Universidade Estadual de Campinas

Instituto de Física Gleb Wataghin

1º semestre de 2010

Projeto:

Projeto e montagem de Robô Industrial Didático.

Relatório Final

Disciplina: F-609 – Instrumentação para Ensino

Aluno: Antonio Celso Lins de Souza
[atractorz at gmail.com]

Orientador: Eng. Pedro Miguel Raggio Santos
administrador dos Laboratórios de Ensino Básico e Informática LEB / LEI – IFGW
[praggi at ifi.unicamp.br]

1. Introdução

O uso de robôs no ensino de física é bem sólido, principalmente em países desenvolvidos. Um dos exemplos mais famosos é a empresa dinamarquesa Lego, que é um ícone da proposta de robótica pedagógica.

A Lego possui o Kit “MindStorm”, composto por um conjunto de sistemas, estruturas e interfaces que permitem uma infinidade de combinações, mas há outros kits, em geral todos eles são feitos com tecnologia específica e fechada, inviabilizando o intercambiamento de peças e ainda limitando o uso ao proposto pela empresa.

Por outro lado, há cerca de 5 anos, com o surgimento e disseminação do uso de software livre surgiu um debate e ações em torno do que se convencionar chamar de hardware livre, que propõe os conceitos das liberdades fundamentais aplicado ao hardware.

O software livre surge justamente das discussões mais atuais, como o conservacionismo, a racionalização, o domínio e a consciência sobre a tecnologia, enfim questões que com certeza estão ou estarão na pauta do dia desse século, por isso é importante estar atento a estas questões. E quando usamos esses conceitos ao ensino, parece muito promissor esse caminho, justamente na questão da robótica pedagógica que possui esse “ímpeto” mercantilizado, a oportunidade de apresentar algo barato e livre parece ainda mais interessante.

A proposta não seria construir algo livre similar ao Lego, pelo contrário, mas de criar algo que pudessem se complementar, inclusive com propostas pedagógicas diferente, enquanto o Lego tem um marketing de ser um brinquedo, mas que pode virar uma coisa séria, a proposta é apresentar a robótica nos problemas cotidianos, nas reais situações. Tudo indica que essa é uma forma mais correta de mostrar a importância e o lugar da robótica nos dias atuais.

Minha proposta com esse relatório final é mostrar para outros estudantes um pouco o caminho que tomei, e servir como uma base segura para propostas mais ousadas, por isso as vezes pode ter um certo tom de manual, principalmente por que falarei de cada ponto do ponto de vista prático e teórico.

1.1. Robótica Livre

Um robô é composto em várias partes: O software, o hardware e a interface entre eles, a mecânica e nesta proposta é que todas sejam usadas/baseadas em tecnologias de forma livre e reaproveitada.

Um critério fundamental nestas construções é o uso de sucata eletrônica, principalmente computadores e periféricos, podemos encontrar peças de máquina fina em diversos hardware, principalmente no hard disk, CD/DVD-Rom, impressoras e scanners. Além disso é comum o uso de computadores velhos e obsoletos como parte do controle. Enfim a proposta é justamente usar o mínimo e da forma mais eficiente possível.

Um ponto crítico em projetos deste tipo é a interface, um projeto interessante é o arduíno, um controlador baseado em microcontroladores Atmel, mas com software livre de controle livre.

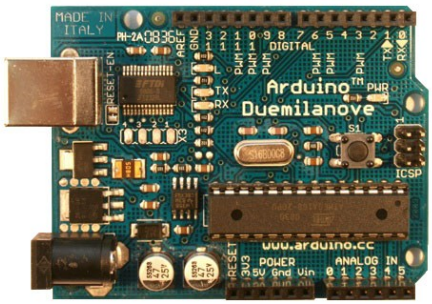

Fig 1- Arduíno Duemilanove, opera usando a porta USB, extremamente fácil e robusto de usar, todo sistema dele é baseado em software livre, disponível para vária linguagens de programação.

Mas prefiro tratar a robótica livre por outro aspecto. Existem diversas propostas, que acredito serem mais inspiradoras, como o osciloscópio feito por placa de som, controle de passo usando um mouse, e até mesmo CNC usando impressoras velhas, enfim tendo como limite a criatividade e disposição de estudar e montar circuitos e sistemas.

