

UNIVERSIDADE FEDERAL DE MATO GROSSO DO SUL
FACULDADE DE ENGENHARIAS, ARQUITETURA E URBANISMO E GEOGRAFIA
CURSO DE ENGENHARIA ELÉTRICA

MARCEL PASCOSKI LUCIO

**IMPLEMENTAÇÃO DE CONTROLE DIGITAL EM UM INVERSOR
FLYBACK TRI-STATE UTILIZANDO DSPIC**

CAMPO GRANDE

Outubro de 2015

MARCEL PASCOSKI LUCIO

**IMPLEMENTAÇÃO DE CONTROLE DIGITAL EM UM INVERSOR
FLYBACK TRI-STATE UTILIZANDO DSPIC**

Trabalho de Conclusão de Curso apresentada
como exigência para obtenção do grau de
Bacharelado em Engenharia Elétrica da
Universidade Federal de Mato Grosso do Sul
– UFMS.

ORIENTADOR: Prof. Dr. Luigi Galotto Junior

CAMPO GRANDE

Outubro de 2015

CURSO DE ENGENHARIA ELÉTRICA
COLEGIADO DO CURSO DE ENGENHARIA ELÉTRICA
COMISSÃO DO TRABALHO DE CONCLUSÃO DE CURSO 1 E 2

CERTIFICADO DE AUTENTICIDADE DE MONOGRAFIA

Eu, MARCEL PASCOSKI LUCIO acadêmico (a) do curso de Engenharia Elétrica da Faculdade de Engenharias, Arquitetura e Urbanismo e Geografia da UFMS, portador (a) do RGA Nº **2010.2103.119-4** certifico que a presente monografia e TRABALHO DE CONCLUSÃO DE CURSO com o título: “**IMPLEMENTAÇÃO DE CONTROLE DIGITAL EM UM INVERSOR FLYBACK TRI-STATE UTILIZANDO DSPIC**” é de minha exclusiva autoria, com a orientação do Prof. Dr. **Luigi Galotto Junior**.

Outrossim, declaro que estou ciente das sanções na esfera civil, penal e ética, sujeitas, caso seja comprovado o plágio e/ou aquisição do trabalho e/ou realização por outra pessoa.

Declaro também que estou ciente das medidas de caráter pedagógico, como a reprovação na disciplina de TRABALHO DE CONCLUSÃO DE CURSO 1 e 2, podendo acarretar impedimento da conclusão do curso.

CAMPO GRANDE-MS, _____ DE _____ DE 2015.

Assinatura do acadêmico

MARCEL PASCOSKI LUCIO

Deus, sabedoria em si.

AGRADECIMENTOS

Primeiramente agradeço a Deus a sabedoria em si, pelo dom da vida.

Gostaria de manifestar a gratidão a minha mãe, Maria Aparecida Pascoski, pela dedicação, amor e apoio incondicional para comigo.

Meu muito obrigado ao professor Me. André Muniz Soares pelo auxílio, confiança e ajudas com os segredos dos microcontroladores, que sem os quais não teria chegado até aqui.

Agradeço ao meu orientador professor Dr. Luigi Galotto Junior, peça fundamental para o labore, pela confiança depositada, presença constante, disponibilidade nos momentos difíceis, pela experiência e presteza nas transmissão do conhecimento.

Agradeço também ao corpo docente da Universidade Federal do Mato Grosso do Sul, os quais por meio da docência, me proporcionaram à possibilidade de uma formação profissional sólida.

Aos amigos Glauber de Freitas, Rodrigo Calunga, Marcelo Rossignolo, Sthefany Fernandes e Guilherme Paraná, que ao longo desta jornada serviram de grande apoio.

Também agradeço o Batlab por ter cedido equipamentos para o experimento e as empresas Wat e Hiperon por terem cedido os esquemáticos e códigos.

RESUMO

A obra apresenta a implementação do controle digital em um protótipo de um Inversor Flyback com modulação Tri-State, com possíveis aplicações em geração distribuída e também na alimentação de sistemas de energia ininterrupta (*UPS - Uninterruptible Power Supply*). O protótipo utilizado foi desenvolvido para atender às especificações de um UPS comercial no padrão monofásico e com tensão de 127 V com frequência de operação 60 Hz e potência de 1 kW para uso em geração ilhada. Inicialmente, este protótipo estava sendo controlado com uma plataforma de prototipagem rápida de controle dSPACE, porém esta plataforma inviabiliza a utilização prática como produto. Desta forma, a principal contribuição deste trabalho está no uso de um controlador digital de sinais (*DSC – Digital Signal Controller*) com ponto fixo dsPIC da família 33F em substituição da plataforma de controle previamente utilizada neste inversor. Foram utilizados controladores clássicos tipo proporcional integral, ressonante e a composição dos mesmos. Os ensaios foram feitos em malha aberta e fechada com o novo controlador implementado em condições de potência e tensão inferiores às condições nominais do protótipo por não ser o objetivo da proposta inicial. Porém foram levantados resultados de tensão e correntes assim como os pulsos de aquisição de sinais e controle para validar o projeto do controlador desenvolvido. Este projeto (*firmware*) montado no MPLAB servirá de modelo para o desenvolvimento de diversas aplicações de controle em projetos futuros, uma vez que o módulo controlador é independente do inversor.

Palavras-chave: Tri-State, Inversor, Controlador Digital de Sinais, Geração Distribuída, Controlador Ressonante

ABSTRACT

The work presents the implementation of digital control in a prototype Flyback Inverter with Tri-State modulation, with possible applications in distributed generation and also the power Uninterruptible Power Systems (UPS - Uninterruptible Power Supply). The prototype used was developed to meet the specifications of a standard single-phase UPS business and 127V voltage with operating frequency 60 Hz and 1 kW for use in islanded generation. Initially this prototype was being controlled with a rapid prototyping platform from dSPACE control, but this platform makes it impossible to practical use as a product. Thus, the main contribution of this work is the use of a digital controller signals (DSC - Digital Signal Controller) with fixed point dsPIC 33F of family control platform replacing previously used in this drive. Classic controllers were used proportional integral type, resonant and composition thereof. The tests were done in open and closed loop with the new controller implemented at lower power and voltage conditions at nominal prototype conditions for not being the objective of the initial proposal. But results have been raised voltage and current as well as pulse signal acquisition and control to validate the project developed controller. This design (firmware) mounted in MPLAB serve as a model for the development of diverse control applications in future designs, since the controller module is independent of the inverter.

Keywords: Tri-State, Inverter, Digital Signal Controller, Distributed Generation, Resonant Controller

LISTA DE FIGURAS

| | |
|--|-----------|
| Figura 1. Estados de Modulação Tri-State..... | 16 |
| Figura 2. Inversor Flyback | 18 |
| Figura 3. Inversor Flyback funcionando com modulação Tri-State | 19 |
| Figura 4. Esquemático de pinos do DSPIC33FH256GP710A | 22 |
| Figura 5. Placa de controle com DSPIC | 22 |
| Figura 6. Conector para Inversor | 23 |
| Figura 7. Código da configuração dos Timers | 27 |
| Figura 8. Bancada em teste dos primeiros pulsos de PWM gerados pelo dsPIC | 28 |
| Figura 9. ICD 3 Tool Kit | 28 |
| Figura 10. Esquemático Circuito de modulação..... | 29 |
| Figura 11. Foto do PLD no circuito do inversor | 30 |
| Figura 12. Esquemático do circuito de Gate Drive..... | 31 |
| Figura 13. Acesso ao sinais modulados..... | 32 |
| Figura 14. Núcleo | 33 |
| Figura 15. Esquemático do circuito de Potência..... | 34 |
| Figura 16. Circuito de condicionamento da Corrente no núcleo, da rede, tensão na Bateria e Externa..... | 35 |
| Figura 17. Circuito de condicionamento da tensão de saída, da rede e da corrente da rede | 36 |
| Figura 18. Implementação da chave “tic-tac” e botão | 37 |

| | |
|---|-----------|
| Figura 19. Fluxograma Main..... | 38 |
| Figura 20. Fluxograma Controle..... | 39 |
| Figura 21. Processo de amostragem e armazenamento | 40 |
| Figura 22. Modulação com sobreposição dos pulsos | 41 |
| Figura 23. Pulsos de Amostragem e Controle..... | 42 |
| Figura 24. A Rampa de partida de fluxo no núcleo e a tensão na saída no inversor | 43 |
| Figura 25. Frequência 67 Hz | 43 |
| Figura 26. Corrente no Sensor e Onda senoidal..... | 44 |

LISTA DE TABELAS

| | |
|---|-----------|
| Tabela 1. Mappin do principais sinais condicionados da planta..... | 25 |
| Tabela 2. Tabela dos acessos ao sinais gerados..... | 32 |

SUMÁRIO

| | | |
|--------------|---|-----------|
| 1 | INTRODUÇÃO..... | 13 |
| 2 | REVISÃO BIBLIOGRÁFICA | 15 |
| 2.1 | Aplicações de controles digitais em inversores | 15 |
| 2.2 | Modulação Tri-State..... | 15 |
| 2.3 | Inversor Flyback..... | 17 |
| 3 | METODOLOGIA | 20 |
| 3.1 | Aspectos teóricos fundamentais..... | 20 |
| 3.1.1 | Microcontrolador DSPIC 33FJ256GP710A | 20 |
| 3.1.2 | Visão Geral Do DSPIC 33FJ256GP710 | 21 |
| 3.1.3 | Pinagem do DSPIC33F | 21 |
| 3.1.4 | Esquemáticos para funcionamento do dsPIC | 22 |
| 3.1.5 | Ajuste da velocidade de Operação | 24 |
| 3.1.6 | Principais componentes utilizados | 24 |
| 3.1.7 | Timer e Interrupção | 25 |
| 3.1.8 | PWM | 27 |
| 3.1.9 | MPLAB | 28 |
| 3.2 | Placa do Inversor | 28 |
| 3.2.1 | Circuito de Modulação | 29 |
| 3.2.2 | O Programmable Logic Device AM16V8 | 29 |
| 3.2.3 | Circuito Gate Drive | 31 |
| 3.2.1 | Circuito de Potência e Núcleo | 32 |
| 3.2.1 | Circuito de Sensoriamento e Condicionamento de Sinais..... | 35 |
| 3.3 | Botão e Chave para Acionamento | 36 |
| 3.4 | O Algoritmo..... | 37 |
| 4 | SIMULAÇÃO E ANÁLISE DE RESULTADOS | 41 |
| 4.1 | Pulsos de Modulação com Sobreposição..... | 41 |
| 4.2 | Pulsos de amostragem e controle..... | 41 |
| 4.3 | Rampa de partida e onda senoidal | 42 |

| | | |
|-----|--|----|
| 4.4 | Onda senoidal gerada e leitura do sensor de corrente | 44 |
| 5. | CONCLUSÃO | 45 |
| | REFERÊNCIAS | 46 |
| | APÊNDICE A – CONFIGURANDO TIMERS | 48 |
| | APÊNDICE B – CONFIGURAÇÃO DO PWM..... | 50 |
| | APÊNDICE C – FUNÇÃO SWITCH E DMA..... | 51 |
| | APÊNDICE D – MAIN E CONTROLE..... | 53 |
| | ANEXO A – ESQUEMÁTICO DA PLACA DE CONTROLE | 63 |
| | ANEXO B - ADC | 64 |

1 INTRODUÇÃO

Com a crescente demanda energética associada a redução da oferta de fontes de combustíveis convencionais e ao crescimento populacional, e também por consequência da preocupação ambiental, surgem pesquisas e o desenvolvimento de fontes alternativas de energia. Dentre a diversas formas de se gerar energia elétrica, estão as fontes alternativas de energia têm sido desenvolvidas e estão ganhando força impulsionadas principalmente como já dito, pelo aumento da preocupação com as questões ambientais, notadamente o efeito estufa, que promove a mobilização mundial a fim de reduzir a emissão de gases poluentes.

Os conversores encontrados na Eletrônica de Potência[1,2], estão diretamente ligados as fontes renováveis que em sua maioria geram em corrente contínua, estes conversores consistem em circuitos eletrônicos responsáveis por converter energia elétrica de um nível de tensão, de corrente e de frequência para outros utilizando chaves interruptoras eletrônicas. Uma característica essencial destes circuitos é que as chaves operam somente em bloqueio e condução. E que por meio de modulação e da utilização da topologia adequada, pode-se converter energia elétrica com rendimento elevado, diferentemente do que ocorre quando se utilizam interruptores operando na região ativa[3].

Os conversores CC-CA, são responsáveis pela aquisição de níveis de tensão CC na entrada e de níveis alternados de tensão ou corrente em sua saída, sendo assim deve-se apresentar simetria em amplitude e sua frequência de operação pode ser fixa ou variável. Em âmbito industrial, os inversores são amplamente utilizados, como por exemplo acionamento de motores de indução em velocidade variável, cuja entrada de um conversor CC-CA pode ser uma bateria, célula solar, ou uma fonte CC.

Uma das principais aplicações dos inversores são os sistemas de energia ininterrupta ou simplesmente os nobreaks. Neste sistema o inversor é a etapa de saída dos nobreaks, quando a rede está ativa este bloco é alimentado pela tensão de saída do retificador, retirando sua energia da rede. Quando por um momento existir uma falta de energia na rede este bloco passa a ser alimentado pelo conversor DC/DC, retirando sua energia das baterias, de modo que não falte energia na saída, sendo esta a ideia central de um nobreak.

A tensão de saída de um inversor pode ser fixa ou pode variar conforme uma frequência, seja ela fixa ou variável. Uma tensão variável de saída pode ser obtida variando-se a amplitude da tensão CC de entrada e mantendo-se o ganho do inversor constante. Por outro lado, se a tensão CC de entrada for fixa e não controlável, uma tensão variável de saída pode ser obtida

pela variação do ganho do inversor, a qual é normalmente realizada pelo controle modulação por largura de pulso (do inglês *pulse width modulation* – PWM) dentro do inversor.

O trabalho a seguir tem por consequência, a implementação do controle digital em um inversor monofásico de tensão senoidal na saída, com auxílio de um processador digital de sinais para a realização das malhas de controle, com uma metodologia que seja de fácil aplicação e visando baixo custo e a eficiência.

No capítulo 2, está apresentada uma breve revisão bibliográfica e no capítulo 3 é detalhada a metodologia, onde estão descritos os materiais utilizados, os hardwares dos circuitos de controle digitais e de potência e a explicação sobre os principais algoritmos implementados. Os resultados e conclusões estão apresentadas nos capítulos 4 e 5, enfocando os resultados da implementação digital. O projeto utilizado para a gravação também está disponibilizado nos Apêndices.

2 REVISÃO BIBLIOGRÁFICA

2.1 Aplicações de controles digitais em inversores

Na tese apresentada em[4], mostra um uso de um controlador digital de sinais da FreeScale para se obter inversor Trifásico Buck-Boost com modulação Tri-State, e também na referência citada mostra-se um protótipo e os esquemáticos em Simulink, na plataforma DSpace para o mesmo inversor. A diferença principal de se utilizar um dispositivo embarcado, é que com o embarcado pode-se ter um maior grau de liberdade para aplicações industriais, sem a presença de um computador.

