PROFESSOR DANILO *PROJETOS DE CIÊNCIAS – 13/04/2022*

SEU NOME: _____

AULA 11

ATENÇÃO: vamos utilizar esta folha por algumas semanas, portanto tragam esta folha nas próximas aulas. Por isso há um espaço para colocar o seu nome.

O SENSOR ULTRASÔNICO

Veja na figura abaixo a imagem do sensor:

Figura 1: sensor HC-SR04 que utilizaremos na aula de hoje.

Note a existência de quatro pinos:

- VCC: você deve alimentar o dispositivo usando 5 volts do Arduino;
- Trig: este seria o gatilho, isto é, neste fio você deve enviar um pulso para o sensor para que ele comece a enviar os pulsos ultrasônicos. Vamos conectá-lo à uma porta digital do Arduino;
- Echo: aqui o sensor envia para o Arduino um pulso informando o tempo que demora para o som ir e voltar. Também conectamos à uma porta digital do Arduino;
- GND: fio a ser conectado no GND do Arduino.

Agora você já tem uma ideia de como serão feitas as ligações. Vamos então ao circuito.

O professor escolheu a porta digital 2 para ligar no trigger e a digital 3 para ligar no echo.

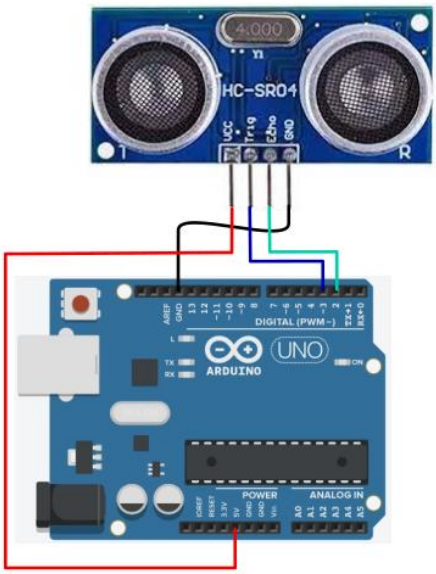


Figura 2: ligação do sensor no Arduino

Sempre que você for trabalhar com um novo módulo ou com qualquer componente eletrônico, você pode começar buscando por "[**nome do módulo**] datasheet". Por exemplo, procure na internet por "HC-SR04 datasheet".

Vamos discutir o funcionamento deste módulo colocando neste documento alguns prints do datasheet que encontrei.

DATASHEET

Primeira parte do datasheet.

Ultrasonic Ranging Module HC - SR04

Product features:

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:
(1) Using IO trigger for at least 10us high level signal,
(2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
(3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.
Test distance = (high level time×velocity of sound (340M/S) / 2,

Figura 3: primeira metarde da primeira página do datasheet. Vemos uma descrição sobre como o sensor funciona.

O datasheet encontrado possui três páginas e, pasmem, alguns componentes, como microprocessadores, podem conter centenas de páginas.

A primeira parte do datasheet pode ser vista na Figura 3: primeira metarde da primeira página do datasheet. no qual podemos ver que o sensor é capaz de medir distâncias entre 2 cm e 400 cm com uma precisão de 3 mm sendo que o módulo possui um transmissor e um receptor de ultrassom, além de um circuito controlador. Na sequência, é descrito o princípio de funcionamento:

1. Usando o gatilho (trigger) através de uma porta IO (Input Output – entrada e saída, ou seja, podemos usar uma porta do Arduino) você deve enviar um sinal de nível alto por 10 microssegundos. No Arduino, significa que você pode usar um digitalWrite(2, HIGH), da sequência um delayMicroseconds(10) e um digitalWrite(2, LOW). Perceba que, numa tradução simples, você precisa ligar o trigger em 5 V por 10 microssegundos.
2. O módulo envia, automaticamente, oito pulsos de 40 kHz (onda ultrasônica) e detecta se houve algum pulso que voltou.
3. SE o sinal voltar, o módulo então gera um pulso na porta IO echo com duração igual ao tempo que o ultrassom leva para ir e volta ao módulo. Ou seja, digamos que o ultrassom demore 500 microssegundos para bater em um obstáculo e voltar, então o módulo liga 5 volts na porta echo por 500 microssegundos.