É inacreditável, mas cerca de 90% do projeto foi feito com peças encontrado nas lixeiras da minha vizinhança!

2. O projeto

O robô/sistema proposto consiste em um sistema com dois eixos de liberdade (X,Y) , levando uma pequena carga entre pontos.

Usando um scanner (HP), um driver de CD-Rom, transistores e resistores, um computador obsoleto (também encontrado no lixo...) e por uma fonte PC-ATX, foi possível fazer este robô, realmente um dos desafios era fazer ele de forma barata e fácil (pelo menos o entendimento).

2.1. Mecânica

Usando a caixa do scanner como suporte do todo o sistema, o trilho do canhão de leitura, dessa forma acontece o movimento no eixo X. Usando a gaveta do CD-Rom, os motores acoplado a esse, com exceção do motor de leitura do disco (motor de passo), foi possível fazer o movimento associado à Y, algumas fotos da carcaça apenas com o sistema mecânico:

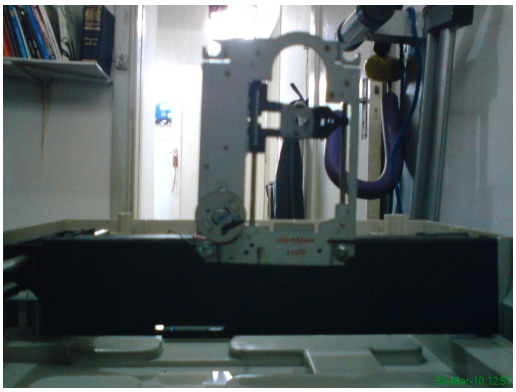

fig 2 – vista frontal, a gaveta de CD-ROM (eixo Y)

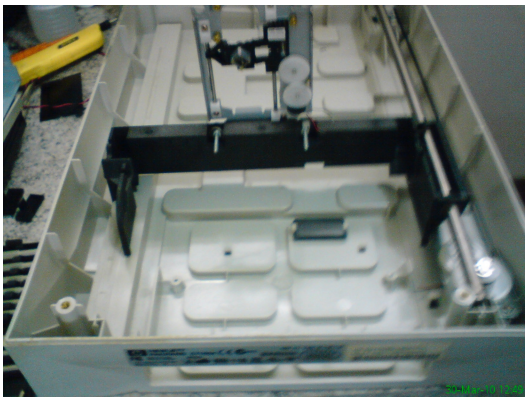

fig 3 - A caixa e o trilho (eixo X)

Durante a desmontagem do hardware foram encontrados alguns componentes que poderiam ser utilizado, alguns de forma fácil e óbvia, como dois sensores de contato encontrado no CD-ROM, utilizado como sensores de movimento em Y.

Entretanto alguns componentes tiverem o uso impossibilitado, a questão são os componentes integrados, por exemplo tentou-se usar o sensor óptico de mouse, integrado a um CI, infelizmente era praticamente impossível usar esse sistema.

2.2. Sensores e interfaces

Tanto os sensores dos mouse óptico quanto mecânico praticamente tem o uso inviável se não consideramos os seus respectivos controle digital.

Não há uma padronização do Mouse, mas tudo indica que o modelo de hardware que temos se origina em um conjunto de norma apresentado pela Microsoft no começo do anos 90. Apesar de não ter usado no projeto, creio que possa contribuir mostrando as questões que me levaram a rejeitar o uso dos sensores do mouse.

a) O mouse serial (Microsoft)

Todos os mouse atuais são derivados do modelos compatível da microsoft, em geral o esquema mecânico deles é este:

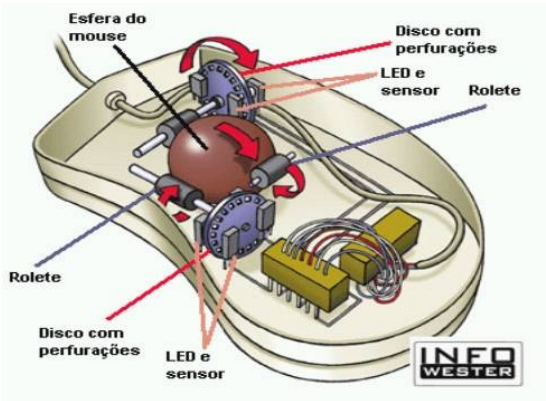

fig 4 – Esquema mecânico do mouse compatível microsoft.

