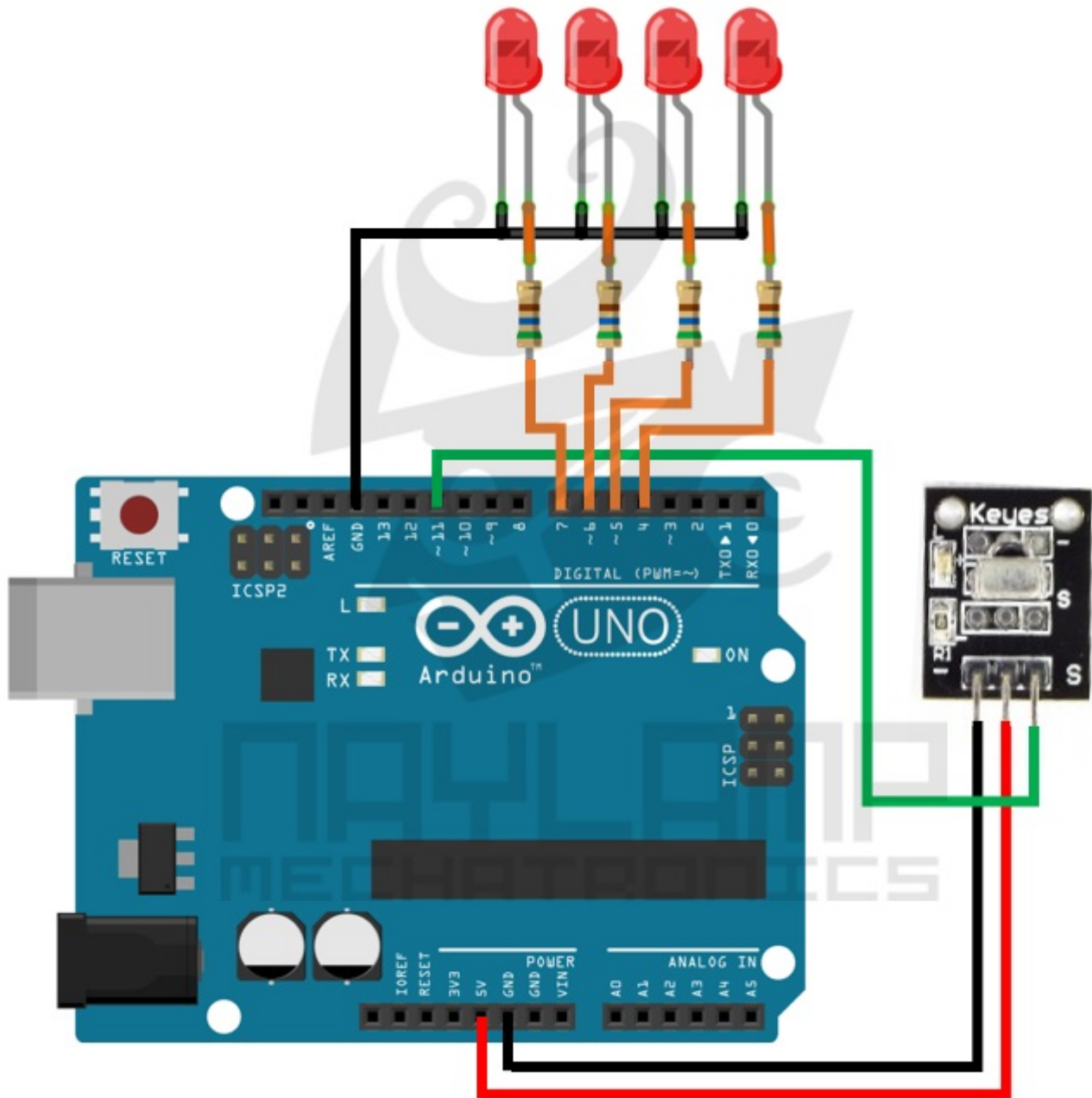
☒ Desplazamiento automático Ambos NL & CR 9600 baud

Con este programa podemos averiguar el valor del dato correspondiente a la tecla presionada, el mismo programa funciona para otras marcas de control remoto.

Controlar Pines digitales con control remoto por infrarrojos

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.

