RELATÓRIO DE ESTÁGIO

George BARRETO PRATA
Tutor acadêmico : Gérard SCHIAPARELLI
Tutor : Fabrice REYNAUD

Março – Agosto 2004

Biblioteca Setorial do CDSA. Março de 2021.

Sumé - PB

AGRADECIMENTOS

Meus agradecimentos se destinam a todas as pessoas que contribuiram para a realização deste projeto.

Agradeço à empresa LIM por ter me dado a possibilidade de fazer um estágio numa área de meu interesse e dentro de um ambiente de trabalho muito agradável.

Agradeço especialmente aos engenheiros **Fabrice Reynaud** e **Yoann Di Ruzza** por terem me proposto um título de estágio de meu total interesse, pelas suas explicações, pela disponibilidade, pelos seus conselhos, pelo apoio na procura de informações e pela ajuda na realização deste projeto.

Por fim agradeço a cada funcionário da empresa por toda a ajuda recebida e pela recepção no ambiente de trabalho da empresa durante esses 5 meses de estágio.

SUMÁRIO

1 CONTEXTO DO ESTÁGIO

1.1 A empresa

O objetivo de **LIM S.A.** é a concepção, a fabricação e a comercialização de aparelhagem eletrônica e de programas associados especializados para a instrumentação durante a perfuração e a injeção.

Os principais setores de atividade de **LIM S.A.** são as minas e os canteiros, as fundações especiais, os trabalhos subterrâneos e a geotecnia.

A **LIM S.A.** é uma empresa de pequeno porte com apenas 15 funcionários divididos, segundo o organograma da figura 1, em três departamentos : Recursos Humanos (RH), Pesquisa e Desenvolvimento (P&D) e Produção.

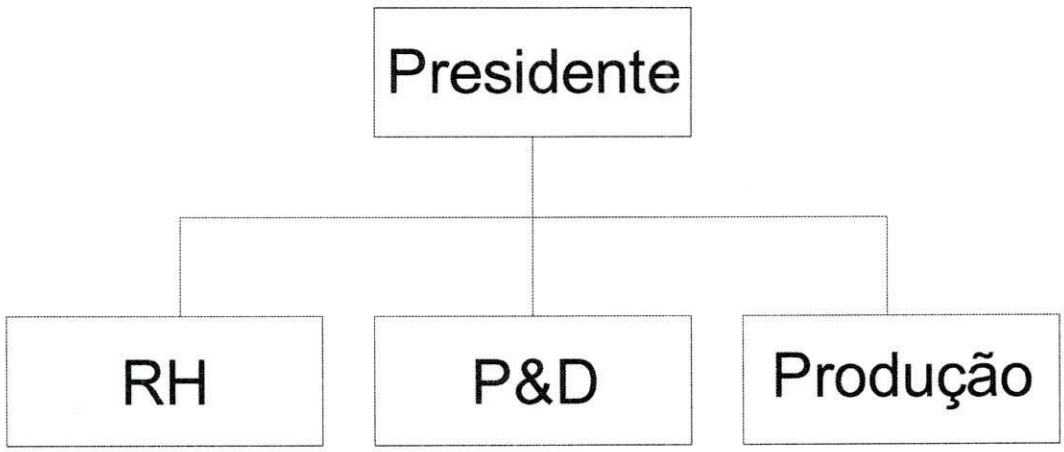

Figura 1 - Orgonograma da empresa

No departamento de Recursos Humanos trabalham 6 pessoas. Uma delas cuida do recrutamento de estagiários e admissão de novos funcionários. Existe uma pessoa que cuida da parte jurídica e contábil da empresa. Uma secretária é responsável pela compra de material para a empresa e contato com fornecedores. As outras três pessoas são da parte de venda de produtos da empresa. Entre elas uma é o diretor comercial que chefia os outros dois e o qual se reporta diretamente ao presidente da empresa para definir metas e estratégias de vendas.

No departamento de Pesquisa e Desenvolvimento estão os engenheiros e programadores da empresa. São dois engenheiros eletricistas e dois

programadores. Eles são responsáveis pelo desenvolvimento de todos os produtos e também pela manutenção e desenvolvimento das ferramentas da empresa. Um dos engenheiros é o chefe de desenvolvimento e a ele ficam submetido os demais. O estágio em questão foi realizado nesse departamento com a orientação do chefe de desenvolvimento e com ajuda dos demais funcionários da empresa. O tema do estágio foi voltado para o aperfeiçoamento de uma ferramenta da empresa, não incluindo portanto nenhuma pesquisa ou desenvolvimento com produtos a serem vendidos.

No departamento de Produção estão as pessoas que fazem a soldagem e montagem das placas, o cabeamento dos produtos, fabricação de caixas que servirão de embalagem para os produtos, etc. Nesse departamento trabalham 5 pessoas sendo uma delas o chefe de produção o qual está em constante contato com o chefe de desenvolvimento e com o diretor comercial.

O presidente da empresa atua como um diretor-geral, coordenando os departamentos para que os objetivos sejam alcançados. Apesar de atuar em todos os departamentos, este se situa mais próximo do diretor comercial atuando mais nas vendas dos produtos.

A empresa possui dois principais tipos de produtos : produtos para perfuração e para injeção. Alguns desses dois tipos serão mostrados a seguir.

FORALIM 4G

O FORALIM 4G, mostrado na figura 3, é uma central de medidas para o armazenamento de parâmetros de perfuração. Consiste em um aparelho que faz a medição de parâmetros de perfuração (velocidade de avanço da ferramenta de perfuração, pressão sobre a ferramenta de perfuração, pressão de injeção do fluido de perfuração, velocidade de rotação da ferramenta de perfuração, etc..), os armazena e os mostra através de um mostrador digital.

A figura 2 abaixo mostra o FORALIM 4G sendo usado como instrumento de visualização de parâmetros de perfuração na construção do túnel Lyon-Genébre.

Figura 2 - Foralim 4G sendo usado na construção do túnel Lyon-Genébre

Figura 3 – Foralim 4G – central de medidas para parâmetros de perfuração

INJECTLIM 4G

O INJECTLIM 4G, mostrado na figura 4, é uma central de medidas para o armazenamento de parâmetros de injeção. Consiste em um equipamento que faz a medição de parâmetros de injeção (de um a quatro pontos de injeção simultaneamente), os armazena e os mostra através de um mostrador digital.

Os produtos apresentados são desenvolvidos em Lyon, na LIM S.A. Todas as placas eletrônicas contidas nesses produtos são desenvolvidas pelos engenheiros e programadores da empresa. As placas eletrônicas desenvolvidas são colocadas dentro de caixas vermelhas, como mostrado nas figuras 3 e 4, para depois serem vendidas. Essas caixas vermelhas também são feitas na empresa por um técnico em mecânica a partir de uma mesa de usinagem que se situa no subsolo da empresa e que foi o objeto de trabalho do estágio.

Figura 4 – Injectlim 4G – central de medidas para parâmetros de injeção

1.2 Atividades do estágio

O objetivo do estágio é fazer uma placa eletrônica utilizando uma nova tecnologia e que seja compatível com o protótipo antigo. Posteriormente o objetivo será fazer um novo programa de interação com o usuário, em G-code [1], que possa enviar comandos simples à placa para execução da usinagem.

G-code é um nome dado para a linguagem de programação que é usada em máquinas NC (Controlada Numericamente) e CNC (Controlada Numericamente por Computador). É também o nome de qualquer palavra em um programa CNC que começa com a letra G, e geralmente é um código que diz à máquina que tipo de ação ela deve executar, como:

- Movimento controlado em linha reta ou arco;
- Movimento rápido;
- Seqüência de movimentos controlados que irão resultar em buracos sendo perfurados ou em perfis decorativos nos limites da peça;
- Troca de ferramenta.

Existem ainda outros códigos que podem ser pensados como registradores em um computador. Um deles é o M-code (código de máquina) que controla a máquina por completo causando sua parada, seu início, etc.

