Department of Computer Science & Engineering

Motilal Nehru National Institute of Technology Allahabad

UNIVERSAL REMOTE CONTROLLER

Submitted As
EMBEDDED SYSTEM PROJECT (CS - 1601)

Submitted To:
Er. Dushyant K. Singh
B.Tech CSE VI Semester
by

T.U. Akash Unni : 20134141
Johnson Bhengra : 20134094
Thara Anil Kumar : 20134137
Banda Prashanth Yadav : 20134164
Surendra Kumar : 20134149
Bhavesh Nemade : 20134130
Vikas Yadav : 20134054

Universal Remote Controller

Objective:

The main aim of this project is to introduce a new universal remote control that gives easy-to-control interface for home devices such as TV, video/audio player, room lighting etc.

Introduction:

With advances in technology, there are more electronic devices in our daily lives than we can actually efficiently handle! So, with the advent of the present technological revolution comes the difficult task of controlling our electronic equipment. Naturally comes into our minds the ‘All for one, one for all’ strategy and we ponder over the possibilities of a universal remote controller.

One controller for all our devices does sound lucrative, doesn’t it! With the objective of making lives easier, we set sail on our mission, to design a low cost ‘learning’ universal remote controller, a remote that could ‘learn’ from other remotes.

Implementation:

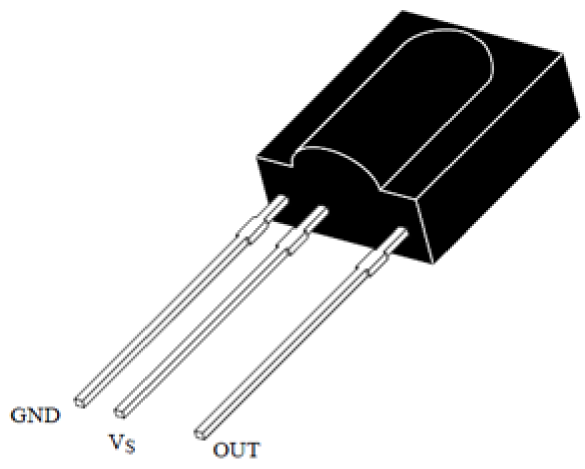
Major Components:

- **Arduino UNO**:

  The Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller.

- **TSOP 1738:**

TSOP1738 is a commonly used IR receiver for Infrared PCM remote control systems. It is used in TVs, DVD Players, and Burglar Alarms etc. This component is built with PIN Diode, Preamplifier and internal filter for PCM (Pulse Code Modulation) frequency and its epoxy package is designed as an IR filter. These are available with different carrier frequencies out of which TSOP1738 is very common whose carrier frequency is 38KHz. The output of TSOP1738 receivers can be directly connected to a microcontroller or microprocessor for further processing.