Em 1995 a National Semiconductor publicou um guia de uso do microcontrolador COP800 em dispositivo Mouse, usando a porta serial do computador RS-232.

O sinal obtido no sensor realmente é uma onda quadrada:

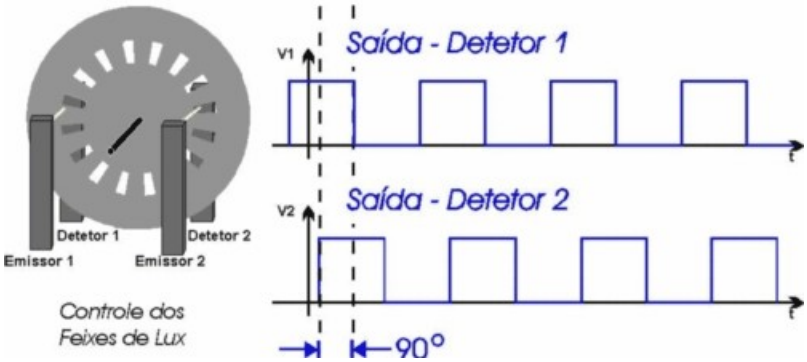

fig 5 – esquema do sensor de movimento o mouse.

Dentro da interface RS- 232, o mouse usa as linhas RTS, DTR, TXD e RXD, sendo que as linha TRS e DTR são usadas para fornecer a energia de funcionamento, ficando as linha TXD e RXD responsáveis pela comunicação.

A arquitetura de funcionamento do RS-232 é um tanto quanto complicada, ele não opera com dados binários, pelo contrário o CI microcontrolador entrega os dados acumulado, isto é, ele passa efetivamente a posição do mouse, o ganho é a possibilidade de leitura muito mais rápida e distribuição da tarefa do processador central.

O Artigo *PC-MOUSE Implementation Using COP800, AN-681, 1995,* além do site *http://www.ustr.net/8051pc/starting.shtml* podem ser fontes valiosas de informação.

Portanto seria muito difícil usar o sensor através do microcontrolador, mas seria mecanicamente fácil, uma saída para o uso do Mouse é usar os dados obtidos pelo BUS, e entregue ao sistema, ou seja ele ligado na porta serial (ou USB) obtendo os dados, e tratando esses dados em nível de sistema.

b) A porta paralela

Pela simplicidade acabou-se adotando a Porta Paralela, haveria a possibilidade ainda de se usar a entrada/saída analógica (usada por joystick), mas a literatura é mais farta além da porta paralela trabalhar por padrão com sistemas binários.

A porta paralela foi criada pela IBM, e foi normatizada pela IEEE 1284 em 1994, originalmente foi proposta como uma porta para impressoras.