Na tese acima evidencia-se o uso de tabelas para simplificar alguns cálculos que ocuparão ciclos de *clock*. Desta forma, a economia destes ciclos de tempo é papel fundamental para desempenho do código, pois “sacrifica-se” a memória do controlador digital, porém ganha-se em desempenho. Isto significa que seu código ficará mais simplificado e por consequência mais rápido. Este inversor será aplicado em geração distribuída especificadamente para sistemas ligados a rede para geração de energia por meio de placas fotovoltaicas, para sistemas Trifásicos.

Também evidencia-se o trabalho em[5], onde foi implementado um inversor monofásico ponte completa que poderá ser utilizado na geração distribuída e também para sistema de energia ininterrupta. Em seu protótipo foi utilizado um dsPIC para o controle digital com ponto fixo, com complemento a dois e sem sinal. Ele tem modulação PWM senoidal e trabalha com controlador Proporcional Integral. E também utiliza se da estratégia de tabelas para melhora do desempenho.

2.2 Modulação Tri-State

Os conversores Tri-State foram a priori utilizados para melhorar o desempenho dinâmico reduzindo o efeito do zero no semi-plano direito [6], apresentado no conversor boost convencional. Esta melhoria se deu utilizando apenas um grau de liberdade, enquanto outra variável foi mantida constante. Os conversores Boost Tri-state ou Buck-boost tem a vantagem de aumentar o limite do ganho estático em comparação com os conversores convencionais com dois estados de modulação[7-8].

A modulação Tri-State pode ser implementada usando um sinal adicional de modulação, para definir o ciclo de trabalho do estado adicional (D2), como é mostrado na figura 3.

Os estados de operação são:

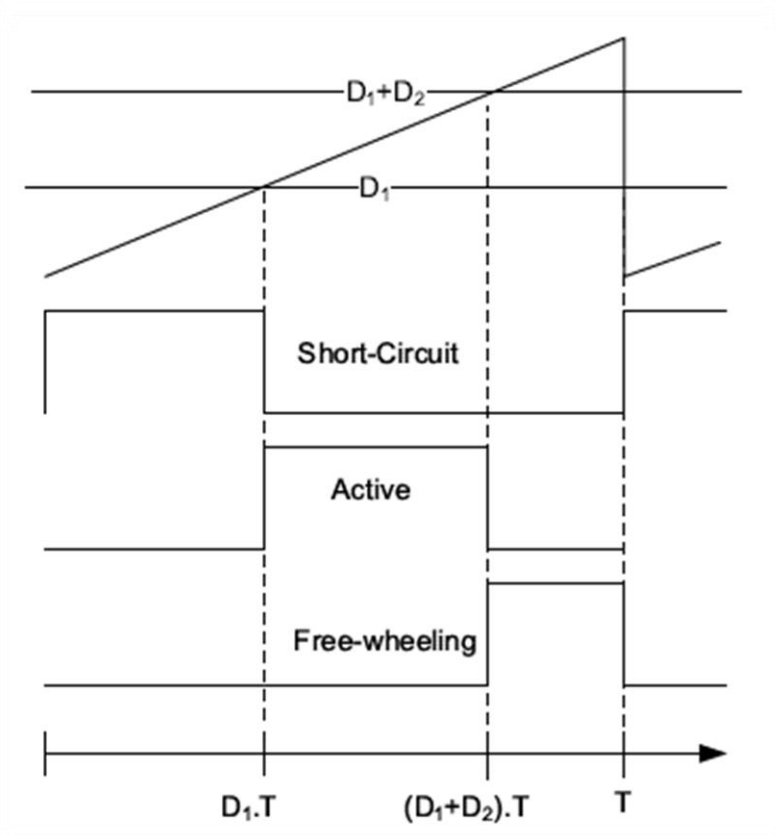
- Estado De Curto Circuito

- Estado Ativo (positivo ou negativo)
- Estado Roda Livre

Para se executar a modulação Tri-State em conversores, basta dois pulsos de comutação, já que o diodo está em estado normalmente ativo. Já em inversores, para melhorar o controle o diodo é substituído por um interruptor eletrônico, ou seja, uma chave de potência.

No estado de curto-circuito citado acima, é responsável por carregar o indutor durante $D_1.T$, mesmo período de um inversor convencional. A segunda etapa estado ativo, a energia do indutor é utilizada para carregar um capacitor de saída, e este ocorre durante o período $(1-D_1).T$ que é o tempo para $D_2.T$. O intervalo que restou fica para o tempo de roda Livre $(1-D_1-D_2).T$, com o objetivo de manter a energia no indutor de modulação constante, como mostra a figura 1.

Figura 1. Estados de Modulação Tri-State



Fonte: Galotto Jr, 2011

As características comumente exploradas quando se utiliza da modulação Tri-State, é que nesta modulação podemos obter um controle com as saídas independentes das entradas[8], que por exemplo em painéis fotovoltaicos permite a extração de maior potência do módulo.

Isto torna interessante o uso desta técnica para inversores aplicados a fontes renováveis de energia.

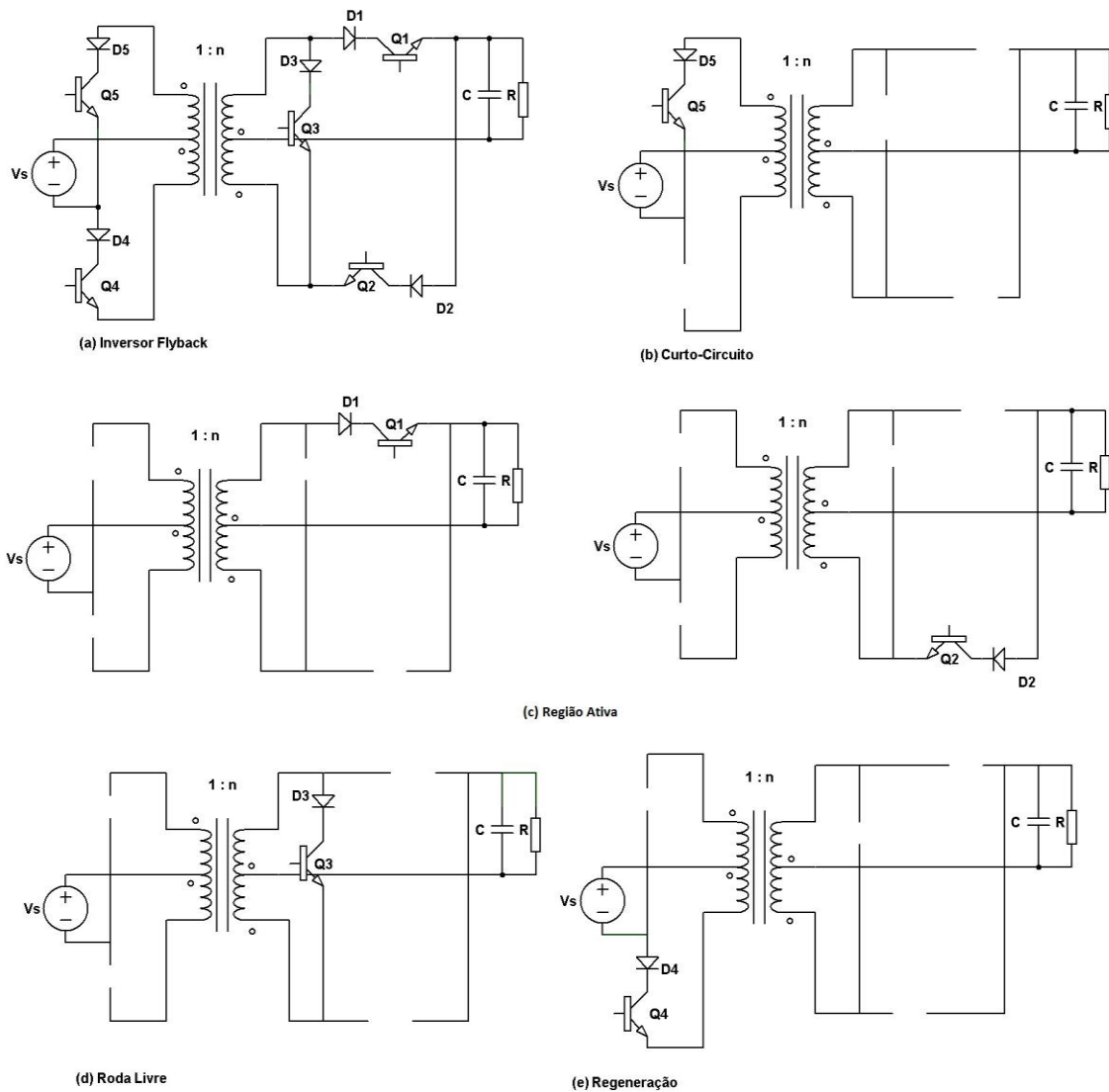
2.3 Inversor Flyback

O inversor de tensão também conhecido como conversor CC-CA, realiza a transformação da tensão contínua em tensão alternada com uma determinada frequência e amplitude. A topologia a ser estudada, será um inversor na configuração Flyback e modulação Tri-State que se trata de uma topologia isolada do inversor Buck-Boost[9-11].

Os inversores isolados têm característica peculiar que consiste no ganho de tensão apenas com a alteração no número de espiras. Assim como os outros inversores, o inversor flyback tem uma tensão de contínua na entrada e na sua saída a tensão é alternada.

A Fig.2 (a) mostra o esquemático da topologia de um inversor flyback. As chaves Q1,Q2, e Q5 são chaves ativas do processador inversão, já Q3 e Q4 são chaves auxiliares, a chave Q3 era um diodo que foi substituído por uma chave e Q4 serve para trabalhar com regeneração. Na Fig. 2 (b) o inversor liga apenas a chave Q5 curto-circuitando o indutor e carregando através da fonte de alimentação, este estado trabalha semelhante a um funcionamento de um transformador. A Fig. 2 (c) mostra as regiões ditas ativas, estas regiões de funcionamento são fundamentais para que apareça na saída do inversor, as tensões positiva e negativa (dita alternada). Já a Fig. 2 (d) consiste na etapa que em que a energia não descarregada do indutor passa para a carga no modo roda livre, deixando apenas a chave Q3 ligada. Por fim a Fig. 2 (e) deixa a chave Q4 ligada e a tensão que está à direita do inversor é transferida para a fonte, funcionando como se fosse um carregador, este processador é chamado regeneração. O processo de inversão ocorre na dita “região ativa” e essa energia é drenada para a carga. Neste processo como mostra a figura Fig. 2 (c), é possível observar a obtenção da tensão positiva e negativa causando o efeito de alternância da tensão gerada.

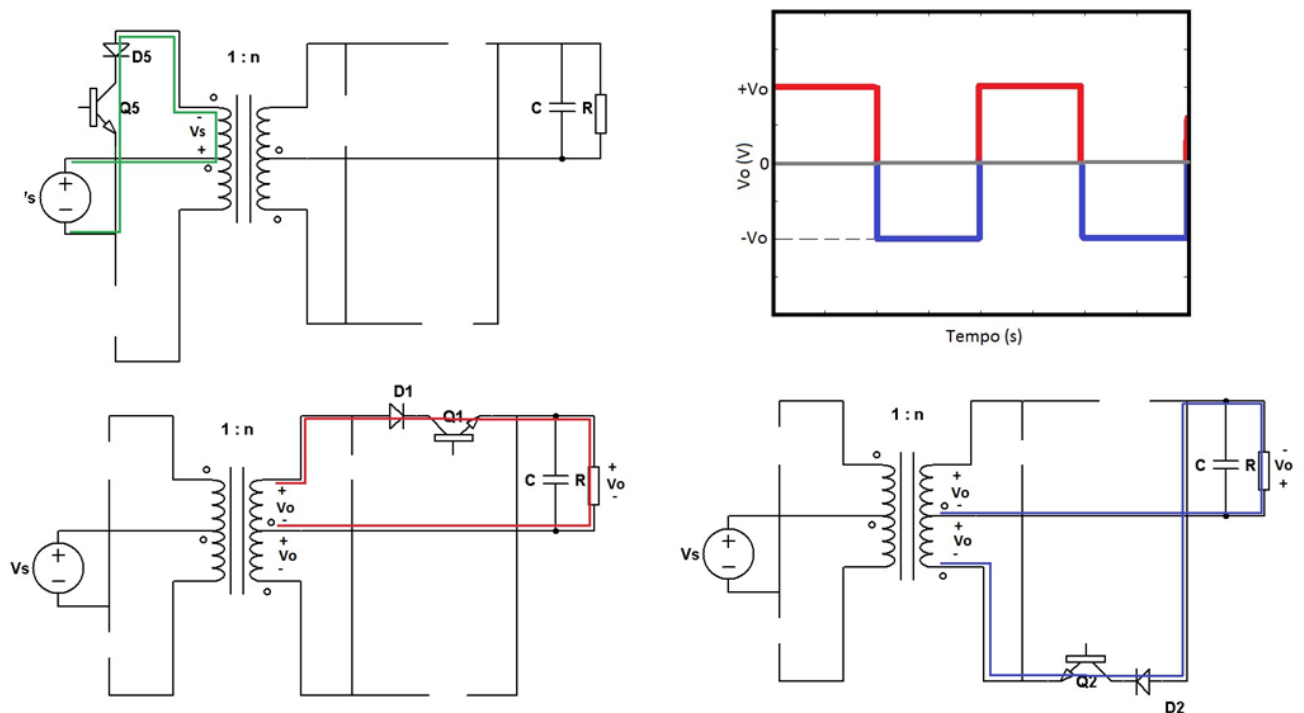
Figura 2. Inversor Flyback



Fonte: Marcela Onikuza, 2013

A figura 3 mostra um inversor flyback trabalhando em modulação Tri-State, no estágio de curto-circuito é carregado o indutor do primário e então, quando a chave Q5 abre e chave que fecha será apenas Q1, esta dará caminho a corrente e uma tensão positiva aparecerá na saída, logo após é aberta a chave Q1 e fechada a chave Q2, dando caminho a corrente e parecerá uma tensão negativa na saída do inversor. Desta forma temos uma inversão de polaridade da tensão gerando as tensões positiva e negativa, claro que desprezando o efeito do capacitor na saída. Na presença do capacitor aparecerá uma tensão de saída que será o valor médio da tensão que aparece semelhante a um conversor CC-CC.

Figura 3. Inversor Flyback funcionando com modulação Tri-State



Fonte: Marcela Onikuza, 2013

Como um inversor se assemelha a uma fonte de corrente, não podemos deixá-lo com todas as chaves abertas, para isso sempre deverá ter uma chave aberta, para dar caminho a corrente. Para se fazer uma corrente de forma senoidal é preciso trabalhar com o funcionamento do roda-livre, isto justifica a sua substituição por uma chave controlável.

