



Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Departamento de Engenharia Elétrica

Ricardo Soares Chinarro

Desenvolvimento de *hardware* e *software* de controle para os conversores CA/CC e CC/CC acoplados a uma placa genérica para o laboratório de eletrônica de potência

Campina Grande, Paraíba.

Novembro de 2014

RICARDO SOARES CHINARRO

Desenvolvimento de *hardware* e *software* de controle para os conversores CA/CC e CC/CC acoplados a uma placa genérica para o laboratório de eletrônica de potência

Relatório de Trabalho de conclusão de curso submetido ao Departamento de Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica.

Área de Concentração: Eletrônica Industrial

Orientador:

Professor Alexandre Cunha Oliveira, Dr.

Campina Grande, Paraíba.

Novembro de 2014

RICARDO SOARES CHINARRO

Desenvolvimento de *hardware* e *software* de controle para os conversores CA/CC, CC/CC acoplados a uma placa genérica para o laboratório de eletrônica de potência

Relatório de Trabalho de conclusão de curso submetido ao Departamento de Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica.

Área de Concentração: Eletrônica Industrial

Aprovado em / /

Professor Avaliador

Universidade Federal de Campina Grande
Avaliador

Professor Alexandre Cunha Oliveira, Dr. Sc.

Universidade Federal de Campina Grande
Orientador

Dedico este aos meus pais, Adolfo e Andrea e à minha namorada, Ana Gabrielle, por sempre me apoiarem e acreditarem no sucesso da minha jornada.

A mente que se abre a uma nova ideia
jamais voltará ao seu tamanho original.

Albert Einstein

Agradecimentos

Agradeço a Deus por ter me dado a oportunidade e força necessária para que fosse possível vencer todos os obstáculos durante o curso e pela experiência de vida proporcionada ao decorrer destes anos.

Aos meus pais, Adolfo e Andrea, por terem sempre me apoiado, orientado e acreditado em minha capacidade.

À minha namorada, Ana Gabrielle, que sempre me compreendeu, ajudou, ouviu e esteve ao meu lado em todos os momentos.

Aos professores orientadores José Gutemberg e Alexandre Cunha por toda a sua ajuda durante o curso de graduação e por serem exemplos de professores.

Todos os meus amigos que compartilharam esta batalha, em especial Ezequiel e Thiago Balbino.

Resumo

Este trabalho tem como objetivo o desenvolvimento de uma placa de controle genérica, para os conversores CA/CC, CC/CC e CC/CA. No trabalho foram desenvolvidos *software* e *hardware* para os conversores de eletrônica de potência, com exceção do conversor CC/CA que foi desenvolvido apenas o hardware faltando o desenvolvimento do software.

Índice de Ilustrações

Figura 1: Tipo de chaves utilizadas em diversos conversores.....	2
Figura 2: Retificador monofásico em ponte completa semi-controlada (A) e controlada (B).....	3
Figura 3: Retificador trifásico em ponte completa controlada.....	3
Figura 4: SCR.....	4
Figura 5: Sinal dente de serra (A) e cosenoidal (B).....	4
Figura 6: Conversor Buck.....	6
Figura 7: Conversor Boost.....	6
Figura 8: Conversor Buck-Boost.....	6
Figura 9: IGBT (A) e MOSFET (B).....	7
Figura 10: Sinal de referência sinal PWM.....	8
Figura 11: IHM.....	9
Figura 12: Módulo de potência CA/CC.....	9
Figura 13: Módulo de potência CC/CC.....	10
Figura 14: Módulo de potência CC/CA.....	10
Figura 15: Placa de controle genérica.....	11
Figura 16: Conexões microcontrolador.....	12
Figura 17: Circuito de reset, e circuito oscilador.....	12
Figura 18: Fonte de alimentação primária.....	13
Figura 19: Fonte de alimentação secundária.....	13
Figura 20: Sinal de referência.....	14
Figura 21: Sinal de sincronismo com a rede.....	14
Figura 22: Circuito disparo SCR.....	15
Figura 23: Circuito de disparo do IGBT.....	16
Figura 24: sinal dente de serra.....	18
Figura 25: Sinal Cosenoidal.....	18
Figura 26: Conversor D/A R-2R.....	19
Figura 27: Fresadora LPKF.....	20
Figura 28: Esquemático circuito.....	21
Figura 29: layout PCI.....	22
Figura 30: Tela Inicial CA/CC.....	23
Figura 31: Seleção de referência.....	24
Figura 32: Disparo 180°.....	24
Figura 33: Disparo 164°.....	24
Figura 34: Tela inicial CC/CC.....	25
Figura 35: Tipo de conversor.....	25
Figura 36: Seleção de frequência.....	26
Figura 37: Ciclo de trabalho ou tensão de saída.....	26
Figura 38: Escolha ciclo de trabalho.....	27
Figura 39: Tela final.....	27
Figura 40: Rotina principal conversor CA/CC.....	28
Figura 41: Rotina das interrupções Timer 0, Timer 1, Timer 2, externa 1 e externa 2.....	29

Figura 42: Interrupção externa 0	29
Figura 43: Interrupção timer 3	30
Figura 44: Condigo principal conversor CC/CC	31
Figura 45: Subrotina e interrupções externa 1, externa 2 e timer 1	32
Figura 46: Circuito ponte controlada completa	33
Figura 47: Circuito de teste retificar controlado onda completa	33
Figura 48: Disparo 40°	34
Figura 49: Disparo 90°	35
Figura 50: Sinal Cosenoidal disparo 90°	35
Figura 51: Sinal rampa disparo 90°	36

LISTA DE ABREVIATURAS E SIGLAS

MOSFET - *Metal Oxide Semiconductor Field Effect Transistor*

IGBT - *insulated-gate bipolar transistor*

SCR - *Silicon Controlled Rectifier*

PWM – Modulação por largura de pulso

IHM – Interface homem maquina

Sumário

1.	Introdução.....	1
2.	Embasamento Teórico	1
2.1	Conversor CA/CC	2
2.1.1	Funcionamento básico dos tiristores.....	3
2.1.2	Sinais de Referencia	4
2.2	Conversor CC/CC	5
2.2.1	Funcionamento básico dos IGBTs e MOSFETs	6
2.2.2	Sinais de Referência	7
3.	Descrição do projeto.....	8
3.1	Circuitos da placa de controle.....	11
3.1.1	Microcontrolador	11
3.1.2	Fonte de Alimentação.....	12
3.1.3	Sinal de sincronismo com a rede.....	13
3.1.4	Circuitos de disparo	15
3.1.4.1	Disparo do SCR.....	15
3.1.4.2	Disparo do IGBT	16
3.1.5	Sinais gerados pelo microcontrolador	16
4.	Desenvolvimento <i>layout</i> e confecção da PCI	19
5.	Funcionamento do sistema	23
5.1	Funcionamento do conversor CA/CC.....	23
5.2	Funcionamento do conversor CC/CC.....	25
6.	Fluxograma do <i>firmware</i>	27
6.1	Fluxograma <i>firmware</i> conversor CA/CC	28
6.2	Fluxograma <i>firmware</i> conversor CC/CC	31
7.	Análise de resultados.....	32
7.1	Conversor CA/CC ponte completa	33
8.	Possíveis melhorias.....	36
9.	Conclusão.....	37
10.	Bibliografia	38
	APÊNDICE A	39
	APÊNDICE B	49

1. Introdução

Os recursos didáticos disponíveis no Laboratório de Eletrônica de Potência (LEP) apresentam sinais de desgaste em virtude do seu longo tempo de uso e por isso deixaram de ser usados a partir do ano de 2013. Hoje o laboratório de Eletrônica de Potência utiliza apenas simulações para o estudo dos conversores. Por causa desses fatores, foi desenvolvida uma ação de construir módulos didáticos para atender a demanda do laboratório. Nesta ação, serão desenvolvidos novos módulos didáticos para os conversores CA/CC, como também, para os conversores CC/CC (especificamente, os conversores Buck, Boost e Buck-Boost) e em um posterior trabalho o conversor CC/CA. Estes módulos são parte de um trabalho de modernização que está sendo realizado no laboratório de Eletrônica de Potência.

Uma parte dos módulos didáticos já foi desenvolvida. A parte desenvolvida foi a parte de potência e a interface homem máquina. Através da IHM os alunos poderão configurar os sinais que controlam as chaves eletrônica (IGBTs e Tiristores), para cada tipo de conversor existe uma placa de potência, assim existindo três tipos diferentes de placas CA/CC, CC/CC CC/CA. Neste trabalho abordaremos apenas as placas: CA/CC e CC/CC.

2. Embasamento Teórico

Os conversores estáticos são estruturas que utilizam chaves semicondutoras, que permitem a transferência de energia entre a fonte e a carga. O controle do fluxo de energia é realizado através das chaves que atuam em condução ou bloqueio (Figura 1).

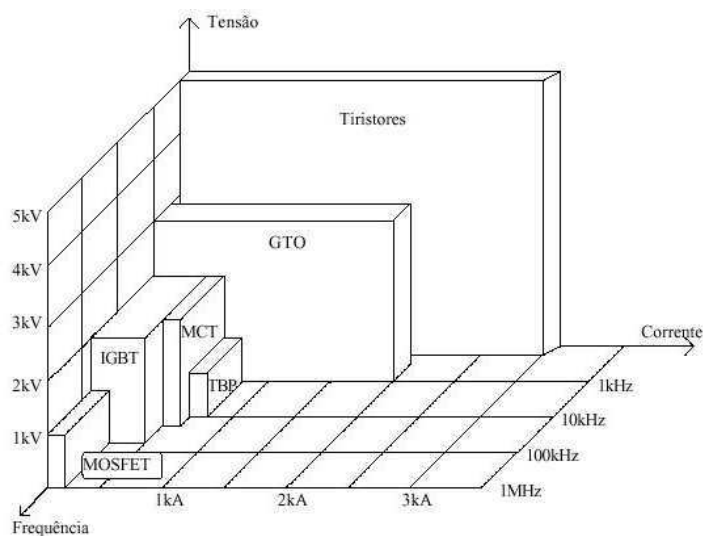


Figura 1: Tipo de chaves utilizadas em diversos conversores

2.1 Conversor CA/CC

Em muitas aplicações a carga exige uma tensão contínua. A conversão de CA/CC é realizada por conversores chamados de retificadores.

Os retificadores podem ser classificados por sua capacidade de ajustar a tensão de saída, sendo eles controlados ou não controlados.

Os retificadores não controlados são aqueles que utilizam diodos como elementos de retificação, enquanto os controlados utilizam tiristores.

Existem diversos tipos de conversores CA/CC, dentre os quais podemos destacar as seguintes topologias:

- Ponte monofásica não controlada;
- Ponte monofásica semi-controlada (Figura 2 A);
- Ponte monofásica totalmente controlada (Figura 2 B);
- Ponte trifásica não controlada;
- Ponte trifásica semi-controlada;
- Ponte trifásica totalmente controlada (Figura 3).

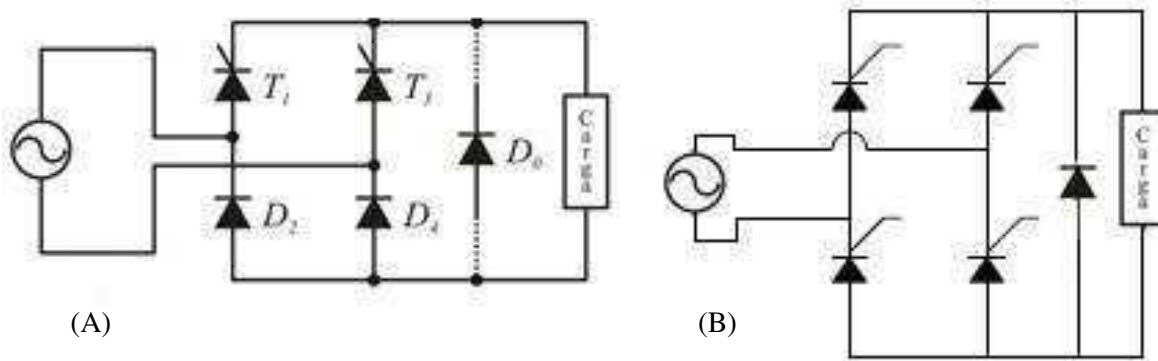


Figura 2: Retificador monofásico em ponte completa semi-controlada (A) e controlada (B)

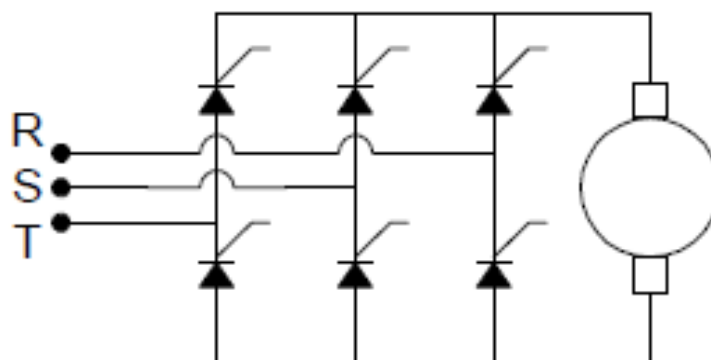


Figura 3: Retificador trifásico em ponte completa controlada

2.1.1 Funcionamento básico dos tiristores

Os conversores CA/CC normalmente utilizam tiristores em seus circuitos. Os tiristores mais utilizados são os retificadores controlados de silício (SCRs) como mostra na (Figura 4).

Os SCRs (Figura 4) são chaves que controlam a passagem de corrente em um circuito. Se aplicarmos uma corrente no terminal *gate* destas chaves, e estas estiverem polarizadas diretamente será iniciada a condução de corrente. Se o SCR estiver polarizado reversamente e aplicarmos um pulso de corrente no terminal *gate* o SCR não entrará em condução. Quando o SCR inicia a condução, ele tem as características de uma chave fechada, quando entrar bloqueio tem as características de uma chave aberta. Em um SCR a

condução cessa quando a corrente vai para zero então o componente muda de estado indo para condição bloqueio (chave aberta).

É sempre necessário consultar o *datasheet* do SCR para saber a corrente mínima para o disparo deste.

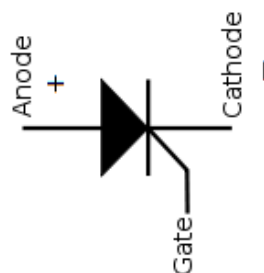


Figura 4: SCR

2.1.2 Sinais de Referencia

São necessários sinais de referência para o controle dos retificadores. Estes estão em fase com a rede elétrica onde está a ponte retificadora.

Os sinais de referência normalmente utilizados para o disparo de SCRs são:

- Dente de serra
- Cosenoidal

Podem ser vistos na Figura 5.

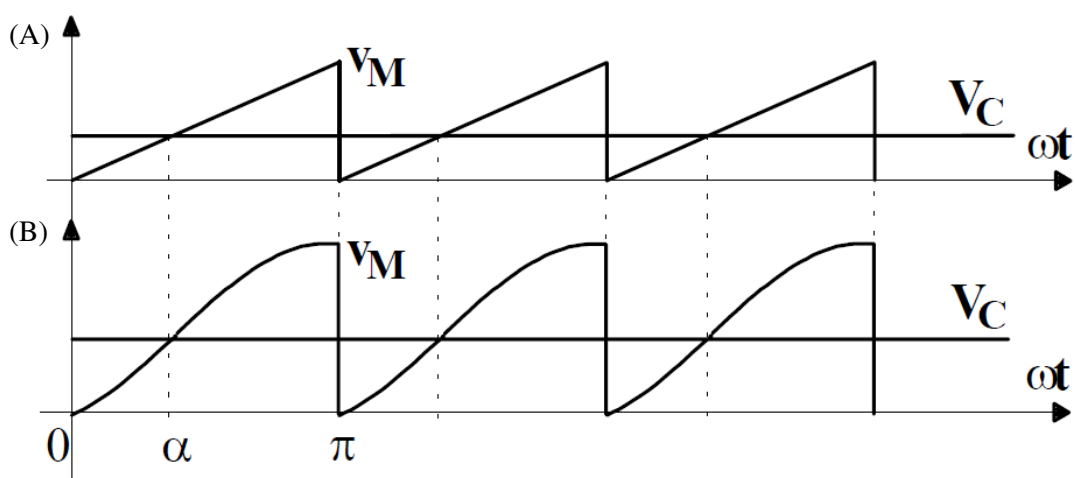


Figura 5: Sinal dente de serra (A) e cosenoidal (B)

A referência cosenoidal resulta em uma característica linear no conversor. Apesar da relação linear ser interessante para a modelagem do sistema, a referência normalmente utilizada é a dente de serra por ser uma solução mais simples.

