PROJETO DE UM ROBÔ COM CONTROLE DE SEU TRAJETO ATRAVÉS DE SENSORES OPTOACOPLADORES E UTILIZAÇÃO DE MOTORES CC

Evelyn Renata de Moraes (evisbr@yahoo.com.br)
Philippe Eduardo Pamplona Cardoso (philippeeduardo@yahoo.com.br)
Riciana de Souza Guanabara Santiago (ricisantiago@netscape.net)

A integração de disciplinas, a necessidade do trabalho em equipe e o desenvolvimento de facilidades e idéias foram os principais objetivos deste projeto, além de ensinar aos alunos noções de gerenciamento e execução de projetos, realidades no cotidiano de um engenheiro e que têm que ser estudadas e absorvidas o quanto antes.

Este projeto foi realizado em duas etapas trabalhadas de forma independentes: o desenvolvimento de um Software para fazer o controle automático e servir de "inteligência" para o Robô (ao se falar robô pode-se entender também como carrinho e vice-versa) e o desenvolvimento do Hardware do carrinho com seus motores, bem como as placas para suportarem o "cérebro" e as outras junções para comunicação entre o Software e Hardware.

O Projeto objetivava que o Robô recebesse um sinal dizendo qual o caminho a seguir e o mesmo pudesse percorrer o trajeto escolhido sem interferência humana, eis a figura com o esquema dos caminhos:

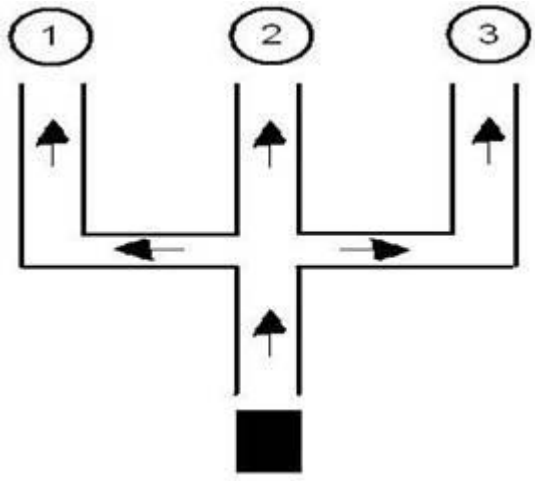

Figura 1: Opções de caminho que podem ser percorridos pelo robô.

Devido a isso, foi montado um esquema (tabela verdade) com as possibilidades de leitura dos sensores e as atitudes que deveriam ser tomadas pelo circuito, ao interpretar tais sinais recebidos. O esquema a seguir mostra a tabela verdade desenvolvida:

A	B	C	Instrução
0	0	0	Parado
0	0	1	Giro horário
0	1	0	Frente
0	1	1	Giro horário
1	0	0	Giro anti-horário
1	0	1	(posição inexistente)
1	1	0	Giro anti-horário
1	1	1	Decisão

Figura 2: Tabela verdade enviada pelos sensores para interpretação do Robô sobre as atitudes a serem tomadas.

Agora trataremos as fases de desenvolvimento do SW e do HW de forma independente, começando pelo Software.

SOFTWARE

O Software foi desenvolvido utilizando a ferramenta PICLITE, no MPLAB, com a qual pudemos programar utilizando linguagem C de uma forma voltada à programação dos microcontroladores e Ports dos mesmos.

