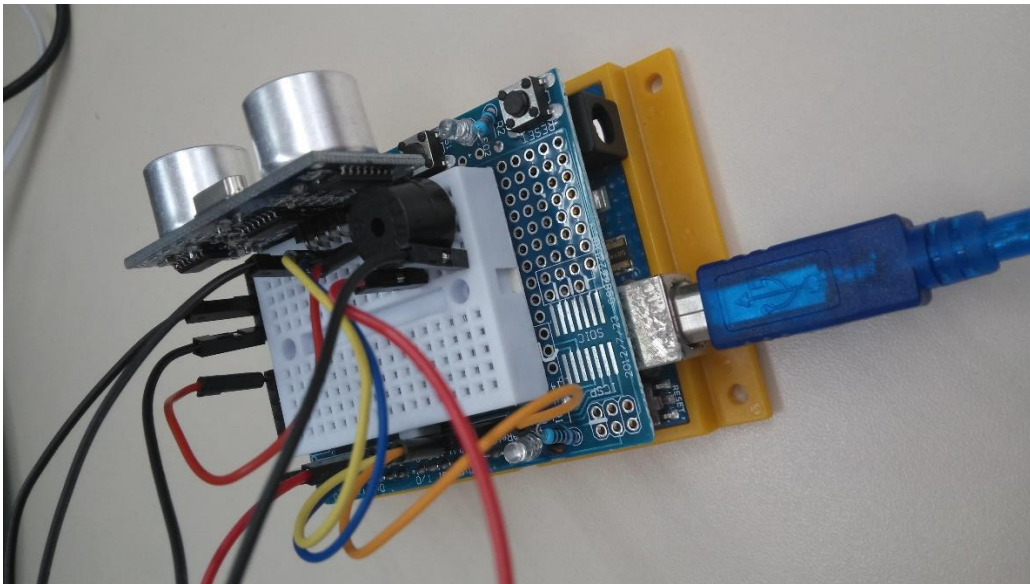


Demo Arduino com Sensor Ultrasonico, Led e Buzzer aplicando MATLAB e SIMULINK

Alberto Y. Shimahara
Janeiro/2020



Introdução

Mostrar a funcionalidade do MATLAB/SIMULINK em gerar prototipagem rápida de geração de código em placa de baixo custo e acesso a periféricos.

Projeto inicial para aplicação de sensor de ultrassom em Arduino, onde ao detectar um objeto efetiva a geração de um aviso sonoro e ativação de um led.

Baseado em apresentação da Mathworks:

Dan Doherty (2020). Mapping Your Surroundings Using MATLAB and Arduino

(<https://www.mathworks.com/matlabcentral/fileexchange/58434-mapping-your-surroundings-using-matlab-and-arduino>), MATLAB Central File Exchange. Retrieved January 17, 2020.

Material Necessário e Software aplicado

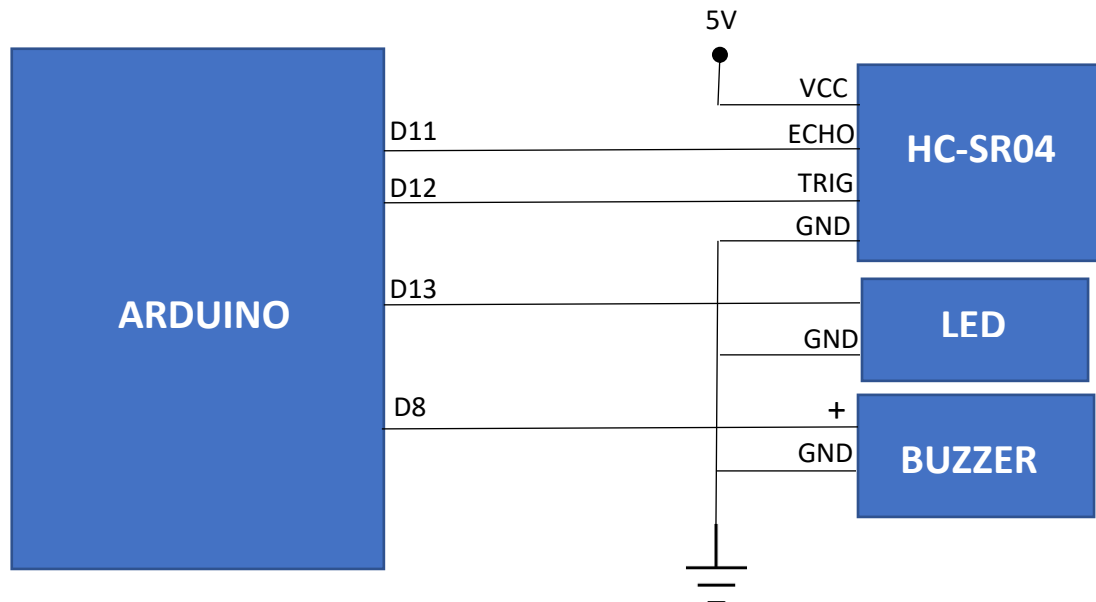
1. Material (Componente)

