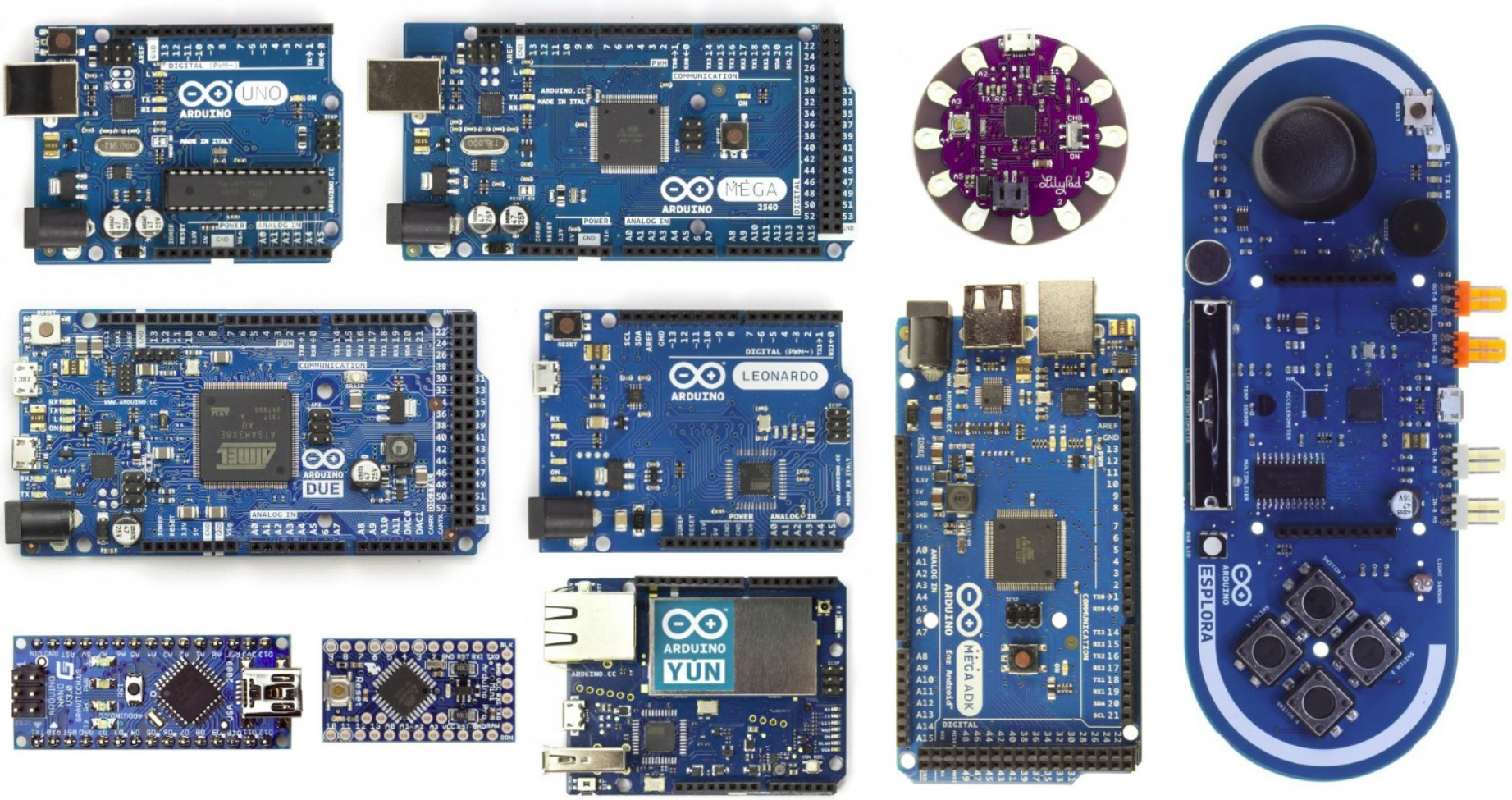
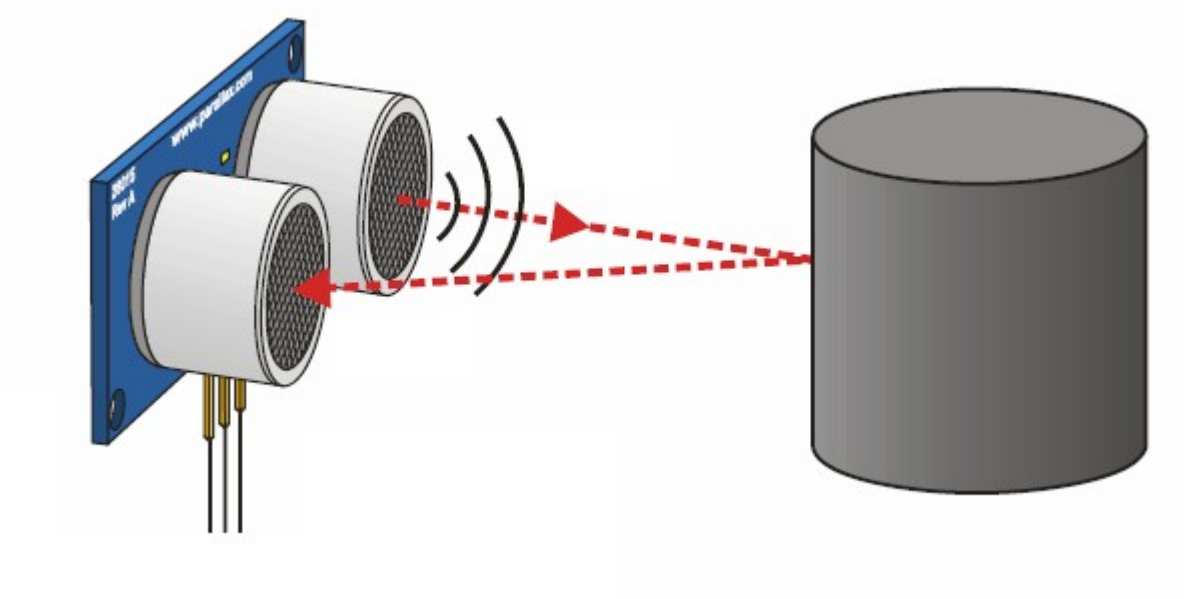
Curso Arduíno – Aula 5