Eis o código fonte comentado:

```
#include<pic.h>
#include <stdio.h>


char le_sensor1 (void);
char le_sensor2 (void);
char le_sensor3 (void);
void ver_parado (char s1, char s2, char s3);
void ver_reto (char s1, char s2, char s3);
void ver_direita (char s1, char s2, char s3);
void ver_esquerda (char s1, char s2, char s3);
```

```
void ver_caminho_escolhido (char s1, char s2, char s3);
void caminho_esquerda (char s1, char s2, char s3);
void caminho_direita (char s1, char s2, char s3);
void poe_valor_no_MD(char valor);
void poe_valor_no_ME(char valor);


void main (){
      char s1, s2, s3;
      TRISC=0xFC;                                       // define os
PORTS 6 e 7 como input e os PORTS de 0 a 5 como
outputs
      while(1){                                         //loop infinito//
            s1=le_sensor1();                            //lê o sensor 1//
            s2=le_sensor2();                            //lê o sensor 2//
            s3=le_sensor3();                            //lê o sensor 3//
            ver_parado(s1,s2,s3);                       //vai para a
função para ver se está parado//
            }
}


char le_sensor1 (void){
      char sensor;
      sensor = RC2;                                     //pega o valor
que tá no PORTC 0
      return sensor;
}

char le_sensor2 (void){
      char sensor;
      sensor = RC3;                                     //pega o valor
que tá no PORTC 1
      return sensor;
}

char le_sensor3 (void){
      char sensor;
      sensor = RC4;                                     //pega o valor
que tá no PORTC 2
      return sensor;
}

void ver_parado (char s1, char s2, char s3){

      if(s1==0 && s2==0 && s3==0){                      //vê se o
carrinho tá fora do caminho, se estiver ele desliga os motores
            poe_valor_no_MD(1);                         // lógica
invertida para polarização de referencia dos motores: 0 liga e 1
desliga
            poe_valor_no_ME(1);
      }
      else
            ver_reto(s1,s2,s3);                         //senão ele vê se
está em linha reta
}

void ver_reto (char s1, char s2, char s3){

      if(s1==0 && s2==1 && s3==0){                      //vê se está em
linha reta e liga os motores
```

```
            poe_valor_no_MD(0);
            poe_valor_no_ME(0);
      }
      else
            ver_direita(s1,s2,s3);                    //senão ele vê se
está para a direita
}

void ver_direita (char s1, char s2, char s3){

      if(s1==0 && s2==0 && s3==1){                    //vê se está para
a direita
            poe_valor_no_MD(1);                       //liga o motor
esquerdo
            poe_valor_no_ME(0);
      }
      else
            ver_esquerda(s1,s2,s3);                   //senão ele vê se
está para a esquerda
}

void ver_esquerda (char s1, char s2, char s3){

      if(s1==1 && s2==0 && s3==0){                    //vê se está para
a esquerda
            poe_valor_no_MD(0);                       //liga o motor
direito
            poe_valor_no_ME(1);
      }
      else
            ver_caminho_escolhido(s1,s2,s3);          //senão ele vê se
está na linha de decisão
}

void ver_caminho_escolhido (char s1, char s2, char s3){
      char b1, b2, b3;
      b1=RC5;
      b2=RC6;
      b3=RC7;
      poe_valor_no_MD (1);                            //desliga os
motores
      poe_valor_no_ME (1);
      if(s1==1 && s2==1 && s3==1){                    //vê se está na
linha de decisão
            if(b1==1)                                 //vê se o botão 1
foi acionado, se foi acionado ele vai para a função esquerda
                  caminho_esquerda(s1,s2,s3);
            if(b2==1){                                //senão vê se o
botão 2 foi acionado, se foi acionado ele continua com os dois
                  while(s1!=0 && s2!=0 && s3!=0 || s1=s3==0 && s2==1){
                        poe_valor_no_MD(0);           //motores ligados
e anda em linha reta
                        poe_valor_no_ME(0);
                        s1=le_sensor1();              //lê o sensor 1//
                        s2=le_sensor2();              //lê o sensor 2//
                        s3=le_sensor3();              //lê o sensor
3//
                  }
            }
            if(b3==1)
```

```
                  caminho_direita(s1,s2,s3);          //senão vê se o
botão 3 foi acionado, se foi acionado ele vai para a função direita
            else {
                  poe_valor_no_MD(1);                 //desliga os
motores
                  poe_valor_no_ME(1);
            }
      }
}

void caminho_esquerda (char s1, char s2, char s3){
      while(s1!=0 || s2!=0 || s3!=0){
            s1=le_sensor1();                          //lê o sensor 1//
            s2=le_sensor2();                          //lê o sensor 2//
            s3=le_sensor3();                          //lê o sensor 3//
            if(s1==1 && s2==1 && s3==1){              //enquanto o s2
não for 1 e s1 e s3 não forem 0 o carrinho só aciona o motor
                  poe_valor_no_MD(0);
                  poe_valor_no_ME(1);                 //direito, para
ele virar para a esquerda
            }
            if(s1==0 && s2==1 && s3==0){              //enquanto só s2
for 1 os dois motores ficam ligados
                  poe_valor_no_MD(0);
                  poe_valor_no_ME(0);
            }
            if(s1==0 && s2==1 && s3==1){              //enquanto s2 e
s3 forem diferentes de 0 só o motor esquerdo é acionado,
                  poe_valor_no_MD(1);
                  poe_valor_no_ME(0);                 //para o carrinho
virar para a direita
            }
            if(s1==0 && s2==1 && s3==0){              //enquanto só s2
for 1 os dois motores ficam ligados
                  poe_valor_no_MD(0);
                  poe_valor_no_ME(0);
            }
      }
      poe_valor_no_MD (1);
      poe_valor_no_ME (1);
}

void caminho_direita (char s1, char s2, char s3){

      while(s1!=0 || s2!=0 || s3!=0){
            s1=le_sensor1();                          //lê o sensor 1//
            s2=le_sensor2();                          //lê o sensor 2//
            s3=le_sensor3();                          //lê o sensor 3//
           poe_valor_no_MD(0);                        //liga os motores
            poe_valor_no_ME(0);
            if(s1!=0 && s2!=1 && s3!=0){              //enquanto o s2
não for 1 e s1 e s3 não forem 0 o carrinho só aciona o motor
                  poe_valor_no_MD(1);
                  poe_valor_no_ME(0);                 //esquerdo, para
ele virar para a direita
            }
            if(s1==0 && s2==1 && s3==0){              //enquanto só s2
for 1 os dois motores ficam ligados
                  poe_valor_no_MD(0);
                  poe_valor_no_ME(0);
            }
```

```
            if(s1!=0 && s2!=0 && s3!=1){              //enquanto s2 e
s3 forem diferentes de 0 só o motor direito é acionado,
                  poe_valor_no_MD(0);
                  poe_valor_no_ME(1);                 //para o carrinho
virar para a esquerda
            }
            if(s1==0 && s2==1 && s3==0){              //enquanto só s2
for 1 os dois motores ficam ligados
                  poe_valor_no_MD(0);
                  poe_valor_no_ME(0);
            }
      }
      poe_valor_no_MD (1);
      poe_valor_no_ME (1);                           //desliga os
motores
}

void poe_valor_no_MD(char valor){
      RC0 = valor;                                   //poe no PORTC3 o
valor do motor direito para ligar ou desligar
}

void poe_valor_no_ME(char valor){
      RC1 = valor;                                   //poe no PORTC4 o
valor do motor esquerdo para ligar ou desligar
}
```

Agora os Fluxogramas de cada função independente e da função main:

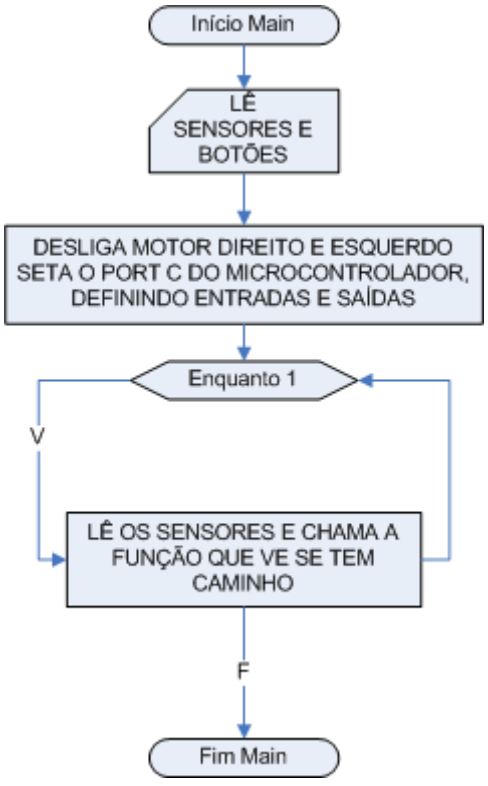

Figura 3: Fluxograma do Main.

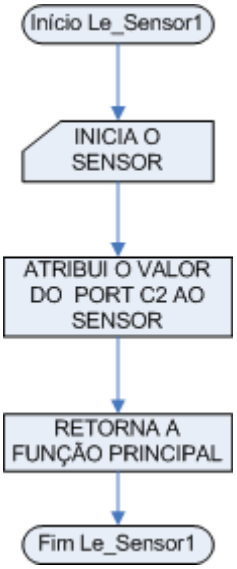

Figura 4: Fluxograma do Lê_sensor1.

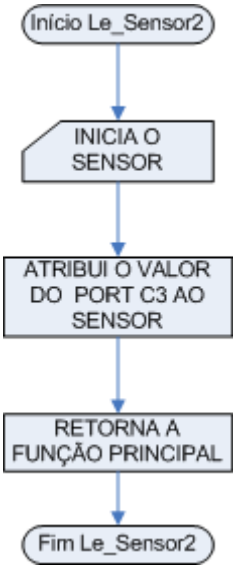

Figura 5: Fluxograma do Lê_sensor2.

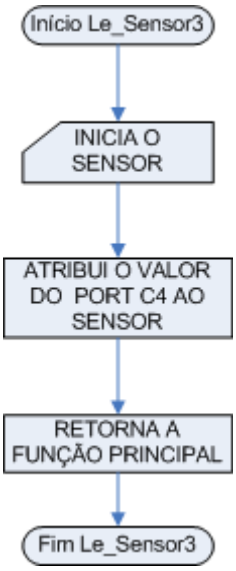

Figura 6: Fluxograma do Lê_sensor3.