Para este ejemplo vamos a hacer las siguientes conexiones:



Implementemos el siguiente sketch:

```
#include "IRremote.h"
int receiver = 11;

IRrecv irrecv(receiver);
decode_results results;

void setup()
{
```

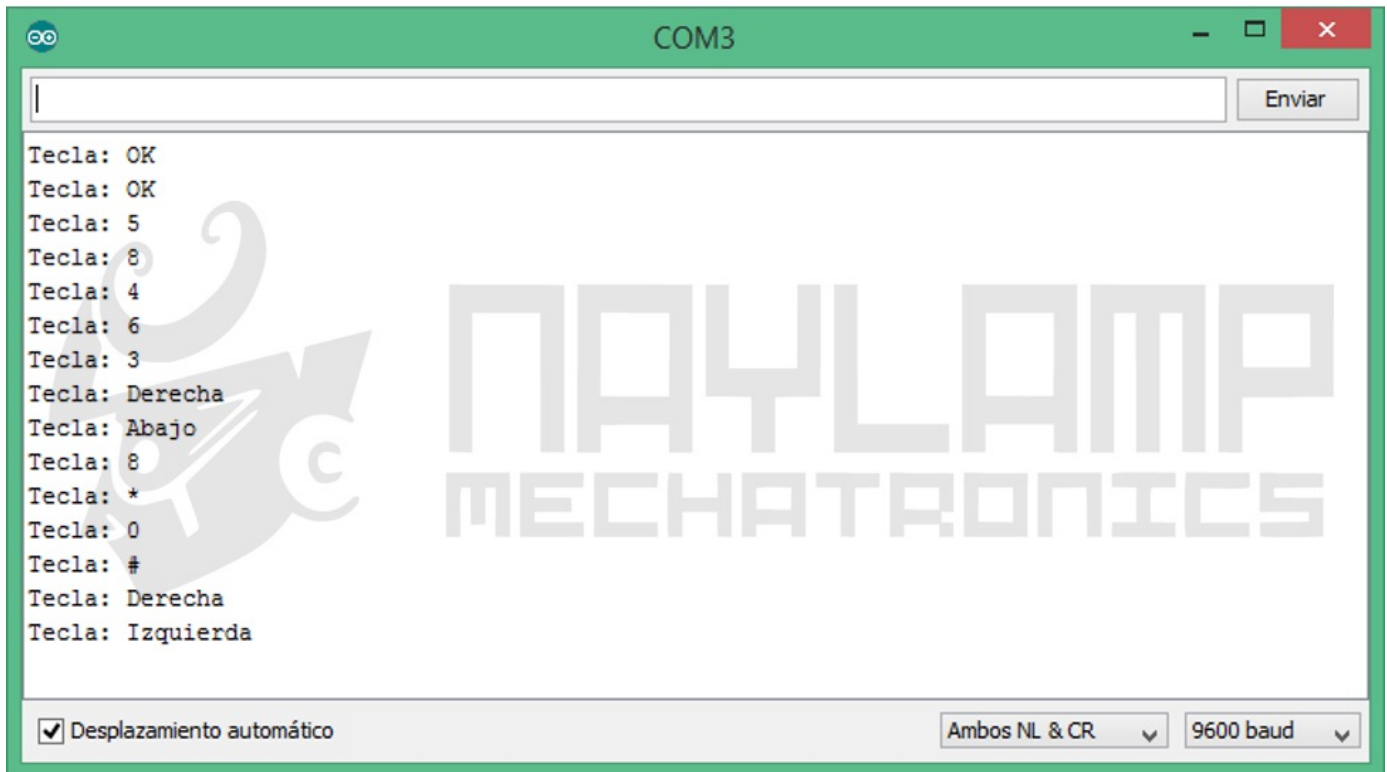


```
Serial.begin(9600);
irrecv.enableIRIn();
pinMode(4, OUTPUT);
pinMode(5, OUTPUT);
pinMode(6, OUTPUT);
pinMode(7, OUTPUT);
}

void loop()
{
  if (irrecv.decode(&results))
  {
    switch(results.value)
    {
      case 0x00FF629D: Serial.println("Tecla: Arriba");
                      break;
      case 0x00FF22DD: Serial.println("Tecla: Izquierda");
                      break;
      case 0x00FF02FD: Serial.println("Tecla: OK");
                      break;
      case 0x00FFC23D: Serial.println("Tecla: Derecha");
                      break;
      case 0x00FFA857: Serial.println("Tecla: Abajo");
                      break;
      case 0x00FF6897: Serial.println("Tecla: 1");
                      break;
      case 0x00FF9867: Serial.println("Tecla: 2");
                      break;
      case 0x00FFB04F: Serial.println("Tecla: 3");
                      break;
      case 0x00FF30CF: Serial.println("Tecla: 4");
                      digitalWrite(4, !digitalRead(4));
                      break;
      case 0x00FF18E7: Serial.println("Tecla: 5");
                      digitalWrite(5, !digitalRead(5));
                      break;
      case 0x00FF7A85: Serial.println("Tecla: 6");
                      digitalWrite(6, !digitalRead(6));
                      break;
      case 0x00FF10EF: Serial.println("Tecla: 7");
                      digitalWrite(7, !digitalRead(7));
                      break;
      case 0x00FF38C7: Serial.println("Tecla: 8");
                      break;
      case 0x00FF5AA5: Serial.println("Tecla: 9");
                      break;
      case 0x00FF42BD: Serial.println("Tecla: *");
                      break;
      case 0x00FF4AB5: Serial.println("Tecla: 0");
                      break;
      case 0x00FF52AD: Serial.println("Tecla: #");
                      break;
    }
    irrecv.resume();
  }
  delay(300);
}
```