Sensor Ultrassom HC-SR04

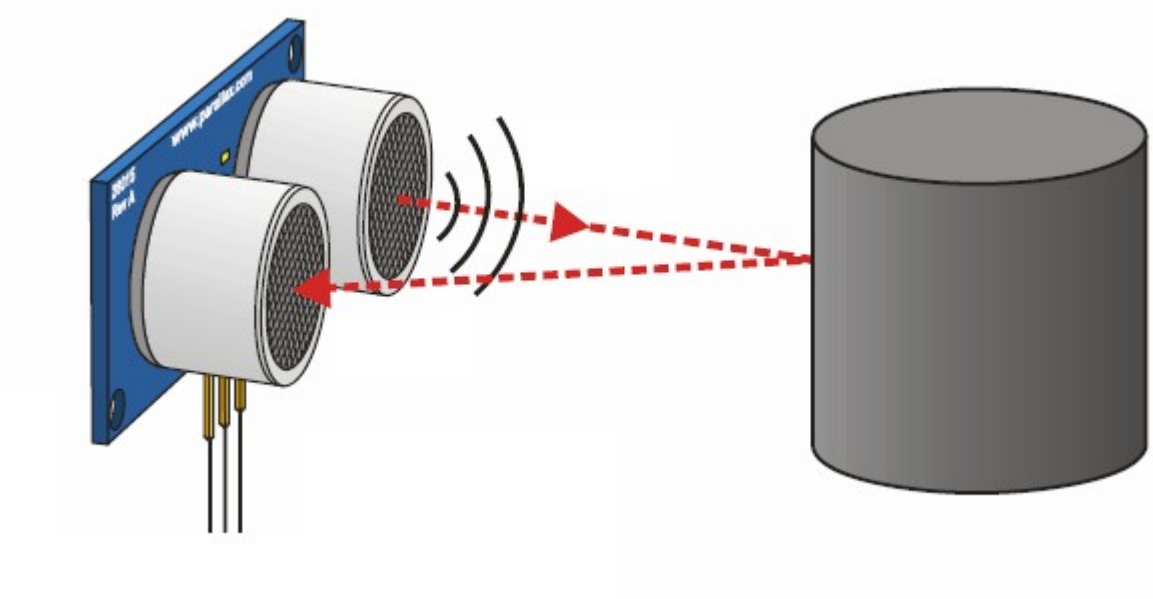
O sensor ultrassom é amplamente utilizado em aplicações onde se deseja medir distâncias ou evitar colisões, como na robótica móvel.



Fonte: http://blog.vidadesilicio.com.br/arduino/sensor-ultrassom-hc-sr04/

Sensor Ultrassom HC-SR04

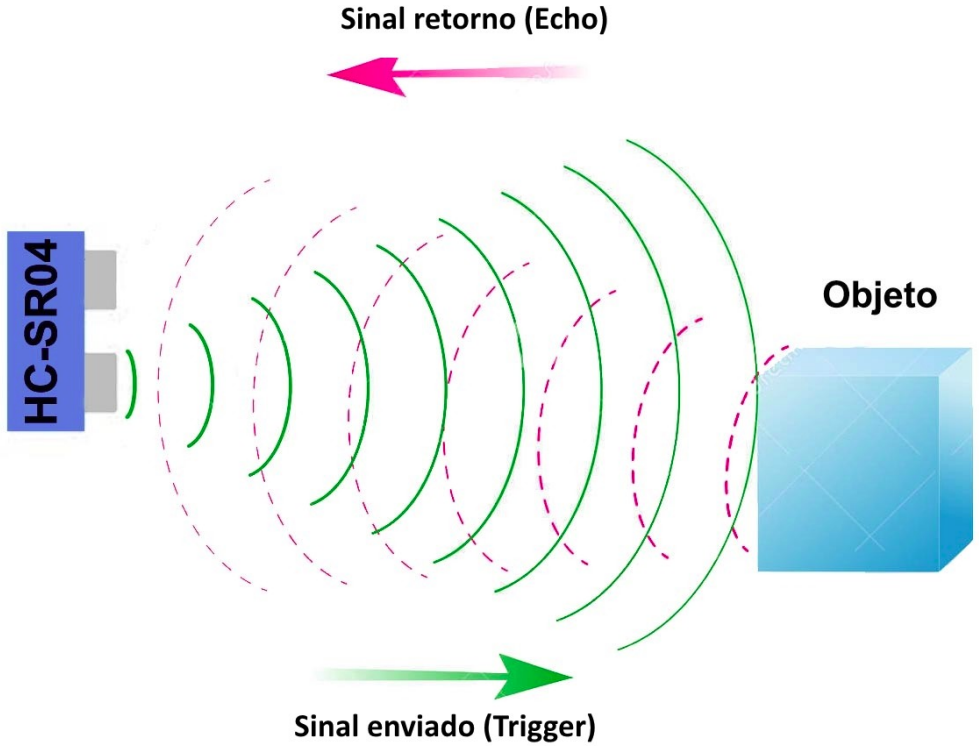
O Sensor Ultrassônico HC-SR04 é um componente muito comum em projetos com Arduíno, e permite que você faça leituras de distâncias até 2 metros, com precisão de 3 mm.



