ARDUINO (HTTPS://WWW.FALCYLIVE.IT/CATEGORY/ARDUINO/)
TUTORIALS (HTTPS://WWW.FALCYLIVE.IT/CATEGORY/TUTORIALS/)

Arduino: Cloniamo un Telecomando ad Infrarossi

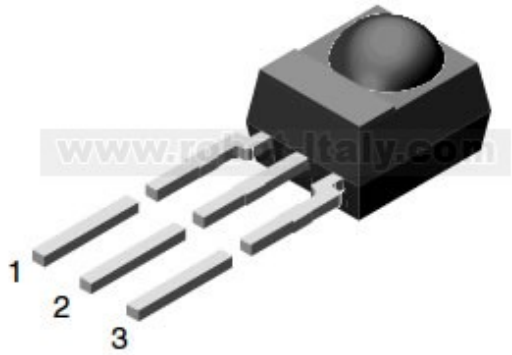
By falcylive (https://www.falcylive.it/author/falcylive/) on 27 Febbraio 2016

Con il nostro Arduino possiamo fare una marea di cose. Oggi cloneremo un qualsiasi telecomando ad infrarossi di casa. Una volta acquisiti tutti i codici del telecomando possiamo darli in pasto per esempio al nostro Raspberry e comandare tutti i nostri device comodamente dal nostro smartphone.

Hardware Necessario

Ovviamente ci servirà un Arduino. Qualsiasi modello va bene !!

Costo: dai 3 ai 30 euro (vanno bene anche i modelli cinesi da 3 euro)



Un Ricevitore ad Infrarossi. Ce ne sono una marea in giro. Io ho usato il modello TSOP34836 e devo dire che funziona benone. Diciamo che qualsiasi TSOP va bene. Guardate solo su internet le specifiche. Deve lavorare intorno ai 36-38Khz.

Costo: 1.5 euro

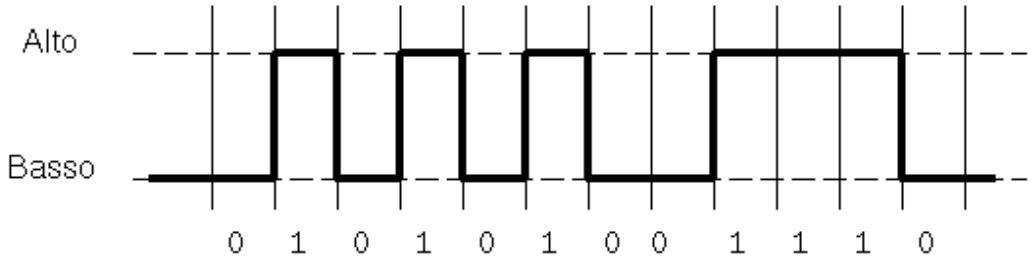
Un LED trasmettitore ad Infrarossi. Vanno bene di qualsiasi tipo !!!

Costo: 1 euro o meno

Come funziona una trasmissione ad infrarossi ?

Il meccanismo di funzionamento di una trasmissione ad infrarossi è abbastanza “semplice”; un fascio luminoso, ad elevata frequenza 38 KHz (per i telecomandi tipici per la TV) viene proiettato contro un ricevente. Questo fascio luminoso non è altro che una sequenza di bit (0 e 1) che vengono codificati dal ricevente, attraverso la polarizzazione di photo-transistor. Facciamo questo semplice caso; nel nostro telecomando, vengono codificati i vari pulsati con varie sequenze bit e per esempio il tasto on/off vale:

010101001110



(http://ismanettoneblog.altervista.org/blog/wp-content/uploads/2014/02/02tipi01.gif)

Il ricevente avrà un segnale in ricezione molto simile, se non uguale a quello inviato e in base alla corrispondenza che ha il memoria, sarà in grado di svolgere la richiesta effettiva, come ad esempio accendere la TV ! Il fascio luminoso che viene inviato dal telecomando, non è possibile vederlo ad occhio nudo perché ha una frequenza troppo veloce per noi !

Tuttavia, ci sono diversi codifiche; tipicamente ogni produttore di dispositivi tecnologici, ha una propria codifica, come ad esempio Sony, NEC, Samsung etc..

Tuttavia, è possibile stabilire la sequenza di bit di qualsiasi segnale ad infrarosso, calcolando il tempo in cui il segnale è alto e il tempo in cui il segnale è basso. Questo tipo di codifica si chiama RAW.