Durante o semi-ciclo positivo da corrente as chaves $Q1$ e $Q3$ conduzem modulando a corrente, quando a corrente chega ao valor zero $Q1$ abre e fica somente $Q3$ fechada deixando caminho a corrente. Analogamente, no semi-ciclo negativo, a diferença é que chaves $Q2$ e $Q3$ conduzem modulando a corrente e quando $Q2$ chega ao valor zero, abre $Q2$ e $Q3$ permanece $Q3$ fechada deixando um caminho a corrente.

3 METODOLOGIA

A metodologia utilizada neste trabalho foi realizar uma pesquisa bibliográfica, buscando informações técnicas do DSPIC família 33F, modulação Tri-State e linguagem C++ e topologia de conversores. Após esta etapa, foi traduzido um código de Simulink para C++ apresentando melhoramentos e finalmente a implementação do controlador digital em bancada do inversor Flyback Tri-State coletando dados que propiciaram a análise e a obtenção dos resultados.

3.1 Aspectos teóricos fundamentais

Para que um controlador digital tenha um bom desempenho quando aplicado a inversores, é necessário satisfazer um conjunto de requisitos. Controlar uma planta é um processo que consiste em várias etapas, e cada etapa é de suma importância para desempenho do inversor. Para que o processador digital possa ter acesso às informações da planta, se faz necessário o uso de sensores de corrente e tensão locados e conectados em pontos estratégicos da planta. Estes sinais devem ser condicionados aos níveis específicos de tensão e corrente das entradas analógicas do dsPIC. Os sinais resultantes são amostrados por meio de um conversor analógico-digital (conversor AD) em um circuito interno do DSPIC, para que possam ser utilizados posteriormente. Analogamente aos sinais de entrada, os sinais de saída do DSPIC, ou seja, os sinais digitais de PWM para acionamento dos interruptores, devem ser convertidos aos níveis de tensão específicos do circuito do PLD e depois acionar os *Gate drivers* utilizados na planta.

O sistema a seguir é composto por três malhas de realimentação, uma de tensão e duas de corrente. Cada malha é composta ou por um controlador proporcional integral ou proporcional ressonante, para compensação. As amostras digitais da planta são obtidas a uma taxa de 25 kHz, sendo que o algoritmo de controle é executado no DSPIC na mesma taxa, gerando a correspondente ação de controle. Isso permite um controle dinâmico, atuando sempre que houver uma perturbação na planta do sistema.

Serão apresentados nesta metodologia os aspectos básicos para a configuração do dsPIC para deixá-lo apto para aquisição das leituras dos sinais e também para a estratégia de controle. Os demais códigos, configurações ou comentários estarão disponíveis nos Anexos e Apêndices.

3.1.1 Microcontrolador DSPIC 33FJ256GP710A

Para o controlador da planta foi escolhido o microcontrolador dsPIC 33FJ256GP710, devido suas características para aplicações que requerem o uso do PWM e por apresentar alta

performance em controle de sinal digital, tais propriedades são exigidas em aplicações industriais, ou de aplicações residenciais, e desejadas para o presente trabalho. “A família para controle suporta uma variedade de aplicações para controle de motores, tais como DC, de indução, relutância chaveada monofásicos e trifásicos. Esses produtos são também muito bem utilizados para Sistemas de Energia Ininterrupta (UPS), inversores, correção de fator de potência e também para Gerenciamento de energia em servidores, equipamentos de telecomunicações e outros equipamentos industriais” [12]. Além disso, o dispositivo é dotado de uma grande velocidade de processamento, que pode chegar aos 40 MPIS (milhões de operações por segundo) dependendo do cristal escolhido.

3.1.2 Visão Geral Do DSPIC 33FJ256GP710

A família do DSPIC33F possui uma poderosa arquitetura de 16 bits, que integra as ferramentas de controle de um microcontrolador e as capacidades computacionais de um Processador Digital de Sinais (DSP). Essa combinação, resulta numa funcionalidade ideal para aplicações que se baseiam numa computação cíclica de alta velocidade[12].

As principais características do dispositivo são:

- Possui 256KB de memória flash.
- Até cinco interrupções externas.
- Até 9 Timers de 16 bits.
- Dois módulos de comunicação SPI, Serial, I2C, CAN, ECAN, DCI.
- Dois módulos de A/D de 10 bits ou 12 bits, com velocidades máximas de 2,2 MSps e 1 MSps (*Mega Samples Per Second*), respectivamente.
- Controle do PWM, até 8 canais independentes ou 4 complementares.
- Até 85 portas programáveis de entrada e saída.

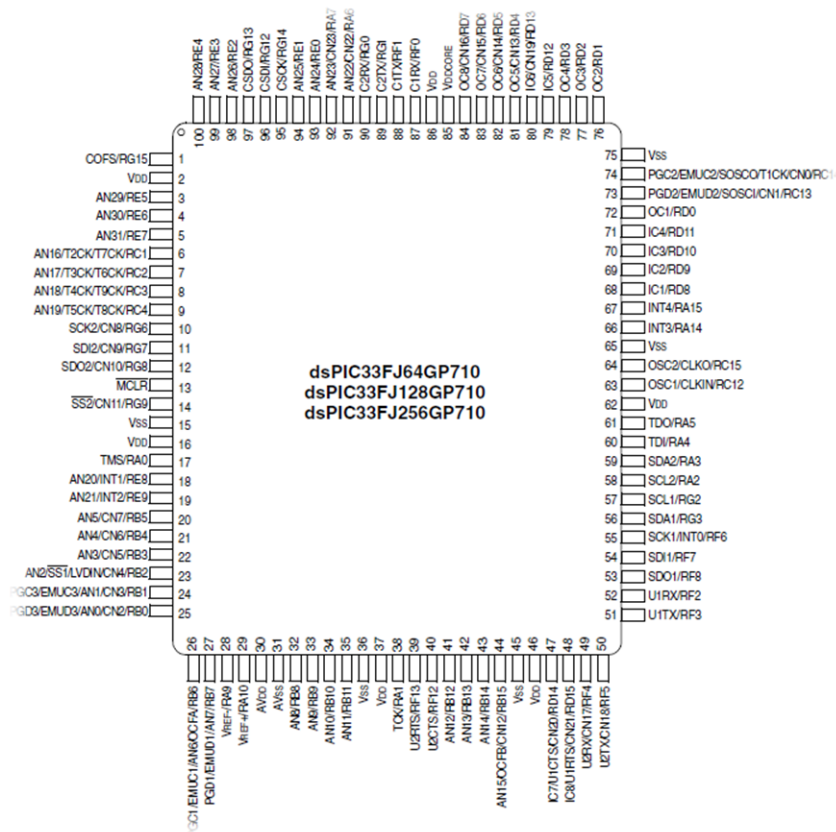
3.1.3 Pinagem do DSPIC33F

A Fig.4 abaixo ilustra a pinagem do DSPIC e também são descritas algumas das suas principais funcionalidades [12].

- OCxRS– Pinos responsáveis pela saída do PWM .
- RxY – onde X assume valores de A a G, e Y assume valores de 0 a 15, e as portas podem ser definidas como entrada e saída.
- Vdd, Vss – Alimentação e terra, respectivamente.

- Vref+, Vref- – Tensões de referência para o conversor A/D, positiva e negativa, respectivamente.
- ANx – onde x assume valores de 0 a 15, entradas analógicas.
- UxRX, UxTX – onde x assume os valores de 1 a 2, pinos de transmissão e recepção via UART (comunicação serial).

Figura 4. Esquemático de pinos do DSPIC33FH256GP710A

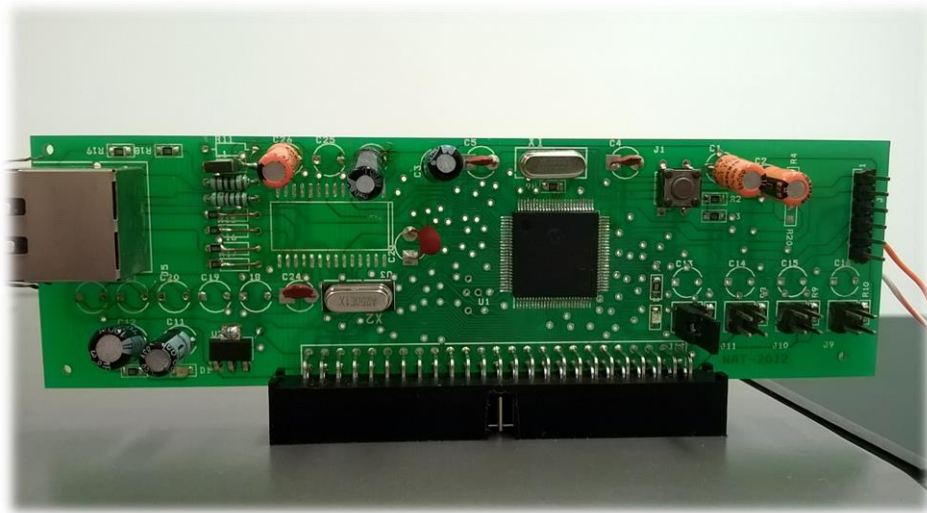


Fonte: Microchip Datasheet DSPIC33F Family

3.1.4 Esquemáticos para funcionamento do dsPIC

Para que um dsPIC seja utilizado é preciso, fazer um circuito que o contenha, e também que contenha circuitos essenciais tais como, circuito de alimentação, circuito de reset, circuitos de programação, circuito oscilador, circuito de conexão com o inversor, circuito de filtragem PWM e também se caso for desejado circuito de comunicação via ethernet. Na figura 5, está o produto dos esquemático com todos circuitos elaborados pela empresa Hiperon e propriedade da empresa Wat. Já o esquemático poderá ser melhor visualizado no Anexo A.

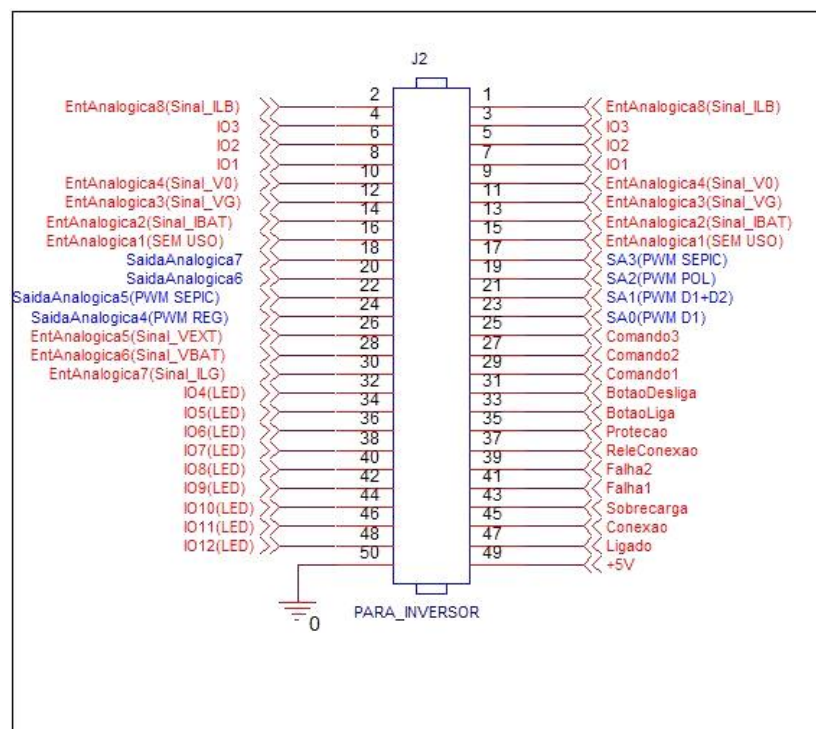
Figura 5. Placa de controle com DSPIC



Fonte: Próprio autor

Para a conexão foi utilizado um conector com 50 pinos, conforme a figura 6.

Figura 6. Conector para Inversor



Fonte: Cortesia Hiperon adaptada

Para que o inversor possa se comunicar com o dsPIC, é necessário que os pinos da placa de controle, estejam configurados conforme os pinos da placa de potência. Esta conexão é de suma importância, pois ela possibilitará a aquisição de sinais da advindos do inversor para

leitura e execução dos algoritmos de controle, que são essenciais para o bom funcionamento do controlador.

3.1.5 Ajuste da velocidade de Operação

Para utilização desejada devemos ajustar o *Clock* de operação que o dispositivo irá trabalhar, para isto a equação abaixo é utilizada:

$$F_{os} = F_{in} \frac{M}{N_1 * N_2} \quad (1)$$

Onde M, N1 e N2 são PLLDIV, PLLPRE e PLLPOST respectivamente, são constantes que podem assumir os seguintes valores:

- PLLPOST – 4 valores (2 a 8)
- PLLPRE – 32 valores (2 a 33)
- PLLDIV – 512 valores (2 a 514)

Por meio dessas configurações, pode-se fazer com que o microcontrolador chegue até a 40 MIPS. Em nosso projeto as seguintes escolhas foram feitas:

- FIN = 8 MHz (Cristal HS)
- PLLDIV = 40, PLLPRE = 2
- PLLPOST = 2

Dessa forma, teremos uma oscilação de 80 MHz e sabido que a velocidade do ciclo de instruções é a metade, logo teremos 40 MIPS.

3.1.6 Principais componentes utilizados

A conversão analógica-digital desempenha um papel fundamental para o controle digital. Se ela não for feita de modo satisfatório, as variáveis de controle poderão ser afetadas, gerando uma resposta não condizente com a esperada.

O módulo do Conversor AD pode ser configurado para 10 ou 12 bits. Neste trabalho utilizaremos 12bits para a conversão e esta configuração está disponível em Anexo B.

Para configura o Conversor AD dever ser obedecidas as seguintes etapas:

- Selecione os pinos das portas para as entradas analógicas desejadas. (ADxPCFGH<15:0>) ou (ADxPCFGL<0:15>).

- Selecione da referência de tensão para que fiquem de acordo com as entradas analógicas (ADxCON2<15:13>).
- Selecionar o Clock de conversão analógica para que essa seja compatível com o Clock do processador (ADxCON3<7:0>).
- Determinar quantos canais de S/H serão utilizados (ADxCON2<9:8> e (ADxPCFGH<15:0> ou ADxPCFGL<15:0>).
- Selecione a sequência de conversão/amostragem apropriada (ADxCON1<7:5> e ADxCON3<12:8>).
- Determine a forma com que os dados serão mostrados (ADxCON1<9:8>).
- Habilite o módulo ADC (ADxCON1<15>). Configure o ADC (se requerido)
- Limpe o ADxIF bit.
- Selecione a interrupção de prioridade do ADC.

Devido a sua extensão, será apresentado um tabela 1 que evidência os principais sinais utilizados fazendo um *Mappin* que relaciona função, pino do dsPIC e a pinagem do conector com o placa.