Fonte: http://blog.vidadesilicio.com.br/arduino/sensor-ultrassom-hc-sr04/

Funcionamento do Ultrassom

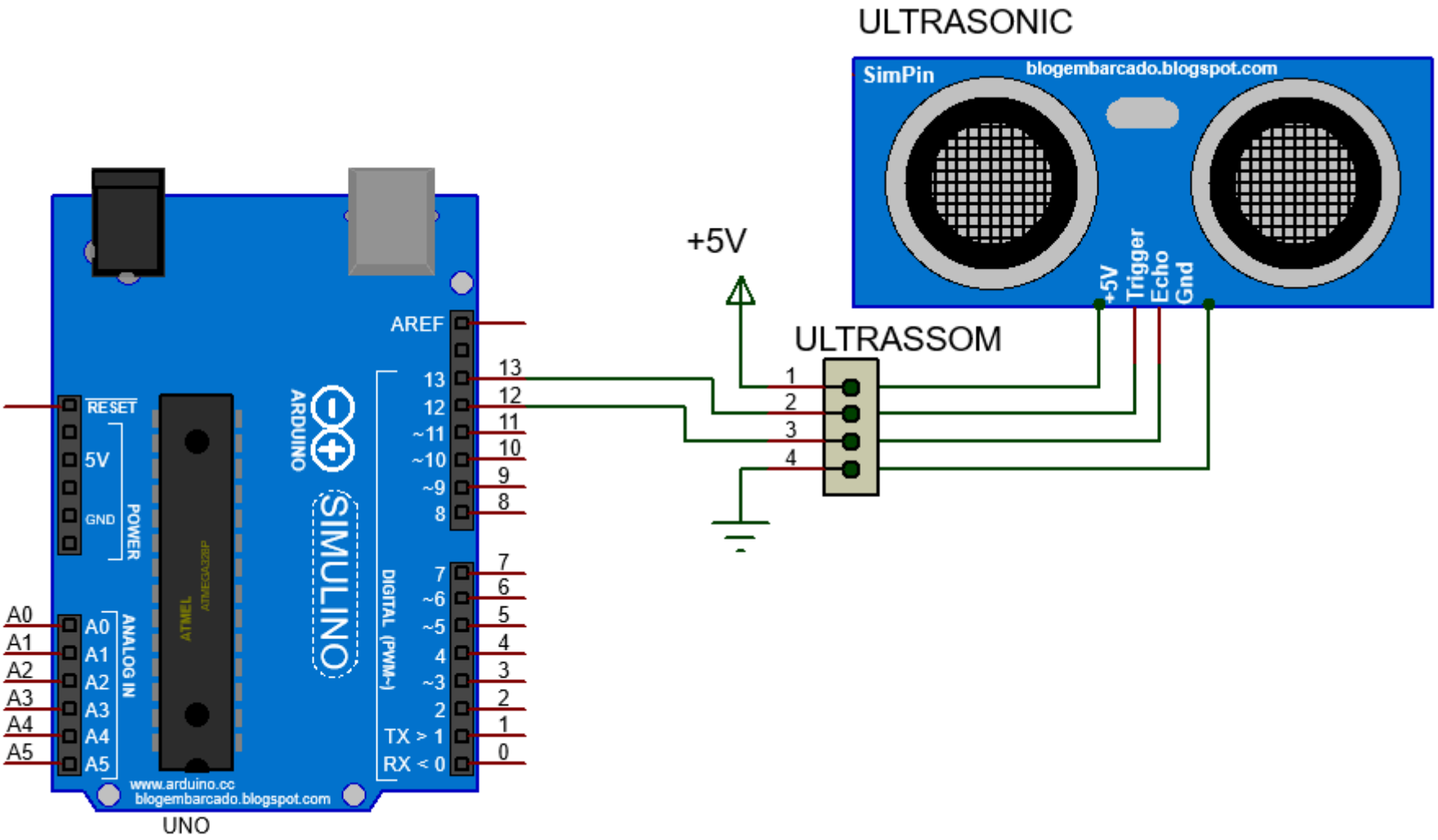
O funcionamento do HC-SR04 se baseia no envio de sinais ultrassônicos pelo sensor, que aguarda o retorno (echo) do sinal, e com base no tempo entre envio e o retorno do sinal é calcula a distância entre o sensor e o objeto detectado.



Pinagem do Sensor Ultrassom

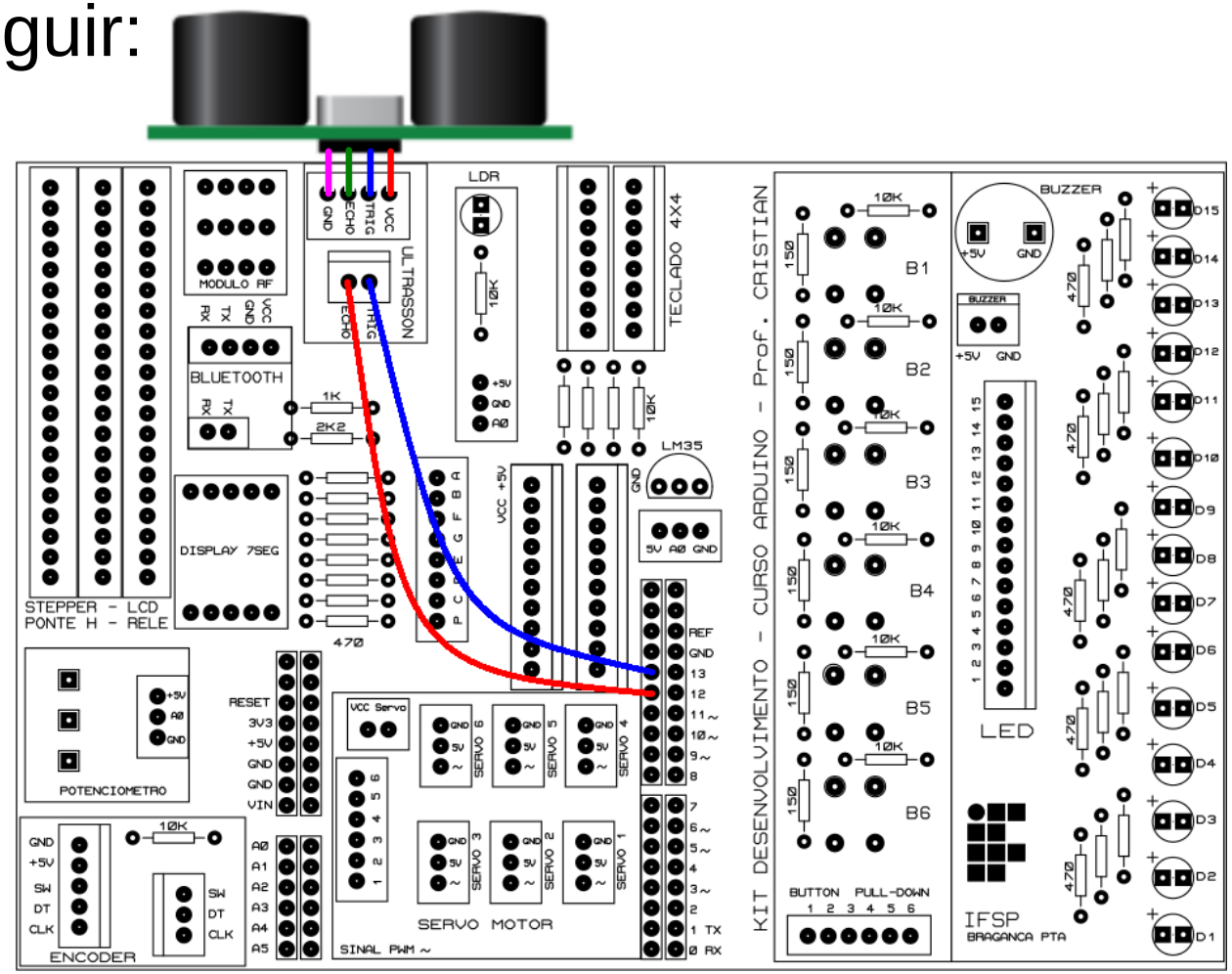
- Vcc – Deve ser conectado a um pino 5V do Arduíno
- Trig – Deve ser conectado a um pino digital configurado como **saída** (**OUTPUT**).
- Echo – Dever ser conectado a um pino digital configurado como **entrada** (**INPUT**).
- Gnd – Dever ser conectado a um pino GND do Arduíno

Prática 15 - Medindo distância com Sensor Ultrassom



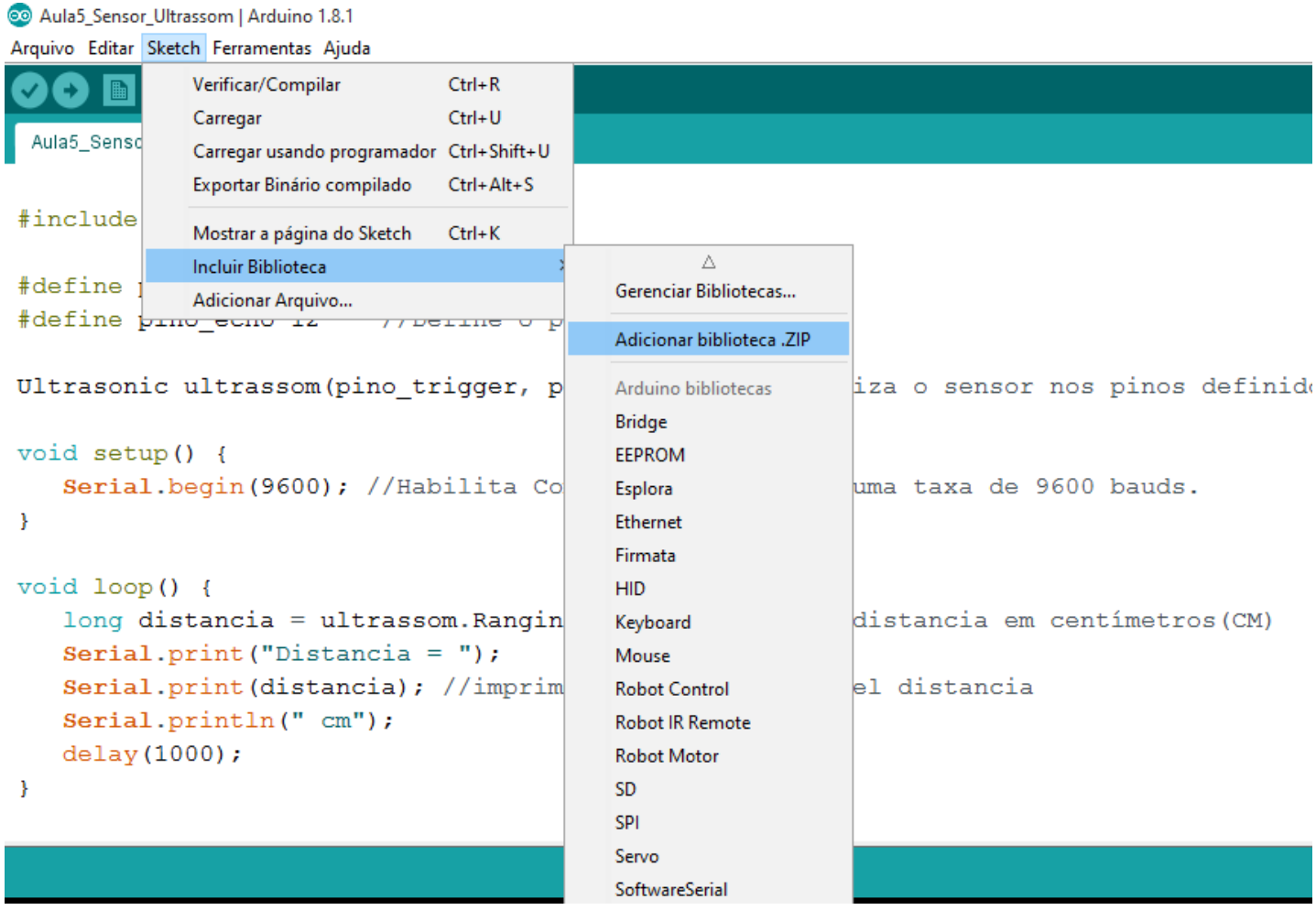
Prática 15 - Medindo distância com Sensor Ultrassom

A ligação do circuito deve ser realizada como visto a seguir:



Biblioteca do Sensor Ultrassom

Antes de iniciarmos o programa é necessário incluir a biblioteca Ultrasonic.h incluindo o arquivo Ultrasonic.zip



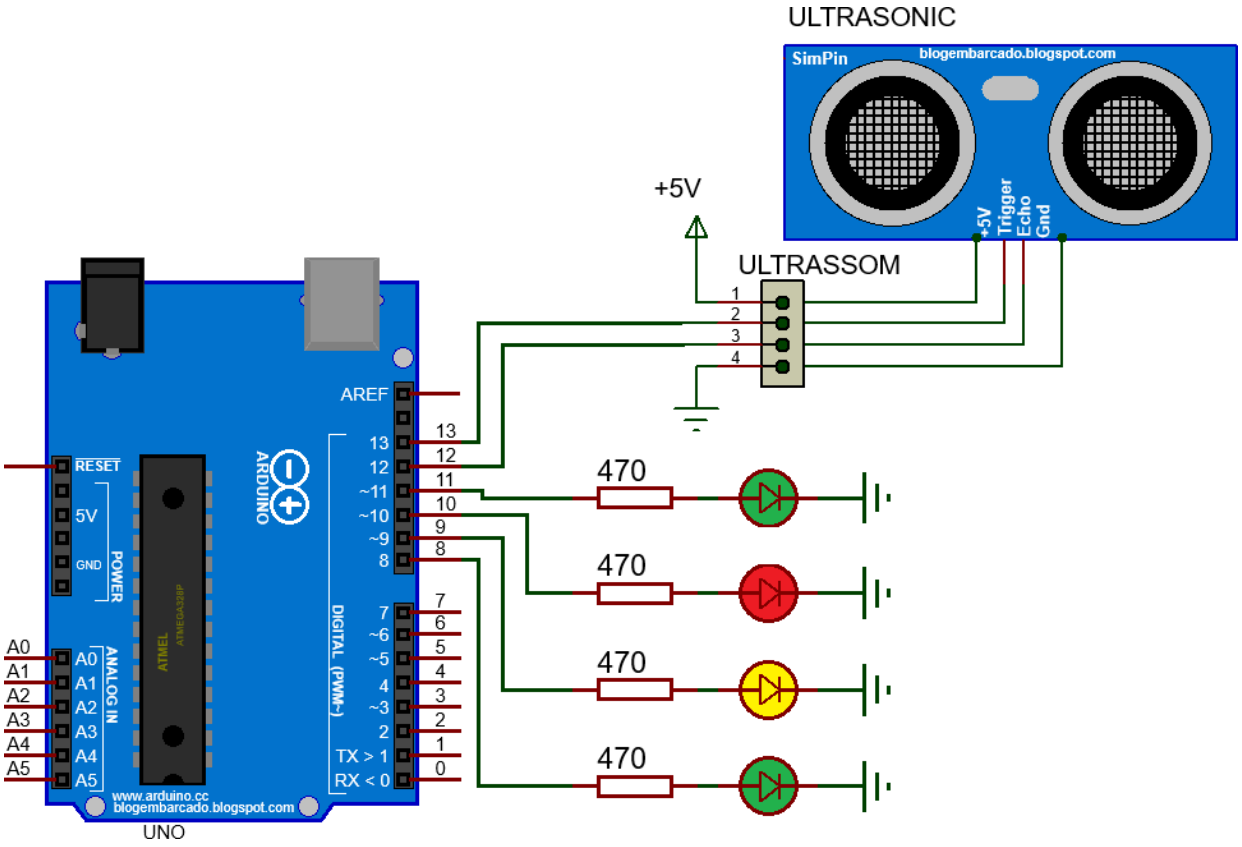
Prática 15 - Medindo distância com Sensor Ultrassom

Programa para medir distância de objetos a frente do sensor ultrassom, em centímetro.

```
#include <Ultrasonic.h>

#define pino_trigger 13 //Define o pino para o trigger
#define pino_echo 12    //Define o pino para o echo

Ultrasonic ultrassom(pino_trigger, pino_echo);//Inicializa o sensor nos pinos definidos

void setup() {
   Serial.begin(9600); //Habilita Comunicação Serial a uma taxa de 9600 bauds.
}

void loop() {
   long distancia = ultrassom.Ranging(CM);// retorna a distancia em centímetros(CM)
   Serial.print("Distancia = ");
   Serial.print(distancia); //imprime o valor da variável distancia
   Serial.println(" cm");
   delay(1000);
}
```

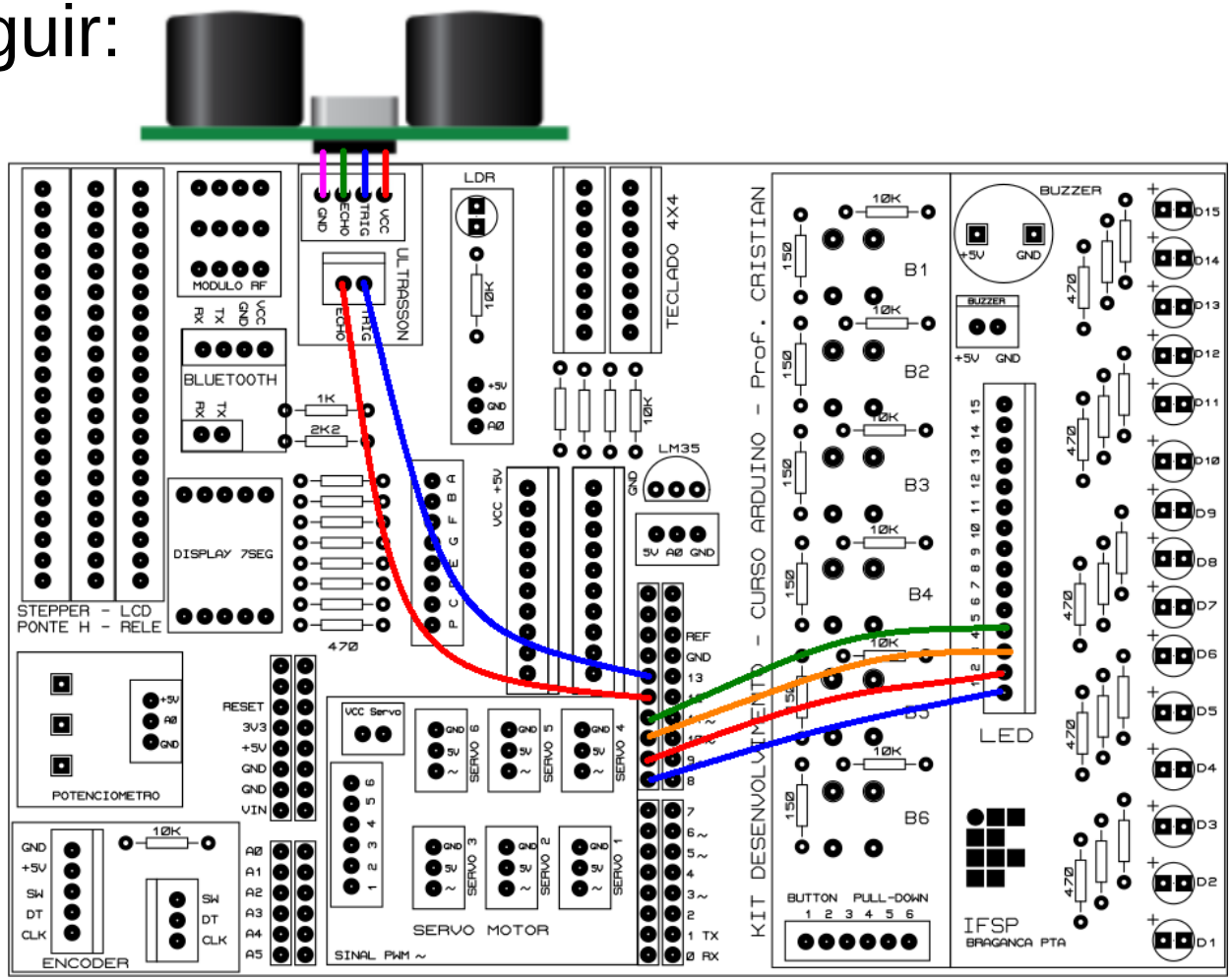
Prática 16 - Medindo distância com Sensor Ultrassom

– Construa um programa que acenda os LEDs conforme um objeto se aproxima do sensor.



Prática 16 - Medindo distância com Sensor Ultrassom

A ligação do circuito deve ser realizada como visto a seguir:



Emissão de Sons pelo Arduíno

No Arduíno podemos emitir sons utilizando a função "**tone**"

A função **tone** tem o seguinte formato :

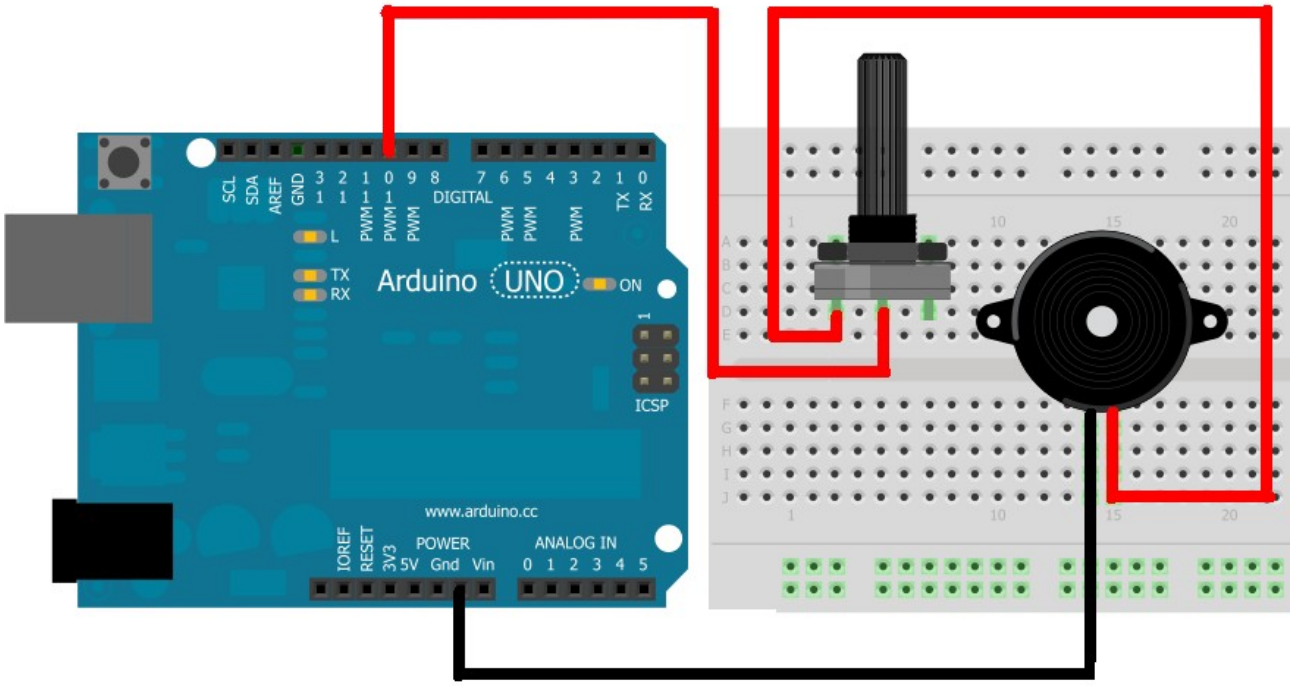
tone(pino, frequência, duração)

onde:

- **pino** - Pino a frequência será gerada para o alto-falante.
- **frequência** - A frequência em Hertz da nota.
- **duração** - Duração em milissegundos da Nota_(opcional).