O ângulo de disparo α pode ser encontrado a partir dos valores de pico de V_c e V_m

- Referência sinal dente de serra:

$$\alpha = 180^\circ \frac{V_c}{V_{pico}}$$

- Referência sinal cosenoidal:

$$\alpha = \arccos\left(\frac{V_{Pico} - 2V_c}{V_{Pico}}\right)$$

2.2 Conversor CC/CC

Os conversores CC/CC são reguladores de tensão eletrônicos. Estes reguladores, têm a finalidade de fornecer a tensão e corrente necessária para o circuito de forma constante e com alta eficiência. As aplicações destes conversores são o controle de motores de corrente contínua, fontes de alimentação chaveadas, carregadores de baterias entre outros.

O controle da tensão de saída nestes conversores é feito pelo ciclo ativo do sinal PWM aplicados as chaves.

Topologia de alguns conversores CC/CC:

- Conversor Buck (Figura 6);
- Conversor Boost (Figura 7);
- Conversor Buck-Boost (Figura 8).

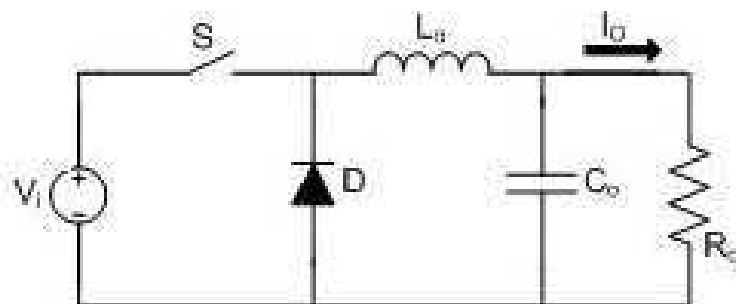


Figura 6: Conversor Buck

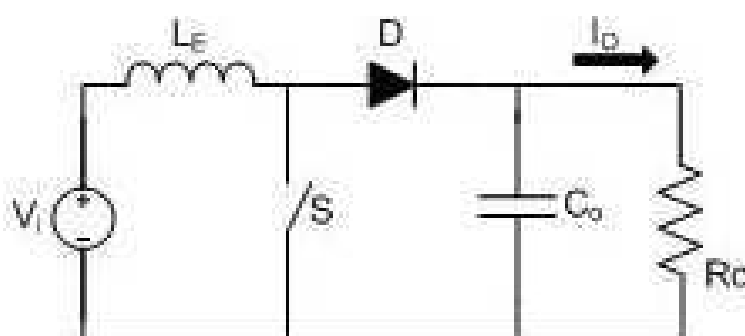


Figura 7: Conversor Boost

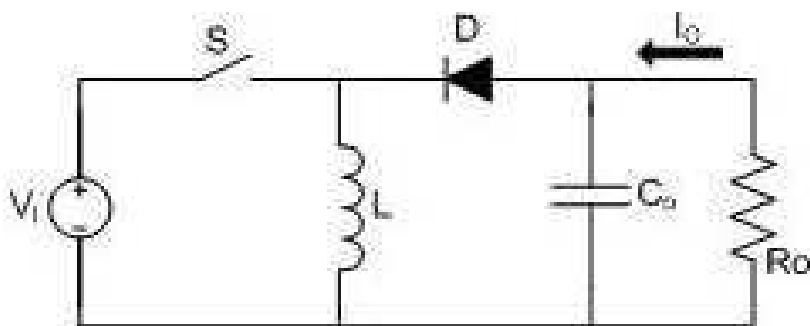


Figura 8: Conversor Buck-Boost

2.2.1 Funcionamento básico dos IGBTs e MOSFETs

As chaves utilizadas normalmente são IGBTs ou Mosfet (Figura 9). Nestes conversores as chaves têm que trabalhar no regime de saturação e corte.

Estas chaves controladas atuam em dois estados *ON/OFF*, normalmente sendo utilizados sinais PWM para o comando destas chaves. Quando as chaves estão no estado *ON* elas podem ser vistas como um fio. Quando no estado *OFF* as chaves funcionam como uma impedância infinita.

Os IGBTs entram em regime de saturação quando a tensão entre *gate* e emissor atinge o nível V_{GE} especificado para o componente. Os MOSFETs entram em regime de saturação quando a tensão entre *gate* e *source* atinge o nível V_{GS} especificado para o componente.

Exemplo: Para o IGBT IRG4BC20U entrar em regime de saturação a tensão $V_{GE} = 15V$. Para o MOSFET IRF540 entrar em regime de saturação a tensão necessária V_{GS} é de 10V.

Pode-se observar que existem diferentes tipos de MOSFETs e IGBTs, e as tensões para que trabalhem no regime de saturação variam de dispositivo para dispositivos, sendo sempre necessárias consultas ao *datasheet* do componente.

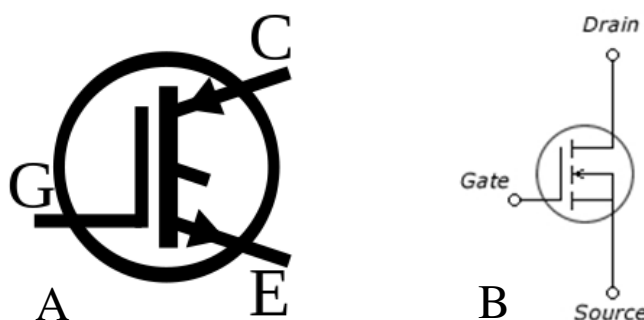


Figura 9: IGBT (A) e MOSFET (B)

2.2.2 Sinais de Referência

São necessários sinais de referência (Figura 10) para o controle dos conversores CC/CC. Para isso é utilizado a modulação por largura de pulso (PWM). Como referência temos a necessidade de um sinal triangular para gerar o sinal PWM que vai ativar e desativar a chave do conversor CC/CC. Para isso, utilizam-se circuitos eletrônicos para gerar um sinal triangular ou dente de serra na frequência desejada e modularmos afim de obter o sinal PWM.

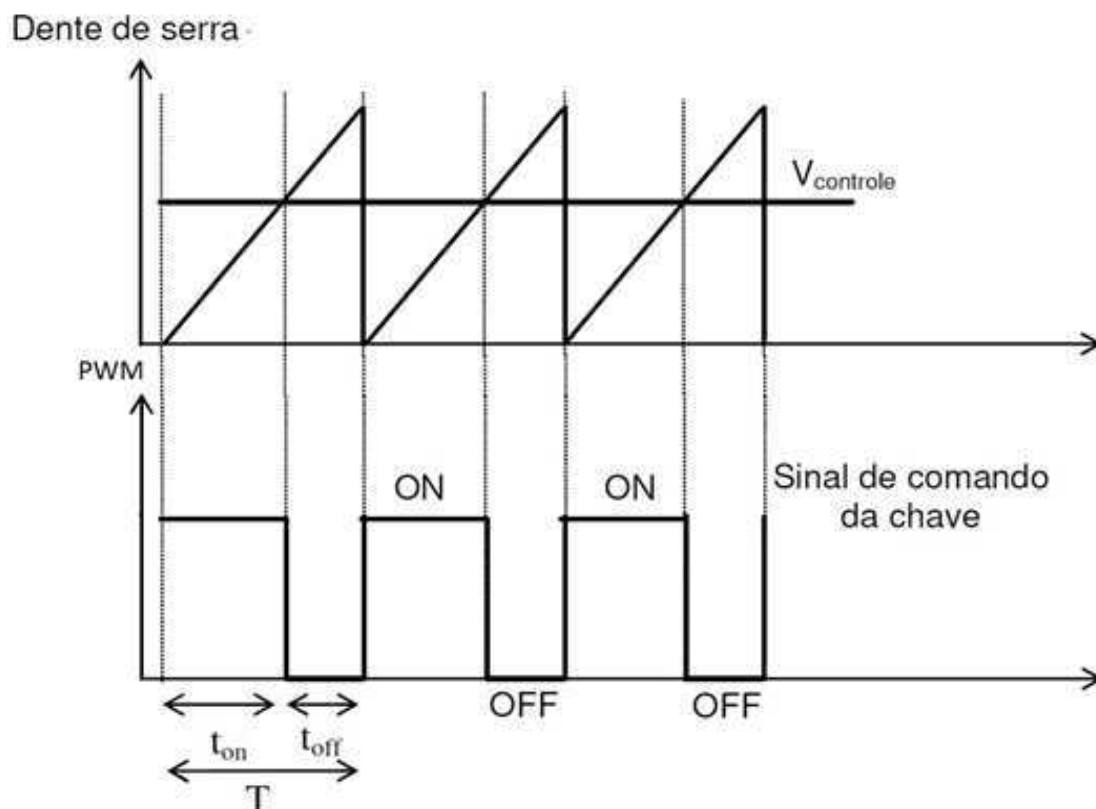


Figura 10: Sinal de referência sinal PWM

3. Descrição do projeto

O projeto consiste na compreensão das placas genéricas de potência e no desenvolvimento de uma placa de controle genérica para o comando dos conversores CA/CC, CC/CC e CC/CA.

Os módulos didáticos são compostos por uma IHM (Figura 11), placas de potência genérica (Figura 12), (Figura 13), (Figura 14) e a placa de controle genérica (Figura 15).

Nas placas de potência estão montados os dispositivos: IGBTs, MOSFETs, diodos, capacitores, indutores, unidades de proteção e um transformador (este não está soldado à placa, mas faz parte dos módulos e conectado através de conectores).

A placa de controle contém os circuitos necessários para o disparo dos SCRs, e MOSFETs utilizados nas placas de potência.

