UNIVERSIDADE FEDERAL DE CAMPINA GRANDE

CENTRO DE CIÊNCIAS E TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

TRABALHO DE CONCLUSÃO DE CURSO

ORIENTADOR: LUÍS REYES ROSALES MONTERO, Dr.

ALUNO: VINÍCIUS DE SOUZA MELO

TÍTULO:

"DESENVOLVIMENTO DE UM REGULADOR DIGITAL DE TENSÃO PARA ENSAIOS DE TRANSFORMADORES DE POTÊNCIA DE DISTRIBUIÇÃO "

Campina Grande, Outubro de 2005, PB – Brasil

UNIVERSIDADE FEDERAL DE CAMPINA GRANDE

CENTRO DE CIÊNCIAS E TECNOLOGIA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

TRABALHO DE CONCLUSÃO DE CURSO

Título:

"DESENVOLVIMENTO DE UM REGULADOR DIGITAL DE TENSÃO PARA ENSAIOS DE TRANSFORMADORES DE POTÊNCIA DE DISTRIBUIÇÃO "

Projeto Final de Conclusão de Curso apresentado à coordenação do curso de Engenharia Elétrica da UFCG como parte dos requisitos necessários à obtenção do título de Engenheiro Eletricista

ORIENTADOR: LUÍS REYES ROSALES MONTERO, Dr.

ALUNO: VINÍCIUS DE SOUZA MELO

Campina Grande, Outubro de 2005, PB – Brasil

Biblioteca Setorial do CDSA. Fevereiro de 2021.

Sumé - PB

"DESENVOLVIMENTO DE UM REGULADOR DIGITAL DE TENSÃO PARA ENSAIOS DE TRANSFORMADORES DE POTÊNCIA DE DISTRIBUIÇÃO "

por

VINÍCIUS DE SOUZA MELO

Banca examinadora:

LUÍS REYES ROSALES MONTERO, Dr., DEE, UFCG

Orientador

LEIMAR DE OLIVEIRA, DR., DEE, UFCG

Componente da banca

Campina Grande, Outubro de 2005, PB – Brasil

Aos meus pais

Agradecimentos

A meu pai, o engenheiro eletricista, **Nicolau Pequeno de Melo**, dono da empresa Transformadores Campinense, pelo apoio logístico e discussão que culminou no tema do trabalho de conclusão de curso em cima de uma aplicação na empresa.

Ao Professor do DEE-UFCG **Luís Reyes Rosales Montero**, Dr., pela orientação do Trabalho de Conclusão de Curso, coordenando e orientando todas as etapas do trabalho, além de discussões técnicas e correção do texto escrito, bem como pelo seu apoio moral.

Ao técnico **Lúcio Hélio da Costa de Santana**, fabricante da placa de treinamento do PIC (microcontrolador utilizado neste trabalho), pela apoio no início dos estudos sobre o microcontrolador PIC, bem como outros aspectos relevantes no trabalho.

Ao professor do CEFET-PE (curso de eletromecânica) **Alberto Willian Mascarenhas**, Doutorando., pela valorosa contribuição quando do estudo do microcontrolar PIC, além de discussões valorosas sobre o tema, dando imenso impulso ao início dos trabalhos com o microcontrolador.

Ao eng. eletricista, professor do DEM-UFCG, **Pedro Ronaldo H. de Holanda**, Doutorando, pelas importantes discussões sobre problemas que surgiram no projeto eletrônico, além de discussões em geral sobre o projeto, ajudando a resolver efetivamente um grande problema do circuito eletrônico.

Ao professor do DEM-UFCG, **Cícero da Rocha Souto**, Doutorando., pelas discussões em geral sobre os problemas de eletrônica e mecânica que iam surgindo, bem como na documentação do trabalho (fotos e filmagens).

Ao professor do DEE-UFCG, **Alexandre Cunha Oliveira**, Dr., pelas discussões a cerca de microcontroladores.

Ao professor do DEE-UFCG, **Leimar de Oliveira**, Dr., pelas importantes contribuições na revisão do texto escrito e apoio.

Ao professor do DEE-UFCG, **Edison Roberto C. da Silva**, Dr., pelas discussões a cerca de eletrônica de potência.

Ao aluno de eng. elétrica, **Adriano Magno Rodrigues da Silva**, que iniciou os estudos de microcontroladores comigo e incentivou-me na continuidade do mesmo.

Ao aluno de eng. mecânica, **Janaílson Oliveira Cavalcante**, que auxiliou nas fotografias do protótipo.

RESUMO

Este trabalho trata da construção de um regulador digital de tensão para ensaios de transformadores de distribuição com auto-transformador de 10 kVA (trifásico) e de pequenos transformadores com variac de 150 VA (monofásico), a partir da utilização de um microcontrolador PIC 16F877, programado com a linguagem C++, que controla um motor CC que por sua vez desloca contatos elétricos no auto-transformador trifásico, ou no variac (monofásico), conforme a necessidade. A não utilização de eletrônica de potência através de tiristores (por exemplo) se justifica pela exigência em normas de ensaios por uma onda senoidal pura na realização dos mesmos.

Foi construído um sistema de ajuste de tensão com interface homem-máquina usando teclado e LCD (display) onde o usuário digita a tensão requerida e acompanha o valor em tempo real através do LCD.

Foram realizados testes experimentais em transformadores de 10 kVA a 75 kVA (ensaio de curto-circuito e circuito aberto) com êxito a partir do protótipo construído.

O sistema mostrou-se consistente na automação do ensaio de transformadores para obtenção rápida de laudos de transformadores exigidos pelas concessionárias para instalação dos mesmos.

Palavras-Chave: controle de tensão, microcontrolador PIC 16F877, motor cc, auto-transformador, regulador digital.

"A termodinâmica clássica, devido à simplicidade de suas premissas e ao grande número de assuntos a que se aplica, é a única teoria física de conteúdo geral sobre a qual estou convencido que dentro dos limites de aplicação de seus conceitos fundamentais nunca será refutada"

Einstein.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

CAPÍTULO 1
INTRODUÇÃO

CAPÍTULO 1 - INTRODUÇÃO

1.1 MOTIVAÇÃO

As empresas concessionárias de distribuição de energia elétrica exigem um laudo técnico do transformador novo ou remanufaturado (ver anexo 6). Para poder instalá-lo, são exigidos dados como relação de tensão, impedância e perdas. Estes dados são calculados a partir de ensaios de curto-circuito e circuito aberto, além de ensaios para se verificar a qualidade do óleo isolante do transformador, entre outras.

No mercado já existem equipamentos de alto nível de automação para realizar ensaios de transformadores. Entretanto devido ao seu alto custo, só os fabricantes os possuem. As empresas de remanufatura necessitam desenvolver suas próprias soluções em termos de equipamentos para realizar seus ensaios.

O objetivo deste trabalho foi iniciar um processo para a busca da automação dos ensaios de transformadores na empresa. Para isso a Empresa de Transformadores Campinense, de propriedade do eng. eletricista Nicolau Pequeno de Melo, localizada no distrito dos mecânicos em Campina Grande, PB, se dispôs a associar-se neste estudo dando todo o suporte para construção do produto alvo deste trabalho que é utilizado pelo laboratório de ensaios da empresa para os transformadores remanufaturados (manutenção) a um custo mais baixo e de maneira mais rápida, buscando automatizar cada vez mais o processo.

1.2 PROPOSTA

O trabalho foi desenvolvido junto à empresa Transformadores Campinense, com o intuito de criar um dispositivo para controlar a tensão em um auto-transformador trifásico de 10 kVA para ensaio de transformadores de distribuição ou controlar a tensão em um variac de 150 VA, utilizado para ensaios de transformadores monofásicos de pequeno porte.

A Fig. 1.1 ilustra esquematicamente o sistema proposto para automatizar o ensaio dos transformadores

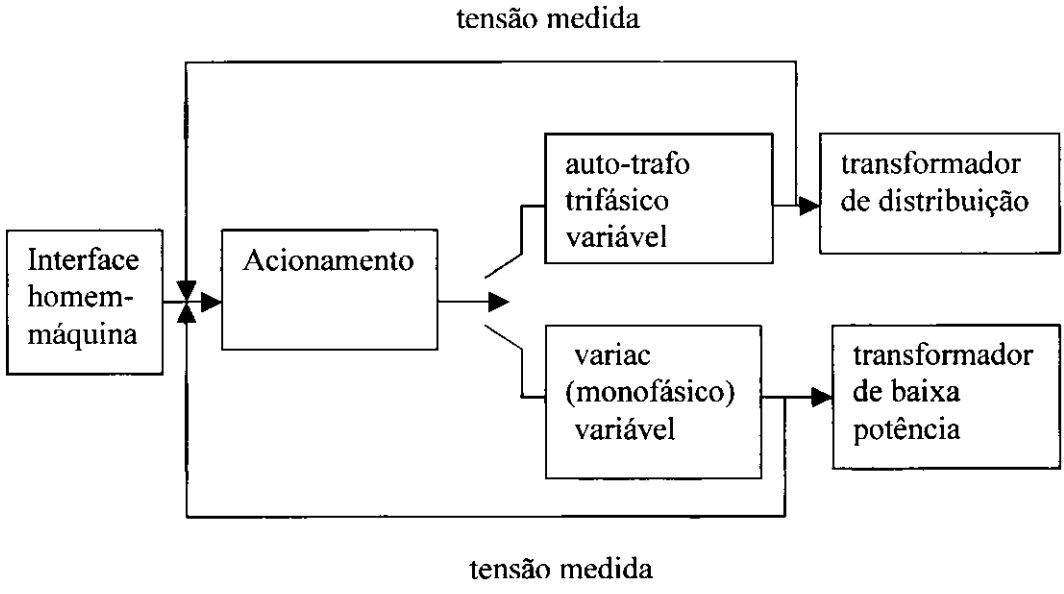

Figura 1.1 - Esquema do dispositivo para ensaio de transformadores

Observe que a partir da interface homem-máquina, através de um teclado e LCD (display) se digita a tensão requerida e se acompanha em tempo real a tensão medida no auto-transformador trifásico ou no variac (conforme o caso), com contatos elétricos móveis se pode acionar um ou outro. A regulagem da tensão é feita por contatos móveis que são acionados a partir de um motor CC que por sua vez é controlado conforme a tensão atual por um sistema de relés que fazem o mesmo girar em sentido horário ou anti-horário, aumentando ou diminuindo a tensão requerida.

Usou-se o microcontrolador PIC 16F877 como o cérebro da aquisição de dados e controle do sistema. Um motor cc é utilizado em série com redutores de velocidade mecânico e transmissão via correias, para movimentar contatos elétricos a base de carvão e fazer assim variar a tensão senoidal pela variação das espiras entre secundário e primário do auto-transformador ou do variac, conforme o caso de se ensaiar transformadores de distribuição ou transformadores de pequeno porte, sem os inconvenientes das harmônicas provocadas quando da utilização de eletrônica de potência com tiristores, visto que com o corte abrupto da onda, com fins de controlar a potência geram-se harmônicos que não são recomendados pela norma de ensaios de transformadores, que indica a utilização de onda senoidal pura.

Na Figura 1.2 está o esquema da realimentação da informação sobre a tensão e a correspondente tomada de decisão para fazer o motor CC girar no sentido horário, anti-horário ou parar, caso se queira aumentar, diminuir ou manter a tensão no auto-transformador trifásico ou no variac conforme o caso requerido.

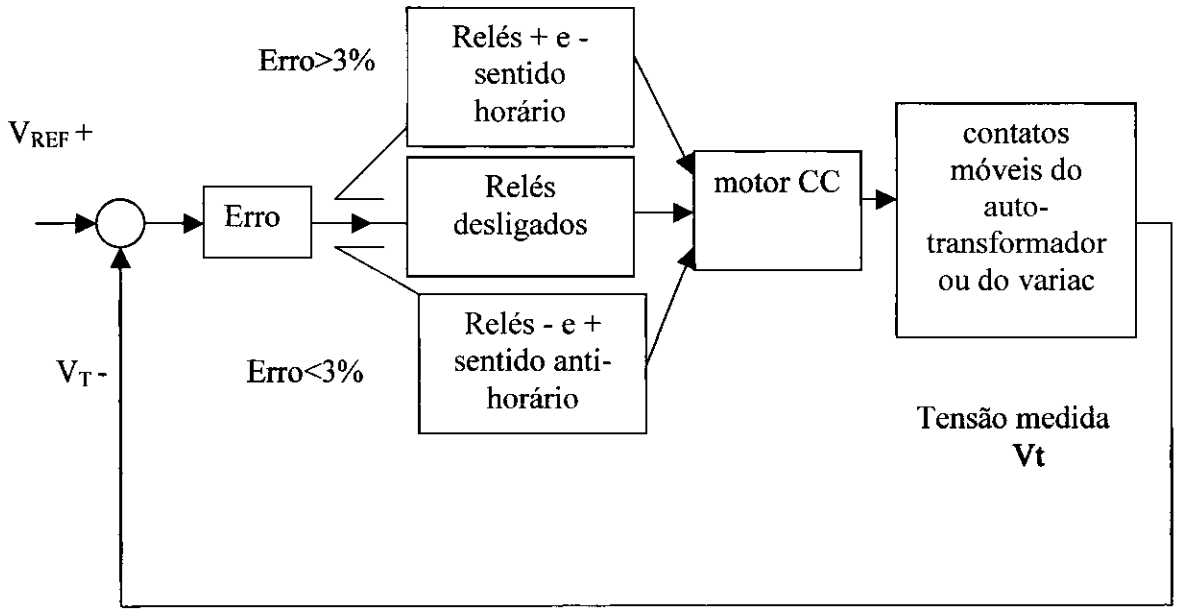

Figura 1.2 - Esquema da tomada de decisão para variar a tensão no auto-transformador ou variac (conforme o caso) através do motor CC.

Ao longo do trabalho serão abordadas todas as etapas desde o conhecimento do microcontrolador, passando pela linguagem de programação C++ utilizada na

elaboração do código do programa do PIC, até a como gravar o programa no mesmo a partir do microcomputador. O esquema elétrico da placa com sua respectiva construção, até os detalhes da construção da fonte de alimentação do motor cc utilizado e do sistema de transmissão de movimento.

O capítulo 2 trata da interface homem-máquina, onde foi desenvolvido com um teclado, onde se digita uma tensão de referência desejada e um LCD (display) onde se acompanha em tempo real a tensão do regulador.

O capítulo 3 trata dos resultados experimentais, consistindo da obtenção de cada etapa da construção do produto, do protótipo eletro-mecânico, desde a alimentação do motor cc até a estrutura de sustentação das partes do protótipo e transmissão do movimento via correias e redutores de velocidade, e demais etapas, como o desacoplamento elétrico do variac e a ligação do auto-transformador trifásico, caso se queira controlar um ou outro.

Já o capítulo 4 trata das conclusões e sugestões para trabalhos futuros.

CAPÍTULO 2

INTERFACE HOMEM-MÁQUINA

CAPÍTULO 2 - INTERFACE HOMEM-MÁQUINA

Neste capítulo é abordado a interface homem-máquina utilizada no projeto, é de imensa importância que o usuário consiga de forma rápida e eficiente fazer com que o dispositivo seja programado para a aplicação requerida e possa ser acompanhado ao longo de sua utilização.

A interface consiste na possibilidade do usuário utilizar-se de um teclado para digitar a tensão de referência requerida e um LCD (display) que apresenta em tempo real a tensão medida do regulador. Esta interface é controlada pelo microcontrolador.

Na sequência detalham-se dados sobre o teclado utilizado e a respectiva subrotina de leitura do mesmo.

2.1 - LEITURA DE UM TECLADO

Para resolver o problema do teclado, de um modo geral, basta observar o nível lógico associado a uma tecla. Caso ela esteja pressionada, teremos um determinado nível, caso esteja solta, outro. De acordo com a tecla pressionada a associamos a determinados pinos do microcontrolador e a uma tabela com os valores correspondentes a tecla pressionada.

Para se economizar pinos do microcontrolador quando o número de teclas aumenta, utiliza-se um sistema de varredura, de uma estrutura em matriz, formada por linhas e colunas, fixando uma varredura ao fixar uma linha e percorrer as colunas. Repetindo-se o procedimento para as outras linhas consegue-se isolar a tecla pressionada.

A Figura (2.1) representa o esquema de ligação de um teclado típico de 3 colunas por 4 linhas, ocupando 7 pinos do microcontrolador. No nosso projeto foi

utilizado um teclado de telefone ocupando 8 pinos do mesmo. Na mesma figura está ilustrado a ligação a um LCD (Display) e uma transmissão serial via MAX 232 (amplificando os sinais do pic para uma entrada via entrada de mouse de computador por exemplo).

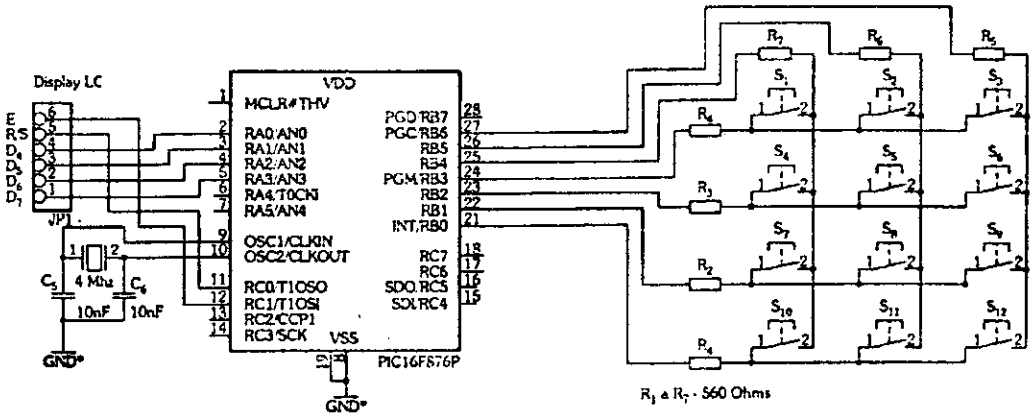

Figura 2.1 - Diagrama de ligação esquemático de um teclado de 12 teclas no sistema linhas e colunas, associado a um LCD

A seguir aborda-se a parte da rotina referente à leitura do teclado. Esta rotina referente ao programa principal do microcontrolador encontra-se no anexo 2.

Verifica-se que ao acionar uma tecla ela fecha um circuito entre dois pinos do microcontrolador. Em um dos pinos o programa se encarrega de colocar em nível alto (5V), ao se apertar uma tecla este nível de tensão passa para o outro pino que é lido pelo microcontrolador e dependendo do pino que recebe a tensão de 5 v o programa se encarrega de dizer a que número de 0 a 9 se trata, ou outro valor como um nome, tipo tecla enter, por exemplo.

Primeiro se faz a definição dos pinos do microcontrolador que estão ligados ao teclado.