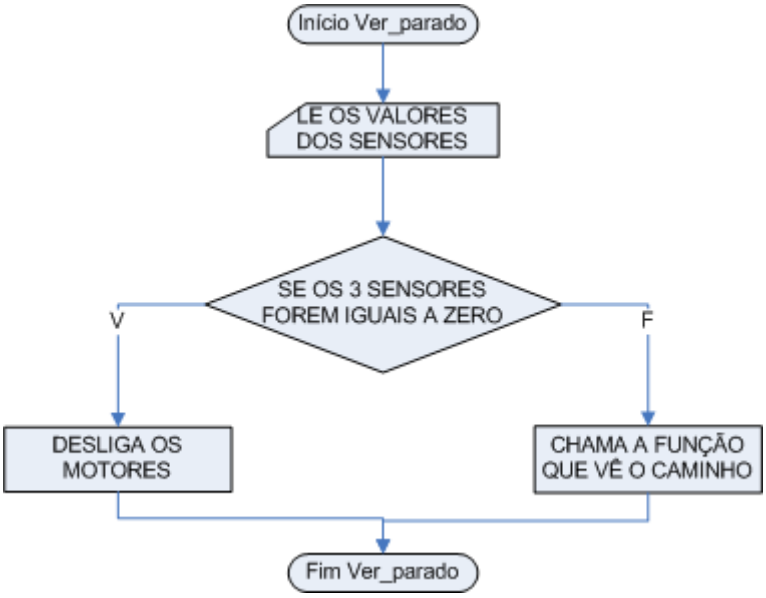

Figura 7: Fluxograma do Ver_parado.

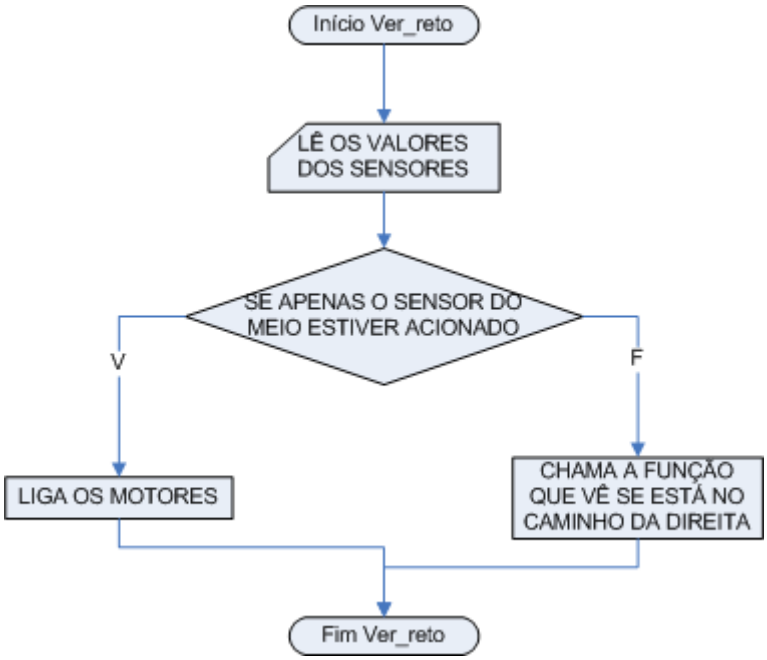

Figura 8: Fluxograma do Ver_reto.

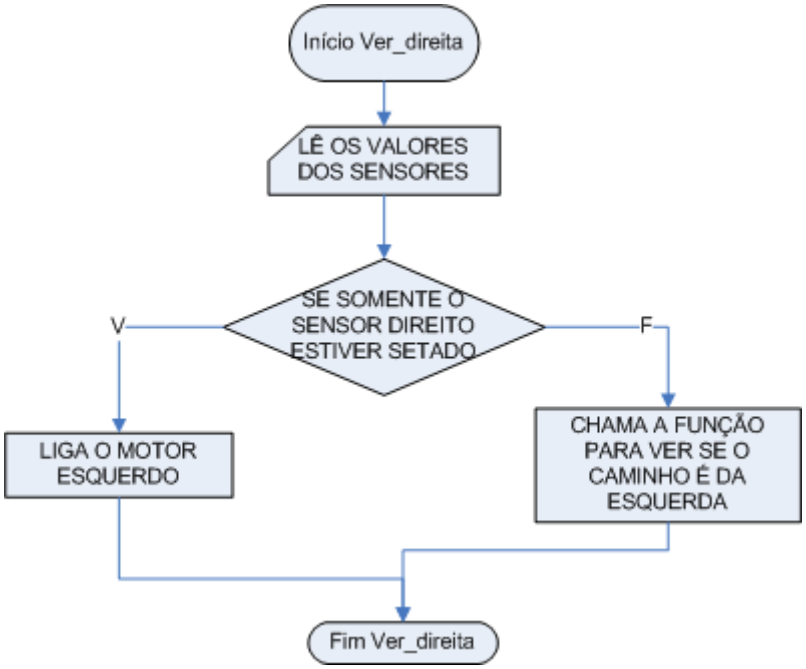

Figura 9: Fluxograma do Ver_direita.

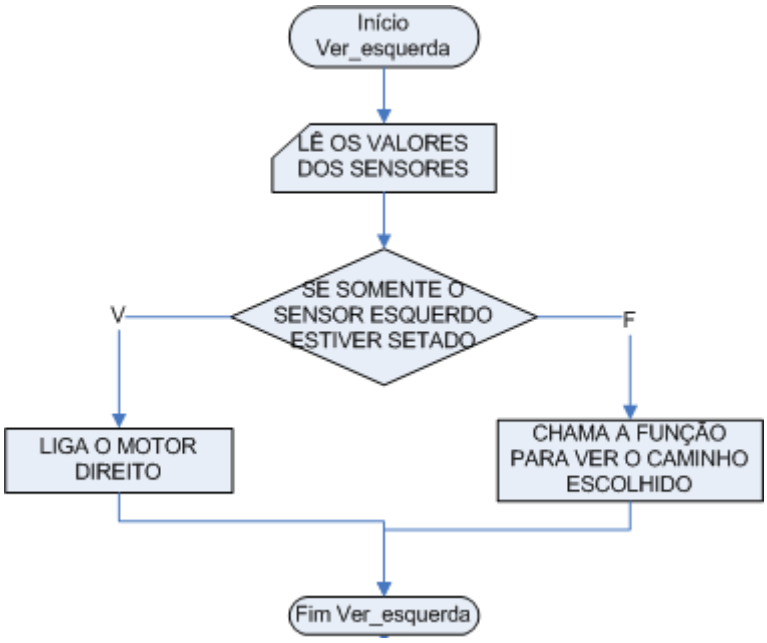

Figura 10: Fluxograma do Ver_esquerda.