Como se observa en el código solo comparamos el dato recibido del mando con los valores correspondientes a las teclas, si coinciden con el valor de la tecla, se realiza la acción correspondiente en el switch(), solo implementamos las teclas del 4 al 7 con leds, pero enviamos por el puerto serial una confirmación de cada tecla presionada.

A continuación se muestra el monitor serial después de presionar algunas teclas



De igual manera se trabaja con el control remoto de SONY:

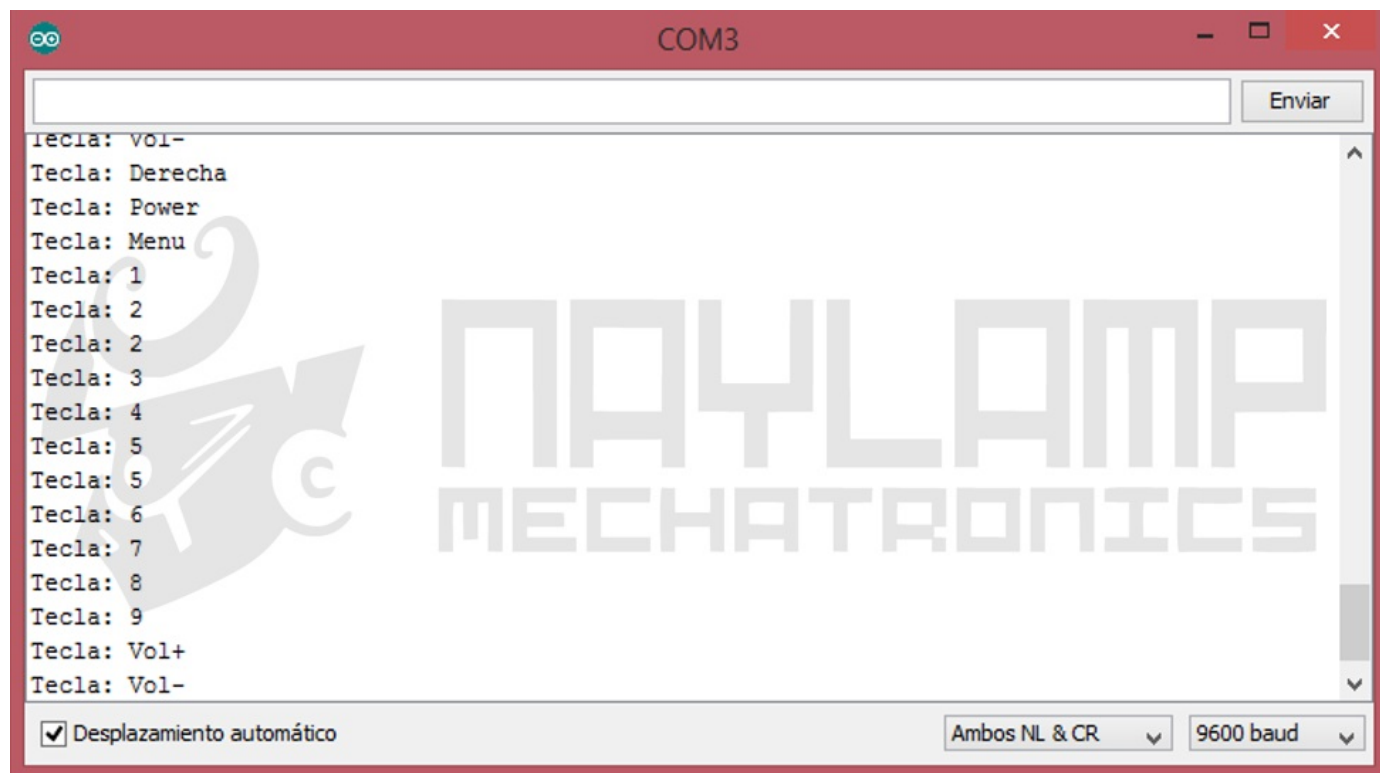
```
#include "IRremote.h"
int receiver = 11;

IRrecv irrecv(receiver);
decode_results results;

void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn();
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
}

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