// Definições da matriz do teclado

```
#define fio0    pin_d0
#define fio1    pin_d1
```

```
#define fio2        pin_d2
#define fio3        pin_d3
#define fio4        pin_d4
#define fio5        pin_d5
#define fio6        pin_d6
#define fio7        pin_d7
```

A seguir se inicia a rotina, que é chamada a um certo intervalo de tempo para verificar se a tecla está sendo pressionada, a velocidade de varredura á tal que ao se apertar a tecla o programa sempre varre todos os pinos referentes ao teclado e determina qual está sendo pressionada.

```
char varre_teclas (void)

/* Realiza a varredura do teclado e retorna o
 valor da tecla pressionada.
 O teclado está conectado da seguinte forma:
```

O seguinte esquema é usado para identificar os nomes referentes à tecla pressionada, ou seja, a tecla chamada fio0 se estiver em nível alto e ao ser pressionada o nível alto fique no fio7 o nome da tecla é o número 1.

```
//      FIO0   FIO1   FIO2   FIO3   FIO4   FIO5   FIO6   FIO7
// FIO0   -      -      -      -      -      3      2      1
// FIO1   -      -      -      -      F      6      5      4
// FIO2   -      -      -      -      9      8      7
// FIO3   -      -      -      -      R      #      0      *-M
```

Os valores retornados pelo programa são conforme abaixo, a tecla 0 retorna o valor 48, a tecla 1 retorna o valor 49 e assim por diante.

```
 Os valores decimais retornados são:
```

```
0  48
1  49
2  50
3  51
4  52
5  53
6  54
7  55
8  56
9  57
*/
{
```

Ao colocar-se um pino em nível alto e os outros em nível baixo, a seguir se lê os 4 pinos restantes e caso um deles esteja em nível alto, conforme a tabela anterior se determina a que número ou letra corresponde a tecla pressionada. O processo se repete para cada um dos 4 pinos a que se vão colocar o nível alto.

```
  int tecla;
  tecla=' ';

  output_high(fio0); // ativa o fio 1
  output_low(fio1);  // desativa o fio 2
  output_low(fio2);  // desativa o fio 3
  output_low(fio3);  // desativa o fio 4
  output_low(fio4);  // desativa o fio 5
  output_low(fio5);  // desativa o fio 6
  output_low(fio6);  // desativa o fio 7
  output_low(fio7);  // desativa o fio 8
  if (input(fio4)) tecla = 'R';
  if (input(fio5)) tecla = ' ';
```

```
  if (input(fio6)) tecla = ' ';
  if (input(fio7)) tecla = ' ';
  output_low(fio0);  // desativa o fio 1
  output_high(fio1); // ativa o fio 2
  if (input(fio4)) tecla = '*';
  if (input(fio5)) tecla = '7';
  if (input(fio6)) tecla = '1';
  if (input(fio7)) tecla = '4';
  output_low(fio1);  // desativa o fio 1
  output_high(fio2); // ativa o fio 2
  if (input(fio4)) tecla = '0';
  if (input(fio5)) tecla = '8';
  if (input(fio6)) tecla = '2';
  if (input(fio7)) tecla = '5';
  output_low(fio2);  // desativa o fio 1
  output_high(fio3); // ativa o fio 2
  if (input(fio4)) tecla = '#';
  if (input(fio5)) tecla = '9';
  if (input(fio6)) tecla = '3';
  if (input(fio7)) tecla = '6';//M
  return (tecla);
}
```

Na Figura 2.2 é mostrado a vista frontal do protótipo com o teclado de telefone utilizado e o LCD ao lado.

Na sequência detalham-se dados sobre o LCD utilizado e a respectiva subrotina de interface com o microcontrolador.

2.2 - O LCD

O LCD é um display que pode ser usado como interface entre o microcontrolador e o usuário.

A Figura (2.3) apresenta alguns tipos de LCD utilizados, basicamente se classificam de acordo com o número de colunas e de linhas dos mesmo,

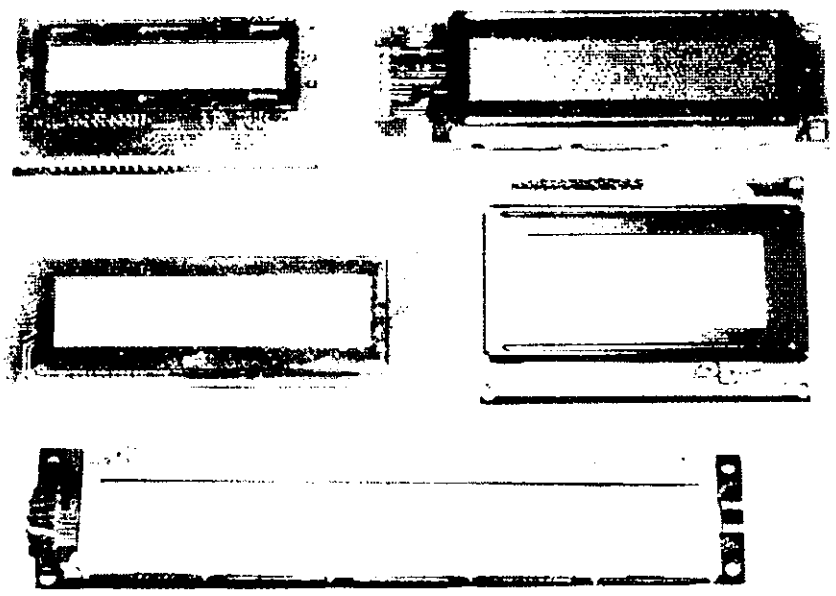

Figura 2.3 - Tipos de LCD mais utilizados.

O esquema de ligação do LCD ao microcontrolador está ilustrado na Fig. (2.4).

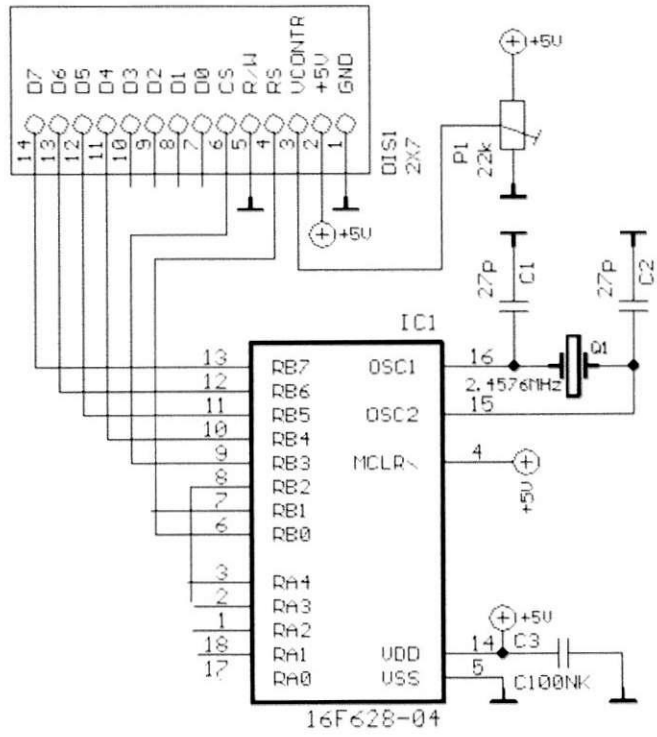

Figura 2.4 - Esquema de ligação entre o LCD e o microcontrolador.

Na Tabela (2.1) está ilustrado as respectivas funções dos pinos do LCD.

Tabela 2.1 - Configuração dos pinos do LCD

Pin No.	Name	Function
1	V_{ss}	Ground
2	V_{dd}	+ve supply
3	V_{ee}	Contrast
4	RS	Register Select
5	R/W	Read/Write
6	E	Enable
7	D0	Data bit 0
8	D1	Data bit 1
9	D2	Data bit 2
10	D3	Data bit 3
11	D4	Data bit 4
12	D5	Data bit 5
13	D6	Data bit 6
14	D7	Data bit 7

A Tabela 2.1 indica como deve ser ligado os pinos do LCD com o microcontrolador, em função desta deve-se programar o mesmo para que os determinados pinos estejam enviando sinais de controle ou dados para o LCD, tal preocupação em programar esta interface é amenizada, pois existem já prontos no compilador da CCS, ou em livros sobre programação em linguagem C de microcontroladores da família PIC.

CAPÍTULO 3

RESULTADOS EXPERIMENTAIS

CAPÍTULO 3 - RESULTADOS EXPERIMENTAIS

Neste capítulo serão mostrados as etapas da construção da parte eletrônica do protótipo para controle de tensão.

3.1 - CIRCUITO ELETRÔNICO

O circuito será apresentado em módulos, explicando suas respectivas funções, em seguida apresenta-se o circuito como um todo.

Na Figura 3.1 apresenta-se a alimentação da placa, o desenho foi feito no software PCAD, utilizado para desenhar circuitos impressos.

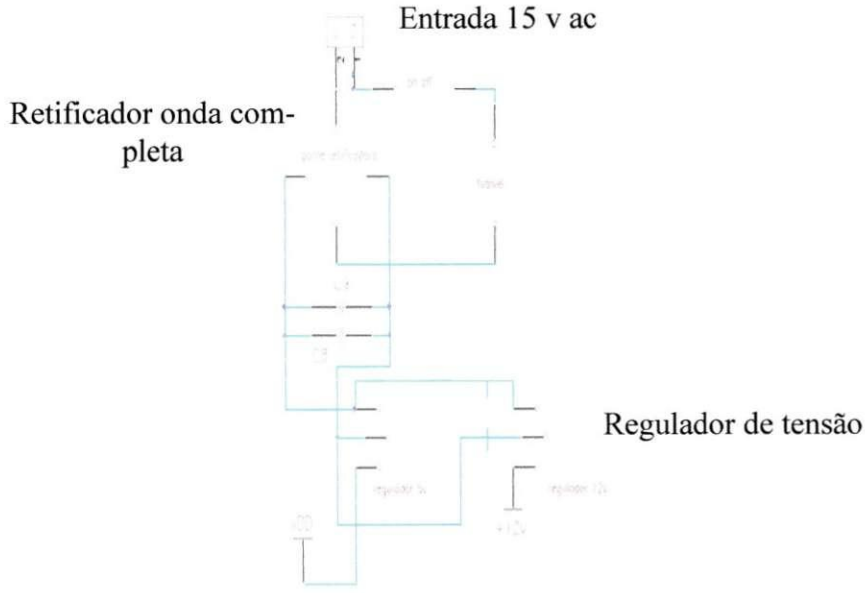

Figura 3.1 - Esquema da alimentação da placa

Observa-se que a partir dos bornes superiores, de onde se liga o secundário de um transformador de 220/15V de 5W de potência, em seguida tem-.se uma ponte retificadora de onda completa, associada com capacitores de 470 μF, sendo que foi ligado em série com uma das fases um fusível, e na sequência dois reguladores de tensão para 5V e 12V, sendo que a tensão de 5 v se destina à alimentação do microcontrolador e a de 12V se destina aos relés que serão usados como chaveadores para alimentação do motor CC.

Na sequência, tem-se a Fig. 3.2 que apresenta a alimentação dos relés de chaveamento. É utilizado transistores para chavear a bobina dos relés, que estão ligadas diretamente aos 12V, a base do transistor é alimentada diretamente por um dos pinos do microcontrolador, no nível de 5 v ou 0 v, nível alto ou baixo, respectivamente. os resistores utilizados são de 1 $k\Omega$ e de 10 $k\Omega$. O transistor é do tipo TBJ.

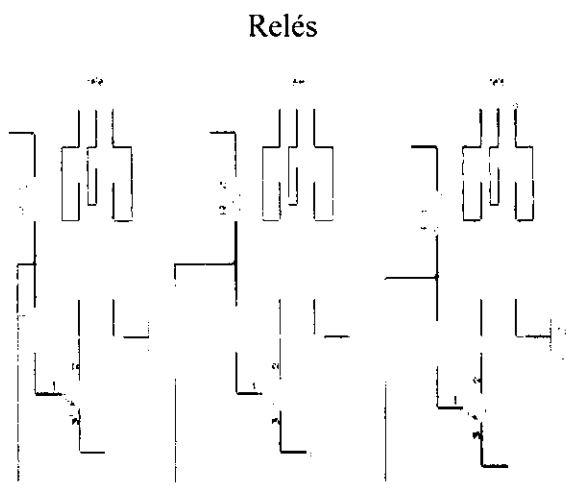

Figura 3.2 - Alimentação e controle dos relés de chaveamento do motor CC.

A Figura 3.3 ilustra o esquema do microcontrolador PIC 16F877 com suas respectivas ligações com o LCD (display), e a ligação na parte inferior com o teclado de telefone, com oito fios, observe a ligação dos resistores, para limitar corrente, ao ser pressionado uma tecla, no pino 3 do LCD é ligado um potenciômetro necessário para visualizar os caracteres do LCD, caso contrário ele fica sem apresentar os caracteres.

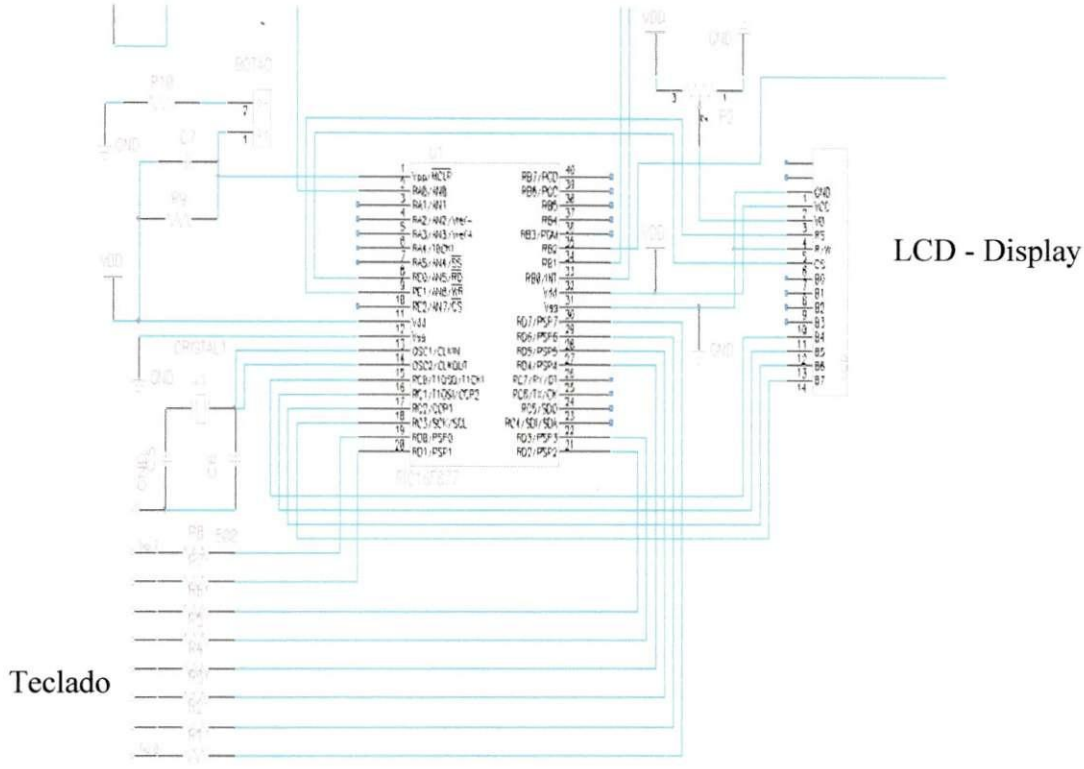

Figura 3.3 - Esquema da alimentação do microcontrolador e ligação com o teclado e LCD..

Observe na parte superior esquerda um botão de reset, utilizado aplicando 5 v de tensão em funcionamento, e 0 v para reinicializar o programa do microcontrolador.

A pinagem do microcontrolador é utilizada para ligar cada componente, o LCD, o teclado, o controle dos transistores dos relés e no pino 2 é ligado a tensão de leitura de tensão, que é rebaixada por um transformador de 350/1,5 v que é aplicada diretamente ao conversor do PIC com uma tensão de pico de aproximadamente 2 v, visto que acima desta tensão, o nível negativo da tensão AC danifica o microcontrolador. O programa do PIC se encarrega de obter a tensão de pico positiva para ser utilizada como parâmetro de controle do motor CC.

A seguir é apresentada na Fig. 3.4 o circuito completo utilizado na construção da placa.

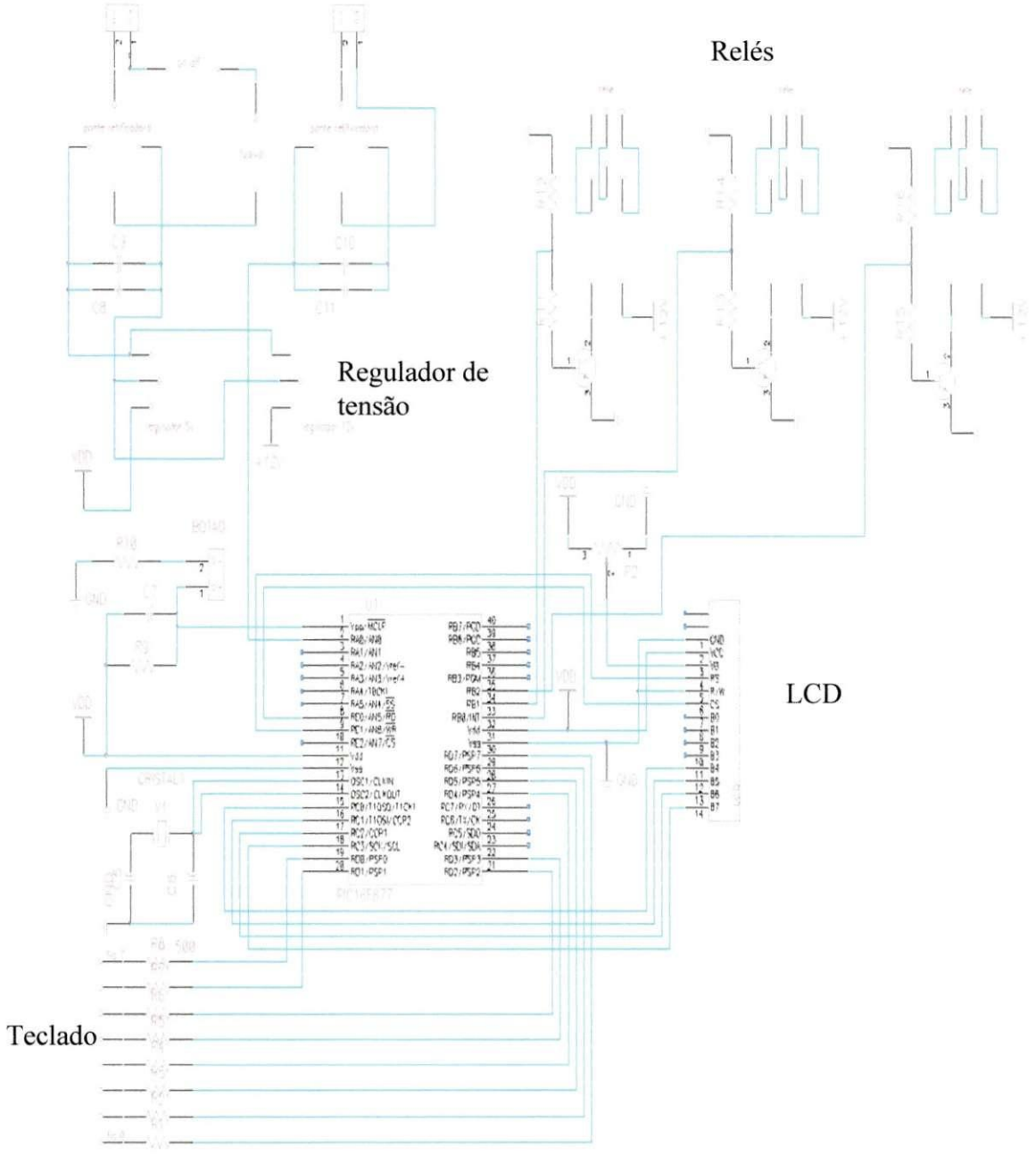

Figura 3.4 - Esquema do circuito total utilizado na confecção da placa.