- i. Arduino Uno
- ii. Sensor Ultrasonico HC-SR04 (Ver Anexo C)
- iii. Led
- iv. Buzzer
- v. ProtoShield
- vi. Jumpers ou cabos de conexão
- vii. Cabo USB A p/ B (uso com Arduino)

2. Software

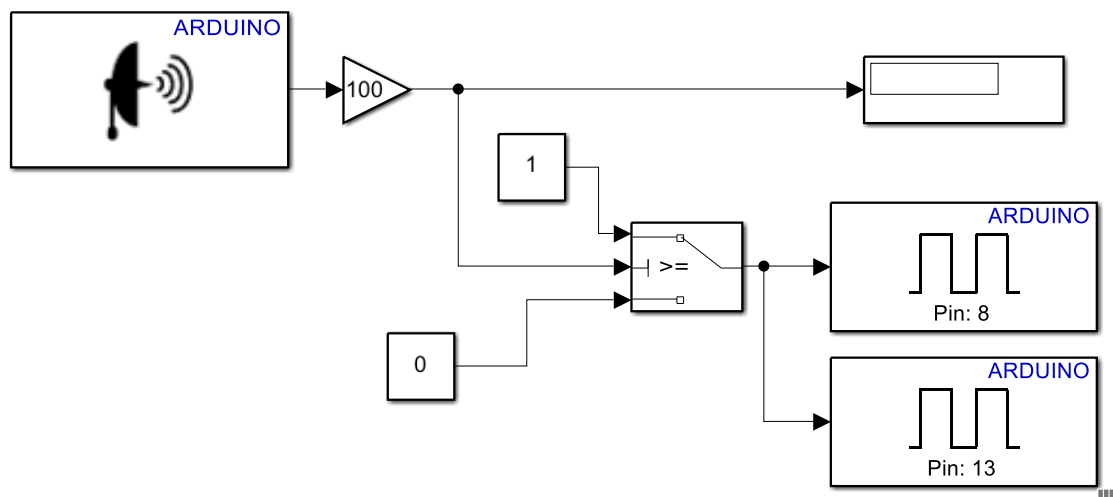
- i. MATLAB R2019b
- ii. MATLAB Support Package for Arduino
- iii. SIMULINK Support Package for Arduino Hardware

Montagem & conexão Arduino (Esquemático)



Como fazer – montar o modelo

1. No MATLAB, efetivar a instalação dos pacotes de suporte a Arduino, inicialmente [MATLAB](#) e posteriormente [SIMULINK](#). Este processo efetiva o download das bibliotecas de blocos e funções MATLAB/SIMULINK; IDE Arduino e drivers adicionais. **Observação: Deve possuir acesso a uma conexão de Internet**. Como instalar veja o anexo A.
2. Efetivar a montagem dos componentes conforme esquemático. Pode se efetivar a montagem diretamente no Arduino sem a necessidade do Protoshield, optamos por usar para facilitar montagem quando estamos em cliente.
 - 2.1. Para testar a conexão do sensor Ultrasonico, pode se aplicar um teste em código Arduino diretamente da IDE; a IDE Arduino instalada pelo suporte estará no seguinte caminho:
C:\ProgramData\MATLAB\SupportPackages\R2019b\3P.instrset\arduinoide.instrset
 - 2.2. Gere um atalho do “Arduino.exe” para sua área de trabalho, ao abri-lo aplique o código existente no Anexo B; desenvolvido por: Rui Santos, <https://randomnerdtutorials.com>.
3. No SIMULINK, abrir um novo modelo, acessar os blocos:



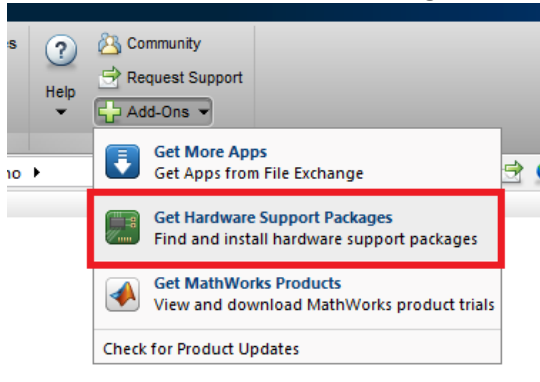
- a. Na Biblioteca de Blocos SIMULINK -> Sub Biblioteca: Simulink Support Package for Arduino Hardware -> Sensors -> **Ultrasonic Sensor**
Clicar no bloco e arrastar na área do modelo.
Configurar os parâmetros do sensor HC-SR04:
 - Numero de pinos com sinal: 2
 - Pino Trigger: 11
 - Pino Echo: 12
 - Sample Time: 0.000301
- b. Na Biblioteca de Blocos SIMULINK -> Sub Biblioteca: Simulink Support Package for Arduino Hardware -> Common -> **Digital Output**
Clicar no bloco e arrastar na área do modelo.
Inserir 2 blocos de saída digital e configurar:
 - Em um, configure o Pin Number: 8; para conectar o buzzer
 - no Outro, configure o Pin Number: 13; para conectar o led

- c. Na Biblioteca de Blocos SIMULINK -> Sub Biblioteca: Sinks -> **Display**
Clicar no bloco e arrastar na área do modelo.
- d. Na Biblioteca de Blocos SIMULINK -> Sub Biblioteca: Math Operations -> **Gain**
Clicar no bloco e arrastar na área do modelo.
Configurar com valor: 100
- e. Na Biblioteca de Blocos SIMULINK -> Sub Biblioteca: Sources -> **Constant**
Clicar no bloco e arrastar na área do modelo.
Inserir 2 blocos de constante e configurar:
 - Em um, valor: 1
 - no Outro, valor: 0
- f. Na Biblioteca de Blocos SIMULINK -> Sub Biblioteca: Signal Routing -> **Switch**
Clicar no bloco e arrastar na área do modelo.
Configurar Threshold: 10
Configurar: Criteria dor passing first input: "u2 >= Threshold"
Esse bloco estabelece que quando o sinal do Sensor de Ultrasom for menor que 10, efetivara a opção de falso no bloco; ou seja não efetiva som e nem led.
Quando maior que 10 efetiva ruído sonor no buzzer e acende o led.
- g. Efetivar a conexão dos sinais conforme ilustração, basta clicar nas saídas dos blocos e conectar (clicar e arrastar) nas respectivas entradas.
- h. Em Model Settings -> Hardware Implementation -> Configure a opção:
Hardware Board -> no menu, opte pelo Arduino Uno. Automaticamente alguns dos parâmetros já se ajustarão ao suporte da placa.
- i. Escolhido a placa, irá se habilitar uma aba "HARDWARE", Clique em "Monitor & Tune"
Sera gerado automaticamente código suportado para o Arduino.
Conectado a placa, será baixado o código na placa e já a inicializando.

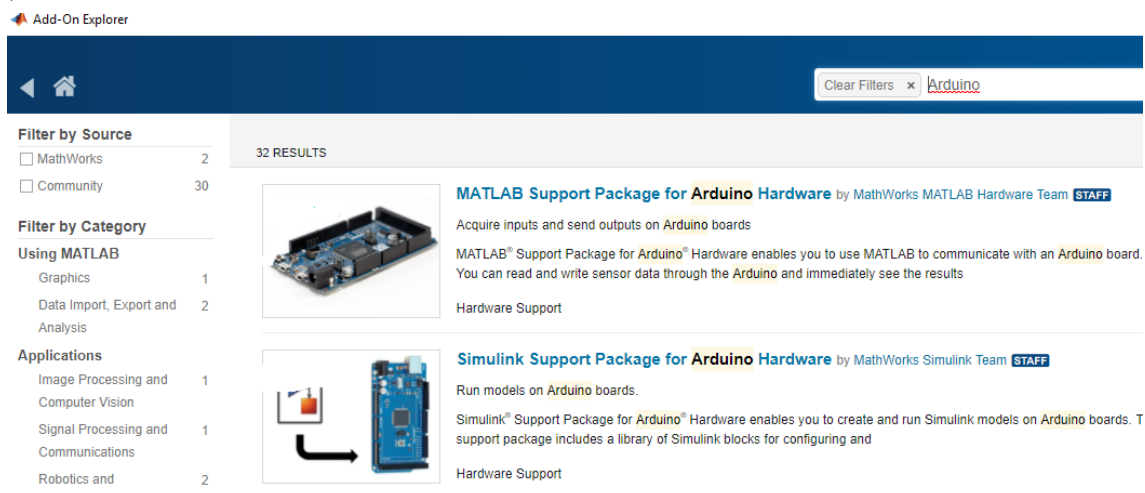
Qualquer problema, contate-nos.