Depois, ele passa uma fórmula que você viu em cinemática:

$$v = \frac{\Delta S}{\Delta t}.$$

Note que é considerado aqui que o som possui velocidade de 340 m/s. Mas nosso objetivo é exatamente medir esta velocidade, então vamos ignorar esta linha por enquanto.

Veja agora a Figura 4: segunda parte da primeira página do datasheet., na página seguinte, onde vemos como devem ser feitas as conecções:

- A alimentação é com 5 V.
- O trigger recebe um sinal (entrada).
- O echo é o pino de saída, portanto, ele quem responde ao Arduino.
- O GND recebe 0 V.

O esquema de ligações pode ser visto na Figura 2: ligação do sensor no Arduino

Na sequência, vemos uma tabela, traduzida na Tabela 1: Parâmetros elétricos.

Wire connecting direct as following:

- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground

Electric Parameter

Working Voltage	DC 5 V
Working Current	15mA
Working Frequency	40Hz
Max Range	4m
Min Range	2cm
MeasuringAngle	15 degree
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in proportion
Dimension	45*20*15mm

Figura 4: segunda parte da primeira página do datasheet. Detalhes sobre ligações elétricas e tabela com parâmetros elétricos.

Tabela 1: Parâmetros elétricos

Voltagem de trabalho	5 V de corrente contínua
Corrente de trabalho	15 mA
Freequência de trabalho	40 kHz*
Alcance máximo	4 m
Alcance mínimo	2 cm
Ângulo de medida	15°
Sinal de entrada para acionar o gatilho	10 microssegundos de pulso TTL**
Sinal se saída na porta echo	Entrada (na verdade, saída do módulo e entrada no Arduino) de nível alto num pulso TTL com duração proporcional ao alcance.
Dimensões	$45 \times 20 \times 15$ mm

* veja que na tabela original temos um erro, uma vez que o sensor trabalha com ultrassom. A frequência de ondas mecânicas que podemos ouvir, e que por isso chamamos de som, varia de 20 Hz até 20 kHZ, portanto, 40 Hz seria som, e não ultrassom.
** do inglês, *Transistor-Transistor Logic* ou lógica transistor-transistor, cujo título faz referência sobre como o cirtuito que gera os pulsos que queremos funciona. Não é algo que entraremos em detalhes, pois faz parte do que chamamos de circuitos lógicos e tem tudo a ver com a tabela verdade.

Alguns detalhes não serão releveantes para nós nesta aula.

Na página dois do datasheet, temos o detalhe do sensor, já exibido na Figura 1: sensor HC-SR04 que utilizaremos na aula de hoje..

Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.

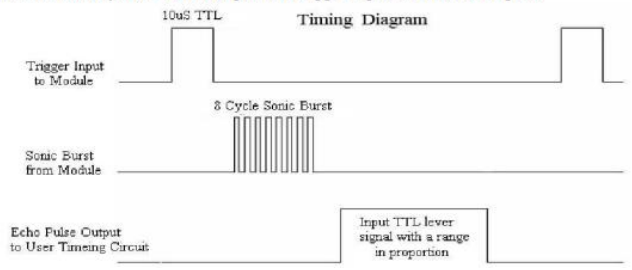


Figura 5:diagrama de tempo dos pulsos.

A outra parte da página dois do datasheet é exibido na Figura 5:diagrama de tempo dos pulsos. onde podemos ver o diagrama de tempo dos sinais envolvidos na medição: o primeiro diagrama representa o pulso que geramos com o Arduino (pulso de 10 microssegundos); o segundo diagrama representa a sequência de oitos pulsos ultrasônico que o sensor irá emitir; por fim, o terceiro diagrama representa a resposta do sensor, dada pela porta echo, que corresponde à um pulso TTL de duração igual ao tempo de ida e volta do ultrassom.