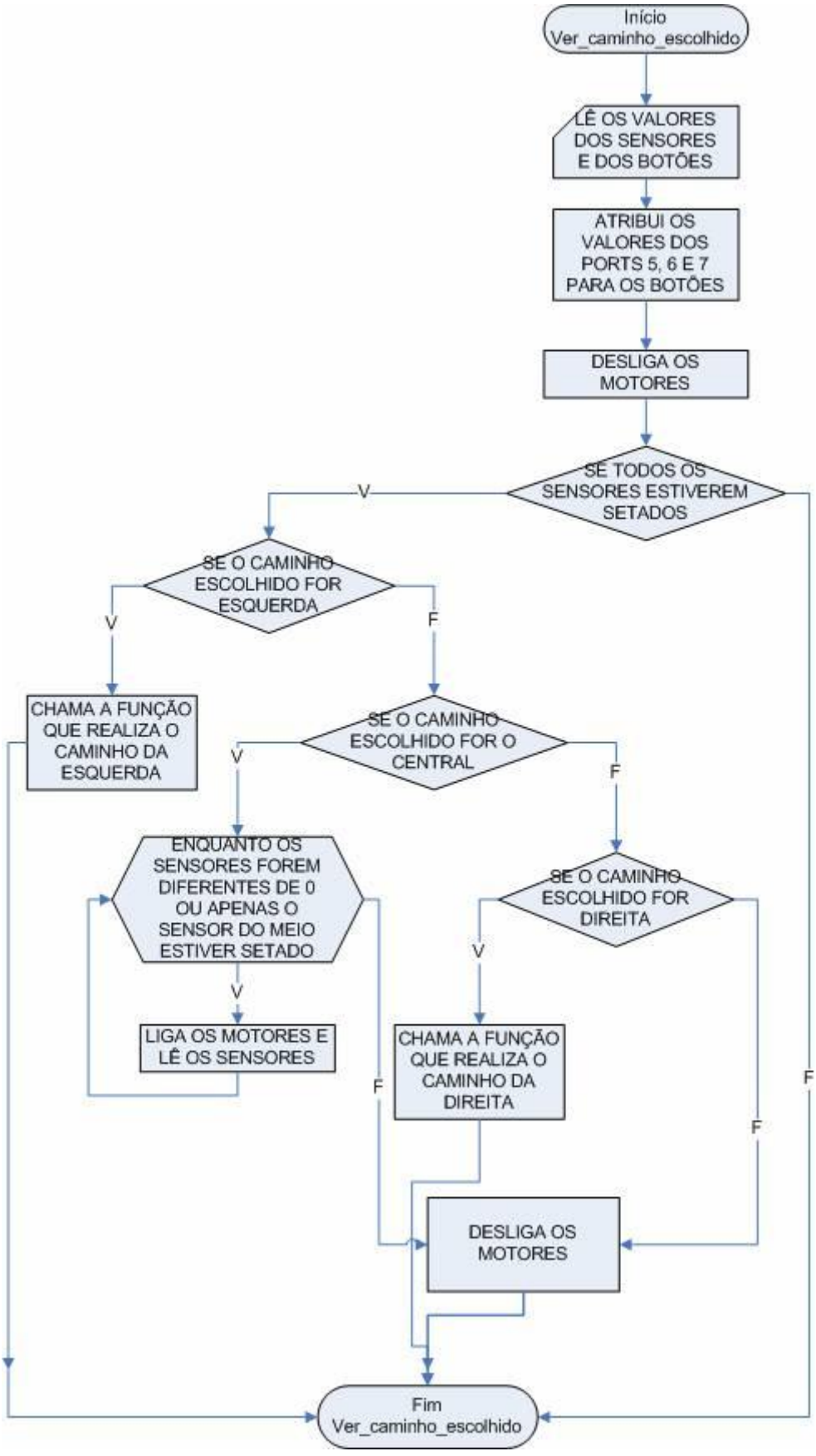

Figura 11: Fluxograma Ver_caminho_escolhido.

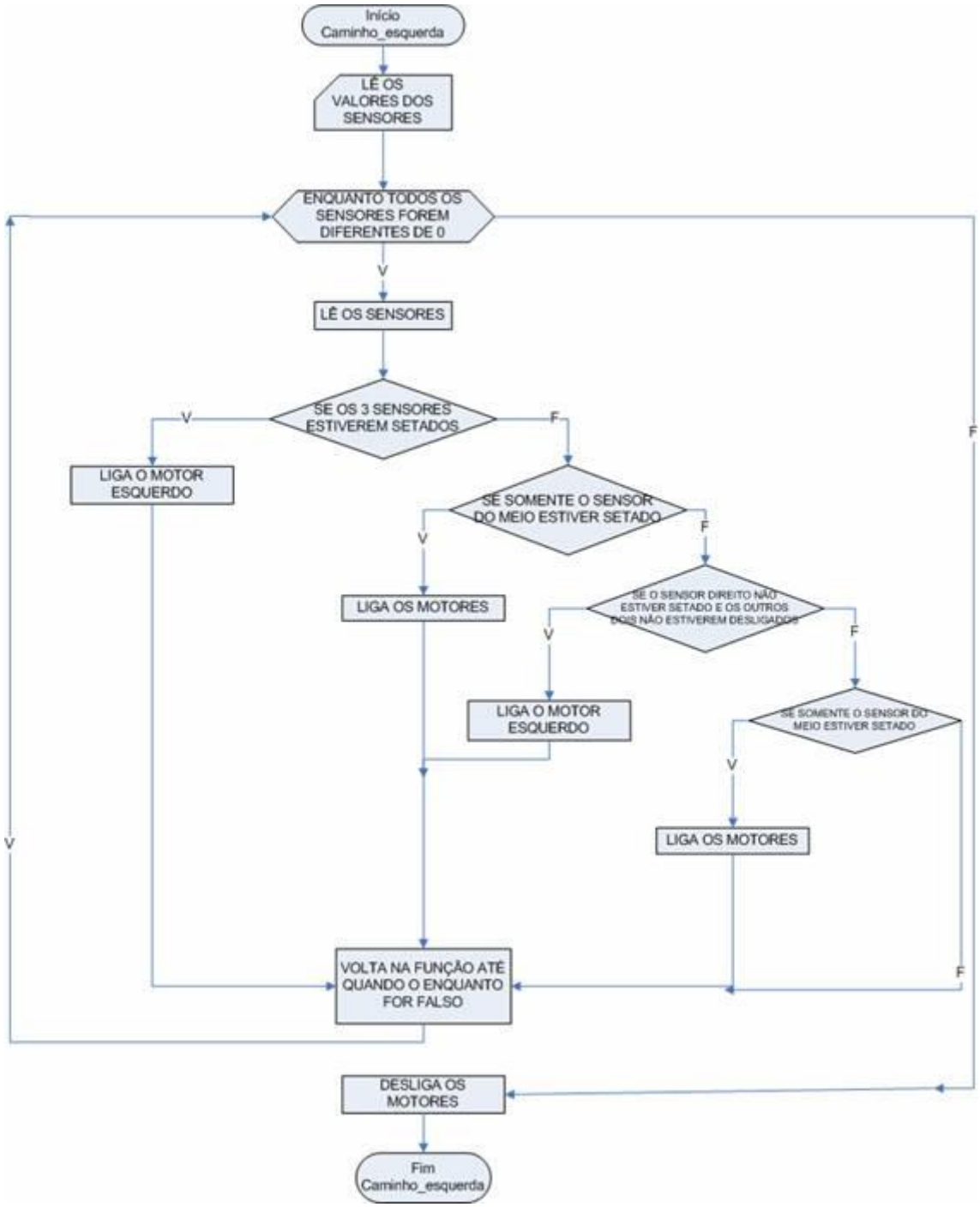

Figura 12: Fluxograma Caminho_esquerda.