Tabela 1. Mappin do principais sinais condicionados da planta

| Sensoriamento do sinais | Porta no dsPIC | Pino do conector |
|-------------------------|---------------------|------------------|
| Sinal_IBAT | AD1PCFGLbits.PCFG9 | 13 |
| Sinal_VG | AD1PCFGLbits.PCFG10 | 11 |
| Sinal_VO | AD1PCFGLbits.PCFG11 | 09 |
| Sinal_VEXT | AD1PCFGHbits.PCFG29 | 26 |
| Sinal_BAT | AD1PCFGHbits.PCFG30 | 28 |
| Sinal_ILB | AD1PCFGLbits.PCFG12 | 1-2 |
| Sinal_ILG | AD1PCFGHbits.PCFG31 | 30 |

Para que o ADC pudesse ler somente as porta desejadas foi implementado a função SWITCH que estará na Apêndice C, para que pudéssemos monitorar somente as variáveis desejada na ordem desejadas.

3.1.7 Timer e Interrupção

Este recurso é presente em quase todos microcontroladores. É utilizado do para criar uma base de tempo fixa para o programa. O Timer na verdade é um contador, que em nosso

caso é de 16 bits. Este é zerado após atingir o valor para o qual foi configurado. No sistema apresentado para controle do inversor, o timer foi utilizado para gerar a base de tempo para algumas funções do firmware de controle.

A interrupção é um recurso que hardware que indica que algum evento ocorreu. Esse pode ser de qualquer natureza, como por exemplo em nosso caso, o final de uma conversão AD ou que foi decorrido um tempo a priori configurado. Uma vez ocorrida uma interrupção, essa é sinalizada e pode - se optar em para a rotina principal e executar a outra rotina, ou simplesmente ignora – lá conforme previamente configurada sua prioridade.

Como no dsPIC existem vários recursos, ele pode gerar interrupções em circunstâncias específicas para que o as exigências código sejam atendidas, e após executar a interrupção volta à sua rotina normal de execução. Como já dito, as interrupções serão atendidas conforme sua prioridade ou grau de prioridade, logo cada periférico terá sua prioridade atribuída conforme seu grau. O grau de prioridade pode ser alterado, configurando-se manualmente a ordem na qual as interrupções serão atendidas[12]. Em nosso sistema, o tempo de timer é configurado segundo a equação:

$$PR2 = \frac{F_{cy}}{PRESCALE} \text{ tempo} \quad (2)$$

Onde PR2 é o período do Timer 2, para que ele conte o tempo desejado. Fcy é o número de operações por segundo do dispositivo, o PRESCALE é o valor que pode ser programado para atingir certos valores e o tempo é o valor em segundos desejado. Em nosso caso foi configurado o timer 2 para trabalhar com 25 kHz para o chaveamento e aquisição dos sinais e para o controle foi escolhido o timer 6 com 7 kHz com evidencia a figura 7. Para mais detalhes consulte código contido no Apêndice A.

Figura 7. Código da configuração dos Timers

```

void Init_Timers(void)
{
    //Configurando o Timer 2 para geração de sinal de saída:.....
    T2CONbits.TON = 0; //Desabilita o timer 2.
    T2CON = 0b0000000000000000;
    IFS0bits.T2IF = 0;
    IPC1bits.T2IP = 5;
    PR2 = periodoSaída;
    TMR2 = 0; //Inicializando com 0 o Timer 2
    IEC0bits.T2IE = 1; //Habilitando a interrupção do Timer 2 gerada devido a
    TMR2 = 0;
    IPC1bits.T2IP = 0x01; //Def. o nível de prioridade da interrupção do Timer2
    IFS0bits.T2IF = 0; //Limpa o Flag de Interrupção do Timer 2.
    IEC0bits.T2IE = 1; //Habilitando a interrupção do timer 2.
    T2CONbits.TON = 1; //Iniciando o Timer 2.
    //Configurando o Timer 6 para geração de sinal de saída:.....
    T6CONbits.TON=0;
    T6CON = 0b0000000000010000; //0b00000000000010000
    IFS2bits.T6IF = 0;
    IPC1bits.T6IP = 5;
    PR6 = 760; //periodoSaída;
    TMR6 = 0;
    IEC2bits.T6IE = 1;
    IPC1bits.T6IP = 0x02;
    IFS2bits.T6IF = 0;
    IEC2bits.T6IE = 1;
    T6CONbits.TON = 1;
} //Fim da função "Init_Timers"

```

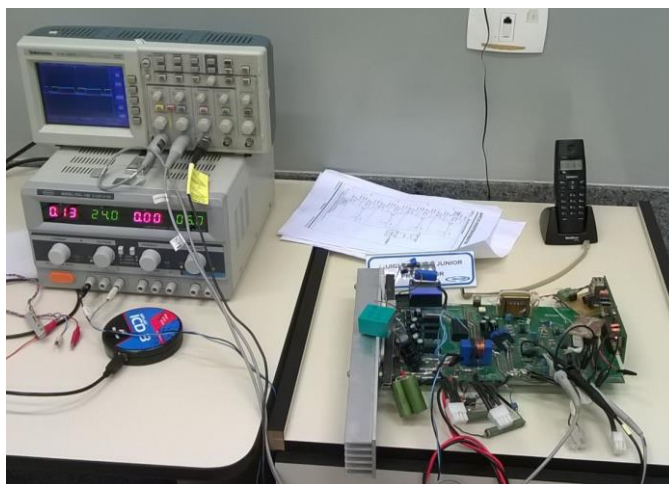
Fonte: Próprio Autor

3.1.8 PWM

Este periférico é de suma importância para o controle, que será bem utilizado nesta aplicação. A família de DSPIC em especial, tem este periférico em todos os modelos. O módulo PWM é composto por até 8 canais independentes, 8 razões cíclicas programáveis, geradores de *DEAD TIME*, sincronização com o conversor AD e outras funções adicionais. A base de tempo do PWM é gerada por um registrador chamado OCRxRS e OCxR. Esses registradores de 15 bits podem contar em ordem crescente, decrescente ou ambos, dependendo do modo de funcionamento. Os modos de funcionamento, tais como Active-Low, One-Shot, Active-High One Shot, Toggle, Delayed One-Shot, Continuous Pulse e PWM. O período é definido pelo registrador OCRxRS, cujo o valor do registrador é inserido conforme solicitado de forma numérica ou por meio de uma função advinda do controle. Mais detalhes sobre o PWM podem ser encontrados em [12].

A nossa planta consta de 7 chaves de potência, para que se tenha controle sobre todas, é preciso ter 7 sinais de PWM saindo do controlador para o circuito modulador. Após feitas as configurações que podem ser encontradas no Apêndice B, foram gerados os primeiros sinais de PWM sem a conexão da potência, com razões cíclicas fixas que serão mostrados com osciloscópio em bancada, conforme figura 8.

Figura 8. Bancada em teste dos primeiros pulsos de PWM gerados pelo dsPIC



Ponte: Próprio Autor

3.1.9 MPLAB

Para desenvolver o código e gravar no dsPIC, utilizou-se o ambiente MPLAB X IDE v3.00.02 -beta com o compilador C30 e a interface ICD3 Tool Kit como mostra a figura 9, ambos da MICROCHIP. O MPLAB é de uso gratuito e recomendado pelo fabricante. Todos os registradores e funções mostradas acima correspondem ao MPLAB, podendo não ser compatíveis com outros ambientes de desenvolvimento.

Figura 9. ICD 3 Tool Kit



Fonte: Próprio autor

3.2 Placa do Inversor

Na placa do inversor consta circuitos fundamentais para que o PWM funcione de maneira a gerar um onda senoidal saída. Estes circuitos são:

- Circuito de modulação
- *Programmable logic device - PLD*

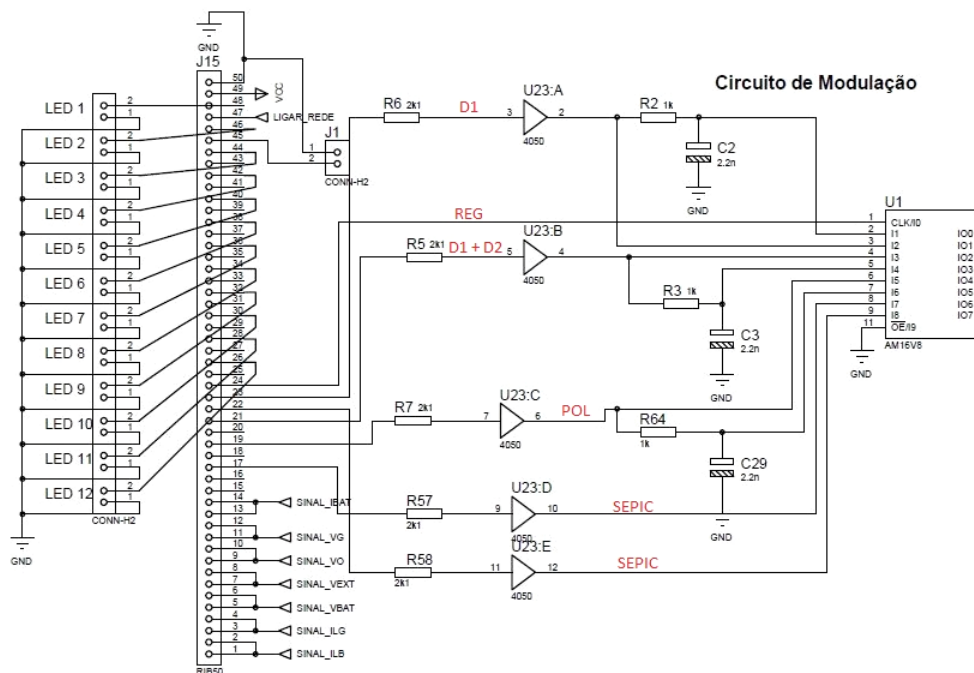
- Circuito de potência
- Circuito de sensoriamento e condicionamento de sinais

Mais detalhes sobre os circuitos citados, serão melhor explanados adiante no texto.

3.2.1 Circuito de Modulação

O circuito de modulação é o circuito que capta os pulsos de PWM gerados pelo controlador e encaminha até o *Programmable logic device – PLD*. Todo esse trajeto dos pulsos até o PLD é feito em 3,3 Volts e com correntes de miliamperes. Acoplado ao circuito de modulação está um circuito muito importante que é o circuito de defasagem, para que os pulsos gerados sempre tenham uma interseção, isto é muito importante pois deve se ter sempre uma chave em condução, caso contrário subirá muito a tensão na chaves provocando um desgaste. A figura 10 abaixo, apresenta o esquemático do circuito modulador e também o PLD que falaremos mais a seguir.

Figura 10. Esquemático Circuito de modulação



Fonte: Cortesia Wat adaptada

3.2.2 O Programmable Logic Device AM16V8

O PLD dispositivo apresentado na figura 11, tem função bem definida que é a combinação dos pulsos PWM para que apareça nos Gate drives, pulsos de acordo com a modulação Tri-State, que é a uma meta do trabalho.

Figura 11. Foto do PLD no circuito do inversor



Fonte: Próprio Autor

Consta no PLD 20 pinos no total foi programado com 9 pinos de entrada e 7 pinos de saída, de maneira que a combinação dos pulsos simulasse a modulação Tri-State, por meio da dos pulsos advindos do controlador digital. Esta programação está apresentada abaixo, e foi desenvolvida pela empresa Wat:

```

Revision 02 ;
Designer Luigi ;
Company BATLAB ;
Assembly None ;
Location UNESP ;
Device gl6v8a ;

/* ***** INPUT PINS ***** */
PIN 1 = Dir; /* Direção do primário */
PIN 2 = Ad; /* D atrasado */
PIN 3 = A; /* D Pulso referente a modulação */
PIN 4 = B; /* Pulso referente a modulação de D2 */
PIN 5 = Bd; /* D2 atrasado */
PIN 6 = C; /* Pulso de Polaridade */
PIN 7 = Cd; /* C atrasado */
PIN 8 = Psepic; /* Pulsos do sepic */
PIN 9 = Sentido; /* Direção do sepic */

/* ***** OUTPUT PINS ***** */
PIN 19 = S5dir; /* S4 CC direto */
PIN 18 = S3; /* S3 roda livre */
PIN 17 = S2; /* S2 CA negativo */
PIN 16 = S1; /* S1 CA positivo */
PIN 15 = S4rev; /* CC Reverso */
PIN 14 = SS1dir; /* SEPIC Direto */
PIN 13 = SS2rev; /* SEPIC Reverso

S1 = (C # Cd) & (!A & Bd);
S2 = (!C # !Cd) & (!A & Bd);
S3 = (!B # !Bd);
S5dir = (A # Ad) & Dir;
S4rev = (A # Ad) & !Dir;
SS1dir = Psepic & Sentido;
SS2rev = Psepic & !Sentido;

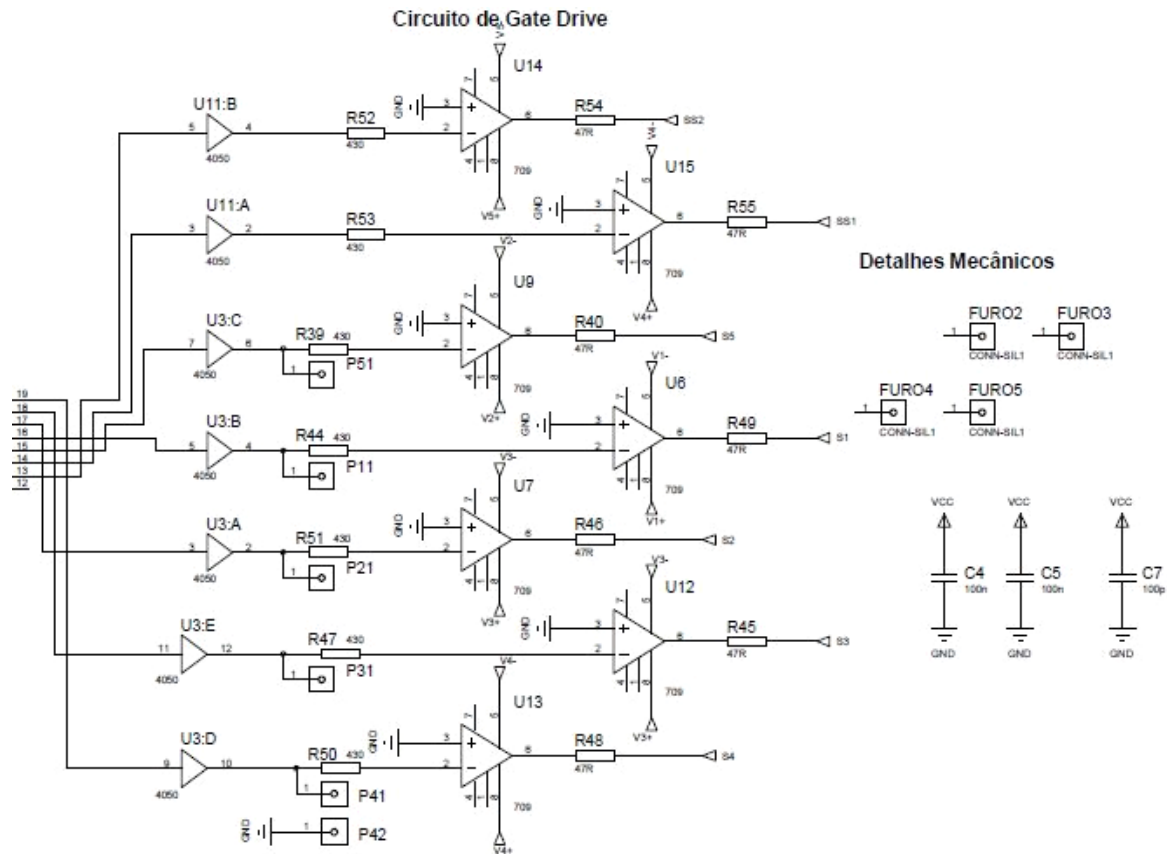
```

Da saída do PLD estão os pulsos que alimentam os circuitos de Gate Drives, que serão melhor explorados a seguir.