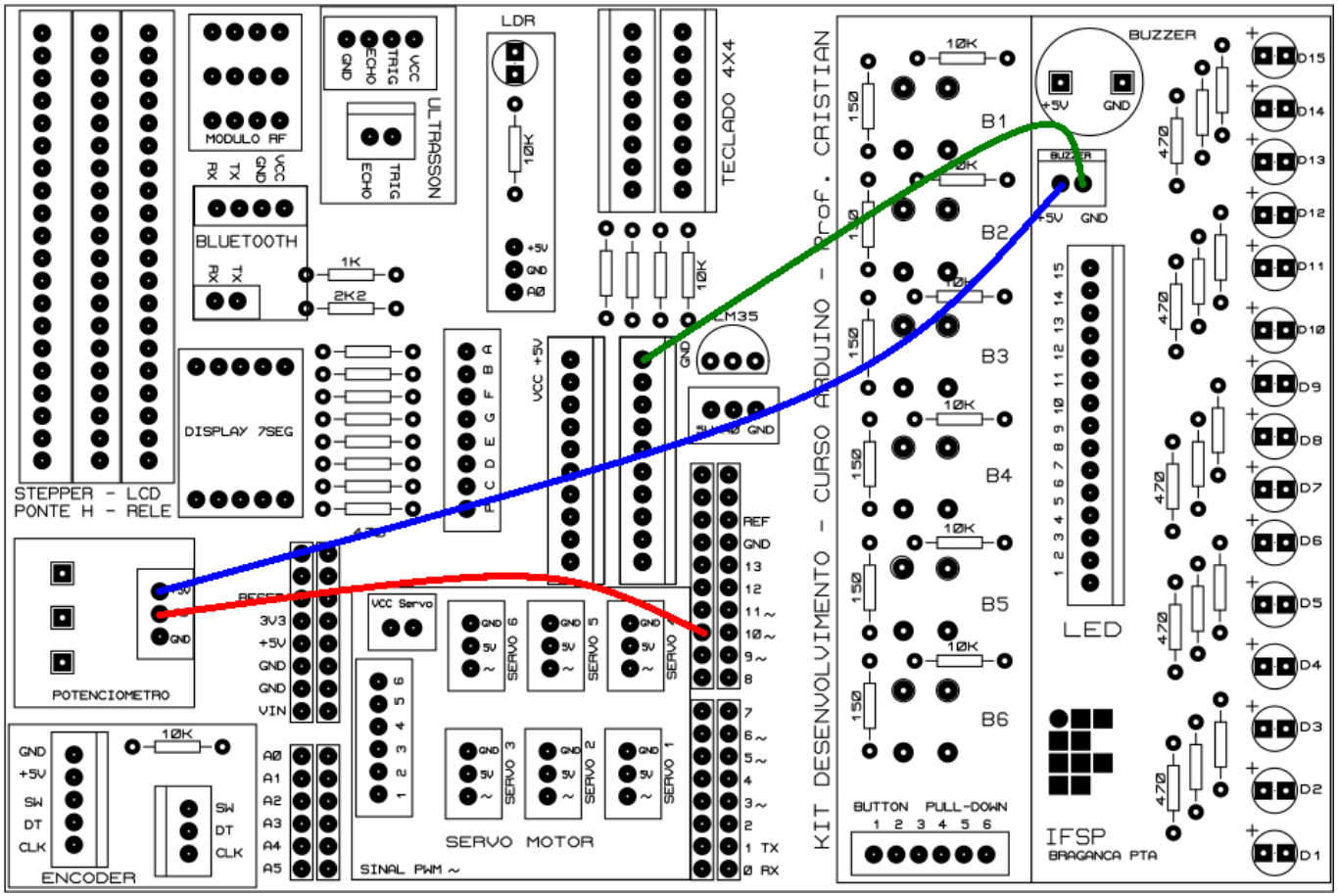
Emissão de Sons pelo Arduíno

A seguir a ligação é feita com o pino 10 do Arduíno ligado ao "+" do buzzer, passando pelo potenciômetro, e o outro pino do buzzer ligado ao GND :
O potenciômetro é utilizado para controle do volume no buzzer.



Prática 17 – Canção Dó Re Mi

A ligação do circuito deve ser realizada como visto a seguir:



Prática 17 – Canção Dó Re Mi

```
void setup() {
pinMode(10,OUTPUT);} //Pino do buzzer

void loop(){
    delay(2000);
    tone(10,262,200); //DO
    delay(220);
    tone(10,294,300); //RE
    delay(220);
    tone(10,330,300); //MI
    delay(220);
    tone(10,349,300); //FA
    delay(330);
    tone(10,349,300); //FA
    delay(330);
    tone(10,349,300); //FA
    delay(330);
    tone(10,262,100); //DO
    delay(220);
    tone(10,294,300); //RE
    delay(220);
    tone(10,262,100); //DO
    delay(220);
    tone(10,294,300); //RE
    delay(330);
    tone(10,294,300); //RE
    delay(330);
    tone(10,294,300); //RE
    delay(330);
    tone(10,262,200); //DO
    delay(220);
    tone(10,392,200); //SOL
    delay(220);
    tone(10,349,200); //FA
    delay(220);
    tone(10,330,300); //MI
    delay(330);
    tone(10,330,300); //MI
    delay(330);
    tone(10,330,300); //MI
    delay(330);
    tone(10,262,200); //DO
    delay(220);
    tone(10,294,300); //RE
    delay(220);
    tone(10,330,300); //MI
    delay(220);
    tone(10,349,300); //FA
    delay(330);
    tone(10,349,300); //FA
    delay(330);
    tone(10,349,300); //FA
    delay(330);
}
```

Música com a tabela pitches.h

Cada nota musical nada mais é do que uma frequência que esta dentro da faixa audível do ouvido da maioria de nós que é entre 20Hz a 20.000Hz.

Com base nas notas musicais, o arquivo pitches.h foi criado para construir canções no Arduíno, com valores aproximados das frequências das notas convertidas, com um nome mais fácil de lembrar.

Por exemplo a frequência 440 é a nota NOTE_A4 e será assim que ela vai ser chamada na matriz melodia[ ] .

Música com tabela pitches.h

A duração de tempo de cada nota vai ser representada no programa pelos valores colocados dentro da matriz tempoNotas[ ].

Símbolo na partitura	Valor de tempo de duração da Nota no Programa
Semibreve	*1*
Mínima	*2*
Semínima	*4*
Colcheia	*8*
Semicolcheia	*16*
Fusa	*32*
Semifusa	*64*

Música com tabela pitches.h

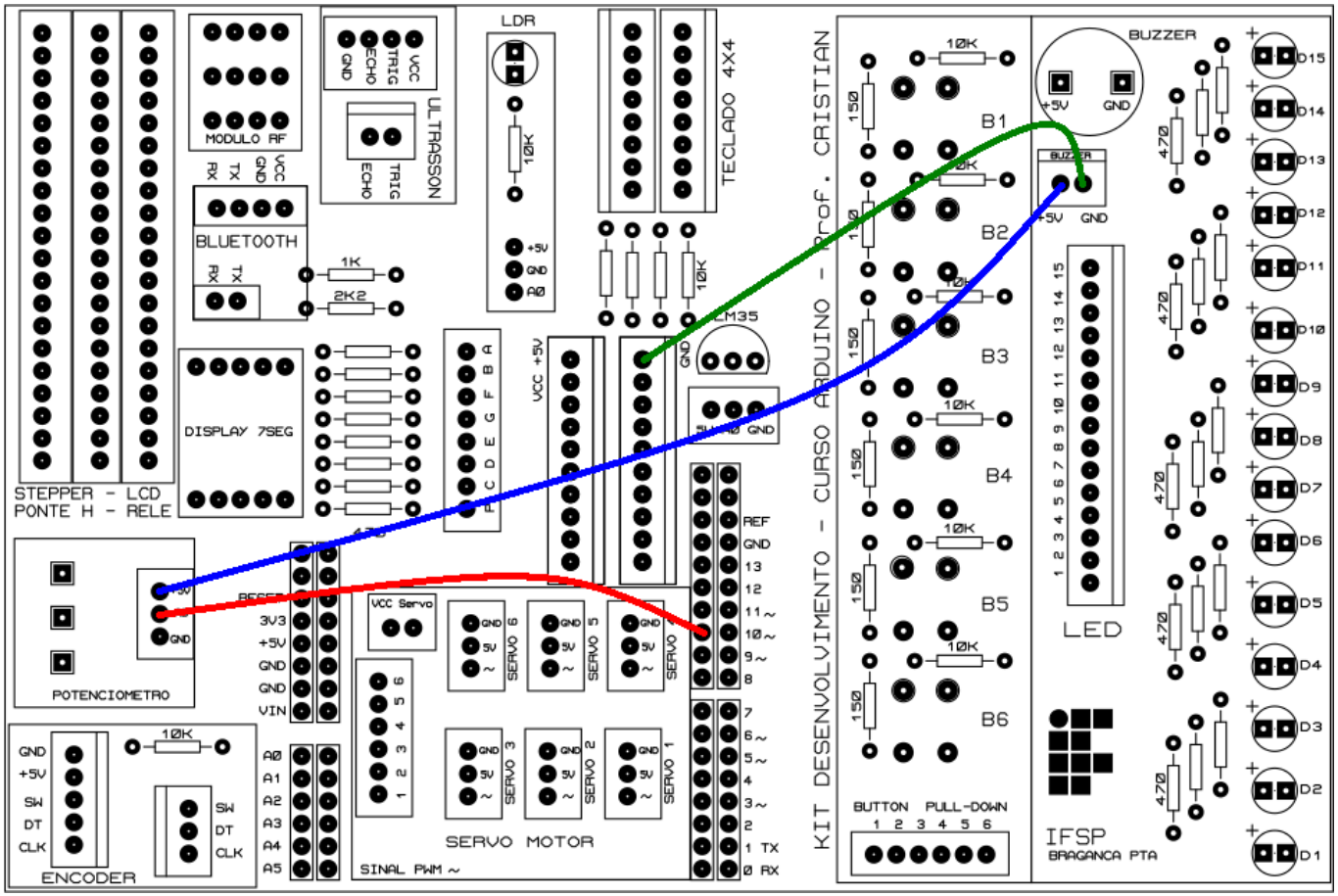
As duas matrizes, melodia[ ] e tempoNotas[ ], trabalham em conjunto e devem receber a mesma quantidade de posição, pois para cada nota ou pausa deve haver seu respectivo tempo de duração e ambos devem ser colocados consecutivamente.