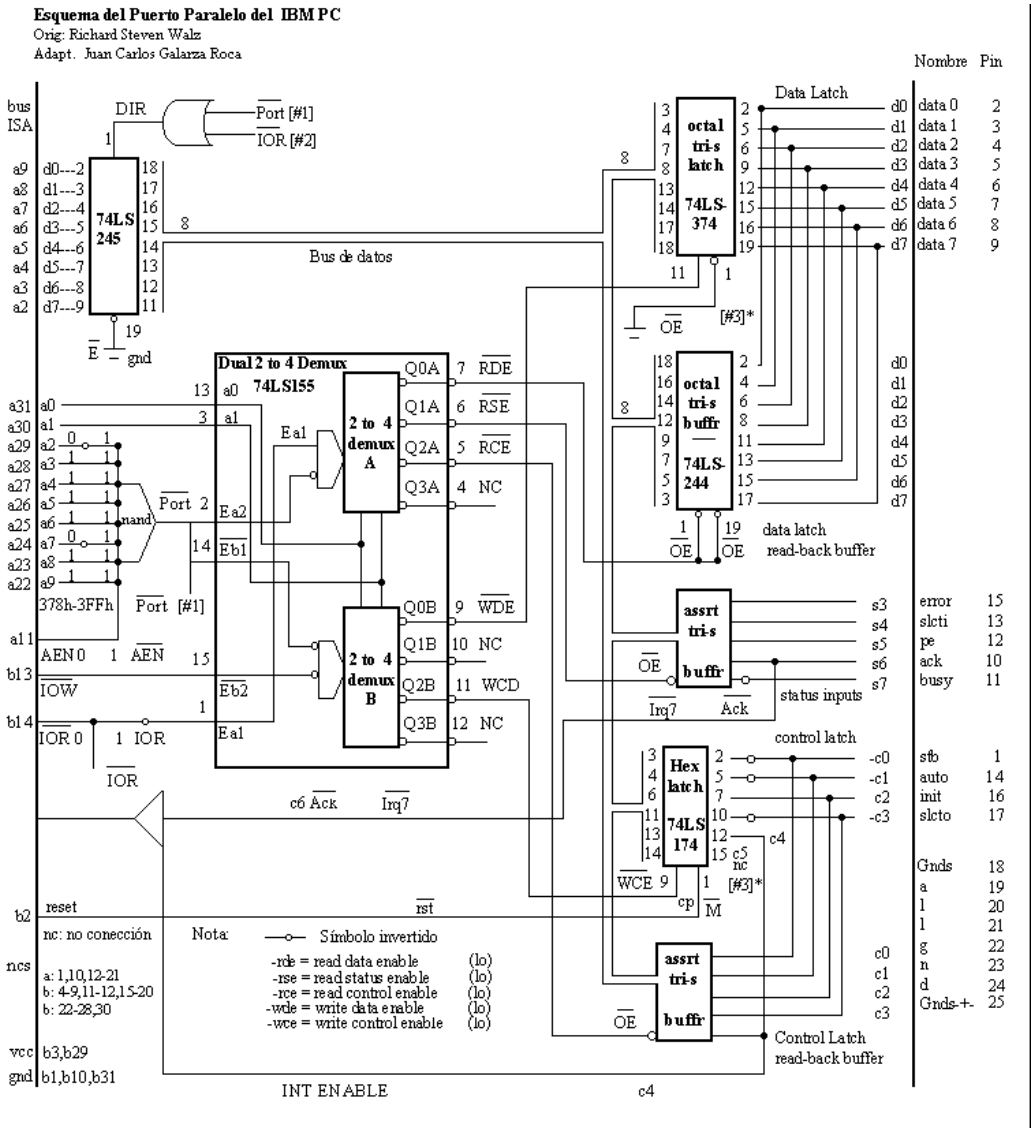

fig 6 – Circuito da porta paralela, baseado no CI 74LS373

O conceito de funcionamento da Porta Paralela é diferente da porta serial, os binários chegam até o circuito onde é feito o tratamento desses dados, pela normatização há 5 tipos de comunicação dos dados:

- Apenas sentido direto – Modo de compatibilidade com o sistema Centronics;
- Modo direção inversa (Modo Nibble) - 4 bits por vez usando as linhas de estado (recebendo dados)(Hewlett Packard Bi-tronics);
- Modo octal (Byte Mode) - 8 bits por vez usando as linhas de dados de datos (porta bidireccional)
- Modo Bidireccional EPP (Enhanced Parallel Port) - Usado principalmente para periféricos que não são impresoras, como CD-ROM, Adaptadores de Rede, etc.
- Modo Bidireccional ECP (Extended Capability Port) - usado principalmente por impressoras modernas e scanners.

Usamos atualmente o modo Bidireccional e suas variantes, isso implica em uma comunicação de 8-bits, a velocidade pode variar entre 50 e 100KB/s.
A conexão pode ser feita por dois tipos de encaixe, o DB-25 e o Centronics (herdado):

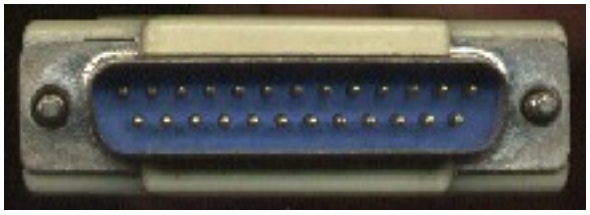

Fig 8 – Conexão Centronics

Com as seguintes disposições de linhas:

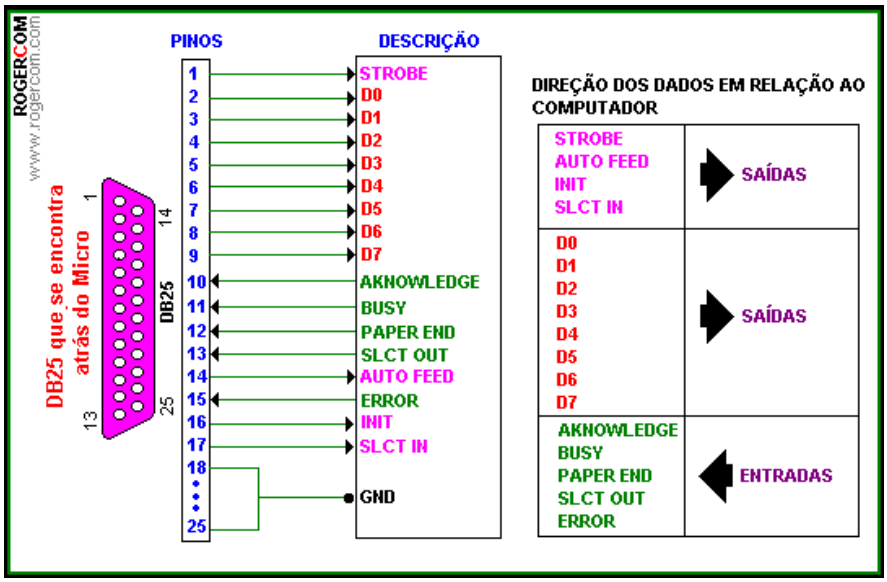

fig 8 – As entradas e saídas da porta paralela

É importante notar que a saída é de 8-bit (8 estados lógicos), no entanto a entrada é composta por 5-bit, isso acontece por uma questão de estrutura no padrão de comunicação, na realidade existem 3 grupos de linhas, o controle (rosa), Status (verde) e dados (Vermelho), para enviar dados para porta paralela usa-se o controle e as saídas, para receber usa-se o status, o BUS do sistema precisa encontrar esses endereços, portanto:

Nome	Endereços LPT1	Endereços LPT2	Descrição
Registro de Dados	378h	278h	Envia um byte para a impressora
Registro de Status	379h	279h	Ler o Status da impressora
Registro de Controle	37Ah	27Ah	Envia dados de controle para a impressora

Tab 1- Endereços dos registros

Assim quando queremos enviar algum dado para a impressora usamos o endereço 0x378 ou 0x278 (notação em C), ou também podemos enviar pelos endereços 0x37A ou 0x27A.
A leitura de variáveis pode ser feita pelos Status, para isso usamos os endereços 0x379 ou x279. Há uma diferença importante entre a leitura e a escrita, podemos escolher uma porta especifica de escrita, mas não conseguimos especificar uma de leitura, precisamos obter o byte completo.

Uma questão importante é que o computador usa como endereçamento a base hexadecimal, e usaremos como referencia dos endereços a base decimal, que é na realidade binária.

Decimal	Binário	Octal	Hexadecimal
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Tab 2 – relação e conversão entre as bases

3. Programação

Optamos pelo C, pela sua simplicidade e portabilidade, além disso é extreamente fácil acessar e enviar dados pela porta paralela. Usou o C padrão ANSI, compatível com o Gnu C Compiler (Padrão do Linux).
O acesso a Porta Paralela em C é extremamente fácil, destacamos aqui as principais funções:

```
#include <sys/io.h> // biblioteca padrão de acesso

ioperm(0x378,3,1);  /*inicializa os registradores dados, o parâmetro 3 é a reserva de endereço, caso o sistema opere em outra porta, e 1 é o estado ligado*/
outb(0x00, 0x378); // escreve na porta, neste caso zera a porta paralela

ioperm(0x379,3,1); // Idem mas reservando agora os registradores de Status
inb(0x379) // faz a leitura da porta
```

O restante do programa apenas opera entre essas funções, de tal forma a permitir que o operador pegue uma moeda em um dos três pontos de parada e determine até o outro ponto.