```
case 0xA90: Serial.println("Tecla: Power");
            break;
case 0x070: Serial.println("Tecla: Menu");
            break;
case 0x2F0: Serial.println("Tecla: Arriba");
            break;
case 0x2D0: Serial.println("Tecla: Izquierda");
            break;
case 0xCD0: Serial.println("Tecla: Derecha");
            break;
case 0xAF0: Serial.println("Tecla: Abajo");
            break;
case 0x010: Serial.println("Tecla: 1");
            break;
case 0x810 : Serial.println("Tecla: 2");
            break;
case 0x410: Serial.println("Tecla: 3");
            break;
case 0xC10: Serial.println("Tecla: 4");
            digitalWrite(4, !digitalRead(4));
            break;
case 0x210: Serial.println("Tecla: 5");
            digitalWrite(5, !digitalRead(5));
            break;
case 0xA10: Serial.println("Tecla: 6");
            digitalWrite(6, !digitalRead(6));
            break;
case 0x610: Serial.println("Tecla: 7");
            digitalWrite(7, !digitalRead(7));
            break;
case 0xE10: Serial.println("Tecla: 8");
            break;
case 0x110: Serial.println("Tecla: 9");
            break;
case 0x910: Serial.println("Tecla: 0");
            break;
case 0x490: Serial.println("Tecla: Vol+");
            break;
case 0xC90: Serial.println("Tecla: Vol-");
            break;
}
irrecv.resume();
}
delay(300);
}
```

Pueden adquirir los materiales usados en este tutorial en nuestra tienda

- Arduino Uno R3
- Control Remoto Infrarrojo

Tags: inalámbrico infrarrojo

14

?

0

0

0

Reordenar esta lista usando las flechas.

2 Comments

**Arturo**

ago 25, 2016

Disculpa amigo muy buen post pero una pregunta como puedo colocar 4 receptores infrarrojos al mismo arduino necesiti hacer una aplicación a si te agradecería me podrías orientar gracias.

[Reply](#)**Naylamp**

ago 26, 2016

Hola Arturo, no veo la necesidad de poner más receptores al mismo arduino si todos van a recibir la misma señal, salvo intentes ponerlo en diferentes posiciones para ganar más ángulo de recepción. Intenta crear diferentes objetos para cada

```
receptor, por ejemplo para un segundo receptor: IRrecv irrecv2(10); decode_results  
results2;
```

[Reply](#)

Leave a Reply

* **Name:**

* **E-mail:**

(Not Published)

Website:

(Site url withhttp://)

* **Comment:**

Boletín

Introduzca su dirección de correo electróni



Información



[Contacte con nosotros](#)

[Entrega](#)

[Condiciones de uso](#)

[Nosotros](#)

[Tutoriales y Proyectos con Arduino](#)

[Mapa del sitio](#)

Mi cuenta



[Mis compras](#)

[Mis vales descuento](#)

[Mis direcciones](#)

Mis datos personales

Mis vales

Información sobre la tienda



Naylamp Mechatronics, Trujillo Perú



Llámanos ahora: 997646858



Email: naylamp.mechatronics@gmail.com