Mas para que o programa executado reconheça as notas é necessário inserir na biblioteca do seu Arduíno o arquivo **pitches.zip** .

A seguir veremos um exemplo com a prática 17.

Prática 18 – Canções e Melodias

Continue na mesma montagem, utilizando o potenciômetro para controle do volume do buzzer.



Prática 18 – Canções e Melodias

```
#include "pitches.h"
#define NO_SOUND 0 // Nota que ajudar a fazer as pausas durante as músicas.

// Notas que devem ser tocadas ordenadamente;

int melodia[] ={
  NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4
};
// Duração das Notas: Colcheia:8; Semínima: 4; Mínima:2; Semibreve:1
int tempoNotas[] ={
    4, 8, 8, 4, 4, 4, 4, 4
};

const int compasso = 1000; // Altera o compasso da música
void setup(){
  //o número 7 indica quantas notas tem a nossa matriz.
  for (int Nota = 0; Nota <7; Nota++){
    //Tempo = compasso dividido pela indicação da matriz tempoNotas.
    int tempo = compasso/tempoNotas[Nota];
    //Toca a nota indicada pela matriz melodia durante o tempo.
    tone(10, melodia[Nota],tempo);
    // Para distinguir as notas adicionamos um tempo entre elas (tempo da nota + 20%).
    delay(tempo*1.2);
  }
}
void loop(){
  //Não é necessária a repetição pois a mesma será feita pelo botão Reset.
}
//Fim de Programa
```

Prática 19 – Hinos de Times

Nesta prática temos 4 exemplos com hinos de times de São Paulo.

Toque um dos hinos como exemplo.

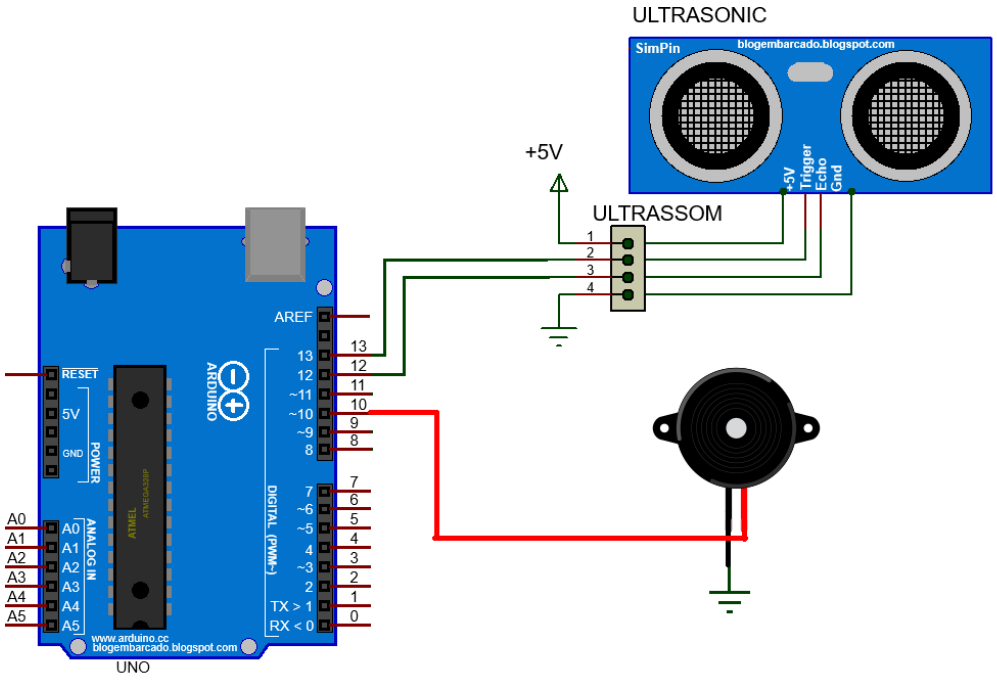
- Hino do Santos;
- Hino do Corinthians;
- Hino do Palmeiras;
- Hino do São Paulo;

Caso o time do seu coração não seja nenhum desses, no link abaixo ensino como construir os hinos.

http://labdegaragem.com/profiles/blogs/hino-de-times-de-futebol-de-sao-paulo-tocados-pelo-arduino

Prática 20 – Sensor de Ré para Automóveis

Agora com o sensor ultrassom e com o buzzer, construa um programa que trabalhe como um sensor de Ré para automóveis, onde quanto mais próximo do obstáculo o sinal do buzzer toca mais rápido conforme se aproxima. Utilize o tom de beep **tone(10,2600,50)**



Prática 20 – Sensor de Ré para Automóveis

Nessa prática, onde usaremo um som de “*bip*”, não é necessário o uso do potenciômetro.