Anexo A

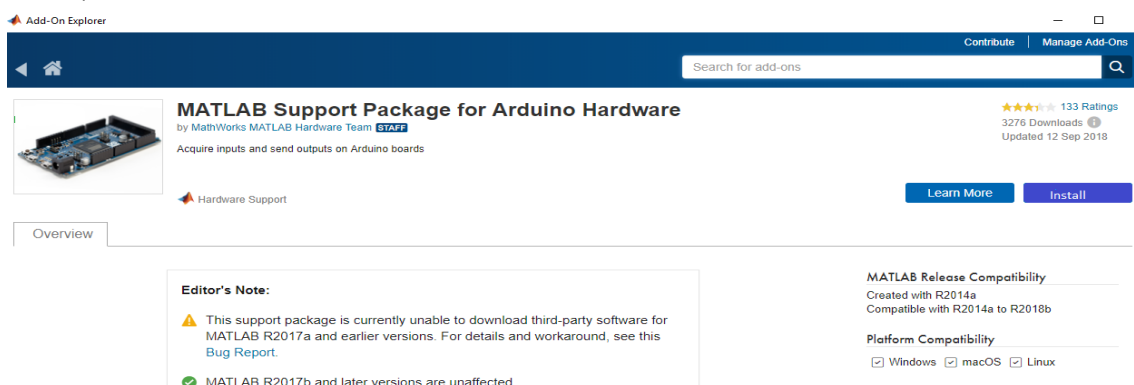
1. Vá ao ícone “Add-on” e em seguida “Hardware Support Package”



2. Efetive a busca por Arduino e escolha a opção a ser instalada, inicialmente MATLAB e posteriormente SIMULINK.



3. Clique em “Install”



4. O processo é efetivado automaticamente, download dos componentes de suporte ao pacote e a instalação

Download and Installation Progress

- ✓ Downloading Support Packages... 100%
- ✓ Downloading Third-Party Packages... 100%
- ✓ Installing Support Packages... 100%
- ✓ Installing Third-Party Packages... 100%
- Configuring your installation

5. Será questionado para configuração do hardware (Arduino) com o MATLAB

Hardware Setup

Arduino Hardware Setup

Do you want to set up your Arduino board connection?

☒ Yes

☐ No

6. Escolha o tipo de conexão que efetivara com o Arduino, neste caso USB.

Hardware Setup

Choose Connection Type

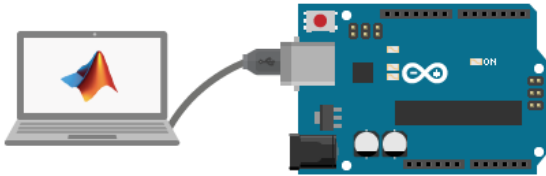
Connect Arduino via USB and choose your connection type.

Supported types:

☒ USB

☐ Bluetooth®

☐ WiFi



Note: Remove any connected Bluetooth device from Arduino board before clicking Next.

[About Your Selection](#)
This connection type determines how the Arduino board will be connected to your host computer after setup.

< Back Cancel Next >

7. Efetive a conexão com o Arduino (informe a placa e a porta aplicada) e a opção de aplicar drivers de componentes, tipo servo motores, I2C, etc.

Hardware Setup

Upload Arduino Server

Connect Arduino to host computer via USB

Choose board:

Choose port:

Select libraries to be included in the server:

- ☐ Adafruit/MotorShieldV2
- ☒ I2C
- ☐ RotaryEncoder
- ☒ SPI
- ☒ Servo
- ☐ ShiftRegister
- ☐ Arduino/MKRMotorCarrier
- ☐ JRodrigoTech/HCSR04
- ☐ PaulStoffregen/OneWire

To begin uploading server to the board, click 'Program'.

Program

About Your Selection
This step uploads a server to Arduino board that allows MATLAB to communicate with it.

What to Consider
Make sure there is no existing connection to the Arduino board in MATLAB workspace before clicking Program.

< Back Cancel Next >

Após escolher as bibliotecas, clicar em “Program”.