O estágio foi dividido em fases para a sua realização. Começou com o estudo da mesa de usinagem e do antigo protótipo. Em seguida foram escolhidos os algoritmos da placa eletrônica e a concepção da placa. Por fim, foram implementados os algoritmos escolhidos e feitos os testes.

2 PROJETO MESA DE USINAGEM XYZ

2.1 Mesa de usinagem XYZ

A mesa de usinagem mostrada na figura 5 possui um braço mecânico com movimentos nos eixos x, y e z. Duas placas eletrônicas (22cmx23cm) controlam essa mesa a partir de um Posto de Comando (PC). O objetivo do estágio é desenvolver uma placa eletrônica que execute funções de deslocamento linear e circular a partir do mesmo Posto de Comando (PC). Essa nova placa deverá conter uma unidade de inteligência (microcontrolador) e um dispositivo PLD (Programmable Logic Device) que simule todas a parte digital das antigas placas. A nova placa deverá ter um tamanho reduzido (13cmx16cm) e, devido ao microcontrolador e ao dispositivo PLD, deverá ser possível alterar a funcionalidade da placa sem alteração do hardware.

A primeira fase do estágio foi dedicada para a minha familiarização com a mesa de usinagem em questão e o antigo protótipo.

A mesa tem quatro eixos : dois eixos para a direção X, um para a direção Y e um para a direção Z.

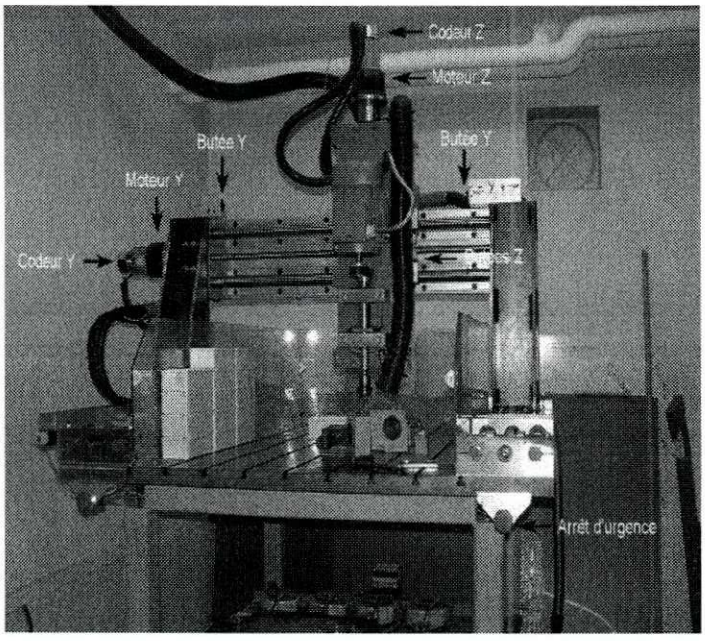

Figura 5 – Motores, contactores e codificadores da mesa de usinagem xyz

Cada eixo tem um motor de passo à 400 passos/volta e um codificador ótico associado (a 500 imp/volta menos o codificador do eixo Z que é a 1000 imp/volta) que faz a medida do número de passos feito por cada motor. Existe também para cada eixo contactores de início e fim trajeto e a parada de emergência que corta a energia de todos os motores.

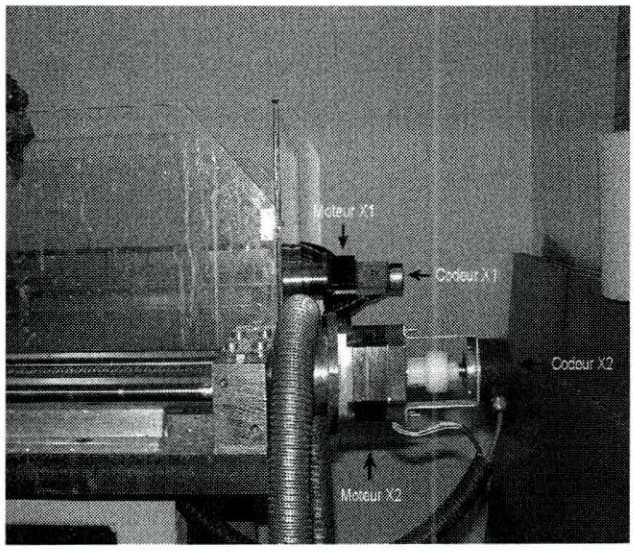

Figura 6 – Parte traseira da mesa de usinagem xyz: Motor e Codificador eixo X

A figura 6 mostra os dois motores que são utilizados para o eixo X: o motor X1 e o motor X2. Trata-se de um eixo particular pois com frequência ocorre atraso entre X1 e X2. Este atraso causa um problema na usinagem da peça e às vezes é

necessário corrigir fazendo com que o motor X2 alcance o X1 (bloqueando os pulsos do relógio de X1 até termos atraso nulo).

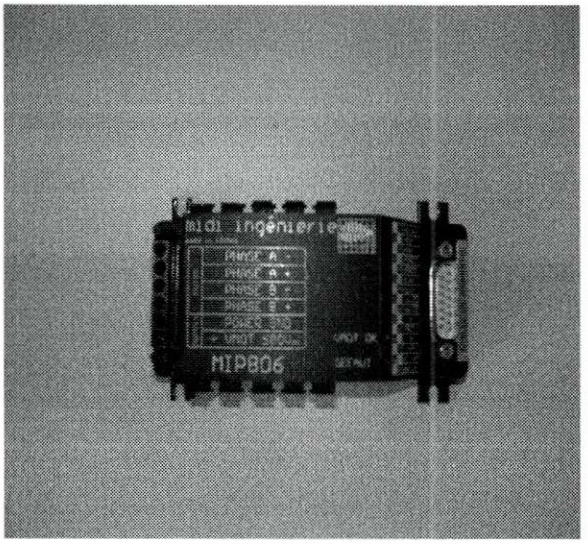

Figura 7 – Módulo de comando dos motores de passo da mesa : MIP806

A figura 7 mostra um módulo de comando passo à passo e meio passo que controla os motores de passo a partir da placa eletrônica, MIP806 [2]. Cada pulso de relógio enviado pela placa ao MIP vai provocar um deslocamento de um passo num sentido, o qual também é definido por um sinal enviado ao MIP. Uma outra função do MIP é de enviar à placa as informações de defeito nos motores.

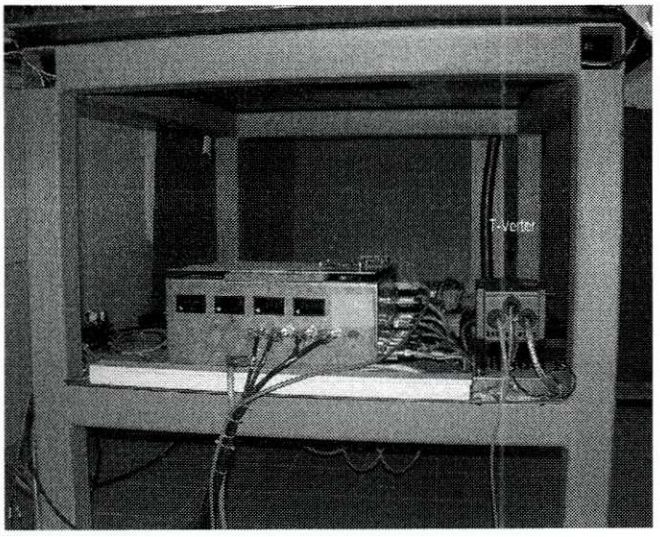

Figura 8 – Visão inferior da mesa de usinagem : T-Verter e caixa das placas eletrônicas

Na mesa de usinagem existe também um motor usado para perfurar a peça que é controlado por um variador de frequência : T-verter [3]. Como mostrado na

figura 8, ao lado do T-verter existe uma caixa onde se encontram as duas placas eletrônicas que controlam a mesa de usinagem. Todas as informações de contactores, defeito dos motores e codificadores vão para essa caixa assim como os comandos para os motores saem dela para os MIPs.

O T-verter se comunica com o computador a partir de uma comunicação série (RS232) e é a partir do computador que partem os comandos para girar o motor, colocá-lo em sentido horário/anti-horário e a escolha da velocidade de giro do motor. Entre a caixa onde estão as placas eletrônicas e o computador existem dois tipos de comunicação : uma série e uma paralela. A partir da comunicação paralela são enviados todos os comandos para os motores de passo, a comunicação série (RS232) serve para zerar os mostradores das placas.

2.2 Motor de passo

Figura 9 - Motor de Passo

Um motor de passo (figura 9) é um dispositivo digital. Como um computador, ele processa uma informação digital para realizar um ato desejado, neste caso, um movimento. Um motor de passo deverá seguir as instruções digitais fielmente, da mesma maneira que é esperado que um computador faça. Esta é a principal característica de um motor de passo. Como mostrado na figura 10, a cada pulso ele faz um incremento rotativo (passo). Cada passo é só uma porção de uma rotação completa. Então, vários pulsos podem ser aplicados para alcançar uma quantia desejada de rotação do eixo como pode ser visto na figura 11 abaixo.

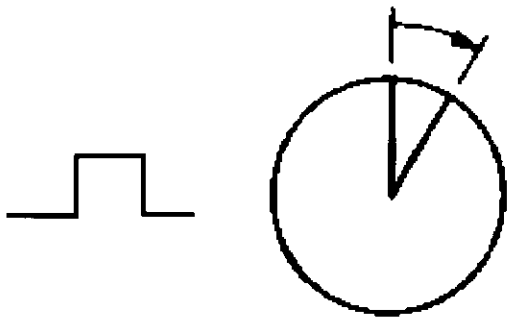

Figura 10 – A cada pulso produz-se um incremento rotativo

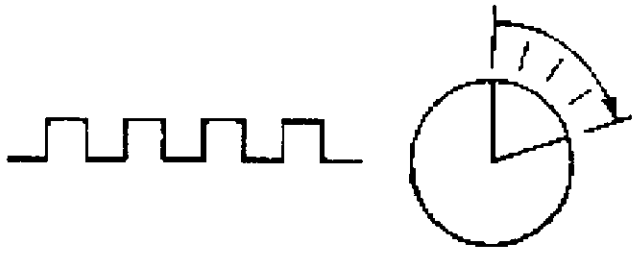

Figura 11 – Para vários pulsos, uma série de incrementos é realizada

A precisão de um motor de passo é principalmente determinada pelo número de passos por rotação (quanto maior for a quantidade de passos, maior será a precisão). Para uma precisão mais alta, alguns controladores de motor de passo dividem passos completos, em meio-passos ou micro passos. Na figura 12(a) é mostrado um passo completo e na 12(b) meio-passo.

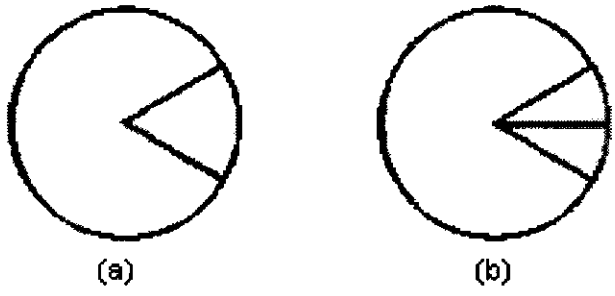

Figura 12 – (a) Incremento em passo completo (b) Incremento em meio-passo

Um motor de passo é eletromagnético. Ele converte mecanicamente pulsos digitais em incrementos de rotação do eixo. A rotação não só tem uma relação direta ao número de pulsos, mas sua velocidade é relacionada à freqüência dos mesmos.

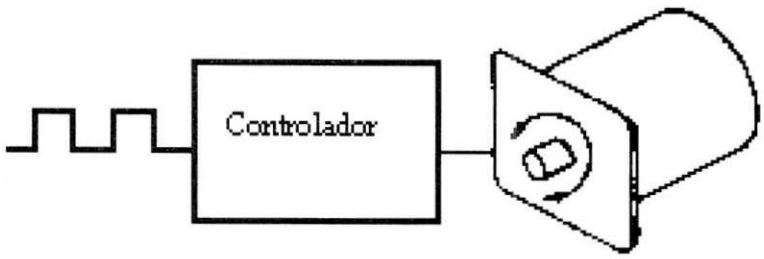

Figura 13 – Diagrama de um motor com controlador, no caso do estágio um MIP

Entre cada passo, o motor pára na posição (com sua carga) sem a ajuda de embreagens ou freios. De acordo com a figura 13, um motor de passo pode ser controlado de uma forma que faz ele girar um certo número de passos, produzindo um movimento mecânico por uma distância específica, e então ele segura a sua carga quando pára. Além disso, ele pode repetir a operação quantas vezes se desejar.

2.3 Codificador Ótico

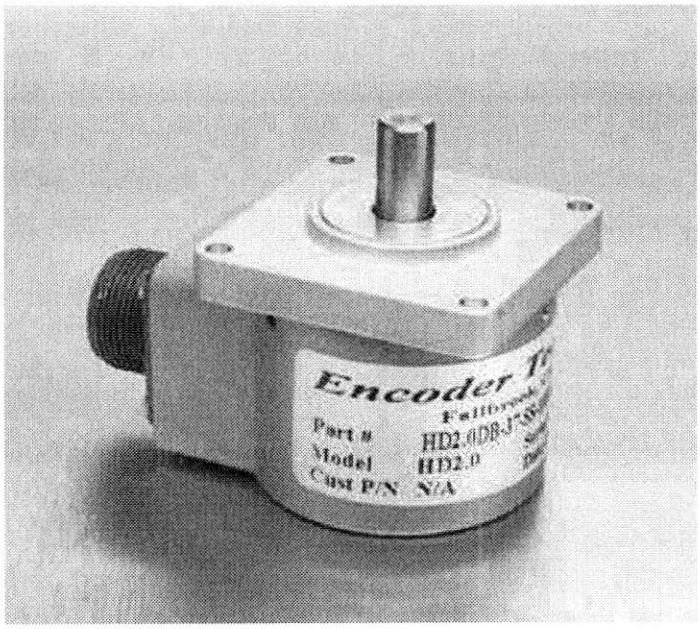

Figura 14 - Codificador Ótico

Um codificador ótico, conforme diagrama da figura 15, é um tipo de codificador consistindo de um disco rotativo, uma fonte de luz e um fotodetetor (sensor de luz). O disco (figura 15), o qual é montado sobre um eixo rotativo, possui partes codificadas compostas de setores opacos e transparentes. À medida que o disco gira, essas partes interrompem a luz emitida para o fotodetetor, gerando um sinal digital ou um pulso na saída.

Assim, um codificador ótico acoplado ao eixo de um motor de passo irá gerar em sua saída pulsos que poderão ser tratados por um microcontrolador. Com isso, o número de passos efetuados pelo motor de passo poderão ser calculados no microcontrolador e mostrados em um display.

Figura 15 – Diagrama ilustrativo do princípio de funcionamento de um codificador ótico

2.4 Especificações do projeto

Três elementos fazem parte do projeto da mesa de usinagem, são eles: o Posto de Comando (PC), de onde partem os comandos para a usinagem de peças ; a Placa Eletrônica, de onde partem os comandos de usinagem para os motores ; a Mesa de Usinagem, onde as peças são trabalhadas.

No Posto de Comando um programa de usuário iria receber instruções de usinagem em G-code. Esse programa em G-code escrito pelo usuário seria interpretado e comandos simples de deslocamento linear e deslocamento circular serão enviados à placa via RS232.

O estágio em questão não se preocupa com o programa do usuário nem em como os programas serão interpretados e decompostos em funções simples. Assim,

considera-se a área de estudo do estágio o conjunto formado pela placa eletrônica e a mesa de usinagem como mostra a figura 16.

O projeto, então, foi desenvolver o hardware da placa eletrônica e um software embutido associado que pudesse realizar as funções de deslocamento linear e circular a partir das informações recebidas via RS232 e a contagem de passos realizados pelos motores a partir das informações dos codificadores. Para isso a placa eletrônica deveria atuar nos motores da mesa de usinagem, fazer a leitura dos sensores (codificadores óticos), informar a ocorrência de defeitos, informar o estado dos contactores e bloquear, instantaneamente, o comando dos motores em caso de problema.

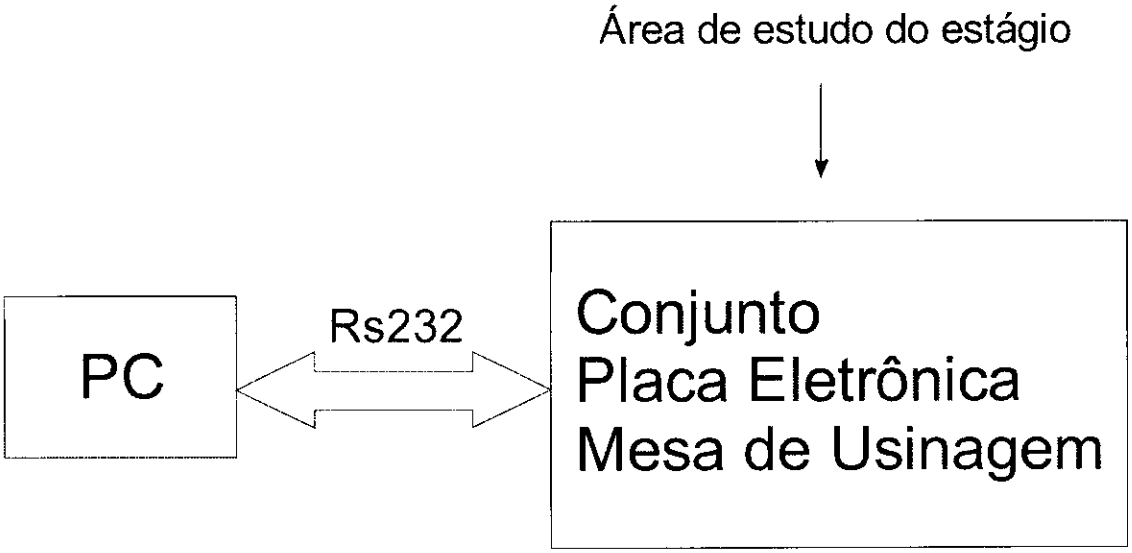

Figura 16 - Área de estudo: conjunto placa eletrônica/mesa de usinagem

2.5 Elaboração dos algorítimos da placa

Para a sequência do projeto era necessário fazer o algorítimo que seria utilizado na placa eletrônica. A placa iria receber de uma comunicação série (RS232) os comandos básicos de um deslocamento linear e circular que seriam decodificados por um programa do usuário.

Este programa do usuário seria escrito em G-code (código ciado pela Schneider muito utilizado na usinagem de peças) e ele mesmo iria decodificar este G-code em um G-code "simples". Todas as funções escolhidas seriam decompostas em funções G01 (deslocamento linear), G02 (deslocamento circular sentido horário), G03 (deslocamento linear sentido anti-horário) e seriam essas funções que seriam enviadas à placa para realizar a usinagem. A lista dos G-codes que foram escolhidos está em anexo.

Após ter escolhido quais funções a placa deveria executar era necessário escolher um algorítimo para o deslocamento linear e um outro para o deslocamento circular. Um algorítimo foi escolhido: o Algoritmo de Bresenham [4] para a reta (figura 17) e para o círculo (figura 18).

O Algoritmo de Bresenham também é conhecido por Algoritmo do Ponto Médio. Para esse algoritmo é calculado um ponto Q que fica no meio de dois pixels. Se o valor da reta ideal, neste ponto x, for maior que o ponto Q, o pixel superior é ligado, caso o valor da reta for menor que o ponto Q se liga o outro pixel, e por padrão é aceso o pixel de baixo se o valor deste ponto da reta for igual ao Q.

Este algoritmo é eficiente, uma vez que trabalha apenas com números inteiros, evitando assim o uso de vírgula flutuante. O fato de não utilizar a multiplicação acelera também os cálculos que são necessários efectuar. Por esses motivos o Algoritmo de Bresenham foi escolhido como algoritmo para o deslocamento linear e circular.

Figura 17 – Algoritmo de Bresenham para a Reta

Para o caso do projeto em questão o deslocamento sempre se dá numa direção (ou X ou Y) e as correções, quando necessárias, ocorrem na outra direção. Se, por exemplo, estivermos no ponto (Xi,Yi) usaremos os seguintes passos para decidir se o próximo deslocamento será para o ponto D (Xi+1,Yi) ou U (Xi+1,Yi+1) :

- Calcularemos o ponto médio M (Xi+1,Yi+1/2)
- Calcularemos a inclinação da reta m = (Yf – Yo)/(Xf – Xo)
- Calcularemos d = f(X,Y) = (Yi+1/2) - m*(Xi+1 - Xo) – Yo
- Por fim teremos :
    - Se d=0, o ponto pertence à reta e iremos nos mover para D (default)
    - Se d>0, M está acima da reta e iremos mover para D
    - Se d<0, M está abaixo da reta e iremos mover para U

Figura 18 – Algoritmo de Bresenham para o círculo

A mesma lógica foi utilizada para o círculo. A diferença é que será utilizada a equação de uma círculo no lugar da equação da reta. Os passos a seguir estão descritos abaixo :

- Calcularemos o ponto médio M (Xi+1,Yi+1/2)
- Calcularemos d = f(X,Y) = $X^2 + Y^2 – r^2$
- Por fim teremos :
    - Se d=0, o ponto pertence à círculo e iremos nos mover para D (default)
    - Se d>0, M está acima da círculo e iremos mover para D

- Se d<0, M está abaixo da círculo e iremos mover para U

O cuidado que foi tomado com relação ao círculo foi verificar se |X|>|Y| ou se |X|<|Y|. Caso |X|>|Y| o deslocamento se dava em Y e se corrigia em X. Caso |X|<|Y| o deslocamento se dava em X e se corrigia em Y.

O algoritmo de Bresenham foi escrito em linguagem de programação C, o que garante a portabilidade desse código. Assim, o programa pode ser transferido para um microcontrolador (no caso o PIC) sem que houvesse nenhuma alteração de código e as funções nele contidas puderam ser executadas.

2.6 Concepção

Nesta parte do projeto foi decidido como seria a nova placa eletrônica. O objetivo era fazer uma placa com menores dimensões possíveis com uma nova tecnologia e que fizesse no mínimo a mesma coisa que as antigas placas.

A eletrônica de base seria a mesma que a das antigas placas porém teríamos muitas coisas novas:

- Na nova placa uma comunicação série (RS232) seria utilizada entre o PC e placa para comandar os motores de passo (nas placas antigas utilizava-se comunicação paralela) e a comunicação paralela seria utilizada por um programa chamado NINOS (utilizado para programar a usinagem de peças);
- Um microcontrolador, PIC18F458 [5], seria utilizado para fazer a gestão de motores e codificadores;
- Uma CPLD, CY37128P84-125JI, seria utilizada para a gestão de contactores e os defeitos atuando no bloqueio dos relógios e da energia dos motores.

➢ 2.6.1 Eagle

Após ter escolhido os componentes a serem utilizados na placa era necessário começar a fazer o posicionamento e roteamento dos componentes. O programa utilizado para este fim foi o Eagle [6].

Eagle nos oferece uma biblioteca contendo um grande número de componentes e a partir desses componentes é possível criar um esquema elétrico fazendo ligações entre os componentes. Também é possível criar uma biblioteca personalizada e criar novos componentes.

Cada vez que um componente é selecionado e colocado na janela do esquema elétrico uma package desse componente é colocado automaticamente na janela de roteamento da placa eletrônica. Assim, após terminado o esuqema elétrico, os componentes podem ser posicionados e a placa pode ser roteada da melhor maneira possível. As figuras 19 e 20 mostram, respectivamente, o ambiente do esquema elétrico e do roteamento da placa eletrônica.

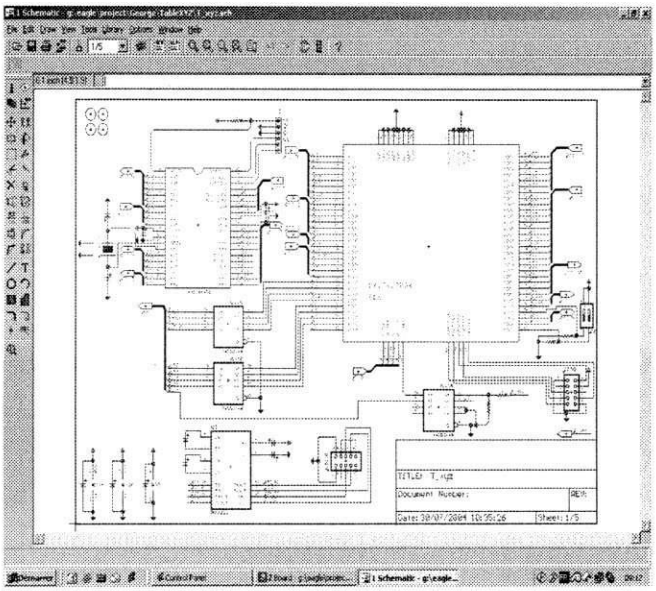

Figura 19 - Janela do esquema elétrico

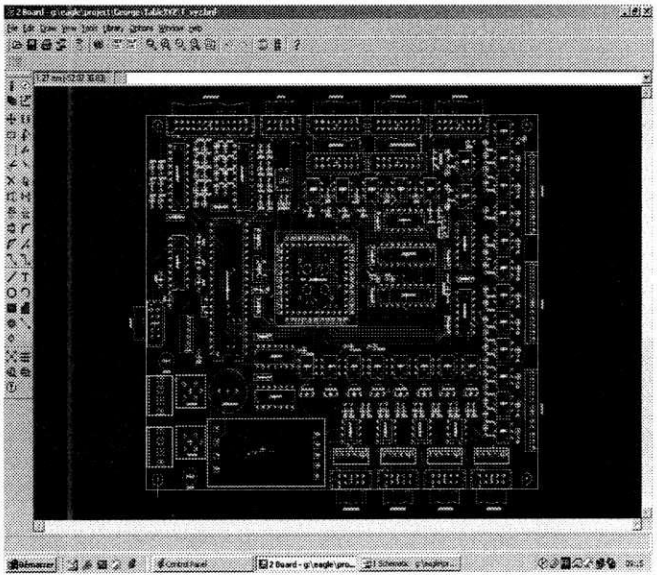

Figura 20 - Janela da placa eletrônica

2.7 Implementação e validação

Uma vez terminado o posicionamento e o roteamento dos componentes a próxima etapa era programar os componentes : CPLD e microcontrolador.

➢ 2.7.1 A programação da CPLD

O objetivo da utilização de uma CPLD na nova placa eletrônica era de reduzir as dimensões da placa (uma vez que ela iria substituir toda a parte digital das antigas placas) e de bloquear, à nível de hardware, todos os problemas de contactores e defeitos atuando nos relógios e energia dos motores de passo.

Como mostra a figura 21, a necessidade foi simular equações lógicas que pudessem interromper o envio de comandos aos motores em caso de se chegar a um contactor ou de se constatar um defeito no sistema. Nesse caso, ao invés de enviar comandos aos motores, a CPLD enviaria ao PIC a informação de que houve um problema.

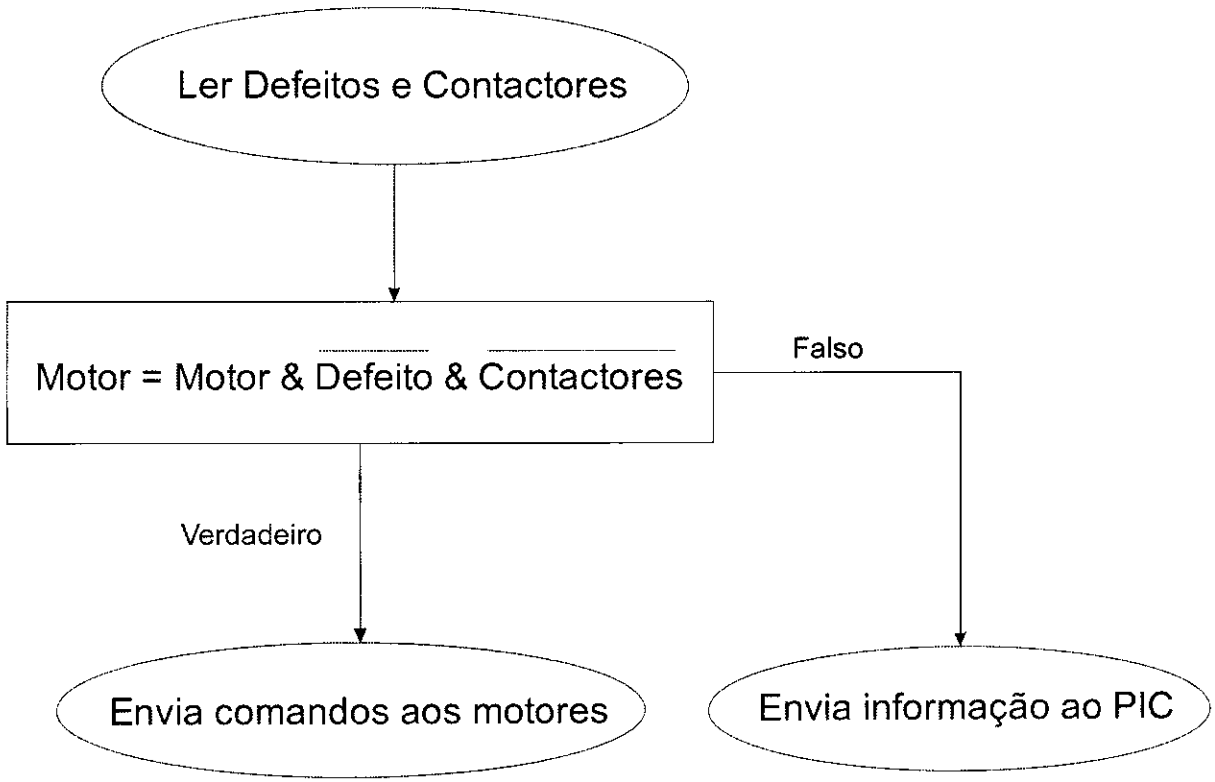

Figura 21 - Diagrama de fluxo do programa contido na CPLD

O programa utilizado para a programação da CPLD foi o Galaxy [7], como mostra a figura 22. Este programa permite o programação em VHDL [8] da CPLD e após compilação é possível nos arquivos de saída gerados as equações realizadas pela CPLD (ver anexo).

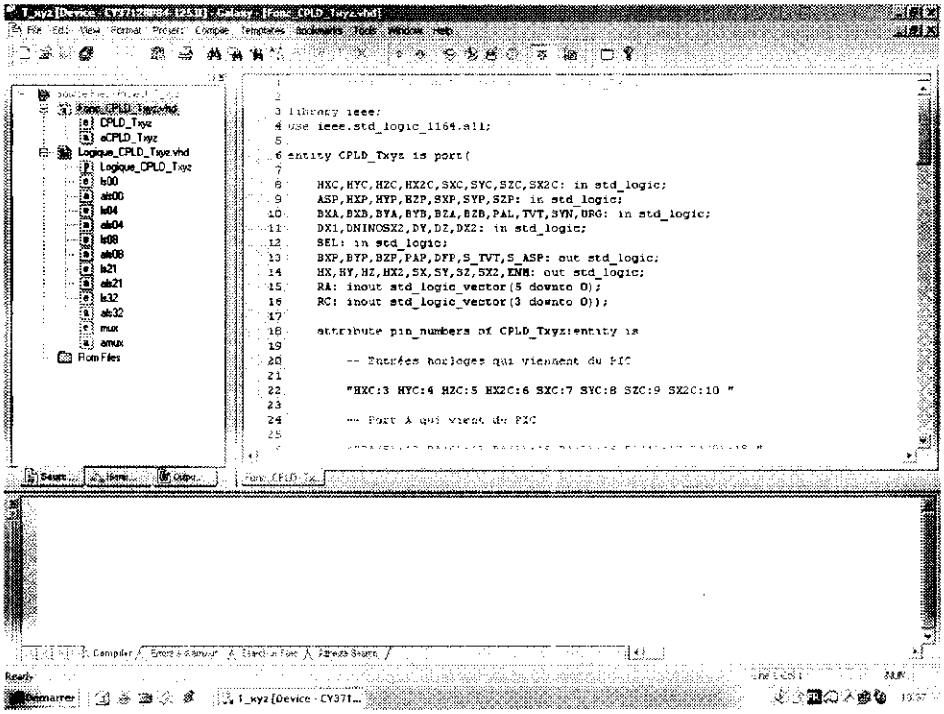

Figura 22 – Programa utilizado para programar a CPLD: Galaxy

➢ 2.7.2 A programação do microcontrolador

A programação do microcontrolador foi feita em linguagem C e as funções a seguir foram implementadas:

- Uma função para enviar as informações dos contactores e defeito dos motores para o PC via RS232;
- Uma função para contar o número de impulsos enviado por cada codificador ótico ;
- Uma função para ligar o aspirador da mesa de usinagem;
- Uma função para fazer a gestão da energia dos motores ;
- Uma função para o deslocamento linear ;
- Uma função para o deslocamento circular.

As funções mais importantes eram as funções de deslocamento linear e de deslocamento circular.

No caso de um deslocamento linear o usuário, a partir da função G01, dava as coordenadas x e y do ponto final e a partir do Algoritmo de Bresenham a ferramenta descrevia uma reta até o ponto desejado.

Para o deslocamento circular existia o problema de que o usuário fornecia, a paritr das funções G02 e G03, o raio do arco de círculo e o deslocamento angular que desejava que fosse executado pela ferramenta. Assim, tornava-se necessário um algoritmo que encontrasse o valor do ponto final para onde a ferramenta deveria ir descrevendo o círculo de raio dado.

Uma solução seria o Cálculo por Tabela onde os valores dos senos seriam calculados previamente e armazenados em uma tabela, sendo necessária uma busca na tabela para encontrar o valor do seno do ângulo e, a paritr de x=r*cosø e y=r*senø, encontrar o ponto final. O problema dessa solução era que quanto maior fosse a precisão maior seria a tabela e, consequentemente, maior necessidade de memória.

A solução encontrada, então, foi usar o Algoritmo CORDIC [9] para encontrar o ponto final.

O Algoritmo CORDIC fornece um método iterativo de rotação vetorial a partir de ângulos arbitários usando apenas deslocamentos e somas. Se um vetor V com componentes (x,y) for rotacionado de um ângulo ø um novo vetor V' com componentes (x',y') é dado por :

$$V' = \begin{pmatrix} X' \\ Y' \end{pmatrix} = \begin{pmatrix} X\cos(\phi) - Ysen(\phi) \\ Y\cos(\phi) + Xsen(\phi) \end{pmatrix}$$

A figura 23 mostra a rotação do vetor V de um ângulo ø.

Figura 23 – Rotação Vetorial de V de um ângulo ø

As equações podem ser reescritas e reorganizadas de forma a chegar nas seguintes equações :

$$X' = \cos(\phi)\left[X - Y\tan(\phi)\right]$$
$$Y' = \cos(\phi)\left[Y + X\tan(\phi)\right]$$

A multiplicação pela tangente pode ser eliminada se os ângulos de rotação (-90° até 90°) e, portanto, a tangente forem restringidos de forma que tanø = 2^{-i} . Com ø = arctan(2^{-i}) o termo em cosseno também pode ser simplificado cos(ø) = cos(-ø). Com essas simplificações as equações ficariam :

$$Xi+1 = Ki\left[Xi - Yidi2^{-i}\right]$$
$$Yi+1 = Ki\left[Yi + Xidi2^{-i}\right]$$

Onde Ki = cos(arctan(2^{-i})) e di = +-1. Esta constante é uma constante para um número fixo de iterações e pode ser calculada no início do programa e utilizada posteriormente. Uma boa maneira de implementar o fator K é inicializar o laço

iterativo com um vetor de tamanho K o que compensaria o ganho inerente ao Algoritmo CORDIC.

Assim, as equações básicas do Algoritmo CORDIC seriam :

$$Xi+1=\left[Xi-Yidi2^{-i}\right]$$
$$Yi+1=\left[Yi+Xidi2^{-i}\right]$$

Por fim, com o ponto final calculado usava-se o Algoritmo de Bresenham para o círculo e deslocava-se a ferramenta até o ponto desejado.

```
// Initialisation des variables
I = 0;
x = 4; // Coordonnée x initiale
y = 4; // Coordonnée y initiale
a = 0;
d2 = 2; // Diviseur
for(i=0; i<15; i++)
{
        d2/= 2; // Multiple de 2-i
        dx=x*d2;
        dy=y*d2;
        da=atan(d2);
        da=180*da/PI; // Pour une valeur en degré
        if(y<0)
        {
                x -= dy;
                y += dx;
                a -= da;
        }
        else
        {
                x += dy;
                y -= dx;
                a += da;
        }
}
```

Trecho de código em linguagem C do Agoritmo CORDIC

O programa utilizado para a programação do microcontrolador foi o MPLAB [10]. Foi à partir do MPLAB que foi possível programar o microcontrolador e também

testar, em modo debug, as funções que foram implementadas. A figura 24 mostra a janela de programação do MPLAB.

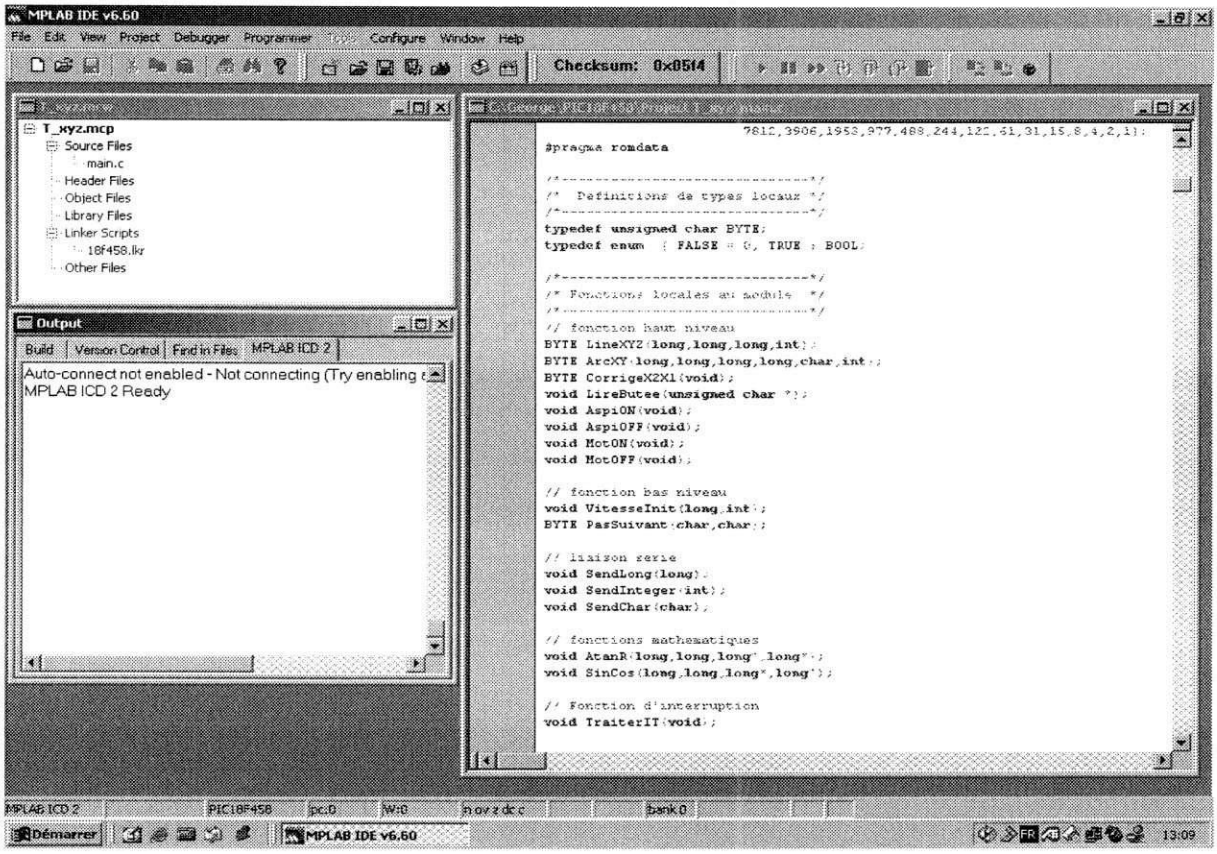

Figura 24 – Programa utilizado para programar o PIC: MPLAB

➢ 2.7.3 Testes

Basicamente os testes foram realizados para verificar se o microcontrolador e a CPLD estavam funcionando de acordo com o esperado. Para que os testes fossem realizados foi necessário retirar as antigas placas eletrônicas e conectar o novo protótipo.

O primeiro teste realizado foi com relação às informações de contactores e defeitos que chegavam à CPLD. Desejava-se saber se essas informações estavam sendo corretamente passadas à comunicação serial (RS232). Para tanto foi utilizado um programa chamado LOOKRS232 que mostra ao usuário aquilo que está na RS232, possibilitando que fosse verificado se as informações estavam chegando corretamente por meio da comunicação serial.

Realizado o primeiro teste, faltava testar se as funções implementadas em linguagem C para a reta e para o arco estavam de acordo com o esperado. Assim, utilizando-se do programa MPLAB, foram testadas várias retas e vários arcos de círculo em algumas peças.

Durante esses testes percebeu-se que seira necessário corrgir a posição do motor X2 com relação à do motor X1, uma vez que frenquentemente eles se encontravam defasados. Para isso foi necessário a criação de uma função CorrigirX2X1 que, no fim do deslocamento, corrigia a posição do motor X2 com relação à posição do motor X1.

Outro problema encontrado foi que ao fim de cada deslocamento a ferramenta de usinagem ficava muito próxima mas não no ponto desejado. Assim, foi imposto que ao fim de cada deslocamento iríamos alcançar o ponto desejado fazendo um deslocamento linear até esse ponto.

Uma vez que tudo foi testado a nova placa eletrônica foi validada e as antigas placas foram recolocadas em seu lugar uma vez que faltava ser feito um programa do usuário para a nova placa.

3 RESULTADOS E PERSPECTIVAS

3.1 Trabalho Realizado

Ao final do estágio uma nova placa eletrônica havia sido desenvolvida para controlar a mesa de usinagem XYZ. Uma visão geral do sistema é mostrado na figura 25. Nela vemos três grandes blocos : PC (posto de comando), Placa Eletrônica e a Mesa de Usinagem.

O estágio objetivou o desenvolvimento da interface entre o PC e a Mesa de Usinagem, ou seja, a Placa Eletrônica. Abaixo na figura 26 está detalhado o bloco da Placa Eletrônica. Nela podemos ver que existem dois blocos principais: o microcontrolador (PIC) e a CPLD (Complex Programable Logic device). Pode ser visto também pelas setas que esses dois blocos trocam informações entre si.

O PIC é a central de inteligência da nova placa. Dele partem as informações que controlam os motores: sentido de avanço e quantos passos deve avançar. Essas informações vão diretamente à CPLD e lá são tratadas antes de serem enviadas aos MIPs (Módulos de comando dos motores), que, por sua vez, enviam aos motores.

As informações são recebidas pelo PIC a partir de uma comunicação série (RS232) e também a partir dessa comunicação são enviadas ao PC as informações de que a ferramente chegou ao fim do curso (contactores) e problemas ocorridos nos motores ou na mesa (defeitos). A RS232 é a comunicação entre o programa do usuário a ser criado e a placa eletrônica.

Existe, porém, uma comunicação paralela (LPT1) entre o PC e a placa eletrônica. Essa comunicação foi colocada pensando na possível utilização de um programa chamado NINOS que realiza usinagens a paritr de um ambiente gráfico. Como esse programa se utiliza de comunicação paralela para controlar os motores e receber informações de contactores e defeitos, uma comunicação paralela foi disponibilizada caso fosse interessante usar o NINOS. Para isso foi colocada uma chave seletora que iria informar à CPLD que comandos deveriam ser enviados aos motores (da comunicação serial ou paralela).

Os contactores e defeitos da mesa de usinagem seriam diretamente enviados à CPLD e tanto seriam enviados ao PC via PIC como essas informações seriam decisivas para possibilitar ou não o envio de comandos de avanço aos motores.

Por fim, os codificadores óticos acoplados aos eixos dos motores iriam enviar as informações de que passos estavam sendo realizados ao PIC e este, a partir de funções de contagem implementadas, iria mostrar através de displays da caixa que continha a placa o número de passos realizados.

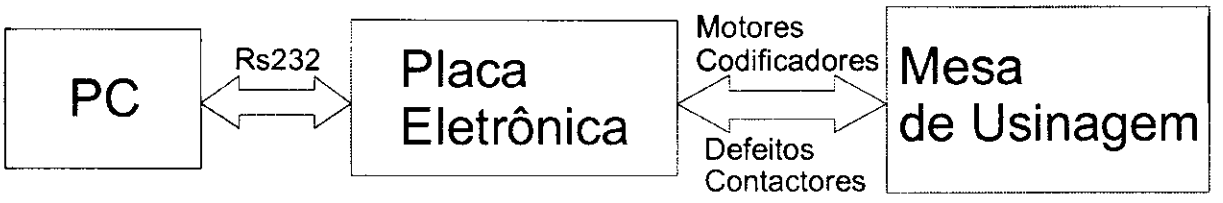

Figura 25 - Visão geral do sistema PC-Placa eletrônica-Mesa de usinagem

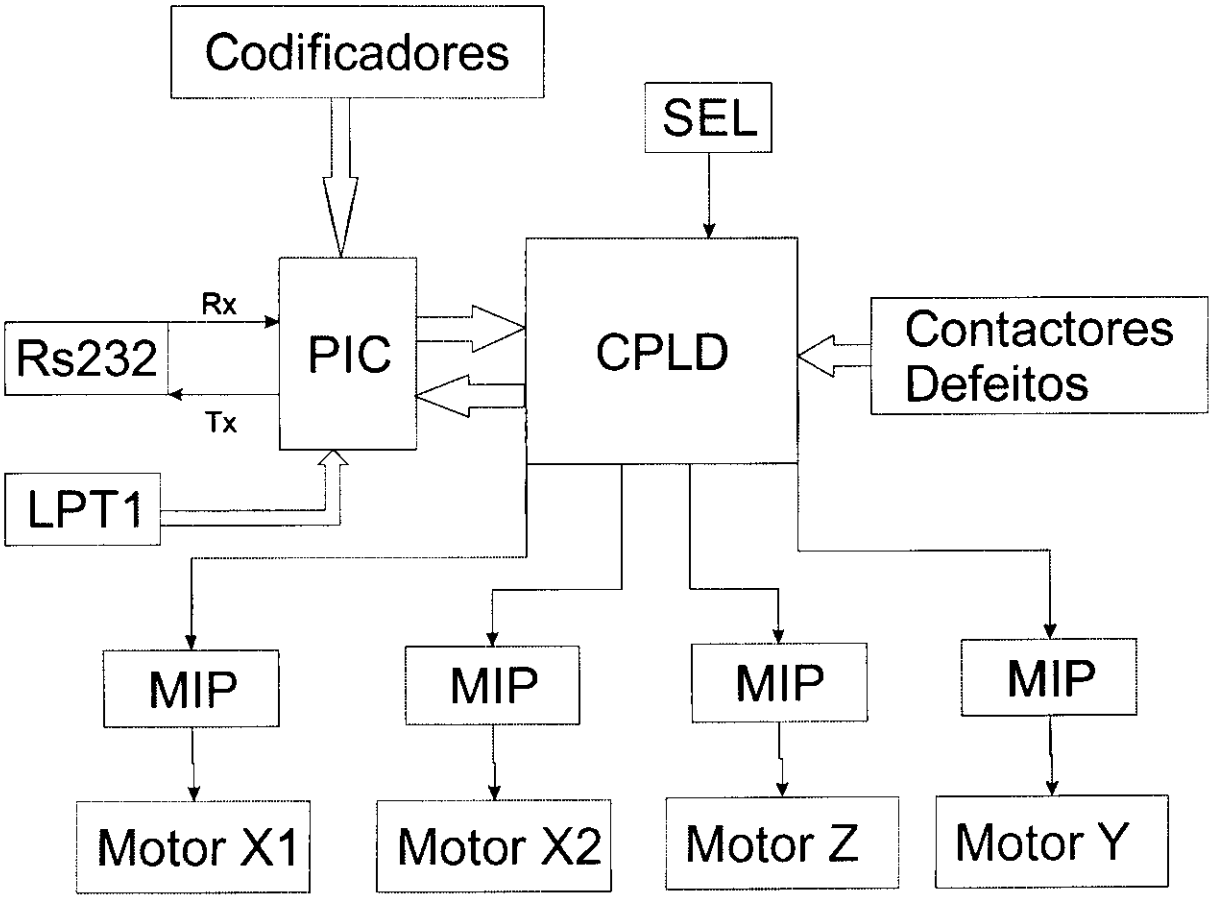

Figura 26 – Diagrama detalhado do bloco da Placa Eletrônica

4 CONCLUSÃO

O estágio apresentado neste relatório foi minha experiência profissinal na empresa LIM S.A., em Lyon na França. Além de me proporcionar um aprendizado em termos de vivência dentro de um ambiente de trabalho, um aperfeiçoamento da língua francesa e a oportunidade de aprofundar meus conhecimentos em controle de sistemas e eletrônica, o estágio também foi aproveitado como uma disciplina no Curso de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande.

Apesar do hardware da placa eletrônica e do software embutido associado já estarem feitos, para o completo funcionamento do sistema ainda falta fazer um programa de interação com o usuário e o protocolo de comunicação série. Isto, junto com o que foi feito, completará o projeto do sistema e poderia começar a se pensar em uma nova versão incluindo uma função helicoidal e uma função para fazer arcos nos planos XZ e YZ.

5 REFERÊNCIAS BIBLIOGRAFICAS

[1] [G-Code] 'Manuel de programmation vol.1 et 2', NUM, 1996.

[2] [MIP] 'Manuel utilisateur du module de commande pas a pas et demi-pas', Midi ingénierie.

[3] [T-verter] 'Variateurs T-verter', Sermes, Juin 1997.

[4] Bresenham, J.E., "Algorithm for computer control of a digital plotter", IBM systems journal, Vol. 4, No 1, 1965.

[5] [PIC] 'PIC18FXX8 Data Sheet', Microchip, 2003.

[6] [Eagle] 'Manuel de référence pour Linux et Windows', Cadsoft, 2000, http://www.cadsoftfrance.com.

[7] [Galaxy] 'User's Guide', Warp HDL development system, Mai 2002.

[8] [Galaxy] 'HDL Reference Manual', Warp HDL development system, 2002.

[9] Ray Andraka, "A survey of CORDIC algorithms for FPGAs," FPGA '98, Proceedings of the 1998 ACM/SIGDA sixth international symposium on Field programmable gate arrays, Monterey, CA, Feb. 1998, pp191-200.

[10] [MPLAB] 'MPLAB C18 C Compiler User's Guide', Microchip, 2004.

6 ANEXOS

5.1 G-code

- **Resumo G-code**

G00 : Interpolação linear em velocidade rápida
G01 : Interpolação linear em velocidade programada
G02 : Interpolação circular sentido anti-trigonométrico em velocidade programada
G03 : Interpolação circular sentido trigonométrico em velocidade programada
G04 : Temporisação programada
G40 : Anulação de correção de raio
G41 : Correção de raio à esquerda do perfil à usinar
G42 : Correção de raio à direita do perfil à usinar
G45 : Ciclos simples
G59 : Defasamento da origem programada
G77 : Chamada de sub-programa
G79 : Salto incondicional ou condicional a uma sequência sem retorno
G81 : Ciclo de perfuração centralizada
G90 : Programação absoluta com relação à origem programada
G91 : Programação relativa com relação ao ponto de partida do bloco
G97 : Velocidade de rotação do motor expressa em voltas/minuto

- **Resumo M-code**

M02 : Fim do programa
M03 : Rotação da ferramenta sentido anti-trigonométrico
M04 : Rotação da ferramenta sentido trigonométrico
M05 : Parar rotação da ferramenta

5.2 Equações realizadas pela CPLD

As equações abaixo apresentadas foram simuladas na CPLD. Como já discutido elas tem como função passar informações ao PIC, interromper o comando dos motores em caso de problema, definir o sentido de rotação dos motores, selecionar de onde virá o comando, etc.

```
bxp =
     bxb * /sel
   + bxa * /sel

byp =
     byb * /sel
   + bya * /sel

bzp =
     bzb * /sel
   + bza * /sel

dfp =
     /sel * urg

enm =
     rc(0) * /urg
   + /sel * /urg

hx =
     /bxb * dx1 * dx2 * hxc * sel * sxc
   + /bxb * dx1 * dx2 * hxp * /sel * sxp
   + /bxa * dx1 * dx2 * hxc * sel * /sxc
   + /bxa * dx1 * dx2 * hxp * /sel * /sxp

hx2 =
     /bxb * dx1 * dx2 * hx2c * sel * sx2c
```

```
   + /bxa * dx1 * dx2 * hx2c * sel * /sx2c

hy =
     /byb * dy * hyc * sel * syc
   + /byb * dy * hyp * /sel * syp
   + /bya * dy * hyc * sel * /syc
   + /bya * dy * hyp * /sel * /syp

hz =
     /bzb * dz * hzc * /pal * sel * /szc
   + /bzb * dz * hzp * /pal * /sel * /szp
   + /bza * dz * hzc * sel * szc
   + /bza * dz * hzp * /sel * szp

pap =
     pal * /sel

ra(3) =
     /dy * /ra(0) * /ra(1) * ra(2)
   + ra(0) * ra(1) * /ra(2) * syn
   + bzb * ra(0) * /ra(1) * /ra(2)
   + bya * /ra(0) * /ra(1) * /ra(2)

ra(4) =
     /dninosx2 * ra(0) * /ra(1) * ra(2)
   + /dx2 * /ra(0) * /ra(1) * ra(2)
   + ra(0) * ra(1) * /ra(2) * tvt
   + bza * ra(0) * /ra(1) * /ra(2)
   + bxb * /ra(0) * /ra(1) * /ra(2)

ra(5) =
     /dz * ra(0) * /ra(1) * ra(2)
   + /dx1 * /ra(0) * /ra(1) * ra(2)
   + ra(0) * ra(1) * /ra(2) * urg
```

```
    + pal * /ra(0) * ra(1) * /ra(2)
    + byb * ra(0) * /ra(1) * /ra(2)
    + bxa * /ra(0) * /ra(1) * /ra(2)

s_asp =
      rc(2) * sel
    + asp * /sel

s_tvt =
      rc(1) * /urg
    + /sel * /urg

sx =
      sel * sxc
    + /sel * sxp

sx2 =
      sel * sx2c

sy =
      sel * syc
    + /sel * syp

sz =
      sel * szc
    + /sel * szp
```