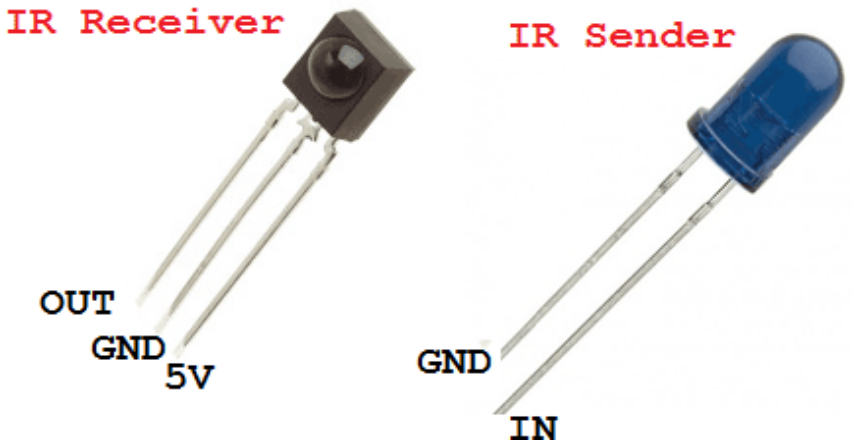
Codici per Arduino

Per far funzionare il tutto dovremo scaricare questa famosissima libreria:

Scarica Libreria (http://adf.ly/1XYFVv)

Come per ogni libreria nuova in Arduino, dobbiamo installarla, copiandola semplicemente nella cartella Libraries, che tipicamente si trova nei propri documenti.

Qualora utilizzassimo il dispositivo TSOP34836 , il collegamento è davvero semplice; basterà alimentare il segnale con i 5 V di Arduino, collegarlo a massa e collegare il pin di output al PIN 11 del nostro Arduino. Per quanto riguarda il LED trasmettitore colleghiamolo al PIN 3 del nostro Arduino con in serie una resistenza di almeno 100 ohm.