Caso ocorra tudo corretamente terá o aviso: **“Success! Click Next to proceed”**

8. Será solicitado para efetivar um teste de conexão. O Arduino deverá estar conectado ao PC/notebook via usb. Clique em **“Test connection”**

Hardware Setup

Test Arduino Connection

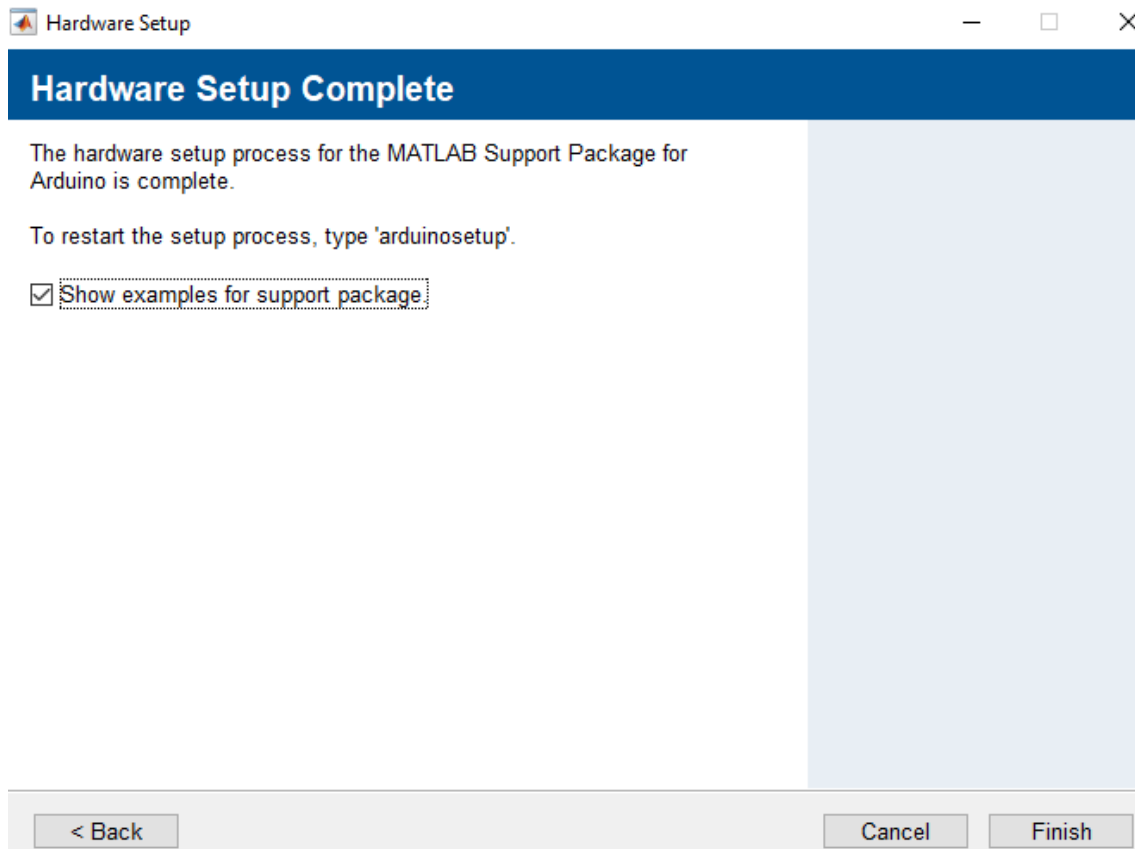
Current Settings:

Connection Type	USB
Port	COM9
Board	Uno
Libraries	I2C, SPI, Servo

Test connection

< Back Cancel Next >

Se ocorrer a conexão sem problemas, abra uma nova tela informando conclusão do processo e acesso ao Help(Ajuda) com exemplos.



9. No MATLAB, para verificar a conexão, no Command Window digite: a = Arduino
Devera obter a seguinte resposta:

a =

arduino with properties:

Port: 'COM3'

Board: 'Uno'

AvailablePins: {'D2-D13', 'A0-A5'}

AvailableDigitalPins: {'D2-D13', 'A0-A5'}

AvailablePWMPins: {'D3', 'D5-D6', 'D9-D11'}

AvailableAnalogPins: {'A0-A5'}

AvailableI2CBusIDs: [0]

Libraries: {'I2C', 'SPI', 'Servo'}

10. No SIMULINK, acesse a Bibiloteca de blocos e devera observar a inserção de uma sub-bibiloteca com o suporte a Arduino.

