Assistência senso-motora para deficientes visuais

Aline Rosa dos Santos Rocha[1],16/0023076, Felipe Lima Alcântara[2], 16/0027918
[1,2]Programa de Engenharia Eletrônica, Faculdade Gama - Universidade de Brasília, Brasil

***Resumo*—O projeto Assistência senso-motora para deficientes visuais(ASMDV) é um equipamento assistivo, construído para facilitar o deslocamento de deficientes visuais, cujos os quais não têm como detectar obstáculos fora da altura alcançável pela bengala. Utilizando o *MSP430G2* e sensores ultrassônicos, buscou-se criar uma alternativa, para detecção de obstáculos altos.**

***Index Terms*—*MSP430*, tecnologia assistiva, deficiência visual, sensor ultrassônico**

I. Introdução

De acordo com a Fundação Dorina Nowill para Cegos[1], do total da população brasileira, $23,9\%$ (45,6 milhões de pessoas) declararam ter algum tipo de deficiência. Entre as deficiências declaradas, a mais comum foi a visual, atingindo $3,5\%$ da população. Em seguida, ficaram problemas motores ($2,3\%$), intelectuais ($1,4\%$) e auditivos (1,1%). Segundo dados do IBGE de 2010, no Brasil, das mais de 6,5 milhões de pessoas com alguma deficiência visual 528.624 pessoas são incapazes de enxergar (cegos), 6.056.654 pessoas possuem baixa visão ou visão subnormal (grande e permanente dificuldade de enxergar) e outros 29 milhões de pessoas declararam possuir alguma dificuldade permanente de enxergar, ainda que usando óculos ou lentes.Dentro deste aspecto, Hogetop e Santarosa [2] afirmam que "a mediação digital vem impreterivelmente favorecer inúmeras novas oportunidades de acesso, em via dupla, ao conhecimento da cultura por parte do indivíduo [...]. A Educação Especial tem agora novas perspectivas de abordar a diversidade humana e "des"cobrir todos que historicamente foram excluídos, escondidos, discriminados, encobertos pelas diferentes sociedades no *continuum* das épocas". De forma menos abrangente, é isto que o projeto se propõe a fazer.

II. Justificativa

Segundo Souza [3], as soluções atuais mais amplamente utilizadas no Brasil para locomoção de cegos são as bengalas comuns - os usuários a movimentam e se ela esbarrar em algum objeto conseguem ter um noção de onde há obstáculos e assim podem desviar, tem preço acessível mas se limita ao sensoriamento abaixo da linha da cintura, - e cães-guia - que é uma boa solução por ser tratar de um ser inteligente auxiliando a locomoção mas pouco acessível devido aos poucos centros especializados no treinamento dos cães e todo o custo financeiro envolvido nesse processo. Alguns projetos de conclusão de curso e pós-graduação desenvolveram uma bengala eletrônica, Figura 1, ela faz uso de um microcontrolador em sua parte superior e dois sensores - um no meio da bengala responsável pelo sensoriamento de objetos acima da linha da cintura e o outro, no final da bengala responsável pelos objetos abaixo da linha da cintura. Porém, além de ainda não estar sendo produzida, o preço de venda ficou estimado em R$ 300,00 em um dos projetos[4].

Figura 1. Bengala eletrônica.

Uma pesquisa de campo foi feita com um portador de deficiência visual, Leonardo Lemos de 22 anos, morador do Distrito Federal e cego desde os 09 anos. Ele relatou que só faz uso da bengala tradicional e que uma das grandes dificuldades que sente são os obstáculos aéreos como galhas de árvores, placas de trânsito ou qualquer outro objeto que esteja próximo à cabeça.

Tendo isso em vista, pensou-se que uma solução que envolvesse sensores que monitorassem a presença de objetos na parte superior do corpo e avisassem ao usuário através de som ou vibração seria de grande valia, principalmente se houvesse baixo custo associado à produção. A solução usaria microcontroladores do tipo *MSP430*, que pudessem facilitar a locomoção de deficientes visuais severos, pensado-se principalmente nos que entraram em tal condição recentemente - haja visto que, pessoas que à pouco tempo tiveram uma perda drástica de visão, terão muitas dificuldades associadas à locomoção que a bengala comum não é capaz de sanar, como evitar esbarrar em objetos que estão na região mais próxima à parte superior do corpo -.

III. Objetivo

Desenvolver protótipo funcional de um dispositivo capaz de detectar objetos acima da linha da cintura e avisar sensorialmente conforme for se aproximando o obstáculo. O projeto é pensado especialmente para portadores de deficiência visual, afim de evitar colisões. Projeta-se que os obstáculos seja sensoreados por 3 Sensores de Distância Ultrassônico HC-SRO4 localizados na região da cabeça - um na direção frontal e dois laterais -,o microcontrolador seja uma *MSP430G2,* da *Texas Instruments* e a alimentação do circuito feita com bateria recarregável.

Além disso temos que levar em conta as características de:

A. Precisão

Como para o cálculo da distância do sensor de ultrassom, é necessário utilizar um timer em modo de captura, vide Secção VII-B, é importante definir bem as interrupções na codificação da placa [5].

B. Conforto e estética

Em pesquisa de campo, um ponto levantado pelos entrevistados foi a questão estética. O público alvo do projeto mostrou preocupação em usar dispositivos que tivessem muitos fios, fossem grandes ou causassem grande estranheza para quem olhasse.

Assim, o projeto deve ser confortável para que não prejudique os usuários e esteticamente agradável. Nesse sentido, as sinalizações com *buzzers* foram desconsideradas - barulho chamativo - e Motores de Vibração Chato com fio (8x4 mm) foram adotados, já que possuem vibrações baixas mas suficientes para alertar.

IV. Metodologia

Para alcance do desenvolvimento completo de um *assistente senso-motor para deficientes visuais*, entendeu-se que seria necessário iniciar o projeto com pesquisa de aplicações, desenvolvimento do software com o complicador *energia*, aprimoramento de *sofwtare* no *Code Composer Studio* e desenvolvimento do hardware e refinamento em C. Com o entendimento da necessidade destas fases, foi então estabelecido quatro pontos de controle, em que, à cada ponto era estabelecida uma meta de desenvolvimento à nível de hardware e de software. Conforme as metas eram alcançadas, objetivos maiores e mais perto do projeto final eram estabelecidos.

A. Pesquisa de Aplicações

Inicialmente foi realizada uma pesquisa do estado da arte, buscando reconhecer o que já foi realizado na área do tema. Junto a pesquisa, foi realizada uma breve conversa com um deficiente visual, buscando se inteirar da real necessidade do produto. Com o projeto definido e o problema identificado deu-se inicio ao projeto.

B. Desenvolvimento do Software com o Compilador Energia

Nesta etapa do projeto foi iniciado o processo de produção do protótipo, com um compilador de utilização mais simples. Realizando assim os primeiros testes com os sensores e o microcontrolador, buscando identificar possíveis ameaças e necessidades de *hardware* e *software*.

C. Aprimoramento de Software no Code Composer Studio

Esta etapa será abordada na Secção VII-E2, buscando já realizar aprimoramentos no código e implementação na linguagem C, e integração de algumas partes do *hardware*, principalmente a identificação e sinalização dos obstáculos.

D. Desenvolvimento do Protótipo e Refinamento em C

Inicia-se aqui a etapa de desenvolvimento do *hardware*, buscando se adequar ao abordado no Secção III-B. Além do trabalho com hardware, o software deverá ser aprimorado, tendo como objetivo demonstrar mais recursos do sistema e integrar o sensor de cabeça.

E. Conclusão do Hardware e Refinamento em C

Esta etapa busca finalizar a construção do *hardware* do protótipo e buscar já haver a temporização do sistema bem definido. Os sensores já deverão estar com regulação confiável.

V. Requisitos

A. Requisitos materiais e custos

Um ponto importante que diferencia o projeto de outros já desenvolvidos é seu baixo custo. Estima-se o custo de produção com material abaixo.

- 1 Microcontrolador *MSP430G2* - R$ 40,00;
- 3 Sensores ultrassônicos $HC-SR04$ - R$ 8,90 casa = R$ 26,70;
- Boné para suporte do sensor da cabeça e do circuito - R$ 20,00;
- 3 Motores de Vibração Chato com fio (8x4 mm) - R$ 4,90 cada = R$ 14,70;
- Bateria alcalina 9V - R$ 18,00;
- 1 CI SN74LS153 - R$ 2,50;
- *Jumpers* macho-macho e macho-fêmea - R$ 17,80;

Assim, estima-se um custo total de produção de R$ 139,60.

B. Requisitos técnicos

C. Programação em C

O *Software* utilizado faz uso da linguagem C para a implementação, sendo assim um requisito é conhecer e se aprofundar nesta linguagem, buscando refinar o código e evitar possíveis logicas não otimizadas.

1) Habilidades em eletrotécnica: Para a construção da parte estrutural, faz necessário o uso de protoboard, jumpers, PCBs, entre outros componentes. Assim, torna-se necessário habilidade em soldagem e manuseio de destes componentes.

2) Formatação dos documentos: A elaboração e manutenção dos documentos produzidos no projeto deverá utilizar *LaTeX* de forma que a apresentação das informações fique organizada, assim como, representará as instruções para a construção do protótipo.

VI. Benefícios

A proposta de projeto tecnológico assistivo de monitoramento senso-motor para deficientes visuais visa melhorar a vida de pessoas que tiveram perdas bruscas de visão, principalmente perdas recentes que ainda não se acostumaram com suas rotinas e trajetos, trazendo mais conforto e confiabilidade em seus trajetos ao monitorar obstáculos que estejam próximos a regiões do corpo do usuário que a bengala tradicional não é capaz de detectar e assim avisá-lo evitando entrocamentos, quedas e até acidentes mais sérios. Em relação a tecnologias já existentes citadas anteriormente, o benefício desse projeto seria o custo de aquisição mais acessível atendendo a um leque maior de consumidores.

VII. Desenvolvimento

A. O microcontrolador $MSP430G2553$

Segundo a fabricante *Texas Instruments*, a família de microcontroladores de baixíssima potência da *Texas Instruments MSP430* consiste em vários dispositivos diferentes e conjuntos de periféricos destinados a várias aplicações. A arquitetura, combinada com cinco modos de operação de baixa potência, é otimizado para aumentar a vida útil da bateria em aplicações de medição portáteis. O dispositivo apresenta *CPU RISC* de 16 bits, registradores de 16 bits e geradores constantes que contribuem para a eficiência de código.[5] O oscilador controlado digitalmente (DCO) permite o despertar dos modos de baixa potência para o modo ativo em menos de 1 µs. Além disso, os membros da família MSP430G2x53 tem um conversor analógico-digital (A/D) de 10 bits. Aplicações típicas incluem sistemas de sensores de baixo custo que capturam sinais analógicos, convertem-nos em valores digitais, e, em seguida, processam os dados para exibição ou para transmissão para um sistema *host*.

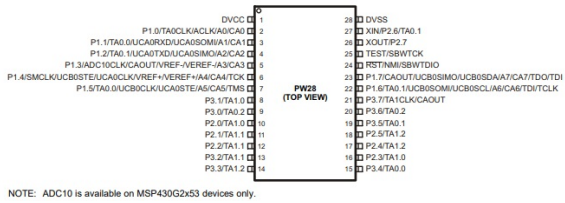

Figura 2. Pinagem da MSP430G2553.

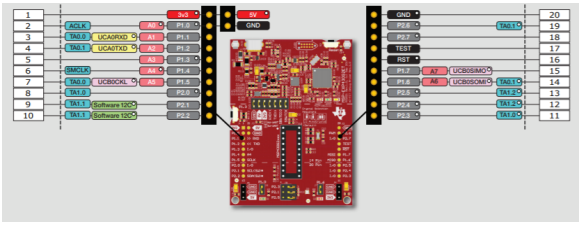

Figura 3. LaunchPad QuickStart Guide (QSG).

B. O Sensor $HC-SR04$

De acordo com o *datasheet*, após a porta *Trig* receber um pulso, a porta *Echo*, depois de ocorrer o envio da onda de ultrassom, iniciará a enviar o valor da tensão em V_cc até o ultrassom receber a onda de eco ou exceder a distância máxima de leitura, segundo o *datasheet* $4m$, o funcionamento do sensor está exemplificado na Figura 4.

O tempo de pulso da porta Echo, será usado para o cálculo da distância. O cálculo é realizado pela equação[6]:

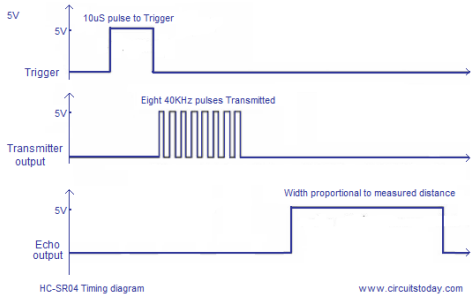

Figura 4. Diagrama de funcionamento no tempo do sensor de ultrasom.

$$Distancia = \frac{T_{Echo} \cdot V_{som}}{2} \quad (1)$$

Algumas vezes é necessário realizar testes para tornar mais preciso o cálculo da distância, para isso escrever um código simples para a leitura serial das distâncias é uma boa prática.

C. Multiplexação do sinal Echo

Uma das grandes dificuldades de todo o projeto estava em conseguir realizar a análise das distâncias por conta do número insuficiente de *Timers* da MSP430G2553. As interrupções de ambos os *timers* afetando uma a outra, fazendo leituras de distância erradas ou ainda não realizando nenhuma leitura no sensor.

Para isso a ideia de multiplexação da saída do sensor surgiu, pois diminuiria o número de portas utilizadas da MSP430, de 6 portas, para 4, sendo antes 3 portas de modo de comparação e com o multiplexador somente uma. A implementação da multiplexação se deu no Ponto de Controle 4, sendo provada vantajosa.[7]

Um multiplexador tem a função de por uma série de entradas, a partir de suas seletoras entregar uma saída igual à uma das entradas, como pode ser analisado em sua tabela verdade, Figura 5. Para a multiplexação foi utilizado o Circuito Integrado *SN74LS153N*.

D. Motores com PWM

Quanto aos motores de vibração, Figura 6, para vibrar, eles receberam ondas moduladas, *PWM*(*Pulse Width Modulation*). A vantagem de tratar o motor desta forma é a possibilidade de modulação da largura do pulso, aumentando e diminuindo a intensidade com aplicação em *Software*[8], além de se levar-se em conta que a potência de um pequeno motor DC depende da tensão aplicada pode-se usar este artifício para variar essa

FUNCTION TABLE

SELECT INPUTS		DATA INPUTS				STROBE	OUTPUT
B	A	C0	C1	C2	C3	$\overline{G}$	Y
X	X	X	X	X	X	H	L
L	L	L	X	X	X	L	L
L	L	H	X	X	X	L	H
L	H	X	L	X	X	L	L
L	H	X	H	X	X	L	H
H	L	X	X	L	X	L	L
H	L	X	X	H	X	L	H
H	H	X	X	X	L	L	L
H	H	X	X	X	H	L	H

Select inputs A and B are common to both sections.
H = high level, L = low level, X = irrelevant

Figura 5. Tabela Verdade do circuito integrado SN74LS153N

Figura 6. Motor de Vibração Chato com Fio .

potência sem, entretanto, modificar a tensão aplicada ao motor.

À parte positiva que pode ter seu pulso modulado é dado o nome de *Duty Cycle*. A Figura 7 trás um gráfico da voltagem pelo tempo exemplificando algumas modulações.

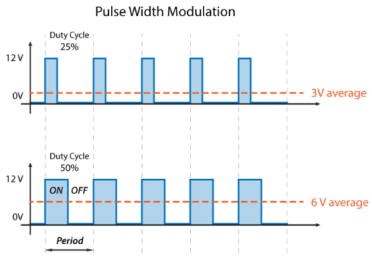

Figura 7. Modulação da largura do pulso.

Um controle *PWM* é realizado ajustando o *Duty Cycle*, aumentando-se a largura deste, a potência do motor aumenta também e se diminuir a largura, o mesmo ocorre com a potência.

Outra vantagem de aplicar o PWM é que o sinal permanece digital em todo o percurso desde o processador até o sistema controlado e nenhuma conversão de digital para analógico é necessária. Ao manter o sinal digital, os efeitos de ruído são minimizados pois um ruído só pode

afetar um sinal digital se ele for forte o suficiente para alterar uma lógica 1 para uma lógica 0 ou vice-versa.[9]

E. Ponto de Controle 2

1) Hardware: Para o ponto de Controle 2, buscou-se realizar a aplicação da identificação de obstáculos determinando a distância deste ao sensor. A intensidade de vibração foi exemplificada pelo uso de *leds*, que foram conectadas em série as saídas do tipo paralelo, do $74HC595$. O design em *protoboard*, realizado no software *Fritzing*, está na Figura 8.

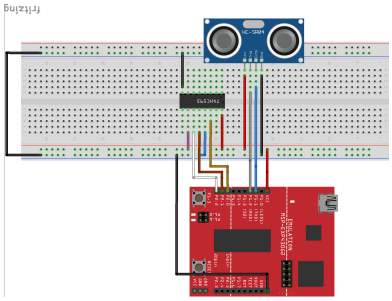

Figura 8. Desing do Circuito em *Protoboard*, sem *leds* conectadas a saída.

2) Software: O código foi realizado no ambiente de desenvolvimento integrado da *Texas Instruments*, *Code Composer Studio*(CCS), o código escrito na liguagem C.

Inicialmente é enviado um pulso para entrada $Trig$, de $20ms$, após isso espera-se $3\mu s$ para um novo pulso no $Trig$. O *Timer A* foi definido em modo de captura para a entrada do pino $Echo$, ao ser encontrada a borda de subida é realizada a interrupção e após a borda de descida é realizado o cálculo da distância.

Com a distância do obstáculo definida, uma variável interna é setada de acordo com o valor da distância, assim enviando pra porta de saída um valor serial pré-setado em uma matriz.

F. Ponto de Controle 3

1) Software : O grande desafio à nível de software do terceiro ponto de controle era conseguir realizar a leitura de dois sensores simultaneamente usando apenas uma *MSP430G2*. Inicialmente pensou-se em utiliza-los de maneira *"paralela"*, enviando o pulso de Trigger, com o único atraso sendo de leitura de tarefa.

```
P1OUT |= TRIG1;
P1OUT |= TRIG2;
__delay_cycles(20);
P1OUT &= ~TRIG1;
P1OUT &= ~TRIG2;
```

Para usá-los, na codificação os sensores foram colocados em portas diferentes, a Figura 9 exemplifica o funcionamento, se havendo a borda de subida em um dos pinos *Echo*, a função de interrupção da porta é chamada, inicializando assim a contagem do pino *Echo* em nível lógico alto.

```
P1IE |= EchoPin1;
P2IE |= EchoPin2;
P1IES &= ~EchoPin1;
P2IES &= ~EchoPin2;
```

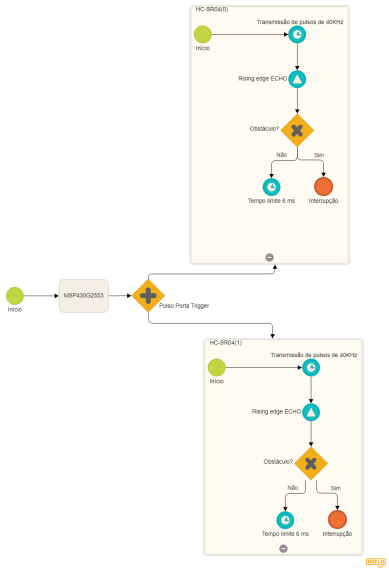

Figura 9. Fluxograma de Funcionamento do Software para os sensores de ultrassom, funcionamento paralelo.

Na interrupção caso seja verificada uma borda de descida significa que um obstáculo foi identificado, assim podendo realizar o calculo da distância. Para o calculo foi usado o numero de vezes que a interrupção do *Timer* foi chamada somado ao valor no registrador *TAR*, o valor resultante sendo dividido por 58, para a transformação total do período do *ECHO* em centímetros.

Por fim foi setada a distância tolerável de $15cm$, caso o obstáculo estivesse numa distância menor deveria ser ligado o *led* correspondente ao sensor.

2) Hardware : À nível de hardware para o ponto de controle 3, foi montado um circuito em *protoboard* para leitura de dos dois sensores.

Para o protótipo final, foi decidido que para conforto, será realizado a confecção unicamente do boné, este contendo os 3 sensores, um frontal e dois diagonais(a ser testado a eficiência). Para a alimentação do circuito será

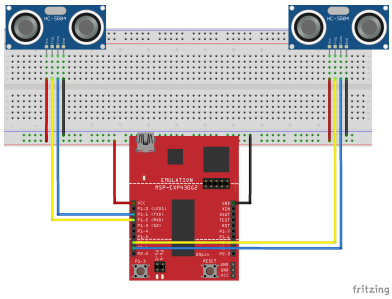

Figura 10. Esquemático circuito com leitura de dois sensores.

utilizada uma bateria de $9V$ e $250mAh$, sendo utilizado o regulador de tensão $LM7805$, para atingir a tensão de $5V$ desejada.

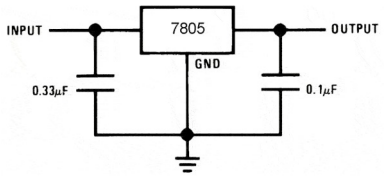

Figura 11. Esquemático circuito regulador de tensão.

G. Ponto de Controle 4

1) Software: Para este ponto de controle o objetivo era conseguir ler os três sensores, mostrar os resultados das distâncias pela comunicação serial e piscar os *LEDs* caso certas distâncias fossem identificadas e realizar a implementação da função do motor *PWM*, explicado como funciona na Secção VII-D. A leitura dos três sensores foi realizada com sucesso utilizando-se a multiplexação, Secção VII-C, porém para o projeto final talvez seja necessário realizar algum tipo de filtragem dos sinais.

No teste do funcionamento foi utilizado o monitor serial do *Energia*, pois este amostra de forma melhor os dados escritos na porta serial. A Figura 12 foi obtida como resultado, durante a realização dos teste, com um objeto na frente do sensor 1. A distância impressa para os outros dois sensores, se dá pelo máximo que o sensor capta e um controle realizado em *Software*.

O ponto principal para realizar esta implementação foi a temporalização das funções, para que as seletoras ficassem setadas por tempo suficiente para o *CI* [10] e ainda os *leds* ficarem acesos por tempo suficiente para a percepção humana. O *Software* funciona da forma exemplificada pelo fluxograma na Figura 13.

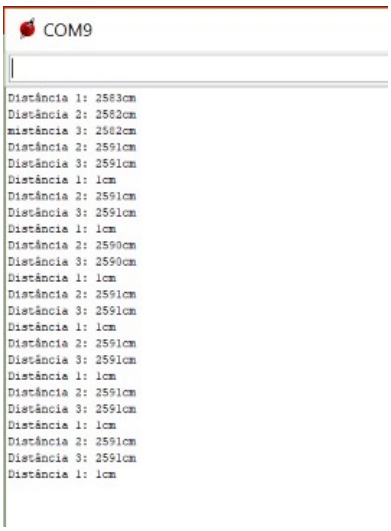

Figura 12. Leitura dos sensores 1, 2 e 3.

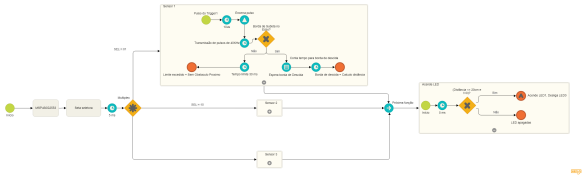

Figura 13. Fluxograma para 3 sensores de ultrassom multiplexados.

Já para o motor *PWM*, a lógica básica usada foi atribuir um tempo grande ao *timer A0* em modo de Captura e Comparação do registrador TACCR0 e porcentagens menores ao mesmo *timer* de registrador TACCR1 - este funciona como o *Duty Cycle*. Como mostra o trecho de código abaixo, conforme o sensor vai lendo as distâncias, é atribuído um valor ao *TACCR1*. Por fim, se o registrador do *timer A0* for menor que o *Duty Cycle* a saída tem seu pino zerado, caso contrário, o pino é setado.

```
while(1){
if (distance>=30 && distance<40)
 TACCR1 = 500; //DUTY CYCLE 50%
else if (distance>=20 && distance<30)
 TACCR1 = 600; //DUTY CYCLE 60%
else if (distance>=10 && distance<20)
 TACCR1 = 700; //DUTY CYCLE 70%
else if (distance<10)
 TACCR1 = 800; //DUTY CYCLE 80%
else
 TACCR1 = 0;
```

```
if (TA0R >= TACCR1)
 P1OUT &= ~PWM;
else
 P1OUT |= PWM;
}
```

2) Hardware: Neste ponto de controle, à nível de hardware o desenvolvido foi o circuito para ligação dos três sensores ao multiplexador e à MSP. A imagem abaixo mostra o circuito montado em *protoboard*.

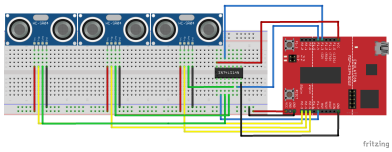

Figura 14. Visão circuito em *protoboard* 3 sensores e MUX

Também foi realizado a implementação teste de um motor com um sensor de ultrassom, porém não há a necessidade de apresentação do circuito, por seu aspecto simples e poucas conexões.

H. Projeto Final

Nesta etapa do projeto, a parte de *Software* já estava concluída restando apenas a prototipagem do projeto. Os três sensores foram ligados, através de jumpers, com seus pinos *ECHO* nas portas de entrada do multiplexador [10], pinos *TRIGGER* na MSP, com portas configuradas como *I/O* [11], pinos *VCC* e GND numa pequena *protoboard*. Eles foram colados nas duas laterais de um capacete e um na frente como mostra a figura 15

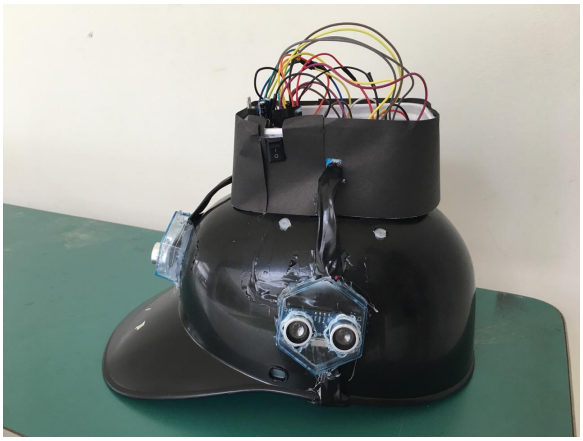

Figura 15. Sensores acoplados no capacete.

Os motores foram acoplados na parte interna do capacete em uma cinta que envolve todo seu diâmetro. Os três fios GND foram soldados em um para reduzir o número de fios que entravam na *protoboard* e os *VCCs* foram ligados na MSP.

Figura 16. Motores acoplados no capacete.

Em uma pequena caixa colada em cima do capacete foram colocados a MSP, a bateria e a protoboard como a figura17 mostra.

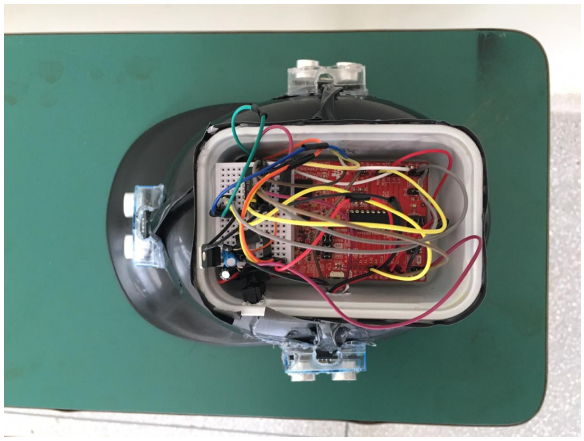

Figura 17. Caixa com os componentes eletrônicos do circuito.

Para alimentação da MSP foi usada uma bateria de 9V e feito um regulador de tensão de $5V$. Essa voltagem foi enviada para um dos pinos de $5V$ da placa, Figura 3, e para um trilha da *protoboard* onde os sensores eram alimentados. A alimentação do multiplexador, foi feita com $3V$, vinha de um dos pinos da MSP, pois esta já possui regulador de tensão interno.[11]

A Figura 18 mostra o protótipo final do assistente senso-motor para deficientes visuais.

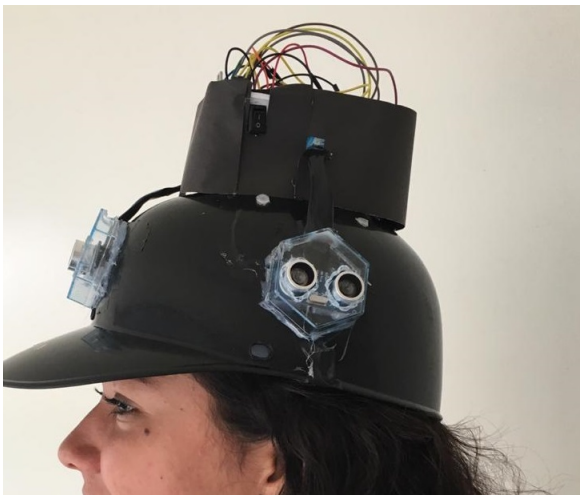

Figura 18. Protótípo final.

VIII. Discussão e Resultados

A. PC2

Nesta etapa do desenvolvimento, o objetivo era realizar a leitura da distância com o sensor, notou-se que a leitura só ocorre de forma eficiente em superfícies planas, porém como primeira etapa de desenvolvimento de *Hardware* e *Software* os resultados foram bons e a partir destes, foi possível visualizar possíveis ameaças de implementação e definir o caminho a seguir para o projeto.

B. PC3

A utilização do código utilizado na Secção VII-E2 para aplicação com dois ou mais sensores não foi possível, pois ao se implementar na placa, não houve acendimento do *led* como era esperado. A não implementação se deu, pois o uso da interrupção como modo de comparação do *Timer A* "corrompia"a leitura dos outros sensores. Com os resultados obtidos a entrega da parte de *Software* não foi efetuada no prazo do PC3, sendo adiada para o PC4, no entanto foi realizado o teste de eficiência para os sensores colocados de forma diagonal com o *Hardware* e *Software* do PC2 e notou-se grandes erros de leitura de distância. Como solução o orientador do projeto deu a ideia de realizar a utilização dos sensores de forma sequencial, enviando o pulso do *Trigger* e análise do Echo, em um sensor de cada vez. Desta forma foi pensado em utilizar a multiplexação do Echo, reduzindo assim número de portas e utilizando uma única porta para interrupção.

C. PC4

Com os resultados obtidos no Ensaio da Secção VII-F, foi necessário uma restruturação do *Hardware* e do *Software*, estas trouxeram resultados bastante satisfatórios, sendo a multiplexação uma saída para varias algumas das ameaças, porém havendo muito ruído e em certos momentos mal contato, sendo que este duplica leitura de um sensor para outro. O teste feito para os motores mostrou que para atribuições ao *TACCR1* abaixo de 30% o tempo de vibração do motor é imperceptível. Assim, para o projeto final os valores de *TACCR0* e *TACCR1* devem ser melhor adaptados.

Devido ao atraso na parte de *Software*, não foi iniciada a prototipagem sendo este um atraso notável.

D. Projeto Final

No projeto final os resultados que necessitavam de correção no ensaio realizado na Secção VII-G, foram resolvidos como o mal contato da multiplexação retirado e o PWM implementado de forma a haver maior sensibilidade de vibração, ao serem implementados ainda fora do capacete o funcionamento foi exato ao estabelecido em *Software*. Porém por algumas deficiências do protótipo a vibração dianteira não foi muito perceptível, pois o motor ficou em contato com o cabelo e não com a testa. Pelo tamanho dos sensores, utilização da *launchpad*, Figura 3, o capacete ficou muito chamativo, sendo necessário para um próximo passo um aquisição de sensores menores e melhor configuração de local para o circuito.

IX. Conclusão

Ao fim, o protótipo exerceu a finalidade para a qual foi criado tendo em vista que os três sensores fizeram a leitura de obstáculos à frente de cada um, enviando os sinais de *ECHO* para a MSP que, por sua vez, encaminhou sinais de vibração para os motores à depender da distância em que os obstáculos se encontravam.

Entende-se que, se a montagem da estrutura houvesse começado antes, problemas do protótipo teriam sido descobertos e resolvidos com mais calma, evitando desperdício de tempo e material, um exemplo seria o melhor posicionamento dos motores de vibração.

Em relação ao conforto e estética, um dos objetivos do projeto, percebe-se que o conforto foi alcançado já que o capacete quando usado não causa incômodo e os motores não vibram em intensidade muito alta a ponto de incomodar. Porém, devido às limitações econômicas, de conhecimento e de tempo, não foi possível a fabricação de um protótipo esteticamente agradável onde os sensores não fossem tão chamativos, os fios não estivessem

tão à mostra e a caixa com os componentes eletrônicos não fosse tão grande.

Em relação ao sensor de ultrassom para uma aplicação real, é necessário um investimento em sensores de maior qualidade, pois o utilizado exige muito gasto computacional, só realiza leituras boas em objetos planos com baixa rugosidade e principalmente sua estrutura não é interressante para a aplicação.

REFERÊNCIAS

[1] F. Dorina. (2010) EstatÍstica da deficiÊncia visual. [Online]. Available: https://www.fundacaodorina.org.br/a-fundacao/deficiencia-visual/estatisticas-da-deficiencia-visual/

[2] L. Hogetop and L. Santarosa, "Tecnologias assistivas: Viabilizando a acessibilidade ao potencial individual," *PGIE - UFRGS*, vol. 5, no. 2, pp. 103–117, 2002.

[3] C. S. G. de SOUZA, J. Sala, A. R. TOGNI, W. O. KAWAMOTO, C. S. P. ARIAS, and L. T. K. JÚNIOR., "Dispositivo para auxílio à locomoção de deficientes vosuais baseado em transdutores ultrasônicos," *Revista Espacios*, vol. 37, no. 09, p. 20, 2016.

[4] L. Brentano. (2011) Brasileiro cria bengala eletrônica de baixo custo para deficientes visuais. [Online]. Available: http://g1.globo.com/tecnologia/noticia/2011/07/brasileiro-cria-bengala-eletronica-de-baixo-custo-para-deficientes-visuais.html

[5] J. H. Davies, *MSP430 microcontroller basics*. Elsevier, 2008.

[6] A. Thomsen. (2015) Como utilizar o sensor ultrasônico hc-sr04. [Online]. Available: http://buildbot.com.br/blog/como-utilizar-o-sensor-ultrasonico-hc-sr04/

[7] T. Lima. (2015) Mux - multiplexador. [Online]. Available: https://www.embarcados.com.br/mux/

[8] C. B. Silveira. (2016) O que é pwm e para que serve? [Online]. Available: https://www.citisystems.com.br/pwm/

[9] N. C. Braga. (2013) Controle pwm de motor dc (mec139). [Online]. Available: http://www.newtoncbraga.com.br/index.php/robotica/5534-mec139

[10] *Dual 4-Line To 1-Line Data Selectors/Multiplexers datasheet (Rev. A)*, Texas Instruments, 5 2007, revised.

[11] *MSP430G2553 $LaunchPad^{TM}$ Development Kit (MSP-EXP430G2ET)*, Texas Instruments, 6 2018.

APÊNDICE A
CÓDIGO FINAL

```
#include <msp430g2553.h>

#define SEL1 BIT7 //B = P1.7
#define SEL0 BIT3 //A = P1.3
#define trigger1 BIT3 // trigger sensor 1 = P2.3 1CO
#define trigger2 BIT4 // trigger sensor 2 = P2.4 1C1
#define trigger3 BIT7 // trigger sensor 3 = P2.7 1C2
#define echo BIT4 // Echo mux = P1.4

#define PWM1 BIT0 // Motor1 = P2.0
#define PWM2 BIT2 // Motor2 = P2.2
#define PWM3 BIT5 // Motor3 = P2.5

#define PWM (PWM1|PWM2|PWM3)
#define triggers (trigger1|trigger2|trigger3)

#define LED1 BIT0 //Led  P1.0
#define LED2 BIT6 //Led  P1.6


int miliseconds;
int distance[3];
long sensor;

void AcendeLED(int dist, int n);
void liga_motor(int dist, int n);
void IniciaUltra(int i);
void setSel(int n);

int main(void)
{

    WDTCTL = WDTPW + WDTHOLD;

    BCSCTL1 = CALBC1_1MHZ;
    DCOCTL = CALDCO_1MHZ;
    // Set de clock TA0
    CCTL0 = CCIE;                             //
        CCR0 interrupt enabled
    CCR0 = 1000;                    // 1us
        at 1mhz
    TACTL = TASSEL_2 + MC_1;
                        // SMCLK, upmode
    CCTL0 = CCIE;                             //
        CCR0 interrupt enabled

    //Set de clock TA1
    TA1CCR0 = 1000; //PER ODO DO PWM
    TA1CCTL1 = OUTMOD_7; //MODO DE SA DA
        DO TIMER0_A: RESET/SET
    TA1CCTL0 = OUTMOD_7;
    TA1CCTL2 = OUTMOD_7;
    TA1CTL = TASSEL_2 + MC_1;         //
        TASSEL_2 -> CLOCK SOURCE: MCLK
        MC_1 -> //TIMER COM CONTAGEM
        PROGRESSIVA DE 0  AT   TACCR1

    _BIS_SR(GIE);

    //Seletoras Multiplex
    P1DIR |= SEL1 + SEL0;
                                      // P1.3
         and P1.7 as output for A e B

    //SET Portas sensor
    P1DIR &= ~echo;          // make pin P1
        .4 input (ECHO)
    P2DIR |= triggers; // P2. 3,4,5 as
        output
    P2OUT &= ~triggers;
    P2SEL &= ~triggers; //Select pin 1.2
        as our PWM output.
    P2SEL2 &= ~triggers;

    // Set motor

    P2DIR |= PWM; //Set pin 1.2 to the
        output direction.
    P2OUT &= ~PWM;


    // Set LEDs
    P1DIR |= LED1 + LED2;
    P1OUT &= ~LED1;
    P1OUT &= ~LED2;

    volatile unsigned int count = 0;
    while(1)
    {
      for (count = 0; count < 3; count++n)
          {
          setSel(count);
          __delay_cycles(100);
          IniciaUltra(count);
          distance[count] = sensor/58;
                        // converting ECHO
              lenght into cm
          AcendeLED(distance[count], count
              );
          liga_motor(distance[count],
              count);

      }
    }
}

void setSel(int n){
  switch(n){
      case 0: // BA = 00
          P1OUT &= ~SEL0;
          P1OUT &= ~SEL1;

          break;
```

```
      case 1: // BA = 01
          P1OUT |= SEL0;
          P1OUT &= ~SEL1;
          break;
      case 2: // BA = 10
          P1OUT &= ~SEL0;
          P1OUT |= SEL1;
          break;

      default:
          P1OUT |= SEL0;
          P1OUT |= SEL1;
        }

}

void IniciaUltra(int i){
        P1IE &= ~0x01;       // disable
            interupt
        switch(i){
          case 0:
            P2OUT |= trigger1;          //
                generate pulse
              break;
          case 1:
            P2OUT |= trigger2;          //
                generate pulse
              break;
          case 2:
            P2OUT |= trigger3;          //
                generate pulse
              break;

          default:
            P2OUT &= ~triggers;
        }
        __delay_cycles(10);             //
             for 10us
        P2OUT &= ~triggers;
        P1IFG = 0x00;                   //
             clear flag just in case
            anything happened before
        P1IE |= echo;           // enable
            interupt on ECHO pin
        P1IES &= ~echo;         // rising
            edge on ECHO pin
        __delay_cycles(30000);          //
             delay for 30ms (after this
            time echo times out if there
            is no object detected)
}

void AcendeLED(int dist, int n)//Acender
    LEDs para debug
{
    if (dist <= 15 && dist != 0){
        switch(n){
        case 0:
            P1OUT &= ~LED1;
            P1OUT |=  LED2;
            break;
        case 1:
            P1OUT |=  LED1;
            P1OUT &=  ~LED2;
            break;
        case 2:
            P1OUT |=  LED1;
            P1OUT |=  LED2;
            break;

        default:
            P1OUT &= ~LED1;
            P1OUT &= ~LED2;
        }
    }
    else{
      if (dist > 21)
      {
          P1OUT &= ~LED1;
          P1OUT &= ~LED2;
      }
    }
}

#pragma vector=PORT1_VECTOR
__interrupt void Port_1(void)
{
    if(P1IFG&echo)
        {
          if(!(P1IES&echo))
          {
            TACTL|=TACLR;   // clears
                timer A
            miliseconds = 0;
            P1IES |= echo;  //
                Interrupi  o para
                falling edge
          }
          else
          {
            sensor = (long)miliseconds
                *1000 + (long)TAR;    //
                Calculo do tamanho do Echo

          }
    P1IFG &= ~echo;             //clear
        flag
    }
}

#pragma vector=TIMER0_A0_VECTOR
__interrupt void Timer_A (void)
{
  miliseconds++;
}


void liga_motor(int dist, int n)
{
    if (dist == 100)
        TA1CCR1 = 300; //DUTY CYCLE DO PWM
```

```
            EM 30%
    else if (dist >= 80  && dist < 100)
        TA1CCR1 = 400; //DUTY CYCLE DO PWM
            EM 40%
    else if (dist >= 60 && dist < 80)
        TA1CCR1 = 500; //DUTY CYCLE DO PWM
            EM 50%
    else if (dist >= 35 && dist < 60)
        TA1CCR1 = 600; //DUTY CYCLE DO PWM
            EM 60%
    else if (dist >= 20 && dist < 35)
        TA1CCR1 = 700; //DUTY CYCLE DO PWM
            EM 70%
    else if (dist < 20 && dist !=0)
        TA1CCR1 = 800; //DUTY CYCLE DO PWM
            EM 80%
    else
        TA1CCR1 = 0;

    switch(n){
                case 0:
                    //executa o motor
                    if (TA1R >= TA1CCR1)
                        P2OUT &= ~PWM1;
                    else
                        P2OUT |= PWM1;
                  break;
                case 1:
                    //executa o motor
                    if (TA1R >= TA1CCR1)
                        P2OUT &= ~PWM2;
                    else
                        P2OUT |= PWM2;
                  break;
                case 2:
                    //executa o motor
                    if (TA1R >= TA1CCR1)
                        P2OUT &= ~PWM3;
                    else
                        P2OUT |= PWM3;
                  break;
                default:
                    P2OUT &= ~PWM;
              }

    }
```