3.2 - CIRCUITO IMPRESSO

A partir do circuito mostrado acima foi feito pelo PCAD o desenho para serigrafia posterior do circuito impresso, que é de dupla face, devido à grande quantidade de conexões, sem o qual teríamos curto-circuito nas trilhas, conforme mostrado na Fig.3.5, a face posterior do circuito.

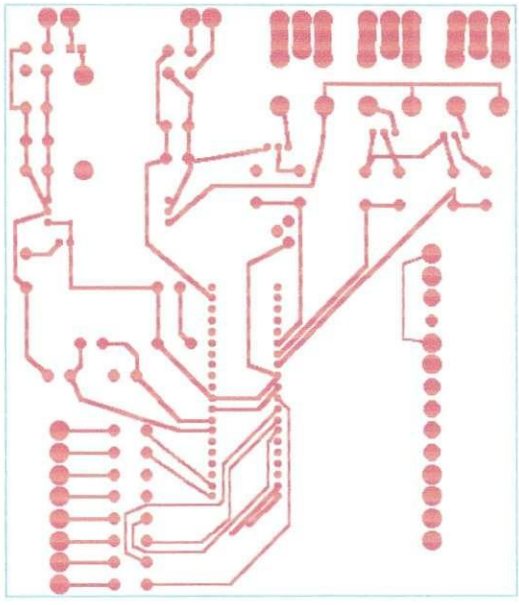

Figura 3.5 - Desenho posterior do circuito impresso.

Tanto o teclado, quanto o LCD e os demais contatos com o exterior da placa é montado em ilhas para posterior soldagem de pinos com bornes de parafuso, para serem soldados no circuito impresso e suas ligações serem feitas por fios.

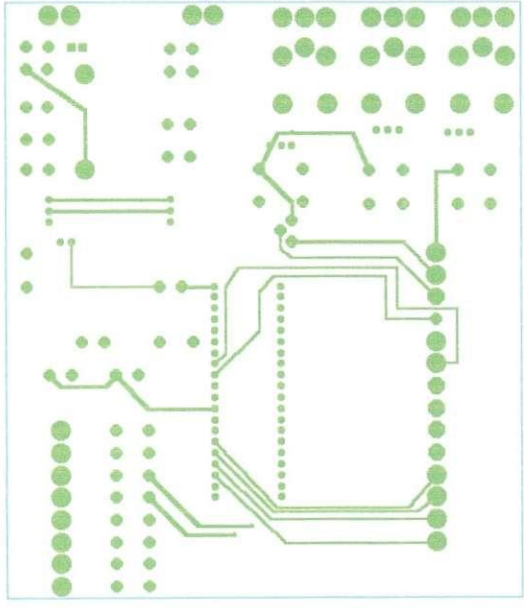

Figura 3.6 - Desenho anterior do circuito impresso.

Os fios que serão presos via parafusos, de modo a que se possa facilmente desacoplar a placa do LCD, teclado e fios de alimentação da placa, da tensão de leitura e da alimentação do motor CC, via relés. A Figura 3.6 apresenta o desenho anterior do circuito impresso.

Na Figura 3.7 está a sobreposição dos dois circuitos, anterior e posterior, apenas para fins de visualização geral do circuito.

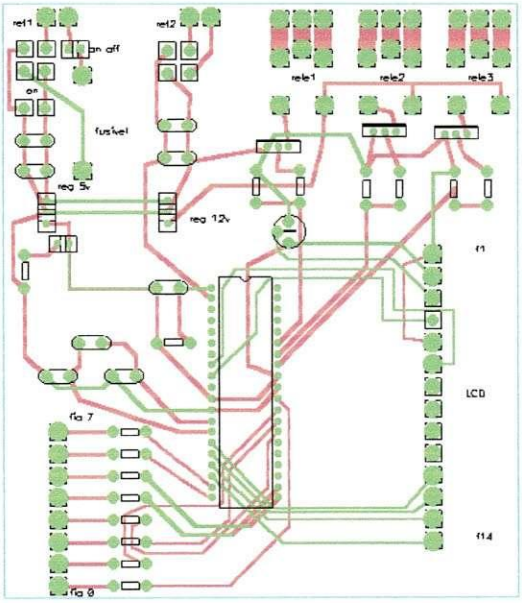

Figura 3.7 - Desenho completo, anterior e posterior do circuito impresso.

Para permitir posteriores modificações em trabalhos futuros, foi acrescentado o desenho ligando os pinos do PIC que não seriam utilizados neste circuito às ilhas para soldagem de bornes de ligação com parafusos, para eventuais usos. A Figura 3.8 é da face posterior do circuito modificado.

Na Figura 3.9 é apresentado a face anterior da mesma placa, a única diferença é o acréscimo de trilhas para os pinos que anteriormente não estavam desenhados, pois não seriam utilizados no circuito ao qual se destinam.

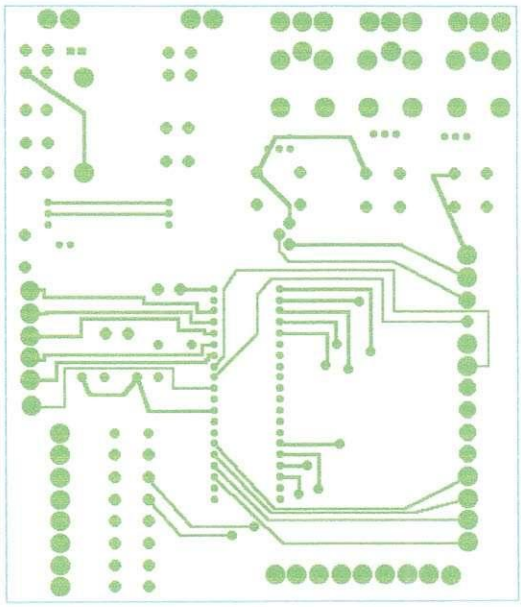

Figura 3.8 - Face posterior da placa modificada.

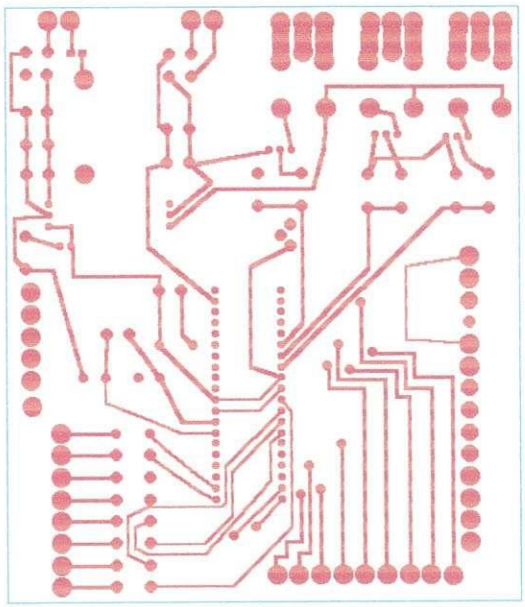

Figura 3.9 - Face posterior da placa modificada.

A Figura 3.10 ilustra o esquema do conjunto, face posterior e anterior da placa, só para fins de visualização geral do circuito.

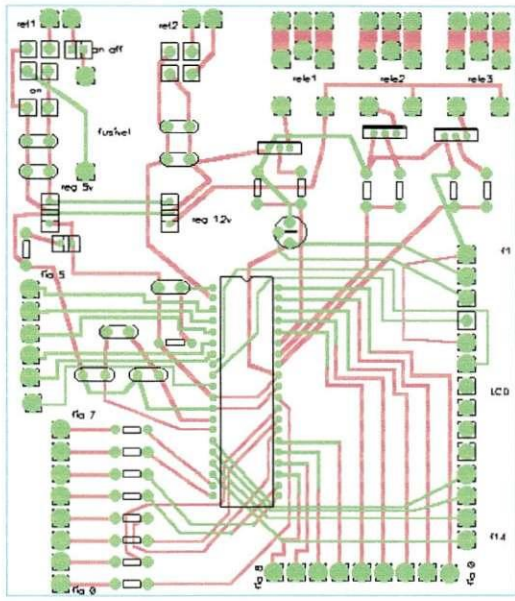

Figura 3.10 - Faces anterior e posterior justapostas da placa modificada.

3.3 - SERIGRAFIA DO CIRCUITO IMPRESSO.

Com base nos desenhos feitos através do software PCAD, foi construída via serigrafia, a tela serigráfica com o desenho das duas faces para ser gravado na placa de cobre, dupla face, e posterior corrosão em ácido, sendo que é utilizado uma tinta serigráfica que é passada com o auxílio de um rodo para em seguida, a tela ser limpa com solvente e a placa de cobre virada para que a outra face seja pintada com o mesmo procedimento

A Figura 3.11 apresenta a tela serigráfica juntamente com o rodo utilizados para imprimir na placa de cobre.

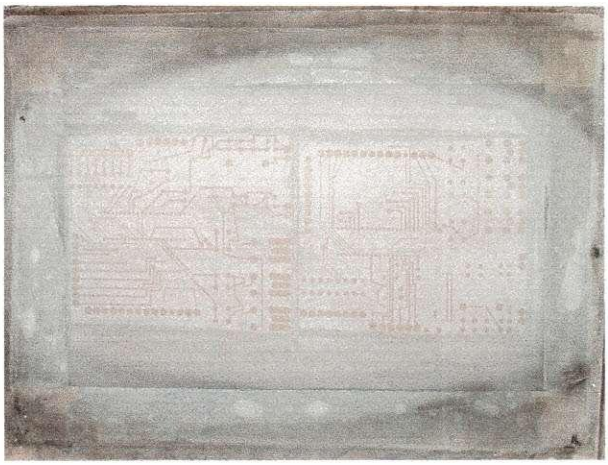

Figura 3.11 - Tela serigráfica utilizada para imprimir na placa de cobre.

A figura 3.12 mostra a placa após serem soldados os componentes, observe que nesta face apresentada estão o microcontrolador, os relés os bornes para ligação do teclado e da fonte de tensão da placa, além do borne para ligação da tensão à ser lida pelo PIC.

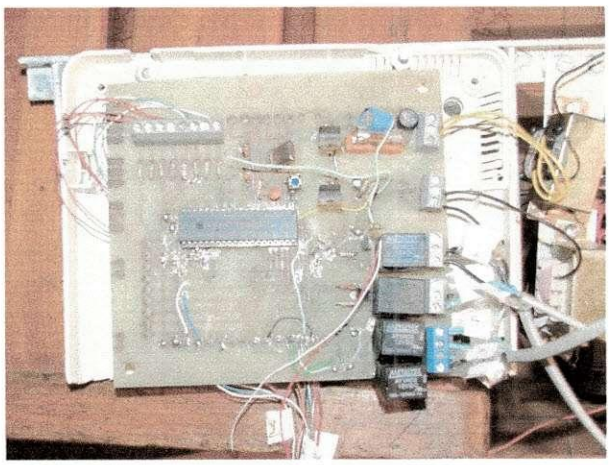

Figura 3.12 - Placa de circuito impresso, face anterior, com os componentes soldados.

A Figura 3.13 mostra a face posterior da placa com os respectivos bornes para ligação do LCD (display gráfico).

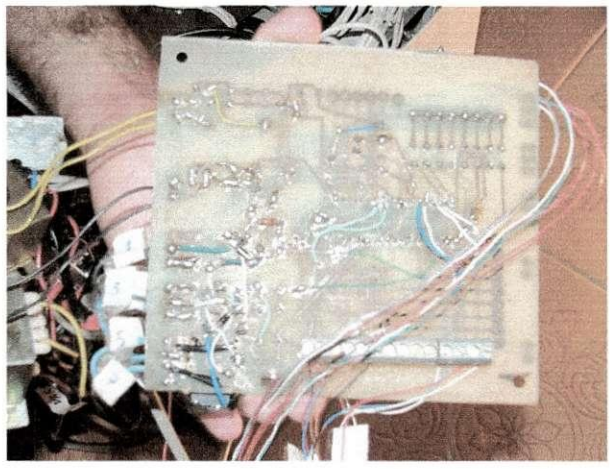

Figura 3.13 - Placa de circuito impresso, face posterior, com os componentes soldados.

A seguir serão mostrados as etapas da construção do protótipo para controle de tensão.

3.4 - CONSTRUÇÃO DA FONTE PARA O MOTOR CC

Inicialmente é apresentado o esquema elétrico para ligação do motor CC, sendo que a bobina que representa o campo do estator, na realidade será um imã permanente. O motor é de limpador de para-brisa de automóvel.

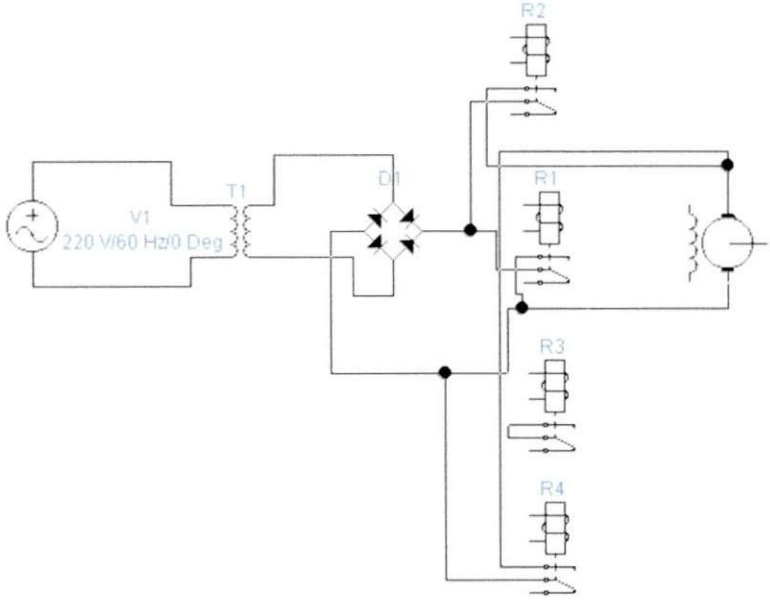

Figura 3.14 - Esquema elétrico da alimentação do motor CC, utilizado para mover os contatos elétricos do Transformador.

Tem-se quatro relés, que ora ligam o motor numa polaridade, ora invertem, conforme se deseje o sentido horário ou anti-horário, sendo que o controle dos relés é feito, vide capítulo anterior, através do microcontrolador, que em função da tensão medida, ele envia nível alto (5 v) ou baixo (0 v) para a base de um transistor que alimenta a bobina dos relés.

Deve-se ligar um diodo em anti-paralelo com as bobinas do relé, visto que ao serem acionados, ocorre uma sobretensão que polariza diretamente o diodo, de modo a limitar em 0,7 v a tensão na bobina, de outro modo tal sobretensão é suficiente para "resetar" o microcontrolador e a ação de controle fica paralisada.

A potência de um motor cc como o utilizado é da ordem de 60 W, sendo de 12 v DC sua alimentação, o motor é mostrado na Fig. 3.15.

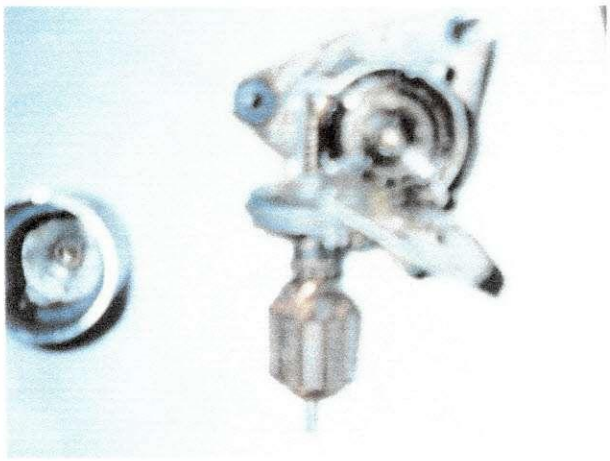

Figura 3.15 - Motor cc utilizado no protótipo.

Observe o rotor (cobreado) tipo gaiola de esquilo, o campo de imã permanente à esquerda, sendo que o imã é o contorno interno (preto), onde se encaixa o rotor.

O transformador deve ser do mesmo porte, o cálculo da área do seu núcleo do transformador é dado por:

$$A = k\sqrt{\frac{pot}{f}} \qquad \text{Eq. 3.1}$$

onde

A - área da seção transversal do núcleo do transformador

k - constante que pode variar entre 7 e 9

pot - é a potência do transformador em W

f - é a frequência de trabalho, no caso 60 Hz

Vale salientar que está foi a única formulação que se encaixou com os dados reais, ou seja, com as dimensões dos transformadores comerciais para uma dada potência normalmente obtidos no comércio, visto que pela lei de Ampere.

$$\oint H\, dL = N\, I \qquad \text{Eq. 3.2}$$

onde

H é o vetor campo magnético, tomado como uma constante ao longo do percurso do núcleo.

dL é o elemento diferencial ao longo do percurso (perímetro) do núcleo, no enlace das bobinas do transformador.

N é o número de espiras da bobina do transformador.

e pela lei de Faraday

$$V = N \frac{\partial \phi}{\partial t} \qquad \text{Eq. 3.3}$$

onde

V é a tensão na bobina do transformador.

t é o tempo.

ϕ é o fluxo magnético

O fluxo magnético é dado pela densidade de fluxo magnético B, vezes a área da seção do núcleo do transformador, e é lido diretamente de tabelas comerciais, para o

Fe-Si, por exemplo, no o eixo das abscissas se entra com o valor dos Amperes-espiras e lê-se no eixo vertical o valor de B em Tesla.

O erro entre as fórmulas empíricas e a calculada girou em torno de 1000 %, ou seja, provavelmente, os valores das tabelas, que diferem entre si, de livro para livro, da ordem de 400% ou mais, para um mesmo valor de Ampere-espira, ainda tem a possibilidade de não representarem o material utilizado na construção do núcleo do transformador, que além de ser de material com impurezas, já apresenta corrosão, e fator de empilhamento das lâminas, que vão se acumulando erros, deveria-se levantar a curva de histerese do próprio material para poder com confiabilidade utilizar então as leis de Ampere e Faraday no projeto, entretanto, a lei empírica mostrou-se suficiente.

Os diodos utilizados na construção da ponte de onda completa são para 10 A, podem ser adquiridos de fontes para alternadores de automóveis, e deve-se ter o cuidado de colocá-los em dissipadores de alumínio, apesar de não serem exaustivamente utilizados, e o nível de calor gerado ser pequeno, uma vez que a tensão desejada é obtida o motor pára, só oscilando conforme a oscilação da tensão da rede.

A Figura 3.16 mostra a caixa onde se inserirá o protótipo, sendo que o transformador da fonte do motor cc está ao fundo (lado direito) em marrom, e a ponte onda completa está no lado superior esquerdo, abaixo no centro (esquerdo) está uma lâmpada para iluminar o interior do protótipo.