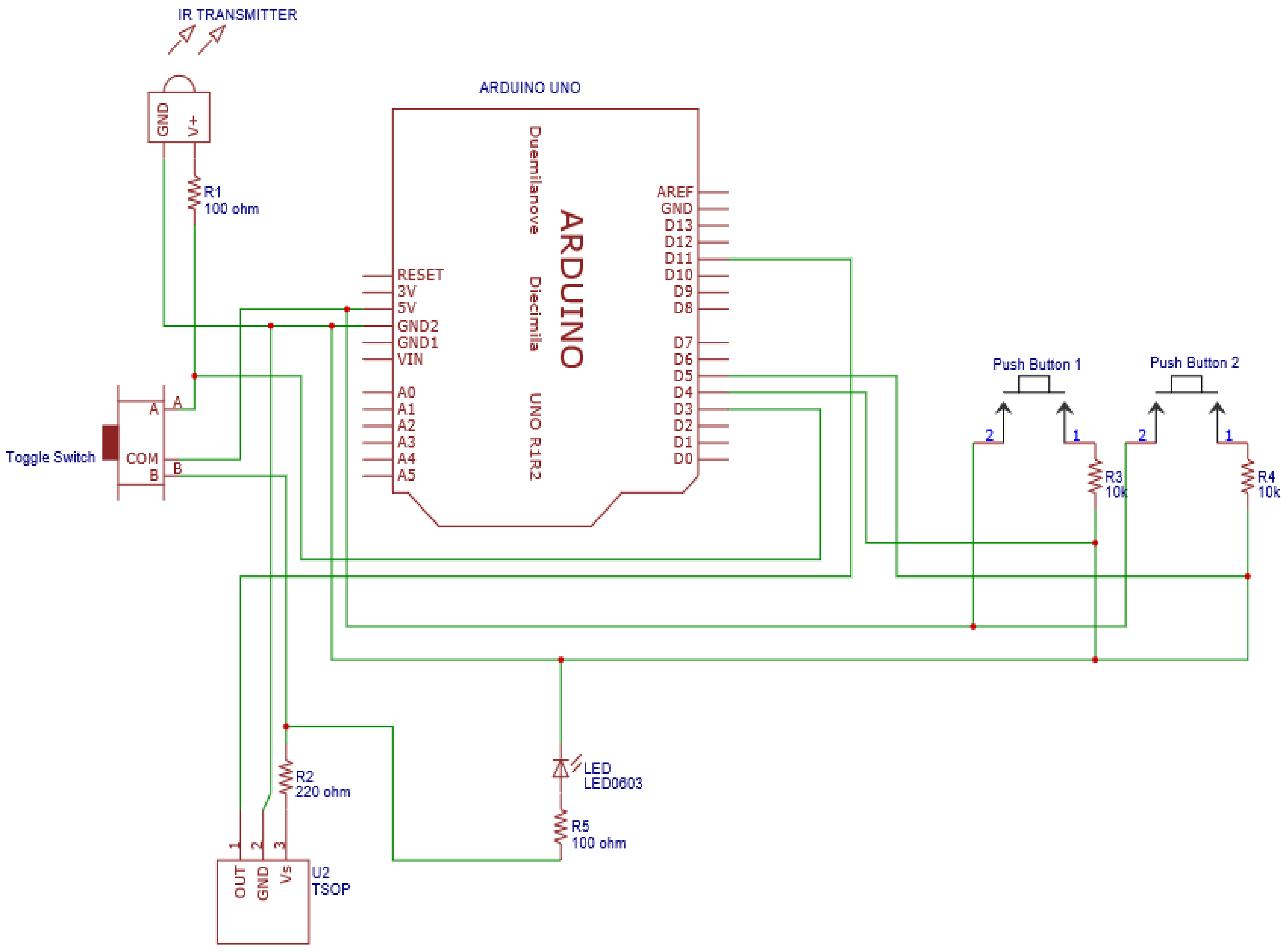


- **IR Transmitter:**

The IR (infrared) transmitter sends a short pulse of modulated infrared light. You can activate it with a trigger from an input module (like a button). Use it to wirelessly activate the AC switch to turn appliances like a lamp or fan on and off! Note: Make sure you pair it with the AC switch before using.

Hardware: <u>The Circuit</u>

Arduino controls both the reception and the transmission parts of the circuit. There is a toggle switch to select between the two modes as shown in the circuit diagram below. Values of a device can be stored at one time. Each function is explained in some detail below.



Working Description:

The work consists of blocks namely IR transmitter, IR receiver, Arduino, Toggle Switch & Push Buttons. IR sensor module receives the IR signal. Now, the Arduino stores the bit pattern of the IR transmitter of remote.

There are two modes in this device:

1. Receiver Mode.

   TSOP receives IR signal from the remote and configures the button pushed.

2. Transmitter Mode.

   Turns on IR Transmitter and transmits the IR signals of the corresponding button pressed.

- **Toggle Switch:**
  Toggles between Reciever mode and Transmitter mode.

- **IR Transmitter:**
  Transmits the IR signal of the button pressed.

- **IR Receiver:**
  Receives IR signals sent by the remote and stores in the memory allocated to the button pressed.

How it works?