4. Compilador e programas

Uma parte fundamental do problema foi achar uma distribuição do GNU/Linux que mais se adpta-se as minhas necessidades, uma máquina obsoleta e que permitisse a execução do GCC.

O GCC é um dos mais importante programa desenvolvido pela Free-Software Foundation, é um poderoso e robusto compilador de linguagem C/C++, extremamente otimizado.

Rodei o GCC sobre um computador com processador pentium II 200MHz, com 128 MB de Ram, com quase nenhum problema, testei algumas distribuições:

4.1. Puppy Linux

Distribuição muito rápida, totalmente otimizada, o tamanho total é de cerca de 50MB, usa como interface gráfica o XFCE . Pode ser montada facilmente em qualquer pen-drive, permitindo diversas possibilidades.

Montada sobre uma distribuição própria e com software compilados exclusivamente, inclusive com um sistema próprio de pacotes.

Por padrão não vem com o GCC, é preciso recompilar o kernel para permitir o uso do GCC. Estranhamente ele tem um funcionamento parecido com o FreeBSD

4.2. Slax

Distribuição derivada do Slackware usando o KDE como interface gráfica, funcionou perfeitamente na máquina, vem por padrão com o VIM (edição do código) e com o GCC, compilou sem qualquer problema todo o código.

4.3. Outros programas

Apenas na modelagem do circuito que não foi possível realizar em software livre, precisou-se usar o Circuit Maker em um ambiente Windows. Há um Software desenvolvido (http://bwrc.eecs.berkeley.edu/classes/icbook/spice/) na Universidade de Berkeley, mas não foi possível a instalação e uso.

5. Eletrônica

A eletrônica envolvida no robô consiste em duas partes: Amplificação e tratamento do sinal de saída e sensores.

O componente básico usado foi o transistor, usando em uma aplicação muito clara e importante como uma chave.

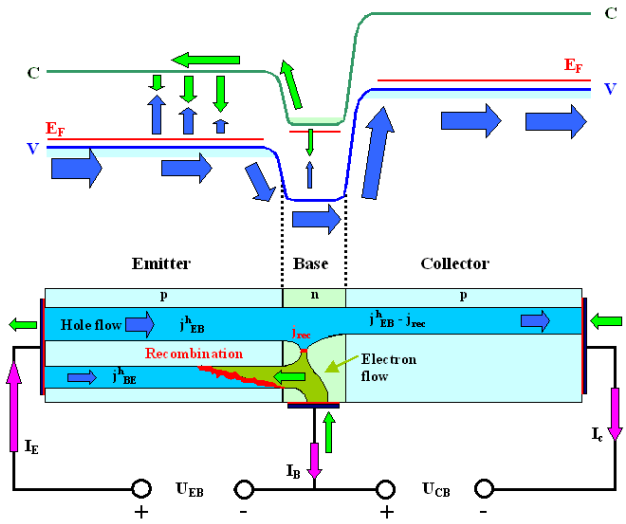

Fig 9 – Um transistor

O funcionamento do transistor é simples, apesar de ocorrerem vários efeitos ao mesmo tempo, uma das razões na dificuldade em projetar um circuito.

5.1. Ponte H

A ponte H é um circuito extremamente simples e muito útil, é uma forma de se inverter o giro em motores de corrente continua, invertendo a polaridade entre seus pólos.
Há muita literatura sobre a questão deste circuito, no entanto no sistema considerado houveram diversos fatores limitantes, tais como: Baixa intensidade do sinal, nível baixo na porta entre 0V e +0,4V, relativa alta corrente no motor, necessidade de simplicidade e baixo custo.
Em vários circuitos se apresenta um transistor pré-amplificador, totalizando 6 transistores, um problema em usar esses tipos de transistores é a alta corrente circulando entre o coletor e o emissor, além da instabilidade gerada.
Propomos o seguinte circuito:

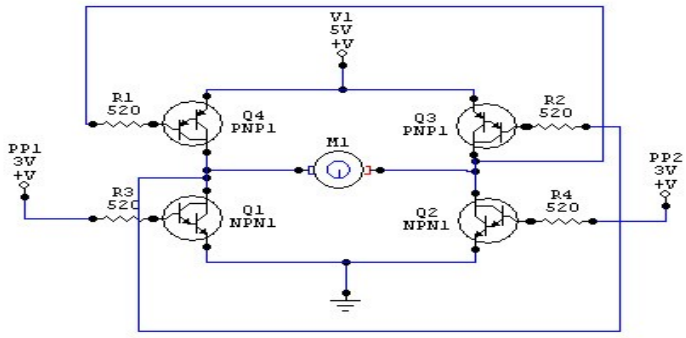

fig 10 – Ponte H

O circuito foi constituído por transistores Darlightons de alto ganho, além de serem muito robusto por serem de potência, além da estabilidade e facilidade na compreensão.

6. Conclusão

A proposta inicial era que algo desse tipo pudesse ser montado em uma escola durante um ano inteiro, adequando um aspecto ou outro seria plenamente possível.

Sem dúvida que a quantidade de conceitos envolvido é muito grande, isso permitiria que os alunos reelaboração a todo instante o conhecimento, além de criar hábitos de conservacionismo e

otimizacão.

É plenamente possível estabelecer a construção de artefatos usando sucata e baseando-se em softwares livres, pode ser um pouco mais difícil, mas sem dúvida o caminho é muito mais belo e instigante com muitas possibilidades.

7. Bibliografia

C: A linguagem de programação, Kernighan B.W., Ritchie D. M., 4˚ed, 1988, Ed Edisa;

Circuitos com diodos e transistoresm, Markus O, 2000, Ed Érica;

Portas Paralelas, Ferrari F., Universidade Federal do Pampa, Bagé, 2008.

Sítios da internet:

http://www.vivaolinux.com.br/artigo/Acessando-a-porta-paralela-via-Linux

http://tldp.org/HOWTO/IO-Port-Programming.html

http://books.google.com.br/books?id=7LNfl28rm3IC&printsec=frontcover&dq=Linguagem+C&source=bl&ots=3_jjQpc-F9&sig=48D6Wo-f7TAo93g-QAYj6mgms6U&hl=pt-BR&ei=vTkSTIeoLcGblgft363dBw&sa=X&oi=book_result&ct=result&resnum=10&ved=0CEMQ6AEwCQ#v=onepage&q&f=false

http://www.vivaolinux.com.br/script/Acendedor-de-leds-pela-porta-paralela?