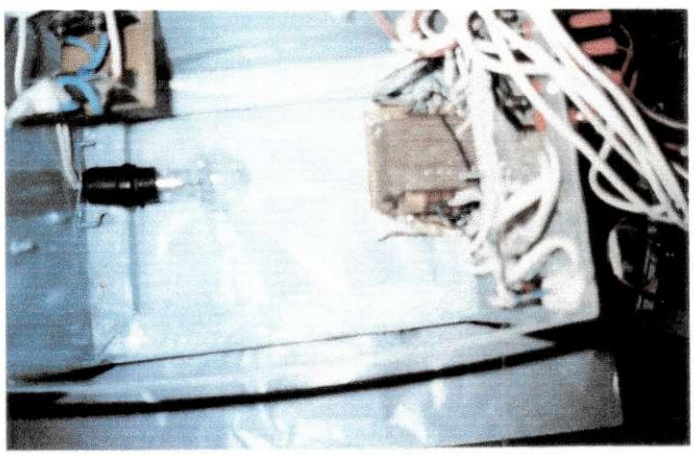

Figura 3.16 - Caixa do protótipo com o transformador e ponte onda completa para o motor cc.

3.5 - CONSTRUÇÃO DA TRANSMISSÃO MECÂNICA ENTRE O MOTOR CC E O VARIAC

A Figura 3.17 mostra a estrutura que fixa o motor cc (na parte inferior) e o variac (autotransformador variável de 100 W, na parte superior da figura), observe a necessidade de tal estrutura, que será posteriormente encaixada na caixa principal, para que a transmissão por correia não seja prejudicado com vibrações mecânicas e folgas na correia.

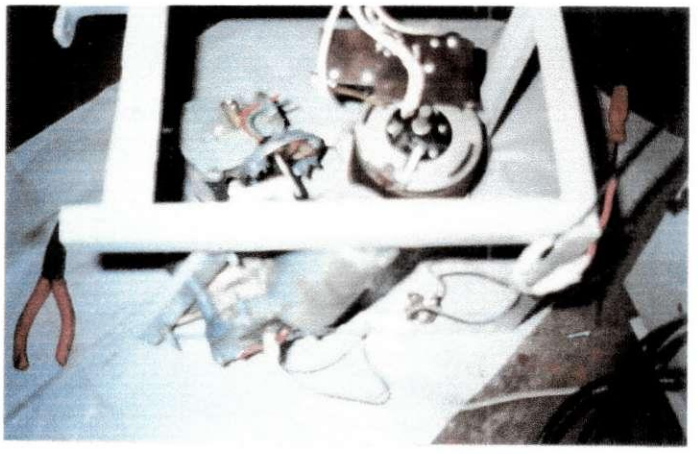

Figura 3.17 - Estrutura com Variac (em cima) e motor cc (embaixo da figura) para transmissão por correia.

Como pode ser visto na Fig. 3.17 sobre o variac (parte superior da figura) está uma estrutura onde se colocou dois fins de curso mecânicos, de modo que se o circuito eletrônico falhar, ao acionar mecanicamente o fim de curso em um sentido ou noutro, a alimentação do motor cc é cortada. É necessário que o microcontrolador seja "ressetado" e digitado uma tensão inferior aos 220 v ou superior a 0 v para que volte a funcionar.

Os fins de curso foram divididos em, fim de curso de 0 v e fim de curso de 220V de modo que foram ligados em série a partir do relé de alimentação, até o motor cc, ao ser acionado (aberto) a alimentação relativa ao sentido horário (ou anti-horário) é cortada, mas a outra anti-horária (ou horário) permanece esperando apenas o controle do pic sobre os dois relés respectivos para funcionar normalmente, desativando assim o fim de curso e fechando novamente o circuito de alimentação do motor cc.

Na Figura 3.18 está mostrada a estrutura com variac e motor cc (acima da figura) e caixa do protótipo (abaixo).

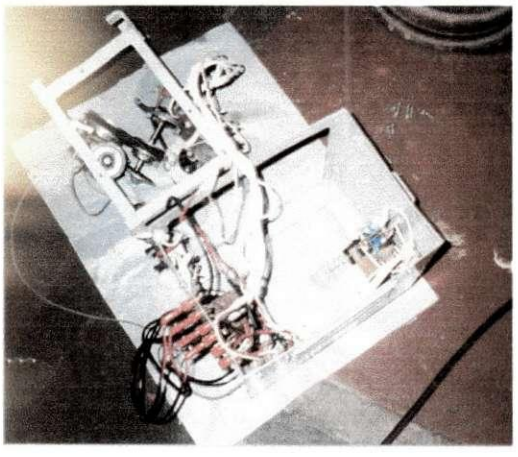

Figura 3.18 - Estrutura com variac e motor cc (parte superior) e caixa do protótipo (abaixo)

A Figura 3.19 mostra o detalhe do acoplamento por correia entre o motor cc, os redutores de velocidade e o variac. Observa-se a necessidade de dois redutores de velocidade de 1/50 em série, de modo que a variação de tensão do variac seja bem lenta, da ordem de 2,5 rpm,

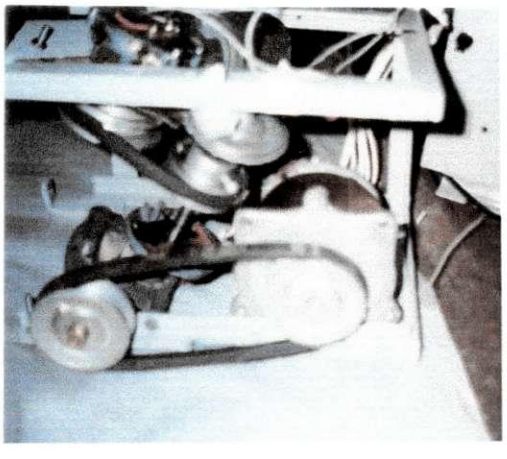

Figura 3.19 - Detalhe da transmissão por correia entre o variac e o redutor de velocidade do motor cc.

A Figura 3.20 mostra, em outra perspectiva, a transmissão por correia do motor cc, redutores de velocidade e variac

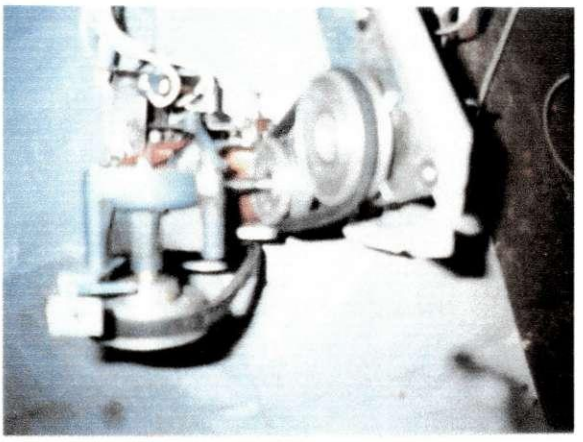

Figura 3.20- Detalhe da transmissão por correia entre o variac e o redutor de velocidade do motor cc (outra vista).

A Figura 3.21 mostra a vista superior do protótipo, já com a estrutura de transmissão de movimento dentro da caixa do protótipo, observe as conexões em vermelho e preto, tipo encaixe (lado direito, externo à caixa).

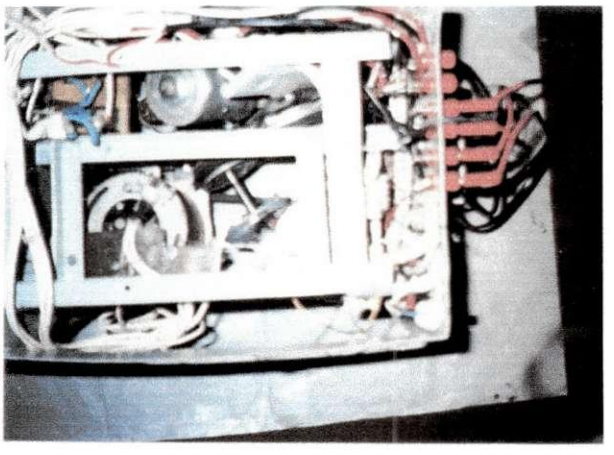

Figura 3.21 - Detalhe da transmissão por correia entre o variac e o redutor de velocidade do motor cc (outra vista).

Os fins de curso servem para desacoplar a alimentação do motor cc, a tensão a ser lida do variac, os fins de curso de 0 v e 220 v do variac, podendo-se então ligar os respectivos fios de outra estrutura a se controlada com os mesmos parâmetros, alimentação do motor cc a partir do protótipo, ligar fins de curso externos de outra estrutura a ser controlada, bem como ler a tensão a partir de outra fonte.

A Figura 3.22 mostra o protótipo, já com a estrutura de transmissão de movimento dentro da caixa do protótipo, observe as conexões em ver

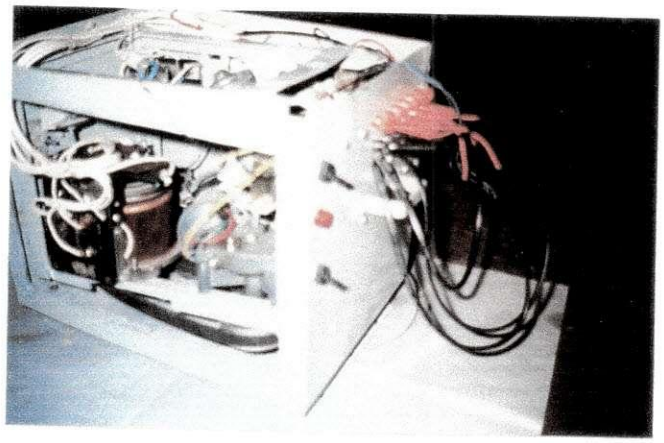

Figura 3.22- Detalhe do protótipo (vista lateral)

A Figura 3.23 mostra a vista em perspectiva do protótipo.

Figura 3.23 - Vista em perspectiva do protótipo.

A Figura 3.24 mostra a vista em superior interna do protótipo, observe no canto inferior direito o variac, com os fins de curso colocados sobre e à direita dele., uma lâmpada no centro inferior, e a transmissão por correia ao centro em ângulo entre o redutor próprio do motor cc e outro redutor da mesma ordem que transmitirá o movimento também por correia ao variac.

Figura 3.24- Vista superior do protótipo sem a tampa.

A Figura 3.25 mostra a vista frontal do protótipo, sem a tampa de vidro, que detalha no canto inferior esquerdo a parte da frente do variac, de onde é alimentado e de onde se tomam suas tensões, e a transmissão por correia ao fundo entre o segundo redutor de velocidade e ele.

Figura 3.25 - Vista frontal do protótipo sem a tampa de vidro.

A sessão seguinte aborda as etapas da construção do acoplamento entre o protótipo e o auto-transformador trifásico.

3.6 - CONSTRUÇÃO DA TRANSMISSÃO MECÂNICA PARA O AUTO-TRANSFORMADOR

Nesta etapa foi construído a transmissão mecânica entre o motor cc, seu próprio redutor de velocidade e um parafuso rosca sem fim, como o segundo redutor de velocidade que movimenta os contatos móveis de roldanas de carvão que ao se deslocarem verticalmente variam a tensão conforme a tensão colocada sobre a coluna do auto-transformador, conforme ilustrado na Fig. 3.26.

Pode ser visto o motor cc no canto superior esquerdo, a alimentação deste é tomada a partir do protótipo, através de contatos elétricos de encaixe no mesmo, além disto, existem dois fins de curso mecânicos um em cima da coluna e outro abaixo, de modo que se os contatos lhe toquem, acionando-os eles cortam a alimentação do motor naquele sentido (horário ou anti-horário) e o microcontrolador, no protótipo terá de ser regulado para uma tensão ou no sentido de diminuí-la ou aumentá-la.

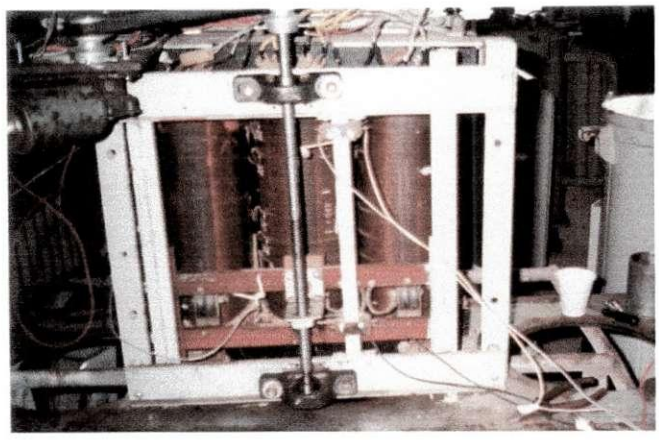

Figura 3.26 - Vista frontal do auto-transformador trifásico e sua transmissão mecânica.

Observe na Figura 3.27 a vista lateral do protótipo, de onde se conectam através dos encaixes ao auto-transformador, para controlá-lo, ou aos contatos internos do próprio protótipo, para controlar o variac interno.

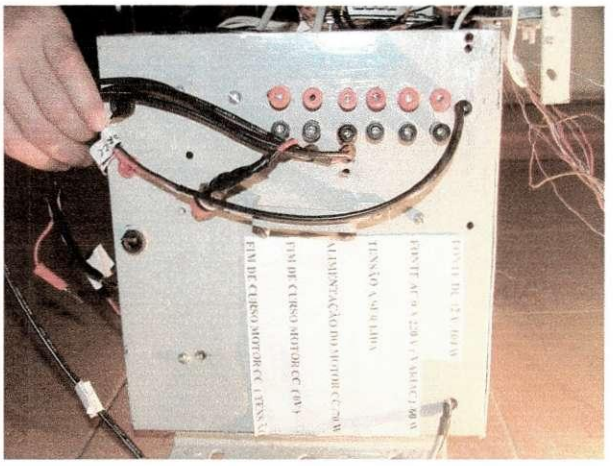

Figura 3.27 - Vista lateral do protótipo com esquema de ligação elétrico variac ou auto-transformador trifásico.

Ainda na mesma figura, da esquerda para a direita contando os encaixes, vermelhos e preto temos.

1- ligação do fim de curso referente à posição da bobina, ou variac de onde se obtém a mais alta tensão.

2 - ligação do fim de curso referente à posição da bobina, ou variac de onde se obtém a mais baixa tensão.

3 - ligação da alimentação do motor CC a ser controlado, ligação direta à ele.

4 - ligação da tensão à qual se quer controlar, no caso referente a um neutro e a fase de tensão variável.

5 - Uma tomada de tensão interna do protótipo para alimentar uma carga externa com tensão ac, se necessário, onde a potência máxima do variac é em torno de 100W em condições normais de uso.

6 - Uma tomada de tensão interna do protótipo para alimentar uma carga externa com tensão dc, se necessário, onde a potência máxima é em torno de 100W em condições normais de uso e 12V

A Figura 3.28 mostra o protótipo ao lado do auto-transformador trifásico de 10 kVa, o qual irá controlar, observe que apenas se trocaram as ligação dos fios do próprio protótipo pelos do auto-transformador.

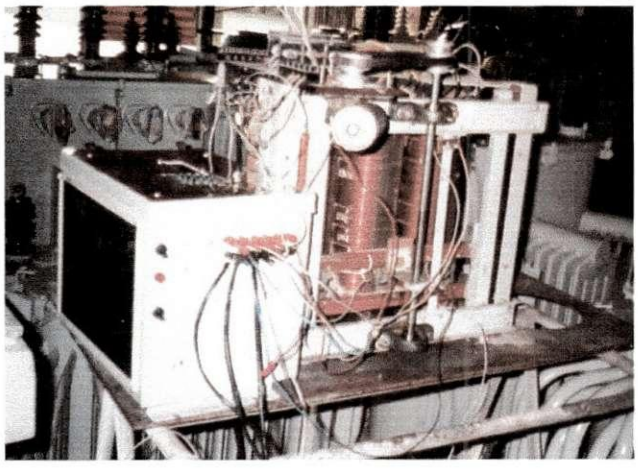

Figura 3.28 - Vista frontal do auto-transformador trifásico e sua transmissão mecânica.

A Figura 3.29 mostra outra vista do protótipo.

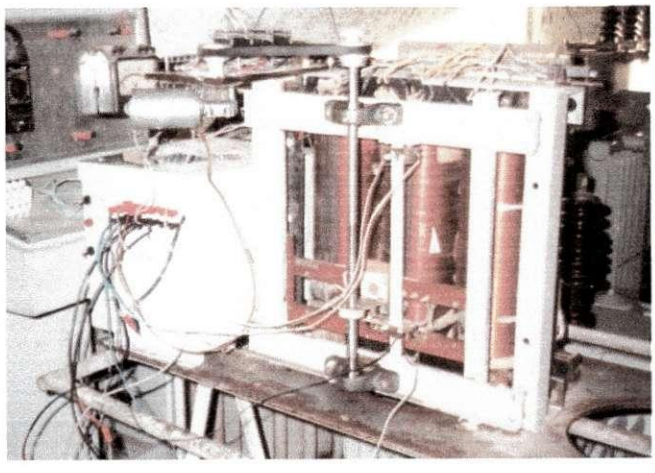

Figura 3.29 - Nova vista do auto-transformador.

A Figura 3.30 mostra uma vista frontal do mesmo.

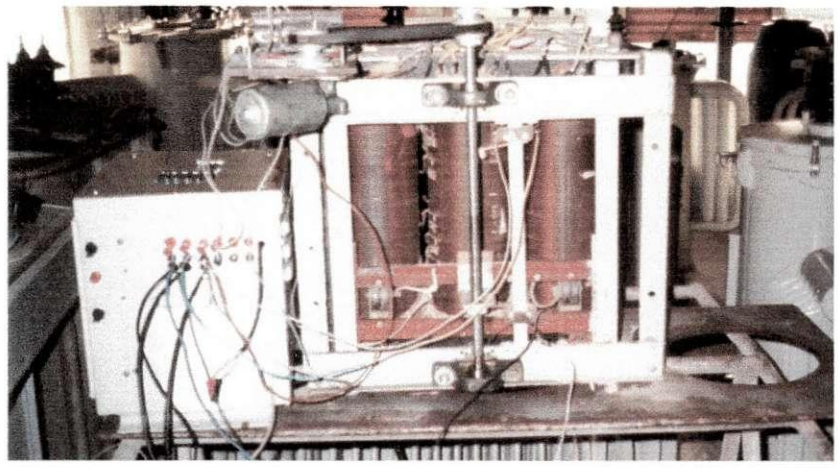

Figura 3.30 - Vista frontal do auto-transformador.

3.7 - DISCUSSÕES

O microcontrolador PIC 16F877 mostrou-se de boa performance e confiabilidade, desde que suas ligações elétricas estejam adequadas, sem variação de tensão em sua alimentação. Observou-se que mesmo com a aplicação de alimentação independente da placa com o pic e os relés, estes ao serem acionados geravam oscilação de tensão suficiente para resetar o pic, tal variação era transportada via base do transistor e pino do pic que lhe enviava o comando, tal distúrbio de tensão foi sanado com a colocação de diodos em anti-paralelo com a bobina dos relés, que lhe limitam o pico de tensão ao entrarem em condução, tal diagnóstico foi notado pelo professor do DEM UFCG, Pedro Ronaldo que culminou na solução do problema.

A questão de controlar a velocidade do motor não foi relevante, logo o projeto de tal controle não foi levado adiante, visto que o motor cc apresentava velocidade de 4000 rpm, em regime, sem carga, externa, e a redução de velocidade no variac a levou para em torno de 2 rpm, de modo que rapidamente se estabelece o regime de velocidade permanente. Um projeto em que não se utilize tal redução brusca com redutores de velocidade daria uma resposta mais rápida e aí sim se necessitaria controlar a velocidade do motor, provavelmente com um controlador PI em sistema em malha fechada, mas visto que o sistema limitante foi o tamanho das bobinas do transformador de 10 kva, teve de se utilizar um parafuso rosca-sem-fim e o protótipo foi pensado a partir de tal limitação.

