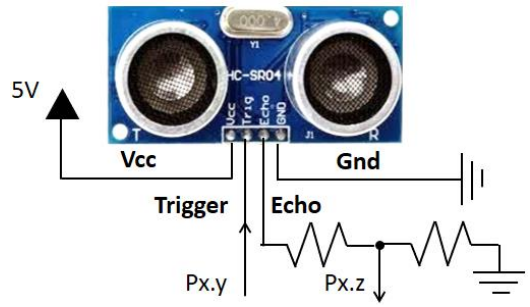
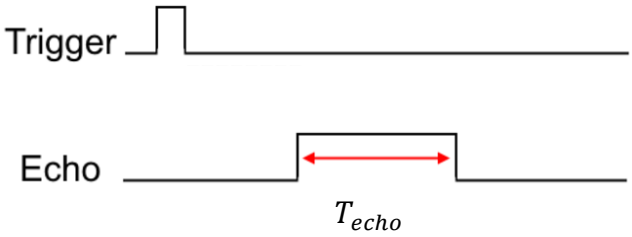
Módulo 2: I/O, Temporizadores e Interrupções

Utilizando o sensor de Proximidade

O sensor de proximidade HC-SR04 é equipado de um autofalante e um microfone ultrassônicos. Quando acionado por um pulso positivo na entrada *trigger* o sensor envia pulsos sonoros ultrassônicos inaudíveis ao ouvido humano. Uma resposta do tempo de propagação é devolvida no sinal *echo*. O sinal *trigger* deve ter duração mínima de 10 µs e o sinal echo irá retornar um pulso com duração máxima de 12 ms (qual é a maior distância que podemos medir com esse dispositivo?).

Atenção, o sensor é alimentado com 5V. Use então um **divisor resistivo no retorno do echo** para não queimar o pino de entrada da sua launchpad. Sugestão: 4K7 e 10K. Pergunta: qual desses dois resistores deve estar ligado ao terra.





PROBLEMA 2: Theremin (R2D2) (Visto 2)

A solução deste problema é individual, de forma a que o aluno comprove seu domínio sobre o assunto. Após resolvê-lo, o aluno deve fazer um pequeno vídeo (máximo de 2 minutos) de forma a comprovar sua solução. Neste vídeo, além de gravar os sons gerados, use uma régua para comprovar o correto comportamento dos leds.

Cada aluno deverá enviar sua listagem e seu vídeo. Serão aceitos apenas dois arquivos por aluno. A listagem de sua solução deve ser auto contida, ou seja, ela simplesmente será copiada para o CCS para comprovar seu correto funcionamento. Será verificado o correto funcionamento de todo programa. **Caso o programa solução entregue não funcione, a nota será zerada**.

Importante: Serão aceitas apenas uma listagem e apenas um arquivo de vídeo por aluno.

Regra para o nome dos arquivos:
Erros na nomeação serão penalizados em 20%

Siga o formato: **TP2-xxxxxxxx.ext**

T = sua turma (A, B, C, ...)

P2 = Problema 2

xxxxx = seu número de matrícula, usando apenas números

ext = extensão que indica o tipo do arquivo (asm, c, avi, ...)

Na primeira linha de seu código deverão estar sua matrícula e seu nome completo.

ÉTICA E HONESTIDADE ESTUDANTIL:

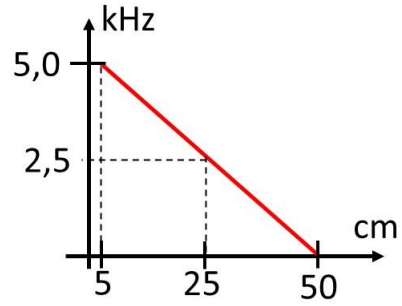
Será verificada a similaridade entre os programas entregues e os demais programas, incluindo os dos semestres anteriores. Caso a similaridade entre dois programas seja grande o suficiente para caracterizar a "cola", os alunos estão **automaticamente reprovadas**.

Problema 2 (programa para receber visto, valendo 20 pontos)

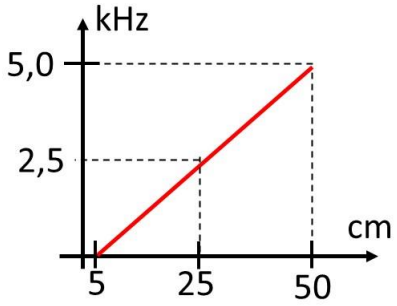
Escreva uma aplicação embarcada que use o sensor HC-SR04 para medir a distância até um obstáculo. A sinalização da distância será indicada de duas formas. Nos dois leds da LaunchPad e por tom gerado por um buzzer passivo (ou mini amplificador com um alto falante) conforme esquematiza na última página. As figuras abaixo ilustram as duas sinalizações esperadas. Você pode interpretar este problema como um sensor de estacionamento ou como um Theremin muito simples.
(https://www.youtube.com/watch?v=K6KbEnGnymk).

Faixa de distância (cm)	Led 1 (P1.0) Vermelho	Led 2 (P4.7) Verde
Acima de 50	0	0
de 30 até 50	0	1
de 10 até 30	1	0
abaixo de 10	1	1

Sinalização pelos leds



Ambas as chaves (S1 e S2) abertas



Uma ou duas chaves (S1 ou S2) acionadas

Sinalização sonora, note que os sons são gerados somente quando a distância indicada pelo sensor está na faixa de 5 a 50 cm.

A medição da distância deve acontecer na taxa de 10 ou de 20 medições por segundo. A sugestão é usar um PWM para tornar o disparo automático. A largura do pulso de eco deve ser medida com o recurso de captura do timer. A cada medição os leds e o tom devem ser atualizados.

É obrigatório o uso dos seguintes recursos, de acordo com a figura abaixo:

- TA0.4 → disparar a medida (Trigger)
- TA1.1 → medir a duração do pulso de eco e
- TA2.2 → acionar o buzzer

Este problema vale 20 pontos, com as seguintes penalidades:

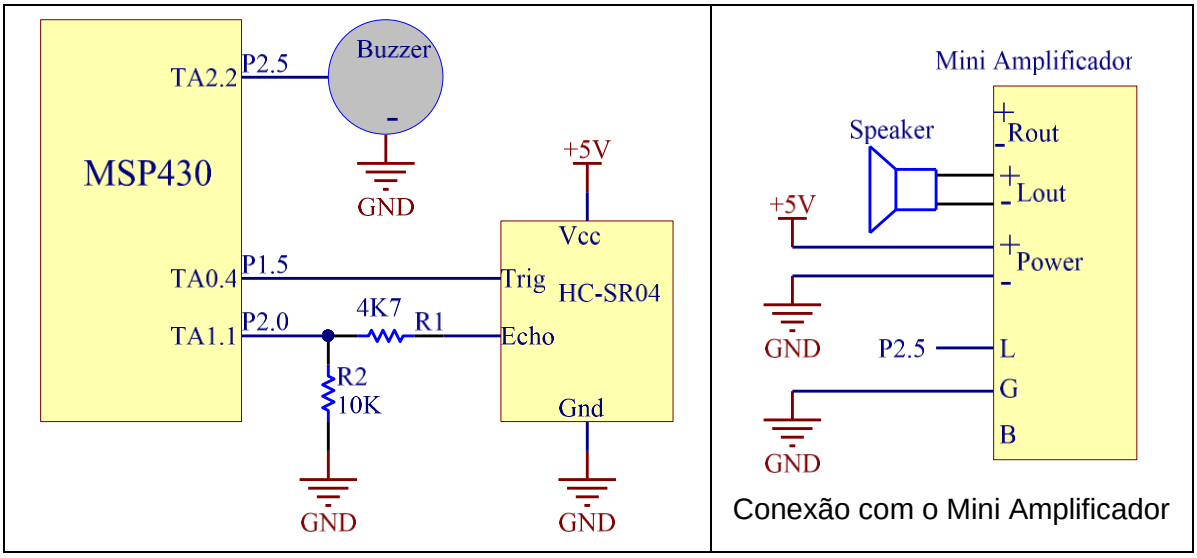
- 5 pontos: por não usar o recurso de captura do timer;
- 5 pontos: por não usar a interrupção do timer.

Dicas para elaboração de seu programa:

1) Prefira montar o sensor na vertical, um pouco afastado da superfície de sua mesa, pois ela pode refletir o sinal. Use um obstáculo grande que possa parar na vertical como um livro ou uma caixa. Em suma, tudo deve estar estável, do contrário você não terá certeza nas medições.
2) Elabore o programa por etapas. Teste cada uma delas em separado para ter certeza de seu correto funcionamento.
3) Inicie com a verificação do disparo do sensor e medição do sinal de eco. Prepara uma configuração com um obstáculo a uma distância conhecida e verifique ser a medição está coerente. Varie a distância e confira. O recurso de breakpoint pode ajudar muito nesta etapa.
4) Depois, vá pouco a pouco adicionando as demais funções. Veja, então, que se deve quebrar o programa em pequenas funções.
5) Saiba que tentar escrever todo o programa de uma só vez é certeza de insucesso.
6) Sugestão para as funções, considerando que a duração do eco já está disponível:
    a. dist = calc_dist(eco); Calcular a distância em cm
    b. leds(dist); Acender os leds em função da distância
    c. freq = calc_freq(dist); Calcular a frequência a ser gerada
    d. ta2_prog(freq.); Programar TA2 de acordo com a frequência

Se você já conseguiu fazer todo o problema, considere os aperfeiçoamentos abaixo.

7) Após ter tudo funcionando, você pode notar que as medições são ruidosas. É possível usar um filtro para reduzir o ruído. São sugestões o filtro média móvel e o filtro mediana. Pode usar outros. Faça vários ensaios para ver se os resultados melhoram.
8) Algumas vezes surgem valores impossíveis. Por exemplo: uma medição resulta em 30 cm e a próxima medição resulta em 2m (é o teleporte). Pense numa forma de eliminar valores absurdos.
9) A atualização de TA2 acontece a cada medida. Pense numa forma de realizar essa atualização de TA2 sem provocar frequências espúrias.



Conexão com o Mini Amplificador

Para auxiliar no entendimento, será distribuído um arquivo executável (.out) com a solução. Nesta solução, com fins de identificação, os leds têm um comportamento um pouco diferente.