Attention:

- The module is not suggested to connect directly to electric, if connected electric, the GND terminal should be connected the module first, otherwise, it will affect the normal work of the module.
- When tested objects, the range of area is not less than 0.5 square meters and the plane requests as smooth as possible, otherwise ,it will affect the results of measuring.

www.Elecfreaks.com

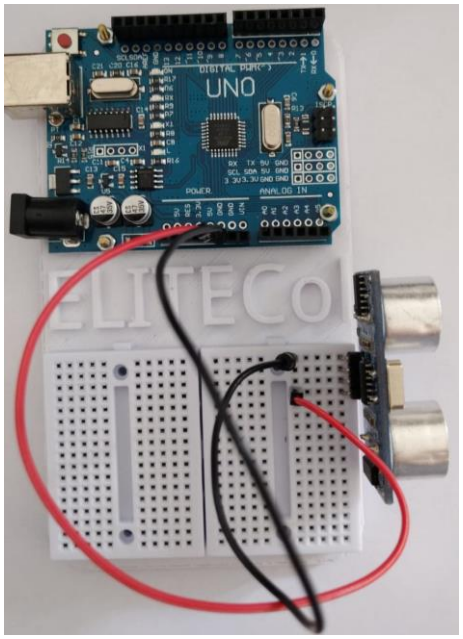
Figura 6: Comentários finais sobre cuidados com o módulo. Note que a área que vai ser usada para refletir os pulsos deve ser plana e o mais suave possível, podendo afetar a o resultado da medida.

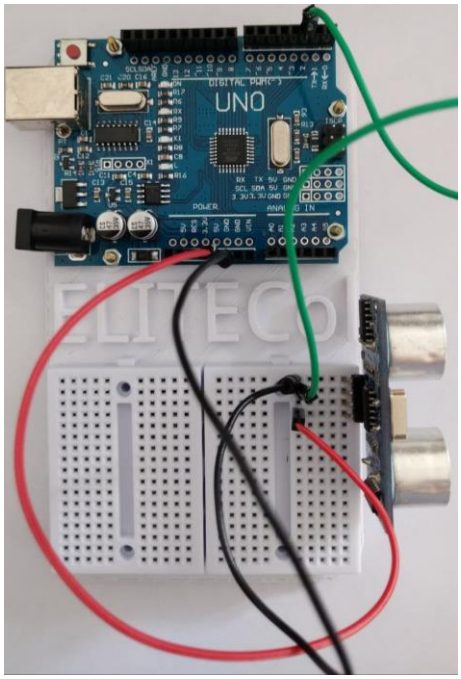
Figura 7: Acesse o datasheet do HC-SR04 no site onde o professor retirou os dados deste material.

O CIRCUITO

Na aula de hoje, você será desafiado, pois o código a seguir não estará finalizado. Você vai calcular a velocidade do som e somente depois disso vai calibrar o sensor para medir distâncias corretamente.

Antes de irmos para o código, você pode consultar a sequência de imagens a seguir para fazer a montagem corretamente no kit que você possui.





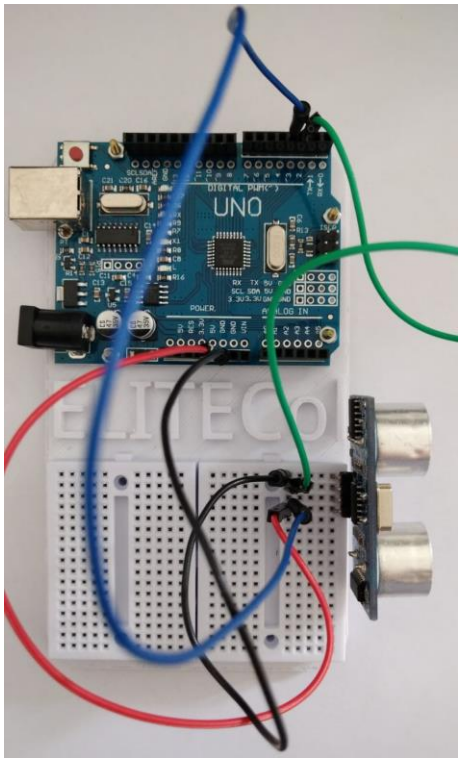


Figura 8: sequência de imagens mostrando o circuito. Se você baixar este arquivo no site do professor você poderá ver estas imagens coloridas.

SKETCH

Para ficar mais fácil a visualização, seu professor colocou o sketch na página a seguir.

Note que temos duas novas funções:

```
delayMicroseconds([tempo de espera em microssegundos]);
```

e

```
pulseIn([porta digital],[ler HIGH ou LOW]);
```

//INÍCIO DO CÓDIGO:

```
int echo = 2, trigger = 3;

long tempo;

void setup() {
  pinMode(echo, INPUT);
  pinMode(trigger, 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();
  delay(1000);//espera um segundo
}
```

//FIM DO CÓDIGO:

ENTENDENDO O CÓDIGO

Nas duas primeiras linhas de códigos declaramos as veriáveis inteiras para as portas digitais e uma variável tipo long para armazenar números maiores. Lembre-se que a memória do Arduino é bem pequena, por isso é necessário gerenciarmos os tipos de variáveis que vamos utilizar.

```
int echo = 2, trigger = 3;

long tempo;
```

Nas linhas que se seguem temos o void loop(), sem novidades para nós: aqui apenas dizemos ao Arduino que o sinal de echo é conectada numa porta de entrada para o Arduino, pois ele recebe tal sinal, e que a porta ligada ao trigger do Arduino é uma porta de saída, pois é por ela que dizemos ao módulo que devemos iniciar as medidas.

```
void setup() {
  pinMode(echo, INPUT);
  pinMode(trigger, OUTPUT);
  Serial.begin(9600);
}
```

Já no void loop() temos algumas novidades, por isso vamos descrever de forma mais detalhada, iniciando com um sinal baixo no pino trigger do sensor apenas para garantir que nenhum erro ocorra e que, sem quer, o módulo faça uma medida em momento inoportuno.

```
void loop() {
  //enviando o pulso TTL
  //vamos forçar o desligamento do trigger
    digitalWrite(trigger, LOW);
```

Vamos agora utilizar uma função nova: enquanto o delay() fazia o Arduino esperar por um tempo, informado nesta função, em milissegundos, a função delayMicroseconds(), como o próprio nome sugere, faz o Arduino esperar por um tempo em microssegundos.

```
  //aqui usamos uma nova função para esperar
microssegundos
  delayMicroseconds(2);
```

Finalmente, vamos enviar o pulso TTL conforme indica o datasheet: ligamos a porta digital do Arduino, que conectamos o trigger do sensor, e aguardamos apenas 10 microssegundos para então desligar esta porta digital.

```
  //inicialndo o pulso TTL
  digitalWrite(trigger, HIGH);
  //aguardando 10 milissegundos
  delayMicroseconds(10);
  //finalizando o pulso TTL
  digitalWrite(trigger, LOW);
```

Conforme dito no datasheet, o módulo envia um sinal com nível lógico alto (pode chamar de Verdadeiro ou HIGH) e o Arduino deverá medir a duração deste pulso. Para isso, usamos uma função chamada pulseIn([porta digital],[ler HIGH ou LOW]) na qual devemos informal qual porta será lida e se leremos o tempo que o sinal fica alto ou baixo (respectivamente, informando HIGH ou LOW). Este valor deve ser armazenado em uma variável.

```
  //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
```

No restante do código, não temos mais novidades: tudo isso você já viu como funciona.

```
  //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();
  delay(1000);//espera um segundo
}
```

Na Figura 9: resultado esperado visto no serial monitor você tem um exemplo do que deverá ver no serial monitor.

```
13:38:50.176 -> O tempo de medida do eco é de 843 microsegundos
13:38:50.176 ->
13:38:51.170 -> O tempo de medida do eco é de 843 microsegundos
13:38:51.170 ->
13:38:52.165 -> O tempo de medida do eco é de 844 microsegundos
13:38:52.165 ->
13:38:53.153 -> O tempo de medida do eco é de 843 microsegundos
13:38:53.201 ->
13:38:54.141 -> O tempo de medida do eco é de 842 microsegundos
13:38:54.189 ->
13:38:55.180 -> O tempo de medida do eco é de 850 microsegundos
13:38:55.180 ->
13:38:56.171 -> O tempo de medida do eco é de 844 microsegundos
13:38:56.171 ->
13:38:57.161 -> O tempo de medida do eco é de 843 microsegundos
13:38:57.209 ->
13:38:58.155 -> O tempo de medida do eco é de 844 microsegundos
13:38:58.202 ->
```

Figura 9: resultado esperado visto no serial monitor

O DESAFIO

Agora vem a parte na qual podemos reacionar com a física: você poderá medir a distância que o ultrassom irá percorrer na ida e na volta, salva na variável tempo, e a distância, que você irá medir em cm (tem uma "régua" no verso desta página).

Porém, você deve prestar muita atenção nas unidades de medidas, pois o tempo é medido em microssegundos.

Inicialmente faça as contas em um papel (use esta folha como rascunho). Posteriormente, devemos tomar cuidado com os cálculos quando o resultado envolver vírgulas. Neste caso, você terá que usar uma variável do tipo float:

```
float tempoEmSegundos;
```

Pode ser interessante declarar várias variáveis tipo float, como o tempo, em segundos, ou a distância em centímetros ou metros.

Essa parte irá ficar mais clara quando nos aprofundarmos um pouco mais na linguagem C++ (dedicaremos três aulas para isso).

Para realizar este experimento, faça cinco medidas de distâncias diferentes e anote os valores que o Arduino irá lhe fornecer, anotando na tabela abaixo. Converta os dados para o sistema internacional, pois você deve estar mais familiarizado a este sistema de unidades. Depois, calcule a velocidade em metros por segundo.

Por fim, calcule a média dos valores obtidos para a velocidade do som. Note que a temperatura, umidade relativa e outros fatores influenciam para a velocidade do som.

Para organizar seus dados, faça isso preenchendo a tabela a seguir.

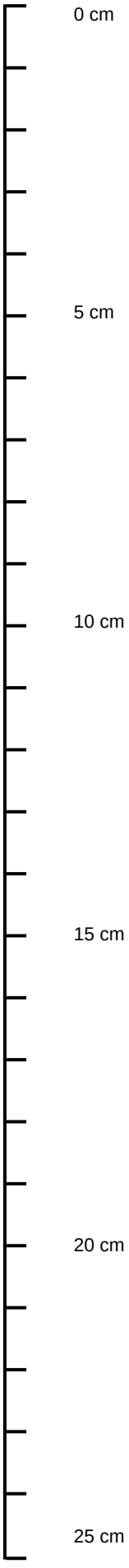
Tabela 2: Dados experimentais de _____

	Tempo (microsse-gundos)	**Tempo (s)**	**Distância (m)**	**Velocidade do som (m/s)**
1				
2				
3				
4				
5				
Média dos valores obtidos para a velocidade do som:				

Normalmente consideramos a velocidade do som como sendo de 340 m/s. Considerando apenas a temperatura e a Tabela 3: Influência da temperatura do ar na velocidade do som, o valor que você obteve é razoável ou você considera que sua medida não foi muito boa? Se a medida não foi satisfatória, saberia elencar alguns motivos para que isso tenha ocorrido?

Tabela 3: Influência da temperatura do ar na velocidade do som

Temperatura ambiente (°C)	Velocidade do som (m/s)
−30 °C	312,7
−25 °C	315,9
−20 °C	319,1
−15 °C	322,2
−10 °C	325,3
−5 °C	328,4
0 °C	331,5
5 °C	334,5
10 °C	337,5
15 °C	340,5
20 °C	343,4
25 °C	346,3
30 °C	349,2


0 cm
5 cm
10 cm
15 cm
20 cm
25 cm