A leitura de tensão pelo conversor A/D do PIC foi bastante precisa, mostrando a mesma medição que um voltímetro comercial, entretanto teve-se que rebaixar a tensão medida diretamente para a tensão de 1,5V ac, que ligada diretamente ao pino de leitura apresenta a tensão mínima de pico -2,1V, a partir da qual o PIC pode queimar, inicialmente foi utilizado um retificador tipo onda completa na entrada do pino de leitura, entretanto, por ter-se que polarizar o mesmo, havia uma tensão em torno de 1,4V dc, a partir da qual o pic começava a receber valores de tensão, de modo que só conseguiria medir valores de tensão no primário de seu transformador de medição rebaixador entre a tensão à ser lida e o pic da ordem de 70V, valores abaixo do qual não seriam lido, este problema foi sanado com a ligação direta de uma onda senoidal pura no pino de leitura.

A fabricação da placa de circuito impresso que abarcou a eletrônica do protótipo foi tipo dupla face, entretanto, não havia tecnologia disponível para furos metalizados, de modo que para sanar o problema de soldagem dos componentes foi utilizado ligações de fios que deixaram a placa de má apresentação, em um protótipo posterior recomenda-se que a placa seja feita com furo metalizado, inclusive por ser mais fácil a soldagem dos componentes.

A ligação dos relés via transistores, controlados pelo PIC apresentou-se confiável, e o relé comercial suportou bem a corrente de 5A necessária para o motor cc. Em projetos posteriores pode-se ver a utilização de foto-acopladores.

O produto tanto busca a tensão desejada quanto por ser um controle em tempo real funciona como um estabilizador da tensão solicitada com um erro em torno de 3 %, visto que está foi a oscilação da tensão da rede observada. Outra forma de regular os limites de atuação do regulador é estimar um módulo de erro entre a tensão desejada e a medida, por exemplo 2V, dentro deste erro o sistema fica sem atuar, para isso basta reprogramar o microcontrolador.

CAPÍTULO 4
CONCLUSÃO E SUGESTÕES

CAPÍTULO 4 - CONCLUSÃO E SUGESTÕES

Este trabalho contribui na área de distribuição de energia elétrica podendo fazer com que as empresas de distribuição e universidades obtenham seus próprios equipamentos para ensaios de transformadores de forma automatizada.

O produto deste trabalho tanto serve para controlar a tensão de um auto-transformador de 10 kva, variando sua amplitude, mantendo a senóide pura, bem como o controle de tensão de um variac de 150 VA, além de fornecer uma fonte dc de 12v /100 W. Tal produto está sendo utilizado na empresa Transformadores Campinense para ensaio de transformadores de distribuição, através de ensaios de curto-circuito e circuito aberto, para obtenção de laudos técnicos com as características como impedância e perdas do transformador, que é um requisito das concessionárias para instalação de transformadores novos ou remanufaturados.

A solução para variar tensão elétrica senoidal para os ensaios através do produto mostrou-se satisfatória, uma vez que não houve utilização de eletrônica de potência que gerariam harmônicos quando do uso de tiristores, por exemplo, de modo que o produto contempla a utilização conforme as normas da ABNT de ensaio de transformadores, de manter a tensão senoidal para ensaio sem distorções (o máximo possível).

No futuro deve-se construir um gabinete todo automatizado, de modo que o laudo técnico já saia impresso após a realização dos experimentos, com os cálculos provenientes dos dados colhidos dos ensaios já calculados por um software em seguida à obtenção dos dados colhidos, agilizando a obtenção dos laudos.

5.0 - Bibliografia

1 - Souza, David José, "Desbravando o PIC 16F628", 6ª ed. Editora Érica, São Paulo, 2003.

2 - Pereira, Fábio, "Microcontroladores PIC programação em C", 1ª ed. Editora Érica, São Paulo, 2003.

3 - Souza, David José, Lavínia, Nicolas Cesar, " PIC 16F877A Conectando o PIC recursos avançados", 1ª ed. Editora Érica, São Paulo, 2003.

4 - Sedra, Adel, S., Smith, Kenneth C., "Microeletrônica", Vol 1, 1ª ed., Editora Makron Books, São Paulo, 1995.

5 - Halliday, Resnick, "Física, eletricidade, magnetismo e óptica", Vol II, 1ª ed., Editora Livro Técnico S.A., São Paulo, 1970.

6 - Glover, J. Duncan, Sarma, Malukutla S. Sarma, "Power System, Analysis and Design", Edit. Brooks Cole, 2002.

7 - Araújo, Tomás Victor Gonçalves Pereira, "Relatório de estágio supervisionado", DEE-UFCG, 2003.

8 - Montero, Luís Reyes Rosales, "Monitoração e controle em tempo real baseados em microcomputador para um microgerador síncrono e motor CC", Tese de doutorado, DEE, UFCG, 1995.

ANEXO I

O MICROCONTROLADOR PIC 16F877

ANEXO I - O MICROCONTROLADOR PIC 16F877

O microcontrolador PIC-16F877 apresenta características que o capacita a ser escolhido em uma série de aplicações práticas, dentre elas:

- 33 portas configuráveis como entrada ou saída;
- 15 interrupções disponíveis;
- Memória de programação E^2PROM FLASH, que permite a gravação rápida do programa diversas vezes no mesmo chip, sem a necessidade de apagá-lo por meio de luz ultravioleta como acontece nos microcontroladores de janela;
- Memória de programa com 8kwords, com capacidade de escrita e leitura pelo próprio código interno;
- Memória E^2PROM (não-volátil) interna com 256 bytes;
- Memória RAM com 368 bytes;
- Três timers (2x8 bits e 1x16 bits);
- Comunicações seriais:SPI, I^2C e USART;
- Conversores analógicos de 10 bits (8x) e comparadores analógicos (2x);
- Dois módulos CCP: Capture, Compare e PWM;
- Programação in-circuit (alta e baixa tensão);
- Power-on Reset (POR) interno;
- Brown-out Reset (BOR) interno;

A Figura (1.1) apresenta o diagrama com a pinagem do PIC-16F877.

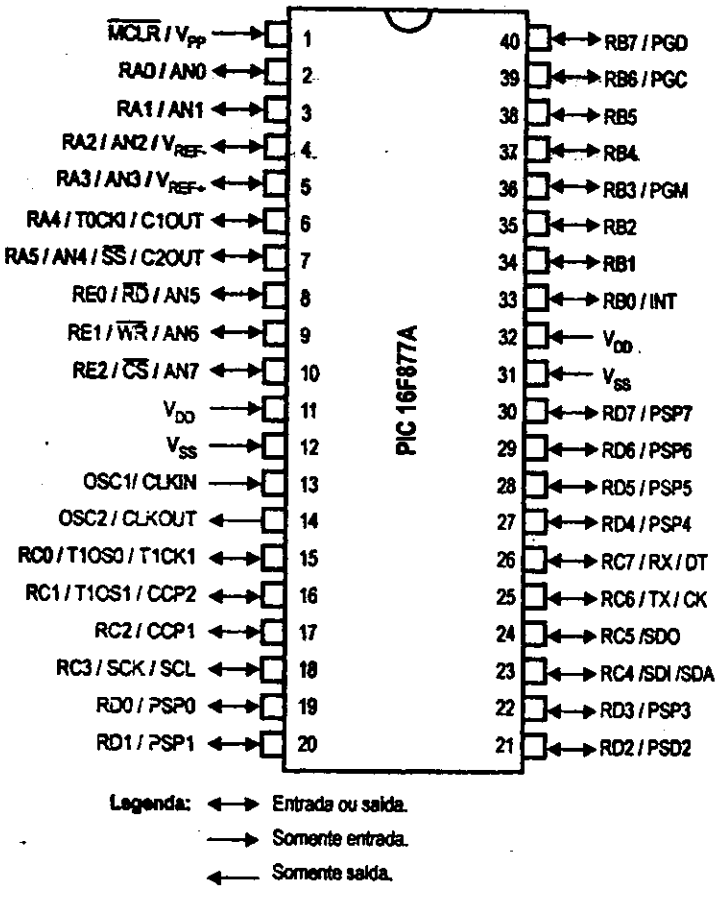

Figura 1.1 - Diagrama com a pinagem do PIC 16F877

A nomenclatura dos pinos mostrados na Fig. (1.1) é apresentado na Tabela (1.1).

Tabela 1.1 - Nomenclatura dos pinos do PIC 16F877

Nome do Pino	Núm. Pino	I/O/P	Tipo	Descrição
OSC1 / CLKIN	13	I	ST/ CMOS(4)	Entrada para cristal. Entrada para osciladores externos (híbridos ou RC).
OSC2 / CLKOUT	14	O	-	Saída para cristal. Os cristais ou ressonadores devem ser ligados aos pinos OSC1 e OSC2. Saída com onda quadrada em ¼ da freqüência imposta em OSC1 quando em modo RC. Essa freqüência equivale aos ciclos de máquina internos.

Nome do Pino	Núm. Pino	I/O/P	Tipo	Descrição
MCLR / Vpp	1	I/P	ST	*Master Clear* (*reset*) externo. O microcontrolador só funciona quando este pino encontra-se em nível alto. Entrada para tensão de programação (13V).
V_{SS}	12/31	P	-	GND.
V_{DD}	11/32	P	-	Alimentação positiva.
				PORTA (I/Os digitais bidirecionais e sistema analógico):
RA0 / AN0	2	I/O	TTL	RA0: I/O digital ou entrada analógica AN0.
RA1 / AN1	3	I/O	TTL	RA1: I/O digital ou entrada analógica AN1.
RA2 / AN2 / V_{REF-} / CV_{REF}	4	I/O	TTL	RA2: I/O digital ou entrada analógica AN2 ou tensão negativa de referência analógica.
RA3 / AN3 / V_{REF+}	5	I/O	TTL	RA3: I/O digital ou entrada analógica AN3 ou tensão positiva de referência analógica.
RA4 / T0CKI / C1OUT	6	I/O	ST	RA4: I/O digital (quando saída é *open drayn*, isto é, não consegue impor nível alto) ou entrada externa do contador TMR0 ou saída do comparador 1.
RA5 / SS / AN4 / C2OUT	7	I/O	TTL	RA5: I/O digital ou entrada analógica AN4 ou habilitação externa (slave select) para comunicação SPI ou saída do comparador 2.
				PORTB (I/Os digitais bidirecionais). Todos os pinos deste PORT possuem *pull-up* interno que podem ser ligados/ desligados pelo software:
RB0 / INT	33	I/O	TTL/ST(1)	RB0: I/O digital com interrupção externa.
RB1	34	I/O	TTL	RB1: I/O digital.
RB2	35	I/O	TTL	RB2: I/O digital.
RB3 / PGM	36		TTL	RB3: I/O digital ou entrada para programação em baixa tensão (5V).
RB4	37		TTL	RB4: I/O digital com interrupção por mudança de estado.
RB5	38		TTL	RB5: I/O digital com interrupção por mudança de estado.
RB6 / PGC	39		TTL/ST(2)	RB6: I/O digital com interrupção por mudança de estado ou clock da programação serial ou pino de *in-circuit debugger.*
RB7 / PGD	40	I/O	TTL/ST(2)	RB7: I/O digital com interrupção por mudança de estado ou *data* da programação serial ou pino de *in-circuit debugger.*
				PORTC (I/Os digitais bidirecionais):
RC0 / T1OSO / T1CKI	15	I/0	ST	RC0: I/O digital ou saída do oscilador externo para TMR1 ou entrada de incremento para TMR1.
RC1 / T1OSI / CCP2	16	I/0	ST	RC1: I/O digital ou entrada do oscilador externo para TMR1 ou entrada do Capture2 ou saídas para Compare2/PWM2.
RC2 / CCP1	17	I/0	ST	RC2: I/O digital ou entrada do Capture1 ou saídas para Compare1/PWM1.
RC3 / SCK / SCL	18	I/0	ST	RC3: I/O digital ou entrada/saída de clock para comunicação serial SPI / I^2C.
RC4 / SDI / SDA	23	I/0	ST	RC4: I/O digital ou entrada de dados para SPI ou via de dados (entrada/saída) para I^2C.

Nome do Pino	Núm. Pino	I/O/P	Tipo	Descrição
RC5 / SDO	24	I/O	ST	RC5: I/O digital ou saída de dados para SPI.
RC6 / TX / CK	25	I/O	ST	RC6: I/O digital ou TX (transmissão) para comunicação USART assíncrona ou *clock* para comunicação síncrona.
RC7 / RX / DT	26	I/O	ST	RC7: I/O digital ou RX (recepção) para comunicação USART assíncrona ou *data* para comunicação síncrona.
				PORTD (I/Os digitais bidirecionais) ou porta de comunicação paralela:
RD0 / PSP0	19	I/O	TTL/ST(3)	RD0: I/O digital ou dado 0 (comunicação paralela).
RD1 / PSP1	20	I/O	TTL/ST(3)	RD1: I/O digital ou dado 1 (comunicação paralela).
RD2 / PSP2	21	I/O	TTL/ST(3)	RD2: I/O digital ou dado 2 (comunicação paralela).
RD3 / PSP3	22	I/O	TTL/ST(3)	RD3: I/O digital ou dado 3 (comunicação paralela).
RD4 / PSP4	27	I/O	TTL/ST(3)	RD4: I/O digital ou dado 4 (comunicação paralela).
RD5 / PSP5	28	I/O	TTL/ST(3)	RD5: I/O digital ou dado 5 (comunicação paralela).
RD6 / PSP6	29	I/O	TTL/ST(3)	RD6: I/O digital ou dado 6 (comunicação paralela).
RD7 / PSP7	30	I/O	TTL/ST(3)	RD7: I/O digital ou dado 7 (comunicação paralela).
				PORTE (I/Os digitais bidirecionais e sistema analógico):
RE0 / RD / AN5	8	I/O	TTL/ST(3)	RE0: I/O digital ou controle de leitura da porta paralela ou entrada analógica AN5.
RE1 / WR / AN6	9	I/O	TTL/ST(3)	RE1: I/O digital ou controle de escrita da porta paralela ou entrada analógica AN6.
RE2 / CS / AN7	10	I/O	TTL/ST(3)	RE2: I/O digital ou habilitação externa da porta paralela ou entrada analógica AN7.

Legenda:

I	=	*Input* (entrada)
O	=	*Output* (saída)
I/O	=	*Input/Output* (entrada ou saída)
P	=	*Power* (alimentação)
-	=	Não-utilizado
TTL	=	Entrada tipo TTL
ST	=	Entrada tipo *Schmitt Trigger*

Notas:

(1) Esta entrada é do tipo ST, somente quando configurado como interrupção externa.

(2) Esta entrada é do tipo ST, somente durante o modo de programação serial.

(3) Esta entrada é do tipo ST, quando configurado como I/O de uso geral e TTL quando usado em modo de porta paralela.

(4) Esta entrada é ST quando em modo RC e CMOS nos demais casos.

A Figura (1.2) apresenta a estruturação interna do PIC 16F877A, está esquematizado um diagrama de blocos detalhando todos os periféricos e comunicações que o compõem.

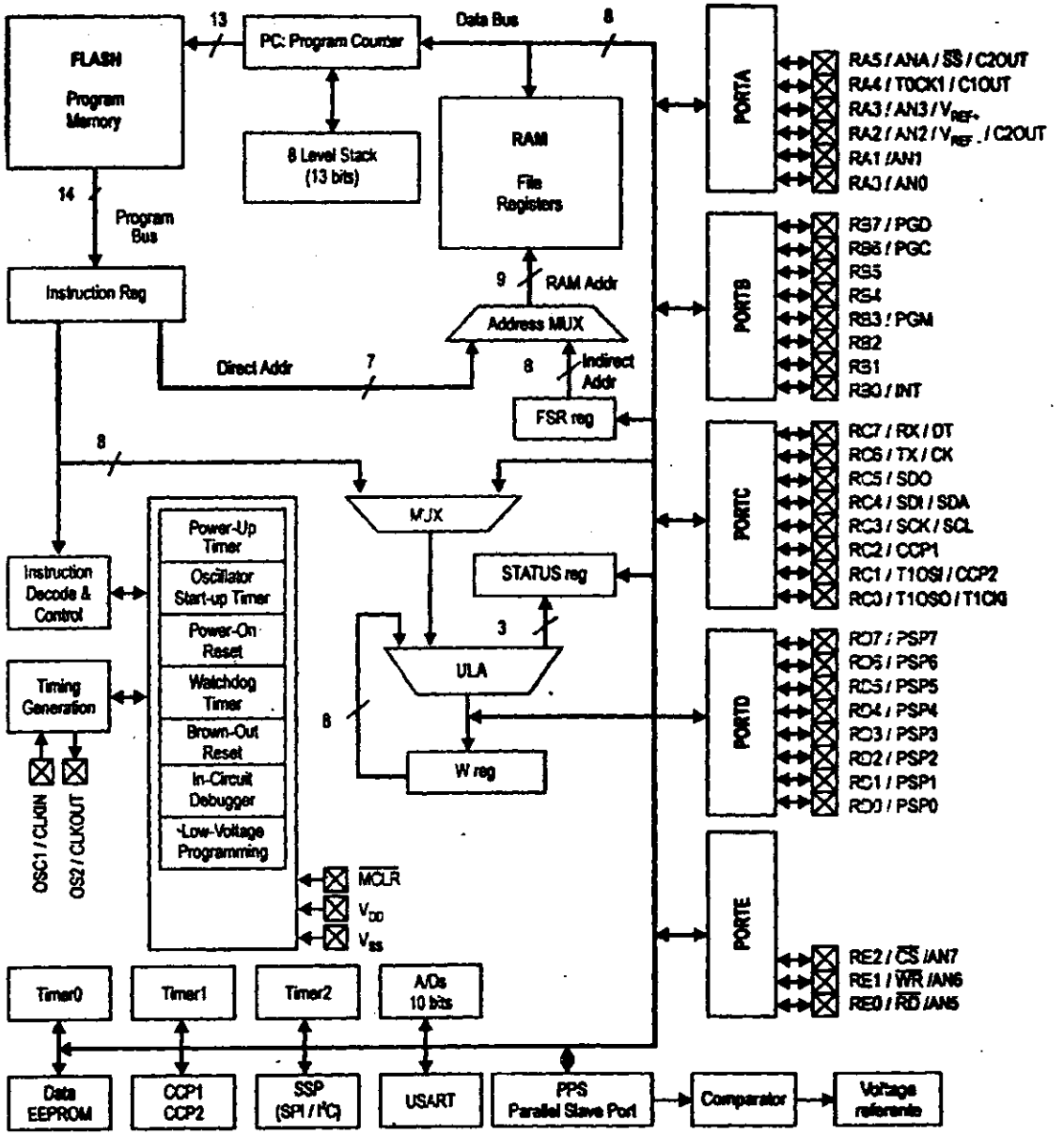

Figura 1.2 - Diagrama de blocos do PIC 16F877 com sua estrutura interna.