Code:

```
/*
 * IRremote: IRrecvDemo - demonstrates receiving 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
 */
#include <IRremote.h>

int RECV_PIN = 11;
int switchPin = 8;
int btn_1 = 4;
int btn_2 = 5;


IRrecv irrecv(RECV_PIN);
IRsend irsend;
decode_results results;

struct button{
  int nbits;
  decode_type_t type;
  unsigned long code[5];
}btn[2];

void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
}
int i=0,j=0,k=0,x;
void loop() {
  //---------------- Reciever Code ------------------//
  if (digitalRead(switchPin) == HIGH){
    //---------- SET Button 1 ----------//
    if (irrecv.decode(&results) && digitalRead(btn_1) == HIGH) {
        btn[0].code[i] = results.value;
        btn[0].nbits = results.bits;
        btn[0].type = results.decode_type;
        i++;
        irrecv.resume(); // Receive the next value

          for(x=0; x<i; x++){
            Serial.println(btn[0].code[x]);
          }
          Serial.println(results.decode_type);
          Serial.println(btn[0].nbits);
          Serial.println("___________");

    }
    //---------- SET Button 2 ----------//
    else if (irrecv.decode(&results) && digitalRead(btn_2) == HIGH) {
        btn[1].code[j] = results.value;
        btn[1].nbits = results.bits;
        btn[1].type = results.decode_type;
        j++;
```

```
      irrecv.resume(); // Receive the next value


        for(x=0; x<j; x++){
          Serial.println(btn[1].code[x]);
        }
        Serial.println(results.decode_type);
        Serial.println(btn[1].nbits);
        Serial.println("___________");

    }
  }


  //------------ Transmitter Code ----------------//
  else{
    //---------- SEND Button 1 ----------//
    if (digitalRead(btn_1) == HIGH) {


      for(x=0;x<i;x++){
        if(btn[0].type == NEC)
          irsend.sendNEC(btn[0].code[x], btn[0].nbits);

        else if(btn[0].type == SONY)
          irsend.sendSony(btn[0].code[x], btn[0].nbits);

        else if(btn[0].type == RC5)
          irsend.sendRC5(btn[0].code[x], btn[0].nbits);

        else if(btn[0].type == RC6)
          irsend.sendRC6(btn[0].code[x], btn[0].nbits);

        delay(40);

          Serial.println(btn[0].code[x]);


        if(Serial.available())
          Serial.println("Transmitter on 1");
    }
    //---------- SEND Button 2 ----------//
    else if (digitalRead(btn_2) == HIGH) {


      for(x=0;x<j;x++){
        if(btn[1].type == NEC)
          irsend.sendNEC(btn[1].code[x], btn[1].nbits);

        else if(btn[0].type == SONY)
          irsend.sendSony(btn[1].code[x], btn[1].nbits);

        else if(btn[0].type == RC5)
          irsend.sendRC5(btn[1].code[x], btn[1].nbits);

        else if(btn[0].type == RC6)
          irsend.sendRC6(btn[1].code[x], btn[1].nbits);

        delay(40);


          Serial.println(btn[1].code[x]);

      }
```

```
            Serial.println("Transmitter on 2");
      }
      delay(500);
    }
    delay(100);
  }
  }
```

Code Explanation:

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

IRremote is a library available to work with IR transmission and reception with Arduino.

```
int RECV_PIN = 11;
int switchPin = 8;
int btn_1 = 4;
int btn_2 = 5;
```

- RECV_PIN is the pin at which TSOP sends the received IR signal.
- switchPin is the pin to find the state of toggle switch.
- btn_1,btn_2 are the pins to which the two push buttons are connected.

```
IRrecv irrecv(RECV_PIN);
IRsend irsend;
decode_results results;
```

- IRrecv,IRsend are the classes in which the receiving and transmitting functions are defined present in the IRremote library.
- decode_results is the data type in which the decoded signal is stored.

```
struct button{
  int nbits;
  decode_type_t type;
  unsigned long code[5];
}btn[2];
```

btn[] is the array of structure consisting of decoded value, number of bits and type. Each element is assigned to a push button.

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

Receiver is enabled.

```
      if (irrecv.decode(&results) && digitalRead(btn_1) == HIGH) {
        btn[0].code[i] = results.value;
        btn[0].nbits = results.bits;
        btn[0].type = results.decode_type;
        i++;
        irrecv.resume(); // Receive the next value


          for(x=0; x<i; x++){
            Serial.println(btn[0].code[x]);
          }
          Serial.println(results.decode_type);
          Serial.println(btn[0].nbits);
          Serial.println("___________");
```

This is the code to receive the signal, decode it and configure the button pressed.

```
if (digitalRead(btn_1) == HIGH) {


      for(x=0;x<i;x++){
        if(btn[0].type == NEC)
          irsend.sendNEC(btn[0].code[x], btn[0].nbits);

        else if(btn[0].type == SONY)
          irsend.sendSony(btn[0].code[x], btn[0].nbits);

        else if(btn[0].type == RC5)
          irsend.sendRC5(btn[0].code[x], btn[0].nbits);

        else if(btn[0].type == RC6)
          irsend.sendRC6(btn[0].code[x], btn[0].nbits);

        delay(40);


          Serial.println(btn[0].code[x]);

      }

                 Serial.println("Transmitter on 1");
}
```

This is the code to transmit the corresponding signal of the button pressed.

How to use? :

- Dump the code into Arduino.
- Toggle into receiver mode by using toggle switch.
- Hold the button to be configured and press the button in the remote whose functionality is to be replicated.
- Switch to transmitter mode, press the configured button to use it.

Limitations:

At present, this device supports SONY, RC5, RC6 and NEC model devices.

Future Development:

- LCD display can be attached so that the user can save multiple remote configurations at a time.
- Accelerometer can be attached to control the actions using gestures instead of buttons.

Picture after completion: