Proposta de Projeto

Cinto de Ecolocalização para Cegos

Adriano Silva de Moraes
Faculdade do Gama
Universidade de Brasília - UnB
Gama-DF, Brasil

Hachid Habib Cury
Faculdade do Gama
Universidade de Brasília - UnB
Gama-DF, Brasil

I. Justificativa

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 brasileira. Entre os deficientes visuais 528.624 pessoas são incapazes de enxergar (cegos), de acordo com os dados do IBGE de 2010. Visando incluir da melhor maneira os deficientes visuais e resolver problemas que os portadores de deficiência visual enfrentam, como a mobilidade reduzida e a dependência de algum auxílio como cães guias, bengalas, parentes, amigos e etc.

O mercado oferece produtos similares ao proposto no projeto. As bengalas sensoriais funcionam com sensores ultrassônicos que possibilitam a identificação que obstáculos no caminho do usuário, o preço desse produto chega a 1350.00 €, ou seja R$ 5895.00 como mostrado na figura 1.

Figura 1: Bengala sensorial comparação de preço.

O mesmo projeto já foi trabalhado por engenheiros como o brasileiro Rodrigo Azevedo. Ele é um dos responsáveis pelo projeto Bengala IoT, que visa criar um modelo replicável de uma bengala inteligente para deficientes visuais. De código aberto e sem fins lucrativos, conforme o criador do projeto, o pacote básico que é composto por dois sensores e um microcontrolador simples, poderia sair na faixa dos R$ 100 ou R$ 200.

II. Objetivos

A dupla propõe a criação de um cinto eletrônico composto por sensores ultrassônicos e vibradores, que ofereçam ao usuário a capacidade de se orientar no ambiente sem os incômodos gerados pelas bengalas, que são comumente utilizadas, já que os produtos que utilizam a mesma tecnologia proposta possuem preços exorbitantes para a maioria da população cega.

III. Benefícios

Os usuários devem tirar os mesmos proveitos que as ferramentas atuais, ou seja, saber com antecedência a distância dos obstáculos, evitando assim acidentes de locomoção, dar mais liberdade ao usuário para se locomover em público com mais autoconfiança, mas com o diferencial de utilizar um equipamento sutil e sem inconvenientes para tal.

IV. hardware

Lista de materiais:

- MSP-EXP430G2553LP
- Quatro sensores de distância ultrassônicos (HC-SR04)
- Quatro vibracall's (Modelo F1030300112)
- Resistores 4.7 KΩ
- Resistores 10KΩ
- Quatro Transistores BCE549B
- Protoboard
- Jumpers
- Cinta elástica

Descrição do Hardware:

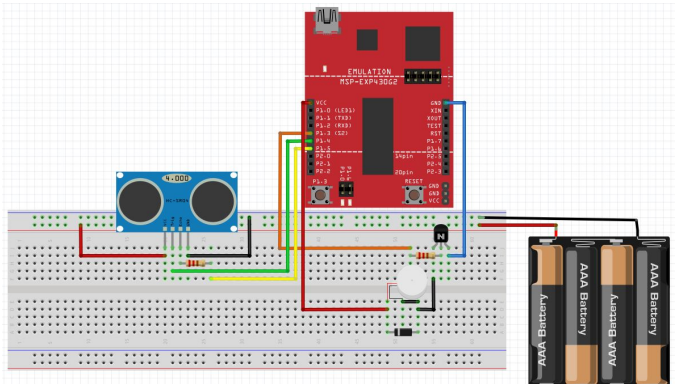

Figura 2: Esquemático do circuito apresentado.

A cinta elastica será a base para os quatro sensores, microcontrolador e para os demais dispositivos eletrônicos, conterá toda a circuitaria.

Os sensores ficarão distribuídos da seguinte maneira, 1 sensor apontado para cima de modo que identifique objetos a 50 cm do rosto do usuário, 1 sensor virado para baixo para que identifique obstáculos a 50 cm dos pés do mesmo e 2 sensores virados para frente, para que além de mostrar ao usuário obstáculos a 1 metro a sua frente, distingue em que lado se encontram, possibilitando assim ao utilizador da cinta de ecolocalização saber onde está a obstrução e a que distância se encontra do mesmo. Para isso, serão distribuídos vibracall's ao longo da cinta que funcionarão da seguinte forma, os vibracall da esquerda e da direita mostram objetos que estão na esquerda e à direita do usuário e sua proximidade por vibração (objetos mais pertos vibraram mais que objetos mais longes), o sensor de cima vibrará com força total quando identificar algo perto do rosto do utilitário e o sensor de baixo terá dois modos de vibração, vibrará ininterruptamente para degraus, meu fios e etc, e múltiplas vibrações para buracos.

O microcontrolador a ser utilizado no projeto será o M430G2553 da Texas Instruments (Figura 3). Esse dispositivo trabalha na faixa de 3.3V, assim como os vibracalls, enquanto os sensores de distância operam a 5V (Figura 4) e a bateria a 9V (Figura 5).

Embora os componentes escolhidos sejam bastante populares e acessíveis exigem a utilização de vários divisores de tensão para operarem em conjunto. Uma alternativa para esse inconveniente é utilização da placa MB102, mostrada na Figura 6. Trata-se de um módulo de fonte de alimentação, que recebe uma tensão de 5V a 12V em uma de suas entradas e oferece as tensões de 3.3V ou 5V em seus terminais de saída. Os terminais de entrada oferecem interface para alimentação via USB ou bateria comum.

Figura 3: Microcontrolador M430G255 (3.3V)

Figura 4: Sensor de distância HC-SR04 (5V)

Figura 5: Bateria alcalina (9V)

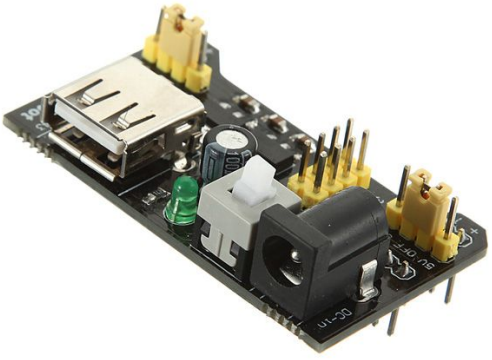

Figura 6: Módulo de alimentação MB102 3.3V/5V

V. SOFTWARE

O código foi criado para ser compilado pelo software Code Composer Studio e possui 2 funções, a função Atraso_us é utilizada para criar atrasos em microsegundos e é utilizada no código para controle do tempo mínimo que Trigger do sensor ultrassônico fique em alto. A função SensorUlt foi criada para que o projetista informe a posição do Trigger e Echo no MSP430 e seu retorno é a distância em centímetros, para isso é necessário levar Trigger para o nível alto durante 10μs e esperar que Echo retorne em nível alto, com isso é utilizado o TIMER_A1 CC2 do MSP em modo de captura para ler o tempo do pulso em microsegundos. Com a leitura do pulso utiliza-se a seguinte equação para descobrir a distância do objeto a frente do sensor.

$$Distancia = \frac{T_{Pulso} \times V_{Som}}{2} \simeq \frac{T_{Pulso}}{58}$$

Já que a velocidade do som igual a:

$$V_{Som} = 0{,}0343 \ \frac{cm}{\mu s}$$

A main do programa possui um loop infinito que chama a função SensorUlt e joga seu retorno em uma variável inteira nomeada "distancia", o valor de distancia é comparado com faixas de distâncias e se seu valor encaixar na faixa é dado um valor para `DUTY_CYCLE` que com a utilização do TIMER_A1 em modo comparação é transmitido para a saída Vibracall e LED uma onda quadrada PWM. A execução da main fornece ao Led2 e ao motor vibracall uma intensidade maior com a diminuição da distância entre o sensor e o objeto, ou seja, quanto mais perto do objeto maior deve ser a vibração do motor e a luminosidade do led.

A geração do sinal PWM foi implementado na função PwmVibra(PWM, distancia) - omitimos o tipo dos argumentos para melhor visualização. O parâmetro PWM indica os pinos de saída do sinal e o parâmetro distancia fornece as informações para modulação desse sinal. Até este ponto 2 sinais de PWM são gerados, utilizando os pinos conectados aos leds (P1.0 e P1.6) da placa Launchpad como saída. Nesses pinos foram também conectados os vibracalls.

Para os vibracalls foram implementadas três velocidades, baseadas na distância do obstáculo. A primeira velocidade é fornecida por um sinal PWM com DUTY_CYCLE de 30%, assim que um obstáculo é detectado a menos de 50cm. A segunda velocidade é fornecida por um sinal PWM com DUTY_CYCLE de 70%, assim que um obstáculo é detectado a menos de 30cm. Por fim, terceira velocidade é fornecida por um sinal PWM com DUTY_CYCLE de 100%, assim que um obstáculo é detectado a menos de 10cm. Essa definição está concentrada na função PwmVibra(PWM, distancia) e as chamadas a ela são feitas dentro do código principal.

VI. DISCUSSÃO

A equipe superou as dificuldades de alimentação dos componentes que operam em diferentes níveis de tensão. Isso ocorreu com a utilização da placa MB102, já explicada anteriormente.

Também obtivemos sucesso com a nova abordagem na geração do sinal PWM, que resolveu o problema da imperceptibilidade que ocorria ao mudar a velocidade do vibracall. Embora essa abordagem não gere os sinais de forma paralela e sim um ciclo PWM por vez para cada vibracall, o resultado alcançado foi satisfatório até o momento.

VII. CONCLUSÃO

A evolução do trabalho ocorreu como desejado, já que o projeto já está sendo executado no software CCS e todos elementos do projeto estão em funcionamento, faltando apenas poucos ajustes. O próximo desafio será a união de todas componentes com suas devidas alimentações na cinta elástica.

VIII. REVISÃO BIBLIOGRÁFICA

[1] Davies, J., MSP430 Microcontroller Basics, Elsevier, 2008.

[2] BRASIL, IBGE. Pesquisa sobre quantidade de cegos no Brasil, Censo Demográfico, 2010. Disponivel em: <www.ibge.gov.br>. Acesso em: 30/04/2018.

[3] Bengala eletrônica Ultracane , comparativo de preço no mercado. Disponivel em:<https://www.megaserafim.pt/online/orientacao-e-mobilidade/41-bengala-electronica-ultracane.html> Acesso em: 03/06/2018.

[4] Brasileiro cria bengala inteligente que usa sensores para guiar deficientes. Disponivel em:<https://www.tudocelular.com/tech/noticias/n119414/cpbr11-brasileiro-bengala-inteligente-iot.htmll> Acesso em: 03/06/2018.

IX. Anexos

```
#include <msp430.h>

//Entradas e Saídas do sensor 0 (Fronal Esquerda)
#define Vibra0 BIT6
#define Trig0 BIT4
#define Echo0 BIT5
unsigned int distancia0=0;

//Entradas e Saídas do sensor 1 (Fronal Direita)
#define Vibra1 BIT0
#define Trig1 BIT1
#define Echo1 BIT2
unsigned int distancia1=0;

//Variaveis Gerais
unsigned int TempoI=0;
unsigned int TempoF=0;
unsigned int TempoPWM=0;
unsigned int DUTY_CYCLE=0;
#define PERIODO 0xFFFF

//Função Criada para causar atraso em microsegundos
void Atraso_us(volatile unsigned int us)
{
	TA1CCR0 = us-1;
	TA1CTL = TASSEL_2 + ID_0 + MC_1;

	while((TA1CTL & TAIFG)==0);
	TA1CTL = TACLR;
}

//Função criada para controlar sensores ultrassônicos a uma distância máxima de 2.5m
int SensorUlt(volatile unsigned int Trig, volatile unsigned int Echo)
{
	int distancia;

	P1DIR |= Trig;
	P1OUT &= ~Trig;
	P1DIR &= ~Echo;

	P1OUT |= Trig;                          //Colocar Trig em alto durante 10 us
	Atraso_us(10);

	TA1CTL = TACLR;                         // Zerando o TAR
	TA1CTL = TASSEL_2 + ID_0 + MC_2;        // Configurando clock do timer
	TA1CCTL2 = CM_3 + CCIS_0 + CAP;  // Configurando modo de capitura
```

```
	P1OUT &= ~Trig;

	while((P1IN&Echo)==0);
	TempoI=TA1R;
	while((P1IN&Echo)==Echo);
	TempoF=TA1R;

	distancia = (TempoF - TempoI)/58;	// Problema para Capiturar o TAR (disrancia =
distancia/58)

	return distancia;
}

//Função Verifica PWM

void PwmVibra(volatile unsigned int PWM, volatile unsigned int distancia)
{
	P1DIR |= PWM;
	P1OUT &= ~PWM;

	//Configura PWM
	if(distancia<=10){
		DUTY_CYCLE= PERIODO-2;
	}
	else if (distancia<=30){
		DUTY_CYCLE= PERIODO*0.70;
	}
	else if (distancia<=50){
		DUTY_CYCLE= PERIODO*0.30;
	}
	else if (distancia<=100){
		DUTY_CYCLE= 0;
	}
	else if (distancia<=125){
		DUTY_CYCLE= 0;
	}
	else {
		DUTY_CYCLE= 0;
	}

	//Executa PWM
	TempoPWM=TA0R;
	if(TempoPWM>=DUTY_CYCLE){
		P1OUT &= ~PWM;
	}else{
		P1OUT |= PWM;
	}

}

void main(void)
{
	BCSCTL1 = CALBC1_1MHZ;
	DCOCTL = CALDCO_1MHZ;			// SMCLK = 1M Hz
```

```
	WDTCTL = WDTPW + WDTHOLD;		// Stop WDT

	TA0CTL = TACLR;				// Zerando o TAR
	TA0CTL = TASSEL_2 + ID_0 + MC_2;	// Configurando clock do timer
	TA0CCTL1 = CM_3 + CCIS_0 + CAP ;	// Configurando modo de capitura + CCIE

while(1){

	distancia0= SensorUlt(Trig0, Echo0);
	PwmVibra(Vibra0,distancia0);
	PwmVibra(Vibra1,distancia1);

	distancia1 =  SensorUlt(Trig1, Echo1);
	PwmVibra(Vibra0,distancia0);
	PwmVibra(Vibra1,distancia1);

 }
}
```