3.2.3 Circuito Gate Drive

O circuito de *Gate Drive* é a combinação das saídas do PLD, que será encaminhada para cada opto-acoplador do circuito, para que posteriormente seja enviado para as chaves da placa. Temos um total de 7 chaves, 5 para o inversor e duas para o SEPIC, logo temos 7 combinações feitas pelo PLD, que serão enviadas para um Opto acoplador HCPL3180 que alimentará uma das 7 chaves constantes no projeto do inversor, conforme o esquemático apresentado na figura 12, é desenvolvido e implementado pela empresa Wat.

Figura 12. Esquemático do circuito de Gate Drive



Fonte: Cortesia Wat adaptada

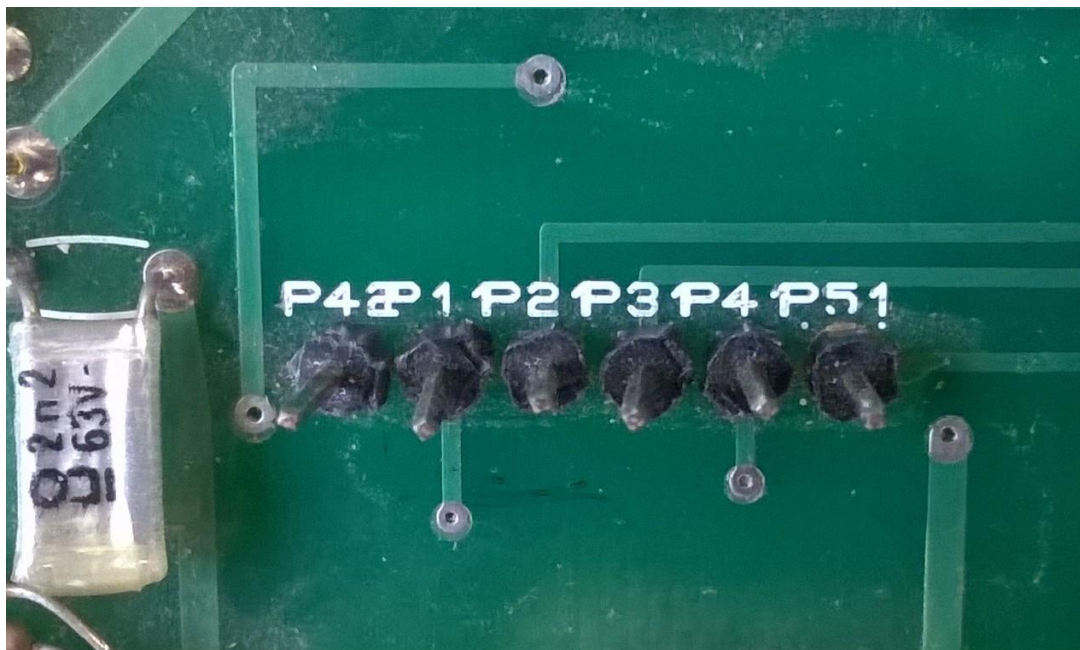
Para se ter acesso aos sinais que vão as chaves, que foram modulados pelo PLD, foi preciso colocar alguns pinos de acesso. A referência ficou com o pino de acesso P42, os demais acessos para medição que estão de acordo com a tabela 2. Esse artifício foi utilizado para realizar a visualização dos pulsos de PWM gerados por meio do dsPIC, via osciloscópio. A

figura 13 representa estes acessos implementados na placa do inversor, que se situam fisicamente na parte central inferior, cuja função foi descrita no parágrafo anterior.

Tabela 2. Tabela dos acessos ao sinais gerados

| Chave Controlada | Função | Acessado pelo pino |
|------------------|--------------------------|--------------------|
| Q1 | Estado ativo positivo | P11 |
| Q2 | Estado ativo negativo | P12 |
| Q3 | Roda livre | P31 |
| Q4 | Regeneração | P51 |
| Q5 | Estado de curto circuito | P41 |

Figura 13. Acesso ao sinais modulados



Fonte: Próprio autor

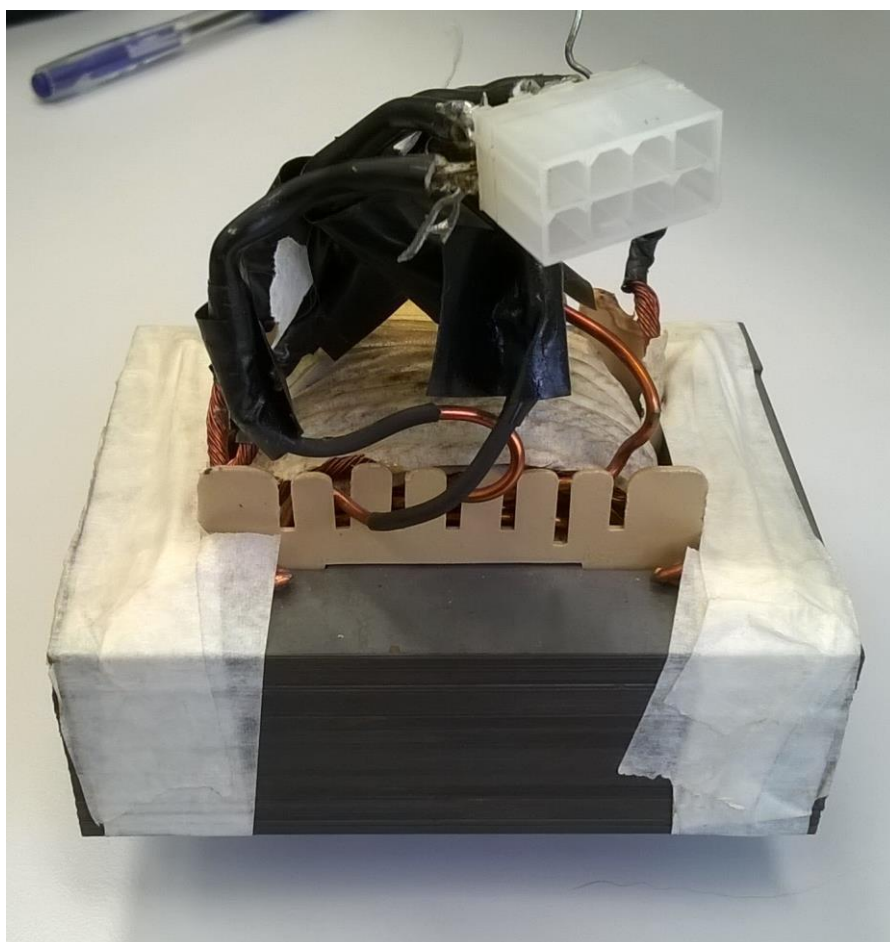
3.2.1 Circuito de Potência e Núcleo

Passadas as apresentações do circuito, eis que surge a vez de apresentar o núcleo, desenvolvido e confeccionado pela empresa Wat, onde será obtida o isolamento da topologia e

tensão desejada devido a relação de espiras, este elemento apresentado na figura 14, é peça fundamental para o bom desempenho do inversor.

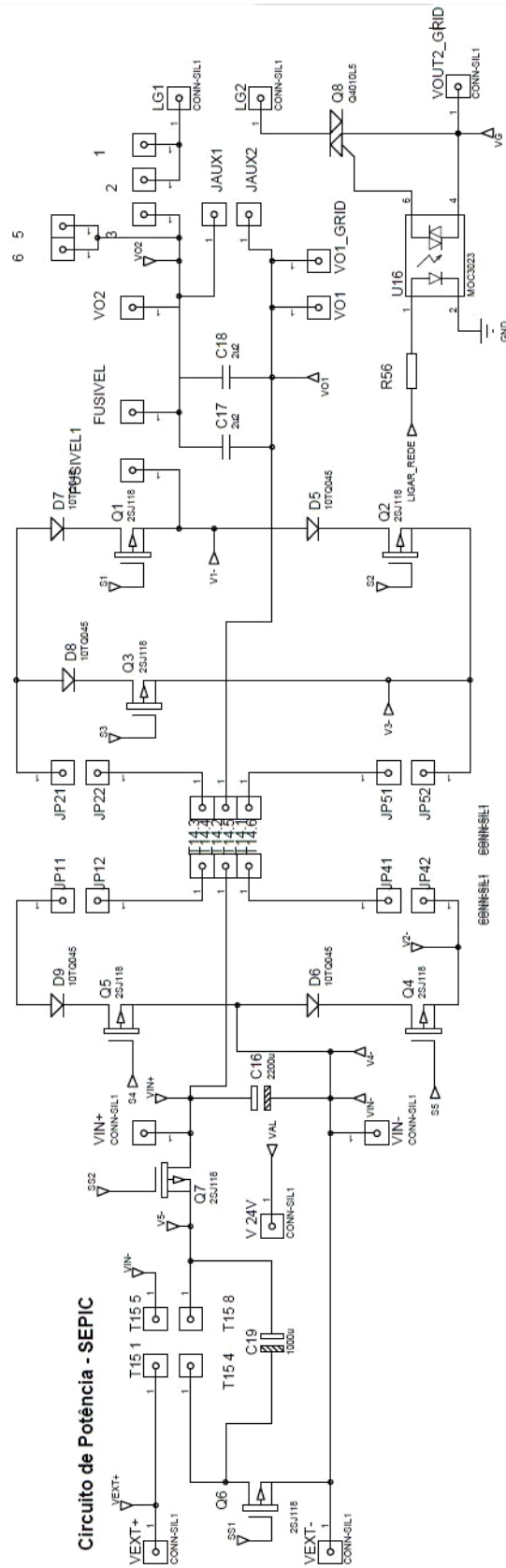
No esquemático apresentado pela figura 15, que por dimensões será posta na vertical, consta 5 chaves, sendo Q5 a chave do período de curto, Q4 atuante na regeneração, Q3 que funciona como um diodo Roda Livre, Q2 representando o estado ativo negativo (-) e pôr fim a chave Q1 que representa os estado ativo positivo (+). Por se tratar de uma topologia isolada, pode-se observar no centro da figura o local que será utilizado para colocar o transformador que será necessário para elevar a tensão, por meio da relação de espiras, para níveis adequados de operação.

Figura 14. Núcleo



Fonte: Próprio Autor

Figura 15. Esquemático do circuito de Potência



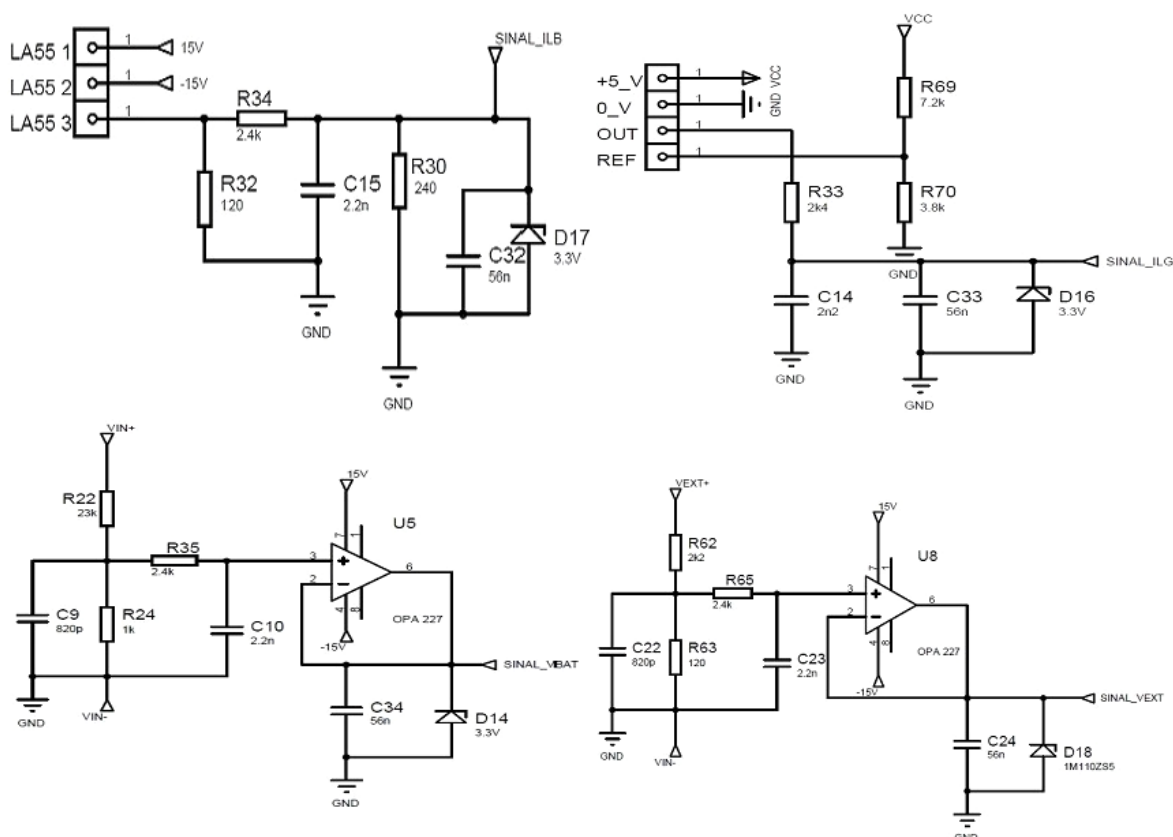
Fonte: Cortesia Wat

3.2.1 Circuito de Sensoriamento e Condicionamento de Sinais

No dsPIC existem entradas analógicas, mas para que essas entradas sejam medidas corretamente devemos observar o nível de tensão que a entrada aceita, neste caso específico o nível de tensão é de 3,3 volts. Logo para que o dispositivo de aquisição de dados efetue a medição de forma eficaz e exata, devemos condicionar o sinal de maneira que obedeça os limites de tensão do dispositivo.