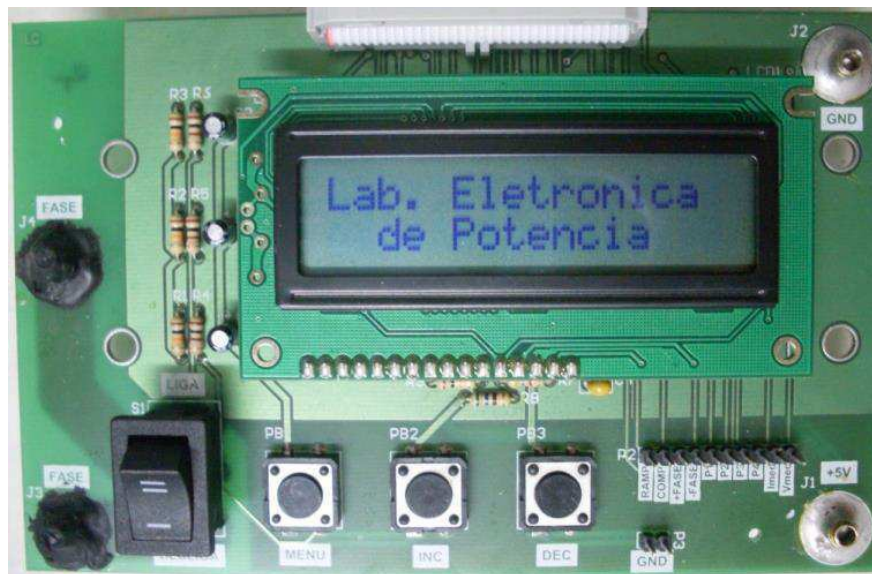


Figura 11:IHM

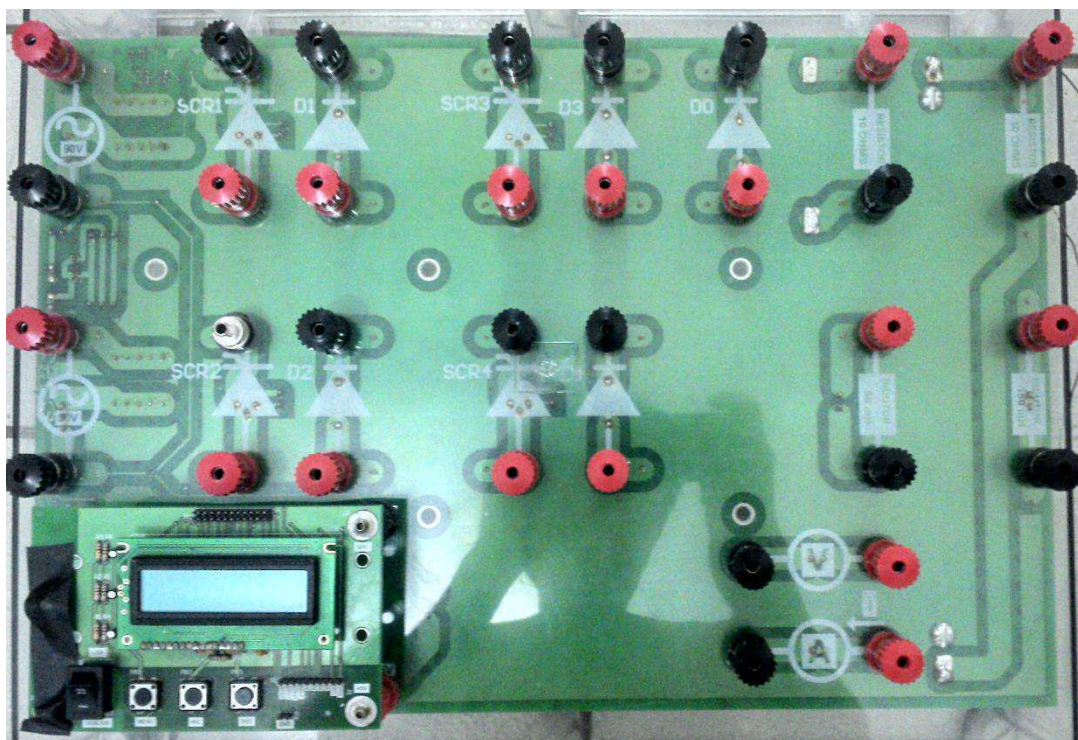


Figura 12: Modulo de potência CA/CC

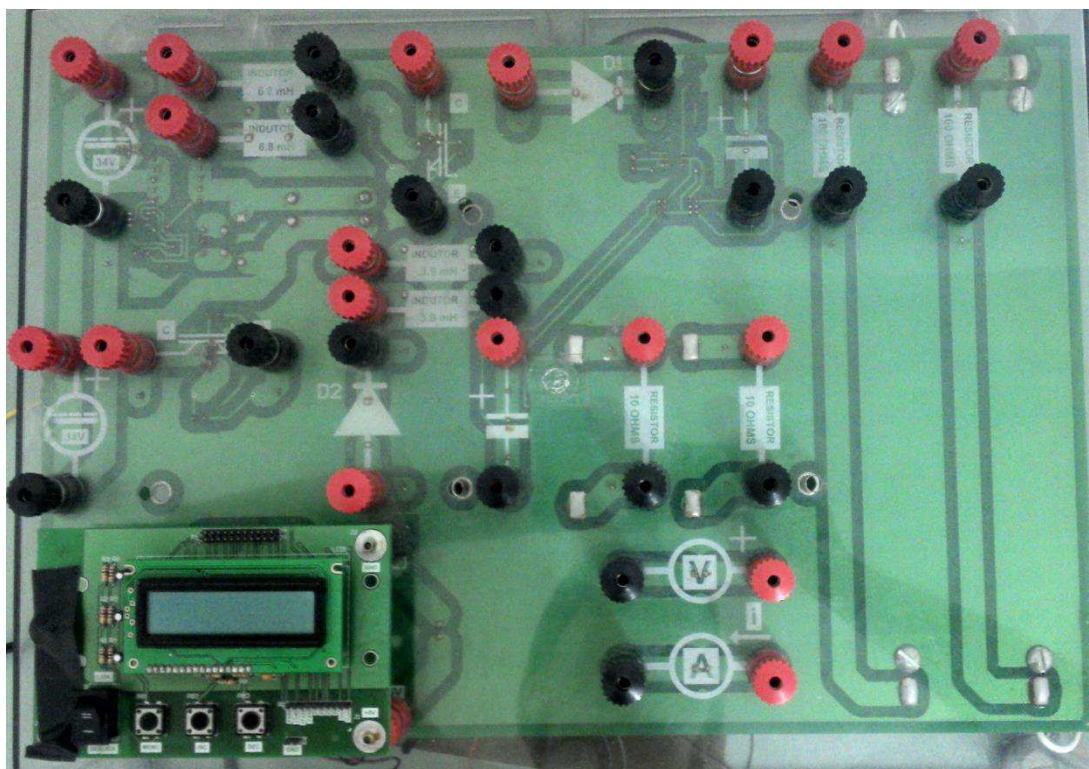


Figura 13:Modulo de potência CC/CC

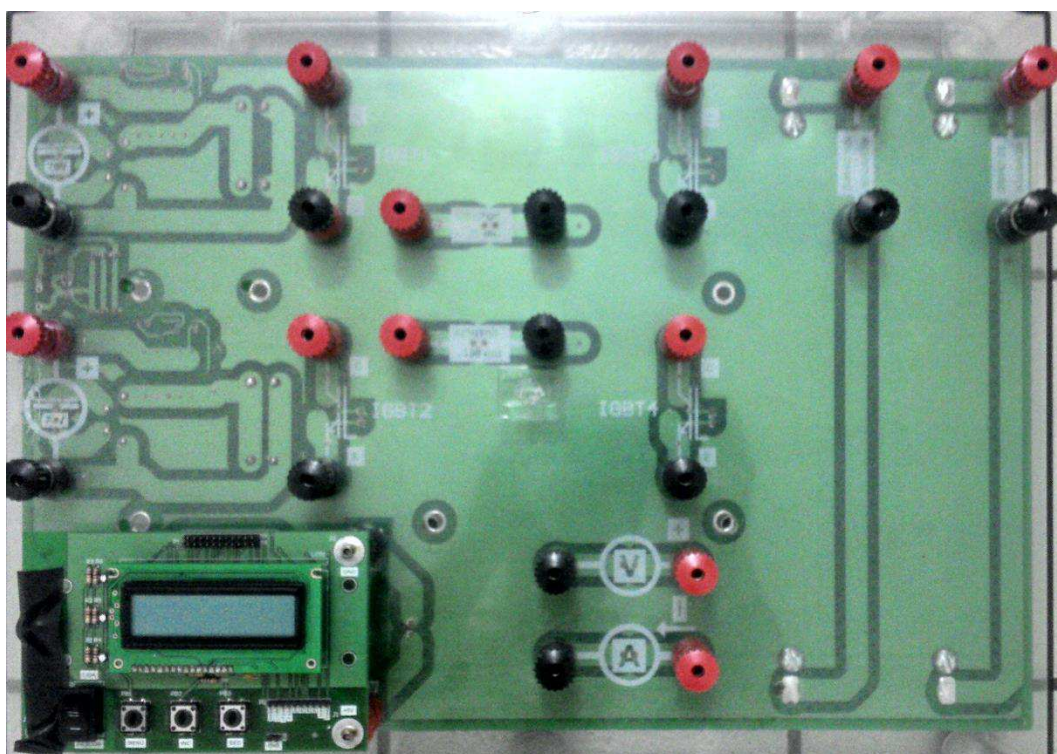


Figura 14:Modulo de potência CC/CA

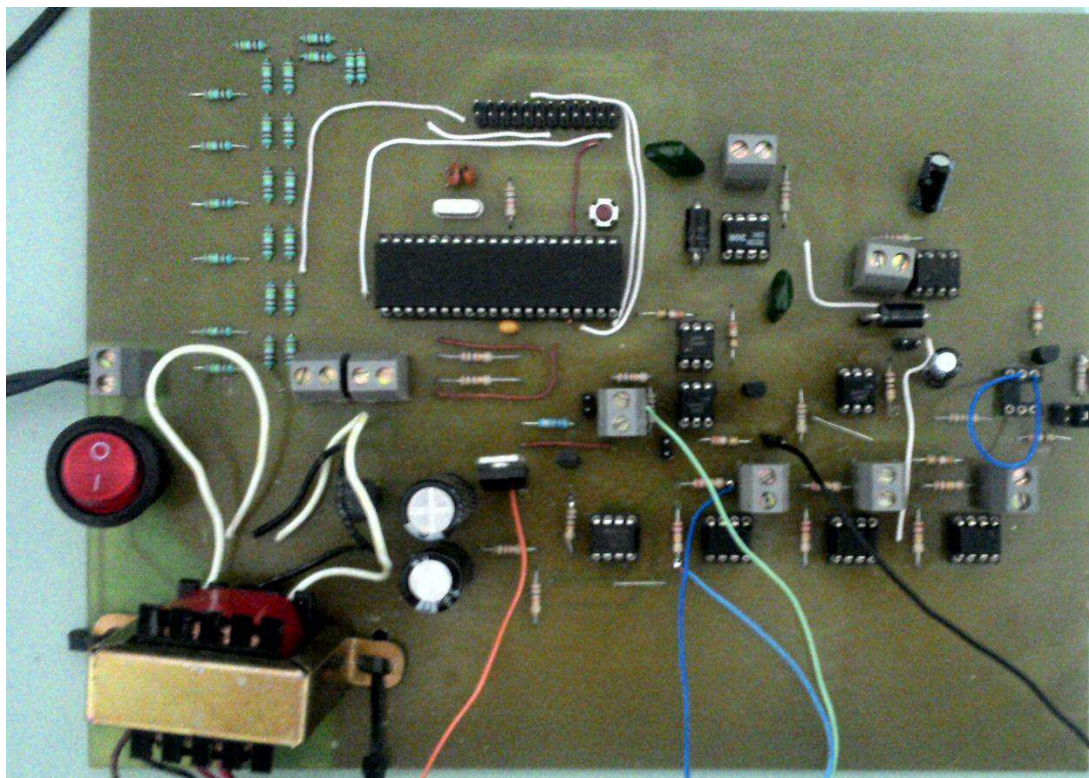


Figura 15: Placa de controle genérica

3.1 Circuitos da placa de controle

Nesta seção serão abordados todos os componentes utilizados na placa de controle e os circuitos desenvolvidos.

3.1.1 Microcontrolador

Utilizou-se o microcontrolador PIC18F4525 (Figura 16), arquitetura *RISC* da Microchip, para realizar todo o processamento de controle da placa. Este possui instruções como multiplicador em *hardware*, interrupções com atribuição de prioridades, multiplicador de *clock* interno, possibilidade de trabalhar com velocidade máxima de 10MIPS (milhões de instruções por segundo), quatro *timers* sendo um de 8bits e três 16bits, 36 de pinos de entrada e saída, dois PWM com frequências ajustáveis a partir do Timer 2. O circuito do oscilador e *reset* podem ser encontrados na (Figura 17).

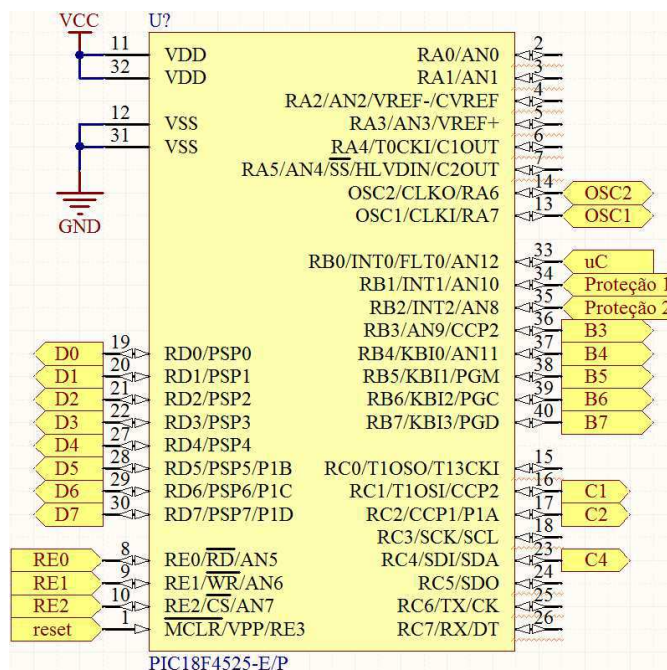


Figura 16: Conexões microcontrolador

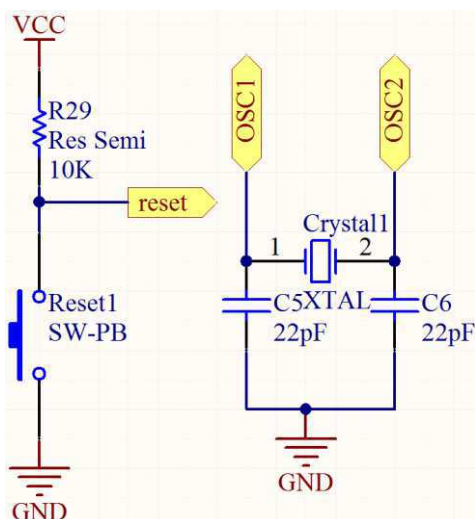


Figura 17: Circuito de *reset*, e circuito oscilador

3.1.2 Fonte de Alimentação

Neste projeto utilizou-se duas fontes de alimentação lineares, pois existia a necessidade de uma segunda fonte para o disparo dos SCRs nos conversores CA/CC e no conversor CC/CA.

Na fonte de alimentação primária, foi utilizado um transformador de 12V com derivação central, retificando o sinal usando uma ponte retificadora completa, e filtrando

utilizando capacitores de $1000\mu\text{F}$, obtendo as tensões de $\pm 15\text{V}$ para alimentar os circuitos com amplificadores operacionais. Também foi utilizado um regulador de tensão linear de 5V para regular a tensão de 15V e alimentar os circuitos do microcontrolador, dos optoacopladores e da IHM. O circuito da fonte pode ser conferido na (Figura 18).

Na fonte de alimentação secundária, foi utilizado um transformador de 12V com derivação central, então retificando o sinal utilizando dois diodos, filtrando o sinal com um capacitor de $1000\mu\text{F}$ e regulando a tensão com o regulador de tensão de 5V . Esta segunda fonte foi utilizada para o disparo de um SCR e a alimentação de um driver de disparo de IGBTs. O circuito da fonte pode ser conferido na (Figura 19)

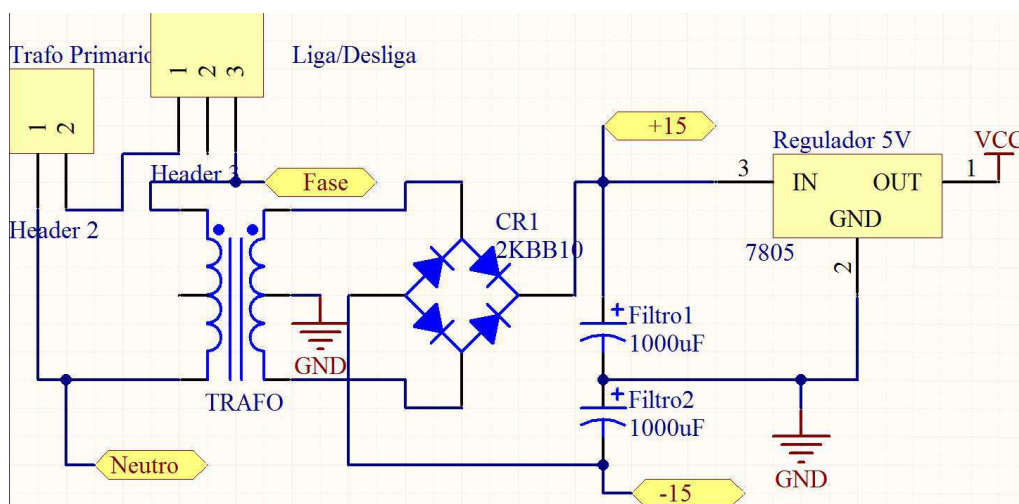


Figura 18: Fonte de alimentação primária

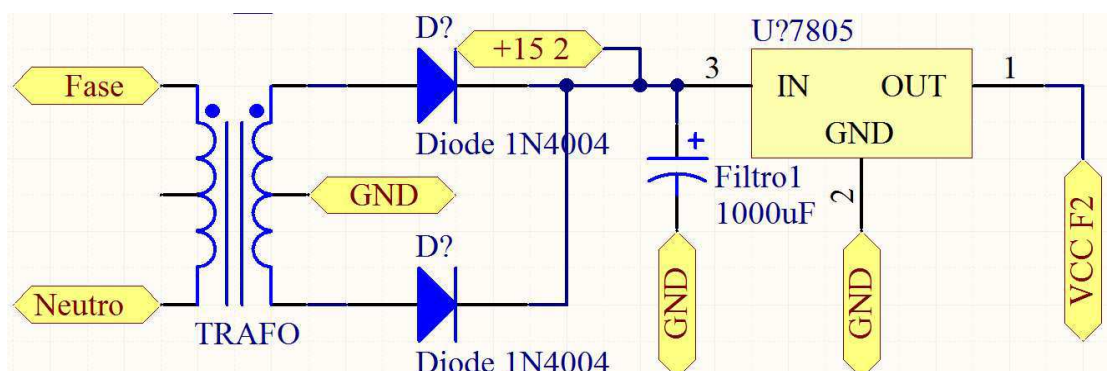


Figura 19: Fonte de alimentação secundária

3.1.3 Sinal de sincronismo com a rede

É de grande importância ter-se um sinal de sincronismo com a rede, pois para o disparo dos SCRs é necessário que seja conhecido o cruzamento da senoide com zero.

Para obtermos o sincronismo com a rede é necessário montar um circuito que detecte o cruzamento com zero da senoide. Para este circuito utilizamos o amplificador operacional TL082 e um transistor de junção bipolar BC557.

Neste circuito, o amplificador operacional compara o valor da senoide com a referência terra do transformador gerando uma onda quadrada. O sinal quadrado excita a base do transistor fazendo com que entre em regime de saturação e corte. O transistor está alimentado com 0V e 5V (Figura 20) de tal forma que a componente negativa vinda do amplificador operacional desaparece, permanecendo apenas um sinal quadrado de 0V e 5V como pode ser visto na (Figura 21)

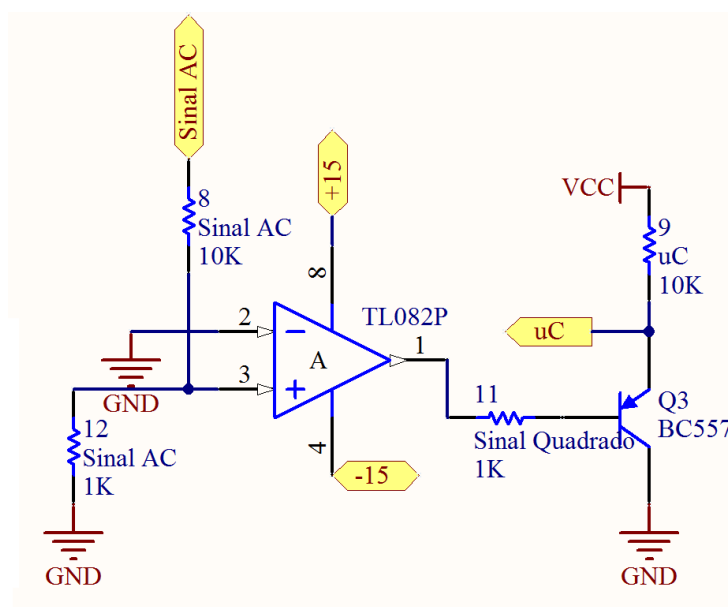


Figura 20: Sinal de referência

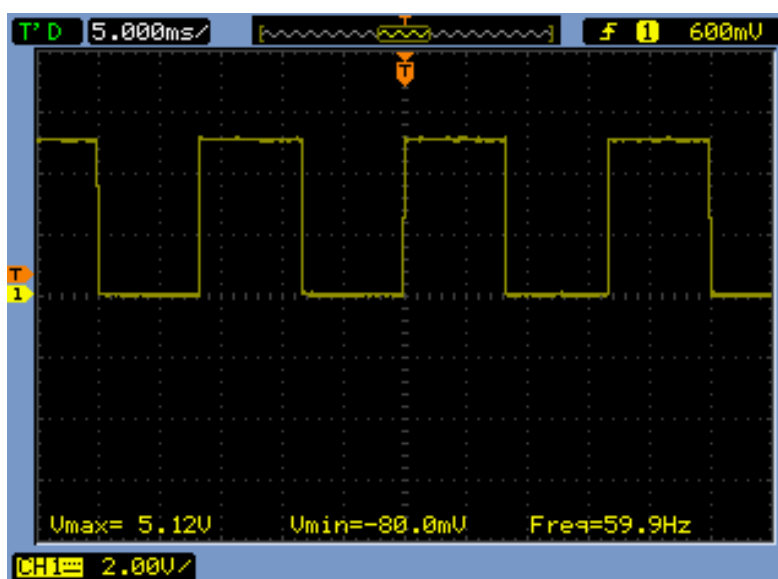


Figura 21: Sinal de sincronismo com a rede

3.1.4 Circuitos de disparo

Para o disparo dos SCRs e dos IGBTs, existem diversos tipos de circuitos. Os circuitos utilizados para o disparo foram isolados com optoacopladores e drivers.

Os circuitos foram montados compartilhando o optoacoplador, de forma que para mudar de conversor CA/CC para CC/CC ou CC/CA é preciso apenas modificação de alguns *jumpers* na placa.

3.1.4.1 Disparo do SCR

A partir do sinal de sincronismo com a rede o microcontrolador consegue disparar entre ângulos de 0 e 180°. Quando o ângulo desejado de disparo é equivalente ao da rede o microcontrolador gera um pulso de 60µS no *gate* do SCR fazendo com que ele passe para o estado de condução.

Para disparar os tiristores da ponte completa monofásica são necessárias quatro fontes de tensão. Para contornar o problema de quatro fontes de tensão foi testado o terra virtual (Figura 22) que funcionou satisfatoriamente, porém ainda foi necessária uma fonte de tensão auxiliar para disparar a ponte completa.