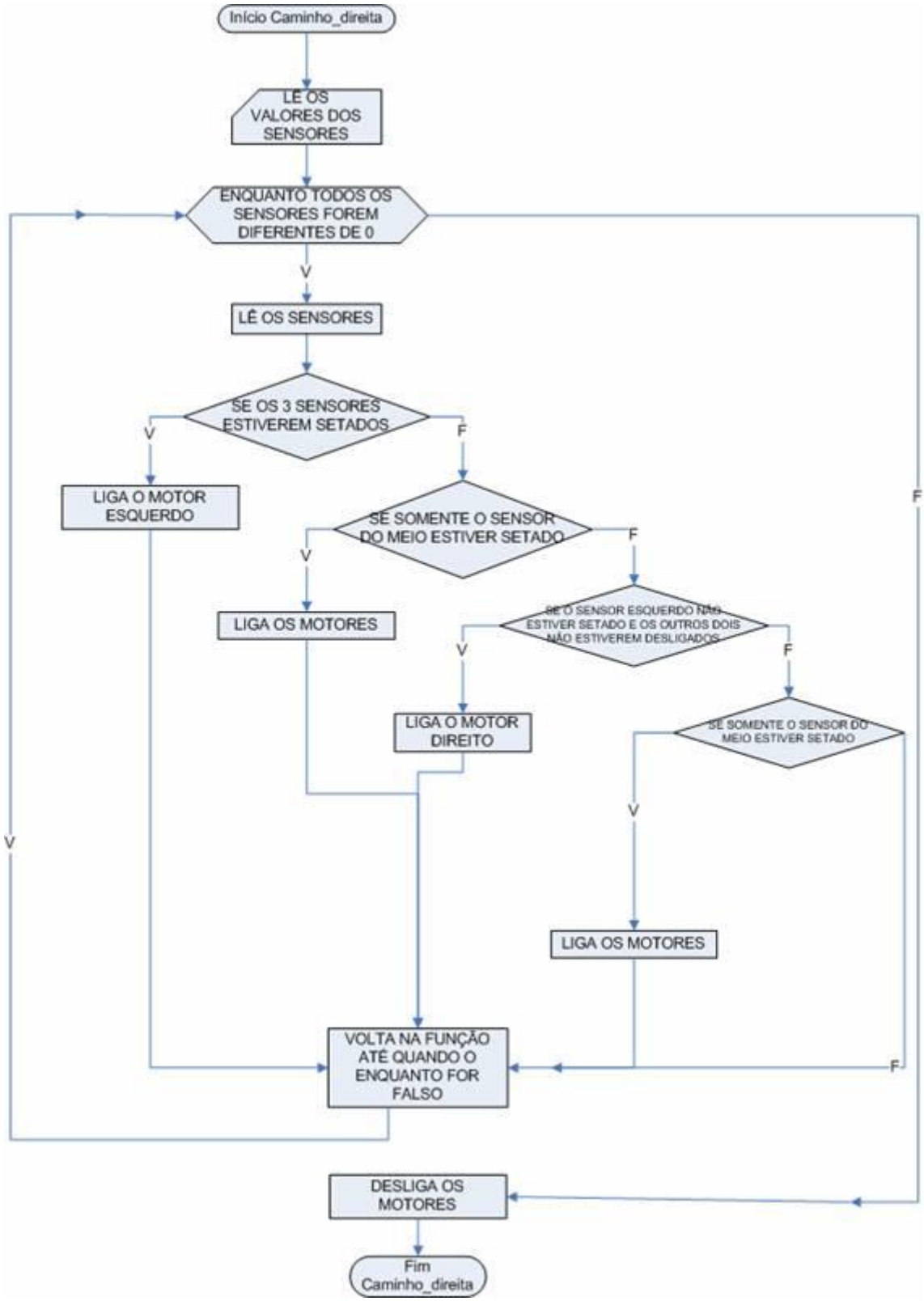

Figura 13: Fluxograma Caminho_direita.

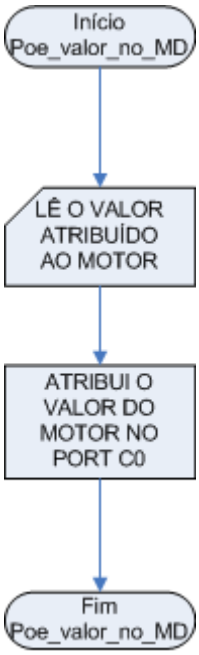

Figura 14: Fluxograma Poe_valor_no_MD.

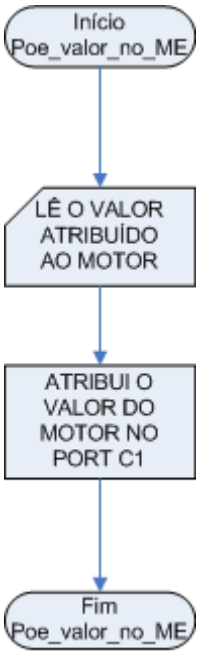

Figura 15: Fluxograma Poe_valor_no_ME.

HARDWARE

Utilizamos como base para nosso raciocínio um projeto semelhante ao nosso encontrado na internet. Este projeto tratava da construção de um robô com o mesmo princípio desejado, mas trabalhando com muito mais sensores e outra linguagem de programação muito diferente da que planejamos utilizar. Mesmo assim, o arquivo encontrado foi de grande valia para o desenvolvimento de nosso raciocínio e pode ser visto integralmente no seguinte link: ProjetoRobo.pdf

Além disso, precisamos desenvolver circuitos e táticas para que o SW e o HW "conversassem", para isso utilizamos a ferramenta PsPice. À baixo temos as estruturas e os layouts das placas utilizadas:

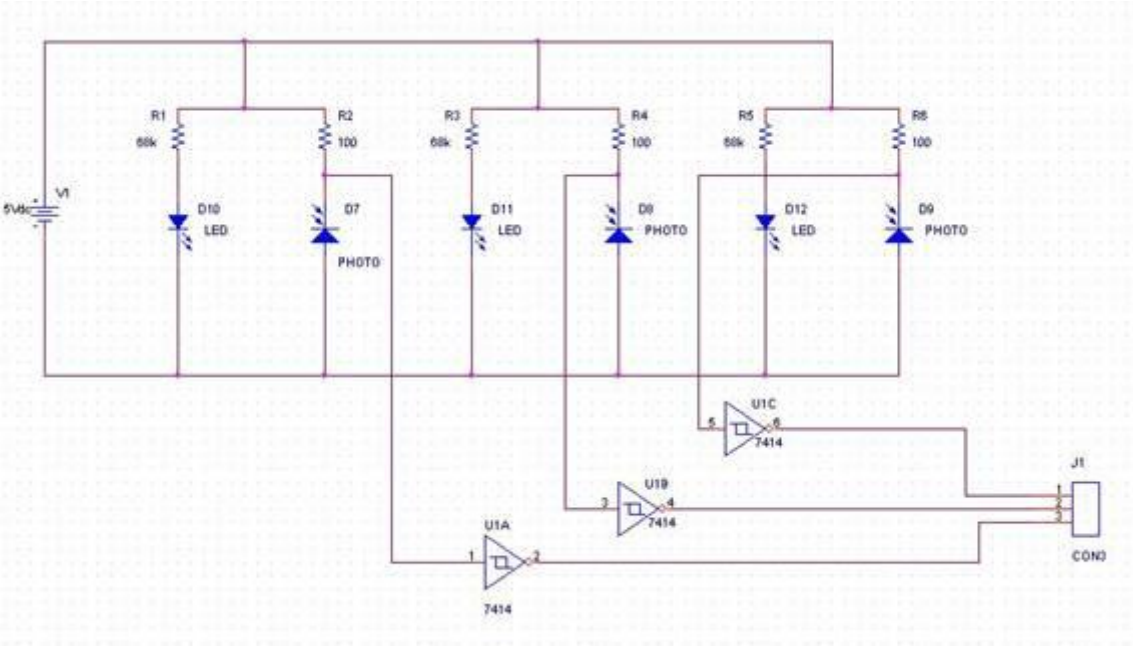

Figura 16: Placa de Sensores e alimentação para a conexão com as outras estruturas do projeto.

A figura a seguir mostra a placa desenvolvida para o uso do PIC 16F877. Esta placa foi desenvolvida por ***Valter Klein Jr***, aluno do 1º período de Engenharia Elétrica e técnico do Laboratório de Engenharia Elétrica da PUC-PR. Eis o Layout da placa:

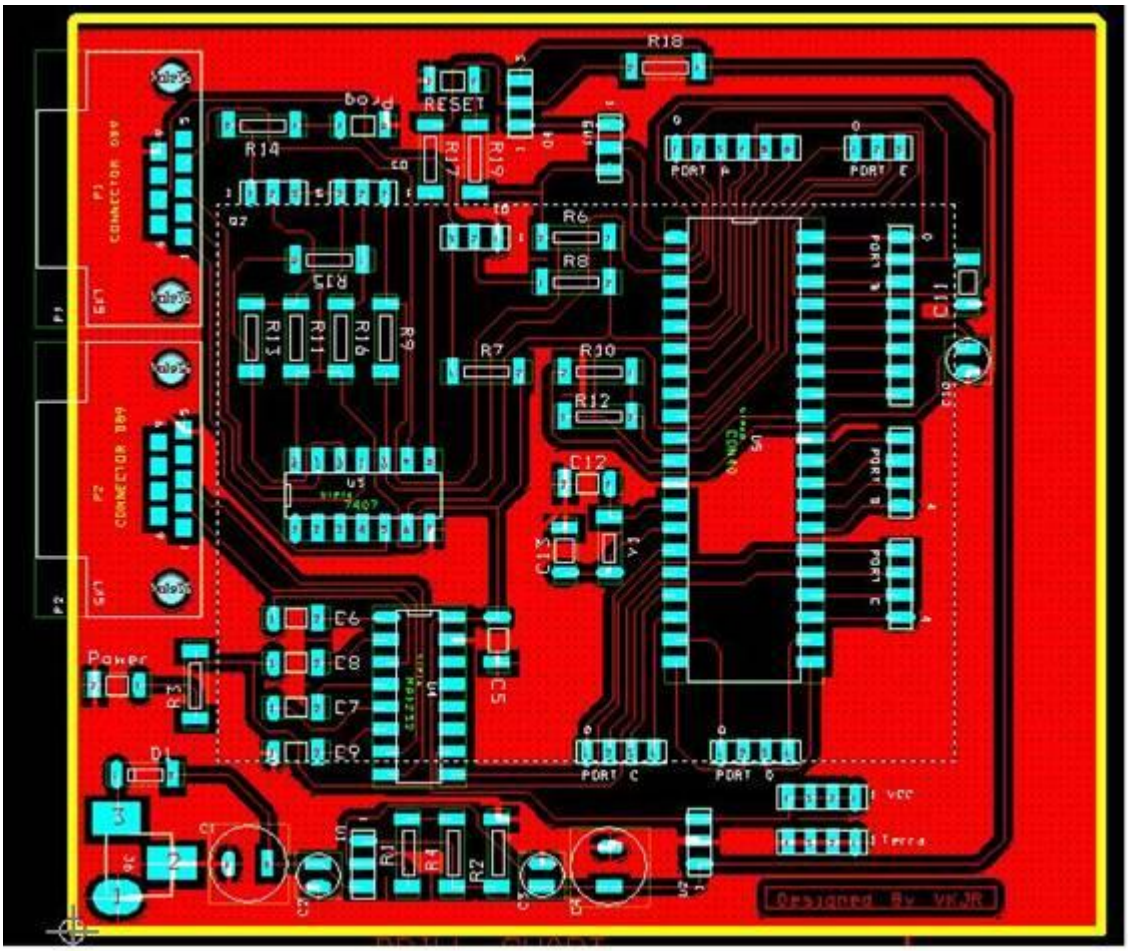

Figura 17: Layout da placa do PIC desenvolvida por Valter Klein Jr.

Para a base dos motores CC foi adquirido um carrinho simples que suportasse o peso dos motores e pudesse se locomover com a estrutura a ser instalada nele. Para alimentarmos esses motores iríamos utilizar uma placa vista no projeto similar (já mencionado acima), mas tivemos que mudar de planos, pois, devido ao peso e à capacidade dos motores que drenavam muita corrente, a placa não suportou tal estrutura. A partir daí começamos a planejar outra técnica de acionamento dos motores.

A idéia de utilizarmos relés para chavearmos os motores não havia sido cogitada, até que o Professor Altair O. Santin nos orientou a tentarmos isso. Esta foi a idéia que melhor funcionou, mas tivemos que amplificar a corrente fornecida pela placa do PIC, mudar a lógica do programa para trabalhar com a

referência invertida e implementar um sistema para deixar o carrinho mais lento usando engrenagens de impressora.

Para nos ajudar na parte mecânica do carrinho tivemos a ajuda da seguinte pessoa: ***Cid Turatti da Silva***, acadêmico do Curso de Tecnologia Mecânica em Gestão de Manufatura.

