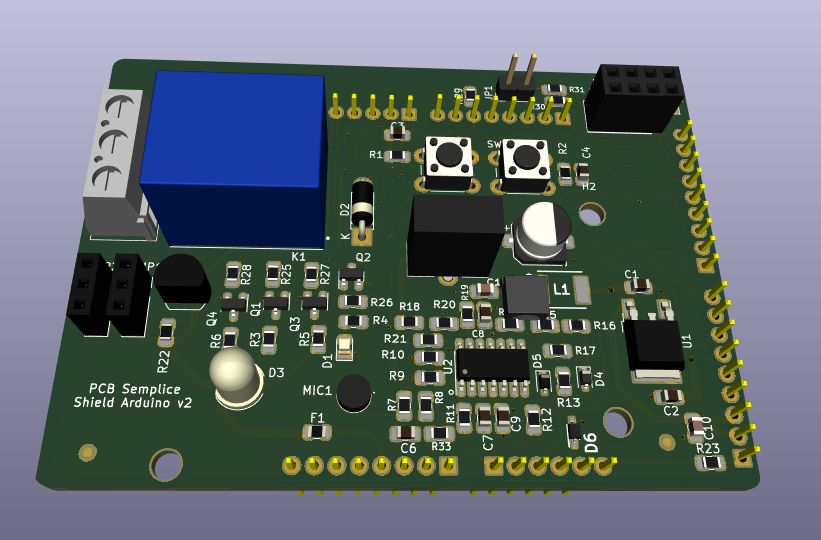
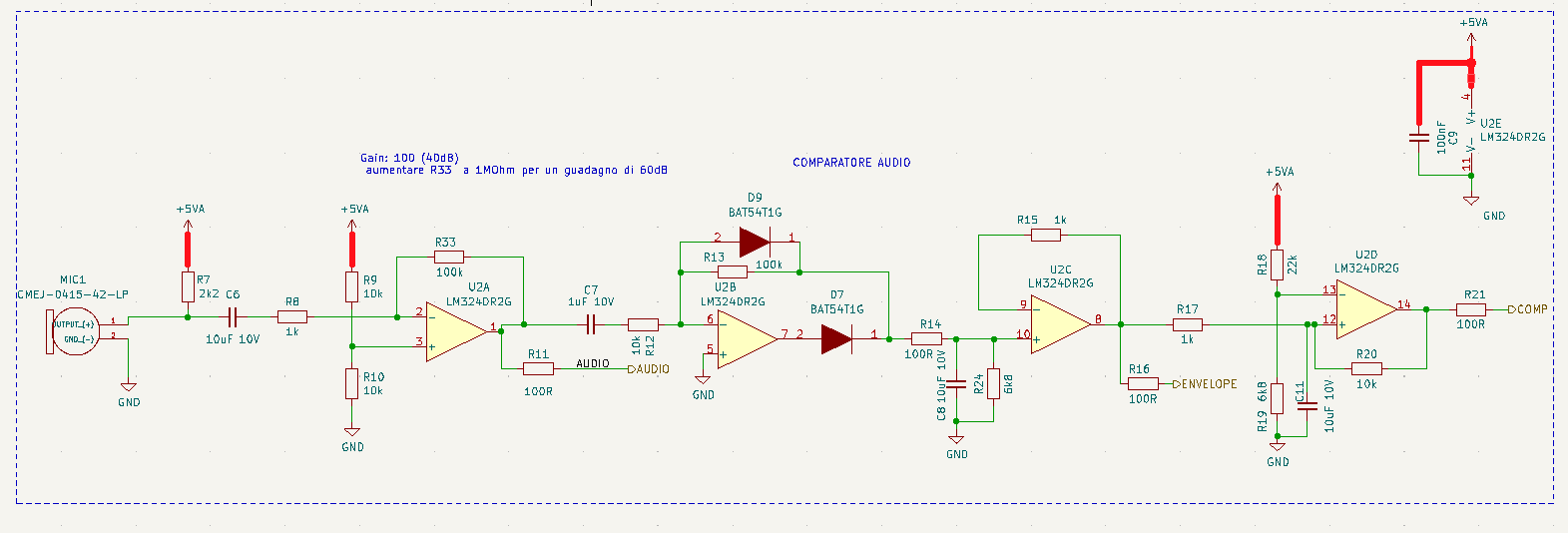
Shield Arduino Uno v2 description

  
The board has these sections

* An audio switch
* An analog trimmer connected to an Arduino analog pin
* Two push-buttons
* A 5mm RGB LED
* A 10 A SPDT relay
* A connector in order to install a Wi-Fi ESP8266 module
* A 3.3V 1 A LDO
* There are two headers in order to connect the shield to a a breadboard

## Audio switch

This stage is based on LM324 operational amplifier

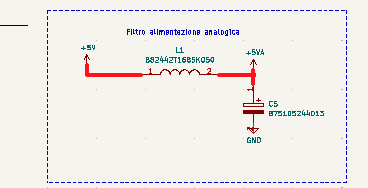
The first section (U2A) of the operational is an inverting amplifier. The microphone signal is amplified and the gain is determined by the resistor R33.