Este diagrama foi retirado do data sheet da Microchip. No centro está a ULA, que é a unidade de processamento e está diretamente ligada ao registrador Work (W reg). No canto superior esquerdo temos a memória de programa (FLASH) saindo desse bloco temos um barramento de 14 bits (Program Bus). Mais à direita está a memória de dados (RAM), ela já possui um barramento de 8 bits (Data Bus). Totalmente do lado direito encontram-se os PORTS, de PORTA a PORTE. Na parte inferior podem ser encontrados os demais periféricos, tais como a E^2PROM (memória de dados não-

volátil), os timers (TMR0, TMR! e TMR2), os A/Ds de 10 bits, os modos CCP (Compare, Capture e PWM), as comunicações seriais (SPI, I^2C e USART), os comparadores e a tensão de referência. Observe que, entre todos os periféricos, a comunicação é feita através de um barramento de oito vias.

Ao centro está o registrador de status (STATUS reg), onde algumas informações importantes sobre as operações aritméticas da ULA ficam armazenadas, e as demais SFRs (registradores especiais). Na parte superior temos o contador de linha de programa (Program Counter) e a pilha de oito níveis (Stack). Tem-se ainda os circuitos internos de reset, POR, BOR, osciladores, Watchdog Timer (WDT) e sistema de programação.

Para execução de uma instrução pelo processador várias operações precisam ser realizadas, são executadas em subciclos do ciclo de máquina, originados pela divisão do clock externo.Esses subciclos são chamados de Q1, Q2, Q3 e Q4.

O PC (contador de programa) é incrementado automaticamente no início de Q1, Durante o decorrer dos quatro tempos (Q1 a Q4), a instrução previamente carregada para dentro da ULA é executada, sendo trocadas informações com a memória de dados e o registrador Work sempre que necessário. Por último, ao final do tempo Q4, a prócima instrução é buscada da memória de programa e armazenada na ULA. Esta busca da próxima instrução na anterior é conhecido como pipeline, a Fig. (1.3) ilustra os quatro subciclos (Q1 a Q4) e o conceito de Pipeline.

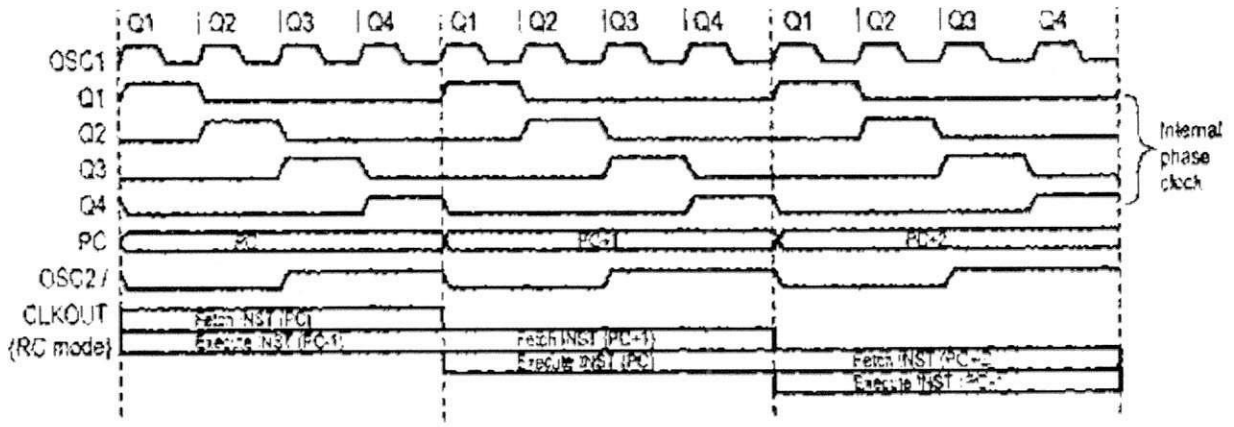

Figura 1.3 - Diagrama dos quatro subciclos da execução de uma instrução no microprocessador.

A Figura (1.4) detalha o mapa da memória de dados do PIC 16F877.

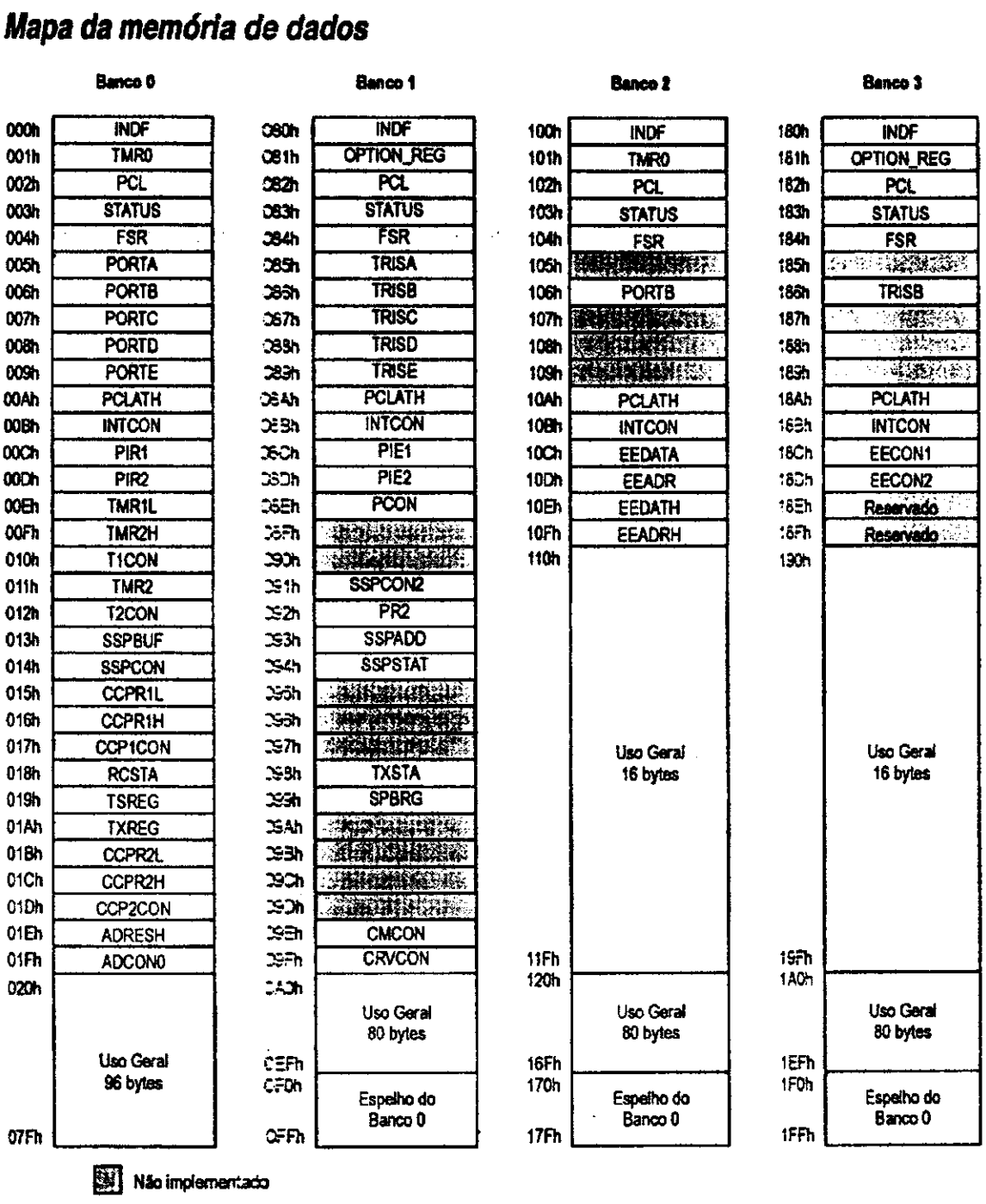

Figura 1.4 - Diagrama dos quatro subciclos da execução de uma instrução no microprocessador.

As características elétricas do microcontrolador são apresentadas na Tab. (1.2).

Tabela 1.2 - Características elétricas.

Temperatura de trabalho	-55°C até 125°C
Tensão de trabalho	4 V a 5,5 V
Tensão máxima no pino V_{DD} (em relação ao V_{ss})	-0,3 V até 7,5 V
Tensão máxima no pino MCRL (em relação ao V_{ss})	0 V até 14 V
Tensão máxima no pino RA4 (em relação ao V_{ss})	0 V até 8,5 V
Tensão máxima nos demais pinos (em relação ao V_{ss})	-0,3 V até (V_{DD} + 0,3 V)
Dissipação máxima de energia	1,0 W
Corrente máxima de saída no pino V_{SS}	300 mA
Corrente máxima de entrada no pino V_{DD}	250 mA
Corrente máxima de entrada de um pino (quando emV_{SS})	25 mA
Corrente máxima de saída de um pino (quando emV_{DD})	25 mA
Corrente máxima de entrada do PORTA, PORTB e PORTE combinados	200 mA
Corrente máxima de saída do PORTA, PORTB e PORTE combinados	200 mA
Corrente máxima de entrada do PORTC, PORTD combinados	200 mA
Corrente máxima de saída do PORTC, PORTD combinados	200 mA

ANEXO II

PROGRAMA PRINCIPAL UTILIZADO NO PIC EM LINGUAGEM C

PROGRAMA PRINCIPAL DO MICROCONTROLADOR

```
/* Este programa transmite serialmente o caractere pressionado no teclado
além de apresentá-lo no display LCD. O teclado inclui facilidades de
auto-repetição e função shift*/

#include <16f877.h>
#device adc=10
#use delay(clock=4000000)
#fuses HS,NOWDT,NOPROTECT,NOLVP,NOBROWNOUT,NOPUT
#use rs232(baud=9600,parity=N,xmit=PIN_C6,rcv=PIN_C7,bits=8)

// configuração dos pinos do LCD
        #define lcd_enable   pin_e0    // pino enable do LCD
        #define lcd_rs                  pin_e1   // pino rs do LCD
        #define lcd_d4                  pin_c0        // pino de dados d4 do LCD
        #define lcd_d5                  pin_c1        // pino de dados d5 do LCD
        #define lcd_d6                  pin_c2        // pino de dados d6 do LCD
        #define lcd_d7                  pin_c3        // pino de dados d7 do LCD

#include <LCD_16x2_Libteste.c>
#include <math.h>
#include <stdlib.h>

#fuses HS,NOWDT,NOPROTECT,NOLVP,NOBROWNOUT,NOPUT

// Definições da matriz do teclado

#define fio0        pin_d0
```

```
#define fio1        pin_d1
#define fio2        pin_d2
#define fio3        pin_d3
#define fio4        pin_d4
#define fio5        pin_d5
#define fio6        pin_d6
#define fio7        pin_d7


char varre_teclas (void)

/* Realiza a varredura do teclado e retorna o
 valor da tecla pressionada.
 O teclado está conectado da seguinte forma:

//      FIO0   FIO1   FIO2   FIO3   FIO4   FIO5   FIO6   FIO7
// FIO0  -      -      -      -      -      3      2      1
// FIO1  -      -      -      -      F      6      5      4
// FIO2  -      -      -      -             9      8      7
// FIO3  -      -      -      -      R      #      0      *-M

 Os valores decimais retornados são:

 0  48
 1  49
 2  50
 3  51
 4  52
 5  53
 6  54
 7  55
 8  56
```

```
 9  57
*/
{
  int tecla;
  tecla=' ';

  output_high(fio0); // ativa o fio 1
  output_low(fio1);  // desativa o fio 2
  output_low(fio2);  // desativa o fio 3
  output_low(fio3);  // desativa o fio 4
  output_low(fio4);  // desativa o fio 5
  output_low(fio5);  // desativa o fio 6
  output_low(fio6);  // desativa o fio 7
  output_low(fio7);  // desativa o fio 8
  if (input(fio4)) tecla = 'R';
  if (input(fio5)) tecla = ' ';
  if (input(fio6)) tecla = ' ';
  if (input(fio7)) tecla = ' ';

  output_low(fio0);  // desativa o fio 1
  output_high(fio1); // ativa o fio 2
  if (input(fio4)) tecla = '*';
  if (input(fio5)) tecla = '7';
  if (input(fio6)) tecla = '1';
  if (input(fio7)) tecla = '4';
  output_low(fio1);  // desativa o fio 1
  output_high(fio2); // ativa o fio 2
  if (input(fio4)) tecla = '0';
  if (input(fio5)) tecla = '8';
  if (input(fio6)) tecla = '2';
  if (input(fio7)) tecla = '5';
  output_low(fio2);  // desativa o fio 1
```

```
  output_high(fio3); // ativa o fio 2
  if (input(fio4)) tecla = '#';
  if (input(fio5)) tecla = '9';
  if (input(fio6)) tecla = '3';
  if (input(fio7)) tecla = '6';//M
  return (tecla);
}


int32 potd (int n2)
{

int32 z22=1;

while (n2>0)
{
  z22=z22*10;
  n2=n2-1;
}

  return (z22);
}



main()
{
  int32 i,ii,i3,i4;
  signed int eant,e1;
        long int valor,vlido[5];
        int32 val32,valorant,vmedio,vaceito,valorconv;
  float vaceito2,erro;
```

```
int tecla;

set_tris_a( 0xFF );
setup_adc_ports(RA0_ANALOG);
setup_adc(ADC_CLOCK_DIV_2);
setup_psp(PSP_DISABLED);
setup_spi(FALSE);
setup_counters(RTCC_INTERNAL,RTCC_DIV_2);
setup_timer_1(T1_DISABLED);
setup_timer_2(T2_DISABLED,0,1);
set_adc_channel(0);
delay_us(10);

valor=0;
valorant=0;
vmedio=0;
      val32=0;
i=0;
e1=0;
eant=0;

output_low(pin_b1);
output_low(pin_b2);

ini_lcd_16x2();    // inicializa pinos RS232

for (i=0;i<5;i++)
{
vlido[i]=0;
}

printf (exibe_lcd,"\fDIGITE  VALOR DA\nTENSAO E LND =>");
```

```
                delay_ms(2000);
   exibe_lcd('\f');
   delay_ms(100);

i=0;
  while (true)
  {
    tecla = varre_teclas();
         if (tecla != ' ')
     {
               printf (exibe_lcd,"%c",tecla);
               delay_ms(1000);
               vlido[i]=tecla;
    i=i+1;
          }

         if (tecla == 'R')
     {
               goto pula;
          }

}
pula:

    vaceito=0;
    ii=0;
    i=i-1;

    while (i>0)
    {
    i3=i-1;
    i4=potd((int32) i3);
```

```
      vaceito=vaceito+(vlido[ii]-48)*i4;
      ii=ii+1;
      i=i-1;
    }

  printf (exibe_lcd,"\fvaceito= %lu",vaceito);
  delay_ms(1000);

  vaceito2=1500*( (float)vaceito/340 );
  valorconv=vaceito2;

        exibe_lcd ('\f');  // apaga o display
  delay_ms(100);


ii=1;
i3=1;

le:

i=0;
valor=0;
valorant=0;
val32=0;
        for (i=1;i<50;i++)
        {
                // O escalonamento é realizado da seguinte forma:
                // resultado = (5000 * valor lido) / 1023
                // Para facilitar os cálculos, somamos um ao
                // valor lido:
                // resultado = (5000 * (valor + 1)) / 1024
                // simplificando:
```

```
                // resultado = ((valor + 1) * 4) + ((valor + 1) * 113) / 128
                // Repare que é necessário converter a segunda parte da
                // equação para 32 bits para que o compilador efetue o
                // cálculo corretamente
                valor = read_adc(); // efetua a conversão A/D
                // Se o valor é > 0, soma 1 ao valor lido
                if (valor==0) valor += 1;
        // imprime o valor da tensão no display
                // 5000 = 5,000 Volts ou 5000 milivolts
    val32 = (5100 * ((int32)valor+1)) / 1024;

                if (val32>valorant) valorant = val32;

  }

 valorconv=(float)((valorant*(float)340/1500)/1.4142);

 printf (exibe_lcd,"\fTensao = %lu V",valorconv);
      delay_ms (200);


erro=abs((float)(((float)vaceito-(float)valorconv)/(float)vaceito));


      if(erro<0.03)
      {
      output_low(pin_c5);
      delay_ms(50);
      output_low(pin_c4);
      goto le;
      }
```

```
if (vaceito>valorconv)
{
output_low(pin_c4);
delay_ms(50);
output_high(pin_c5);
goto le;
}
if (vaceito<valorconv)
{
output_low(pin_c5);
delay_ms(50);
output_high(pin_c4);
goto le;
}


}
```

ANEXO III

DEFINIÇÕES E DIRETIVAS UTILIZADAS PELO COMPILADOR PARA O PIC 16F877 EM LINGUAGEM C

FUSES VÁLIDOS

LP	Low power osc < 200 khz
XT	Crystal osc <= 4mhz
HS	High speed Osc (> 4mhz)
RC	Resistor/Capacitor Osc with CLKOUT
NOWDT	No Watch Dog Timer
WDT	Watch Dog Timer
NOPUT	No Power Up Timer
PUT	Power Up Timer
PROTECT	Code protected from reads
PROTECT_5%	Protect 5% of ROM
PROTECT_50%	Protect 50% of ROM
NOPROTECT	Code not protected from reading
NOBROWNOUT	No brownout reset
BROWNOUT	Reset when brownout detected
LVP	Low Voltage Programming on B3(PIC16) or B5(PIC18)
NOLVP	No low voltage prgming, B3(PIC16) or B5(PIC18) used for I/O
CPD	Data EEPROM Code Protected
NOCPD	No EE protection
WRT	Program Memory Write Protected
NOWRT	Program memory not write protected
DEBUG	Debug mode for use with ICD
NODEBUG	No Debug mode for ICD

INTERRUPÇÕES VÁLIDAS

RB	Port B any change on B4-B7
EXT	External interrupt
AD	Analog to digital conversion complete
TBE	RS232 transmit buffer empty
RDA	RS232 receive data available
TIMER1	Timer 1 overflow
TIMER2	Timer 2 overflow
CCP1	Capture or Compare on unit 1
CCP2	Capture or Compare on unit 2
SSP	SPI or I2C activity

PSP Parallel Slave Port data in
BUSCOL Bus collision
EEPROM Write complete
TIMER0 Timer 0 overflow (using TIMER0 name)