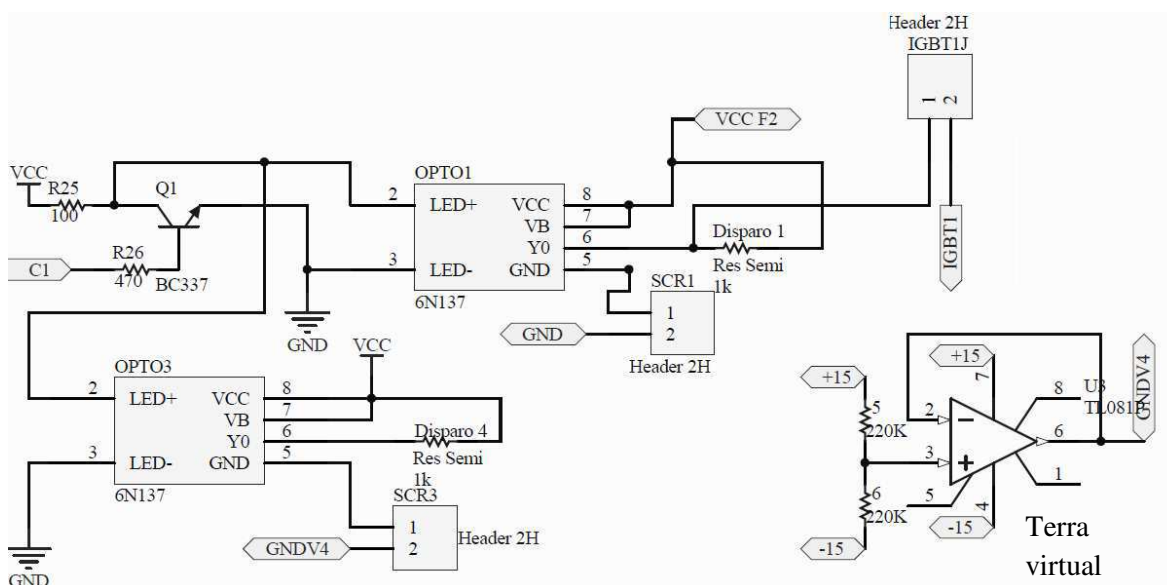


Figura 22: Circuito disparo SCR

3.1.4.2 Disparo do IGBT

Para o disparo dos IGBTs foi utilizado o driver IR2108 (Figura 23) que permite disparar o braço de um inversor. Este driver garante que o IGBT entre em saturação e pode trabalhar em frequências maiores que 20kHz podendo chegar até 200kHz. O circuito desenvolvido seguiu o circuito recomendado pelo “datasheet” do componente. Para o disparo do IGBT temos um optoacoplador antes do driver, este optoacoplador é quem envia o sinal PWM para o driver. Foi necessário que o optoacoplador fosse ultra rápido, por isso utilizou-se o 6N137 que tem a capacidade de trabalhar com altas frequências. Este tem tempos de subida e descida muito reduzidos não distorcendo o sinal PWM.

Os sinais de disparo do IGBT vêm do microcontrolador que contém um bloco para geração do sinal PWM. Estes sinais podem ser modificados tanto em frequência como em ciclo de trabalho possibilitando uma maior flexibilidade para a placa didática.

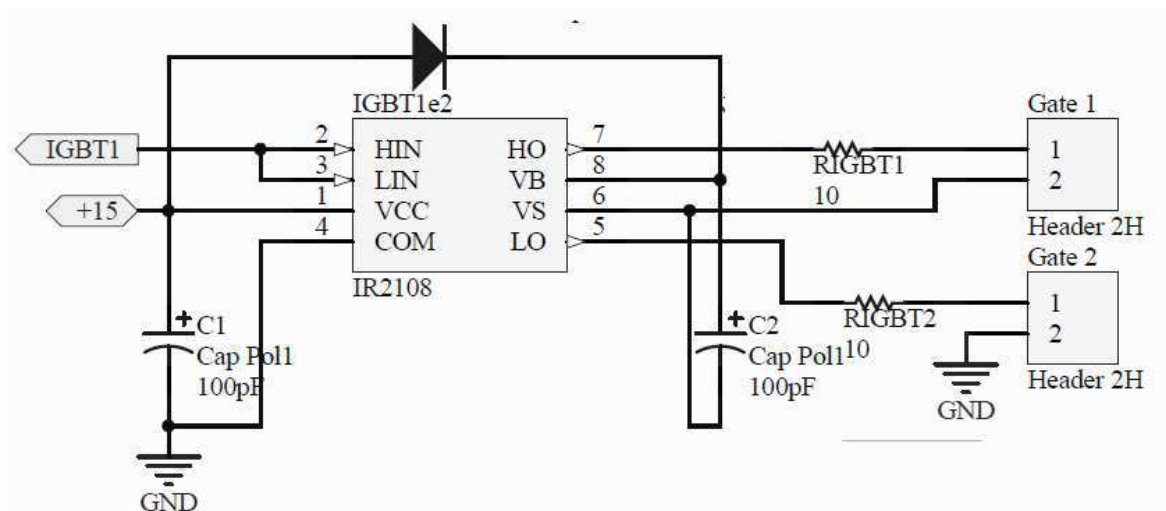


Figura 23: Circuito de disparo do IGBT

3.1.5 Sinais gerados pelo microcontrolador

Além dos sinais PWM e os pulsos de disparo para os SCRs o microcontrolador mais dois sinais de referência, sinal dente de serra e cosenoidal, estes são apenas sinais didáticos para o aluno verificar o funcionamento dos disparos dos SCRs e se estão em fase com a rede.

Para gerar os sinais dente de serra e rampa foi necessário discretizar estes sinais. Para discretizar o sinal dente de serra apenas, fez-se que a variável sempre comece em zero e iríamos incrementando a cada $33\mu\text{S}$, até o próximo evento de sincronismo (3.1.3 Sinal de sincronismo com a rede) onde as saídas do conversor D/A vão para zero (Figura 24). Para o sinal cosenoidal foi necessário tabelar os valores em um *array* de 256 posições e a cada $33\mu\text{S}$ acessávamos este *array* e atualizávamos a saída até o evento de sincronismo onde o sinal ia para zero (Figura 25). Para gerar o array foi utilizado o matlab gerando todos os dados do *array* bastando adicionar os dados ao código fonte.

Código matlab utilizado:

```
n = 0:255;
tensao = 2.5;
saida = -tensao*cos(n*pi/255)+tensao;
plot(saida);
```

Para gerar estes sinais utilizou-se o conversor digital analógico. O conversor utilizado foi o R-2R ilustrado na (Figura 26).

Neste conversor D/A, a saída dos pinos do microcontrolador passa por divisores de tensão. Dessa maneira, para cada *bit* estará associado um ganho de tensão e a saída analógica será a soma da tensão de cada nó.

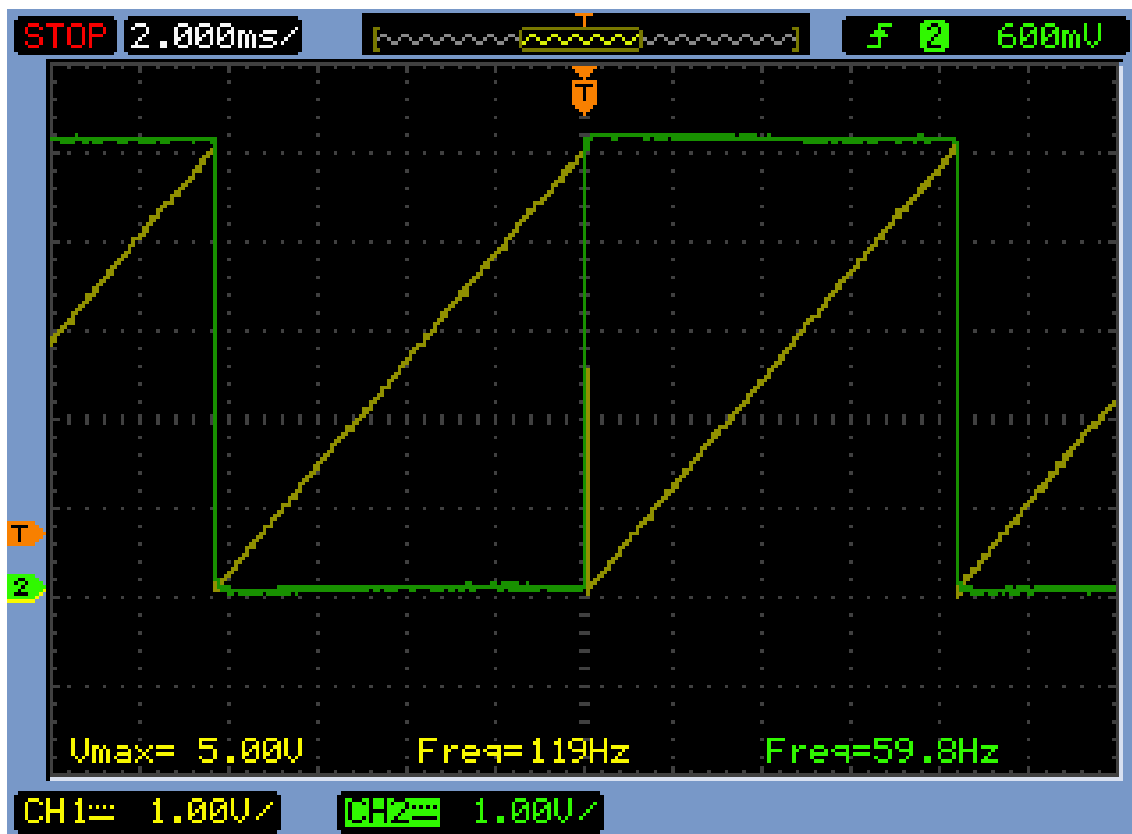


Figura 24:sinal dente de serra

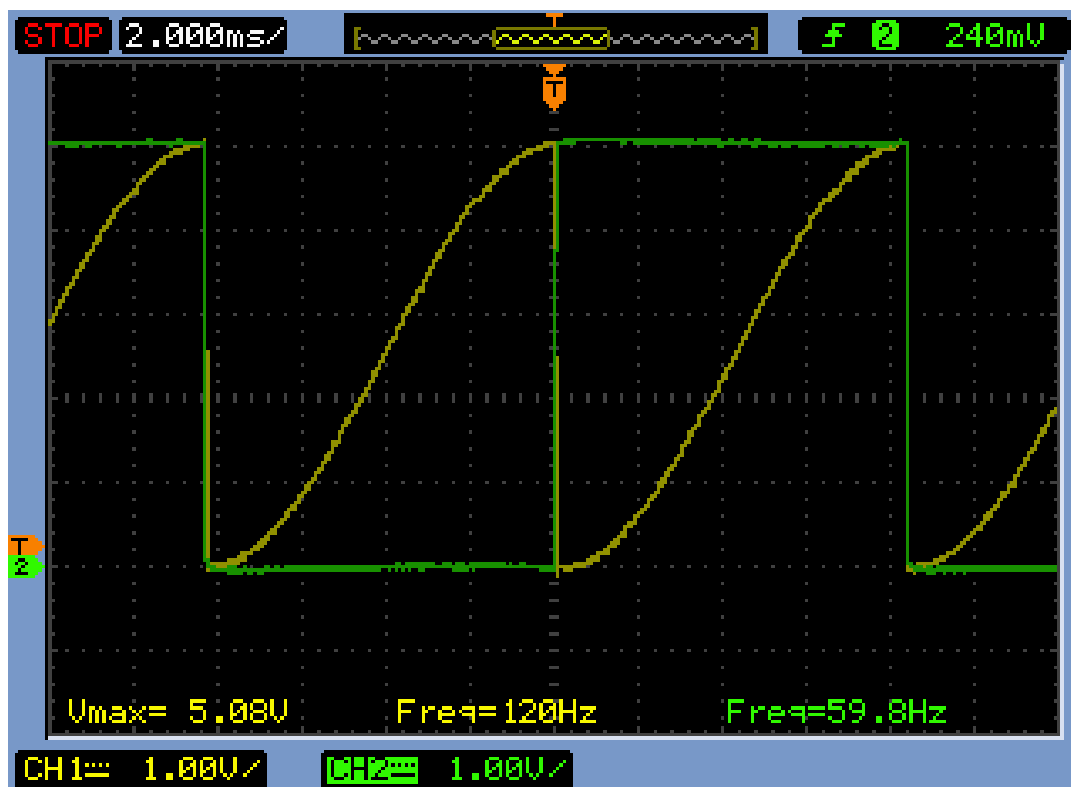


Figura 25: Sinal Cosenoidal

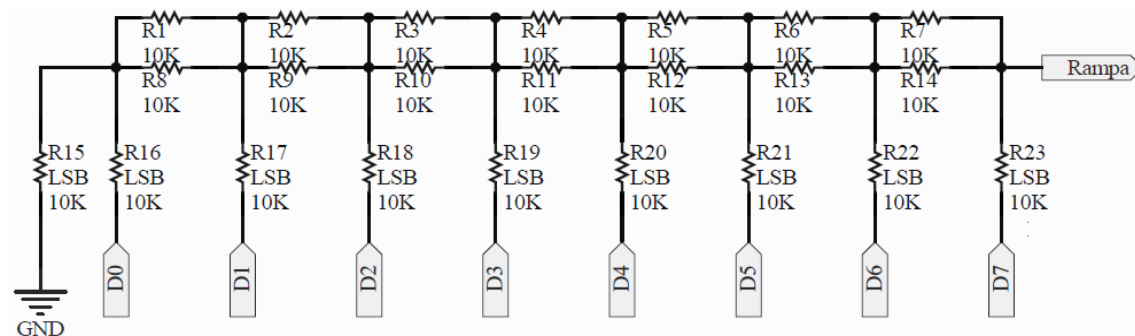


Figura 26: Conversor D/A R-2R

4. Desenvolvimento *layout* e confecção da PCI

Inicialmente, montou-se o circuito em uma matriz de contato. Os circuitos foram montados na matriz de contato para verificar o seu funcionamento, possíveis problemas e melhor dimensionamento dos componentes, sem ser necessário dessoldar os componentes da placa eletrônica.

Após a montagem dos circuitos eletrônicos no *protoboard*, foi utilizado o *software*, **Altium Designer** para realizar o desenho do esquemático (Figura 28). Após o término do esquemático iniciou-se o desenvolvimento do *layout* (Figura 29), quando pronto foram gerados os arquivos gerber e a placa foi confeccionada na fresadora LPKF (Figura 27) do laboratório E-Robótica da UFCG. Uma vez que a placa de circuito impresso estava pronta, os componentes foram soldados e todo sistema testado em conjunto com as placas de potência.



Figura 27:Fresadora LPKF

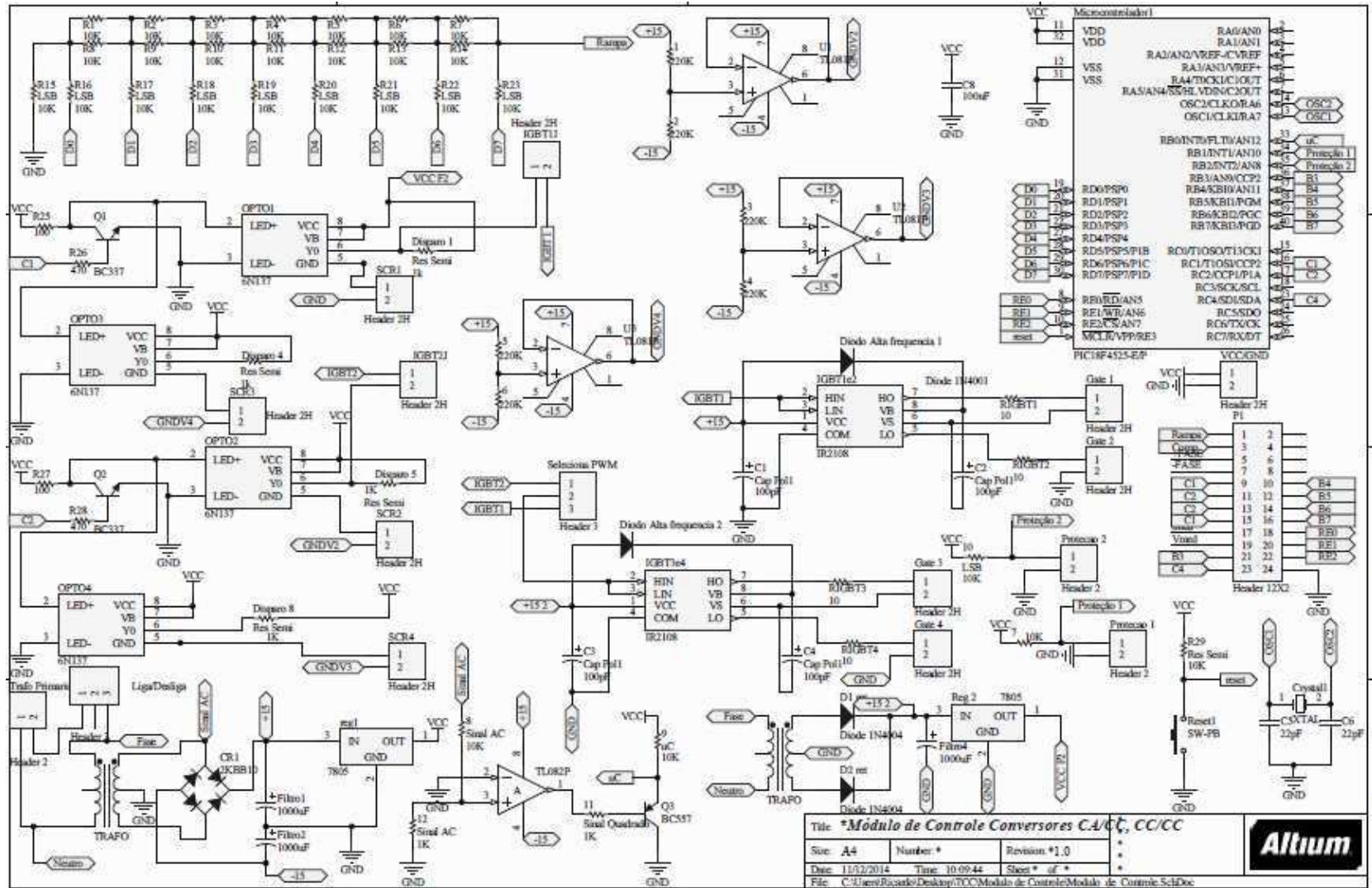


Figura 28:Esquemático circuito

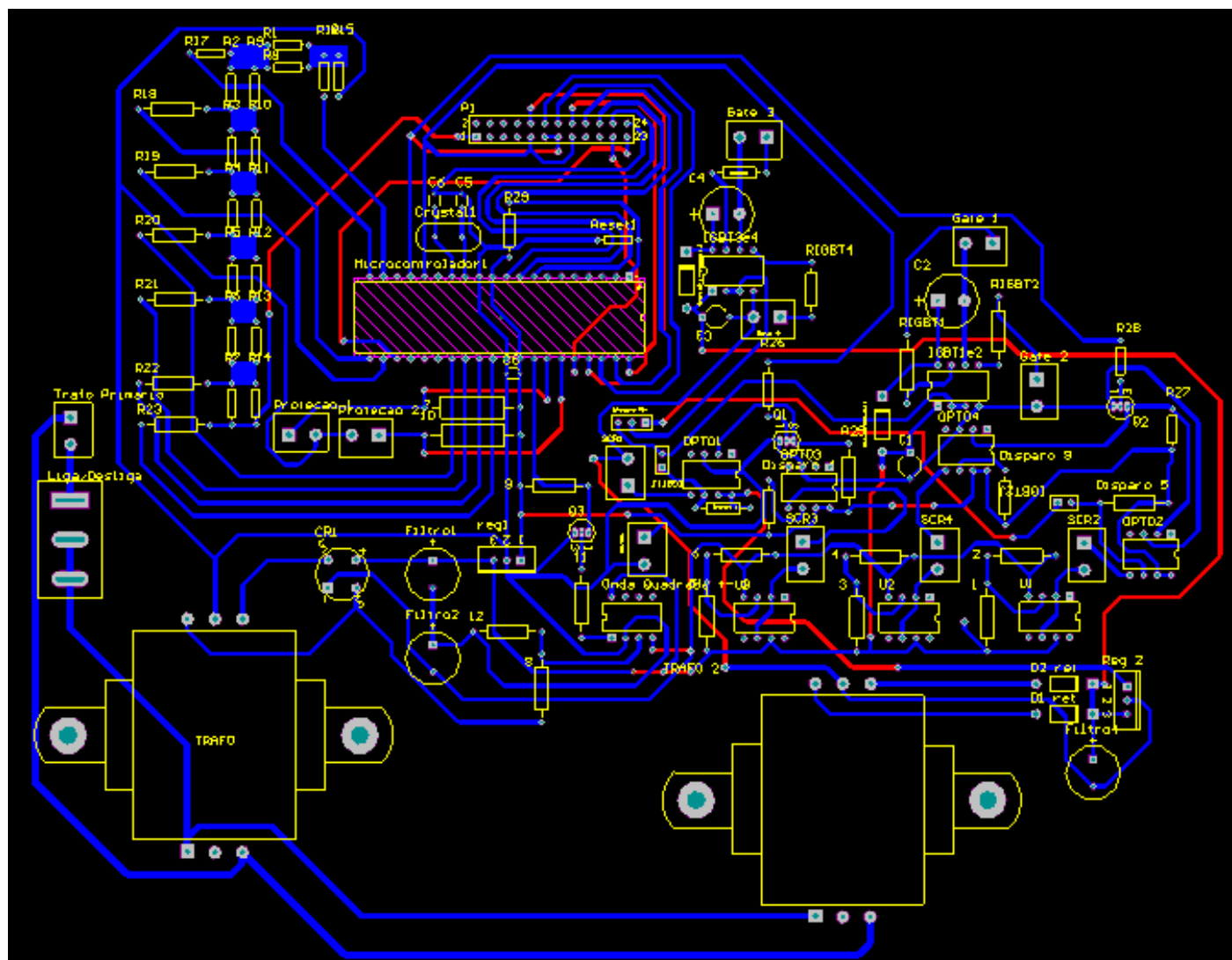


Figura 29:layout PCI

5. Funcionamento do sistema

Para o funcionamento do sistema é necessário fazer as devidas ligações da placa de controle com o módulo de potência e a IHM. A IHM foi conectada através de um cabo flat à placa de controle, e o módulo de potência foi ligado à placa de controle utilizando-se *bornes* para conectar os sinais de controle aos IGBTs e SCRs.

5.1 Funcionamento do conversor CA/CC

Inicialmente, liga-se o sistema e é mostrada a tela (Figura 30). Após a tela inicial ser apresentada, temos a opção de seleção de sinal de referência (Figura 31). Isto é feito selecionando a referência através dos botões da IHM, onde pode-se escolher o ângulo de disparo dos tiristores utilizando os botões da IHM (Figura 32 e Figura 33). É possível voltar à tela inicial do sistema apertando a tecla menu quando o sistema encontra-se no estado mostrado na Figura 32.



Figura 30: Tela Inicial CA/CC

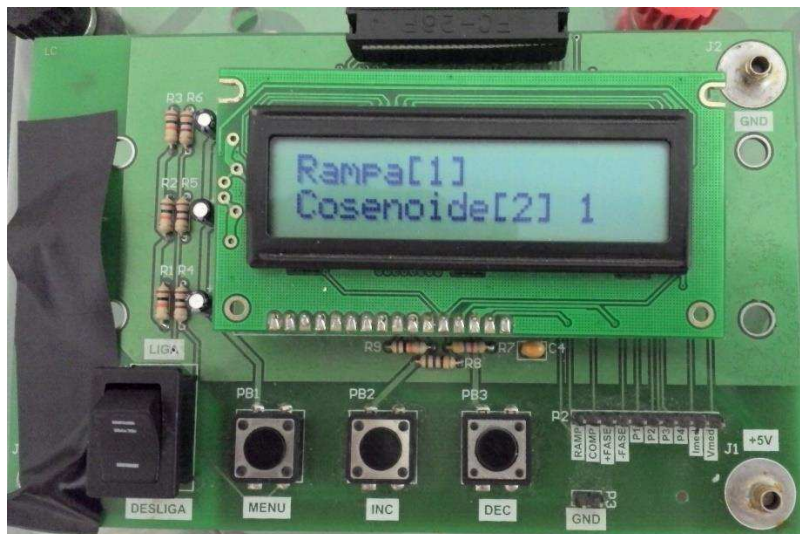


Figura 31:Seleção de referência



Figura 32: Disparo 180°



Figura 33:Disparo 164°

5.2 Funcionamento do conversor CC/CC

Inicialmente, liga-se o sistema e é mostrada a tela (Figura 34). Após a tela inicial ser apresentada, temos a opção da escolha do conversor utilizando os botões da IHM (Figura 35). Ao selecionar o conversor, deve ser escolhida a sua frequência (Figura 36), e então pode-se selecionar entre ciclo de trabalho ou tensão de saída (Figura 37). Caso seja selecionado o ciclo de trabalho, é preciso definir o ciclo (Figura 38). Então, é mostrada a tela final (Figura 39). Para retornar à tela inicial, é necessário pressionar os três botões (*menu*, *inc* e *dec*) ao mesmo tempo.



Figura 34: Tela inicial CC/CC

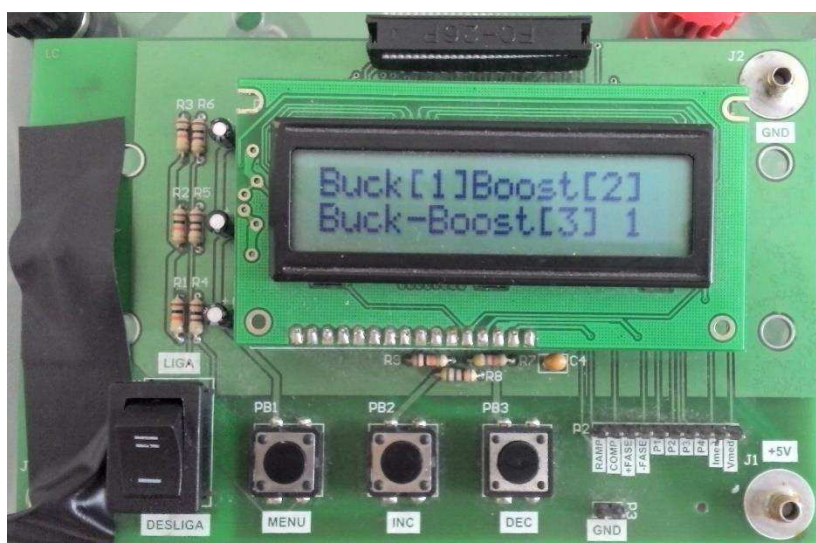


Figura 35: Tipo de conversor



Figura 36: Seleção de frequência



Figura 37: Ciclo de trabalho ou tensão de saída



Figura 38: Escolha ciclo de trabalho

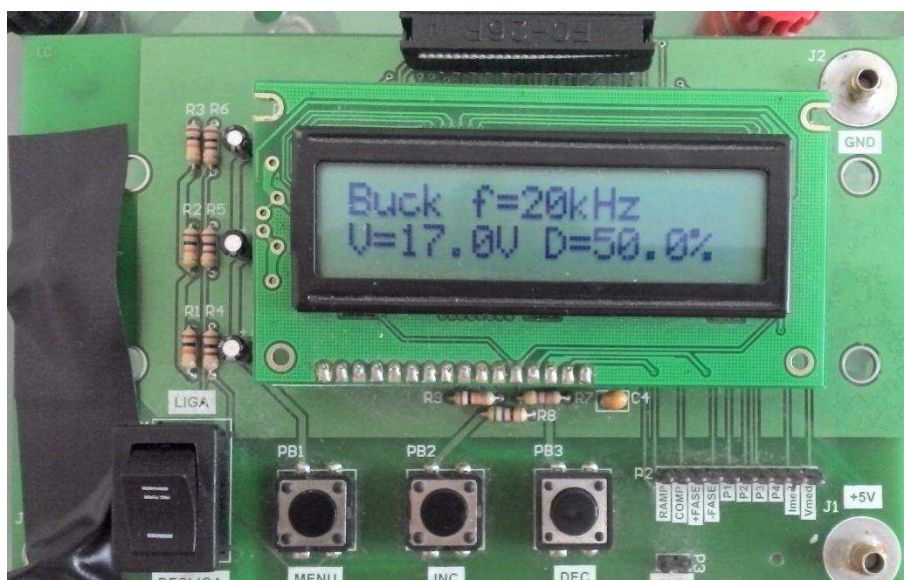


Figura 39: Tela final

6. Fluxograma do *firmware*

Para o desenvolvimento do *firmware* foi utilizado o compilador PIC C pela familiaridade com este, e pela sua facilidade de acesso ao *hardware* do microcontrolador. Os *softwares* desenvolvidos encontram-se no apêndice A e apêndice B. O fluxograma do *software* do conversores CA/CC encontram-se nas (Figura 40), (Figura 41), (Figura 42),

(Figura 43). O fluxograma do *software* do conversor CC/CC encontra-se nas (Figura 44), (Figura 45).

6.1 Fluxograma *firmware* conversor CA/CC

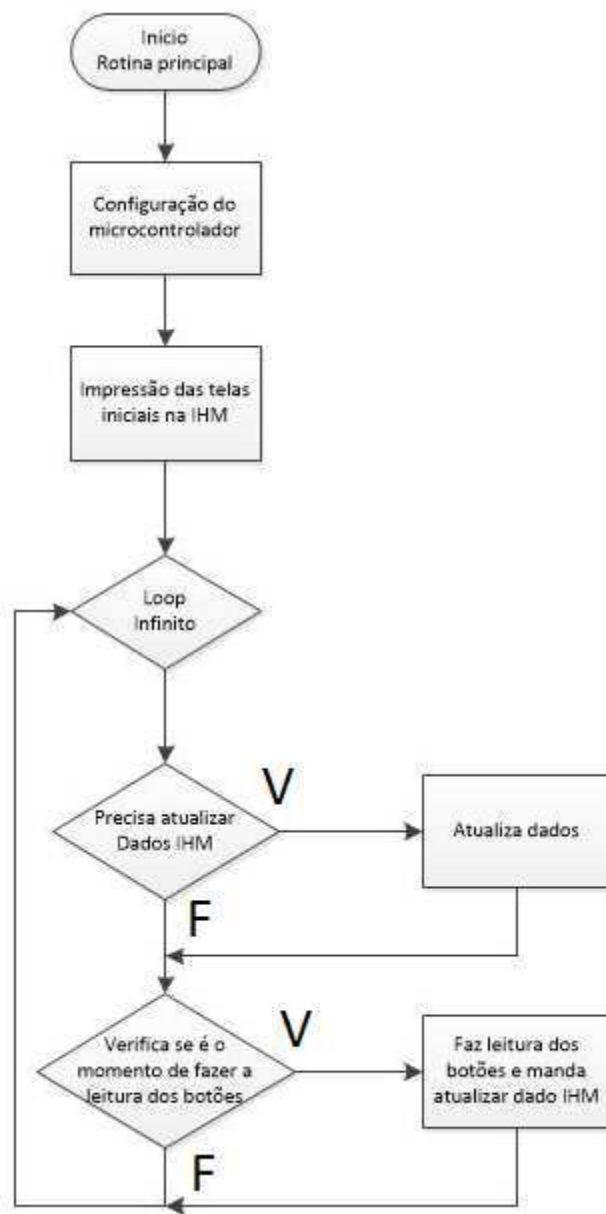


Figura 40: Rotina principal conversor CA/CC

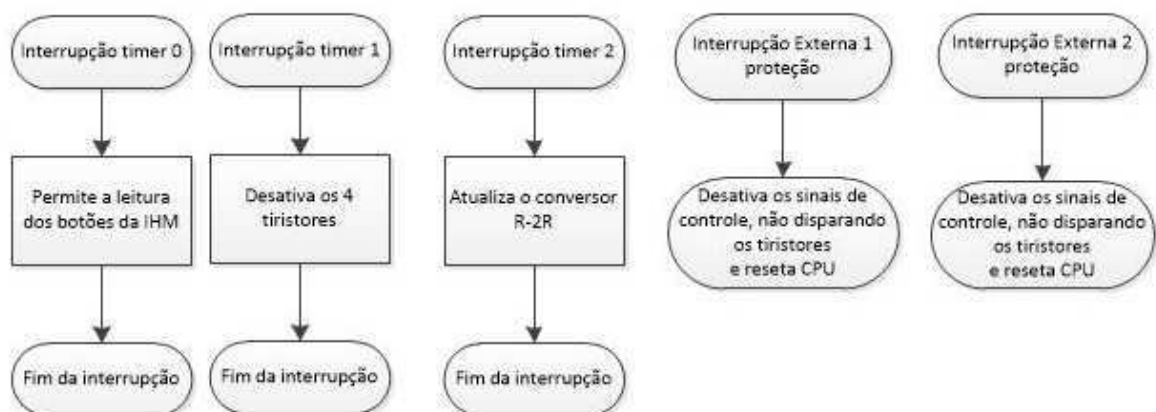


Figura 41:Rotina das interrupções Timer 0, Timer 1, Timer 2, externa 1 e externa 2