Simulink Library Browser

Enter search term

Simulink Support Package for Arduino Hardware/Common

Image Acquisition Toolbox

Instrument Control Toolbox

> LTE HDL Toolbox

> Model Predictive Control Toolbox

OPC Toolbox

> Phased Array System Toolbox

> Powertrain Blockset

Report Generator

> RF Blockset

> Robotics System Toolbox

Robust Control Toolbox

SimEvents

> Simscape

> Simulink 3D Animation

> Simulink Coder

> Simulink Coder Support Package for NXP FRDM-KL

> Simulink Control Design

> Simulink Design Optimization

> Simulink Design Verifier

> Simulink Desktop Real-Time

> Simulink Extras

> Simulink Real-Time

> Simulink Requirements

Simulink Support for Arduino MKR Motor Carrier

Simulink Support Package for Arduino Hardware

Common

Ethernet Shield

Sensors

Utilities

WiFi

Simulink Test

Stateflow

> System Identification Toolbox

ARDUINO

Pin: 4

Analog Input

ARDUINO

DAC0

Analog Output

ARDUINO

Pin 2

Continuous Servo Write

ARDUINO

Pin 8

Digital Input

ARDUINO

Pin 9

Digital Output

ARDUINO

Slave: 0xA

I2C Read

ARDUINO

Slave: 0xA

I2C Write

ARDUINO

Pin: 5

PWM

ARDUINO

Port 0

Serial Receive

ARDUINO

Port 0

Serial Transmit

ARDUINO

SS pin 10

SPI WriteRead

ARDUINO

Pin 2

Standard Servo Read

Anexo B

Teste de conexão do Arduino com o sensor Ultrasonico

Fonte: Rui Santos,

<https://randomnerdtutorials.com>

```
/*
 * created by Rui Santos, https://randomnerdtutorials.com
 *
 * Complete Guide for Ultrasonic Sensor HC-SR04
 *
 Ultrasonic sensor Pins:
 VCC: +5VDC
 Trig : Trigger (INPUT) - Pin11
 Echo: Echo (OUTPUT) - Pin 12
 GND: GND
 */

int trigPin = 11; // Trigger
int echoPin = 12; // Echo
long duration, cm, inches;

void setup() {
  //Serial Port begin
  Serial.begin (9600);
  //Define inputs and outputs
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop() {
```

```
// The sensor is triggered by a HIGH pulse of 10 or more
microseconds.
```

```
// Give a short LOW pulse beforehand to ensure a clean
HIGH pulse:
```

```
digitalWrite(trigPin, LOW);
```

```
delayMicroseconds(5);
```

```
digitalWrite(trigPin, HIGH);
```

```
delayMicroseconds(10);
```

```
digitalWrite(trigPin, LOW);
```

```
// Read the signal from the sensor: a HIGH pulse whose
```

```
// duration is the time (in microseconds) from the sending
```

```
// of the ping to the reception of its echo off of an object.
```

```
pinMode(echoPin, INPUT);
```

```
duration = pulseIn(echoPin, HIGH);
```

```
// Convert the time into a distance
```

```
cm = (duration/2) / 29.1; // Divide by 29.1 or multiply by
0.0343
```

```
inches = (duration/2) / 74; // Divide by 74 or multiply by
0.0135
```

```
Serial.print(inches);
```

```
Serial.print("in, ");
```

```
Serial.print(cm);
```

```
Serial.print("cm");
```

```
Serial.println();
```

```
delay(250);
```

```
}
```