MATERIAIS E LOJAS DE COMPRA

Os materiais eletrônicos (componentes e placas de fenolite) compramos na loja Pares, na rua 24 de maio, no centro de Curitiba. Os motores e as engrenagens e algumas partes mecânicas (como rodinhas) compramos respectivamente na loja Big Alves (Vila Hauer) e uma loja de Ferragens na rua André de Barros, em frente à praça Rui Barbosa.

Eis a lista de materiais utilizados:

Componentes da placa PIC

- cap. elet. radial 47uF/50V (1)
- cap. ceramico 22pF (2) e 150pF (1)
- placa de fenolite 05x10 cm (2) e 10x15 cm (eu axo) (1)
- PIC 16F877 - 20/P (1)
- MAX 232 (1)
- LM317T - regulador de tensão positiv 1,5A/ajust (1)
- uA 7805 (1)
- BC 557 (6)
- Cristal 3,579545 MHz (1)
- RS 232 09 pinos femea 90G - DBPN1F-09 (2)
- SOQ.CI.16 pinos torneado celis sptq.cb08 (1)
- SOQ.CI.40 pinos torneado celis sptq.cb20 (1)
- SOQ.CI.14 pinos torneado celis sptq.cb07 (1)
- chave tactila 06 - 5,0 - (6x6x5,0) metaltez (1)
- RS 232 09 pinos macho - DBSM-09 (1)
- RS 232 09 pinos capa c/ trava plastica - kit curto (1)
- RS 232 25 pinos macho - 880.189-2 (1)
- RS 232 25 pinos capa c/ kit curto (trava plástica) (1)
- 4N33 (2)
- B. de pinos BPSC-40 180º metaltex (1)
- modulx40 180º (1)

Componentes da placa dos sensores

- CI 7414
- Soquete DIP 14
- 3 LED´s
- 3 TIL 78 (fotosensor)
- 3 resistores de 100 omhs x 1/8 Watt
- 3 resistores de 68K omhs x 1/8 Watt
- Conector tipo Jack 2 pinos
- Conector tipo Jack 3 pinos
- Placa de fenolite 5cm x 5cm

Componentes placa dos relés

- 2 Relés 12 VDC / 6 A
- 2 Transistores TIP 31
- 2 Resistores 10K omhs x 1/8 Watt
- 3 Conectores de pressão
- 2 Conectores tipo Jack 1 pino

Componentes dos circuitos amplificadores de corrente

- 2 Transistores TIP 31
- 2 Resistores 150 omhs x 1/8 Watt
- 2 Resistores 100 omhs x 5 Watt

INTEGRAÇÃO E DIFICULDADES ENCONTRADAS

A integração de todas as placas, o carrinho e o software se deu de forma não muito complexa e trabalhosa. As maiores dificuldades encontradas foram o gerenciamento não apenas do projeto, mas, principalmente, do tempo. Demos prioridades para algumas atividades externas e deixamos o projeto de lado, e somente às vésperas da entrega percebemos o erro e corremos contra o tempo para solucioná-lo.

Além disso, controlar a parte mecânica não foi fácil, mesmo com ajuda.

Apesar de tudo o projeto funcionou, com ônus, mas pudemos nos contentar com os resultados obtidos, até mesmo aproveitarmos a sensação de satisfação de tarefa cumprida.

A estrutura final do carrinho ficou assim:

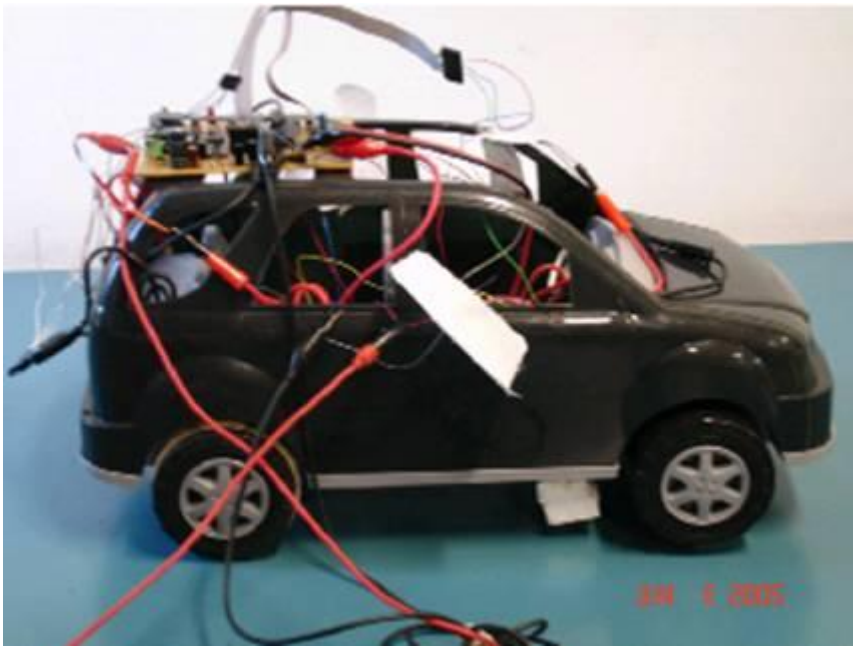

Figura 18: Visão da lateral direita do carrinho.

Figura 19: Visão da lateral esquerda do carrinho.

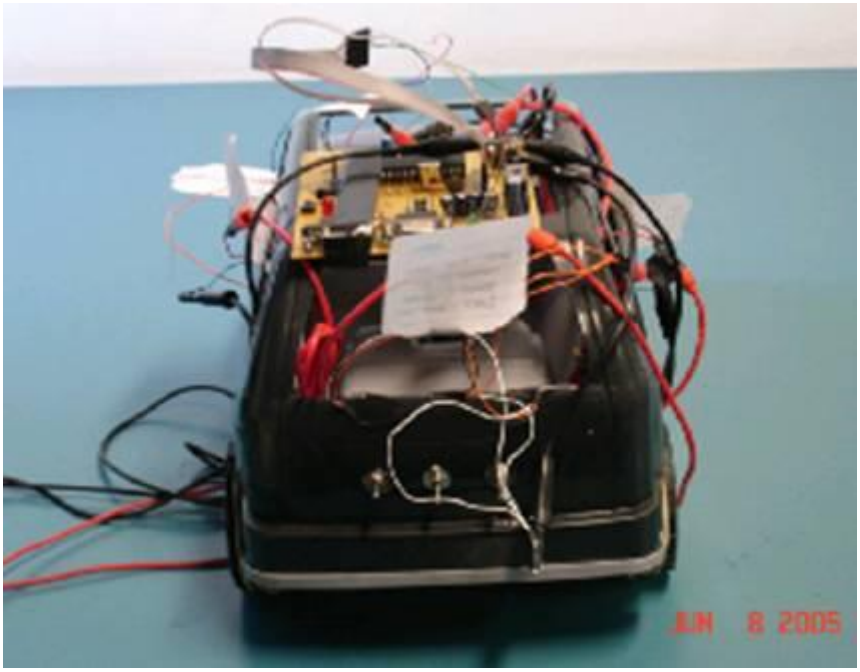

Figura 20: Visão da traseira do carrinho.

