

Vamos montar o sensor de ré, como o apresentado na feira de ciências.

NOTA IMPORTANTE: NESTA AULA, TERÁ ATIVIDADE PARA A PRÓXIMA AULA.

## MATERIAIS

Você vai precisar de:

- 1 sensor ultrassônico;
- 1 buzzer;
- Fios.



Figura 1: sensor ultrassônico.

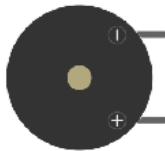


Figura 2: Buzzer.

## MONTAGEM DO CIRCUITO

Na Figura 3 temos a montagem do circuito. Você pode colocar o ultrassom em outras direções: a única coisa que importa é que as conexões estejam certas.

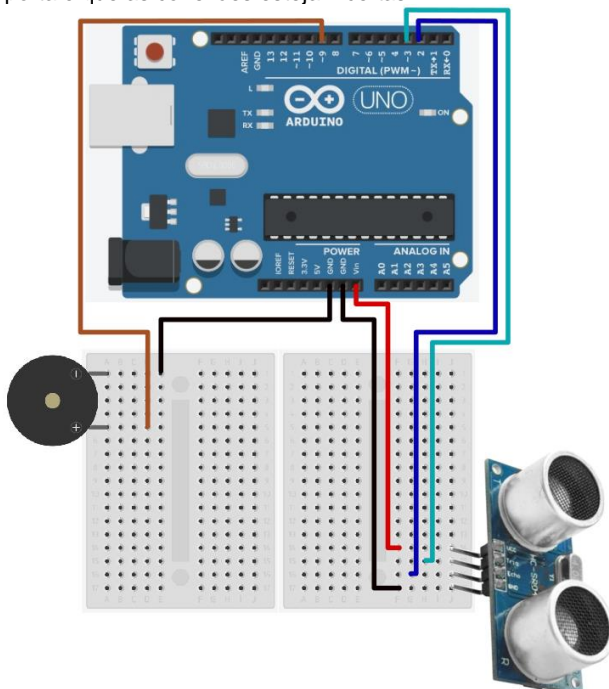


Figura 3: Montagem do circuito.

## CÓDIGO

Lembre-se que é mais fácil [copiar este código pelo GitHub](#).

```
int echo = 2, trigger = 3, buzzer = 9,
buildInPin = 13;
long tempo;
void setup() {
    pinMode(echo, INPUT);
    pinMode(trigger, OUTPUT);
    pinMode(buzzer, OUTPUT);
    pinMode(buildInPin, OUTPUT);
    Serial.begin(9600);
}
void loop() {
    //enviando o pulso TTL
    //vamos forçar o desligamento do
    trigger
    digitalWrite(trigger, LOW);
    //aqui usamos uma nova função para
    esperar microssegundos
    delayMicroseconds(2);
    //inicialndo o pulso TTL
    digitalWrite(trigger, HIGH);
    //aguardando 10 milissegundos
    delayMicroseconds(10);
    //finalizando o pulso TTL
    digitalWrite(trigger, LOW);
    //nova função: para medir o pulso
    //TTL enviada pelo módulo
    tempo = pulseIn(echo, HIGH);
    //Note que temos que informar a porta
    //e se o sinal lido é alto ou baixo
    //vamos exibir isso no serial monitor
    Serial.print("O tempo de medida do eco
    é de ");
    Serial.print(tempo);
    Serial.println(" microssegundos");
    //pulando linhas e esperando
    //1 segundo para fazer outra medida
    Serial.println();
    if (tempo < 250) {
        digitalWrite(buildInPin, HIGH);
        tone(buzzer, 555);
    } else if (tempo < 600) {
        digitalWrite(buildInPin, HIGH);
        noTone(buzzer);
        tone(buzzer, 555);
        delay(100);
        digitalWrite(buildInPin, LOW);
        noTone(buzzer);
        delay(50);
    } else if (tempo < 900) {
        digitalWrite(buildInPin, HIGH);
        noTone(buzzer);
        tone(buzzer, 555);
        delay(200);
        digitalWrite(buildInPin, LOW);
        noTone(buzzer);
        delay(100);
    } else if (tempo < 1200) {
        digitalWrite(buildInPin, HIGH);
```

