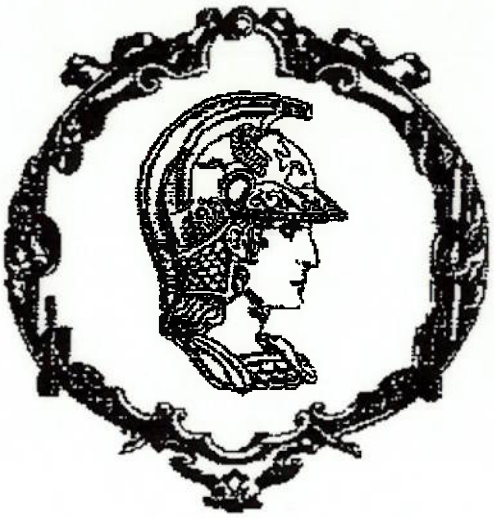
Universidade de São Paulo
Escola Politécnica



PMC 580/581
Projeto Mecânico I & II

Comando Numérico para eixo de posicionamento de robô

Coordenador da Disciplina: Jun Okamoto / Marcelo Godoy
Professor Orientador: Marcos Ribeiro Pereira Barreto / Lucas Moscato

nome:	nºUSP:
Jerri Regis Biscuola	2842884
Vinícius Arouche de Toledo	2809148

São Paulo, Julho de 1996

ÍNDICE

1. MOTIVAÇÃO **3**

1.1 EMBASAMENTO TEÓRICO 4

2. DESCRIÇÃO DO PROBLEMA **8**

3. DESENVOLVIMENTO REALIZADO **13**

3.1 DEFINIÇÃO DAS FUNÇÕES DE PROGRAMAÇÃO 13
3.1.1 Funções Preparatórias *13*
3.1.2 Função Número de Seqüência *14*
3.1.3 Função Auxiliar de Avanço *14*
3.1.4 Funções Miscelânea *14*
3.1.5 Função de posicionamento *15*
3.1.6 Sintaxe da linguagem G *15*
3.2 INTERFACE HOMEM-MÁQUINA 16
3.2.1 Tela inicial *16*
3.2.2 Tela de principal *17*
3.2.3 Tela de Selecionar Arquivos *19*
3.3 DEFINIÇÃO DAS RAMPAS DE ACELERAÇÃO E DESACELERAÇÃO 20
3.4 ESTRUTURA DE DADOS DO PROGRAMA DE CONTROLE 22
3.5 PARAMETRIZAÇÃO DA PLACA CONTROLADORA DO SERVO-AMPLIFICADOR 23
3.6 INTERFACE D.N.C. 27
3.7 ARQUITETURA DO HARDWARE 28
3.7.1 Placa Contadora *29*
3.7.2 Chaves óticas *30*
3.7.3 Conector placa contadora - entrada digital (CN5) *31*
3.7.4 Conector placa contadora - encoder *33*
3.7.5 Conector final de curso - entradas digitais (CN3) *34*
3.8 ARQUITETURA DO SOFTWARE 34
3.8.1 Compilador *34*
3.8.2 Interpretador *35*
3.8.3 Loop PID *36*

4. ATIVIDADES REALIZADAS **39**

5 - RESULTADOS **41**

6. BIBLIOGRAFIA **42**

7. ANEXOS **43**

7.1 ESQUEMA DA PLACA DE ACIONAMENTO DO SERVO-MOTOR 43
7.2 ÁRVORE DE DIRETÓRIOS CRIADA PELO ARQUIVO COMPACTADO 43
7.3 LISTAGENS 45

1. Motivação

A manufatura celular é uma aplicação do conceito de Tecnologia de Grupo, onde uma parte do sistema produtivo de uma empresa é convertido em células. Uma célula de manufatura é um conjunto de máquinas ou processos diferentes localizados próximos entre si e dedicados à manufatura de uma família de peças. Estas peças devem ser semelhantes em suas necessidades de processo (operações necessárias, tolerâncias, capacidade de máquina exigida, etc.).

O alvo da manufatura celular é a redução de tempo de ajuste de máquina - tempo de "set-up" - e do tempo de fluxo de peças. Com isso acaba por reduzir estoques e também o tempo de resposta da empresa às necessidades de mercado. [WEMMERLÖV, 1989]

A célula de manufatura, existente no Departamento de Engenharia Mecânica, é composta por um robô ASEA com 6 graus de liberdade, um torno CNC (Comando Numérico Computadorizado), uma fresadora CNC e um trilho (que corresponde ao sétimo grau de liberdade do robô).

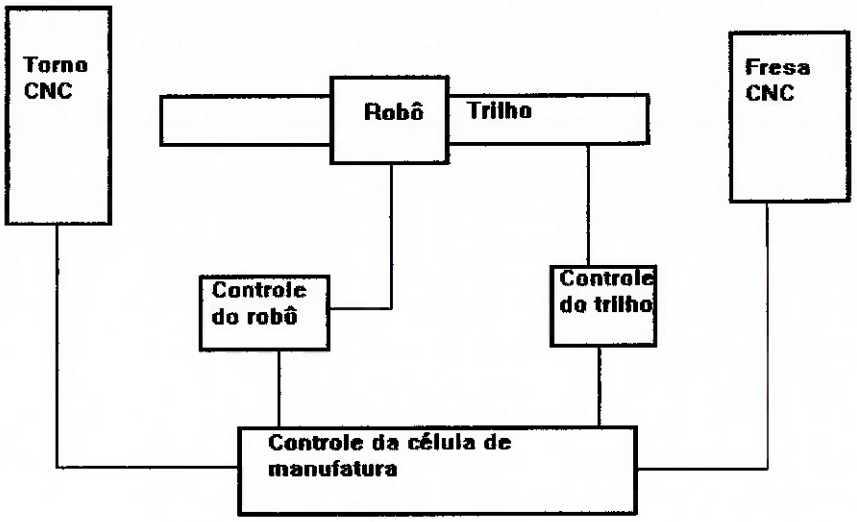


figura 1. Estrutura da célula de manufatura

Este trabalho tem por objetivo implementar o controle de posição do trilho para a movimentação do robô.

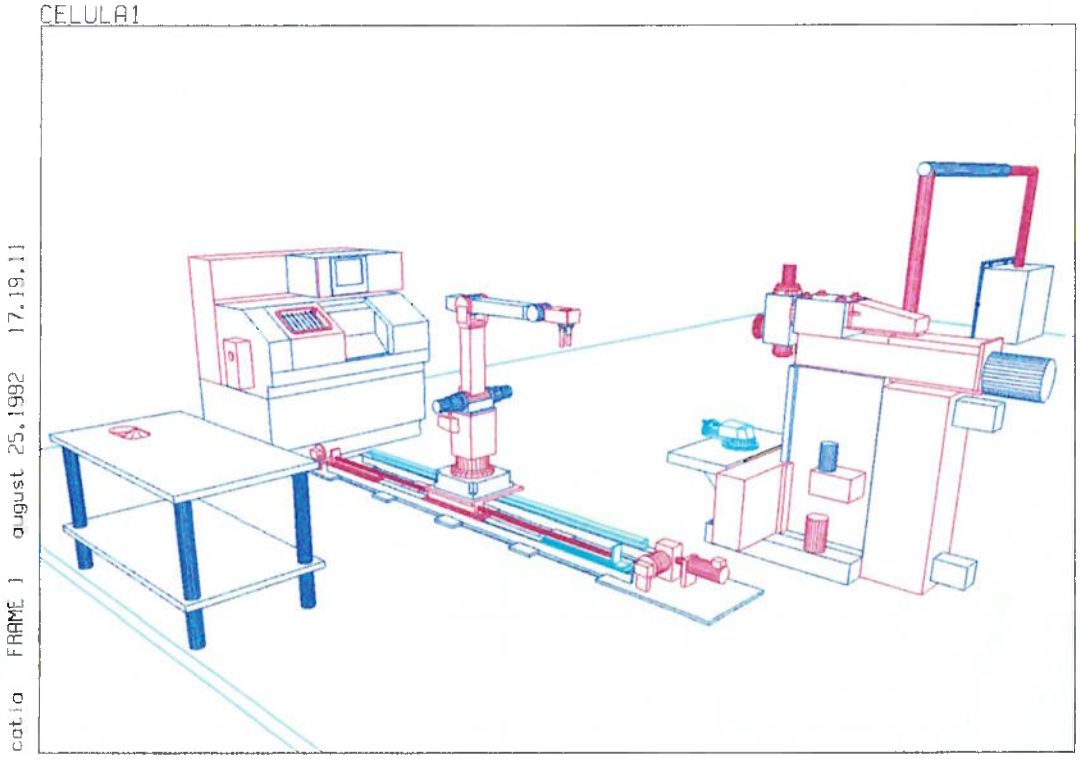


figura 2: Vista da célula de manufatura

1.1 Embasamento Teórico

Sistemas Integrados de Fabricação

Consistem de duas ou mais máquinas com Comando Numérico interligadas entre si com trocadores e transportadores automáticos de peça, máquinas estas que perfazem operações conseqüentes ou simultâneas.

A célula de fabricação normalmente é supervisionada por uma central de processamento de dados que coordena todos os CNC envolvidos, sistema de troca de peças, de programas e outros eventos. Permitem grande flexibilidade em relação a tipos e quantidades e permite um fluxo de produção com a mínima intervenção humana. [MACHADO, 1990]

Comando Numérico Distribuído (DNC)

Consiste basicamente em um conjunto de máquinas equipadas com CNC, controladas ou conectadas por uma unidade central de computador. No caso da célula utilizada neste projeto, têm-se três máquinas CNC controladas por uma unidade central. Estas máquinas são o torno, a fresa e o trilho, que pode ser considerado uma máquina CNC com um único eixo.

Dependendo do nível da dependência entre as máquinas e o sistema de processamento de dados, podem existir inúmeras soluções envolvendo equipamentos, programas dedicados e específicos de gerenciamento e meios de comunicação próprios.

A aplicação mais simples do DNC, consiste na utilização de um microcomputador cuja principal finalidade é ser o meio de edição dos programas bem como o meio de armazenamento desses programas. Esse computador é conectado às diversas máquinas por meio de um sistema de comunicação para uso industrial e, portanto, imune aos ruídos desta transmissão.

À medida que aumenta a quantidade de máquinas a serem interligadas à rede DNC, ou é aumentado o número de variáveis e eventos que o computador deve monitorar, os sistemas tendem a exigir computadores de maior porte e programas mais complexos e abrangentes.

Uma das grandes vantagens do Comando Numérico é a capacidade que este equipamento deu às máquinas de poderem ser parte integrante de um sistema de processamento e controle de dados, isto é, de formar o Comando Numérico Distribuído. [MACHADO, 1990]

Controlador PID

Os controladores PID são muito freqüentemente usados em sistemas de controle industriais. A função de transferência Gc(s) do controlador PID é:

$$Gc(s) = kp.(1 + \frac{1}{Ti.s} + Td.s)$$

onde kp = ganho proporcional

Ti = tempo integral

Td = tempo derivativo

Se e(t) for a entrada do controlador PID, a saída u(t) do controlador é dada por:

$$u(t) = kp.\left[e(t) + \frac{1}{Ti} \int_{-\infty}^{t} e(t).dt + Td.\frac{de(t)}{dt} \right]$$

As constantes kp, Ti e Td são os parâmetros do controlador. A equação de Gc(s) pode também ser escrita como:

$$Gc(s) = kp + \frac{ki}{s} + kd.s$$

onde kp = ganho proporcional

ki = ganho integral

kd = ganho derivativo

Neste caso kp, ki e kd tornam-se parâmetros do controlador.

Se um modelo matemático da planta puder ser deduzido, então é possível aplicar várias técnicas de projeto para determinação dos parâmetros do controlador que satisfazem às especificações transitórias e de

regime estacionário do sistema de malha fechada. No entanto, se a planta é tão complicada que seu modelo matemático não pode ser facilmente obtido, nesse caso a abordagem analítica do projeto de um controlador PID não é possível. Então devemos recorrer à técnicas experimentais para o projeto de controladores [OGATA, 1990]. No caso do trilho, será obtido o equivalente digital do controlador PID para que o mesmo seja implementado, através de equações de diferenças, em um computador.

Para podermos projetar o controlador PID é necessário modelarmos o sistema trilho, motor e redutor de forma a obtermos sua função de transferência. O levantamento desta função de transferência será feito de forma experimental colocando-se no acionamento do motor um sinal de referência na forma de degrau. Analisando a resposta do sistema para esta entrada pode-se determinar sua ordem, pólos e zeros.

Como requisito de projeto para o sistema controlador-trilho, foi definido que o sobre-sinal deve ser zero, com isto a plataforma irá tender para a posição definida sem oscilar em torno da mesma. Além disto, a resposta do sistema, ou seja, o tempo de assentamento deve ser o mais rápido possível.

Para se atender a estes requisitos deve-se ter uma solução de compromisso já que sistemas muito rápidos apresentam sobre-sinal elevado. Como conseqüência, o sistema deverá ter coeficiente de amortecimento igual a 1 pois este garante a resposta mais rápida quando sobre-sinal não é desejado. Além disto, o pólo de operação do sistema deverá ser escolhido de forma a tornar o controlador imune a possíveis variações no trilho. Isto é necessário já que a massa sobre a plataforma irá variar dependendo da peça que o robô estiver carregando.

2. Descrição do Problema

Nosso objetivo básico neste projeto de formatura é de obter um resultado satisfatório no que diz respeito ao posicionamento da plataforma onde se encontrará o robô em posições pré-definidas, para que este realize certas operações em conjunto com outros equipamentos que compõe a célula. Entenda-se por resultado satisfatório, um posicionamento com erro dentro do previsto para que, no trabalho em conjunto entre o robô e o outro equipamento, não haja interferências mecânicas capazes de prejudicar o trabalho, ou até mesmo danificar algum parte da célula. Além disto, o tempo de posicionamento deve ser da ordem de segundos. Podemos verificar aqui uma solução de compromisso na qual, de um lado temos o posicionamento correto e do outro um tempo pequeno. Quanto mais preciso o posicionamento, maior o tempo gasto para realizá-lo. Caso se queira um posicionamento muito rápido, o controle da posição fica prejudicado.

Como em toda célula de manufatura, esta também contém uma unidade supervisora e controladora de nível global (estação de trabalho) que envia e recebe informações de todos os equipamentos que compõe a célula. No nosso caso, a estação de trabalho que coordenará a célula "verá" nosso sistema como um comando numérico e enviará comandos em linguagem G reconhecida por tais equipamentos. Definimos esta maneira de trabalho para que houvesse uma padronização da comunicação da estação de trabalho com todos os equipamentos da célula.

Para isto, desenvolvemos um compilador de linguagem G que deve estar no microcomputador que controla o trilho. Este software foi implementado em Visual Basic, onde está a interface homem-máquina, e em Borland C, onde estão as funções que realizam a troca de sinais com as placas de interface colocadas no micro (placa de entradas digitais e placa conversora digital-analógica). A ligação entre o Visual Basic e o Borland C se dá através de uma DLL onde estão as funções implementadas em Borland C e que são utilizadas pelo Visual Basic. Este compilador deverá ter opções

para operação manual ou automática comunicando com a estação de trabalho.

Além disso, este software será responsável pelo acionamento de um motor que movimentará a plataforma sobre o trilho e pelo controle de posição da plataforma, realizado por uma malha fechada com controle proporcional integral derivativo (controlador PID), com realimentação por um encoder. Os cálculos das constantes deste controlador, para que os requisitos de projeto sejam satisfeitos, serão realizados utilizando-se conceitos de controle analógico e digital. A função de transferência da planta será obtida experimentalmente.

Para que este projeto seja possível alguns equipamentos foram tidos como dados entrada do problema, ou seja, já existiam na célula. Alguns estavam desligados, outros estavam funcionando sem nenhuma interface e a montagem do conjunto também faz parte do nosso projeto. Outros dispositivos serão projetados e implementados por nós ou por técnicos da faculdade. Um resumo das características destes equipamentos se encontra abaixo :

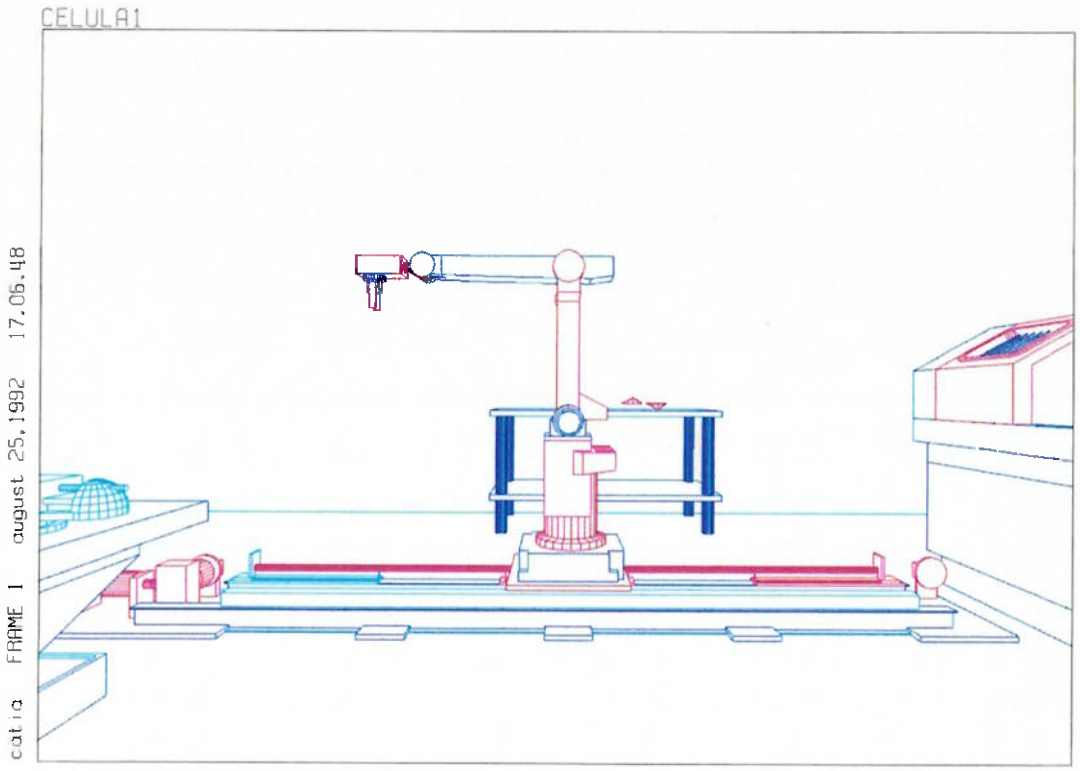


figura 3: Trilho de posicionamento do robô

- Trilho de 259 mm projetado e implementado com metal e madeira na própria faculdade, com uma guia para permitir deslocamento da plataforma e que já se encontrava pronto;
- Plataforma quadrada de madeira e base de metal de 52 mm onde será fixado o robô e que se movimenta sobre o trilho através de rolamentos que são fixados na parte superior e inferior da guia do trilho, permitindo somente o movimento em uma única direção. Além disso, possui uma guia para ser percebida pelos sensores de fim de curso, e é amarrada a um cabo de aço que se enrola à um tambor com 9 voltas;
- Tambor de aço com 120 mm de diâmetro no qual é enrolado o cabo de aço que é ligado à plataforma. De um lado deste tambor é ligado o redutor e do outro o gerador de pulsos (encoder);

- Redutor com relação de 1:10 com entrada e saída perpendiculares de marca CESTARI tipo K-40, com potência de 1,08 CV a 1750 rpm;
- Gerador de pulsos óptico (encoder) marca DIADUR modelo ROD-428, com dois discos graduados que permitem discernir se este está girando no sentido horário ou anti-horário;
- Servo-motor de corrente contínua de marca VARIMOT S.A. modelo SD-2530 com as seguintes características :
  - Tensão máxima: 130 Vcc
  - Torque: 3N.m
  - Rotação de saída em vazio: 3000rpm
  - Tacogerador: 9,5V/1000rpm.

