Relatório Mini Projeto com Arduino

Sistemas Reativos - INF1805

Nome: Willian Simões Pinto

Este documento descreve alguns dos principais pontos da execução do mini projeto.

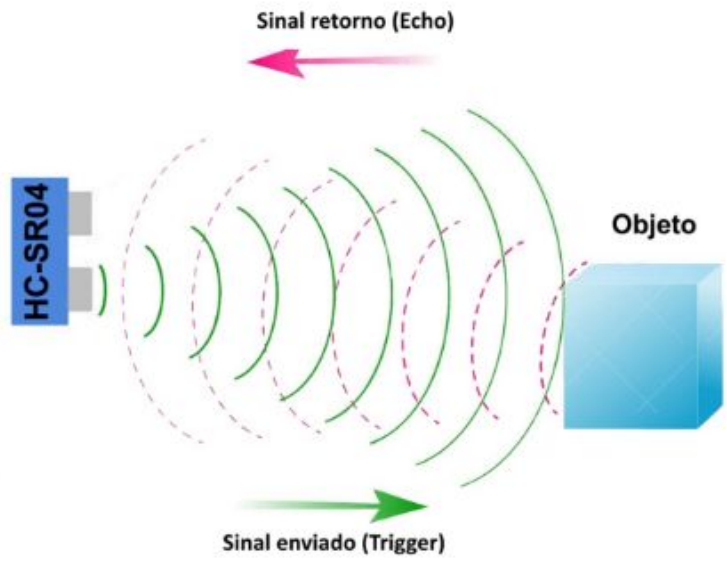
Componentes usados:

- Arduino Mega 2560
- Sensor de Distância Ultrassônico HC-SR04
- Buzzer Ativo 5V
- Resistor de 330 Ω

Ideia

Um dispositivo para cegos que através de um sensor de distância, um buzzer e um arduíno indicará a proximidade com eventuais obstáculos através de sons.

Funcionamento do sensor de distância



Fonte da imagem: Blog Flip Flop[1]

O sensor de distância ultrassônico possui um transmissor e um receptor. O pino de Trigger em nível HIGH envia uma onda sonora de altíssima frequência (40kHz - inaudível para o ser humano) e, ao encontrar algum objeto, a onda é refletida e volta, fazendo com que o pino Echo fique em nível HIGH. Tendo em mente que o som se propaga a 340 m/s, é possível calcular a distância que o sensor está do objeto. Basta saber quanto tempo o pino Echo permaneceu

HIGH logo após o pino Trigger ter sido colocado em nível HIGH e fazer as contas de acordo com as unidades desejadas.

Cálculo da distância

Vamos supor que se queira calcular a distância em cm do objeto, tendo em mãos o tempo em microssegundo que a onda chegou HIGH no pino Echo (tempo total da propagação da onda).

Se a velocidade do som é 340 m/s, temos: 34000 cm/s
Se 1 µs = 10^{-6} s, temos que 1s = 10^{6} µs.
34000 cm ___ 1s ___ 10^{6} µs
1 cm ___ x µs

x = 29,4 µs
Em 1cm a onda demora 29,4 µs para se propagar.

Usando a clássica forma para calcular velocidade, temos que:
A distância que a onda se propagou desde sua saída do pino Trigger até o pino Echo

distancia_total_cm = velocidade x tempo_total
distancia_objeto_cm = [(1cm/29,4µs) x tempo_total] / 2

Essa distância é dividida por 2 já que queremos apenas a distância do objeto até o sensor, e o tempo calculado é o tempo de ida e volta da onda.

Então o cálculo de distância fica:
distancia_objeto_cm = tempo_total / 29,4 / 2

Implementando com a função pulseIn

Com o cálculo em mente, a função pulseIn pode ser usada para que o tempo total de propagação seja calculado. Ela mede o tempo desde o pino Echo entrar em HIGH até ir para LOW novamente. Espera-se pelo menos 10 microsegundos para iniciar a medição, que é o tempo da primeira onda propagada pelo sensor.

```
digitalWrite(TRIG_PIN, HIGH);
delayMicroseconds(10);
digitalWrite(TRIG_PIN, LOW);

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

Funcionamento

Foi implementado dois modos de bipagem para avisar ao usuário que um obstáculo está próximo: um que aumenta a frequência conforme diminui a distância com o objeto e outro que bipa mais rápido quanto mais próximo do objeto fica. Para variar a frequência conforme o objeto chega mais próximo do sensor, foi usada a função map da biblioteca padrão do arduino que mapeia uma faixa de valor para outra (no caso, o intervalo das distâncias para o intervalo das frequências desejadas). Dois botões do Arduino Mega são usados para alternar entre os modos.

Um led rgb também é usado, mas apenas como acessório (não faz parte da ideia inicial do projeto). A luz fica verde quando a distância do objeto é de 30 a 20cm, amarela quando está de 20 a 10cm e vermelha quando está a menos de 10cm. Quando está a 5cm ou menos, ambos os modos bipam acelerados.

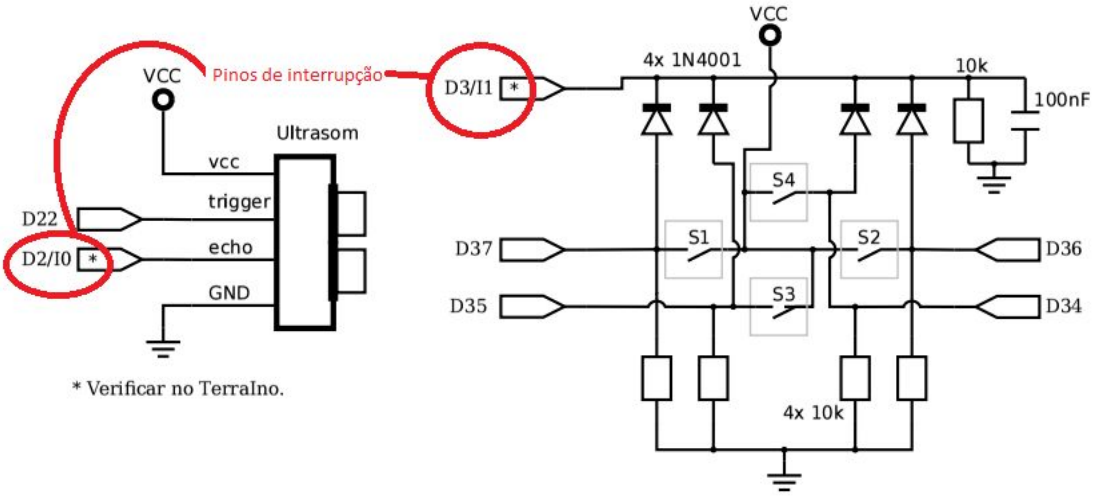
Problemas com a função pulseIn

O uso da função pulseIn apesar de estar correto traz desvantagens para a programação em sistemas reativos. Ela é uma função bloqueante, ou seja, enquanto ela é executada o processador fica ocupado. No caso do projeto por exemplo, ela ficará esperando um sinal HIGH no pino indicado para poder começar a medir o pulso. Por mais que se defina um timeout (por default é 1s), o sinal pode demorar a chegar ou ter grande duração, impedindo que o sistema reaja a outras mudanças/interações por ser bloqueante.

Usando interrupções

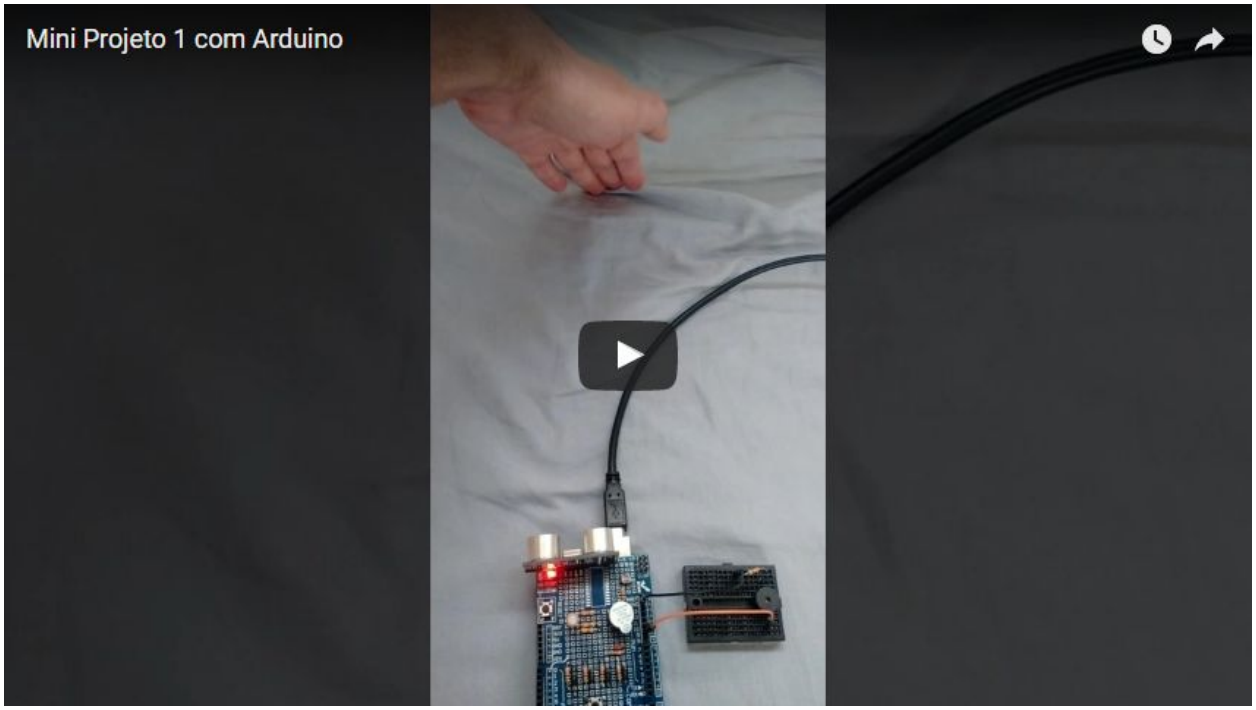
Para tratar melhor então esse problema, foi utilizado o uso de interrupções em dois momentos: ao pressionar o botão UP ou DOWN ou quando é enviado um sinal ao pino ECHO. Assim, foi possível mudar o modo de bipagem e medir o tempo do pulso sem interromper desperdiçar processamento. As rotinas intSensor e intBotao definidas no próprio código são chamadas ao ocorrer uma interrupção correspondente a elas.

```
// interrupção do sensor de distância - é acionada quando o sinal do pino Echo muda (CHANGE)
attachInterrupt(0, intSensor, CHANGE);
//interrupção do botão - é acionada quando o botão passa pro estado HIGH (RISING)
attachInterrupt(1, intBotao, RISING);
```



Vídeo de demonstração

https://www.youtube.com/watch?v=5GI45HpKztM



Clique no link ou na imagem

Código fonte

GitHub:
https://github.com/willsimoes/INF1805/blob/master/mini-arduino/mini_projeto/mini_projeto.ino

Referências

https://www.arduino.cc/en/Reference/pulseIn
https://www.arduino.cc/en/Reference/attachInterrupt
https://www.arduino.cc/en/reference/map
https://www.arduino.cc/en/reference/tone
http://blog.filipeflop.com/sensores/sensor-ultrassonico-hc-sr04-ao-arduino.html
http://blog.fazedores.com/sensor-ultrassonico-com-arduino/
https://programmingelectronics.com/an-easy-way-to-make-noise-with-arduino-using-tone/
http://blog.vidadesilicio.com.br/arduino/sensor-ultrassom-hc-sr04/
http://www.bosontreinamentos.com.br/eletronica/arduino/funcao-map/
http://blog.vidadesilicio.com.br/arduino/usando-a-interrupcao-externa-no-seu-arduino/