The output is sent both to an analog channel of the Arduino Uno board and to the second stage (U2B) which creates a half wave rectifier.

The rectifier output is sent to the third stage (U2C) which forms a buffer with a low-pass filter.

The output of this last stage is sent both to an analog pin of the Arduino Uno board and to the fourth stage (U2D) which is a Schmitt trigger. The output (COMP) of this last stage is sent to a digital pin of the Arduino Uno.

To avoid spurious switching, the supply voltage (+5VA) of the operational amplifier is obtained from the voltage +5V through a filter consisting of the inductor L1 and the capacitor C5



The sketch loaded into the ATMEGA328P microcontroller of the Arduino Uno board activates a relay when the "a" sound is pronounced twice. The relay is deactivated when the sound “a” is pronounced three times

const byte ledPin = 12;

const byte rele = 4;

const byte interruptPin = 2;

volatile byte state = LOW;

unsigned char count =0;

unsigned char ON =0;

unsigned char OFF =0;

unsigned long timeComparison = 0;

unsigned char alreadyON =0;

void setup() {

 pinMode(ledPin, OUTPUT);

 pinMode(interruptPin, INPUT\_PULLUP);

 attachInterrupt(digitalPinToInterrupt(interruptPin), checkComparator, RISING);

**Serial**.begin(9600);

}

void loop() {

**Serial**.println(count);

 if (ON==1 && alreadyON==0)

 {

 digitalWrite(ledPin, HIGH);

 digitalWrite(rele, HIGH);

 alreadyON=1;

 count=0;

**Serial**.println("ON");

 }

 if (OFF==1){

  digitalWrite(ledPin, LOW);

  digitalWrite(rele, LOW);

**Serial**.println("OFF");

  alreadyON =0;

  count=0;

  OFF=0;

 }

}

void checkComparator() {

 count++;

switch (count){

   case 1:

   timeComparison=millis();

    break;

    case 2:

    timeComparison=(millis())-timeComparison;

     if (timeComparison <2000)

      {

       ON=1;

       OFF=0;

       timeComparison=millis();

       }

       else{

         count=0;

          }

       break;

  case 3:

    timeComparison=(millis())-timeComparison;

     if (timeComparison <2000)

      {

       ON=0;

       OFF=1;

       count =0;

       }

       else{

         count=0;

          }

       break;

   default:

   count=0;

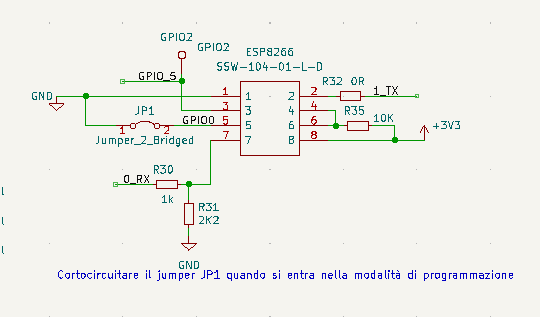
   break;

}

  }

## ESP8266 interface

It is possible to use a ESP8266 Wi-Fi module by mounting the component in J4 socket. Place the module so that it is all within the shield.



Before programming the module, load the sketch below into the ATMEGA328P microcontroller of the Arduino Uno board so that there are no conflicts at the serial communication level (pin 0 and pin 1 of the Arduino UNO board) The sketch also allows you to command the relay based on the state of the GIPO5 pin.

const int EspOutput = 5;    // ESP8266 output

const int rele = 4;      // the number of the LED pin

const int led= 13;

void setup() {

 pinMode(EspOutput, INPUT);

 pinMode(rele, OUTPUT);

 pinMode(led, OUTPUT);

 // put your setup code here, to run once:

}

void loop() {

 if (digitalRead(EspOutput))

 {

   delay(500);

  if (digitalRead(EspOutput))

  {

    digitalWrite(rele,HIGH);

    digitalWrite(led,HIGH);

  }

  else

 {

   digitalWrite(led,LOW);

   digitalWrite(rele,LOW);

 }

 }

}

The R29 resistor allows you to select whether to connect a digital output of the module to the GPIO5 pin of the Arduino Uno board. In this case the pin of the Arduino board must be configured as an input.

For the setting of the module, refer to the video ( It is in English) <https://www.youtube.com/watch?v=AvSCAxbqvvE>.

Before programming the Wi-Fi module, remove power from the shield and insert a jumper in correspondence with the JP1 connector. At this point, supply power again and program the microcontroller using the "Blynk\_NodeMCU\_ESP8266" sketch by selecting "Generic ESP8266 module" as board. It is essential to replace the parameters BLYNK\_TEMPLATE\_ID and BLYNK\_DEVICE\_NAME with the values ​​provided when creating the application through the portal <https://blynk.cloud/>