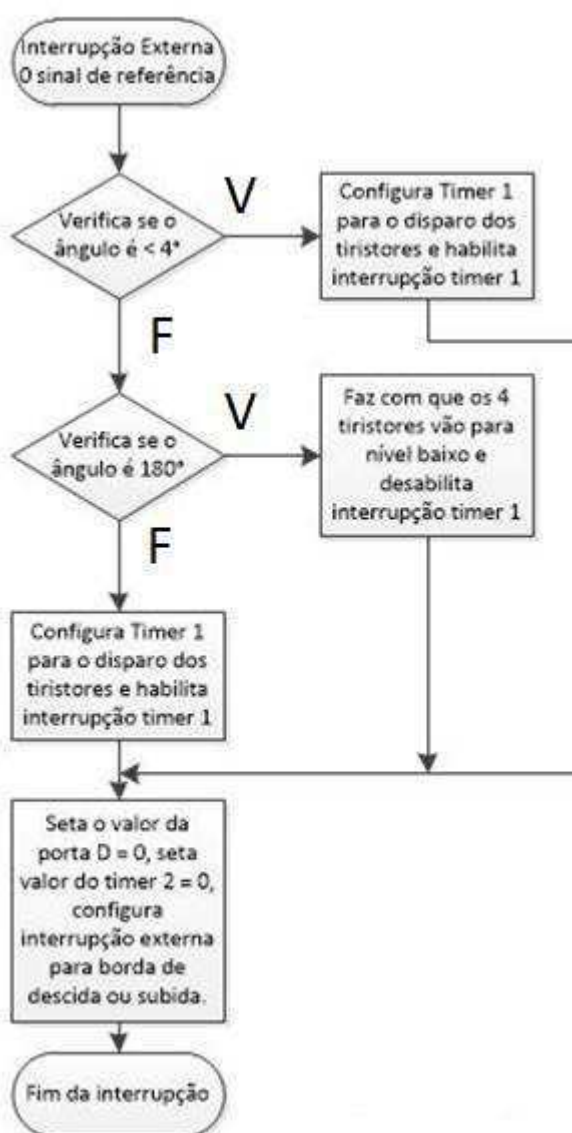


Figura 42:Interrupção externa 0

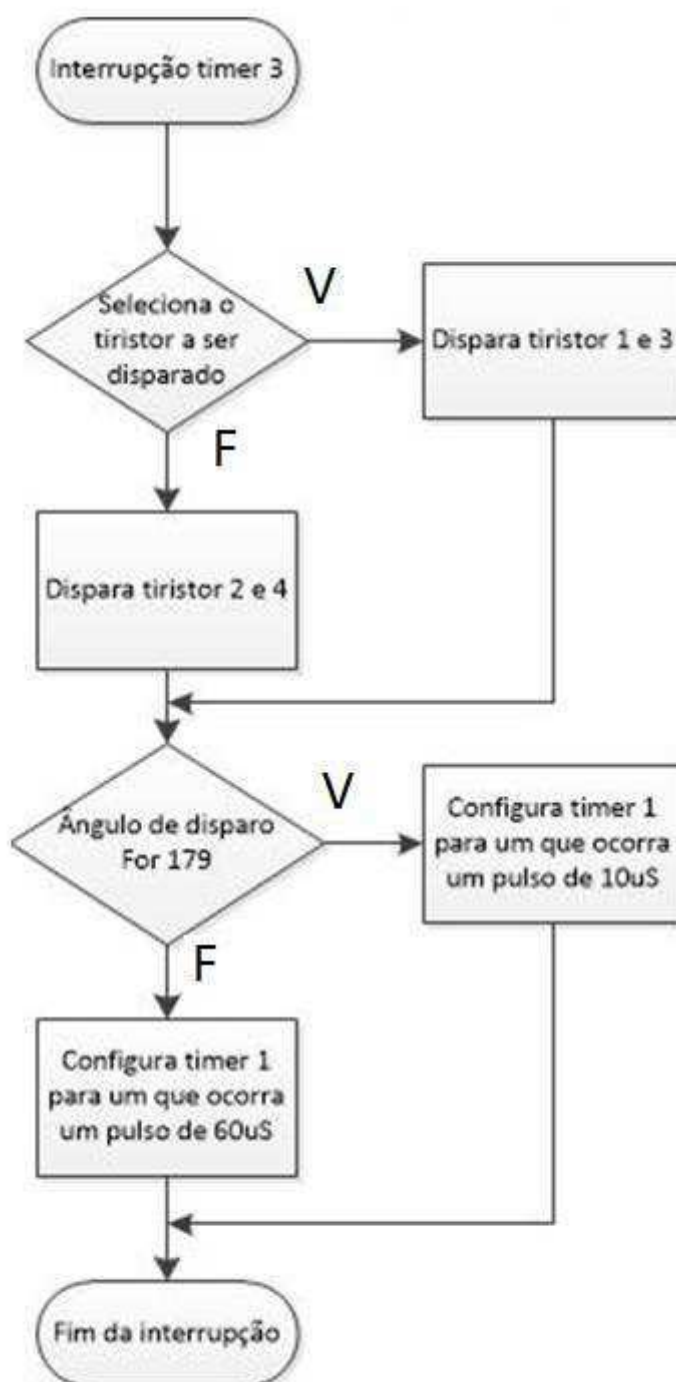


Figura 43: Interrupção timer 3

6.2 Fluxograma *firmware* conversor CC/CC

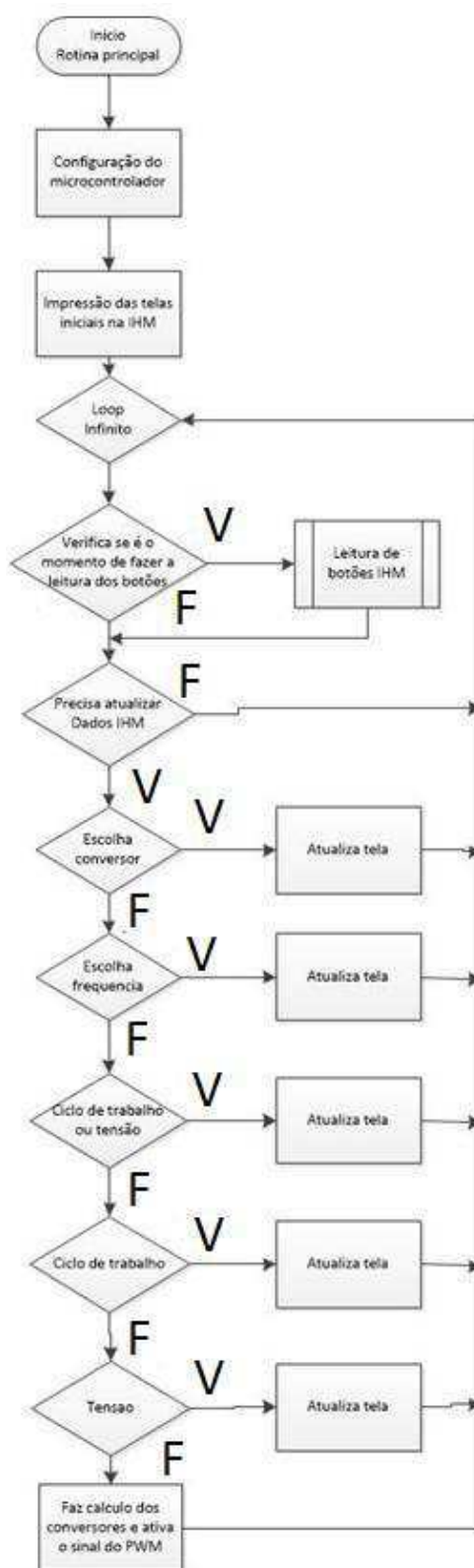


Figura 44:Condigo principal conversor CC/CC

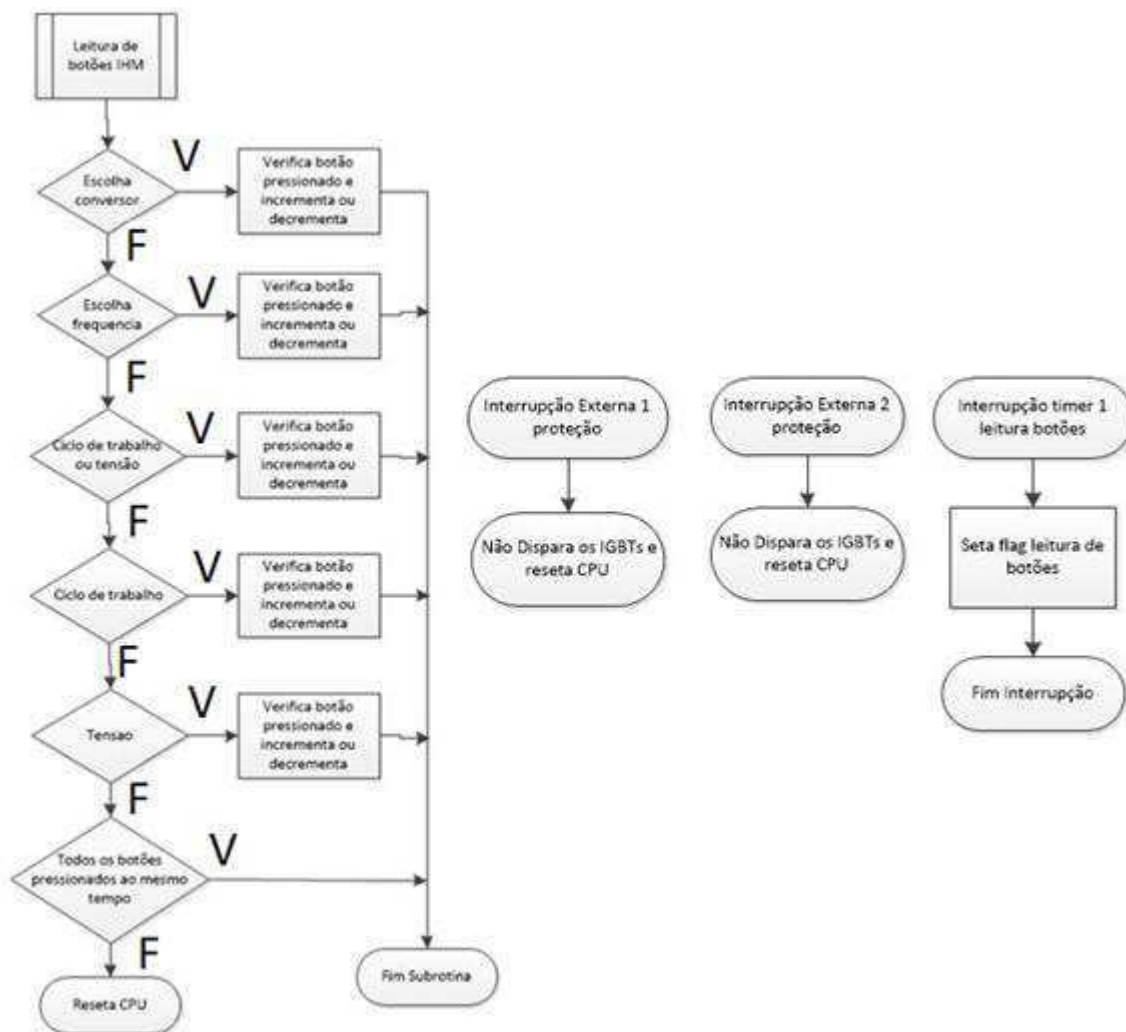


Figura 45: Subrotina e interrupções externa 1, externa 2 e timer 1

7. Análise de resultados

Neste tópico serão analisados os resultados obtidos de experimentos dos conversores CA/CC e CC/CC, utilizando a placa de controle genérica acoplada à placa de potência. A placa de potência do retificador é composta, basicamente, por quatro SCRs, cinco diodos, resistores de potência, indutores e bornes de conexão. Estes bornes permitem a conexão da tensão de entrada e a conexão da carga. Ademais, tratando-se de um módulo didático, os bornes também permitem que os alunos durante os experimentos em laboratório possam conectar os elementos do módulo conforme configuração desejada. Analogamente, a placa de potência do conversor CC/CC é composta por dois IGBTs, dois diodos, dois capacitores, dois indutores e os bornes de conexão. Para a placa CC/CA não foi feito nenhum software desta forma não foi possível realizar nenhum teste com a mesma.

7.1 Conversor CA/CC ponte completa

A estrutura utilizada para testar a placa de controle acoplada à placa de potência é um retificador monofásico controlado de onda completa alimentando um resistor de potência de 50Ω , conforme ilustrada nas Figura 46 e Figura 47.

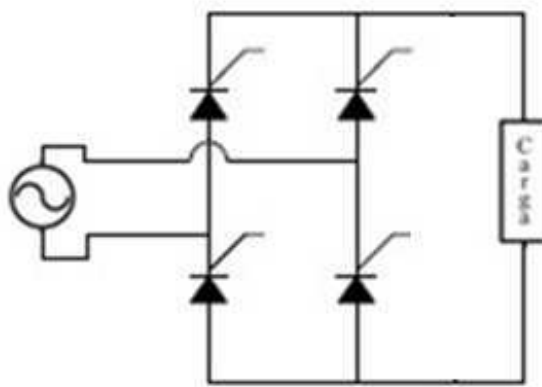


Figura 46: Circuito ponte controlada completa

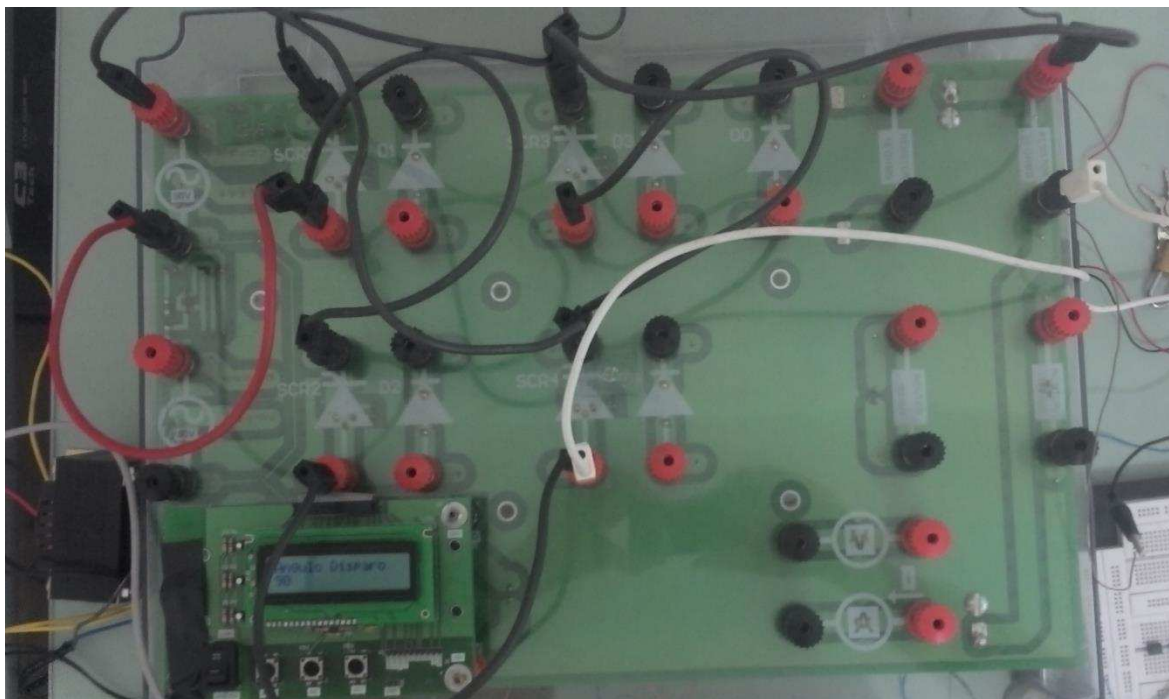


Figura 47: Circuito de teste retificar controlado onda completa

Para a avaliação dos resultados foi feita uma tabela com os valores de tensão média na carga a partir dos resultados obtidos nas (Figura 48) e (Figura 49).

$$V_{avr} = \frac{V_{rms}\sqrt{2}}{\pi} * (1 + \cos(\alpha))$$

Tabela 1: Avaliação dos resultados

Ângulo de disparo	Tensão de saída		Tensão rms média no secundário do transformador	Erro absoluto	Erro relativo
	Tensão de saída média sobre a carga teórica	Tensão de saída média sobre a carga real			
40°	36.73V	34,4V	46.18V	0.33V	6.34%
90°	20.79V	21.2V	46.18V	-0.41V	1.97%