  Este servo-motor é conectado à entrada do redutor por uma junta fixa;
- Um servo-amplificador (parametrizável) de marca TRANSAXE série 700 modelo 710 (conversor estático para motores CC) com uma malha controlada por um circuito proporcional integral, é realimentado por um tacogerador e trabalha com pulsos (PWM - Pulse Width Modulator ou modulador de largura de pulsos). Tem saída para o motor e entrada o micro, que irá fornecer o sinal de referência;
- Placa contadora de pulsos, desenvolvida por técnicos do próprio departamento e que conta os pulsos gerados pelo encoder;
- Placa de aquisição de dados marca LINX modelo ESD6401 (parametrizável) que será responsável pela aquisição do número de pulsos contados pela placa contadora, bem como pela entrada de sinal das chaves de fim-de-curso;
- Conversor analógico/digital e digital/analógico marca LINX modelo CAD 10/26 (parametrizável) de 10 bits de resolução que será responsável pela interface entre o microcomputador e o servo-amplificador;

- Três circuitos de final de curso óptico a serem projetados e desenvolvidos por nós;
- Microcomputador compatível com IBM modelo 80486DX2 com 66 MHz de velocidade e 4 Mbytes mínimos de RAM (para que seja possível funcionar Windows 3.1);
- Estação de trabalho marca IBM modelo RISC-6000 com cartão de comunicação padrão ETHERNET;
- Placa de comunicação (rede) entre o microcomputador e a estação de trabalho.

Em resumo, nosso projeto consiste em assumir uma série de requisitos para que o sistema funcione satisfatoriamente, montar e calcular as equações que regem o sistema, parametrizar as placas, montar os equipamentos e comunicar todo o sistema, além de desenvolver o software de controle e interpretação de código G.

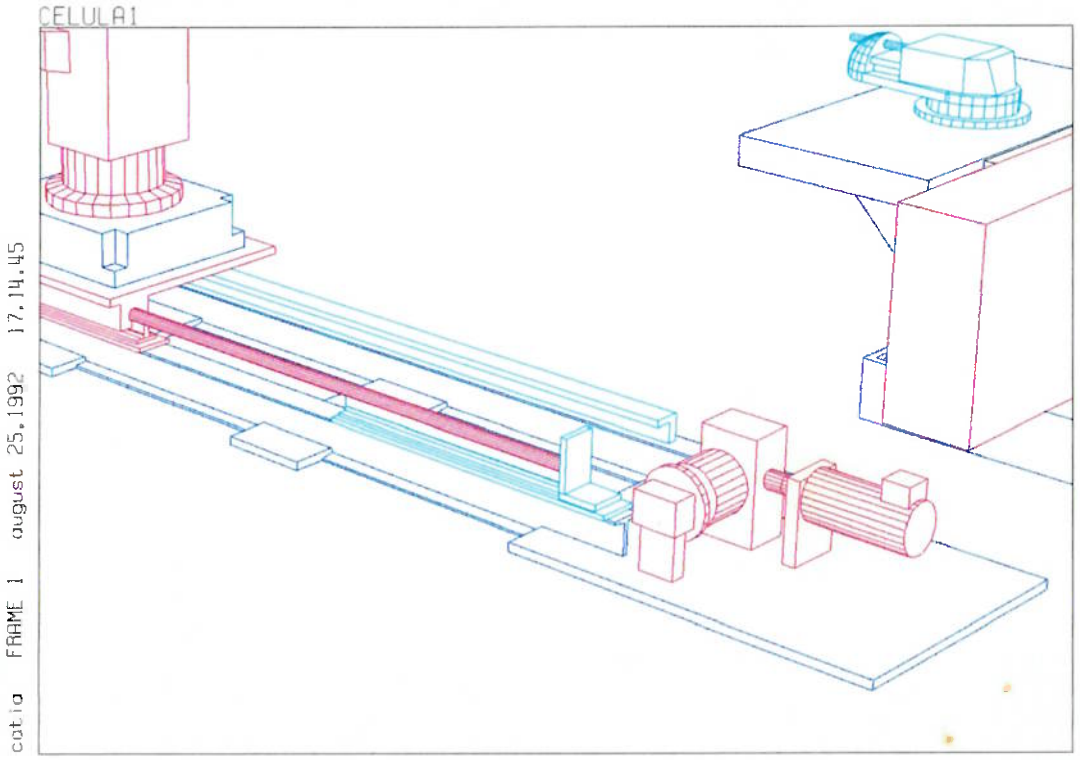


figura 3b: Detalhe do acionamento da plataforma

3. Desenvolvimento realizado

3.1 Definição das funções de programação

Após estudo das funções disponíveis na linguagem G, definimos aquelas que se aplicam ao trilho de posicionamento do robô.

3.1.1 Funções Preparatórias

- g00 - Modo de Posicionamento:

Sob este modo preparatório o posicionamento é feito de maneira precisa até a posição programada, em avanço rápido. O avanço rápido será definido como a maior velocidade permitida pelo sistema.

Esta função é modal, ou seja, a partir de sua programação torna-se válida para todo o programa até que uma função que a cancele seja programada, e é cancelada por g01.

- g01 - Interpolação Linear:

A interpolação linear, definida por g01, faz com que o trilho mova-se em velocidade programada de avanço. O vetor velocidade de avanço terá a direção do movimento.

A função g01 é modal e é cancelada por g00.

- g04 - Permanência:

É a função preparatória que define a permanência para o programa. Sob este modo o programa permanece parado no tempo previsto e, esgotado este tempo, parte para a execução do bloco seguinte programado.

O valor do tempo programado entra, no bloco onde é programada a função g04, através da função 'x', que, sob este modo, define o tempo em segundos.

Esta função é não modal, ou seja, toda a vez que for necessária deve ser programada pois é válida apenas no bloco que a contém.

- g90 - Programação com sistema de coordenadas absolutas:

Sob este modo a programação é feita de forma que todas as coordenadas referem-se a um sistema de coordenadas absolutas. A origem

do eixo de movimentação no trilho corresponde ao centro do mesmo. No entanto esta referência pode ser alterada pela função g92 (ver abaixo).

A função g90 é modal e é cancelada por g91.

- g91 - Programação com sistema de coordenadas incremental:

Sob este modo de programação, o posicionamento especificado em um bloco é feito em relação ao ponto anterior e não em relação ao zero do trilho.

Esta função é modal e é cancelada por g90.

- g92 - Definição da origem do sistema:

Esta função deve ser complementada pela função 'x', onde é especificado, em relação ao centro do trilho, o novo zero do eixo de movimentação.

A função g92 é modal e é cancelada por g93.

- g93 - Zera trilho:

Esta função retorna o zero de referência para seu valor padrão, ou seja, para o centro do trilho. A função g93 é modal e é cancelada por g92.

3.1.2 Função Número de Seqüência

A programação do número de seqüência é feita pela função "n", cujo formato é n seguido de uma seqüência de 4 dígitos (ex: n0010), que é usada para identificar o bloco de informação.

3.1.3 Função Auxiliar de Avanço

A função f define a velocidade de avanço na direção do movimento do trilho. Esta velocidade deve ser definida em porcentagem da velocidade máxima (variando entre 0 e 100 %) e, uma vez programada, torna-se válida para todo o programa já que a função f é modal. Caso se necessite de uma outra velocidade, a partir de um determinado bloco, nova função f deve ser programada.

3.1.4 Funções Miscelânea

- m00 - Parada do programa. Função não modal.
- m02 - Fim do programa. Função não modal.
- m03 - Função "GO TO". Função não modal que desvia a execução do programa para um determinado bloco.

3.1.5 Função de posicionamento

A função x especifica a posição a ser ocupada pelo robô. Esta posição pode ser em relação ao zero do sistema ou em relação à posição anterior, dependendo do sistema de coordenadas escolhido.

É utilizada após as funções g01 e g00. Também é utilizada com g04, mas, nestes blocos, especifica o tempo de espera do programa e não a posição a ser ocupada pelo robô. A função x é não modal. Além disto, a função x é usada após m03 para definir o número da linha para a qual o programa será desviado.

O interpretador da linguagem G, interpreta o caracter que representa a tecla "Enter" como indicador de final de bloco.

3.1.6 Sintaxe da linguagem G

Cada bloco do arquivo a ser executado, deve ter a seguinte sintaxe:

n9999 g99 x9999 f999 m99 EOB

Todos os zeros à esquerda nos números devem ser digitados para que o compilador reconheça o comando. Por exemplo n10 deve ser escrito como n0010.

3.2 Interface Homem-Máquina

3.2.1 Tela inicial

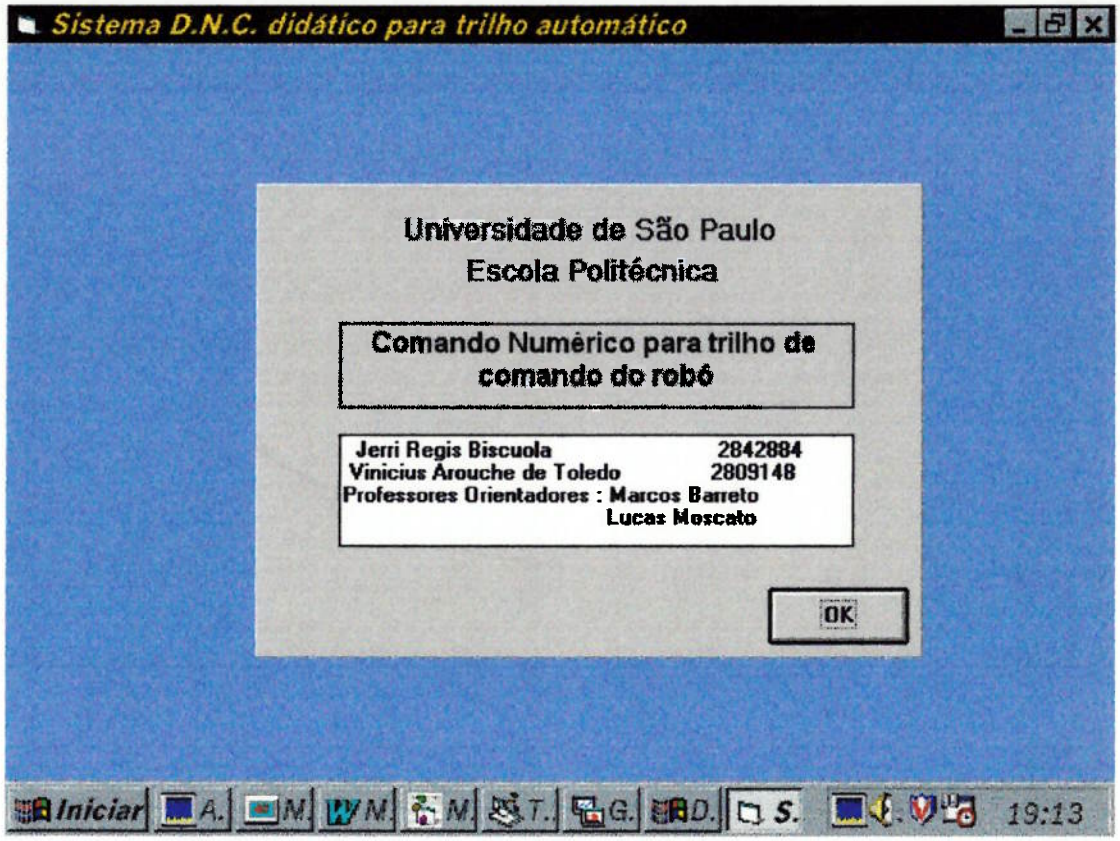


figura 4: Tela Inicial do sistema de controle do trilho

Esta tela é a tela inicial do sistema de controle do trilho. Contém o título do projeto, bem como seus projetistas. Clicando-se sobre a tecla OK, o sistema entra no módulo de controle, que será visto à frente.

3.2.2 Tela de principal

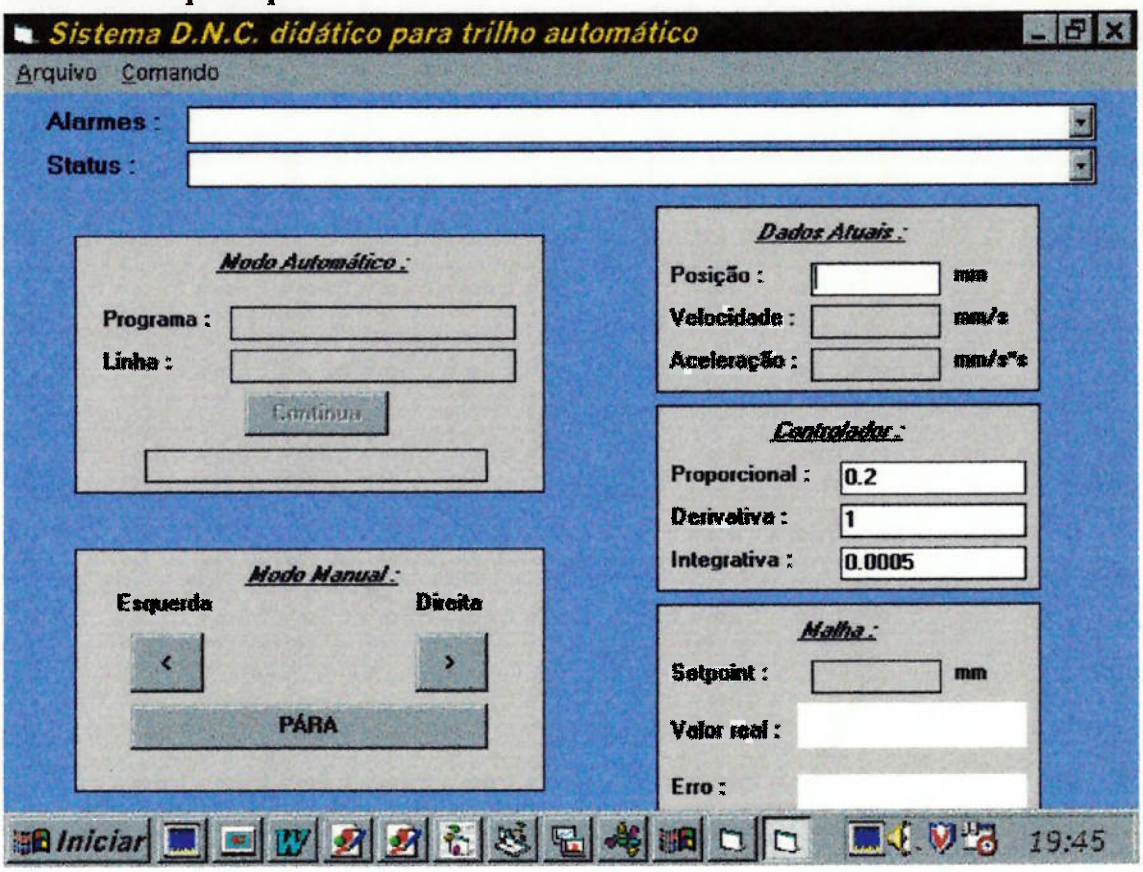


figura 5: Tela de Principal

Nesta tela encontramos a opção de Arquivo, para selecionarmos o arquivo que iremos trabalhar, e a opção Comando, que seleciona de que modo o trilho irá receber os dados.

Com ALT+A ou ALT+O entramos diretamente nas opções Arquivo e Comando respectivamente.

Nesta opção, podemos escolher também entre modos manual e automático, ou ainda optar pela execução ou compilação de um programa de extensão G, selecionados com o comando abrir arquivo (que será visto mais a frente).

No modo manual o controle do trilho é dado ao operador com teclas já definidas, e é o modo default, ou seja, quando o programa é acionado pela primeira vez ele entre neste modo (campos do modo automático serão inibidos). Já no modo automático o controle é feito por uma unidade de comando superior e a comunicação é via rede (que não foi implementada) ou

por arquivos já existentes no próprio microcomputador que controla o trilho. Neste caso serão inibidos os comandos do modo manual.

Quando a seleção Automático é selecionado a plataforma irá para o centro do trilho automaticamente para começar o processamento do arquivo.

Quando optamos por executar um arquivo, este arquivo irá fornecer comandos para o trilho. Ao invés disso, se quisermos somente testar o arquivo de programa para verificar se este não possui erro de digitação, escolhemos compilar arquivo. Este arquivo deve estar em disco (rígido, flexível ou CD-ROM).

Quando optamos pelo modo de operação manual, o controle do trilho é todo realizado pelo teclado do microcomputador que o supervisiona. Neste caso o usuário aumenta ou diminui a velocidade de trabalho (que inicialmente é zero) e se esta velocidade for inferior a velocidade máxima permitida no trilho, essa será a velocidade a qual o trilho tentará atingir com a rampa de aceleração. Caso contrário o computador enviará uma mensagem para o usuário diminuir a velocidade abaixo da velocidade máxima.

Os comando são simples : a tecla < leva a plataforma para a esquerda, a tecla > leva a plataforma para a direita e a tecla de espaço para a plataforma.

Devemos tomar muito cuidado nestas condições para que a plataforma não bata no final do trilho com velocidade muito alta, pois neste caso o sensor de final de curso gerará um degrau para velocidade zero e a desaceleração será muito alta podendo danificar algum componente.

Com a opção compilar, o arquivo selecionado em ABRIR ARQUIVO aparecerá no campo "Arquivo". Isto quer dizer que o sistema está compilando este arquivo.

A opção executar, compila o arquivo novamente para verificar se este não foi alterado e então começa a execução, sempre iniciando no centro do trilho. Na execução do comado m00, será apresentado ao usuário a tecla continua, que deverá ser pressionada caso o este deseje continuar a execução.

Qualquer mensagem de erro ou status do acionamento será mostrado na linha de alarmes e status.

As atualizações da tela quanto a posição, velocidade, bar graph de erro e setpoint são instantâneas.

3.2.3 Tela de Selecionar Arquivos

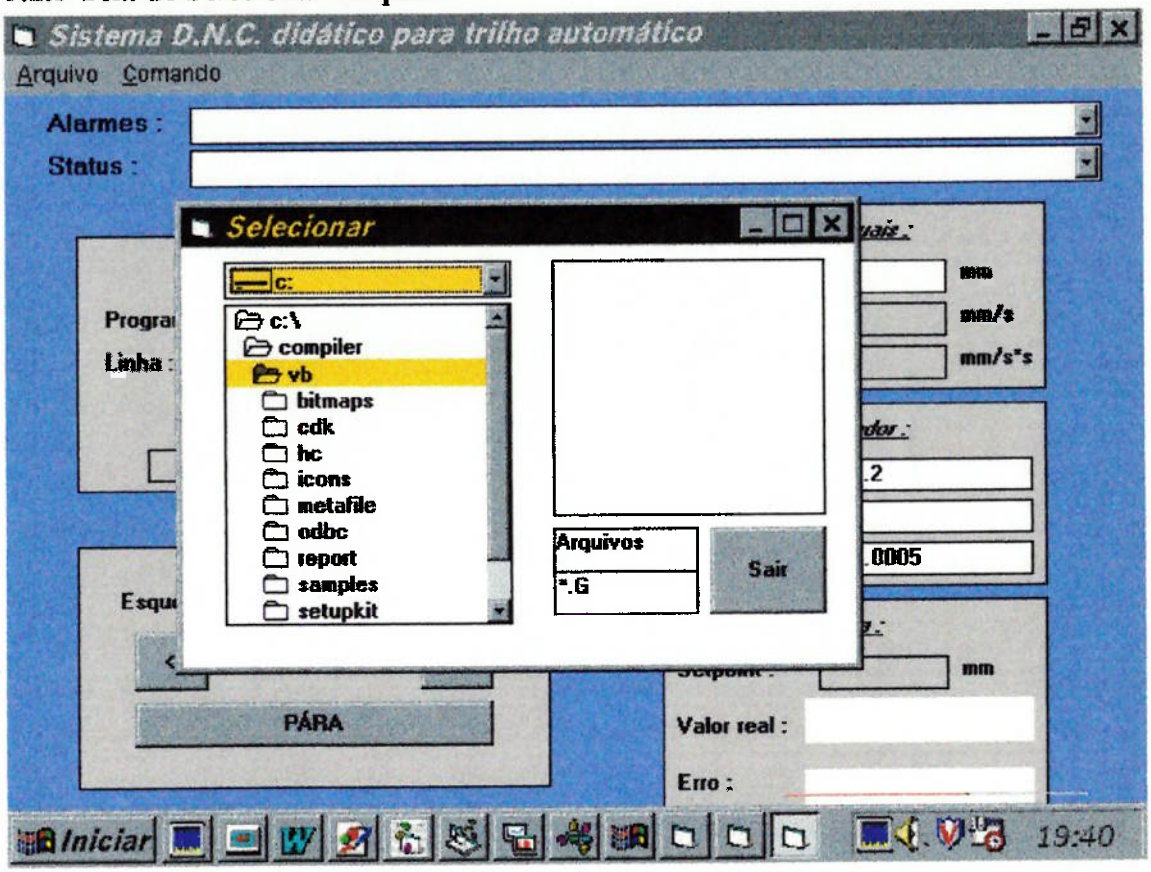


figura 6: Tela de Selecionar Arquivos

Nesta tela podemos selecionar arquivos para compilação e/ou execução. Estes arquivos deverão ter terminação *.G e são selecionados, dando-se um duplo click do mouse sobre eles. Podemos também navegar no interior de diretórios bem como mudar a unidade de disco no qual o programa se encontra.

Uma vez selecionado o arquivo, toda vez que escolhermos as opções compilar ou executar será tomado como referência o arquivo selecionado nesta etapa.

A opção SAIR, cancela a seleção de um novo arquivo e mantém o anterior.

3.3 Definição das rampas de aceleração e desaceleração

As rampas de aceleração e desaceleração poderiam ser feitas de duas maneiras. Na primeira delas, o início da movimentação da plataforma se dá com a maior aceleração permitida pelo conjunto; caso seja atingida a velocidade especificada pelo programa, antes de se atingir a metade do percurso, a velocidade permanece neste valor até atingir a metade da movimentação especificada. Após a metade do percurso, a plataforma começa a freiar até atingir a posição especificada (ver figura 13). Caso a metade do percurso seja atingida, antes da plataforma atingir a velocidade especificada, o comando do trilho freia a plataforma até que esta atinja a posição determinada. Isto ocorre quando a diferença entre a posição atual da plataforma e a próxima posição é muito pequena e o motor não consegue atingir a velocidade especificada pelo código G (ver figura 14).

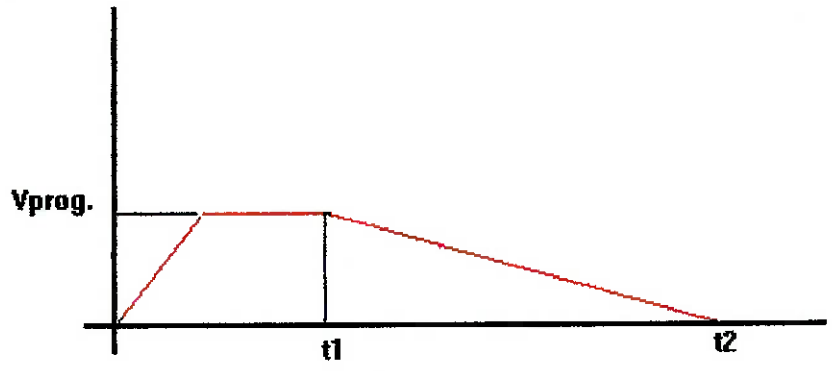


figura 7: 1ª opção para rampas de aceleração

Na figura acima tem-se:

- Vprog.: Velocidade programada;
- t1: tempo até atingir metade do percurso;
- t2: tempo até atingir a posição especificada.

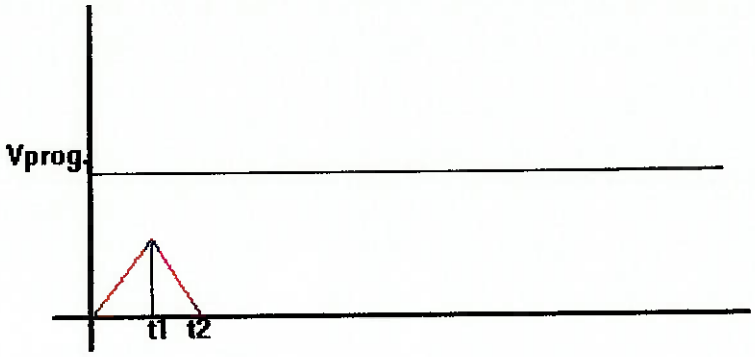


figura 8: Motor não atinge a velocidade especificada

Na figura acima tem-se:

-Vprog.: Velocidade programada;

- t1: tempo até atingir metade do percurso;

- t2: tempo até atingir a posição especificada.

Na segunda opção de rampas de aceleração, a plataforma inicia sua movimentação com a máxima aceleração permitida pelo conjunto. Caso seja atingida a velocidade especificada, antes da plataforma atingir a metade do percurso, o motor permanece nesta velocidade e começa a freiar próximo à posição especificada, de forma que o tempo de desaceleração seja igual ao de aceleração (ver figura 15). Caso o motor não atinja a velocidade especificada antes da metade do percurso, as curvas de aceleração e desaceleração serão iguais àquelas dadas pela 1ª opção (ver figura 14).

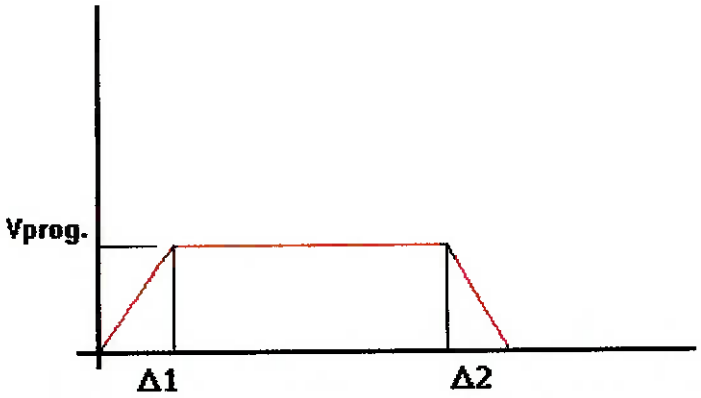


figura 9: 2ª opção para curvas de aceleração

Na figura acima, tem-se:

-Vprog.: Velocidade programada;

-Δ1= tempo de aceleração;

-Δ2= tempo de desaceleração;

-Δ1=Δ2.

Para o nosso projeto a plataforma do trilho terá curvas de aceleração dadas pela 2ª opção. Com isto, a movimentação é feita na velocidade especificada por um maior período de tempo, o que faz com que a posição desejada seja atingida mais rapidamente.

3.4 Estrutura de dados do programa de controle

O compilador de código G foi implementado em Visual Basic e deve receber o programa na forma de um arquivo ASCII, que irá ser interpretado. No entanto, a execução deste programa não ocorrerá enquanto o arquivo estiver sendo lido pois isto faria com que a execução do programa ficasse muito lenta. Por isto, ao compilar o código G, o software verifica a sintaxe do programa linha a linha e, caso a mesma possua erros, adiciona este erro em uma lista que será mostrada ao operador do sistema. Em paralelo, o interpretador cria um vetor de registros (ver figura 16) onde cada elemento do vetor irá conter um bloco do programa. O elemento do vetor é uma estrutura de dados que armazena as funções de seu respectivo bloco.

Para executar o programa o interpretador percorre este vetor, executando as funções armazenadas em cada elemento, ou seja, as funções de cada bloco.

A escolha de um vetor como estrutura de dados se deve à facilidade de se implementar o mesmo no Visual Basic. Além disto, com a utilização de um vetor foi dispensada a necessidade de se criar as funções de manipulação de listas ligadas:

- Cria lista;
- Insere elemento na lista;

- Remove elemento da lista;
- Consulta elemento da lista;
- Deleta lista.

Outra vantagem da utilização de um vetor é a facilidade de se desviar a execução do programa no caso da utilização da função GOTO (m03). Com relação à desvantagem da utilização de um vetor pode-se citar a limitação do programa em código G já que o tamanho do vetor é estipulado em tempo de compilação do programa, ou seja, uma vez programado, o tamanho do vetor é fixo.

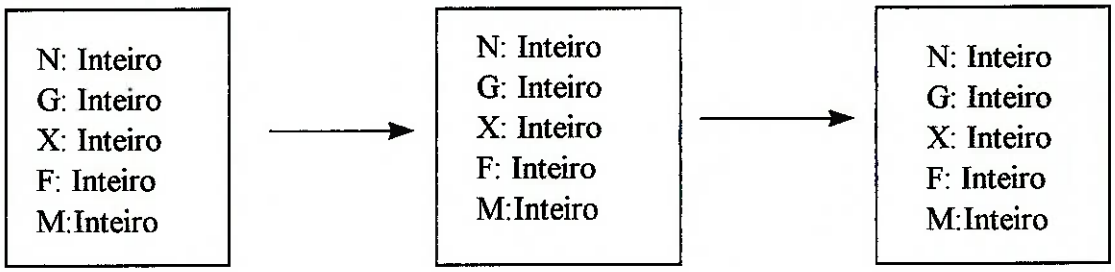


figura 10: Estrutura de dados para o interpretador

3.5 Parametrização da placa controladora do servo-amplificador[1]

Esta placa tem formato duplo-Europa, encaixável e contém as funções necessárias para a regulação de servo-motores de corrente contínua. O modelo 710 trabalha com uma corrente de regime de 12,5 A, corrente máxima de 25 A e com potência de 2000 W. Além disso podemos destacar algumas características importantes:

- Máximo sinal de referencia : ± 10 V;
- Máxima tensão de tacômetro : 100 V;
- Faixa de temperatura de operação : 0 - 45 ºC;
- Tensão de alimentação : 50 - 160 Vcc ou 3 x 40 - 115 Vac ± 10% 50/60Hz;
- Capacitor de filtro de 1000 μF / 250 V;

[1] Estas informações foram obtidas do manual da placa, fornecido pelo fabricante.

- Dissipação instantânea do excedente da energia de frenagem;
- Regulação de velocidade PI;
- Regulação de corrente PI;
- Controle em quatro quadrantes, com pulsos de potência modulados em PWM;
- Precisão de regulação menor que 0,5 % com velocidade máxima;
- Proteções incorporadas :
  - Falta de taco
  - Corrente eficaz
  - Curto-circuito
  - Termostato
  - Limitação automática da corrente em função da temperatura
  - Limitação da corrente em função da rotação
  - Supervisão de sobre tensão no momento da frenagem do motor
- Indicações de estado por meio de LED :
  - Fonte
  - Falha de tacogerador
  - Sobre corrente
  - Stand-by
  - Sobre temperatura
  - Dispositivo de frenagem acionado
- Regulagens no frontal do equipamento :
  - Ganho
  - Corrente máxima
  - Corrente eficaz
  - Tacômetro
  - Offset
- Funções auxiliares :
  - Entrada de Stand-by em 24 V opto-acoplada

- Duas entrada de fim de curso em 24 V opto-acopladas
- Entrada de tensão de referência nula em 24 V opto-acoplada
- Sinal de servo pronto através de contato de relê
- Saídas de ± 15 V e terra (máximo de 20 mA)

As entradas dos sinais de comando e saídas dos sinais de supervisão são realizadas através de bornes numerados de 1 a 16. Assim temos:

- Bornes 1 e 2 : Saídas dos contatos do relê "pronto" que indica que o servo amplificador está funcionando com o contato Stand-by fechado e sem alarme.
- Bornes 3, 4 e 5 : Saídas das tensões +15 V, 0 V e -15 V, respectivamente, que estão a disposição do usuário com corrente nominal de 20 mA.
- Borne 6 : Stand-by por sinal opto-acoplado.
- Borne 7 : Velocidade nula por sinal opto-acoplado.
- Borne 8 : Fim de curso 1 por sinal opto-acoplado.
- Borne 9 : Fim de curso 2 por sinal opto-acoplado.
- Borne 10 : Entrada de referência da tensão opto-acoplada.
- Borne 11: Conectado ao + do tacogerador
- Borne 12: Conectado à blindagem do cabo do tacogerador
- Borne 13: Conectado ao - do tacogerador
- Borne 14, 15 e 16 : Entrada do sinal de referência (± 10 V) onde o borne 15 é conectado à blindagem do cabo da tensão de referência.

Como este tipo de placa pode ser usada para uma enorme variedade de aplicações, quando ela sai de fábrica já testada, vem com alguns componentes que devem ser redimensionados pelo usuário, e soldados em seus respectivos lugares conforme diagrama da placa que se encontra no anexo 7.1.

Para a realização dos cálculos, tomamos por base que utilizaremos a configuração para regulação de velocidade com tacogerador. Com isso, seguindo o roteiro dado pelo manual do fabricante, obtivemos :

- Colocar o jumper W1 na posição 1-2
- Calculo da resistência R2 :

$$R2 \leq \frac{250}{(Vdin \cdot Rmax) - 10}(kOhm)$$

onde Vdin: Tensão do dínamo;

Rmax: Rotação máxima.

A tensão do dínamo, dada em V / 1000 rpm e a rotação máxima, dada em milhares de rpm são características do servo-motor e no nosso caso são 9,5V/1000rpm e 3000 rpm, respectivamente.

Assim:

$$R2 \leq \frac{250}{(9{,}5 \cdot 3{,}0) - 10} = 13{,}5kOhm$$

Utilizamos a supervisão do tacogerador. Assim o manual recomenda :

- Colocar o jumper W3 na posição "on".
- Calcular e soldar os resistores R43 e R45, onde:

$$R43 = R45 = \frac{5400}{Rmax \cdot Ke}(kOhm)$$

No nosso caso temos 3000 rpm de rotação máxima (como já foi visto acima) e a constante Ke é uma constante elétrica do motor e tem o valor de 45V/1000rpm (dado pelo manual do fabricante do motor).

Assim :

$$R43 = R45 = \frac{5400}{3{,}0 \cdot 45} = 40kOhm$$

-Deixar o resistor R44 que vem de fábrica.

-Ajustar trimpot P8 no mínimo (totalmente no sentido horário).

-Ajustar trimpot P9 no mínimo (totalmente no sentido anti-horário).

3.6 Interface D.N.C.

A definição interface D.N.C. (Distributed Numeric Control) basicamente consiste em definir quais tipos de dados o controlador e a máquina trocarão. Assim definimos que do controlador para a máquina teremos a descarga de programas (DOWNLOAD) e a supervisão. Essa supervisão consiste basicamente na obtenção da posição da plataforma no trilho (cuja precisão depende da resolução do encoder), na determinação e obtenção das constantes do compensador proporcional integral derivativo (PID). A posição e as constantes do compensador serão números reais.

Além disso, o controlador pode mandar a máquina executar um determinado programa que se encontra na memória da mesma (identificado por um número inteiro), parar esse programa por um determinado tempo até enviar um comando de continue, para que a máquina continue do ponto onde parou ou até mesmo parar um programa por completo para iniciar outro.

Por outro lado, no que se refere à troca de dados da máquina para o controlador, basicamente se resume no envio de código de erros (representados por números inteiros) que serão interpretados pelo controlador. Esses códigos de erros podem requerer que o usuário reconheça a mensagem (pressionando-se uma tecla de reconhecimento por exemplo) para que a máquina continue a funcionar, ou podem servir somente como aviso não sendo de suma importância para o funcionamento da máquina.

A implementação desta interface dependia de um perfeito funcionamento do trilho. No entanto, ao término da montagem do hardware foi dispendido um certo tempo para se solucionar problemas de ruído nos sinais das chaves fim-de-curso e no sinal de contagem resultante da placa contadora. Como resultado não houve tempo para a implementação da mesma, mas a escolha do ambiente Windows deve facilitar a colocação em funcionamento desta interface já que a plataforma escolhida conta com vários pacotes para redes disponíveis no mercado.

3.7 Arquitetura do Hardware

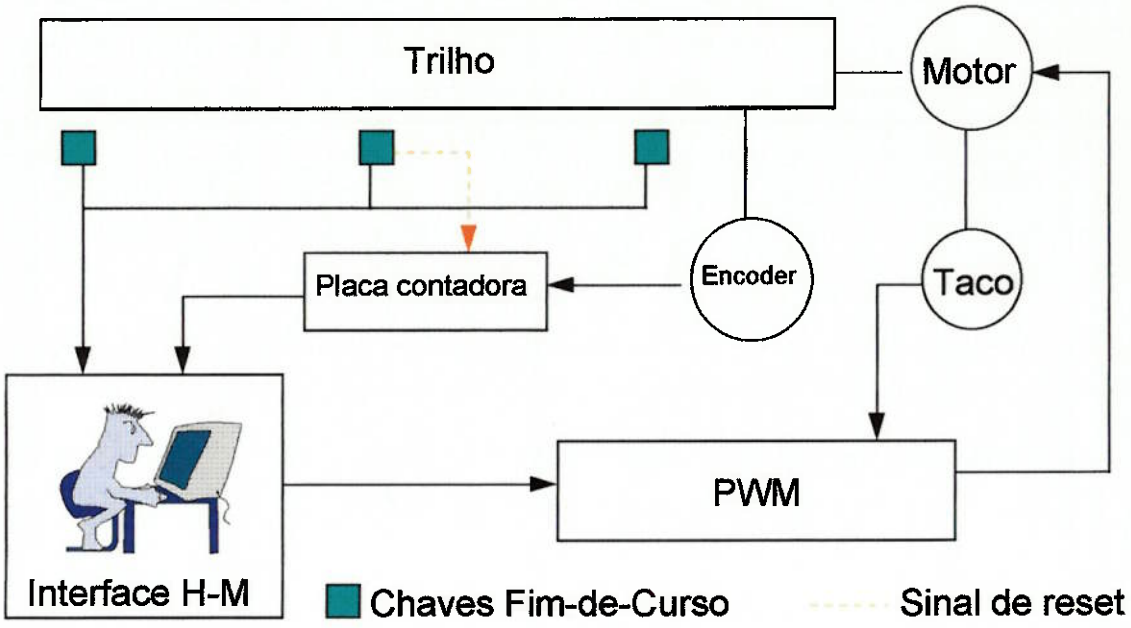


Figura 11: Arquitetura de hardware

Basicamente, a parte mecânica da célula foi constituída como o esquema acima. No trilho há três chaves de fim de curso ópticas montadas pelos próprios alunos. As duas das extremidades mandam um bit de controle diretamente para o computador através de entradas digitais, para que este possa parar a plataforma com segurança. A chave do centro manda um bit de controle para a placa contadora, e este bit é usado como reset desta placa para anular eventuais pequenos erros de contagem ou de deslizamento do cabo de aço que aciona a plataforma.

Uma outra entrada digital de 16 bits que está no microcomputador, recebe a contagem da placa contadora, e com isso temos a posição instantânea da plataforma no trilho, pois esta placa contadora conta pulsos elétricos enviados pelo encoder óptico. Estes pulsos são defasados para que a placa identifique se é preciso incrementar ou decrementar a contagem.

Esta placa contadora será detalhada mais adiante.

Dentro do microcomputador que controla o trilho, além das entradas digitais, existe uma placa conversora digital-analógica, que envia dois sinais

variando de -5 à +5 Volts para o acionamento. Estes sinais são proporcionais ao número que mandamos escrever na porta de saída (0 significa -5 Volts / 255 significa +5 Volts / 128 significa 0 Volts).

O acionamento (P.W.M.) diferencia estes dois canais (ou seja recebe um sinal variando de -10 a +10 Volts) e gera uma tensão e uma corrente para o motor poder acionar a plataforma (-10 Volts significa rotação máxima no sentido horário / +10 Volts significa rotação máxima no sentido anti-horário / 0 Volts significa motor travado). Este acionamento em particular tem um controlador proporcional derivativo no seu interior de tensão e corrente, e para isso precisa de uma realimentação que vem através do tacogerador.

Na figura abaixo detalharemos um pouco mais como os bits "chegam e saem" do controlador do trilho.

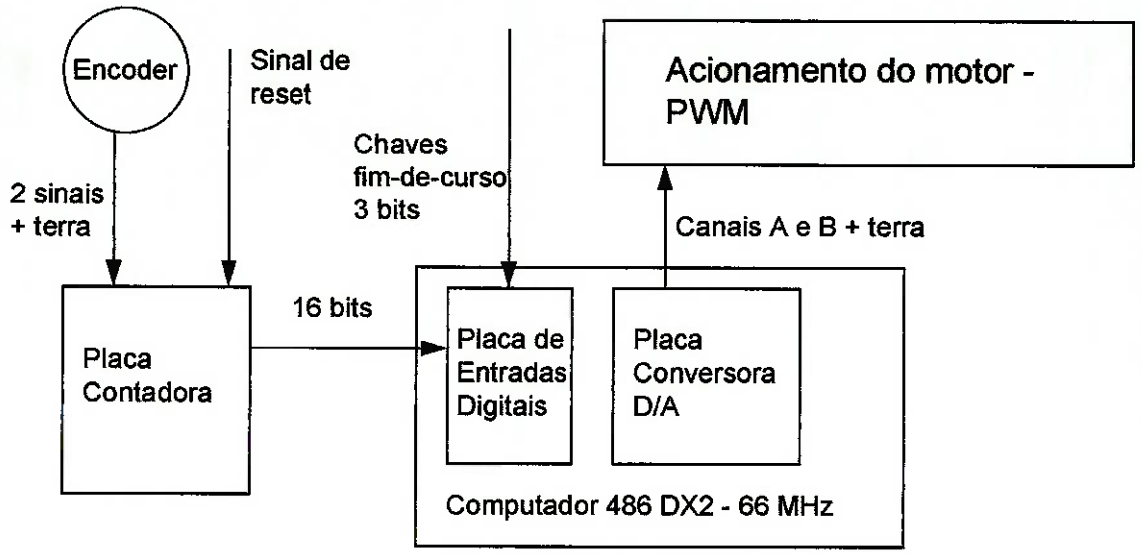


Figura 12: Detalhamento dos sinais de controle

3.7.1 Placa Contadora

Para a contagem dos pulsos gerados pelo encoder foi construída um circuito digital contador. Este contador é formado por 4 contadores de 4 bit's ligados em cascata de maneira a se ter um resultado de 16 bits. A contagem para cima ou para baixo, ou seja, o sinal de COUNT-UP ou COUNT-DOWN para os contadores é gerado por portas lógicas que identificam a defasagem entre os 2 sinais provenientes do encoder e, com isto, pode ser estabelecido o sentido de giro do encoder (horário ou anti-horário).

Na saída dos contadores foram colocados dois *buffers* de 8 bits de forma a se isolar a placa contadora da placa de entradas digitais colocada no computador. Isto é feito para que problemas em uma das placas não acabem por prejudicar a outra.

Com relação ao sinal de RESET para os contadores, este é gerado pela chave ótica colocada no centro do comprimento total das guias. Como este é o zero absoluto do trilho, garante-se que a contagem seja sempre zero ao se passar neste ponto. Isto acaba por reduzir erros de posicionamento caso a contagem não volte a zero quando a plataforma passa pelo centro.

Os sinais de entrada da placa são recebidos em um conector para cabo do tipo *flat-cable* onde estão ligados a alimentação da placa, do circuito e dos sensores do encoder, os dois sinais provenientes do encoder e o sinal de RESET da contagem (chave ótica no centro do trilho).

Já os sinais de saída também são enviados através de um *flat-cable* que envia os 16 bits de contagem para o computador.

3.7.2 Chaves óticas

Neste projeto o fim de curso da plataforma, nos dois lados, e o zero foram implementados através de chaves óticas que enviam um sinal de +5Vcc para o computador quando as mesmas são acionadas.

O componente principal destas chaves é um sensor que aloja em um mesmo componente um LED e um FOTO-TRANSISTOR. Este foto-transistor funciona como uma chave liga-desliga que é acionada quando recebe a luz do LED e desliga quando esta é interrompida. Além deste sensor, as chaves também possuem 2 resistores cerâmicos que limitam a corrente de alimentação.

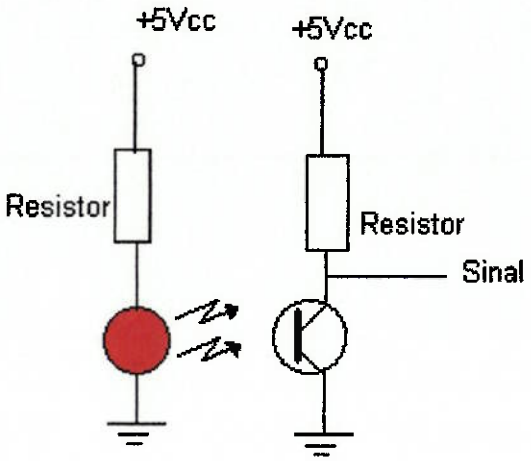
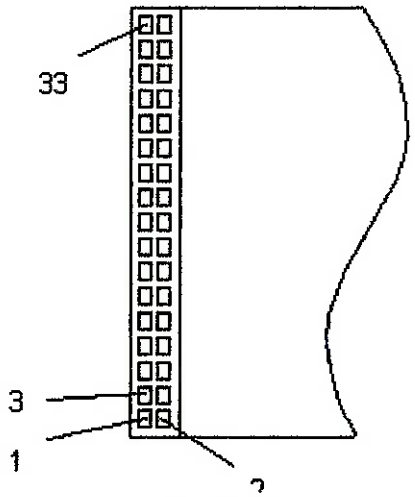


figura 13: Chave Ótica

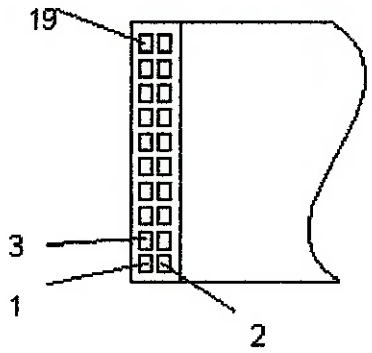
Quando algum objeto interrompe a luz que chega no foto-transistor, este se torna uma chave aberta e a corrente é desviada

3.7.3 Conector placa contadora - entrada digital (CN5)



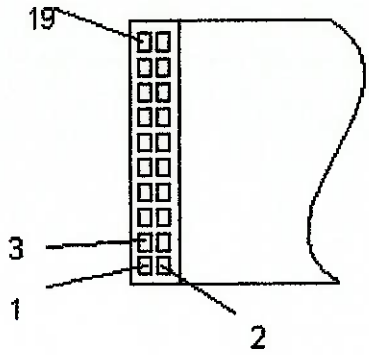
1 - livre	2 - livre
3 - terra	4 - terra
5 - bit 1	6 - bit 2
7 - bit 3	8 - bit 4
9 - bit 5	10 - bit 6
11 - bit 7	12 - bit 8

13 - bit 9	14 - bit 10
15 - bit 11	16 - bit 12
17 - bit 13	18 - bit 14
19 - bit 15	19 - bit 16
21 - livre	22 - livre
23 - livre	24 - livre
25 - livre	26 - livre
27 - livre	27 - livre
29 - livre	30 - livre
31 - livre	32 - livre
33 - livre	34 - livre



1 - bit 1	2 - bit 2
3 - bit 3	4 - bit 4
5 - bit 5	6 - bit 6
7 - bit 7	8 - bit 8
9 - livre	10 - livre
11 - bit 9	12 - bit 10
13 - bit 11	14 - bit 12
15 - bit 13	16 - bit 14
17 - bit 15	18 - bit 16
19 - livre	20 - terra

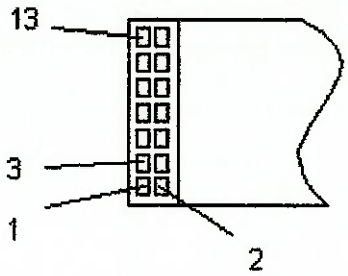
3.7.4 Conector placa contadora - encoder



1 - livre	2 - livre
3 - livre	4 - livre
5 - livre	6 - livre
7 - livre	8 - livre
9 - livre	10 - reset
11 - terra	12 - terra
13 - 5 V	14 - 5 V
15 - 5 V	16 - 5 V
17 - terra	18 - terra
19 - sinal 1 encoder	20 - sinal 2 encoder

Con. encoder	Cor cabo	Significado
2	azul	sensor + 5 V
5	laranja	sinal 1
8	marrom claro	sinal 2
9	shield	shield
10	branco	0 V
11	cinza	sensor 0 V
12	marrom escuro	5 V

3.7.5 Conector final de curso - entradas digitais (CN3)



1 - livre	2 - terra
3 - livre	4 - livre
5 - livre	6 - livre
7 - livre	8 - sinal do fim de curso central
9 - sinal do fim de curso esquerdo	10 - sinal do fim de curso direito
11 - livre	12 - livre
13 - terra	14 - terra

Obs.: Todos os fins de curso possuem + 5 V no cabo vermelho, terra no cabo preto, sinal no cabo branco e sua malha aterrada.

3.8 Arquitetura do Software

3.8.1 Compilador

O compilador é executado quando se clica na opção *"Compilar Arquivo"* do menu *"Comando"*. Este bloco do programa verifica se um arquivo texto (código G) foi selecionado ou não. Caso a seleção não tenha sido feita ele gera uma caixa de mensagem pedindo que a seleção seja feita. Caso contrário, o arquivo é aberto e a primeira linha é lida.

Após a leitura desta linha, são inseridos os caracteres de fim de linha e a sintaxe da mesma é verificada. Para esta verificação, o programa começa a varrer a linha lida e, através de IF..THEN..ELSE testa a

formatação da linha. Se esta estiver correta, o vetor de dados é atualizado com os comandos da linha, mas se houver algum erro o vetor também é atualizado mas uma mensagem de erro é adicionada na lista de erros de compilação.

Outra característica deste compilador é que como a linguagem G possui comandos modais, isto é, comando que uma vez programados não necessitam ser repetidos, o compilador verifica quais foram os comandos modais programados e os coloca automaticamente no vetor de dados. Com isto o interpretador é executado mais rapidamente pois cada elemento do vetor de dados irá conter todas os comandos necessários para sua execução. Por exemplo: se o operador programar a seguinte seqüência:

N0010 G01 X0010 F020

N0020 X0020

Neste caso, ao ler a primeira linha o compilador identifica os comandos modais G01 (posicionamento com velocidade programada) e F020 (velocidade igual a 20% da velocidade máxima) e automaticamente, ao montar o vetor de dados, coloca estes comandos no elemento do vetor correspondente à linha 20, onde os mesmos não foram programados. Esta inclusão automática dos comandos é cancelada quando um outro comando modal for programado e este cancele o primeiro.

Com relação às coordenadas de posicionamento, mesmo que sejam programadas coordenadas incrementais ou que o zero do trilho seja deslocado, o compilador calcula a posição absoluta e este valor é colocado no vetor. Isto também foi feito para que o interpretador seja executado mais rapidamente, já que o mesmo não irá fazer estes cálculos.

3.8.2 Interpretador

O interpretador é executado quando se clica na opção *"Executar Arquivo"* do menu *"Comando"*. Este bloco do programa também verifica se um arquivo texto (código G) foi selecionado ou não. Caso a seleção não tenha sido feita ele gera uma caixa de mensagem pedindo que a seleção

seja feita. Caso contrário, o arquivo é aberto e compilado utilizando-se a procedure descrita acima.

Após a compilação o programa retorna um vetor seguindo a estrutura de dados já descrita. O programa então começa a varrer vetor por vetor, e através de IF..THEN..ELSE testa qual deve ser a atitude tomada para acionar a plataforma do trilho através da dll. A cada atitude tomada pelo interpretador, um status on line é mostrado instantaneamente e também armazenado em um histórico para o usuário.

Outra característica deste interpretador é que ele já recebe do compilador todas as posições em coordenadas absolutas. Isto é, não é necessário se preocupar com G90, G91, G92 e G93 (somente para atualizar os status).

O interpretador realiza a interpretação dos comandos até encontrar um comando m02 que significa final de arquivo.

3.8.3 Loop PID

Controle PID

```
_export FAR PASCAL PID(long Kd, long Ki, long Kp, int TC, long M, long erro, long erroant, long s)
{
        V=floor((Vzero+Kp*erro+(Kd/(TC))*(erro-erroant)+Ki*s)+0.5);
        if (V>=(M*(Vmax-Vzero)+Vzero))
                {
                 V=floor(M*(Vmax-Vzero)+0.5)+Vzero;
                }
        if (V<=(Vzero-M*(Vzero-Vmin)))
                {
                 V=Vzero-floor(M*(Vzero-Vmin)+0.5);
                }
        outp(drivera,V);
        outp(driverb, 255-V);
        pos=inport(port);
        return(pos);

}
```

Na listagem acima pode ser vista a implementação da equação de diferenças de um controlador PID. Este controlador foi implementado em Borland C como uma função da DLL. Sua entrada é formada pelas constantes integrativa, derivativa e proporcional do PID, pelo período de

amostragem, pela porcentagem da velocidade máxima desejada para o posicionamento e pelos erros atual, anterior e a soma dos erros.

De posse destes valores a função calcula a tensão que deve ser enviada ao acionamento do motor. Caso esta tensão seja suficiente para gerar uma velocidade maior do que a programada, o valor da tensão enviado para o motor é limitado ao valor necessário para gerar a velocidade desejada.

Após esta verificação do valor da tensão, a função a escreve no endereço de I/O ocupado pela placa conversora D/A e, em seguida, lê a posição ocupada pela plataforma para retornar este valor para o Visual Basic.

A verificação do erro, isto é, se o loop PID deve ser repetido ou não, é feita pelo Visual Basic que executa a função de posicionamento, através de um timer. Este timer permanece habilitado enquanto a posição não for satisfatória. Este funcionamento do controlador pode ser representado através de um Rede de Petri:

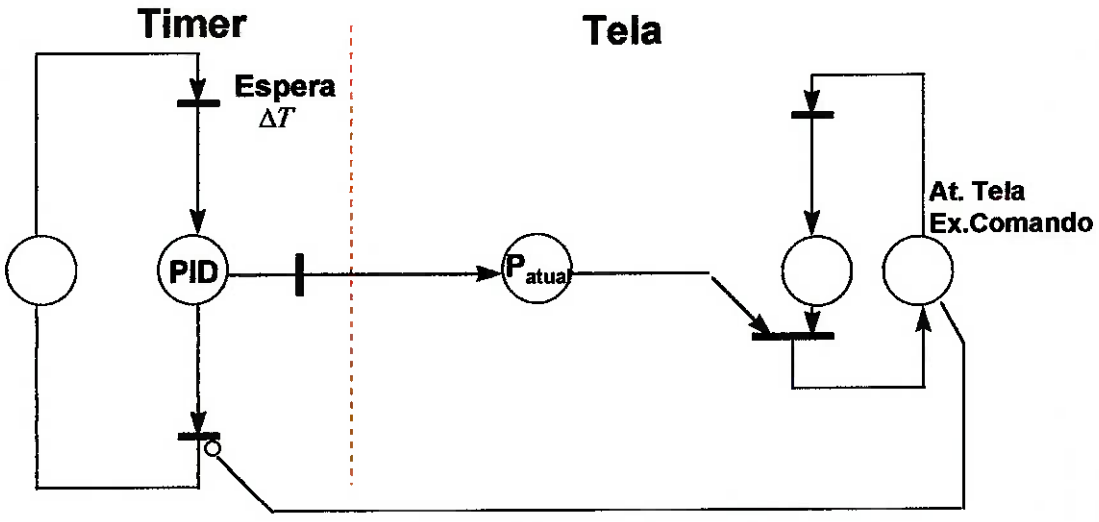


figura 13: Rede de Petri do controlador

Nesta Rede o timer é composto por uma transição temporizada que representa o período de amostragem. Após este período é feito o cálculo do PID e a próxima transição faz com que novo cálculo seja feito somente se esta estiver habilitada.

Esta transição é habilitada ou não pelo interpretador de Código G que ativa o timer caso a função a ser executada seja alguma função de posicionamento ou caso o *set-point* ainda não tenha sido atingido.

4. Atividades realizadas

Abaixo encontram-se os cronogramas individuais para cada integrante do grupo, onde são comparadas as atividades previstas e as efetivamente realizadas.

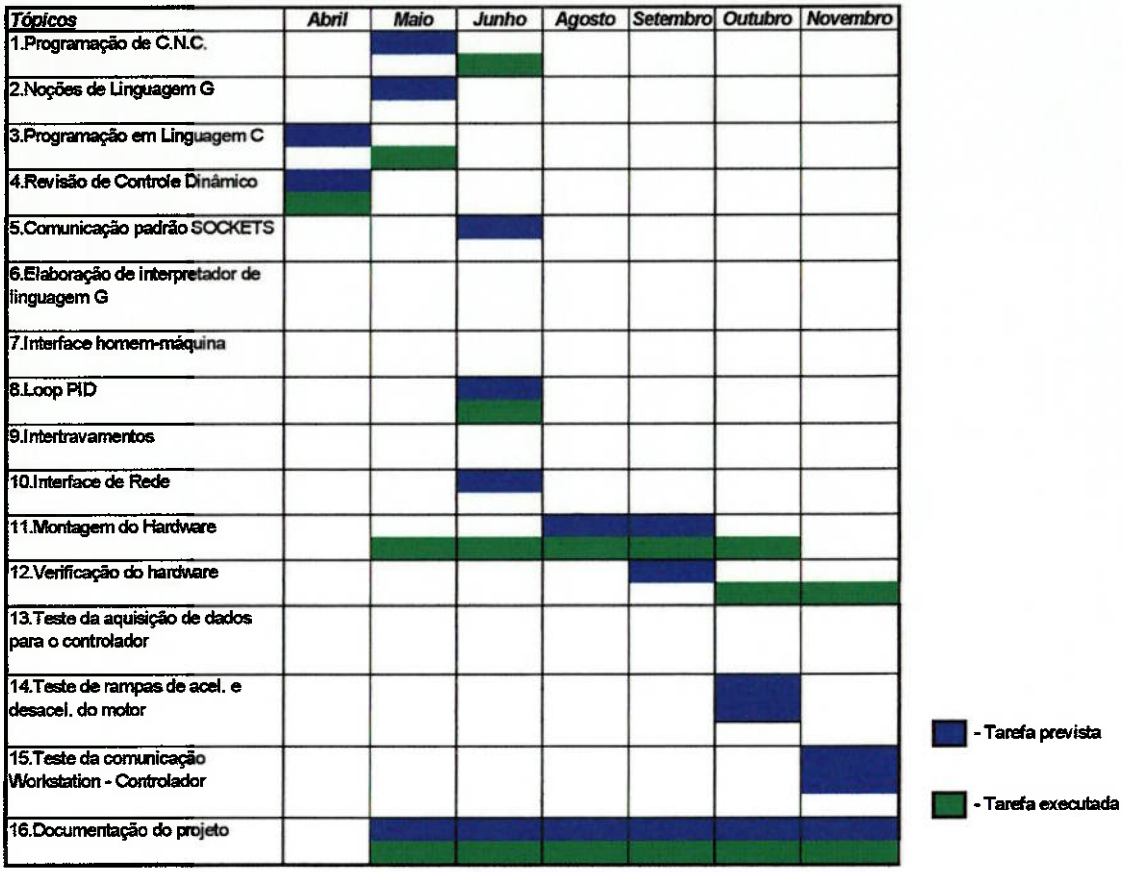


figura 16: cronograma para o aluno Vinícius Arouche de Toledo

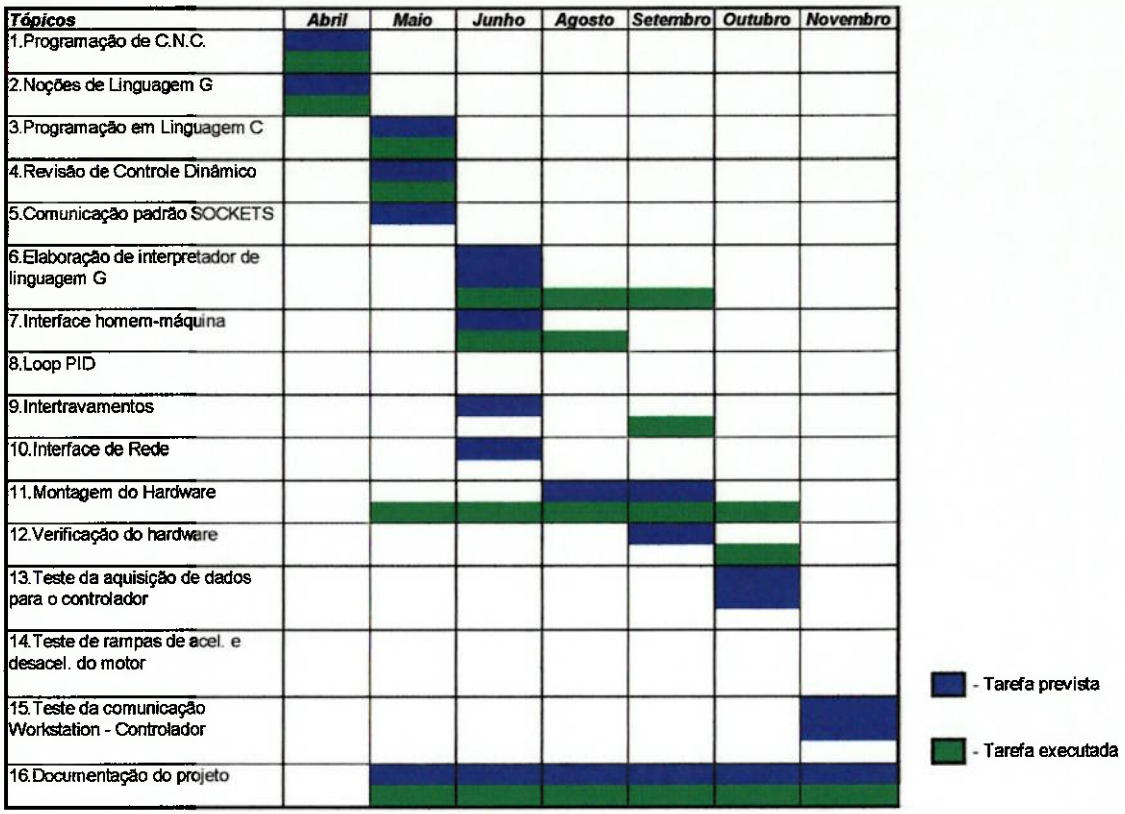


figura 17: cronograma para o aluno Jerri Regis Biscuola

As explicações sobre o que foi atingido e o que não foi atingido se encontram nos resultados.

5 - Resultados

A parte mecânica do trilho e seus sub-conjuntos foram totalmente implementados e estão perfeitamente funcionando, bem como o software de compilação e execução de comando em código g.

A parte do software podemos subdividir em dois grupos: modo manual e modo automático.

O modo manual está perfeitamente funcionando como previsto e não há problemas de nenhum tipo.

O modo automático está pronto, porém não conseguimos visualizar seus resultados já que problemas de ruídos elétricos na realimentação do sistema prejudicam nosso controlador. Esses ruídos foram minimizados 90% com um aterramento bem feito dos cabos de sinais, porém eles ainda existem e o mínimo de ruído presente nestes sinais já estragam o desempenho do controlador e dos sinais de fim de curso. Achamos que o ruído restante se deve à utilização de flat-cable que são muito sensíveis a este tipo de interferências, porém infelizmente não tivemos como trocar estes tipos de cabos por outros.

Além disto, a troca de um encoder de 400 pulsos/volta por outro de 5000 pulsos/volta, devido a problemas de curto-circuito no primeiro, não trouxe resultados satisfatórios pois, com este novo encoder, a freqüência com que os bit's de contagem menos significativos variavam tornou-se muito rápida a ponto de a placa de entrada digitais ignorar esta variação o que prejudicou a contagem proveniente da placa contadora (a placa de entrada digital mostrava uma contagem de 8 em 8). Outro problema verificado com a utilização deste encoder foi que, por ser a variação de pulsos muito rápida, quando o Borland C lia o byte menos significativo e, em seguida lia, o mais significativo, ás vezes a contagem variava entre estas duas leituras o que estragava o valor lido.

Nossa intenção era ainda levantar curvas de velocidades do sistema e dados sobre repetibilidade que são os resultados mais importantes neste tipo

de controlador de posição. Queríamos ainda otimizar os ganhos dos parâmetros do controlador PID, já que nosso software permite a fácil modificação destes parâmetros, mas a impossibilidade de visualização dos resultados do sistema funcionando também não permitiram esta otimização.

Devido à enorme quantidade de tempo despendido para resolvermos este problema de ruído, a parte de comunicação com uma estação de trabalho superior controladora da célula não foi implementada, mas como o software foi desenvolvido para funcionar no ambiente Windows, acreditamos que esta implementação não seja tão complicada, já que existem rotinas prontas neste ambiente utilizadas parta comunicação.

6. Bibliografia

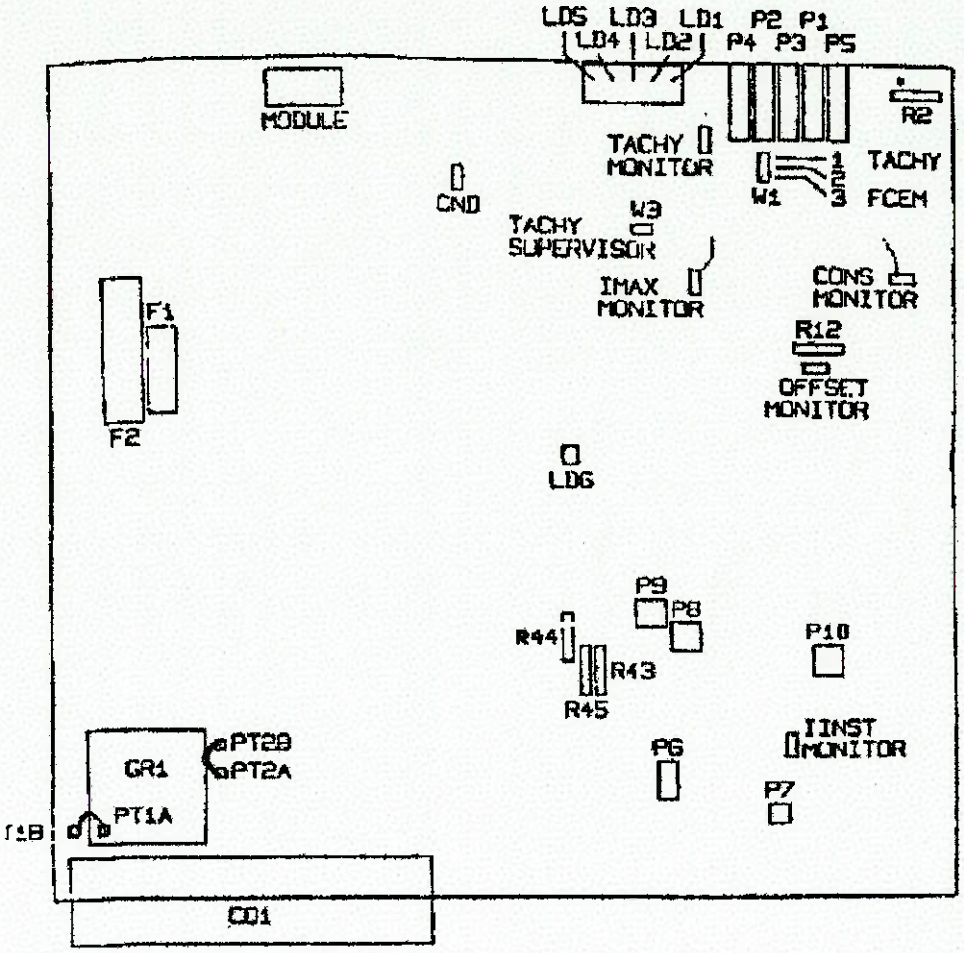
WEMMERLOV, U; HYER N.L. Cellular manufacturing in the U.S. industry: a survey of users. **International Journal of Production Research**, v.27, n.9, p.1511-1530, 1989.

MACHADO, A. **Comando Numérico aplicado às máquinas ferramenta.** 4 ed. São Paulo, Ícone Editora, 1990.

OGATA, K. **Modern Control Engineering.** 2 ed. New Jersey, Prentice- Hall, 1990.

7. Anexos

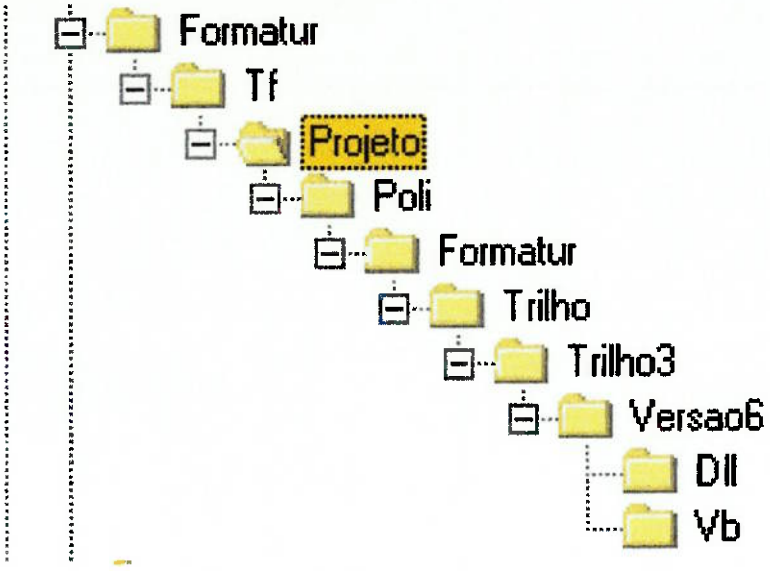
7.1 Esquema da placa de acionamento do servo-motor



7.2 Árvore de diretórios criada pelo arquivo compactado

O arquivo executável que se encontra anexado a este relatório contém, compactados, o arquivo fonte do software em Visual Basic e o

arquivo fonte da DLL implementada em Borland C 4.5. A árvore de diretórios criada pelo mesmo é a seguinte:



No diretório DLL encontra-se o arquivo hard_c.c que contém o código da Dll. Já no diretório Vb encontra-se o arquivo Trilho_c.mak que contém o projeto da interface homem-máquina em Visual Basic.

7.3 Listagens

```
ub caminho1_Change ()
ainform.file1.Pattern = mainform.caminho1.Text
nd Sub

ub caminho1_LostFocus ()
ainform.file1.Pattern = mainform.caminho1.Text
nd Sub

ub Dir1_Change ()
ile1.Path = dir1.Path

nd Sub

ub Drive1_Change ()
ir1.Path = drive1.Drive


nd Sub

ub exitdemo_Click ()
rquivoselecionado = ""
ide

nd Sub

ub File1_DblClick ()
f file1.listindex >= 0 Then
   arquivoselecionado = file1.Path
   If Right$(arquivoselecionado, 1) <> "\" Then
       arquivoselecionado = arquivoselecionado + "\"
   End If
   arquivoselecionado = arquivoselecionado + file1.list(file1.listindex)
   arquivosempath = file1.list(file1.listindex)
nd If
ide
nd Sub
```

```
RINCIPA.FRM - 1


ub ok_Click ()
oad execucao
xecucao.Show 1


nd Sub
```

```
ABALHO.FRM - 1


nction calcula_pos (absoluta As Integer, zero_abs As Integer, pos_anterior As
teger, nova_pos As Integer, novo_zero As Integer, novozeroant As Integer) As I
eger
   If (absoluta And zero_abs) Then
       calcula_pos = nova_pos
   Else
       If (absoluta And Not zero_abs) Then
           calcula_pos = nova_pos + novo_zero
       Else
           If (Not absoluta And zero_abs) Then
               calcula_pos = nova_pos + pos_anterior
           Else
               calcula_pos = nova_pos + pos_anterior + novo_zero - novozeroant
           End If
       End If
   End If
d Function

b Command1_Click ()




d Sub

b corigem_Click ()

m zero As Integer

l_manual = 128


d Sub

b posiciona (posicao As Integer, velocidade As Integer)

m vreal As Double
m epsilon As Double
m aux1 As Currency, zero As Integer

riodo = 10
silon = .5
rc = velocidade / 100
ro_anterior = 0
ro = motor(128)
ma_erro = 0
elocidade.Text = porc * velmax
etpoint.Text = posicao
f pos_geral < posicao Then
    GRAFICO1.Max = CInt(posicao)
    GRAFICO1.Min = CInt(pos_geral)
    GRAFICO2.Max = CInt(posicao - pos_geral)
lse
```

```
ABALHO.FRM - 2

   If pos_geral > posicao Then
        GRAFICO1.Max = CInt(pos_geral)
        GRAFICO1.Min = CInt(posicao)
        GRAFICO2.Max = CInt(pos_geral - posicao)
   End If

nd If


ostragem.Enabled = True
)

   DoEvents

op Until (Abs(erro_atual) <= epsilon) Or fimcurso() = 249 Or fimcurso() = 252

 fimcurso() = 249 Then
   cstatus.AddItem "Fim de curso da esquerda atingido !"
d If
 fimcurso() = 252 Then
   cstatus.AddItem "Fim de curso da direita atingido !"
d If

ostragem.Enabled = False
ro = motor(128)
x1 = lecontador1() - 34603008

s_geral = aux1 / f


d Sub

b Amostragem_Timer ()

m aux1 As Currency
m pos_aux As Integer

   aux1 = lecontador1() - 34603008
   pos_atual = aux1 / f
   erro_atual = posicao - pos_atual
   tvelocidade.Text = pid(kd, ki, kp, periodo, porc, erro_atual, erro_ant, soma
rro)
   aux1 = lecontador1() - 34603008
   pos_c = aux1 / f
   erro_anterior = erro_atual
   soma_erro = soma_erro + erro
   tposicao.Text = pos_c
   GRAFICO1.Value = pos_c
   GRAFICO2.Value = erro_atual

d Sub

b Automatico_Click ()

m zero As Integer
```

```
RABALHO.FRM - 3

ero = motor(128)
anual.Checked = False
utomatico.Checked = True
direita.Enabled = False
squerda.Enabled = False
para.Enabled = False

)
   zero = motor(131)
oop Until fimcurso() = 252 Or fimcurso() = 249 Or fimcurso() = 250
 fimcurso() <> 250 Then
   Do
       zero = motor(125)
   Loop Until fimcurso() = 250
id If
ero = motor(128)

ompilar.Enabled = True
xecutar.Enabled = True
rograma.BackColor = &H80000005
inha.BackColor = &H80000005
tatusonline.BackColor = &H80000005
etpoint.BackColor = &H80000005
elocidade.BackColor = &H80000005
celeracao.BackColor = &H80000005



id Sub

ub bcontinua_Click ()
ertou_continua = True
id Sub

ub cdireita_Click ()

ertou_esquerda = False
ertou_direita = True
ertou_pare = False

id Sub

ub cesquerda_Click ()

ertou_esquerda = True
ertou_direita = False
ertou_pare = False

id Sub

ub Compilar_click ()

m linha_de_comando As String * 80
m i As Integer, snumaux As String * 4, inumaux As Integer
m x_ant As Integer, x_aux As Integer, n_ant As Integer, carac As String
```

```
ABALHO.FRM - 4

m absoluta As Integer, zerosist As Integer, valorzero As Integer, zeroant As I
eger
m ultimog As Integer, letra As Integer
m first_c As String

lf = Chr$(13) + Chr$(10)
 Error GoTo trataerro

larmes.Clear
larmes.Text = ""

ant = 0        'Inicializa variáveis
ro = 0
 = 0
 = 0
aux = 0
ant = 0
soluta = True
rosist = True
arq = False
lorzero = 0
roant = 0
timog = 9

 arquivoselecionado = "" Then    'Se não houver arquivo selecionado exibe mens
em de erro
  MsgBox "Selecione o arquivo primeiro !", 48, "Mensagem de Arquivo"
  noarq = True
  Exit Sub
d If

en arquivoselecionado For Input As #1 'Abre arquivo em disco

 While Not EOF(1) 'Enquanto não termina arquivo
   letra = 1   'Inicializa ponteiro de linha
   Line Input #1, lincom 'Lê uma linha
   lincom = lincom + crlf  'Coloca caracter de fim de linha
   i = i + 1
   programa(i).n = 0   'Inicializa elemento do vetor
   programa(i).g = 99
   programa(i).x = 0
   programa(i).f = 0
   programa(i).m = 99
   carac = Mid(lincom, letra, 1)   'Lê 1° caracter

   If (carac <> "n") Then  'Verifica se o caracter é "n"
       erro = erro + 1
       calarmes.AddItem "Erro: Toda linha deve iniciar com n. "
   Else
       letra = letra + 1   'Atualiza ponteiro
       snumaux = Mid(lincom, letra, 4) 'Lê mais 4 dígitos
       inumaux = CInt(snumaux)
       programa(i).n = inumaux 'Atualiza vetor
       tlinha.Text = lincom
       If (programa(i).n <= n_ant) Then    'Verifica se o número está em ordem
escente
```

```
            erro = erro + 1
            calarmes.AddItem "Erro: Seqüencia n errada na linha " + CStr(program
(i).n)
        Else
            n_ant = programa(i).n
        End If
    End If
    letra = letra + 4    'Atualiza ponteiro
    carac = Mid(lincom, letra, 2)    'Lê mais dois dígitos
    If (carac = crlf) Then  'Verifica se a linha já terminou
        erro = erro + 1
        calarmes.AddItem "Erro: Seqüencia n sozinha na linha " + CStr(programa(i
n)
    Else
        carac = Mid(lincom, letra, 1)    'Lê mais um dígito
        If (carac <> " ") Then  'Verifica se há espaço entre os comandos
            erro = erro + 1
            calarmes.AddItem "Erro: Espaços para separação requeridos na linha "
 CStr(programa(i).n)
        Else
            letra = letra + 1    'Atualiza ponteiro
            carac = Mid(lincom, letra, 1)    'Lê um dígito
            first_c = carac
            If (first_c = "g") Then 'Se 1° comando é "G"
                letra = letra + 1    'Atualiza ponteiro
                snumaux = Mid(lincom, letra, 2) 'Lê dois dígitos
                inumaux = CInt(snumaux)
                programa(i).g = inumaux 'Atualiza vetor
                If (programa(i).g = 0) Then 'Se for G00
                    letra = letra + 2    'Atualiza ponteiro
                    carac = Mid(lincom, letra, 1)    'Lê um dígito
                    If (carac <> " ") Then  'Verifica se há espaço
                        erro = erro + 1
                        calarmes.AddItem "Erro: Espaços para separação requerido
na linha " + CStr(programa(i).n)
                    Else
                        ultimog = 0      'Guarda comando G programado
                        letra = letra + 1    'Atualiza ponteiro
                        carac = Mid(lincom, letra, 1)    'Lê um dígito
                        If (carac = "x") Then    'Se for X
                            letra = letra + 1    'Atualiza ponteiro
                            carac = Mid(lincom, letra, 1)    'Lê um dígito
                            If (carac = "-") Then    'Se o n° for negativo
                                letra = letra + 1    'Atualiza ponteiro
                                snumaux = Mid(lincom, letra, 4) 'Lê número
                                inumaux = CInt(snumaux)
                                x_aux = (-1) * Abs(inumaux)

                            Else
                                snumaux = Mid(lincom, letra, 4)
                                inumaux = CInt(snumaux)
                                x_aux = inumaux

                            End If
                            programa(i).x = calcula_pos(absoluta, zerosist, x_an
 x_aux, valorzero, zeroant) 'Calcula coordenada absoluta e atualiza o vetor
```

```
                            zeroant = valorzero     'Guarda coordenadas de posiç
)
                            x_ant = programa(i).x
                            x_aux = 0
                            letra = letra + 4   'Atualiza ponteiro
                            carac = Mid(lincom, letra, 2)   'Lê mais dois dígito

                            If (carac <> crlf) Then     'Testa fim de linha
                                erro = erro + 1
                                calarmes.AddItem "Erro: Não pode haver mais coma
dos após g00 na linha " + CStr(programa(i).n)
                            End If
                        Else
                            erro = erro + 1
                            calarmes.AddItem "Erro: g00 requer comando x na linh
" + CStr(programa(i).n)
                        End If
                    End If
                End If
                If (programa(i).g = 1) Then 'Para G01 o código é semelhante a G0
 só detalhes serão comentados
                    letra = letra + 2
                    carac = Mid(lincom, letra, 1)
                    If (carac <> " ") Then
                        erro = erro + 1
                        calarmes.AddItem "Erro: Espaços para separação requerido
na linha " + CStr(programa(i).n)
                    Else
                        ultimog = 1
                        letra = letra + 1
                        carac = Mid(lincom, letra, 1)
                        If (carac = "x") Then
                            letra = letra + 1
                            carac = Mid(lincom, letra, 1)
                            If (carac = "-") Then
                                letra = letra + 1
                                snumaux = Mid(lincom, letra, 4)
                                inumaux = CInt(snumaux)
                                x_aux = (-1) * Abs(inumaux)

                            Else
                                snumaux = Mid(lincom, letra, 4)
                                inumaux = CInt(snumaux)
                                x_aux = inumaux

                            End If
                            programa(i).x = calcula_pos(absoluta, zerosist, x_an
x_aux, valorzero, zeroant)
                            zeroant = valorzero
                            x_ant = programa(i).x
                            x_aux = 0
                            letra = letra + 4
                            carac = Mid(lincom, letra, 2)
                                If (carac <> crlf And Left(carac, 1) <> " ") The

                                    erro = erro + 1
```

```
                                    calarmes.AddItem "Erro: Espaços para separaç
 requeridos na linha " + CStr(programa(i).n)
                                End If
                                If (Right(carac, 1) <> "f" And carac <> crlf) Th
t
                                    erro = erro + 1 'Verifica se comando após x
 f
                                    calarmes.AddItem "Erro: Caracter após o coma
lo x não é válido na linha " + CStr(programa(i).n)
                                End If
                                If (Right(carac, 1) = "f") Then
                                    letra = letra + 2
                                    snumaux = Mid(lincom, letra, 3) 'Lê velocida
 e guarda seu valor
                                    inumaux = CInt(snumaux)
                                    programa(i).f = inumaux
                                    fgeral = True
                                    fvalor = programa(i).f
                                    If (programa(i).f) < 0 Then 'Testa velocidad

                                        erro = erro + 1
                                        letra = letra + 1
                                        calarmes.AddItem "Erro: Comando f negati
 impossível na linha " + CStr(programa(i).n)
                                    End If
                                    letra = letra + 3
                                    carac = Mid(lincom, letra, 2)
                                    If (carac <> crlf) Then
                                        erro = erro + 1
                                        calarmes.AddItem "Erro: Não pode haver m
s comandos após g01 na linha " + CStr(programa(i).n)
                                    End If
                                Else
                                    If (carac = crlf And Not fgeral) Then   'Ver
ica se algum f já foi programado
                                        erro = erro + 1
                                        calarmes.AddItem "Erro: Comando f não de
nido na linha " + CStr(programa(i).n)
                                    Else
                                        If (carac = crlf) Then
                                            programa(i).f = fvalor  'Atualiza ve
r com valor de f
                                        Else
                                            erro = erro + 1
                                            calarmes.AddItem "Erro: Não pode hav
 mais comandos na linha " + CStr(programa(i).n)
                                        End If
                                    End If
                                End If
                            Else
                                erro = erro + 1
                                calarmes.AddItem "Erro: g01 sem comando x na linha "
 CStr(programa(i).n)
                            End If
                        End If
                    End If
```

```
            If (programa(i).g = 4) Then     'Se comando G for 04
                letra = letra + 2
                carac = Mid(lincom, letra, 1)
                If (carac <> " ") Then
                    erro = erro + 1
                    calarmes.AddItem "Erro: Espaços para separação requerido
na linha " + CStr(programa(i).n)
                Else
                  letra = letra + 1
                  carac = Mid(lincom, letra, 1)
                  If (carac = "x") Then     'Verifica se comando é X
                    letra = letra + 1
                    carac = Mid(lincom, letra, 1)
                    If (carac = "-") Then
                            erro = erro + 1
                            calarmes.AddItem "Erro: Caracter inválido após g
 na linha " + CStr(programa(i).n)
                    Else
                            snumaux = Mid(lincom, letra, 4) 'Lê dígitos do c
ando X
                            inumaux = CInt(snumaux)
                            programa(i).x = inumaux 'Atualiza valor
                            letra = letra + 4
                            carac = Mid(lincom, letra, 2)
                            If (carac <> crlf) Then
                                erro = erro + 1
                                calarmes.AddItem "Erro: Não pode haver mais
mandos após g04 na linha " + CStr(programa(i).n)
                            End If
                    End If
                  Else
                    erro = erro + 1
                    calarmes.AddItem "Erro: g04 sem comando x (tempo) na lin
 " + CStr(programa(i).n)
                  End If
                End If
            End If
                If (programa(i).g = 90) Then    'Se for G 90
                    letra = letra + 2
                    carac = Mid(lincom, letra, 2)
                    If (carac <> crlf) Then     'Verifica se é final de linh

                        erro = erro + 1
                        calarmes.AddItem "Erro: Não pode haver mais comandos
pós g90 na linha " + CStr(programa(i).n)
                    Else
                        absoluta = True     'Aciona flag para coordenada abs
uta
                    End If
                End If
                If (programa(i).g = 91) Then    'Se for G91
                    letra = letra + 2
                    carac = Mid(lincom, letra, 2)
                    If (carac <> crlf) Then 'Verifica se é final de linha
                        erro = erro + 1
                        calarmes.AddItem "Erro: Não pode haver mais comandos
```

```
pós g91 na linha " + CStr(programa(i).n)
                    Else
                        absoluta = False    'Desliga flag para coordenada ab
luta,
                    End If                  'portanto a coordenada passa par
incremental
                End If
                If (programa(i).g = 92) Then    'Se for G92
                    letra = letra + 2
                    carac = Mid(lincom, letra, 1)
                    If (carac <> " ") Then
                        erro = erro + 1
                        calarmes.AddItem "Erro: Espaços para separação reque
dos na linha " + CStr(programa(i).n)
                    Else
                        letra = letra + 1
                        carac = Mid(lincom, letra, 1)
                        If (carac = "x") Then   'Lê comando X
                            letra = letra + 1
                            carac = Mid(lincom, letra, 1)
                            If (carac = "-") Then   'Lê n° negativo
                                letra = letra + 1
                                snumaux = Mid(lincom, letra, 4)
                                inumaux = CInt(snumaux)
                                programa(i).x = (-1) * Abs(inumaux) 'Atualiz
vetor

                            Else
                                snumaux = Mid(lincom, letra, 4) 'Lê n° posit
ro
                                inumaux = CInt(snumaux)
                                programa(i).x = inumaux 'Atualiza vetor

                        End If
                        zerosist = False    'Flag indica zero deslocado
                        zeroant = valorzero
                        valorzero = programa(i).x   'Guarda novo zero
                        letra = letra + 4
                        carac = Mid(lincom, letra, 2)
                        If (carac <> crlf) Then 'Verifica fim de linha
                            erro = erro + 1
                            calarmes.AddItem "Erro: Não pode haver mais coma
los após g92 na linha " + CStr(programa(i).n)
                        End If

                        Else
                            erro = erro + 1
                            calarmes.AddItem "Erro: g92 requer comando x na
nha " + CStr(programa(i).n)
                        End If
                    End If
                End If
                If (programa(i).g = 93) Then    'Se for G93
                    letra = letra + 2
                    carac = Mid(lincom, letra, 2)
                    If (carac <> crlf) Then
```

```
                            erro = erro + 1
                            calarmes.AddItem "Erro: Não pode haver mais comandos
pós g93 na linha " + CStr(programa(i).n)
                        Else
                            zerosist = True 'Desliga zero deslocado e volta zero
ara centro do trilho
                            valorzero = 0
                            zeroant = 0
                        End If
                    End If
                    If (programa(i).g <> 0 And programa(i).g <> 1 And programa(i
g <> 4 And programa(i).g <> 90 And programa(i).g <> 91 And programa(i).g <> 92
nd programa(i).g <> 93) Then
                        erro = erro + 1     'Verifica se comando G é válido
                        calarmes.AddItem "Erro: Comando g inválido na linha " +
tr(programa(i).n)
                    End If
            End If
            If (first_c = "x") Then     'Se 1° comando é X
                If (ultimog = 9) Then   'Verifica se G00 ou G01 já foram program
los
                    erro = erro + 1
                    calarmes.AddItem "Erro: g01 ou g00 não definidos na linha" +
Str(programa(i).n)
                Else
                    If (ultimog = 1) Then      'Atualiza vetor com último G prog
mado
                        programa(i).g = 1
                    Else
                        programa(i).g = 0
                    End If
                    letra = letra + 1
                    carac = Mid(lincom, letra, 1)
                    If (carac = "-") Then                    'Lê n° do comando X
                        letra = letra + 1
                        snumaux = Mid(lincom, letra, 4)
                        inumaux = CInt(snumaux)
                        x_aux = (-1) * Abs(inumaux)
                    Else
                        snumaux = Mid(lincom, letra, 4)
                        inumaux = CInt(snumaux)
                        x_aux = inumaux

                    End If
                    programa(i).x = calcula_pos(absoluta, zerosist, x_ant, x_aux
valorzero, zeroant)  'Calcula posição absoluta
                    zeroant = valorzero
                    x_ant = programa(i).x
                    x_aux = 0
                    letra = letra + 4
                    carac = Mid(lincom, letra, 2)
                    If (ultimog = 0 And carac <> crlf) Then
                        erro = erro + 1
                        calarmes.AddItem "Erro: Caracter inválido ou g00 não req
r f na linha" + CStr(programa(i).n)
                    End If
```

```
                If (ultimog = 1) Then
                    If (carac <> crlf And Left(carac, 1) <> " ") Then
                        erro = erro + 1
                        calarmes.AddItem "Erro: Espaços para separação reque
.dos na linha " + CStr(programa(i).n)
                    End If
                    If (Right(carac, 1) <> "f" And carac <> crlf) Then
                        erro = erro + 1
                        calarmes.AddItem "Erro: Caracter inválido na linha"
 CStr(programa(i).n)
                    End If
                    If (carac = crlf And Not fgeral) Then   'Verifica se com
ıdo f já foi programado
                        erro = erro + 1
                        calarmes.AddItem "Erro: g01 precisa de f definido na
.inha" + CStr(programa(i).n)
                    End If
                    If (Right(carac, 1) = "f") Then 'Leitura de velocidade (
›mando F)
                        letra = letra + 1
                        snumaux = Mid(lincom, letra, 3)
                        inumaux = CInt(snumaux)
                        programa(i).f = inumaux
                        fgeral = True
                        fvalor = programa(i).f
                        If (programa(i).f) < 0 Then
                            erro = erro + 1
                            letra = letra + 1
                            calarmes.AddItem "Erro: Comando f negativo impos
.vel na linha " + CStr(programa(i).n)
                        End If
                        letra = letra + 3
                        carac = Mid(lincom, letra, 2)
                        If (carac <> crlf) Then
                            erro = erro + 1
                            calarmes.AddItem "Erro: Não pode haver mais coma
los após f na linha " + CStr(programa(i).n)
                        End If
                    End If
                    If (carac = crlf And fgeral) Then
                        programa(i).f = fvalor
                    End If
                End If
            End If
        End If
        If (first_c = "f") Then     'Se 1° comando é F
            letra = letra + 1
            snumaux = Mid(lincom, letra, 3) 'Lê dígitos
            inumaux = CInt(snumaux)     'Atualiza vetor
            programa(i).f = inumaux     'Guarda valor de f
            fgeral = True
            fvalor = programa(i).f
            If (programa(i).f) < 0 Then     'Testa f
               erro = erro + 1
               letra = letra + 1
               calarmes.AddItem "Erro: Comando f negativo impossível na linh
```

```
" + CStr(programa(i).n)
                End If
                letra = letra + 3
                carac = Mid(lincom, letra, 2)
                If (carac <> crlf) Then
                    erro = erro + 1
                    calarmes.AddItem "Erro: Não pode haver mais comandos após f
 linha " + CStr(programa(i).n)
                End If
            End If
            If (first_c = "m") Then      'Se 1° comando é m
                letra = letra + 1
                snumaux = Mid(lincom, letra, 2)      'Lê comando M
                inumaux = CInt(snumaux)
                programa(i).m = inumaux      'Atualiza vetor
                If (programa(i).m = 3) Then 'Se for m03
                    letra = letra + 2
                    carac = Mid(lincom, letra, 1)
                    If (carac <> " ") Then
                        erro = erro + 1
                        calarmes.AddItem "Erro: Espaços para separação requerido
na linha " + CStr(programa(i).n)
                    Else
                      letra = letra + 1
                      carac = Mid(lincom, letra, 1)
                      If (carac = "x") Then      'Lê comando X
                        letra = letra + 1
                        carac = Mid(lincom, letra, 1)
                        If (carac = "-") Then
                                erro = erro + 1
                                calarmes.AddItem "Erro: Caracter inválido após m
 na linha " + CStr(programa(i).n)
                        Else
                                snumaux = Mid(lincom, letra, 4)
                                inumaux = CInt(snumaux)
                                programa(i).x = inumaux 'Atualiza vetor
                                letra = letra + 4
                                carac = Mid(lincom, letra, 2)
                                If (carac <> crlf) Then 'Testa fim de linha
                                    erro = erro + 1
                                    calarmes.AddItem "Erro: Não pode haver mais
mandos após m03 na linha " + CStr(programa(i).n)
                                End If
                        End If
                      Else
                        erro = erro + 1
                        calarmes.AddItem "Erro: m03 sem comando x (linha) na lin
 " + CStr(programa(i).n)
                        End If
                    End If
                End If
                If (programa(i).m = 2 Or programa(i).m = 0) Then     'Se for m02
 m00
                    letra = letra + 2
                    carac = Mid(lincom, letra, 2)
                    If (carac <> crlf) Then 'Testa fim de linha
```

```
                        erro = erro + 1
                        calarmes.AddItem "Erro: Não pode haver mais comandos após
 m00 ou m02 na linha " + CStr(programa(i).n)

                    End If
                End If
                If (programa(i).m <> 0 And programa(i).m <> 2 And programa(i).m
> 3) Then  'Verifica se comando m é válido
                    erro = erro + 1
                    calarmes.AddItem "Erro: Comando m inválido na linha" + CStr(
:ograma(i).n)
                End If
            End If
        End If
    End If

)op

: (programa(i).m <> 2) Then    'Verifica se foi colocado m02 no fim do programa
    erro = erro + 1
    calarmes.AddItem "Erro: Programa requer m02 (END) no final"
ıd If

: erro > 0 Then    'Se houver erro de compilação exibe mensagem
    calarmes.Text = "Há " + CStr(erro) + " erro(s) de compilação !"
.se
    calarmes.Text = "Compilação completada com sucesso !"
ıd If

.inha.Text = "Final"
;tatusonline.Text = "Compilação terminada !"
.ose #1    'Fecha arquivo em disco


:it Sub

:ataerro:
  erro = erro + 1
  calarmes.AddItem "Erro: Caracter inválido na linha " + CStr(programa(i).n)
  inumaux = 0
  Resume Next

ıd Sub

ıb cpara_Click ()

)ertou_esquerda = False
)ertou_direita = False
)ertou_pare = True

ıd Sub

ıb Executar_Click ()

.m i As Integer                                    'Indice do vetor
.m tempoinicial As Variant, tempofinal As Variant
```

```
im tempodecorrido As Variant                    'Diferenca entre t final e
 inicial
im fim As Integer

 = 1                          'Inicializa indice
im = False
empoinicial = ""
empofinal = ""
empodecorrido = ""
os_geral = 0                  'Inicializa variaveis

ompilar_click                 'Compila primeiro

status.Clear
f (erro > 0) Then             'Se nao houver erro de compilacao
   tstatusonline.Text = "Há erro(s) de compilação!     "

lse
f Not noarq Then              'Se existir arquivo
  Do While (programa(i).m <> 2)     'Faca ate encontrar final do programa (m02)
  apertou_continua = False
  If programa(i).g = 0 Then         'Se for g00 posiciona com veloc. maxima
       pos = programa(i).x
       vel = 100
       kp = CCur(tproporcional.Text)
       ki = CCur(tintegrativa.Text)
       kd = CCur(tderivativa.Text)
       tstatusonline.Text = "POSICIONANDO"  'Atualiza status
       tlinha.Text = programa(i).n
       cstatus.AddItem CStr(programa(i).n) + ": Posicionamento para x=" + CStr(
ograma(i).x) + " com velocidade máxima."
       Call posiciona(pos, vel)
  End If
  If programa(i).g = 1 Then              'Se for g01 posiciona com velocidade
       pos = programa(i).x               'programavel
       vel = programa(i).f
       kp = CCur(tproporcional.Text)
       ki = CCur(tintegrativa.Text)
       kd = CCur(tderivativa.Text)
       tstatusonline.Text = "POSICIONANDO"
       tlinha.Text = programa(i).n
       cstatus.AddItem CStr(programa(i).n) + ": Posicionamento para x=" + CStr(
ograma(i).x) + " com " + CStr(programa(i).f) + "% da velocidade máxima."
       Call posiciona(pos, vel)
  End If
  If programa(i).g = 4 Then             'Se for g04 aguarda um determinado temp

       tempoinicial = Now
       tempodecorrido = ""
       tstatusonline.Text = "AGUARDANDO"
       tlinha.Text = programa(i).n
       cstatus.AddItem CStr(programa(i).n) + ": Aguardando " + CStr(programa(i)
) + " segundos."
       Do While tempodecorrido < CVar(programa(i).x)
           tempofinal = Now
           tempodecorrido = tempofinal - tempoinicial     'Calcula tempo
```

```
      Loop
 End If
 If programa(i).g = 90 Then
      tlinha.Text = programa(i).n
      tstatusonline.Text = "Coordenada absoluta"
      cstatus.AddItem CStr(programa(i).n) + ": Mudança para coordenada absolut
"
 End If
 If programa(i).g = 91 Then
      tlinha.Text = programa(i).n
      tstatusonline.Text = "Coordenada incremental"
      cstatus.AddItem CStr(programa(i).n) + ": Mudança para coordenada increme
al."
 End If
 If programa(i).g = 92 Then
      tlinha.Text = programa(i).n
      tstatusonline.Text = "Zero do sistema mudado"
      cstatus.AddItem CStr(programa(i).n) + ": Mudança do zero do sistema para
=" + CStr(programa(i).x) + "."
 End If
 If programa(i).g = 93 Then
      tlinha.Text = programa(i).n
      tstatusonline.Text = "Zera sistema trilho"
      cstatus.AddItem CStr(programa(i).n) + ": Retorno para zero absoluto do t
lho."
 End If

      '  g90 g91 g92 g93  nao interessam para o interpretador pois todos os
      '  valores vem em coordenadas absolutas. Eles so sao utilizados para
      '  atualizar status

 If programa(i).m = 0 Then              'Se for m00 aguarda tecla continua
      tlinha.Text = programa(i).n
      tstatusonline.Text = "Pressione Continua"
      cstatus.AddItem CStr(programa(i).n) + ": Aguardando tecla Continua."
      bcontinua.Enabled = True
      Do
          DoEvents
      Loop Until apertou_continua
      bcontinua.Enabled = False
 End If
 If programa(i).m = 3 Then           'Se dor m03 pula para linha especificada
      i_aux = 1
      fim = False
      Do While programa(i_aux).n <> programa(i).x And Not fim
          i_aux = i_aux + 1            'Procura linha
          If (programa(i_aux).m = 2 And programa(i_aux).n <> programa(i).x) Th

              fim = True
          End If
      Loop
      If fim Then                       'Se linha nao existir para programa
          tstatusonline.Text = "GO TO Inválido"
          cstatus.AddItem CStr(programa(i).n) + ": Comando de desvio para linh
" + CStr(programa(i).x) + " errado. Linha inexistente."
          Exit Do
```

```
      Else
          tstatusonline.Text = "Desvio de programa"
          cstatus.AddItem CStr(programa(i).n) + ": Programa desviado para linh
" + CStr(programa(i).x) + "."
          i = i_aux - 1         'Desvio da linha
      End If
 End If
 i = i + 1        'Soma indice (proximo comando)
 Loop
 tstatusonline.Text = "Programa terminado !"
 cstatus.Text = "Programa finalizado !"
 cstatus.AddItem "Programa finalizado !"
 tsetpoint.Text = ""                          'Atualiza tela final
 tlinha.Text = ""
 tvelocidade.Text = "0"

nd If
d If
d Sub


b Manual_Click ()

m zero As Integer, Vmin As Integer, Vmax As Integer, Vzero As Integer
m aux1 As Currency
m fimaux As Integer, posicao As Double

ertou_pare = True
ertou_esquerda = False
ertou_direita = False
nual.Checked = True
tomatico.Checked = False
l_manual = 128
ro = motor(vel_manual)
= 1
ero = 128
in = 0
ax = 255

mpilar.Enabled = False
ecutar.Enabled = False
rograma.BackColor = &HC0C0C0
inha.BackColor = &HC0C0C0
tatusonline.BackColor = &HC0C0C0
etpoint.BackColor = &HC0C0C0
elocidade.BackColor = &HC0C0C0
celeracao.BackColor = &HC0C0C0
squerda.Enabled = True
ireita.Enabled = True
ara.Enabled = True



  DoEvents
  fimaux = fimcurso()
```

```
  If (fimaux = 249 And vel_manual < 128) Or (fimaux = 252 And vel_manual > 128
Then
      vel_manual = 128
      If fimaux = 249 Then
          cstatus.AddItem "Cuidado : Fim de curso Esquerdo atingido !!!"
          cstatus.Text = "Cuidado : Fim de curso Esquerdo atingido !!!"
      End If
      If fimaux = 252 Then
          cstatus.AddItem "Cuidado : Fim de curso Direito atingido !!!"
          cstatus.Text = "Cuidado : Fim de curso Direito atingido !!!"
      End If


  End If
  If apertou_esquerda And vel_manual > Vzero - k * (Vzero - Vmin) And fimaux <
249 Then
      vel_manual = vel_manual - 1
      apertou_esquerda = False
      cstatus.Text = ""
  End If

  If apertou_direita And vel_manual < (k * (Vmax - Vzero) + Vzero) And fimaux
 252 Then
      vel_manual = vel_manual + 1
      apertou_direita = False
      cstatus.Text = ""
  End If
  If apertou_pare Then
      vel_manual = 128
      cstatus.Text = ""
  End If
  zero = motor(vel_manual)
  aux1 = lecontador1() - 34603008
  posicao = aux1 / f
  tposicao.Text = CStr(posicao)



op Until automatico.Checked

d Sub

b Sair_Click ()
d
d Sub

b Selecionar_Click ()
quivoselecionado = openfiledlg()
rograma.Text = arquivosempath
inha.Text = ""
tatusonline.Text = ""
larmes.Clear
larmes.Text = ""

d Sub
```

```
DULE1.BAS - 1

pe linha            'Estrutura de dados: Vetor de records
   n As Integer     'Comando N
   g As Integer     'Comando G
   x As Integer     'Comando X
   f As Integer     'Comando F
   m As Integer     'Comando M
d Type

nst velmax = 50 'Este valor não é válido pois não pudemos fazer o posicionamen


obal arquivoselecionado As String
obal arquivosempath  As String
obal programa(1 To 2000) As linha 'Vetor de records
obal noarq As Integer
obal erro As Integer
obal apertou_continua As Integer
obal erro_atual As Double, erro_anterior As Double 'Variáveis para erros de po
cionamento
obal pos_atual As Double, soma_erro As Double, pos_c As Double
obal pos_geral As Double, porc As Double
obal kp As Double, kd As Double, ki As Double
obal vel As Integer, pos As Integer
obal vel_manual As Integer
obal k As Integer 'Porcentagem da velocidade utilizada no modo manual
obal apertou_direita As Integer
obal apertou_esquerda As Integer
obal apertou_pare As Integer

hamada das funções na DLL
clare Function pid Lib "c:\formatur\TF\PROJETO\poli\formatur\trilho\trilho3\ve
ao6\dll\hard_c.dll" (ByVal kd As Long, ByVal ki As Long, ByVal kp As Long, ByV
. periodo As Integer, ByVal porc As Long, ByVal erro_atual As Long, ByVal erro_
t As Long, ByVal soma_erro As Long) As Long
clare Function motor Lib "c:\formatur\TF\PROJETO\poli\formatur\trilho\trilho3\
rsao6\dll\hard_c.dll" (ByVal valor_tensao As Integer) As Integer
clare Function lecontador1 Lib "c:\formatur\TF\PROJETO\poli\formatur\trilho\tr
.ho3\versao6\dll\hard_c.dll" () As Long
clare Function lecontador2 Lib "c:\formatur\TF\PROJETO\poli\formatur\trilho\tr
.ho3\versao6\dll\hard_c.dll" () As Integer
clare Function fimcurso Lib "c:\formatur\TF\PROJETO\poli\formatur\trilho\trilh
\versao6\dll\hard_c.dll" () As Integer

obal Const f = 22.8  'Fator de conversão: 100mm = 2280 pulsos

nction openfiledlg () As String
ad mainform
inform.File1.Pattern = mainform.caminho1.Text
quivoselecionado = ""
quivosempath = ""
inform.Show 1
enfiledlg = arquivoselecionado

d Function
```

```
.INFORM.FRM - 1

RSION 2.00
gin Form Mainform
 Caption         =   "Selecionar"
 ClientHeight    =   3555
 ClientLeft      =   1485
 ClientTop       =   1965
 ClientWidth     =   5670
 Height          =   4020
 Left            =   1425
 LinkTopic       =   "Form1"
 ScaleHeight     =   3555
 ScaleWidth      =   5670
 Top             =   1560
 Width           =   5790
 Begin CommandButton exitdemo
    Caption          =   "Sair"
    Height           =   735
    Left             =   4440
    TabIndex         =   5
    Top              =   2400
    Width            =   975
 End
 Begin TextBox caminho1
    Height           =   375
    Left             =   3120
    TabIndex         =   3
    Text             =   "*.G"
    Top              =   2760
    Width            =   1215
 End
 Begin FileListBox File1
    Height           =   2175
    Left             =   3120
    TabIndex         =   2
    Top              =   120
    Width            =   2295
 End
 Begin DirListBox Dir1
    Height           =   2730
    Left             =   360
    TabIndex         =   1
    Top              =   480
    Width            =   2415
 End
 Begin DriveListBox Drive1
    Height           =   315
    Left             =   360
    TabIndex         =   0
    Top              =   120
    Width            =   2415
 End
 Begin Label Label1
    BorderStyle      =   1  'Fixed Single
    Caption          =   "Arquivos"
    Height           =   375
    Left             =   3120
```

```
.INFORM.FRM - 2

    TabIndex            =   4
    Top                 =   2400
    Width               =   1215
 End
d
```

```
INCIPA.FRM - 1

RSION 2.00
gin Form Principal
  BackColor       =   &H00FF8080&
  Caption         =   "Sistema D.N.C. didático para trilho automático"
  ClientHeight    =   4020
  ClientLeft      =   2115
  ClientTop       =   1545
  ClientWidth     =   6345
  Height          =   4485
  Left            =   2055
  LinkTopic       =   "Principal"
  ScaleHeight     =   4020
  ScaleWidth      =   6345
  Top             =   1140
  Width           =   6465
  WindowState     =   2  'Maximized
  Begin SSPanel entrada
     BackColor       =   &H00C0C0C0&
     FloodColor      =   &H000000C0&
     Font3D          =   0  'None
     Height          =   4095
     Left            =   2160
     TabIndex        =   0
     Top             =   1200
     Width           =   5775
     Begin TextBox alunos
        Height          =   975
        Left            =   720
        MultiLine       =   -1  'True
        TabIndex        =   2
        Text            =   "  Jerri Regis Biscuola                                    2842
        Top             =   2160
        Width           =   4455
     End
     Begin TextBox Escola2
        Alignment       =   2  'Center
        BackColor       =   &H00C0C0C0&
        BorderStyle     =   0  'None
        FontBold        =   -1  'True
        FontItalic      =   0   'False
        FontName        =   "MS Sans Serif"
        FontSize        =   12
        FontStrikethru  =   0   'False
        FontUnderline   =   0   'False
        Height          =   495
        Left            =   1800
        TabIndex        =   5
        Text            =   "Escola Politécnica"
        Top             =   600
        Width           =   3735
     End
     Begin TextBox titulo
        Alignment       =   2  'Center
        BackColor       =   &H00C0C0C0&
        FontBold        =   -1  'True
        FontItalic      =   0   'False
```

```
        FontName          =   "Arial"
        FontSize          =   12
        FontStrikethru    =   0   'False
        FontUnderline     =   0   'False
        Height            =   735
        Left              =   720
        MultiLine         =   -1  'True
        TabIndex          =   4
        Text              =   "Comando Numérico para trilho de comando do robô"
        Top               =   1200
        Width             =   4455
    End
    Begin TextBox Escola
        Alignment         =   2  'Center
        BackColor         =   &H00C0C0C0&
        BorderStyle       =   0  'None
        FontBold          =   -1  'True
        FontItalic        =   0   'False
        FontName          =   "MS Sans Serif"
        FontSize          =   12
        FontStrikethru    =   0   'False
        FontUnderline     =   0   'False
        Height            =   420
        Left              =   240
        MultiLine         =   -1  'True
        TabIndex          =   3
        Text              =   "Universidade de São Paulo"
        Top               =   240
        Width             =   5295
    End
    Begin SSCommand ok
        Caption           =   "OK"
        Font3D            =   0  'None
        Height            =   495
        Left              =   4440
        Picture           =   (none)
        TabIndex          =   1
        Top               =   3480
        Width             =   1215
    End
 End
d
```

```
ABALHO.FRM - 1

RSION 2.00
gin Form execucao
 BackColor       =   &H00FF8080&
 Caption         =   "Sistema D.N.C. didático para trilho automático"
 ClientHeight    =   3930
 ClientLeft      =   1845
 ClientTop       =   1980
 ClientWidth     =   7365
 Height          =   4710
 Left            =   1785
 LinkTopic       =   "Form3"
 ScaleHeight     =   3930
 ScaleWidth      =   7365
 Top             =   1260
 Width           =   7485
 WindowState     =   2  'Maximized
 Begin Timer Amostragem
    Enabled          =   0   'False
    Interval         =   10
    Left             =   4800
    Top              =   1440
 End
 Begin CommandButton bcontinua
    Caption          =   "Continua"
    Enabled          =   0   'False
    Height           =   375
    Left             =   2040
    TabIndex         =   10
    Top              =   2520
    Width            =   1215
 End
 Begin CommandButton cesquerda
    Caption          =   "<"
    Height           =   495
    Left             =   1080
    TabIndex         =   44
    Top              =   4560
    Width            =   615
 End
 Begin CommandButton cdireita
    Caption          =   ">"
    Height           =   495
    Left             =   3480
    TabIndex         =   43
    Top              =   4560
    Width            =   615
 End
 Begin CommandButton cpara
    Caption          =   "PÁRA"
    Height           =   375
    Left             =   1080
    TabIndex         =   42
    Top              =   5160
    Width            =   3015
 End
 Begin TextBox tstatusonline
```

```
    BackColor       =   &H00C0C0C0&
    ForeColor       =   &H00C00000&
    Height          =   285
    Left            =   1200
    TabIndex        =   41
    Top             =   3000
    Width           =   2895
 End
 Begin TextBox tprograma
    BackColor       =   &H00C0C0C0&
    Height          =   285
    Left            =   1920
    TabIndex        =   40
    Top             =   1800
    Width           =   2415
 End
 Begin TextBox tlinha
    BackColor       =   &H00C0C0C0&
    Height          =   285
    Left            =   1920
    TabIndex        =   39
    Top             =   2160
    Width           =   2415
 End
 Begin ComboBox cstatus
    Height          =   300
    Left            =   1560
    TabIndex        =   31
    Top             =   480
    Width           =   7695
 End
 Begin ComboBox calarmes
    Height          =   300
    Left            =   1560
    TabIndex        =   29
    Top             =   120
    Width           =   7695
 End
 Begin Gauge GRAFICO2
    Autosize        =   -1  'True
    ForeColor       =   &H000000C0&
    Height          =   375
    InnerBottom     =   5
    InnerLeft       =   5
    InnerRight      =   5
    InnerTop        =   5
    Left            =   6720
    Max             =   100
    NeedleWidth     =   1
    Picture         =   (none)
    TabIndex        =   28
    Top             =   5760
    Width           =   1935
 End
 Begin Gauge GRAFICO1
    Autosize        =   -1  'True
```

```
    BackColor       =   &H00FFFFFF&
    ForeColor       =   &H0000C000&
    Height          =   375
    InnerBottom     =   5
    InnerLeft       =   5
    InnerRight      =   5
    InnerTop        =   5
    Left            =   6720
    Max             =   100
    NeedleWidth     =   1
    Picture         =   (none)
    TabIndex        =   27
    Top             =   5160
    Width           =   1935
End
Begin TextBox tintegrativa
    Height          =   285
    Left            =   7080
    TabIndex        =   22
    Text            =   "0.0005"
    Top             =   3840
    Width           =   1575
End
Begin TextBox taceleracao
    BackColor       =   &H00C0C0C0&
    Height          =   285
    Left            =   6840
    TabIndex        =   21
    Top             =   2160
    Width           =   1095
End
Begin TextBox tsetpoint
    BackColor       =   &H00C0C0C0&
    Height          =   285
    Left            =   6840
    TabIndex        =   20
    Top             =   4800
    Width           =   1095
End
Begin TextBox tproporcional
    Height          =   285
    Left            =   7080
    TabIndex        =   19
    Text            =   "0.2"
    Top             =   3120
    Width           =   1575
End
Begin TextBox tderivativa
    Height          =   285
    Left            =   7080
    TabIndex        =   18
    Text            =   "1"
    Top             =   3480
    Width           =   1575
End
Begin TextBox tvelocidade
```

```
    BackColor       =   &H00C0C0C0&
    Height          =   285
    Left            =   6840
    TabIndex        =   17
    Top             =   1800
    Width           =   1095
End
Begin TextBox tposicao
    Height          =   285
    Left            =   6840
    TabIndex        =   16
    Top             =   1440
    Width           =   1095
End
Begin Label ldireita
    BackColor       =   &H00C0C0C0&
    Caption         =   "Direita"
    Height          =   255
    Left            =   3480
    TabIndex        =   35
    Top             =   4200
    Width           =   615
End
Begin Label lesquerda
    BackColor       =   &H00C0C0C0&
    Caption         =   "Esquerda"
    Height          =   255
    Left            =   960
    TabIndex        =   46
    Top             =   4200
    Width           =   855
End
Begin Label lmanual
    BackColor       =   &H00C0C0C0&
    Caption         =   "Modo Manual :"
    FontBold        =   -1  'True
    FontItalic      =   -1  'True
    FontName        =   "MS Sans Serif"
    FontSize        =   8.25
    FontStrikethru  =   0   'False
    FontUnderline   =   -1  'True
    Height          =   255
    Left            =   2040
    TabIndex        =   45
    Top             =   3960
    Width           =   1455
End
Begin Label lautomatico
    BackColor       =   &H00C0C0C0&
    Caption         =   "Modo Automático :"
    FontBold        =   -1  'True
    FontItalic      =   -1  'True
    FontName        =   "MS Sans Serif"
    FontSize        =   8.25
    FontStrikethru  =   0   'False
    FontUnderline   =   -1  'True
```

```
    Height          =   255
    Left            =   1800
    TabIndex        =   38
    Top             =   1320
    Width           =   1935
End
Begin Label lprograma
    BackColor       =   &H00C0C0C0&
    Caption         =   "Programa :"
    Height          =   255
    Left            =   840
    TabIndex        =   37
    Top             =   1800
    Width           =   1215
End
Begin Label llinha
    BackColor       =   &H00C0C0C0&
    Caption         =   "Linha :"
    Height          =   255
    Left            =   840
    TabIndex        =   36
    Top             =   2160
    Width           =   1215
End
Begin Label quadro5
    BackColor       =   &H00C0C0C0&
    BorderStyle     =   1  'Fixed Single
    Height          =   2055
    Left            =   600
    TabIndex        =   34
    Top             =   3840
    Width           =   3975
End
Begin Label quadro4
    BackColor       =   &H00C0C0C0&
    BorderStyle     =   1  'Fixed Single
    Height          =   2175
    Left            =   600
    TabIndex        =   33
    Top             =   1200
    Width           =   3975
End
Begin Label lalarmes
    BackColor       =   &H00FF8080&
    Caption         =   "Alarmes :"
    FontBold        =   -1  'True
    FontItalic      =   0   'False
    FontName        =   "MS Sans Serif"
    FontSize        =   9.75
    FontStrikethru  =   0   'False
    FontUnderline   =   0   'False
    Height          =   255
    Left            =   360
    TabIndex        =   32
    Top             =   120
    Width           =   1095
```

```
End
Begin Label lstatus
   BackColor       =   &H00FF8080&
   Caption         =   "Status :"
   FontBold        =   -1  'True
   FontItalic      =   0   'False
   FontName        =   "MS Sans Serif"
   FontSize        =   9.75
   FontStrikethru  =   0   'False
   FontUnderline   =   0   'False
   Height          =   255
   Left            =   360
   TabIndex        =   30
   Top             =   480
   Width           =   855
End
Begin Label lmm2
   BackColor       =   &H00C0C0C0&
   Caption         =   "mm"
   Height          =   255
   Left            =   8040
   TabIndex        =   26
   Top             =   4800
   Width           =   495
End
Begin Label lmmporss
   BackColor       =   &H00C0C0C0&
   Caption         =   "mm/s*s"
   Height          =   255
   Left            =   8040
   TabIndex        =   25
   Top             =   2160
   Width           =   615
End
Begin Label lmmpors
   BackColor       =   &H00C0C0C0&
   Caption         =   "mm/s"
   Height          =   255
   Left            =   8040
   TabIndex        =   24
   Top             =   1800
   Width           =   495
End
Begin Label lmm
   BackColor       =   &H00C0C0C0&
   Caption         =   "mm"
   Height          =   255
   Left            =   8040
   TabIndex        =   23
   Top             =   1440
   Width           =   375
End
Begin Label lsetpoint
   BackColor       =   &H00C0C0C0&
   Caption         =   "Setpoint :"
   Height          =   255
```

```
    Left            =   5640
    TabIndex        =   15
    Top             =   4800
    Width           =   1095
End
Begin Label lerro
    BackColor       =   &H00C0C0C0&
    Caption         =   "Erro :"
    Height          =   255
    Left            =   5640
    TabIndex        =   14
    Top             =   5760
    Width           =   735
End
Begin Label lvalreal
    BackColor       =   &H00C0C0C0&
    Caption         =   "Valor real :"
    Height          =   255
    Left            =   5640
    TabIndex        =   13
    Top             =   5280
    Width           =   975
End
Begin Label lmalha
    BackColor       =   &H00C0C0C0&
    Caption         =   "Malha :"
    FontBold        =   -1  'True
    FontItalic      =   -1  'True
    FontName        =   "MS Sans Serif"
    FontSize        =   8.25
    FontStrikethru  =   0   'False
    FontUnderline   =   -1  'True
    Height          =   255
    Left            =   6720
    TabIndex        =   9
    Top             =   4440
    Width           =   855
End
Begin Label quadro3
    BackColor       =   &H00C0C0C0&
    BorderStyle     =   1  'Fixed Single
    Height          =   2055
    Left            =   5520
    TabIndex        =   12
    Top             =   4320
    Width           =   3255
End
Begin Label lderivativa
    BackColor       =   &H00C0C0C0&
    Caption         =   "Derivativa :"
    Height          =   255
    Left            =   5640
    TabIndex        =   7
    Top             =   3480
    Width           =   1215
End
```

```
Begin Label lposicao
   BackColor       =   &H00C0C0C0&
   Caption         =   "Posição :"
   Height          =   255
   Left            =   5640
   TabIndex        =   0
   Top             =   1440
   Width           =   1335
End
Begin Label lvelocidade
   BackColor       =   &H00C0C0C0&
   Caption         =   "Velocidade :"
   Height          =   255
   Left            =   5640
   TabIndex        =   2
   Top             =   1800
   Width           =   1335
End
Begin Label laceleracao
   BackColor       =   &H00C0C0C0&
   Caption         =   "Aceleração :"
   Height          =   255
   Left            =   5640
   TabIndex        =   3
   Top             =   2160
   Width           =   1335
End
Begin Label lintegrativa
   BackColor       =   &H00C0C0C0&
   Caption         =   "Integrativa :"
   Height          =   255
   Left            =   5640
   TabIndex        =   8
   Top             =   3840
   Width           =   1215
End
Begin Label lproporcional
   BackColor       =   &H00C0C0C0&
   Caption         =   "Proporcional :"
   Height          =   255
   Left            =   5640
   TabIndex        =   5
   Top             =   3120
   Width           =   1215
End
Begin Label ldadosatuais
   BackColor       =   &H00C0C0C0&
   Caption         =   "Dados Atuais :"
   FontBold        =   -1  'True
   FontItalic      =   -1  'True
   FontName        =   "MS Sans Serif"
   FontSize        =   8.25
   FontStrikethru  =   0   'False
   FontUnderline   =   -1  'True
   Height          =   255
   Left            =   6360
```

```
    TabIndex        =   4
    Top             =   1080
    Width           =   1455
End
Begin Label lcontrolador
    BackColor       =   &H00C0C0C0&
    Caption         =   "Controlador :"
    FontBold        =   -1  'True
    FontItalic      =   -1  'True
    FontName        =   "MS Sans Serif"
    FontSize        =   8.25
    FontStrikethru  =   0   'False
    FontUnderline   =   -1  'True
    Height          =   255
    Left            =   6480
    TabIndex        =   1
    Top             =   2760
    Width           =   1335
End
Begin Label quadro1
    BackColor       =   &H00C0C0C0&
    BorderStyle     =   1  'Fixed Single
    Height          =   1575
    Left            =   5520
    TabIndex        =   6
    Top             =   960
    Width           =   3255
End
Begin Label quadro2
    BackColor       =   &H00C0C0C0&
    BorderStyle     =   1  'Fixed Single
    Height          =   1575
    Left            =   5520
    TabIndex        =   11
    Top             =   2640
    Width           =   3255
End
Begin Menu Arquivo
    Caption         =   "&Arquivo"
    Begin Menu Selecionar
       Caption          =   "&Selecionar"
    End
    Begin Menu linha1
       Caption          =   "-"
    End
    Begin Menu Sair
       Caption          =   "Sai&r"
    End
End
Begin Menu Comando
    Caption         =   "&Comando"
    Begin Menu Manual
       Caption          =   "&Manual"
    End
    Begin Menu Automatico
       Caption          =   "&Automático"
```

```
    End
    Begin Menu linha
       Caption          =    "-"
    End
    Begin Menu Compilar
       Caption          =    "&Compilar arquivo"
       Enabled          =    0   'False
    End
    Begin Menu Executar
       Caption          =    "&Executar arquivo"
       Enabled          =    0   'False
    End
 End
d
```

```
1    #include <stdlib.h>
2    #include <windows.h>
3    #include <conio.h>
4    #include <math.h>
5
6
7
8    _export  FAR PASCAL motor(int num)
9    {
10     unsigned  drivera = 0x2d4;
11     unsigned  driverb = 0x2d5;
12     outp(drivera, num);
13     outp(driverb, 255-num);
14     return  0;
15   }
16
17   _export  FAR PASCAL lecontador1()
18   {
19     long result;
20     unsigned  port = 0x210; //endereco da porta de entrada 1 do encoder ;
21     result = inport(port);
22     return  result;
23   }
24
25   _export  FAR PASCAL lecontador2()
26   {
27     int result;
28     unsigned  port = 0x211; //endereco da porta de entrada 2 do encoder ;
29     result = inp(port);
30     return  result;
31   }
32
33
34   _export  FAR PASCAL PID(long Kd, long Ki, long Kp, int TC, long M, long erro, long erro
     ant, long s)
35   {
36    unsigned  drivera=0x2d4;  //endercos de saida da placa D/A
37    unsigned  driverb=0x2d5;
38    int V;
39    const  Vzero=128;
40    const  Vmax=255;
41    const  Vmin=0;
42
43      V=floor((Vzero+Kp*erro+(Kd/(10))*(erro-erroant)+Ki*s)+0.5);
44      if  (V>=(M*(Vmax-Vzero)+Vzero))
45          {
46           V=floor(M*(Vmax-Vzero)+0.5)+Vzero;
47          }
48      if  (V<=(Vzero-M*(Vzero-Vmin)))
49          {
50           V=Vzero-floor(M*(Vzero-Vmin)+0.5);
51          }
52      outp(drivera,V);
53      outp(driverb, 255-V);
54      return (V);
55
56   }
57
58
59   _export  FAR PASCAL fimcurso()
60   {
61     int result;
62     int port = 0x212; //endereco da porta de entrada binaria ;
63     result = inp(port);
64     return  result;
65   }
66
```