ANEXO C

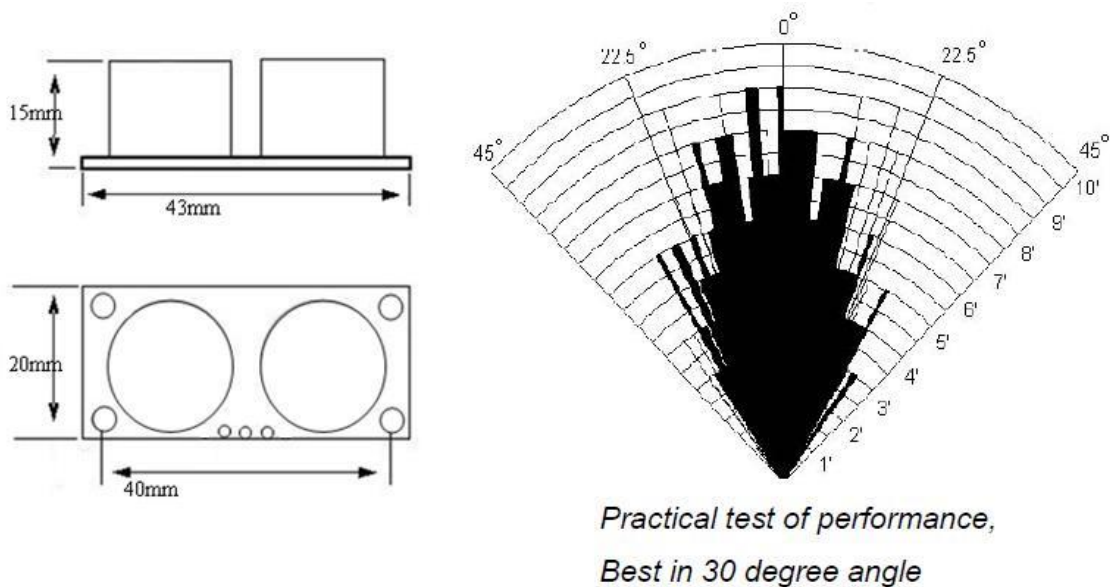
Sensor Ultrasonico HC-SR04 – Conceitos

Fonte: Site Eletrogate

(<https://blog.eletrogate.com/sensor-ultrassonico-hc-sr04-com-arduino/>)

Sensores ultrassônicos são aplicados como um detectores de objetos e são muito populares principalmente na **robótica**, onde são utilizados para identificar obstáculos e corrigir continuamente o trajeto feito por um robô.

Por meio da emissão de sinais Ultrassônicos, é possível especificar a distância do sensor até um determinado obstáculo. O range de atuação do HC-SR04 é da ordem de 4 metros, com distância mínima de medição de 2 cm e incluindo obstáculos dentre de um ângulo de abertura de 15 graus.



HC-SR04 datasheet

Aplicações

Os sensores ultrassônicos podem medir variáveis como enchimento e altura sem ter que entrar em contato com os elementos do meio, o que é uma grande vantagem quando comparado com outros tipos de sensores. Uma outra vantagem é que o sensor ultrassônico não possui sua operação prejudicada pela transparência, poeira, sujeira ou vapores/gases presentes no ambiente. Desde que o objeto reflita as ondas sonoras, é possível usar um sensor ultrassônico independentemente de seu acabamento superficial ou cor. Existem sensores que podem medir distâncias de dezenas de metros com ótima precisão. Devido a todas essas características, os sensores ultrassônicos são amplamente utilizados na indústria e em várias aplicações de robótica e automação.

As principais aplicações são:

- Detecção de objetos e verificação de presença;

- Medição de altura e largura;
- Medição de níveis de enchimento;
- Detecção de obstáculos;
- Posicionamento de sistemas robóticos;
- Correção de rota de robôs e outros mecanismos móveis como carros de controle remoto;

Introdução

O princípio de funcionamento do HC-SR04 consiste na emissão de sinais ultrassônicos pelo sensor e na leitura do sinal de retorno (reflexo/eco) desse mesmo sinal. A distância entre o sensor e o objeto que refletiu o sinal é calculada com base no tempo entre o envio e leitura de retorno.

” Sinais Ultrassônicos são ondas mecânicas com frequência acima de 40 KHz“

Como ouvido humano só consegue identificar ondas mecânicas até a frequência de 20KHz, os sinais emitidos pelo sensor Ultrassônico não podem ser escutados por nós.

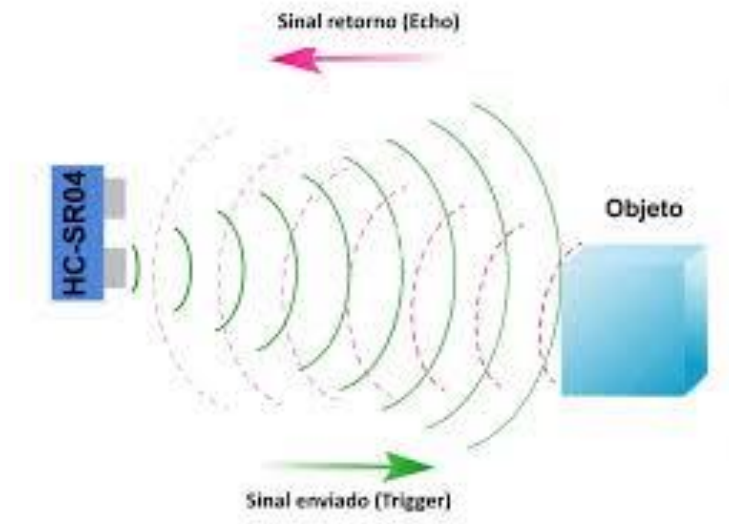
O sensor HC-SR04 é composto de três partes principais:

- **Transmissor Ultrassônico** – Emite as ondas ultrassônicas que serão refletidas pelos obstáculos;
- **Um receptor** – Identifica o eco do sinal emitido pelo transmissor;
- **Circuito de controle** – Controla o conjunto transmissor/receptor, calcula o tempo entre a emissão e recepção do sinal;

Funcionamento

O funcionamento consiste em três etapas:

1. O circuito externo (Arduino) envia um **pulso de trigger de pelo menos 10us em nível alto**;
2. Ao receber o sinal de trigger, o sensor envia **8 pulsos de 40khz** e detecta se há algum sinal de retorno ou não;
3. Se algum sinal de retorno for identificado pelo receptor, o **sensor gera um sinal de nível alto no pino de saída cujo tempo de duração é igual ao tempo calculado entre o envio e o retorno do sinal ultrassônico**;



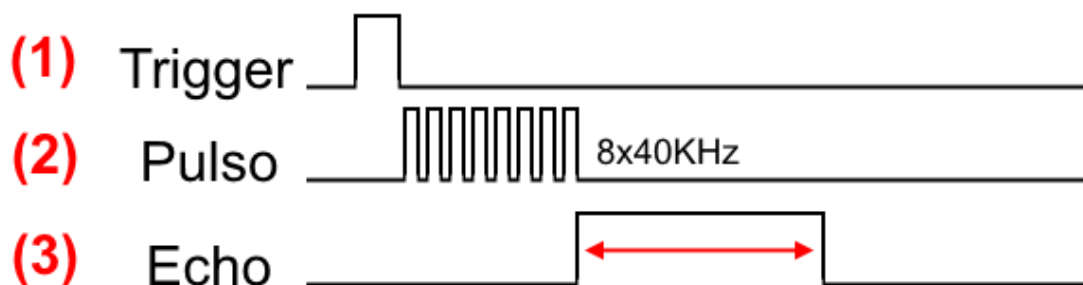
Fonte: [Flip e Flop](#)

Por meio desse pulso de saída cujo tempo é igual a duração entre emissão e recepção, nós calculamos a distância entre o sensor e o objeto/obstáculo, por meio da seguinte equação:

$$\text{Distância} = (\text{Tempo de duração do sinal de saída} \times \text{velocidade do som}) / 2$$

Em que:

- **Tempo de duração do sinal de saída** é o tempo no qual o sinal de output permanece em nível alto;
- **Velocidade do som** = 340 m/s;



Fonte: [Flip e Flop](#)

Repare que as unidades devem estar coerentes para o resultado da conta ser correto. Assim, se a velocidade do som é dada em metros/segundo, o tempo de duração do sinal de saída deve estar em segundos para que possamos encontrar a distância em metros.

Pinagem e características elétricas



Sensor Ultrassônico com indicação de pinagem. Créditos: [Eletrogate](#)

Na figura, podemos ver os **quatro pinos do sensor HC-SR04**. Temos um pino de **VCC**, alimentado com 5V, um **GND**, e os dois pinos de controle e leitura do sensor: O **Trigger**, no qual nós aplicamos o sinal para comandar o envio dos pulsos ultrassônicos, e o **Echo**, que retorna para o Arduino os pulsos com o tempo de duração entre o envio e recepção do sinal de retorno. A corrente elétrica de operação do sensor é de 15mA, portanto é uma aplicação de baixo consumo energético.

Referências

- Balbinot, A.; Brusamarello, V.J. Instrumentação e Fundamentos de Medidas 2a. ed. LTC, 2011
- [Tutorial Aplicações, Funcionamento e Utilização de Sensores](#) Site Maxwell Bohr
- [Como conectar o Sensor Ultrasonico HC-SR04 ao Arduino](#) – Site Flip e Flop
- [Sensor Ultrasonico HC-SR04 com Arduino](#) – Blog Eletrogate
- [Sensores Arquivado em](#) 16 de agosto de 2016, no [Wayback Machine](#). Site FEUP
- [Object Distance Measurement by Stereo VISION](#) Site CiteSeerX