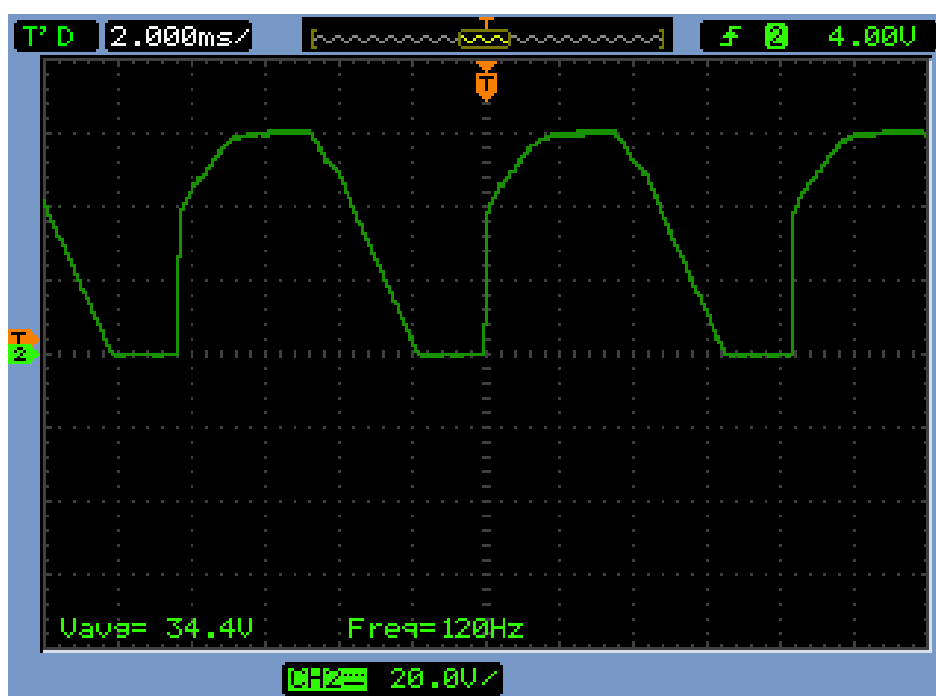


Figura 48: Disparo 40°

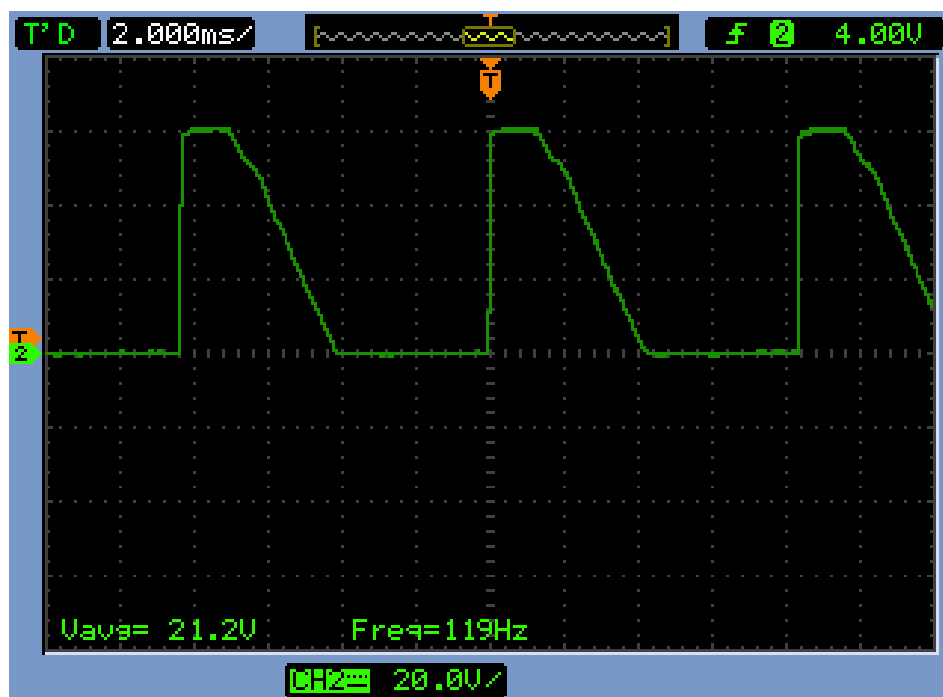


Figura 49:Disparo 90°

Para os sinais rampa e cosenoidal gerados, foi feita apenas a análise gráfica que pode ser vista nas Figura 50 e Figura 51. Considerou-se o resultado gráfico satisfatório com ângulos de disparos bem próximos ao esperado.

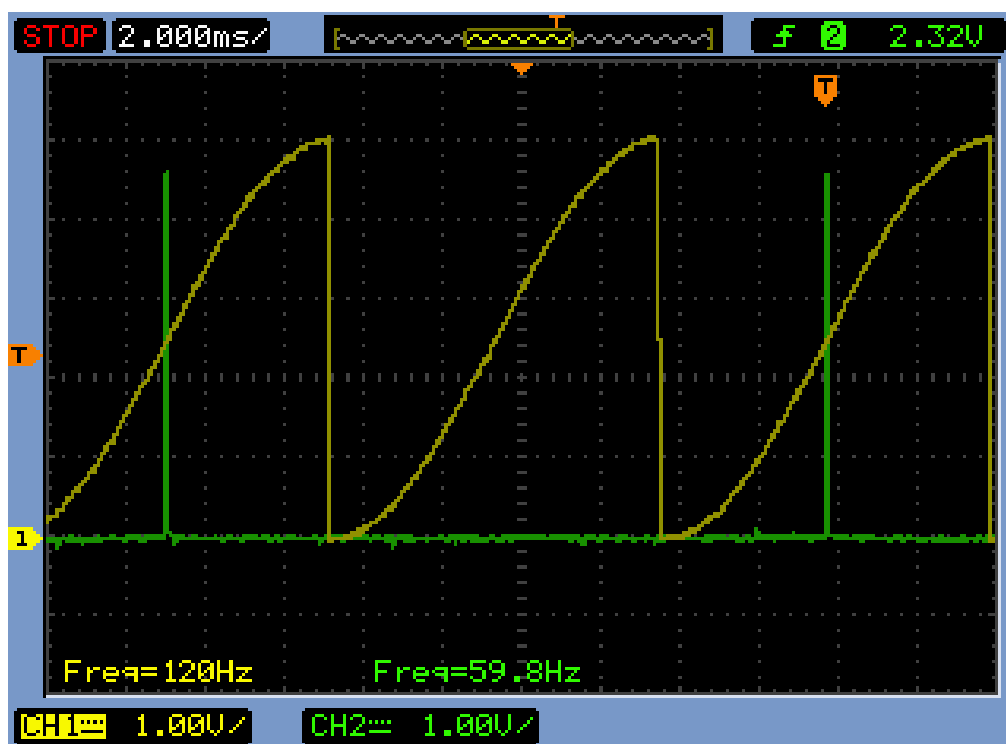


Figura 50:Sinal Cosenoidal disparo 90°

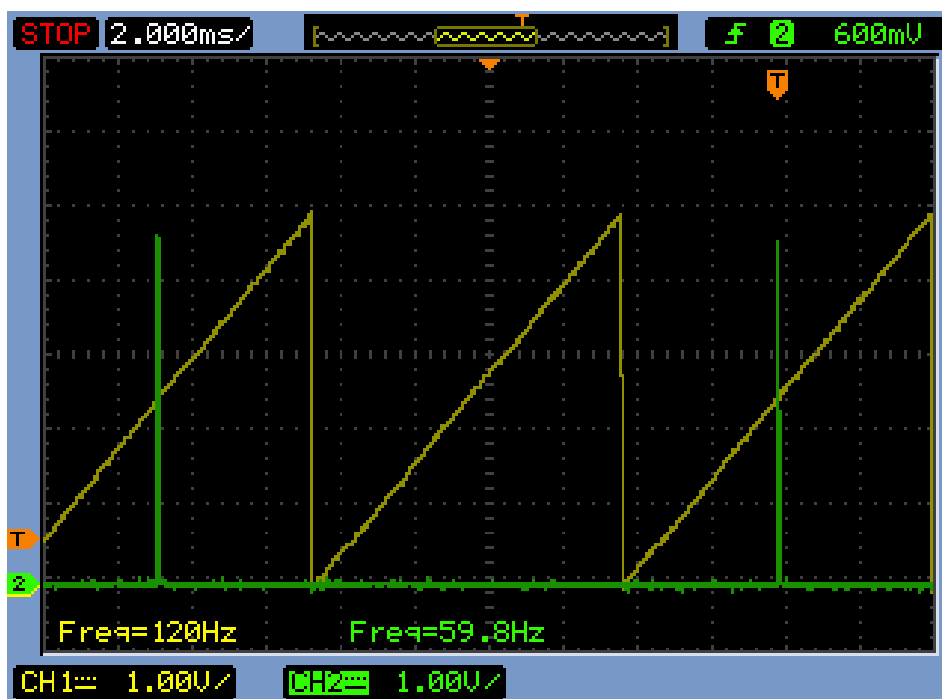


Figura 51: Sinal rampa disparo 90°

8. Possíveis melhorias

As possíveis melhorias para o trabalho realizado:

- Desenvolvimento do código fonte para o funcionamento do conversor CC/CA;
- Atualização do *software* dos conversores CC/CC para suporte do conversor CUK;

9. Conclusão

A experiência proporcionada pelo trabalho de conclusão de curso permitiu que o aluno colocasse em prática o conhecimento de diversas disciplinas do curso. As disciplinas de maior importância para o desenvolvimento deste projeto foram: Circuitos Elétricos 1, Circuitos Elétricos 2, Dispositivos Eletrônicos, Eletrônica, Eletrônica de Potência, Arquitetura de Sistemas Digitais.

A partir dos resultados obtidos no experimento, verificou-se que a placa de controle e as placas de potência dos conversores CA/CC tiveram o funcionamento validado.

Conclui-se com este trabalho, a implementação de uma unidade de controle digital para os conversores CA/CC, CC/CC e CC/CA com um conjunto de recursos, tais como a geração e controle do ângulo de disparo dos tiristores, a geração de pulsos PWM para acionamento dos IGBTs, faltando apenas o software para o conversor CC/CA e a realização de testes das topologias dos conversores CC/CC.

10. Bibliografia

Italo, R. F. M. P.S, Ensaios de conversores CA/CC e CC/CA acoplados a uma placa de controle genérica para os módulos do Laboratório de Eletrônica de Potência. Universidade Federal de Campina Grande. Campina Grande. 2012.

James W. Nilsson, Circuitos Elétricos, 8° edição, Pearson Prentice Hall

Robert L. Boylestad, Dispositivos Eletrônicos e teoria de circuitos, 8° edição, Pearson Prentice Hall

Charles B. Lima, Técnicas de projetos eletrônicos com os microcontroladores AVR, Clube de autores

Daniel W. Hart, Eletrônica de Potência, AMGH editora LTDA.

Fábio Pereira, Microcontroladores PIC Programação em C, 6 edição, Editora Erica

MICROSHIP TECHNOLOGY INC. **PIC18F4525 Datasheet**. Eletronic Publication. 2009.

INTERNATION RECTIFIER. **Datasheet: Current Limiting Single Channel Driver**. Eletronic Publication. 2003.

APÊNDICE A

```

#include <18F4525.h>

#device adc=8

#FUSES NOWDT           //No Watch Dog Timer
#FUSES WDT128         //Watch Dog Timer uses 1:128 Postscale
#FUSES H4             //High speed Osc (> 4mhz) 4PLL
#FUSES NOPROTECT      //Code not protected from reading
#FUSES NOIESO         //Internal External Switch Over mode disabled
#FUSES BROWNOUT       //Reset when brownout detected
#FUSES BORV45         //Brownout reset at 4.3V
#FUSES PUT            //Power Up Timer
#FUSES NOCPD          //No EE protection
#FUSES STVREN         //Stack full/underflow will cause reset
#FUSES NODEBUG        //No Debug mode for ICD
#FUSES NOLVP          //No low voltage prgming, B3(PIC16) or B5(PIC18) used for I/O
#FUSES NOWRT          //Program memory not write protected
#FUSES NOWRTD         //Data EEPROM not write protected
#FUSES NOEBTR         //Memory not protected from table reads
#FUSES NOCPB          //No Boot Block code protection
#FUSES NOEBTRB        //Boot block not protected from table reads
#FUSES NOWRTC         //configuration not registers write protected
#FUSES NOWRTB         //Boot block not write protected
#FUSES FCMEN          //Fail-safe clock monitor enable
#FUSES RESERVED       //Used to set the reserved FUSE bits
#FUSES NOPBADEN       //PORTB pins are configured as digital I/O on RESET
#FUSES LPT1OSC        // configured for low-power operation
#FUSES MCLR           //Master Clear pin enabled

#use delay(clock=4000000)

```

```
#ZERO_RAM
```

```
#PRIORITY int_TIMER2,int_TIMER3, int_EXT
```

```
#use fixed_io (c_outputs=PIN_C1,PIN_C2,PIN_C4)
```

```
#use fast_io (C)
```

```
#include <LCD_16x2_lib.c>
```

```
#define incrementa PIN_E0
```

```
#define decrementa PIN_E2
```

```
#define confirma_tela PIN_E1
```

```
#define tiristor_1_4 PIN_C1
```

```
#define tiristor_2_3 PIN_C2
```

```
const float onda_cosenoidal[256] = {0,0.0097,0.0387,0.0871,0.1548,0.2418,0.3482,0.4738,0.6188,0.7830
```

```
,0.9664,1.1690,1.3908,1.6318,1.8918,2.1709,2.4691,2.7862,3.1222,3.4771,3.8509,4.2434,4.6546,5.0845,5.5
```

```
329,6,  
6.4853,6.9891,7.5111,8.0514,8.6098,9.1862,9.7806,10.3929,11.0230,11.6707,12.3360,13.0188,13.7189,
```

```
14.4364,  
15.1709,15.9226,16.6912,17.4766,18.2786,19.0973,19.9324,20.7839,21.6515,22.5352,23.4349,24.3503,25.2
```

```
814,  
26.2280,27.1900,28.1672,29.1595,30.1667,31.1887,32.2253,33.2764,34.3417,35.4212,36.5147,37.6220,38.7
```

```
429,  
39.8773,41.0250,42.1858,43.3596,44.5462,45.7453,46.9568,48.1806,49.4164,50.6641,51.9234,53.1942,54.4
```

```
763,  
55.7694,57.0735,58.3882,59.7134,61.0489,62.3945,63.7500,65.1152,66.4898,67.8737,69.2666,70.6684,72.0
```

```
788,  
73.4976,74.9246,76.3596,77.8023,79.2526,80.7102,82.1749,83.6465,85.1248,86.6094,88.1003,89.5972,91.0
```

```
998  
92.6080,94.1214,95.6399,97.1633,98.6912,100.2235,101.7600,103.3004,104.8444,106.3919,107.9426,109.4
```

```
962  
,111.0526,112.6115,114.1726,115.7358,117.3007,118.8672,120.4350,122.0039,123.5736,125.1439,126.714
```

```
6,128.2854  
,129.8561,131.4264,132.9961,134.5650,136.1328,137.6993,139.2642,140.8274,142.3885,143.9474,145.503
```

```
8,147.0574
```

,148.6081,150.1556,151.6996,153.2400,154.7765,156.3088,157.8367,159.3601,160.8786,162.3920,163.9002,165.4028

,166.8997,168.3906,169.8752,171.3535,172.8251,174.2898,175.7474,177.1977,178.6404,180.0754,181.5024,182.9212,

184.3316,185.7334,187.1263,188.5102,189.8848,191.2500,192.6055,193.9511,195.2866,196.6118,197.9265,199.2306,

200.5237,201.8058,203.0766,204.3359,205.5836,206.8194,208.0432,209.2547,210.4538,211.6404,212.8142,213.9750,

215.1227,216.2571,217.3780,218.4853,219.5788,220.6583,221.7236,222.7747,223.8113,224.8333,225.8405,226.8328,

227.8100,228.7720,229.7186,230.6497,231.5651,232.4648,233.3485,234.2161,235.0676,235.9027,236.7214,237.5234

,238.3088,239.0774,239.8291,240.5636,241.2811,241.9812,242.6640,243.3293,243.9770,244.6071,245.2194,245.8138

,246.3902,246.9486,247.4889,248.0109,248.5147,249.0001,249.4671,249.9155,250.3454,250.7566,251.1491,251.5229

,251.8778,252.2138,252.5309,252.8291,253.1082,253.3682,253.6092,253.8310,254.0336,254.2170,254.3812,254.5262

,254.6518,254.7582,254.8452,254.9129,254.9613,254.9903,255.0000};

```
long onda_triangular_cosenoidal = 0,angulo = 180;
```

```
short seleciona_tiristor = 0;
```

```
short leitura_botoes = 0, atualiza_tela = 0;
```

```
int telas = 0;
```

```
int cosenoide_rampa = 1;
```

```
int32 tempo_disparo = 0;
```

```
#int_TIMER1
```

```
void le_botoes(void)
```

```
{
```

```
    output_low(tiristor_1_4);
```

```
    output_low(tiristor_2_3);
```

```
}
```

```
#int_TIMER0
```

```

void desliga_tiristor(void)
{

    leitura_botoes = 1;
}

/*****
**** Interrupção responsável pela disparo ****
**** dos tiristores ****
*****/

#include <TIMER3>
void Disparo_tiristores(void)
{

if(seleciona_tiristor == 0)
    output_high(tiristor_1_4);
else
    output_high(tiristor_2_3);

if(angulo == 179)
    set_timer1(65435);//10us
else
    set_timer1(65135);//40us
enable_interrupts(INT_TIMER1);

}

/*****
**** Interrupção responsável pela geração ****
*****/

```

```

***** do sinais de referencia: Sinal Rampa, *****)
***** sinal cossenoidal *****)
*****)

#int_TIMER2

void sinal_rampa_ou_cossenoidal(void)
{

onda_triangular_cossenoidal++;
if(onda_triangular_cossenoidal == 257)
    onda_triangular_cossenoidal = 0;

if(cosenoide_rampa == 1)//Rampa
    output_d(onda_triangular_cossenoidal);
else
    output_d((long)onda_cossenoidal[onda_triangular_cossenoidal]);

}

#int_EXT1

void EXT1_isr(void)
{

    output_low(tiristor_1_4);
    output_low(tiristor_2_3);
    reset_cpu();

}

#int_EXT2

void EXT2_isr(void)
{

    output_low(tiristor_1_4);