## PROFESSOR DANILO

```
noTone(buzzer);  
tone(buzzer, 555);  
delay(300);  
digitalWrite(buildInPin, LOW);  
noTone(buzzer);  
delay(200);  
} else if (tempo < 1700) {  
digitalWrite(buildInPin, HIGH);  
noTone(buzzer);  
tone(buzzer, 555);  
delay(400);  
digitalWrite(buildInPin, LOW);  
noTone(buzzer);  
delay(300);  
} else if (tempo < 2000) {  
digitalWrite(buildInPin, HIGH);  
noTone(buzzer);  
tone(buzzer, 555);  
delay(500);  
digitalWrite(buildInPin, LOW);  
noTone(buzzer);  
delay(400);  
} else {  
digitalWrite(buildInPin, LOW);  
noTone(buzzer);  
}  
}
```

## ANÁLISE DO CÓDIGO

Note que, no início, declaramos as portas onde o buzzer será conectado, assim como as portas do sensor ultrassônico. A variável buildInPin é para que o LED incorporado na placa, que é conectado ao pino digital 13, pisque de acordo com regras elaboradas durante o código.

```
int echo = 2, trigger = 3, buzzer = 9,  
buildInPin = 13;  
long tempo;  
void setup() {  
pinMode(echo, INPUT);  
pinMode(trigger, OUTPUT);  
pinMode(buzzer, OUTPUT);  
pinMode(buildInPin, OUTPUT);  
Serial.begin(9600);  
}
```

As primeiras linhas do código, dentro do setup, enviam um pulso de 10 microssegundos para que o sensor ultrassônico comece a medir. Isso é feito no código abaixo. Note que inicialmente desligamos a porta 3 (trigger) por 2 milissegundos para evitar algum erro, conforme recomenda o fabricante. Depois, a porta 3 é ligada, o Arduino espera 10 microssegundos (ou seja, 0,00001 segundo) e então desliga.

```
void loop() {  
digitalWrite(trigger, LOW);  
delayMicroseconds(2);  
digitalWrite(trigger, HIGH);  
delayMicroseconds(10);  
digitalWrite(trigger, LOW);  
}
```

Feito isso, o sensor ultrassônico envia um pulso ultrassônico e mede o tempo de ida e volta. Para informar isso a Arduino, o

## ROBÓTICA – 9º ANO 20/10/2022

sensor liga o pino Echo por um tempo igual ao tempo que demora para o ultrassom ir e voltar. Por exemplo, se demorar 10 milissegundos para o pulso ultrassônico ir e voltar, então o módulo ultrassônico irá ligar por 10 milissegundos o pino Echo.

Usaremos a função pulseIn para ler isso, conforme o trecho abaixo.

```
tempo = pulseIn(echo, HIGH);
```

Podemos ver este tempo no monitor serial através dos comandos abaixo.

```
Serial.print("O tempo de medida do eco  
é de ");  
Serial.print(tempo);  
Serial.println(" microssegundos");  
Serial.println();
```

Por fim, criamos diversas condições baseadas no tempo de retorno do som. Esses valores podem ser escolhidos colocando a mão na frente do sensor e olhando o valor lido no monitor serial.

Vamos pegar apenas um trecho do código:

```
if (tempo < 250) {  
digitalWrite(buildInPin, HIGH);  
tone(buzzer, 555);
```

No trecho acima, se o tempo de ida e volta for menor que 250 milissegundos, o buzzer começa a emitir um som de 555 Hz de frequência de forma contínua. Caso contrário, se o tempo for menor que 600 (porém maior que 200, devido ao "else"), o buzzer começa a tocar de forma intermitente, deixando ligado por 100 milissegundos. Note que esta parte começa com noTone para que o buzzer pare de tocar caso esteja tocando.

```
} else if (tempo < 600) {  
digitalWrite(buildInPin, HIGH);  
noTone(buzzer);  
tone(buzzer, 555);  
delay(100);
```

Note que o trecho acima o Arduino liga o LED incorporado e no trecho abaixo ele desliga, para que o LED pisque conforme o som emitido pelo buzzer. Note também que, após o noTone, o Arduino espera 50 milissegundos fazendo com que o buzzer apite por 100 milissegundos e depois fique desligado por 50 milissegundos.

```
digitalWrite(buildInPin, LOW);  
noTone(buzzer);  
delay(50);  
}
```

O código que se segue mantém este padrão: liga o LED; desliga o buzzer; liga o buzzer; espera um pouco; desliga o LED; desliga o buzzer; e por fim espera novamente.

Você pode escolher os trechos para mudar a forma de apitar, porém este é o comportamento que esperamos de um sensor de ré que nos informa, através de sons intermitentes, quando um obstáculo está a determinada distância.