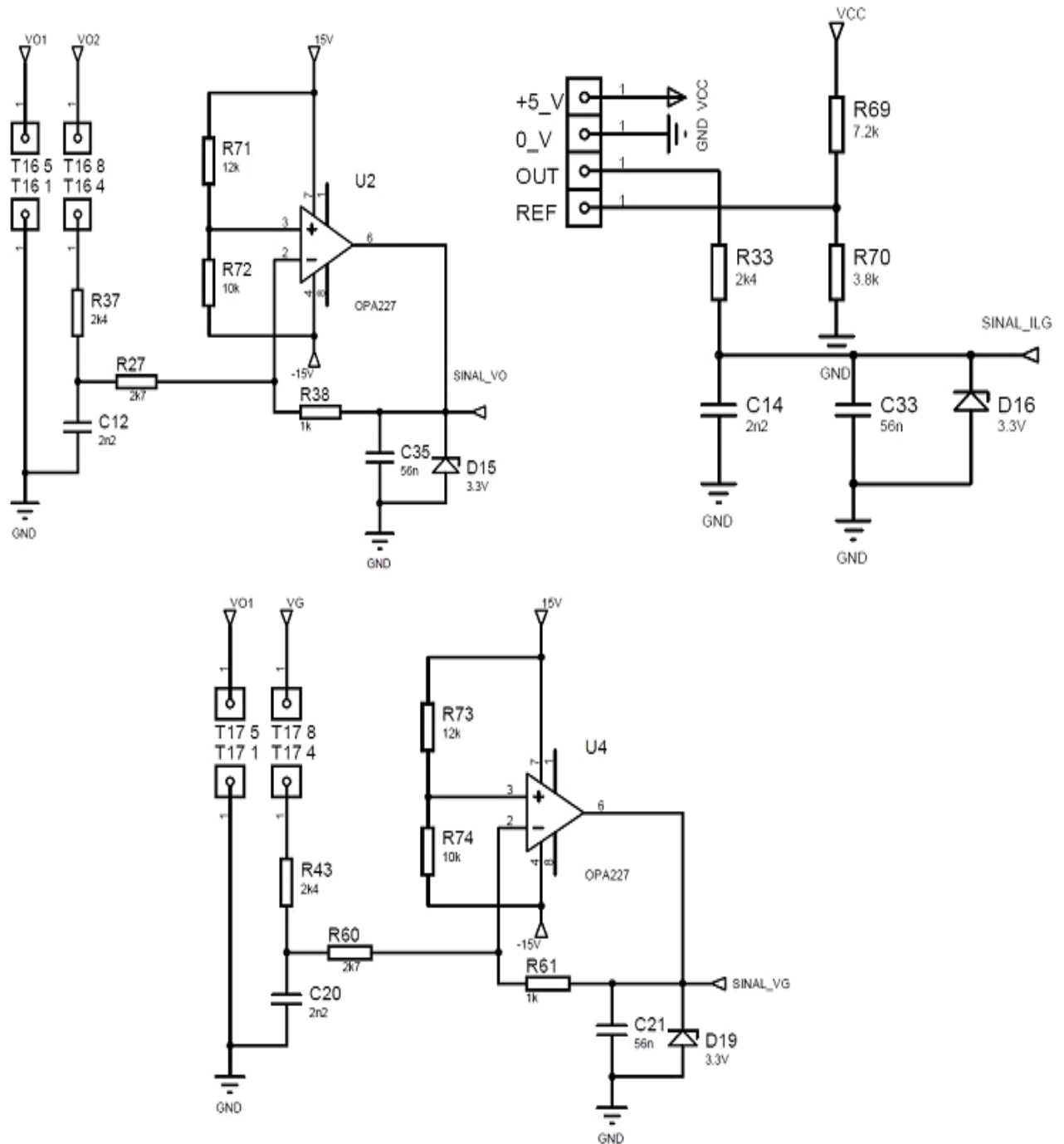
Desta forma, a empresa Wat esquematizou os circuitos de condicionamento para que as tensões dos sinais medidos, não ultrapassassem os limites previstos dos dispositivos, que devido esta, poderá até causar um dano irreversível ao dsPIC. Sendo assim, os esquemáticos da figura 16 e 17 abaixo, representam os esquemas dos sinais coletados que podem ser a corrente no núcleo, a corrente da rede elétrica, a tensão nas baterias, a tensão externa, a tensão na saída do inversor, tensão da rede elétrica e corrente das baterias, que no presente projeto foram chamado respectivamente de Sinal_ILB, Sinal_ILG, Sinal_VBAT, Sinal_VEXT, Sinal_VO, Sinal_VG e Sinal_IBAT.

Figura 16. Circuito de condicionamento da Corrente no núcleo, da rede, tensão na Bateria e Externa



Fonte: Cortesia Wat adaptada

Figura 17. Circuito de condicionamento da tensão de saída, da rede e da corrente da rede



Fonte: Cortesia Wat adaptada

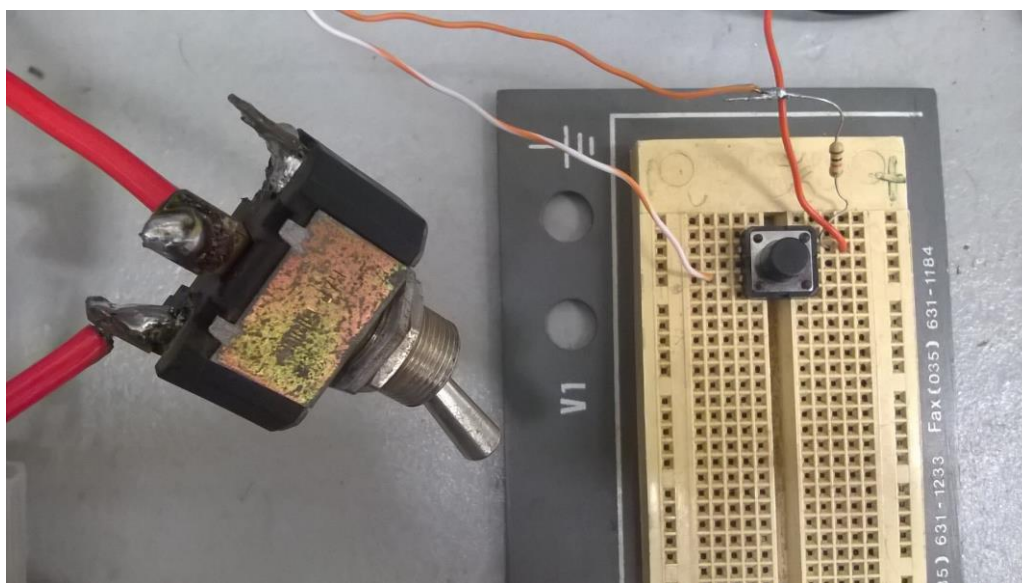
3.3 Botão e Chave para Acionamento

Para o acionamento prático da planta foi implementado um botão para ligar e desligar o inversor, como também uma chave do tipo “tic-tac” de três posições para acionamento da fonte de alimentação da parte de potência do inversor. A implementação do botão se deu pela

praticidade e também para que o inversor tivesse um acionamento em rampa, pois na programação do dsPIC quando se dá o acionamento do botão é contado um tempo e depois acionado uma rampa de partida, o que é natural para acionamento de conversores. Isso evita pico de corrente e tensão que podem ocasionar alguns danos ao conversor.

Para desenvolver o código do botão foi utilizado uma porta lógica PORTGbits.RG12 conectada ao pino 33 do conector da placa e funciona de maneira simples. O dsPIC monitora a porta digital estiver em “1” binário, ou seja, na arquitetura do dsPIC 3,3 volts, nada acontece. Já na situação em que a porta estiver em “0” binário, ou seja com tensão zero volts, contasse um tempinho para evitar um transitório e após este o conversor é acionado em rampa. A figura 18 abaixo representa a implementação tanto da chave “tic-tac” quanto do botão.

Figura 18. Implementação da chave “tic-tac” e botão

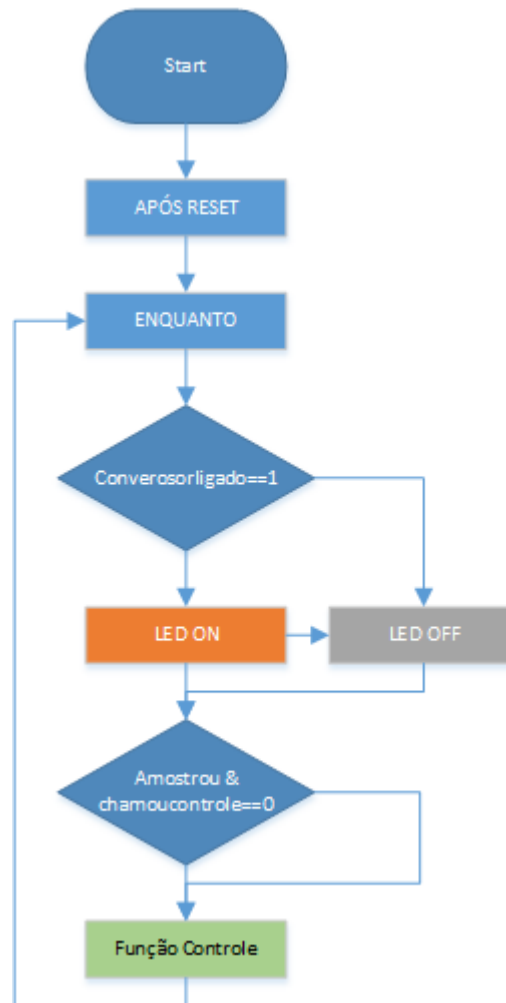


Fonte: Próprio Autor

3.4 O Algoritmo

O algoritmo foi simplificado em fluxograma para melhor visualização do mesmo, as decisões do algoritmo semelham a uma máquina de estados finitos programada em loop contínuo, pois se trata de uma implementação digital para o fim de controlar o inversor via dsPIC, onde exige o fechamento da malha via controlador digital. A figura 19 trata-se deste diagrama sendo que os blocos na cor azul significam funções simples, como uma AND por exemplo.

Figura 19. Fluxograma Main

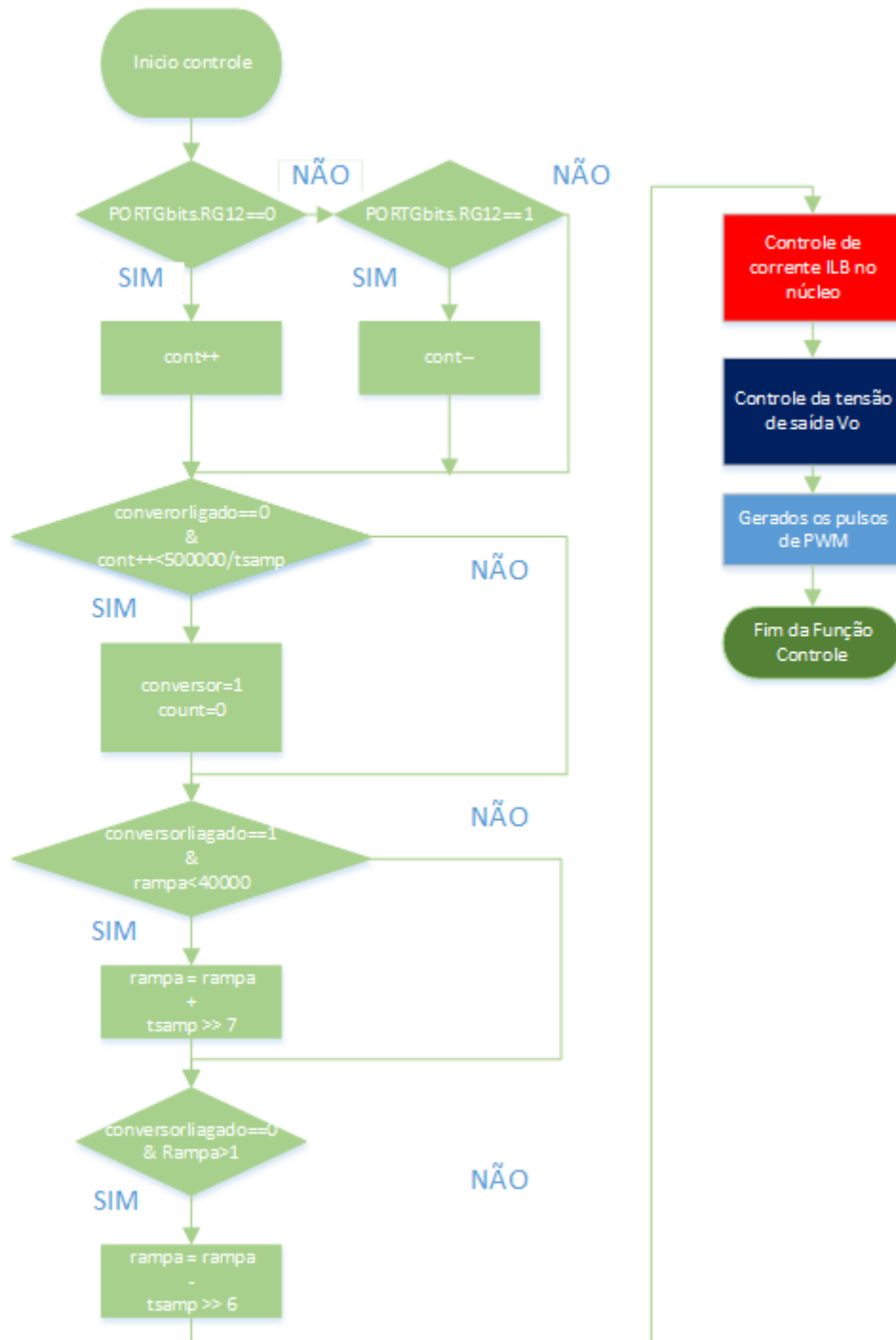


Fonte: Próprio Autor

A função do bloco LED é indicar se o conversor está ligado ou desligado, se o conversor estiver ligado o LED está aceso, caso contrário o LED estará apagado. Tem mais um caso a ser explorado, que é o LED piscando. Se o mesmo estiver piscando, indica que há um problema na conexão do botão e que o mesmo não está ligado ao pino certo do conector da placa.

Podemos observar no diagrama acima um bloco chamado de “Função Controle” que motivos de espaço foi simplificado, cujo seu objetivo é descrever o funcionamento do algoritmo de controle que será mostrado na figura 20, e ele é peça elementar para o bom funcionamento do inversor.