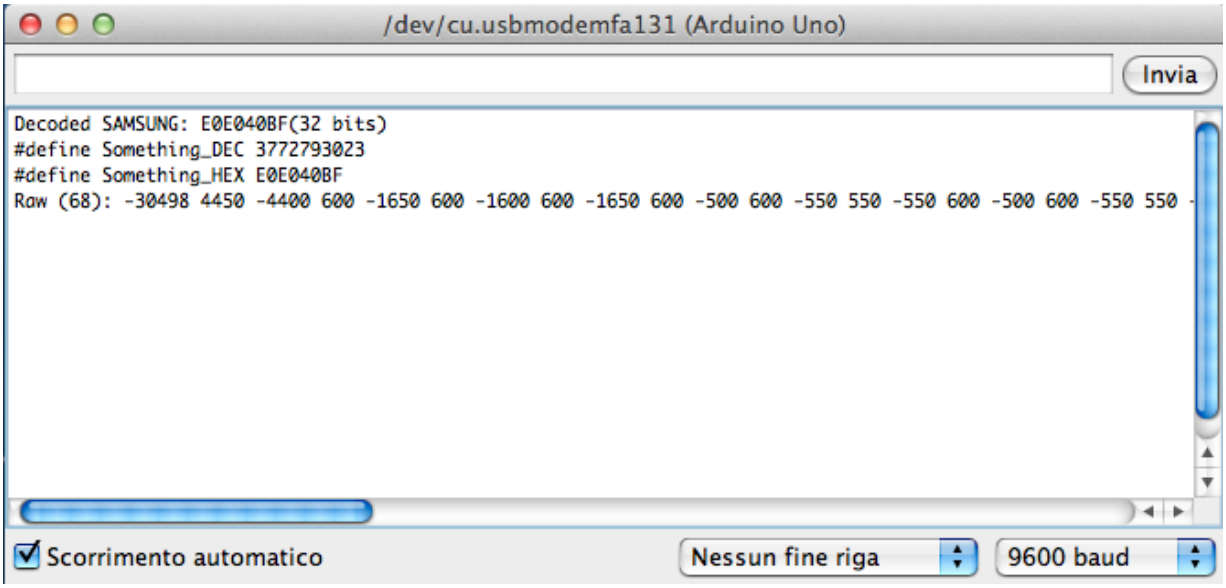


Ora apriamo il tool di Arduino e carichiamo questo codice:

```
/*

 * Modified by Chris Targett
 * Now includes more protocols
 * Novemeber 2011

 * IRremote: IRrecvDump - dump details of IR codes with IRrecv
 * An IR detector/demodulator must be connected to the input RECV_PIN.
 * Version 0.1 July, 2009
 * Copyright 2009 Ken Shirriff
 * http://arcfn.com
 *
 * Modified by Chris Targett to speed up the process of collecting
 * IR (HEX and DEC) codes from a remote (to put into and .h file)
 *
 */

#include <IRremote.h>

int RECV_PIN = 11;

IRrecv irrecv(RECV_PIN);

decode_results results;

void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
}

// Dumps out the decode_results structure.
// Call this after IRrecv::decode()
// void * to work around compiler issue
//void dump(void *v) {
//  decode_results *results = (decode_results *)v
void dump(decode_results *results) {
  int count = results->rawlen;
  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 == SAMSUNG) {
    Serial.print("Decoded SAMSUNG: ");
  }
  else if (results->decode_type == JVC) {
    Serial.print("Decoded JVC: ");
```

```
  }
  else if (results->decode_type == PANASONIC) {
    Serial.print("Decoded Panasonic: ");
  }
  Serial.print(results->value, HEX);
  Serial.print("(");
  Serial.print(results->bits, DEC);
  Serial.println(" bits)");
  Serial.print("#define Something_DEC ");
  Serial.println(results->value, DEC);
  Serial.print("#define Something_HEX ");
  Serial.println(results->value, HEX);
  Serial.print("Raw (");
  Serial.print(count, DEC);
  Serial.print("): ");
  for (int i = 0; i < count; i++) {
    if ((i % 2) == 1) {
      Serial.print(results->rawbuf[i]*USECPERTICK, DEC);
    }
    else {
      Serial.print(-(int)results->rawbuf[i]*USECPERTICK, DEC);
    }
    Serial.print(" ");
  }
  Serial.println("");
}

void loop() {
  if (irrecv.decode(&results)) {
    dump(&results);
    irrecv.resume(); // Receive the next value
  }
}
```

Apriamo la console seriale e puntiamo il nostro telecomando, vedremo una cosa del genere:



Fantastico !!! Se leggiamo Decoded NEC, SAMSUNG ecc.. vuol dire che la decodifica è stata semplice ed è andata a buon fine. Se leggiamo invece Unknown encoding: vuol dire che dovremo lavorare con il codice raw per riprodurre il segnale.

Nel nostro caso abbiamo letto **E0E040BF** a **32 bit**. Segnamoci questi valori e carichiamo il nostro Arduino il seguente codice:

```
/*
 * IRremote: IRsendDemo - demonstrates sending IR codes with IRsend
 * An IR LED must be connected to Arduino PWM pin 3.
 * Version 0.1 July, 2009
 * Copyright 2009 Ken Shirriff
 * http://arcfn.com
 */


#include <IRremote.h>

IRsend irsend;

void setup()
{
  Serial.begin(9600);
}

void loop() {
  Serial.print("Sent Signal...");
  irsend.sendSAMSUNG(0xE0E040BF, 32);
  delay(5000); //5 second delay between each signal burst
}
```

Come vedete il codice è molto semplice. **Il segnale verrà sempre inviato al PIN 3 !!!**

Cambiate la riga sendSAMSUNG con quello che avete letto all'inizio, quindi può essere sendNEC, sendSONY ecc.. Seguite questa logica.

```
irsend.sendMODELLO_LETTO(0xValore_Letto, bitLetti)
```

Ora con il codice caricato invierete il segnale (pultroppo a massimo 60-70 cm di distanza) al vostro dispositivo.

Se invece avete avuto Unknown encoding allora:

Ora dobbiamo prendere il codice RAW, cancellare il primo valore, convertire i segni – con degli spazi e mettere le virgole tra i vari valori.

30498,4450,4400,600,1650,600,1600,600,1650,600,500,600,550,550,550,

600,500,600,550,550,1600,650,1600,600,1650,600,550,550,550,550,550,600,

550,550,550,600,500,600,1600,600,550,600,550,550,550,550,550,600,500,600,

550,600,1600,600,550,550,1650,600,1600,600,1650,600,1600,600,1650,600,1600,600,

Il codice per Arduino che serve per inviare il segnale al nostro dispositivo è il seguente:

```
#include "IRremote.h"

IRsend irsend;

void setup()
{
  Serial.begin(9600);
}

unsigned int ON[68]={4450,4400,600,1650,600,1600,600,1650,600,500,600,550,550,550,
600,500,600,550,550,1600,650,1600,600,1650,600,550,550,550,550,550,
600,550,550,550,600,500,600,1600,600,550,600,550,550,550,550,550,600,
500,600,550,600,1600,600,550,550,1650,600,1600,600,1650,600,1600,600,1650,600,1600,600

void loop() {
 irsend.sendRaw(ON,68,38);
 Serial.println("Accendo la TV Samsung");
 delay(10000);
 }
```

come vedete abbiamo usato il sendRaw che funziona così:

irsend.sendRaw(rawbuf, rawlen, frequency);

Il rawbuf rappresenta il vettore che contiene tutti i codici, rawlen rappresenta il numero di elementi del vettore, frequency è il frequenza di trasmissione in KHz.

Ciao