http://www.clubedohardware.com.br/artigos/1147

http://www.arduino.cc/

http://www.infowester.com/mouse.php

http://edsonmelo.trix.net/professor/estudos/mouse/index.html

http://www.ustr.net/8051pc/8051pc.shtml

http://www.thinklabs.in/resources/?paged=5

http://www.epanorama.net/circuits/parallel_output.html

8. Apêndice:

8.1. O código de controle

```
#include <stdio.h>
#include <sys/io.h>
#include <stdlib.h>

//sa√≠das
#define EST_CW 2 //00000010
#define EST_CCW 1 //00000001
#define GUI_CIMA 4 //0000100
#define GUI_BAIXO 8 //0001000
#define IMA 16 // 00010000

//entradas, barrada
#define GUI_S 67 // 0111111  o sensor de cima esta em paralelo com o sensor de baixo, assim o mesmo sinal ser√° enviado no max cima e baixo
#define POS_1 95 // 1011111
#define POS_2 223 // 1001111
#define POS_3 255 //1101111




int main(void)
{
int opcao,op; // controle de la√ßos
int origem,dest,pos_atual; //ponteiros para passagem de valores entre fun√ßoes

ioperm(0x378,3,1); //inicializa a porta paralela
ioperm(0x379,3,1);
outb(0x00, 0x378);

        op=1;
        origem=1;
        dest=3; //inicializa√ß√£o de variaveis
        pos_atual=1; // o trem tem come√ßar na posi√ß√£o 1

do {

        printf ("\n=====Menu principal=====\n");
        printf (" 1.Executar movimento\n");
        printf (" 2.Sair\n\n");
        printf (" Selecione uma opcao: ");
        scanf ("%d", &opcao);

if (opcao == 1)
{
        do
        {
                printf ("\n====Entre com o ponto de origem (1,2 ou3)=====\n");
                scanf ("%d", &origem); //carrega decimal
```

```
	} while (origem == dest || origem >= 4); // mantem a solicita√ß√£o enquanto as condi√ßoes nao forem satisfeitas

	do
	{
		printf ("\n====Entre com o ponto de destino (1,2 ou3)=====\n");
		scanf ("%d", &dest); //carrega decimal
	} while( dest == origem || dest >= 4); // mantem a solicita√ß√£o enquanto as condi√ßoes nao forem satisfeitas

	if (posi(0) == 0)
			{
				printf("reinicializando o programa ... coloque o trem na posi√ß√£o 1 \n");
				return(0);
			}

	mover(origem, dest);
	sleep(1); // espera 1 segundo antes de pegar a pe√ßa
	guind(1); // opera o guindaste abaixa, pega , levanta (op√ß√£o 1)
	sleep(1);
	mover(dest,origem);
	guind(0);// abaixa, solta, levanta

	outb(0x00, 0x378); // sa√≠da paralela em n√≠vel baixo, evitar superaquecimento

if (opcao == 2){
	printf("\nTchau!\n");
	}

};

return (0);
}while(opcao != 2);
}

int mover(int o,int d){


	while(posi(0) != o){ // permite buscar o ponto de origem com o im√£ desligado
		if (posi(0)>=o){
			outb(EST_CCW + GUI_CIMA, 0x378);
		}

		if (posi(0)<=o){
			outb(EST_CW + GUI_CIMA, 0x378);
		}
	}
		while(o!=d){


			if (posi(0)>=o){
```

```
                                outb(EST_CCW + IMA, 0x378);
                                while(posi(1)==0){
                                        if (inb(0x379) == GUI_S){
                                                outb(EST_CCW + GUI_CIMA+IMA, 0x378); //
permite que continue em movimento e
                                }
                        }
                if (posi(0)<=o){
                        outb(EST_CW + IMA, 0x378);
                        if (inb(0x379) == GUI_S){
                                        outb(EST_CW + GUI_CIMA + IMA, 0x378);
                        }
                        }
                if (posi(1) != 0){
                        outb(IMA, 0x378); // desliga o motor da esteira, mas mantem o im√£
                        sleep(1);
                        }
                }

        }
return(0);
}



int guind(int modo){ //modo 1 - Pega , modo 2 - solta

        if (modo == 1){
                        outb(GUI_BAIXO + IMA, 0x378);
                do{// faz um la√ßo pra verificar o sensor do guindaste
                        if(inb(0x379) == GUI_S){
                                outb(GUI_CIMA + IMA, 0x378);
                        }
                }while(inb(0x379) != GUI_S);
        }

        if (modo == 2){ // Abaixa e desliga o ima
                        outb(GUI_BAIXO + IMA, 0x378);
                do{// faz um la√ßo pra verificar o sensor do guindaste
                        if (inb(0x379) == GUI_S){

                                outb(GUI_CIMA, 0x378);
                        }
                }while (inb(0x379) != GUI_S);
        }



return(0);
}
```

```
int posi(int modo){ // l√™ a posi√ß√£o atual retorna 1, 2 ou 3 , modo 1 - entra em loop, sai apenas se estiver em alguma posi√ß√£o, modo 0 - retorna a posi√ß√£o ou se esta fora de posi√ß√£o

	if (modo == 0){
		if(inb(0x379) == 127){
			return(0);
			}
		}
	do {

		if (inb(0x379) == POS_1) {
			sleep(0.5);//para confirmar que estamos sobre a posi√ß√£o 1

			if (inb(0x379) == POS_1){
			return (1);
			}
		}

		if (inb(0x379) == POS_2){
			return(2);
		}

		if (inb(0x379) == POS_3) {
			sleep(0.2);//para confirmar que estamos sobre a posi√ß√£o 1

			if (inb(0x379) == POS_3){
			return (3);
			}
		}
	}while (inb(0x379)==127);
return(0);

}
```

O meu orientador realizou os seguintes comentários:

O Eng. Pedro, solicitou que em uma nova versão ele colocará os comentários