Figura 20. Fluxograma Controle

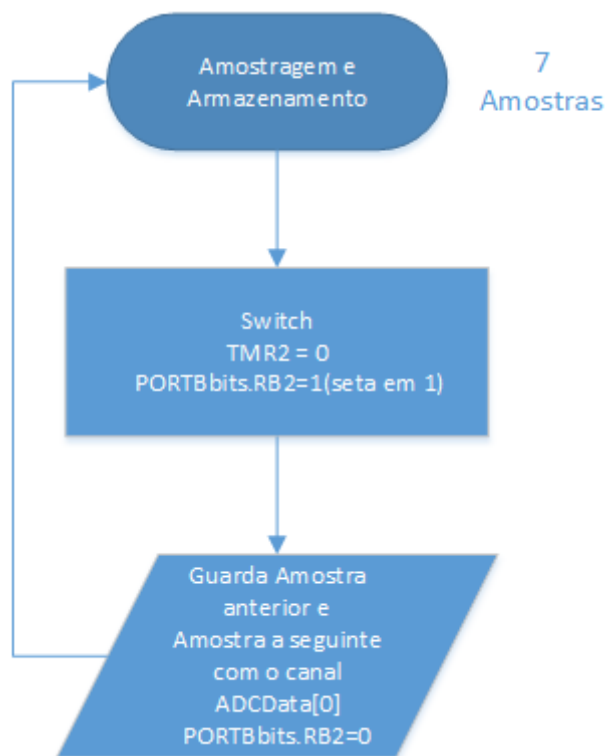


Fonte: Próprio Autor

No fluxograma apresentado acima, exige duas malhas a ser controladas que é a malha de corrente ILB no núcleo e a tensão V_o de saída, como também as contas que geram os pulsos para serem enviados as chaves caracterizando o controle sobre as mesmas. O diagrama acima é uma esboço do algoritmo apresentado no Apêndice D.

O processo de amostragem mostrado na figura 21 abaixo é o diagrama de fluxo, cujo objetivo é mostrar o funcionamento do algoritmo da amostragem e armazenamento. Já o código completo deste processo poderá ser melhor explorado no Apêndice C.

Figura 21. Processo de amostragem e armazenamento



Fonte: Próprio Autor

O processo descrito no diagrama anterior visa a coleta de amostra e armazenagem dos sinais para a realização do controle. Como visto acima, há 7 sinais para sensoriar e condicionar. Desta forma, esse processo se repete 7 vezes parando ao final de cada amostra. Com relação à armazenagem, no primeiro ciclo se faz a amostra pelo ADC1 canal zero e só no próximo ciclo haverá o armazenamento e será feita outra amostra.

O processo de amostragem e armazenamento só será encerrado após a passagem de 7 vezes pela função SWITCH e SampleRead=1. Para monitorar o processo de amostragem configurou-se a porta lógica PORTBits.RB2=1 no começo e PORTBits.RB2=0 no final. Para que o pulso indique que a função SWITCH esteja funcionando corretamente e também para que possa ser controlado o tempo de duração para fazer todas as amostras e o tempo para se fazer o controle da planta.

4 SIMULAÇÃO E ANÁLISE DE RESULTADOS

4.1 Pulsos de Modulação com Sobreposição

A modulação e a sobreposição dos pulsos são uma condições necessárias para o funcionamento deste inversor. Como o inversor se trata de uma fonte de corrente, é preciso que se tenha uma chave sempre em condução a fim de evitar que se tenha uma elevação na tensão, que pode causa a queima de uma das chaves devido a um surto repentino de tensão. A figura 22 exibe o resultado dos pulsos de modulação com sobreposição.

Figura 22. Modulação com sobreposição dos pulsos



Fonte: Próprio Autor

O pulso na cor amarela corresponde ao estado de curto do inversor que corresponde a chave Q5, já o na cor verde corresponde ao estado de roda livre que está ligado a chave Q3 e o na cor roxa corresponde ao estado ativo negativo correspondente a chave Q2.

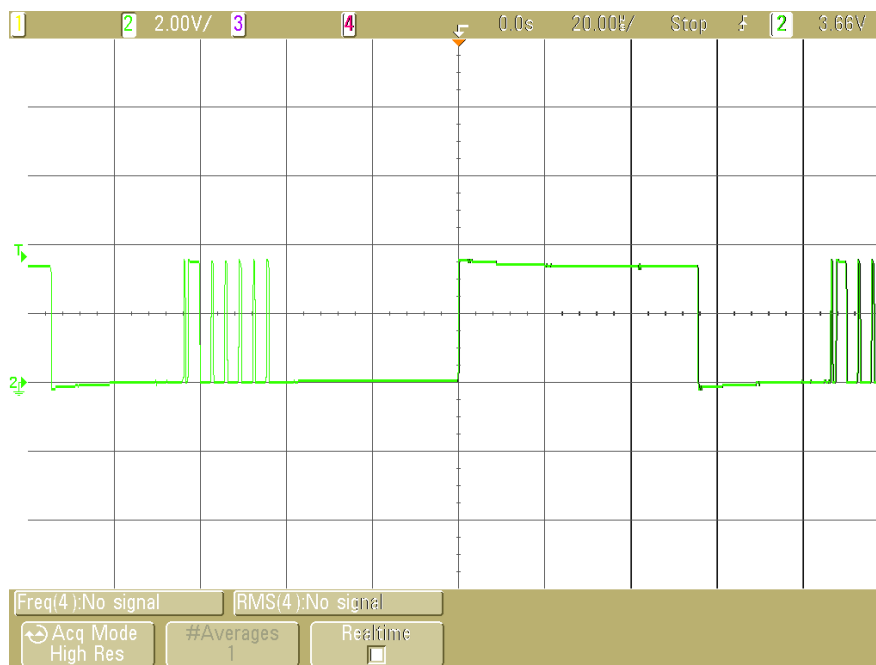
Esta sobreposição se deve ao circuito de atraso que foi mostrado na figura 23??. Observe que antes da Q3 fechar, Q2 já está ativa e antes do Q2 fechar Q5 está ativo, para esta situação.

4.2 Pulsos de amostragem e controle

Como artifício de visualização, foi configurado uma porta digital como meio de medir e sensoriar as leituras do ADC1. A figura 23 representam este processo, onde os sete pulsos menores consecutivos na figura representam as amostras do ADC1 e a Função SWITCH em operação com interrupção de DMA, já o pulso maior representa o tempo que se leva para fazer

os algoritmos depois da amostragem e da chamada do controle, e mandar o resultado para a saída do PWM.

Figura 23. Pulsos de Amostragem e Controle



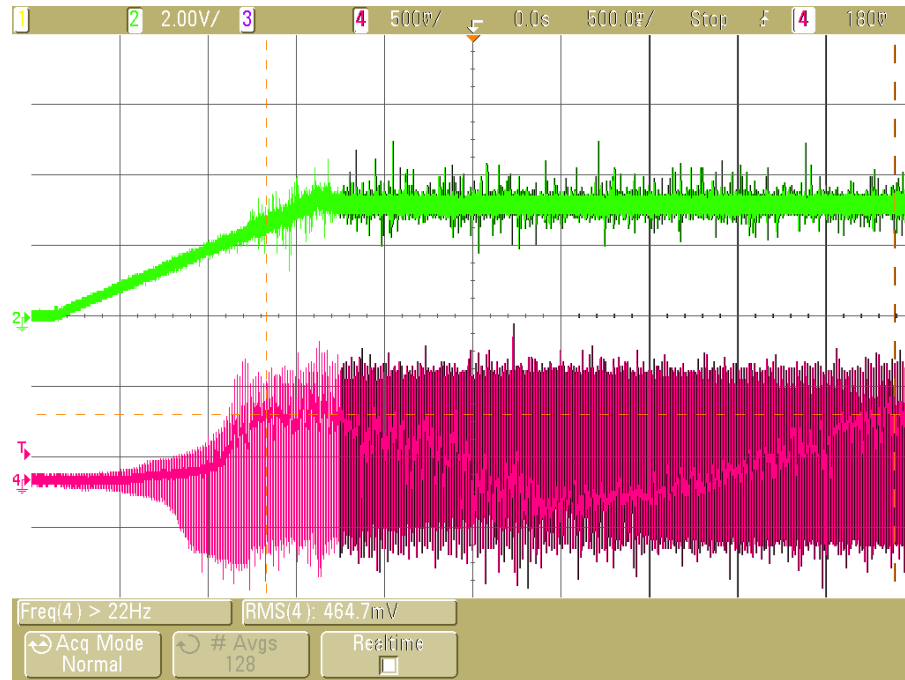
Fonte: Próprio Autor

4.3 Rampa de partida e onda senoidal

A figura 24, representa a rampa de partida e a tensão senoidal. Na cor verde está sensoriada a corrente do inversor que está na escala de dois volts por divisão e está indicando aproximadamente 3 volts, já na cor vermelha observamos o nascimento da onda senoidal medida a partir da saída em seus primeiros, para medir essa onda, foi preciso usar uma ponteira diferencial da Agilent na escala 20:1, um osciloscópio na escala de tempo de 100ms por divisão e com escala de tensão de 500m volts por divisão, pois desta forma foi possível a visualização da onda senoidal formada no núcleo do indutor de acoplamento utilizado na topologia do inversor Flyback.

Observe que em seus primeiros ciclos enquanto a corrente vai aumentando, a amplitude da onda senoidal vai aumentando proporcionalmente até chegar em regime permanente com frequência de operação.

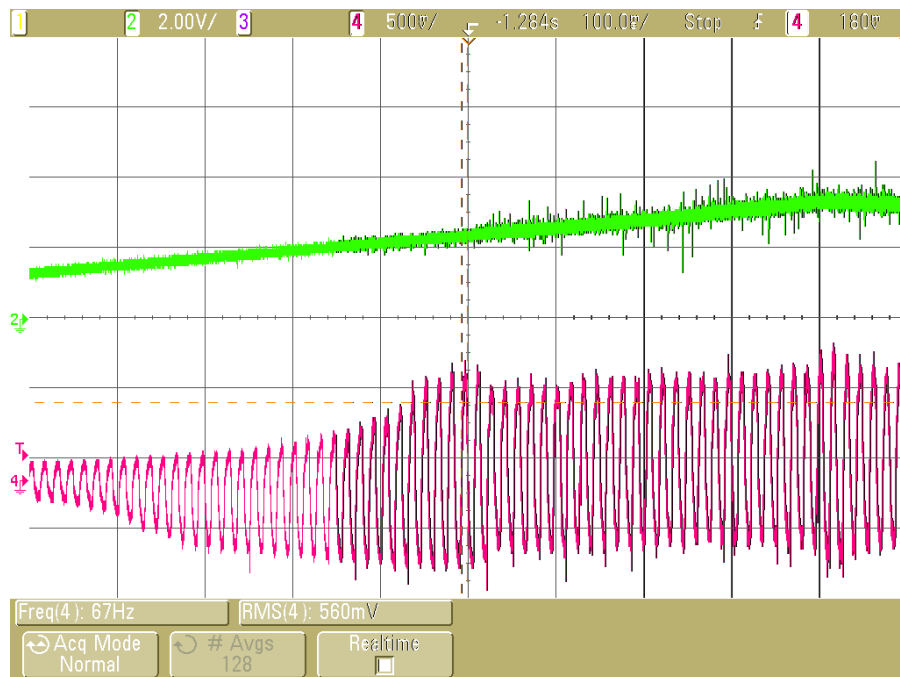
Figura 24. A Rampa de partida de fluxo no núcleo e a tensão na saída no inversor



Fonte: Próprio Autor

Agora observe na figura 25, que é o meio do processo de partida, em que vai se corrigindo a frequência da onda por meio da ação do controlador.

Figura 25. Frequência 67 Hz

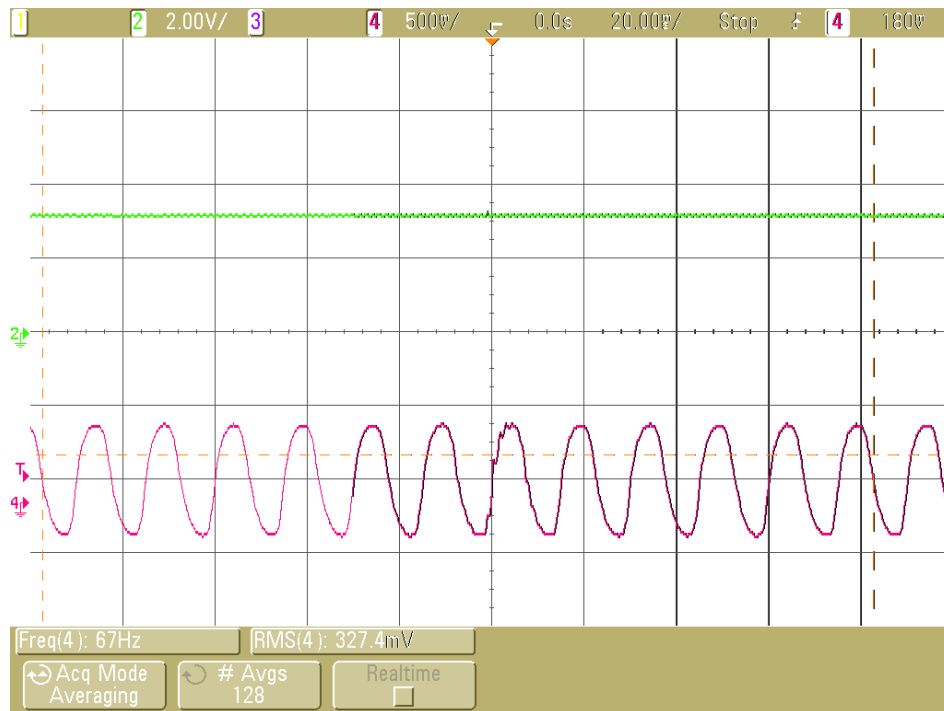


Fonte: Próprio Autor

4.4 Onda senoidal gerada e leitura do sensor de corrente

Para escala de 20:1 utilizada para medir tensão no núcleo e com um escala de 500 m volts por divisão, temos que a onda tem 22 volts pico-a-pico e com frequência de 67 Hz. E isso é mostrado na figura 26 abaixo.

Figura 26. Corrente no Sensor e Onda senoidal



Fonte: Próprio Autor

5. CONCLUSÃO

Apresentou-se neste trabalho os resultados da implementação do controle digital em um inversor Flyback com modulação Tri-State, utilizando controlador Proporcional Integral para o fluxo no núcleo e Proporcional Ressonante para a tensão de saída V_o .

Confirmam-se com as simulações e resultados do controlador que foi implementado de forma satisfatória. Partindo-se da topologia Flyback, foi testado o controlador proporcional integral e o ressonante com carga baixa.

A frequência de chaveamento do conversor foi projetada para 25 kHz. Com relação à frequência de amostragem e controle foi realizada inicialmente em 3,125 kHz (correspondente a 8 períodos de 25 kHz) mantendo o sincronismo entre o chaveamento e a amostragem. Porém, esta frequência se mostrou insuficiente para o controle proposto e foi necessário aumentar para 7 kHz, porém foi possível este aumento somente de forma assíncrona ao chaveamento. Para uma frequência maior do que 7 kHz será necessária a otimização do controle e da amostragem que ficaram no limite de utilização do período disponível. Contudo, 7 kHz foram suficiente para reduzir as distorções na forma de onda da tensão de saída do inversor com a estratégia de controle implementada.