```

```

output_low(tiristor_2_3);

reset_cpu();
}

/*****
**** Interrupção responsavel por receber ****
**** referencia gerada pela rede eletrica ****
*****/

#int_EXT

void EXT_isr(void)
{

if(angulo < 4){
    set_timer3(64610);
    enable_interrupts(INT_TIMER3);
}

else if(angulo == 180){
    output_low(tiristor_1_4);
    output_low(tiristor_2_3);
    disable_interrupts(INT_TIMER3);
}

else{
    set_timer3((unsigned int)16)tempo_disparo);
    enable_interrupts(INT_TIMER3);
}

output_d(0);
set_timer2(0);
onda_triangular_cosenoidal = 0;
seleciona_tiristor !=seleciona_tiristor;

```



```
if(seleciona_tiristor == 0)
    ext_int_edge(0,L_TO_H);
else
    ext_int_edge(0,H_TO_L);

}

void main()
{
    set_Trис_C (0x00); // Configura porta C como saida

    setup_timer_0(RTCC_INTERNAL|RTCC_DIV_16);
    setup_timer_1(T1_INTERNAL|T1_DIV_BY_1);
    setup_timer_3(T3_INTERNAL|T3_DIV_BY_2);
    setup_timer_2(T2_DIV_BY_1,65,5);
    setup_comparator(NC_NC_NC_NC);
    setup_vref(FALSE);
    ini_lcd_16x2();
    enable_interrupts(INT_TIMER0);
    enable_interrupts(INT_TIMER1);
    //enable_interrupts(INT_TIMER3);
    enable_interrupts(INT_EXT1);
    enable_interrupts(INT_EXT2);
    ext_int_edge(0,L_TO_H);
    ext_int_edge(1,L_TO_H);
    ext_int_edge(2,L_TO_H);
    ini_lcd_16x2();
    output_low(tiristor_2_3);
    output_low(tiristor_1_4);
```

```
output_d(0);
set_timer2(0);
printf(exibe_lcd,"\fLab Eletronica\nde Potencia");
delay_ms(1000);
printf(exibe_lcd,"\fConversor\nCA-CC");
delay_ms(1000);
printf(exibe_lcd,"\fSelecione a\nreferencia");
delay_ms(1000);
printf(exibe_lcd,"\fRampa[1]\nCosenoide[2] %d",cosenoide_rampa);
enable_interrupts(GLOBAL);

// TODO: USER CODE!!

cosenoide_rampa = 1;

while(1){

if(atualiza_tela == 1){
    atualiza_tela = 0;
    switch(telas){
        case 0:
            printf(exibe_lcd,"\fRampa[1]\nCosenoide[2] %d",cosenoide_rampa);
            break;
        case 1:
            printf(exibe_lcd,"\fAngulo Disparo\n%d",angulo);
            tempo_disparo = (65535) - (float)(231.1*angulo);
            break;
    }
}

if(leitura_botoes == 1){
```

```
leitura_botoes = 0;
switch(telas){
  case 0:
    if(input(incrementa) == 0){
      cosenoide_rampa++;
      atualiza_tela = 1;
      if(cosenoide_rampa == 3)
        cosenoide_rampa = 2;
    }

    if(input(decrementa) == 0){
      cosenoide_rampa--;
      atualiza_tela = 1;
      if(cosenoide_rampa == 0)
        cosenoide_rampa = 1;
    }

    if(input(confirma_tela) == 0){
      atualiza_tela = 1;
      if(cosenoide_rampa == 1)
        printf(exibe_lcd, "\fReferencia Rampa\nSelecionada");
      if(cosenoide_rampa == 2)
        printf(exibe_lcd, "\fRef Cossenoide\nSelecionada");
      delay_ms(1000);
      set_timer2(0);
      enable_interrupts(INT_EXT);
      enable_interrupts(INT_TIMER2);
      telas = 1;
    }
    break;
  case 1:
    if(input(incrementa) == 0){
```

```
    angulo++;  
    atualiza_tela = 1;  
    if(angulo > 180)  
        angulo = 180;  
    }  
  
    if(input(decrementa) == 0){  
        angulo--;  
        atualiza_tela = 1;  
        if(angulo == -1)  
            angulo = 0;  
    }  
  
    if(input(confirma_tela) == 0){  
        reset_cpu();  
    }  
    break;  
    }  
    }  
    }  
    }
```

APÊNDICE B

```

#include <18F4525.h>
#device adc=8

#FUSES NOWDT           //No Watch Dog Timer
#FUSES WDT128         //Watch Dog Timer uses 1:128 Postscale
#FUSES H4             //High speed Osc (> 4mhz)
#FUSES NOPROTECT      //Code not protected from reading
#FUSES NOIESO         //Internal External Switch Over mode disabled
#FUSES BROWNOUT       //Reset when brownout detected
#FUSES BORV45         //Brownout reset at 4.5V
#FUSES PUT            //Power Up Timer
#FUSES NOCPD          //No EE protection
#FUSES NOSTVREN       //Stack full/underflow will not cause reset
#FUSES NODEBUG        //No Debug mode for ICD
#FUSES NOLVP          //No low voltage prgming, B3(PIC16) or B5(PIC18) used for
I/O
#FUSES NOWRT          //Program memory not write protected
#FUSES NOWRTD         //Data EEPROM not write protected
#FUSES NOEBTR         //Memory not protected from table reads
#FUSES NOCPB          //No Boot Block code protection
#FUSES NOEBTRB        //Boot block not protected from table reads
#FUSES NOWRTC         //configuration not registers write protected
#FUSES NOWRTB         //Boot block not write protected
#FUSES NOFCMEN        //Fail-safe clock monitor disabled
#FUSES RESERVED      //Used to set the reserved FUSE bits
#FUSES NOPBADEN       //PORTB pins are configured as digital I/O on RESET

```

```
#FUSES LPT1OSC          //Timer1 configured for low-power operation
#FUSES MCLR             //Master Clear pin enabled

#use delay(clock=4000000)

#include <LCD_16x2_lib.c>

#ZERO_RAM

#define tensao_de_entrada 34 //A tensão de entrada tem um valor fixo de 34V
#define clock_efetivo 10000 // clock / 4
#define Freq_max 100
#define Freq_mim 20
#define clock_efetivo_2 clock_efetivo/4.0

#define incrementa PIN_E0
#define decrementa PIN_E2
#define confirma_tela PIN_E1

void configura_timer2(void);
void calcula_frequencia(void);
void tipo_de_conversor(void);
void frequencia_selecionada(void);
void tipo_de_trab_tensao(int);
void tela_final(void);
void calculo_conversor(void);
void leitura_de_teclas(void);
```

```
short botao_apertado;  
float valor_config_timer2_ponto_flutuante = 0;
```

```
struct dados_telas  
{  
    int telas;  
    int conversor;  
    int tensao_ciclo;  
    long frequencia;  
    float tensao;  
    float ciclo_de_trabalho;  
    int valor_config_timer2;  
    long pwm;  
    float tensao_de_saida;  
  
}tela;
```

```
short leitura_botoes = 0;
```

```
#int_TIMER0  
void leitura(void)  
{  
  
    leitura_botoes = 1;  
  
}
```

```
#int_EXT1
```

```
void protecao1(void)
{

    set_pwm1_duty(0);
    reset_cpu();

}

#ifdef EXT2
void protecao2(void)
{
    set_pwm1_duty(0);
    reset_cpu();
}

void main()
{

    setup_timer_0(T0_INTERNAL/T0_DIV_16);
    setup_timer_1(T1_INTERNAL/T1_DIV_BY_8);
    setup_timer_2(T2_DISABLED,0,1);
    setup_comparator(NC_NC_NC_NC);
    setup_vref(FALSE);
    ext_int_edge(1,L_TO_H);
    ext_int_edge(2,L_TO_H);
    ini_lcd_16x2();
    enable_interrupts(INT_EXT1);
    enable_interrupts(INT_EXT2);
    enable_interrupts(INT_TIMER0);
```



```
enable_interrupts(GLOBAL);

// TODO: USER CODE!!

tela.conversor = 0;
tela.telas = 0;
tela.conversor = 1;
tela.frequencia = 20;
tela.tensao_ciclo = 1;

printf(exibe_lcd, "\fLab Eletronica\nde Potencia");
delay_ms(1000);
printf(exibe_lcd, "\fConversores\nCC-CC");
delay_ms(1000);
printf(exibe_lcd, "\fEscolha o\nConversor");
delay_ms(1000);
printf(exibe_lcd, "\fBuck[1]Boost[2]\nBuck-Boost[3] %d", tela.conversor); //Opções para
a escolha do tipo de conversor a ser usado
delay_ms(1000);

while(1){

    if(leitura_botoes == 1){
        leitura_botoes = 0;
        leitura_de_teclas();
    }

    if(botao_apertado == 1){
        botao_apertado = 0;
        switch(tela.telas){
```

```
case 0:
    printf(exibe_lcd, "\fBuck[1]Boost[2]\nBuck-Boost[3] %d", tela.conversor);
    break;

case 1:
    printf(exibe_lcd, "\fEntre com valor\nDa freq f=%ldkHz", tela.frequencia);
    break;

case 2:
    printf(exibe_lcd, "\fDef CicloTrab[1]\nDef Tensao[2] %d", tela.tensao_ciclo);
    break;

case 3:
    printf(exibe_lcd, "\fDefina Ciclo de\ntrabalho
%.0f%%", tela.ciclo_de_trabalho);
    break;

case 4:
    printf(exibe_lcd, "\fDefina Tensao de\ntrabalho %fV", tela.tensao);
    break;

case 5:
    calculo_conversor();
    calcula_frequencia();
    configura_timer2();
    tela_final();
    break;
}
}
}
}
```

```
void configura_timer2()
{

    if(tela.frequencia > 38)
        setup_timer_2(T2_DIV_BY_1,tela.valor_config_timer2,1);
    else
        setup_timer_2(T2_DIV_BY_4,tela.valor_config_timer2,1);

    tela.pwm = (float)((tela.ciclo_de_trabalho*4)*(tela.valor_config_timer2+1));
    set_pwm1_duty(tela.pwm);
    setup_ccp1(CCP_PWM);

}

void calcula_frequencia()
{
    if(tela.frequencia > 38){
        valor_config_timer2_ponto_flutuante = (float)((clock_efetivo / tela.frequencia)-1);
        tela.valor_config_timer2 = valor_config_timer2_ponto_flutuante;
    }
    else{
        valor_config_timer2_ponto_flutuante = (float)(((clock_efetivo_2) / tela.frequencia)-1);
        tela.valor_config_timer2 = valor_config_timer2_ponto_flutuante;
    }
}
```

```
}
```

```
void tipo_de_trab_tensao(int trab_tensao)
```

```
{
```

```
    if(trab_tensao == 1){
```

```
        printf(exibe_lcd,"\fCiclo trabalho\nselecionado");
```

```
        tela.telas = 3;
```

```
    }
```

```
    if(trab_tensao == 2){
```

```
        printf(exibe_lcd,"\fTensao trabalho\nselecionada");
```

```
        tela.telas = 4;
```

```
    }
```

```
    delay_ms(1000);
```

```
}
```

```
void frequencia_selecionada(void)
```

```
{
```

```
    printf(exibe_lcd,"\fFreq selecionada\nf=%ldk Hertz",tela.frequencia);
```

```
    delay_ms(1000);
```

```
}
```

```
void tipo_de_conversor(void)
{

    if(tela.conversor == 1){
        printf(exibe_lcd, "\f%s\nSelecioneado", "Buck");
        tela.tensao = 0;
    }

    if(tela.conversor == 2){
        printf(exibe_lcd, "\f%s\nSelecioneado", "Boost");
        tela.tensao = 34;
    }

    if(tela.conversor == 3){
        printf(exibe_lcd, "\f%s\nSelecioneado", "Buck-Boost");
        tela.tensao = 0;
    }

    delay_ms(1000);

}

void tela_final(void)
{
    if(tela.conversor == 1)
        printf(exibe_lcd, "\f%s f=%ldkHz\n", "Buck", tela.frequencia);

    if(tela.conversor == 2)
```

```

printf(exibe_lcd, "\f%s f=%ldkHz\n", "Boost", tela.frequencia);

if(tela.conversor == 3)
    printf(exibe_lcd, "\f%s f=%ldkHz\n", "Buck-Boost", tela.frequencia);

printf(exibe_lcd, "V=%.1fV D=%.1f%%\n", tela.tensao, (tela.ciclo_de_trabalho*100));

}

void calculo_conversor(void)
{
    tela.ciclo_de_trabalho = tela.ciclo_de_trabalho / 100.0;

    switch (tela.conversor)
    {
        case 1:// buck
            if(tela.tensao_ciclo == 1){
                tela.tensao = tensao_de_entrada * tela.ciclo_de_trabalho;// Tensao de saida
            }
            else
                tela.ciclo_de_trabalho = (float)tela.tensao/tensao_de_entrada;// Ciclo de trabalho

            break;

        case 2:// boost
            if(tela.tensao_ciclo == 1)
                tela.tensao = tensao_de_entrada/(1-tela.ciclo_de_trabalho);//Calcula Tensao
            else

```

```

        tela.ciclo_de_trabalho = (1 - (tensao_de_entrada/tela.tensao));//Ciclo de trabalho
        break;
    case 3: //Buck-Boost
        if(tela.tensao_ciclo == 1)
            tela.tensao = -tensao_de_entrada * (tela.ciclo_de_trabalho/(1-
tela.ciclo_de_trabalho));//Calcula Tensao
        else
            tela.ciclo_de_trabalho = (float)(-tela.tensao/(-tela.tensao +
tensao_de_entrada));//Ciclo de trabalho
        break;
    }
}

```

```

void leitura_de_teclas(void)
{
    switch(tela.telas)
    {
        case 0:
            if(input(confirma_tela) == 0){
                botao_apertado = 1;
                tipo_de_convertor();
                tela.telas++;
            }
            if(input(incrementa) == 0){
                tela.convertor++;
                botao_apertado = 1;
                if(tela.convertor == 4)
                    tela.convertor = 1;
            }
            if(input(decrementa) == 0){

```

```
tela.conversor--;  
botao_apertado = 1;  
if(tela.conversor == 0)  
    tela.conversor = 3;  
}  
break;  
case 1:  
if(input(confirma_tela) == 0){  
    botao_apertado = 1;  
    frequencia_selecionada();  
    tela.telas++;  
}  
if(input(incrementa) == 0){  
    tela.frequencia = tela.frequencia+1;  
    botao_apertado = 1;  
    if(tela.frequencia > Freq_max)  
        tela.frequencia = Freq_max;  
}  
if(input(decrementa) == 0){  
    tela.frequencia = tela.frequencia-1;  
    botao_apertado = 1;  
    if(tela.frequencia < Freq_mim)  
        tela.frequencia = Freq_mim;  
}  
break;  
case 2:  
if(input(confirma_tela) == 0){  
    botao_apertado = 1;  
    tipo_de_trab_tensao(tela.tensao_ciclo);
```



```
    }  
    if(input(incrementa) == 0){  
        tela.tensao_ciclo++;  
        botao_apertado = 1;  
        if(tela.tensao_ciclo == 3)  
            tela.tensao_ciclo = 2;  
    }  
    if(input(decrementa) == 0){  
        tela.tensao_ciclo--;  
        botao_apertado = 1;  
        if(tela.tensao_ciclo == 0)  
            tela.tensao_ciclo = 1;  
    }  
    break;  
case 3:  
    if(input(confirma_tela) == 0){  
        botao_apertado = 1;  
        tela.telas = 5;  
    }  
    if(input(incrementa) == 0){  
        tela.ciclo_de_trabalho = tela.ciclo_de_trabalho + 1;  
        botao_apertado = 1;  
        if(tela.ciclo_de_trabalho == 101)  
            tela.ciclo_de_trabalho = 100;  
    }  
    if(input(decrementa) == 0){  
        tela.ciclo_de_trabalho = tela.ciclo_de_trabalho - 1;  
        botao_apertado = 1;  
        if(tela.ciclo_de_trabalho == -1)
```

```
        tela.ciclo_de_trabalho = 0;
    }
break;
case 4:
if(input(confirma_tela) == 0){
    botao_apertado = 1;
    tela.telas = 5;
}
if(input(incrementa) == 0){
    tela.tensao = tela.tensao + 1 ;
    botao_apertado = 1;
    if(tela.conversor == 1){
        if(tela.tensao == 35)
            tela.tensao = 34;
    }
    if(tela.conversor == 3){
        if(tela.tensao == 1)
            tela.tensao = 0;
    }
}
if(input(decrementa) == 0){
    tela.tensao = tela.tensao - 1;
    botao_apertado = 1;
    if(tela.conversor == 1){
        if(tela.tensao == -1)
            tela.tensao = 0;
    }
}
if(tela.conversor == 2){
    if(tela.tensao == 33)
```

```
        tela.tensao = 34;
    }
}
break;
case 5:
    if(input(decrementa) == 0 && input(incrementa) == 0 && input(confirma_tela) == 0)
        reset_cpu();
    break;
}
}
```