16F877

```
//////// Standard Header file for the PIC16F877 device ////////////////
#device PIC16F877
#nolist
//////// Program memory: 8192x14  Data RAM: 367  Stack: 8
//////// I/O: 33   Analog Pins: 8
//////// Data EEPROM: 256
//////// C Scratch area: 77   ID Location: 2000
//////// Fuses: LP,XT,HS,RC,NOWDT,WDT,NOPUT,PUT,PROTECT,PROTECT_5%
//////// Fuses:
PROTECT_50%,NOPROTECT,NOBROWNOUT,BROWNOUT,LVP,NOLVP,CPD
//////// Fuses: NOCPD,WRT,NOWRT,DEBUG,NODEBUG
////////
////////////////////////////////////////////////////////////// I/O
// Discrete I/O Functions: SET_TRIS_x(), OUTPUT_x(), INPUT_x(),
//                         PORT_B_PULLUPS(), INPUT(),
//                         OUTPUT_LOW(), OUTPUT_HIGH(),
//                         OUTPUT_FLOAT(), OUTPUT_BIT()
// Constants used to identify pins in the above are:

#define PIN_A0  40
#define PIN_A1  41
#define PIN_A2  42
#define PIN_A3  43
#define PIN_A4  44
#define PIN_A5  45

#define PIN_B0  48
#define PIN_B1  49
#define PIN_B2  50
#define PIN_B3  51
#define PIN_B4  52
#define PIN_B5  53
#define PIN_B6  54
#define PIN_B7  55

#define PIN_C0  56
#define PIN_C1  57
```

```
#define PIN_C2  58
#define PIN_C3  59
#define PIN_C4  60
#define PIN_C5  61
#define PIN_C6  62
#define PIN_C7  63

#define PIN_D0  64
#define PIN_D1  65
#define PIN_D2  66
#define PIN_D3  67
#define PIN_D4  68
#define PIN_D5  69
#define PIN_D6  70
#define PIN_D7  71

#define PIN_E0  72
#define PIN_E1  73
#define PIN_E2  74

//////////////////////////////////////////////////////////////////// Useful defines
#define FALSE 0
#define TRUE 1

#define BYTE int
#define BOOLEAN short int

#define getc getch
#define fgetc getch
#define getchar getch
#define putc putchar
#define fputc putchar
#define fgets gets
#define fputs puts

//////////////////////////////////////////////////////////////////// Control
// Control Functions:  RESET_CPU(), SLEEP(), RESTART_CAUSE()
// Constants returned from RESTART_CAUSE() are:
#define WDT_FROM_SLEEP  0
#define WDT_TIMEOUT     8
#define MCLR_FROM_SLEEP 16
#define NORMAL_POWER_UP 24


//////////////////////////////////////////////////////////////////// Timer 0
// Timer 0 (AKA RTCC)Functions: SETUP_COUNTERS() or SETUP_TIMER0(),
//                              SET_TIMER0() or SET_RTCC(),
//                              GET_TIMER0() or GET_RTCC()
```

```
// Constants used for SETUP_TIMER0() are:
#define RTCC_INTERNAL   0
#define RTCC_EXT_L_TO_H 32
#define RTCC_EXT_H_TO_L 48

#define RTCC_DIV_1      8
#define RTCC_DIV_2      0
#define RTCC_DIV_4      1
#define RTCC_DIV_8      2
#define RTCC_DIV_16     3
#define RTCC_DIV_32     4
#define RTCC_DIV_64     5
#define RTCC_DIV_128    6
#define RTCC_DIV_256    7


#define RTCC_8_BIT      0

// Constants used for SETUP_COUNTERS() are the above
// constants for the 1st param and the following for
// the 2nd param:

////////////////////////////////////////////////////////////////////// WDT
// Watch Dog Timer Functions: SETUP_WDT() or SETUP_COUNTERS() (see above)
//                      RESTART_WDT()
//
#define WDT_18MS        8
#define WDT_36MS        9
#define WDT_72MS        10
#define WDT_144MS       11
#define WDT_288MS       12
#define WDT_576MS       13
#define WDT_1152MS      14
#define WDT_2304MS      15

////////////////////////////////////////////////////////////////////// Timer 1
// Timer 1 Functions: SETUP_TIMER_1, GET_TIMER1, SET_TIMER1
// Constants used for SETUP_TIMER_1() are:
//      (or (via |) together constants from each group)
#define T1_DISABLED         0
#define T1_INTERNAL         0x85
#define T1_EXTERNAL         0x87
#define T1_EXTERNAL_SYNC    0x83

#define T1_CLK_OUT          8

#define T1_DIV_BY_1         0
#define T1_DIV_BY_2         0x10
```

```
#define T1_DIV_BY_4        0x20
#define T1_DIV_BY_8        0x30

//////////////////////////////////////////////////////////// Timer 2
// Timer 2 Functions: SETUP_TIMER_2, GET_TIMER2, SET_TIMER2
// Constants used for SETUP_TIMER_2() are:
#define T2_DISABLED        0
#define T2_DIV_BY_1        4
#define T2_DIV_BY_4        5
#define T2_DIV_BY_16       6

//////////////////////////////////////////////////////////// CCP
// CCP Functions: SETUP_CCPx, SET_PWMx_DUTY
// CCP Variables: CCP_x, CCP_x_LOW, CCP_x_HIGH
// Constants used for SETUP_CCPx() are:
#define CCP_OFF                    0
#define CCP_CAPTURE_FE             4
#define CCP_CAPTURE_RE             5
#define CCP_CAPTURE_DIV_4          6
#define CCP_CAPTURE_DIV_16         7
#define CCP_COMPARE_SET_ON_MATCH   8
#define CCP_COMPARE_CLR_ON_MATCH   9
#define CCP_COMPARE_INT            0xA
#define CCP_COMPARE_RESET_TIMER    0xB
#define CCP_PWM                    0xC
#define CCP_PWM_PLUS_1             0x1c
#define CCP_PWM_PLUS_2             0x2c
#define CCP_PWM_PLUS_3             0x3c
long CCP_1;
#byte  CCP_1   =           0x15
#byte  CCP_1_LOW=          0x15
#byte  CCP_1_HIGH=         0x16
long CCP_2;
#byte  CCP_2   =           0x1B
#byte  CCP_2_LOW=          0x1B
#byte  CCP_2_HIGH=         0x1C
//////////////////////////////////////////////////////////// PSP
// PSP Functions: SETUP_PSP, PSP_INPUT_FULL(), PSP_OUTPUT_FULL(),
//          PSP_OVERFLOW(), INPUT_D(), OUTPUT_D()
// PSP Variables: PSP_DATA
// Constants used in SETUP_PSP() are:
#define PSP_ENABLED                0x10
#define PSP_DISABLED               0

#byte  PSP_DATA=           8

//////////////////////////////////////////////////////////// SPI
// SPI Functions: SETUP_SPI, SPI_WRITE, SPI_READ, SPI_DATA_IN
```

```
// Constants used in SETUP_SSP() are:
#define SPI_MASTER      0x20
#define SPI_SLAVE       0x24
#define SPI_L_TO_H      0
#define SPI_H_TO_L      0x10
#define SPI_CLK_DIV_4   0
#define SPI_CLK_DIV_16  1
#define SPI_CLK_DIV_64  2
#define SPI_CLK_T2      3
#define SPI_SS_DISABLED 1

#define SPI_SAMPLE_AT_END 0x8000
#define SPI_XMIT_L_TO_H 0x4000

//////////////////////////////////////////////////////////// UART
// Constants used in setup_uart() are:
// FALSE - Turn UART off
// TRUE  - Turn UART on
#define UART_ADDRESS 2
#define UART_DATA   4
// TRUE  - Turn UART on
//////////////////////////////////////////////////////////// ADC
// ADC Functions: SETUP_ADC(), SETUP_ADC_PORTS() (aka SETUP_PORT_A),
//                SET_ADC_CHANNEL(), READ_ADC()
// Constants used in SETUP_ADC_PORTS() are:
#define NO_ANALOGS          0x86       // None
#define ALL_ANALOG          0x80       // A0 A1 A2 A3 A5 E0 E1 E2 Ref=Vdd
#define ANALOG_RA3_REF      0x81       // A0 A1 A2 A5 E0 E1 E2 Ref=A3
#define A_ANALOG            0x82       // A0 A1 A2 A3 A5 Ref=Vdd
#define A_ANALOG_RA3_REF    0x83       // A0 A1 A2 A5 Ref=A3
#define RA0_RA1_RA3_ANALOG  0x84       // A0 A1 A3 Ref=Vdd
#define RA0_RA1_ANALOG_RA3_REF 0x85    // A0 A1 Ref=A3
#define ANALOG_RA3_RA2_REF          0x88  // A0 A1 A5 E0 E1 E2 Ref=A2,A3
#define ANALOG_NOT_RE1_RE2          0x89  // A0 A1 A2 A3 A5 E0 Ref=Vdd
#define ANALOG_NOT_RE1_RE2_REF_RA3  0x8A  // A0 A1 A2 A5 E0 Ref=A3
#define ANALOG_NOT_RE1_RE2_REF_RA3_RA2  0x8B  // A0 A1 A5 E0
Ref=A2,A3
#define A_ANALOG_RA3_RA2_REF        0x8C  // A0 A1 A5 Ref=A2,A3
#define RA0_RA1_ANALOG_RA3_RA2_REF  0x8D  // A0 A1 Ref=A2,A3
#define RA0_ANALOG                  0x8E  // A0
#define RA0_ANALOG_RA3_RA2_REF      0x8F  // A0 Ref=A2,A3
// Constants used for SETUP_ADC() are:
#define ADC_OFF             0          // ADC Off
#define ADC_CLOCK_DIV_2     1
#define ADC_CLOCK_DIV_8     0x41
#define ADC_CLOCK_DIV_32    0x81
#define ADC_CLOCK_INTERNAL  0xc1       // Internal 2-6us
```

```
// Constants used in READ_ADC() are:
#define ADC_START_AND_READ    7   // This is the default if nothing is specified
#define ADC_START_ONLY        1
#define ADC_READ_ONLY         6

//////////////////////////////////////////////////////////////////// INT
// Interrupt Functions: ENABLE_INTERRUPTS(), DISABLE_INTERRUPTS(),
//                      EXT_INT_EDGE()
//
// Constants used in EXT_INT_EDGE() are:
#define L_TO_H          0x40
#define H_TO_L          0
// Constants used in ENABLE/DISABLE_INTERRUPTS() are:
#define GLOBAL          0x0BC0
#define INT_RTCC        0x0B20
#define INT_RB          0x0B08
#define INT_EXT         0x0B10
#define INT_AD          0x8C40
#define INT_TBE         0x8C10
#define INT_RDA         0x8C20
#define INT_TIMER1      0x8C01
#define INT_TIMER2      0x8C02
#define INT_CCP1        0x8C04
#define INT_CCP2        0x8D01
#define INT_SSP         0x8C08
#define INT_PSP         0x8C80
#define INT_BUSCOL      0x8D08
#define INT_EEPROM      0x8D10
#define INT_TIMER0      0x0B20

#list
```

ANEXO IV

PROGRAMA PARA COMUNICAÇÃO ENTRE O LCD E O PIC 16F877 EM LINGUAGEM C - FEITO POR FÁBIO PEREIRA EM SEU LIVRO

```
/*************************************************************************
**/
/*  MOD_LCD.C - Biblioteca de manipulação de módulo LCD          */
/*                                                               */
/*  Autor: Fábio Pereira                                         */
/*                                                               */
/*************************************************************************
**/

// As definições a seguir são utilizadas para acesso aos pinos do display
// caso o pino RW não seja utilizado, comente a definição lcd_rw
#ifndef lcd_enable
      #define lcd_enable        pin_e1      // pino enable do LCD
      #define lcd_rs            pin_e0      // pino rs do LCD
      //#define lcd_rw          pin_e2      // pino rw do LCD
      #define lcd_d4            pin_d4      // pino de dados d4 do LCD
      #define lcd_d5            pin_d5      // pino de dados d5 do LCD
      #define lcd_d6            pin_d6      // pino de dados d6 do LCD
      #define lcd_d7            pin_d7      // pino de dados d7 do LCD
#endif

#define lcd_type 2        // 0=5x7, 1=5x10, 2=2 linhas
#define lcd_seg_lin 0x40  // Endereço da segunda linha na RAM do LCD

// a constante abaixo define a seqüência de inicialização do módulo LCD
//byte CONST INI_LCD[4] = {0x20 | (lcd_type << 2), 0xf, 1, 6};
byte CONST INI_LCD[4] = {0x20 | (lcd_type << 2), 0x0c, 1, 6};


byte lcd_le_byte()
```

```
// lê um byte do LCD (somente com pino RW)
{
        byte dado;
        // configura os pinos de dados como entradas
        input(lcd_d4);
        input(lcd_d5);
        input(lcd_d6);
        input(lcd_d7);
        // se o pino rw for utilizado, coloca em 1
        #ifdef lcd_rw
                output_high(lcd_rw);
        #endif
  output_high(lcd_enable); // habilita display
        dado = 0;       // zera a variável de leitura
        // lê os quatro bits mais significativos
        if (input(lcd_d7)) bit_set(dado,7);
        if (input(lcd_d6)) bit_set(dado,6);
        if (input(lcd_d5)) bit_set(dado,5);
        if (input(lcd_d4)) bit_set(dado,4);
        // dá um pulso na linha enable
        output_low(lcd_enable);
        output_high(lcd_enable);
        // lê os quatro bits menos significativos
        if (input(lcd_d7)) bit_set(dado,3);
        if (input(lcd_d6)) bit_set(dado,2);
        if (input(lcd_d5)) bit_set(dado,1);
        if (input(lcd_d4)) bit_set(dado,0);
        output_low(lcd_enable);     // desabilita o display
        return dado;  // retorna o byte lido
}

void lcd_envia_nibble( byte dado )
```

```
// envia um dado de quatro bits para o display
{
        // coloca os quatro bits nas saidas
        output_bit(lcd_d4,bit_test(dado,0));
        output_bit(lcd_d5,bit_test(dado,1));
        output_bit(lcd_d6,bit_test(dado,2));
        output_bit(lcd_d7,bit_test(dado,3));
        // dá um pulso na linha enable
        output_high(lcd_enable);
        output_low(lcd_enable);
}


void lcd_envia_byte( boolean endereco, byte dado )
{
        // coloca a linha rs em 0
        output_low(lcd_rs);
        // aguarda o display ficar desocupado
        //while ( bit_test(lcd_le_byte(),7) ) ;
        // configura a linha rs dependendo do modo selecionado
        output_bit(lcd_rs,endereco);
        delay_us(100);          // aguarda 100 us
        // caso a linha rw esteja definida, coloca em 0
        #ifdef lcd_rw
                output_low(lcd_rw);
        #endif
        // desativa linha enable
        output_low(lcd_enable);
        // envia a primeira parte do byte
        lcd_envia_nibble(dado >> 4);
        // envia a segunda parte do byte
        lcd_envia_nibble(dado & 0x0f);
```

```
}


void lcd_ini()
// rotina de inicialização do display
{
        byte conta;

        output_low(lcd_d4);
        output_low(lcd_d5);
        output_low(lcd_d6);
        output_low(lcd_d7);

        output_low(lcd_rs);

        #ifdef lcd_rw
                output_high(lcd_rw);
        #endif

        output_low(lcd_enable);
        delay_ms(15);

        // envia uma seqüência de 3 vezes 0x03
        // e depois 0x02 para configurar o módulo
        // para modo de 4 bits
        for(conta=1;conta<=3;++conta)
        {
                lcd_envia_nibble(3);
                delay_ms(5);
        }

        lcd_envia_nibble(2);
```

```
        // envia string de inicialização do display
        for(conta=0;conta<=3;++conta) lcd_envia_byte(0,INI_LCD[conta]);
}

void lcd_pos_xy( byte x, byte y)
{
  byte endereco;
  if(y!=1)
        endereco = lcd_seg_lin;
  else
        endereco = 0;
  endereco += x-1;
  lcd_envia_byte(0,0x80|endereco);
}

void lcd_escreve( char c)
// envia caractere para o display
{
  switch (c)
        {
    case '\f'    :      lcd_envia_byte(0,1);
                        delay_ms(2);
                        break;
    case '\n'    :
    case '\r'    :      lcd_pos_xy(1,2);
                        break;
    case '\b'    :      lcd_envia_byte(0,0x10);
                        break;
    default      :      lcd_envia_byte(1,c);
                        break;
```

```
        output_high(lcd_enable);

    }
}

char lcd_le( byte x, byte y)
// le caractere do display
{
        char valor;
        // seleciona a posição do caractere
        lcd_pos_xy(x,y);
        // ativa rs
        output_high(lcd_rs);
        // lê o caractere
        valor = lcd_le_byte();
        // desativa rs
        output_low(lcd_rs);
        // retorna o valor do caractere
        return valor;
}
```

ANEXO V
PROGRAMAÇÃO E GRAVAÇÃO DO MICROCONTROLADOR

ANEXO 5 - PROGRAMAÇÃO E GRAVAÇÃO DO MICROCONTROLADOR

Serão abordados aspectos da programação e da gravação do microcontrolador.

5.1 - GRAVADOR DO MICROCONTROLADOR

Neste projeto foi utilizado o gravador de microcontrolador que está na placa comercial Módulo Básico PIC da Microlabs ("Placa de estudos e aplicações"), idealizada e fabricada por Lúcio Hélio da Costa de Santana, conforme mostrada na Fig. (5.1). Visto que o microcontrolador utilizado neste projeto foi o de 40 pinos, a placa foi utilizada somente para gravação do mesmo, tendo sido montado o módulo de teste em protoboard.

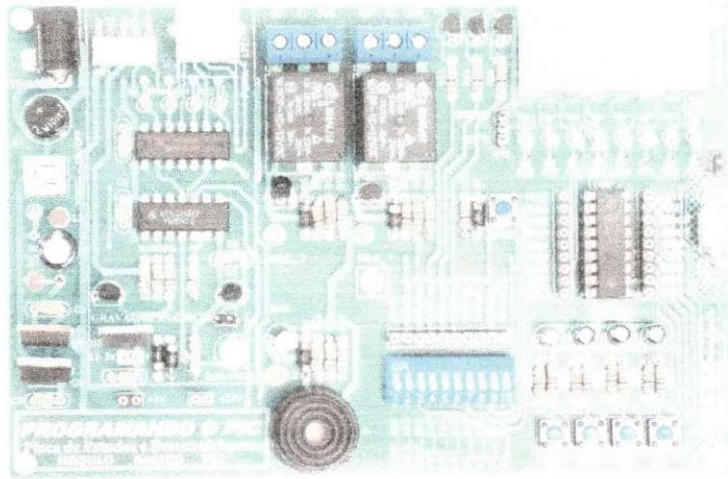

Figura 5.1 - Placa comercial da Microlabs para treinamento e projeto de microcontroladores PIC 16F628A (18 pinos).

O esquema de ligação para gravação é ilustrado na Fig. (5.2), onde as ligações eventualmente utilizadas dos pinos RB6 e RB7 do PIC de 40 pinos devem ser ligadas nos pinos RB6 e RB7 da aplicação que vêm da placa de treinamento.

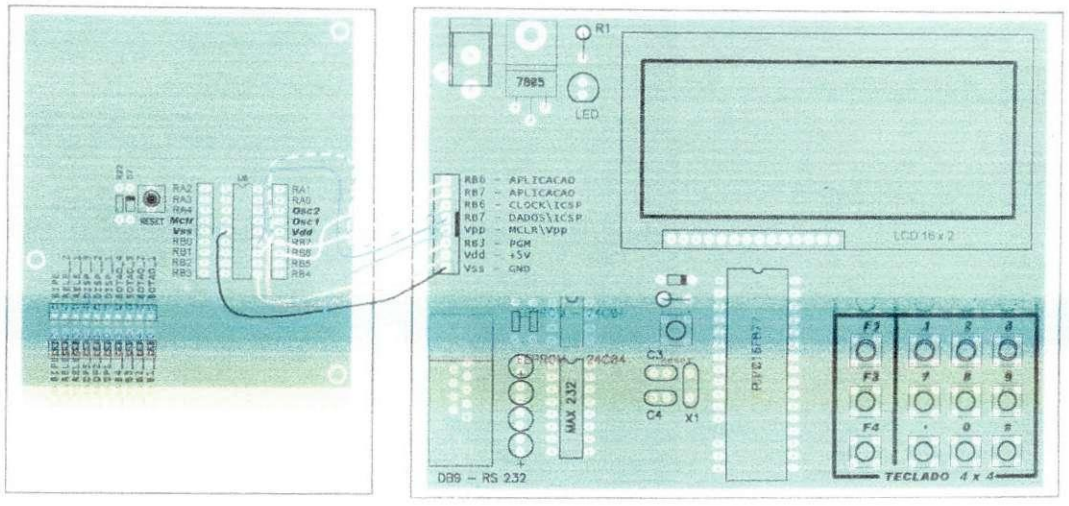

Figura 5.2 - Esquema de ligação entre a placa da Microlabs para pics de 18 pinos e a aplicação em outros tipos de pics.