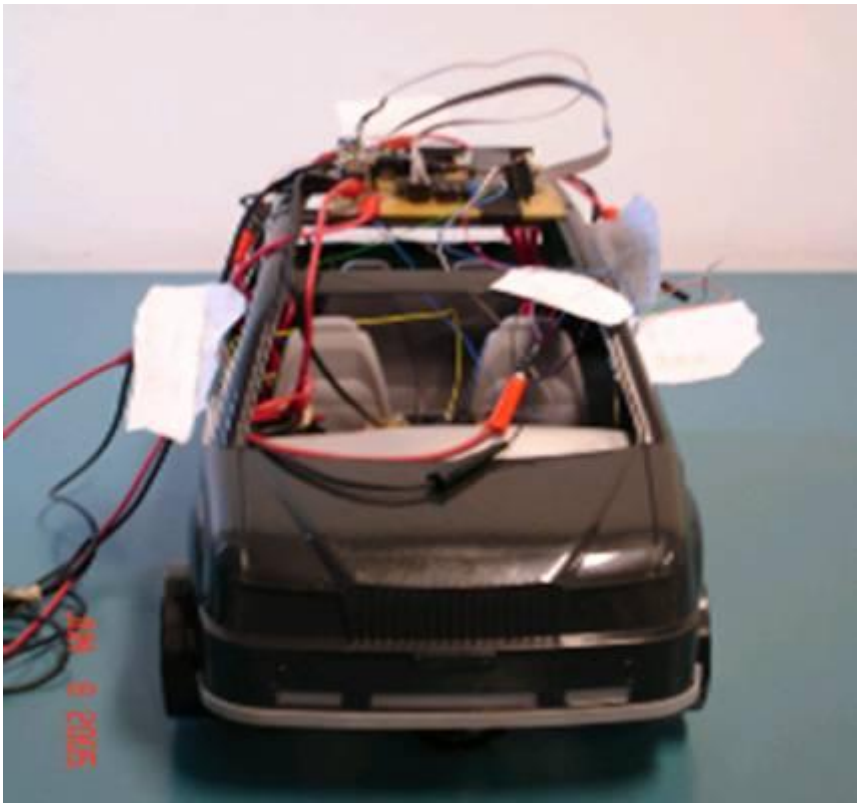

Figura 21: Visão frontal do carrinho.

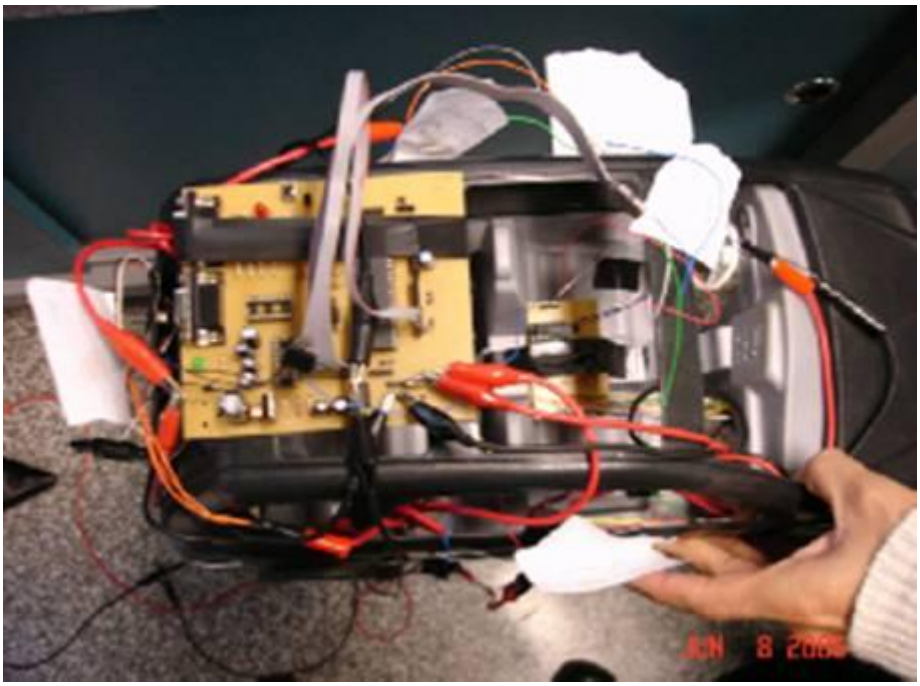

Figura 22: Visão da superior do carrinho.

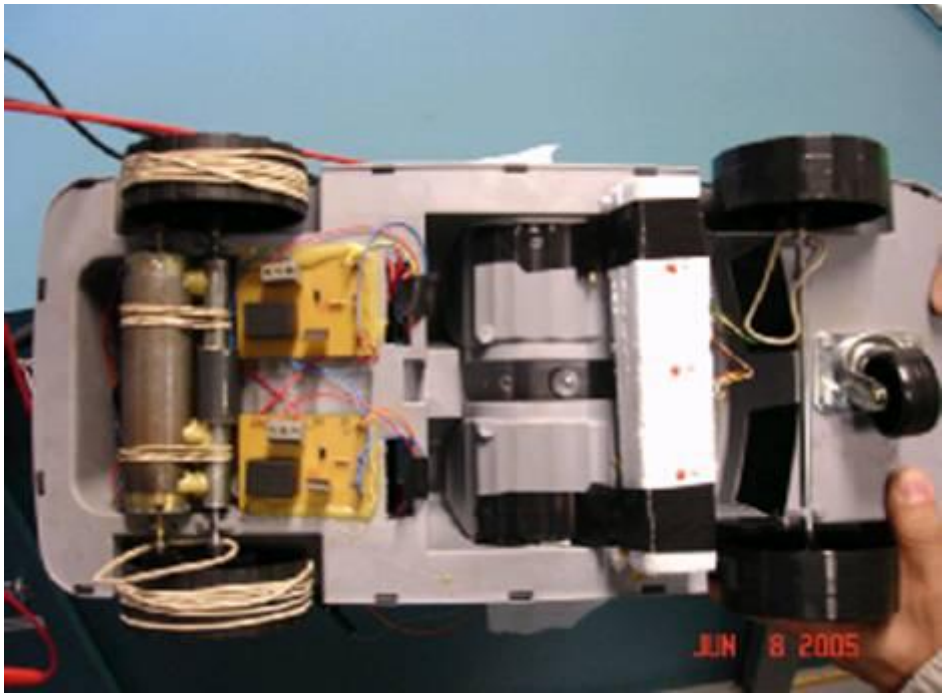

Figura 23: Visão da inferior do carrinho e dos sensores.