Uma outra opção exclusivamente para gravação do microcontrolador é a sua montagem, conforme Fig. (5.3). O PIC possui sistema serial de gravação, possuindo um pino para ativação do modo programação, esse é que tem que estar com a tensão alta (13 V); um pino de comunicação bidirecional, para gravar e ler os programas; um pino de clock; e logicamente a alimentação normal 5V e GND. Os pinos acima são de multiplas funções, assim no modo normal são usados como /MCLR , RB7, RB6, VDD e VSS.

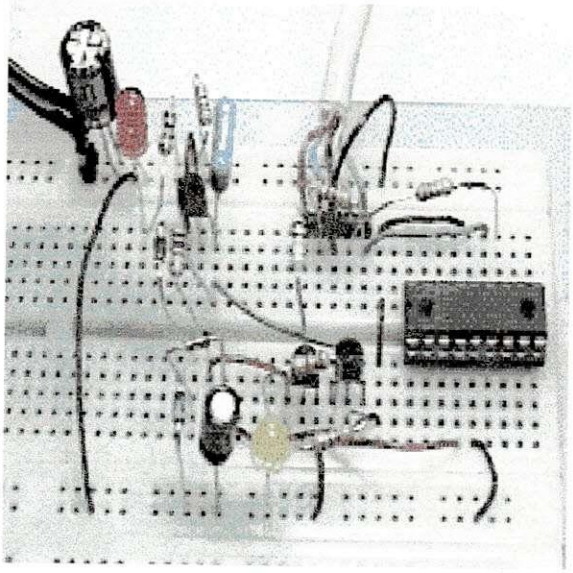

Figura 5.3 - Esquema de um gravador para pics que pode ser montado.

Seu diagrama elétrico é mostrado na Fig. (5.4).

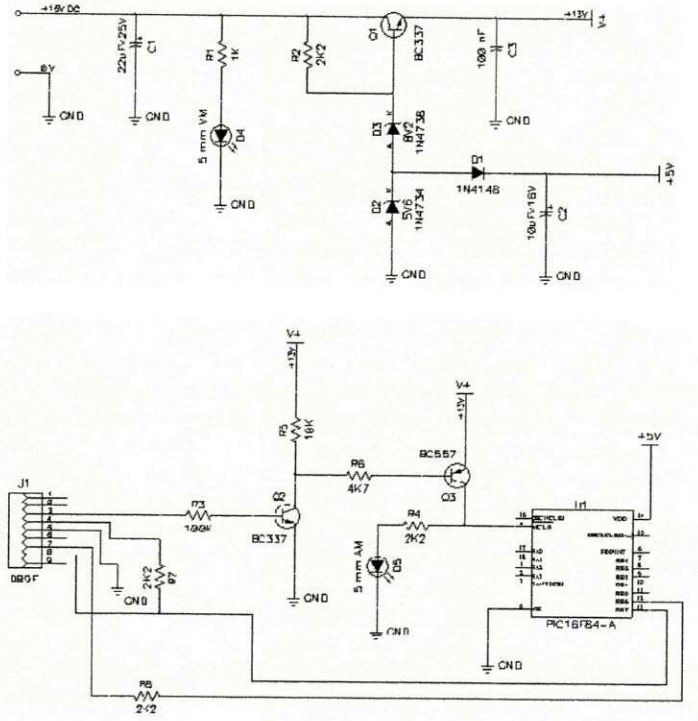

Figura 5.4 - Esquema elétrico do gravador para Pics (opcional).

5.2 - A PROGRAMAÇÃO DO MICROCONTROLADOR

Neste projeto optou-se pela programação na linguagem C++, utilizando o compilador da CCS, as Figs.(5.5) à (5.11) ilustram passo a passo desde a criação de um projeto até sua compilação e geração de um arquivo em hexadecimal para gravação na memória do PIC.

Na Figura 5.5 é mostrada a inicialização do software para programação em C++.

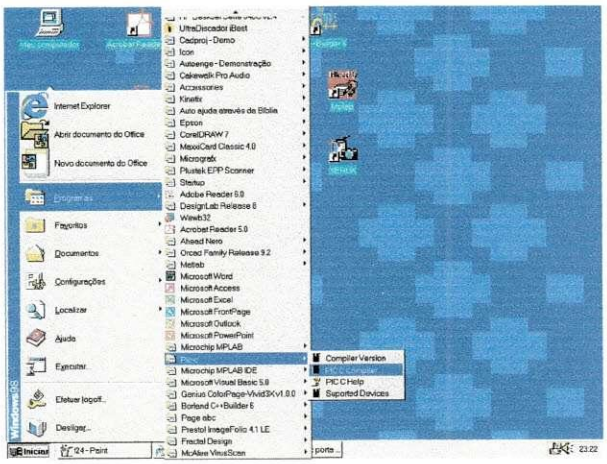

Figura 5.5 - Iniciando o Pic C Compiler

Na Figura 5.6 é mostrada a criação de um projeto para iniciar a fase de programação.

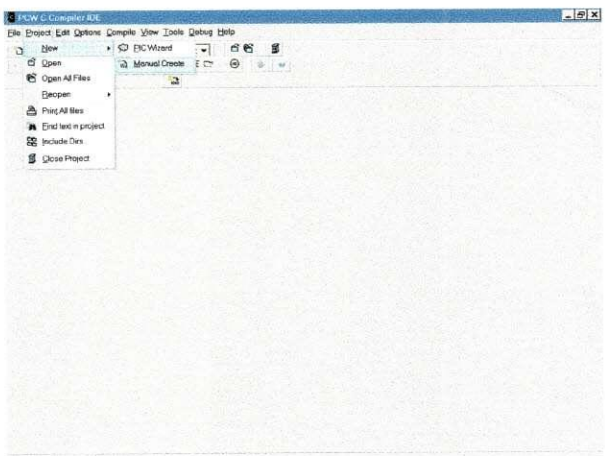

Figura 5.6 - Criando um projeto

Na Figura 5.7 é mostrada a criação de um projeto no modo manual.

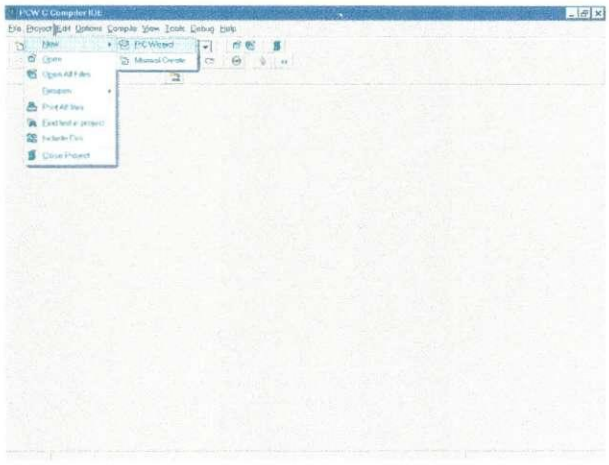

Figura 5.7 - Criando um projeto modo manual

Na Figura 5.8 é mostrada a alocação, salvamento do projeto na pasta alvo de trabalho.

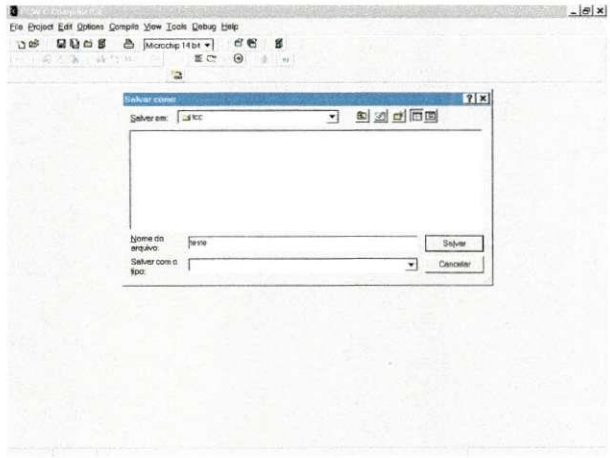

Figura 5.8 - Indicando pasta onde devem ser alocados todos os arquivos gerados pelo projeto

Na Figura 5.9 é mostrada a escolha do tipo de microcontrolador PIC com suas respectivas configurações.

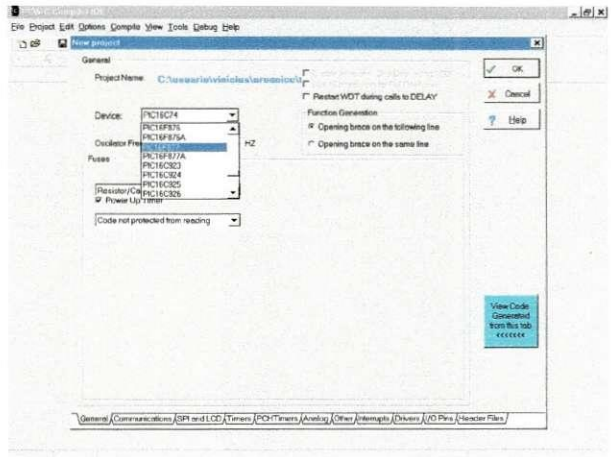

Figura 5.9 - Indicando o tipo de microcontrolador a ser utilizado, no caso o PIC 16F877.

Na Figura 5.10 é mostrado o programa já digitado referente à aplicação desejada, configurado o PIC com o respectivo algoritmo a ser programado.

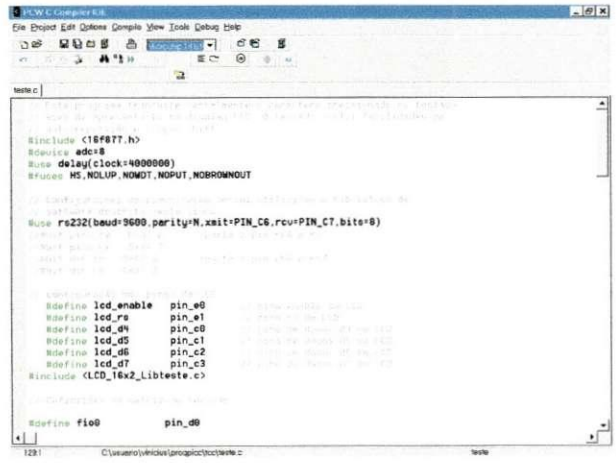

Figura 5.10 - Digitação do código associado ao projeto.

Na Figura 5.11 é mostrado o início da compilação do programa.

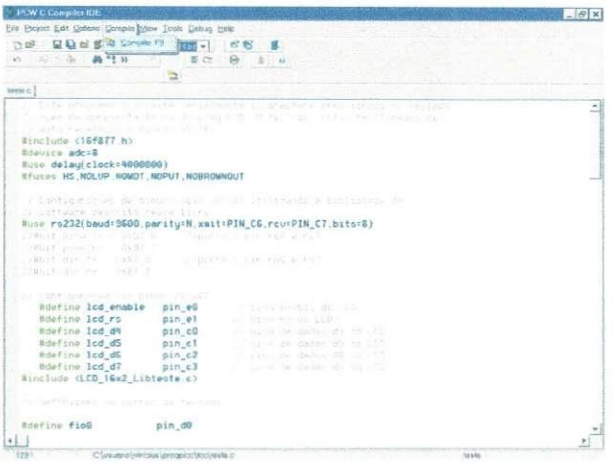

Figura 5.11 - Compilação do projeto e geração dos arquivos em Hexadecimal para gravação.

As Figuras (5.12) à (5.15) ilustram desde a inicialização do programa utilizado para gravação, o EPICWIN, até o momento do envio dos dados para gravação.

Na Figura 5.12 é mostrado a execução do programa EPICWIN a partir da pasta na qual está alocada.

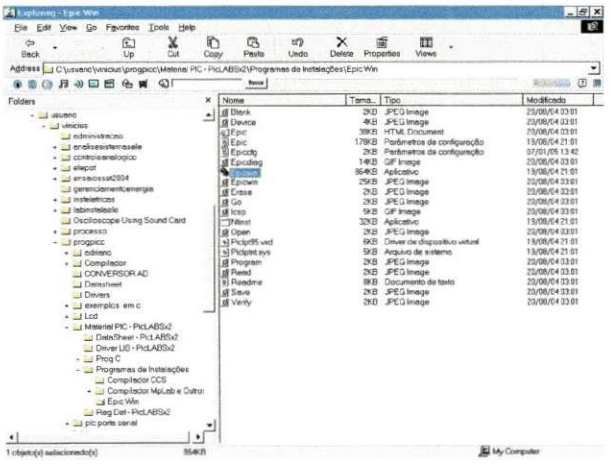

Figura 5.12 - Inicialização do EPICWIN a partir de sua pasta de trabalho.

Na Figura 5.13 é mostrado o ambiente do EPICWIN a ser utilizado na gravação do PIC.

Figura 5.13 - Visão do software EPICWIN quando inicializado.

Na Figura 5.14 é mostrado a abertura de extensão .HEX gerada pelo compilador a ser gravado no PIC.

Figura 5.14 - Abertura do arquivo compilado em hexadecimal para gravação.

Na Figura 5.15 é mostrado a escolha do tipo de microcontrolador a ser gravado.

Figura 5.15 - Indicação do tipo de microcontrolador PIC utilizado.

Na Figura 5.16 é mostrado a abertura de extensão .HEX gerada pelo compilador a ser gravado no PIC.

Figura 5.16 - Indicação do arquivo com extensão .HEX

Na Figura 5.17 é mostrado o comando que inicializa a gravação via porta paralela para o gravador do PIC.

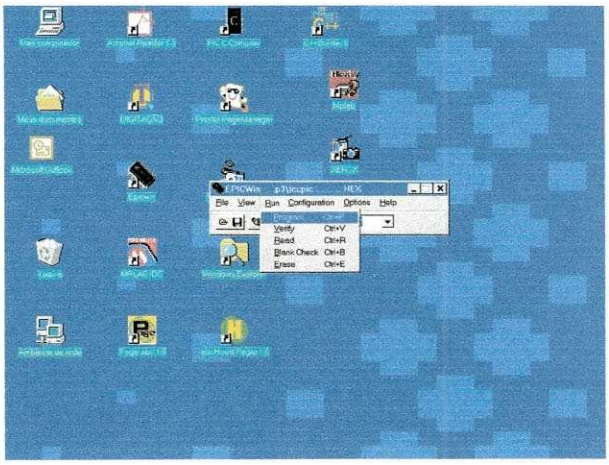

Figura 5.17 - Início da gravação do microcontrolador.

Na Figura 5.18 é mostrado o acoplamento entre a placa comercial fabricada pela empresa do técnico Lúcio Hélio e a montagem em protoboard do PIC de 40 pinos.

Figura 5.18 - Início da gravação do microcontrolador.

Na Figura 5.19 é mostrado outra vista do mesmo acoplamento.

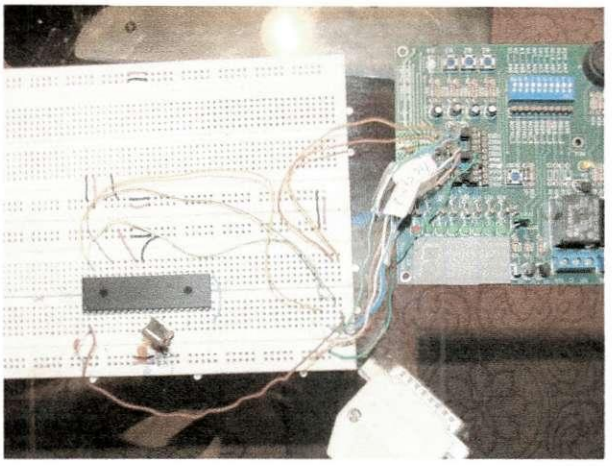

Figura 5.19 - Início da gravação do microcontrolador, outra vista.

Nos final deste trabalho estão os apêndices referentes ao programa principal utilizado na programação deste projeto, à comunicação entre o pic e o LCD, conforme programa desenvolvido por Fábio Pereira e divulgado em seu livro, e as definições do PIC relativas à linguagem C++ do compilador quando do include relativo ao PIC 16F877.

ANEXO VI

MODELO PARA LAUDO TÉCNICO

TRANSFORMADORES CAMPINENSE
CONSERTO E MONTAGEM DE TRANSFORMADORES ELÉTRICOS
R: Mestre Inácio, 25-D – Distrito dos Mecânicos – fone (0xx83) 331-2776 Campina Grande-PB
CGC: 41.208./0001-18 CEP- 58–106 –058

ENSAIO DE ROTINA DE TRASFORMADOR

Cliente:		**Ordem de Serviço**
Marca:	**Tensões de A.T.**	**Corrente A.T.:**
Potência:	**Tensões de B.T**	**Corrente B.T.:**
Série:	**Freqüência**	**Data de fabricação:**
Fases:	**Derivações:**	**Data de Reforma:**
Ligação:	**A.T. Ligado em:**	**Estado:**
Tipos:	**B.T. Ligado em:**	**Peso**

RESULTADOS DOS ENSAIOS

RELAÇÃO DE TRANSFORMAÇÃO	**RESISTÊNCIA ELÉT. DOS ENROLAMENTOS**
Tensões (V)	**H1.H2 (X1/X2**
Relação Nominal	**H1.H3 X1/X3**
H1.H3/X0.X1	**H2.H3 X2/X3**
H2.H1/X0.X2	**Temperatura**
H3.H2/X0.X3	
Erro (%)	

RESISTÊNCIA DE ISOLAÇÃO	**TENSÃO INDUZIDA**	**TENSÃO APLICADA AO DIELÉTRICO**
AT/BT (MΩ):	**Tensão:**	**A.T/BT á Massa (kv):**
AT/ Massa (MΩ):	**Freqüência:**	**B.T/A.T á Massa (kv):**
BT/ Massa (MΩ):	**Temperatura:**	**Temp. Ambiente (°C):**
Megometro (Volts):	**Duração da Leit.**	**Duração Leitura (S):**
Temperatura (°C):	**Resultado:**	**Resultado:**
Duração da Leitura(S):		

PERDAS EM VAZIO	PERDAS **NO COBRE**	**VALORES CORRIGIDOS PARA 75°C**
Tensão de ensaio (V)	**Tensão de C. Circ. (V)**	**Perdas Curto Circ. (W)**
Corrente Excitação (A)	**Corrente C. Circ. (A)**	**Perdas Totais (W)**
Corrente Excitação (%)	**Perda de C. Circ.**	**mpedância (%)**
Perdas em Vazio (W)	**Temperatura (°C)**	**Rendimento (%)**

Óleo Isolante	**PINTURA (NBR 10443)**	**DIVERSOS**
Tipo:		
Volume:	**Pintura de Fundo**	**Polaridade:**
Rigidez Dielétrica:	**Aderência**	**Estanqueiedade**
Classificação:	**Pintura Final**	**Regulação à 75°C**

OBS: Este transformador foi ensaiado nas tensões de referência _____ volts, na temperatura de_____apresentando resultados compatíveis com as normas em vigor, portanto o mesmo está _____pelo nosso controle de qualidade.

DATA **SUPERVISOR** **RESPONSAVEL PELO ENSAIO**