A qualidade das respostas obtidas é semelhante às encontradas em [8], não sendo possível afirmar ainda se atenderá aplicações com conexão a rede devido a capacidade do controlador dsPIC. A resposta do controle poderá ser melhorada por meio de outro microcontrolador digital de melhor desempenho ou pela otimização do algoritmo de controle. Esta busca por melhorar resposta do conversor de potência não foi explorada por estar além do escopo deste trabalho.

Conclui-se que o objetivo deste trabalho foi alcançado, com a implementação do controle digital no inversor protótipo, que anteriormente era controlado dependente de uma plataforma de controle dSPACE em um PC. Os hardwares disponíveis foram estudados, analisados e corrigidos quando necessário para a realização do controle. Devido à sua descrição e documentação preparada, este trabalho poderá contribuir em trabalho futuros que utilizem o mesmo protótipo do inversor, ou que utilizem a mesma placa controladora com dsPIC, que é independente do inversor. Entretanto, a principal contribuição deste trabalho está na preparação de um projeto de código (firmware) que poderá servir de modelo para a melhoria do controle deste mesmo protótipo ou para agilizar o controle em outras aplicações com o mesmo módulo.

REFERÊNCIAS

- [1] Mohan, N., Undeland, T.M. and Robins, W. P., "Power Electronics: Converters, Applications and Design". John Wiley & Sons, Inc., 2nd edition, 1995
- [2] Erickson, Robert W. Fundamentals of power electronics. Norwell, Massachusetts: Kluwer, 1999.
- [3] Godoy, R. B.; Brito, M. A. G.; Galotto Junior, L.; Garcia, R. C.; Kimpara, M. L. M.; Canesin, C. A. Tri-state space vector modulation for three-phase integrated inverters. In: Brazilian Power Electronics Conference - COBEP, 11., 2011, Natal. Conference Natal: IEEE, 2011. p. 911 – 917
- [4] Brito, M.A.G, “ Inversores integrados monofásicos aplicados em sistemas fotovoltaicos: Técnicas de obtenção para MPPT, detecção e proteção de ilhamento, sincronização e paralelismo com conexão à rede de Distribuição de energia elétrica, ” Ilha Solteira, 2013.
- [5] Braga, F. N. , “Controle digital de inversores monofásicos com realimentação de estados internos, ” Uberlandia, março de 2009.
- [6] B R R. Sable, D.M.; Cho, “Use of leading-edge modulation to transform boost and flyback converter into minimum-phase-zero systems,” IEEE Trans. Power Electron, vol 6, no. 4, pp. 704-711, 1991
- [7] P. C. Loh, F. Blaabjerg, C. P. Wong, and P. C. Tan, “Tri-state current source inverter with improved dynamic performance,” *IEEE Trans. Power Electron.*, vol. 23, no. 4, pp. 1631–1640, July 2008.
- [8] Galotto Junior, L., “ Inversores integrados monofásicos aplicados em sistemas fotovoltaicos com conexão à rede de Distribuição de energia elétrica, ” Ilha Solteira, 2011.
- [9] P. C. Loh¹, F. Blaabjerg, C. P. Wong¹ and P. C. Tan, “Tri-State Current Source Inverter

with Improved Dynamic Performance, Vol. 3 No. 1, pp. 2812 - 2817, 2008.

[10] L. Galotto, M. Brito, L. Sampaio, and C. Canesin, “Integrated inverter topologies for low power photovoltaic systems.” in *Proc. INDUS- CON’10*, Sao Paulo, June 9-11 2010, p. 6.

[11] Vilalba Onikuza, Y., M., “Análise e modelagem de conversores chaveados aplicada a um inversor Flyback, ” Campo Grande, setembro de 2013.

[12] Microchip Datasheet dsPIC 33FJ256GP710A disponível em:

<http://ww1.microchip.com/downloads/en/DeviceDoc/70593d.pdf> , Acesso em: 15 de Junho de 2015.

APÊNDICE A – CONFIGURANDO TIMERS

```
#include "p33FJ256GP710.h"

//Resolução de 6,4us por incremento de timer.
//Total 6,4us * 65536 = 0,4194s (ou 2,38Hz acima)
//Resolução de 1,6us por incremento de timer.
//Total 1,6us * 65536 = 0,1048s (ou 9,53Hz acima)
//Resolução de 200ns por incremento de timer.
//Total 200ns * 65536 = 13,10ms (ou 76.29Hz acima)
//Resolução de 25ns por incremento de timer.
//Total 25ns * 65536 = 1,63ms (ou 610Hz acima)
//freqSaida 17
//resolucao 6.4E-6

#define periodoSaida 1600//((1/freqSaida)/resolucao)/2 1600=25kHz 1000=40kHz
#define periodoAmostragem 17//17//((1/freqSaida/numPontosDMARAM/resolucao)/2)-1
// => (1/17Hz/256/6.4E-6/2)-1=17
//Obs.: a divisão por 2 é necessária pois cada ponto na RAM está sendo amostrado
// 2 vezes e 512 pontos na realizade representam 256 pontos

/// Initialise the timers such that TMR 2 will interrupt at 400kHz and will be
// used to generate the 25kHz output waveform.          ///
/// TMR3 is set to generate a 100kHz interrupt used for the ADC start conversion.
////////////////////////////////////

void Init_Timers(void)
{
    //Configurando o Timer 2 para geração de sinal de saída:.....
    T2CONbits.TON = 0; //Desabilita o timer 2.
    //T2CON = 0b00000000000110000; //Padrão: configurando o Timer2 para gerar
    //interrupções a cada 2,5us, ou seja uma frequência de amostragem de 400KHz,
    //ou 16 pontos por ciclo de onda de 25KHz.
    T2CON = 0b00000000000000000; //Padrão: configurando o Timer2 para gerar
    //interrupções a cada 2,5us, ou seja uma frequência de amostragem de 400KHz,
    // ou 16 pontos por ciclo de onda de 25KHz.
    //01 = 1:8

```



```

IFS0bits.T2IF = 0;//Limpando o bit 7 (T2IF) do registrador IFS0
//(Timer2 Interrupt Flag Status bit).
IPC1bits.T2IP = 5;//Definindo prioridade 5 para a interrupção do Timer 2.
PR2 = periodoSaida;//2000: Armazenando o valor 2000 no registrador PR2,
//indicando que depois de 2000 incrementos ( $2000 \cdot T_{cy} \cdot 1 = 2000 \cdot 0,025\mu s \cdot 1 = 50\mu s$ )
//ocorrerá uma interrupção de Timer 2
TMR2 = 0;//Inicializando com 0 o Timer 2
IEC0bits.T2IE = 1;//Habilitando a interrupção do Timer 2 gerada devido a
//Bits TCKPS<1:0> (Timer2 Input Clock Prescale Select bits) deixados com
//valores padrões 00b,indicando relação 1:1 de prescale do no clock do timer
//Realizando configurações finais dos timers:.....
TMR2 = 0;
IPC1bits.T2IP = 0x01;//Define o nível de prioridade da interrupção do Timer2
IFS0bits.T2IF = 0; //Limpando o Flag de Interrupção do Timer 2.
IEC0bits.T2IE = 1; //Habilitando a interrupção do timer 2.
T2CONbits.TON = 1; //Iniciando o Timer 2.
//Configurando o Timer 6 para geração de sinal de saída:.....
T6CONbits.TON=0;
T6CON = 0b00000000000010000;//0b00000000000010000
IFS2bits.T6IF = 0;
IPC11bits.T6IP = 5;
PR6 = 760;//periodoSaida;
TMR6 = 0;
IEC2bits.T6IE = 1;
IPC11bits.T6IP = 0x02;
IFS2bits.T6IF = 0;
IEC2bits.T6IE = 1;
T6CONbits.TON = 1;

} //Fim da função "Init_Timers"

```

APÊNDICE B – CONFIGURAÇÃO DO PWM

```

void Init_PWMs(void)
{
//Inicializa a saída OC1/RD0 como saída PWM://SA0=Pin23 PWM D1
    OC1CONbits.OCM = 0b000; // Disable Output Compare Module
    OC1R = 0;                // Write the duty cycle for the first PWM pulse
    OC1RS = 0;               // Write the duty cycle for the second PWM pulse
    OC1CONbits.OCTSEL = 0;   // Select Timer 2 as output compare time base
    OC1R = 0;                // Load the Compare Register Value
    OC1CONbits.OCM = 0b110; // Select the Output Compare mode
    //Inicializa a saída OC2/RD1 como saída PWM://SA1=Pin21 PWM D1+D2
    OC2CONbits.OCM = 0b000; // Disable Output Compare Module
    OC2R = 0;                // Write the duty cycle for the first PWM pulse
    OC2RS = 0;               // Write the duty cycle for the second PWM pulse
    OC2CONbits.OCTSEL = 0;   // Select Timer 2 as output compare time base
    OC2R = 0;                // Load the Compare Register Value
    OC2CONbits.OCM = 0b110; // Select the Output Compare mode
    //Inicializa a saída OC3/RD2 como saída PWM://SA2=Pin19 PWM POL
    OC3CONbits.OCM = 0b000; // Disable Output Compare Module
    OC3R = 0;                // Write the duty cycle for the first PWM pulse
    OC3RS = 1600;            // Write the duty cycle for the second PWM pulse
    OC3CONbits.OCTSEL = 0;   // Select Timer 2 as output compare time base
    OC3R = 0;                // Load the Compare Register Value
    OC3CONbits.OCM = 0b110; // Select the Output Compare mode
    //Inicializa a saída OC3/RD2 como saída PWM:SA3=Pin17 PWM SEPIC
    OC4CONbits.OCM = 0b110; // Disable Output Compare Module
    OC4R = 0;                // Write the duty cycle for the first PWM pulse
    OC4RS = 0;               // Write the duty cycle for the second PWM pulse
    OC4CONbits.OCTSEL = 0;   // Select Timer 2 as output compare time base
    OC4R = 0;                // Load the Compare Register Value
    OC4CONbits.OCM = 0b110; // Select the Output Compare mode
    //Inicializa a saída OC3/RD2 como saída PWM:SA5=Pin22 PWM SEPIC
    OC6CONbits.OCM = 0b000; // Disable Output Compare Module
    OC6R = 0;                // Write the duty cycle for the first PWM pulse
    OC6RS = 0;               // Write the duty cycle for the second PWM pulse
    OC6CONbits.OCTSEL = 0;   // Select Timer 2 as output compare time base
    OC6R = 0;                // Load the Compare Register Value
    OC6CONbits.OCM = 0b110; // Select the Output Compare mode
    //PWM:SA4=Pin24 PWM REG. MOD. NA PLACA
    OC5CONbits.OCM = 0b000; // Disable Output Compare Module
    OC5R = 0;                // Write the duty cycle for the first PWM pulse
    OC5RS = 0;               // Write the duty cycle for the second PWM pulse
    OC5CONbits.OCTSEL = 0;   // Select Timer 2 as output compare time base
    OC5R = 0;                // Load the Compare Register Value
    OC5CONbits.OCM = 0b110; // Select the Output Compare mode
    Init_Timers();//Configurando o Timer2 para gerar sinal de saída e o Timer3 <=ADC1
    // LockIn sobre os dados na DMA RAM.
}

```

APÊNDICE C – FUNÇÃO SWITCH E DMA

```

#include <dsp.h>

/// Protótipos da funções chamadas pela ISR                                     ///
////////////////////////////////////

fractional* ptCopySig;
int inicio;//Variável guardando o índice de início atual do buffer de ADC
int canal = 31;
unsigned int chamoucontrole=0;
////////////////////////////////////

/// ISR de atendimento de interrupções de DMA                                ///
////////////////////////////////////

void __attribute__((__interrupt__,__no_auto_psv)) _DMA0Interrupt( void )
{ //Esta interrupção é chamada toda vez que o DMAC copia toda uma página
  //dados do ADC para a DMA RAM.

    TMR2 = 0;//Sincroniza a aquisição com a geração do sinal de saída
    PORTBbits.RB2=1;

    //Ordem: AN9,10,11,12,29,30,31
    switch(canal){
    case 9: Sinal_ILG=ADCDData[0];//AN31
        AD1CHS0=0b00000000000001001;//AN9
        AD1CON1bits.SAMP = 0;//Inicia uma nova conv. e finaliza a amostragem anterior.
        PORTBbits.RB2=0;
        canal=10;
        break;
    case 10:Sinal_IBAT=ADCDData[0];//AN9
        AD1CHS0=0b00000000000001010;//AN10
        AD1CON1bits.SAMP = 0;//Inicia uma nova conv. e finaliza a amostragem anterior.
        PORTBbits.RB2=0;
        canal=11;
        break;
    case 11:Sinal_VG = ADCData[0];//AN10
        AD1CHS0=0b00000000000001011;//AN11
        canal=12;
        PORTBbits.RB2=0;
    }
}

```

```

break;

case 12:Sinal_VO = ADCData[0];//AN11
AD1CHS0=0b0000000000001100;//AN12
AD1CON1bits.SAMP = 0;//Inicia uma nova conv. e finaliza a amostragem anterior.
PORTBbits.RB2=0;
canal=29;
break;

case 29:Sinal_ILB=ADCData[0];//AN12
AD1CHS0=0b0000000000011101;//AN29
AD1CON1bits.SAMP = 0;//Inicia uma nova conve. e finaliza a amostragem anterior.
PORTBbits.RB2=0;
canal=30;
break;

case 30:Sinal_VEXT=ADCData[0];//AN29
AD1CHS0=0b0000000000011110;//AN30
AD1CON1bits.SAMP = 0;//Inicia uma nova conv. e finaliza a amostragem anterior.
PORTBbits.RB2=0;
canal=31;
break;

case 31:Sinal_VBAT=ADCData[0];//AN30
AD1CHS0=0b0000000000011111;//AN31
AD1CON1bits.SAMP = 0;//Inicia uma nova conv. e finaliza a amostragem anterior.
PORTBbits.RB2=0;
canal=9;
//break;
};

chamoucontrole=0;

//Atualizando flags indicadores do estado atual do sistema:

SampleReady = 1;//Indica que novos resultados de cálculos estão prontos.

IFS0bits.DMA0IF = 0;//Limpa o flag de interrupção de DMA.

} //Fim da função "_DMA0Interrupt"

```

APÊNDICE D – MAIN E CONTROLE

```

/*****
*   Autores:      Andre M Soares e Rene A Capitania
*
*               Luigi Galoto Junior e Marcel Pascoski
*   Companhia:    Hiperon Engenharia Ltda/ WAT
*   Arquivo:      main.c
*   Date:         07/2006
*   File Version: 1.00
*****/

//#include <p33Fxxxx.h>
#include "p33FJ256GP710.h"
#include "common.h"//Define DMA_BUFFER_SIZE=512 bytes
#include "init_ADC.h"//Rotinas de inicializa??o de ADC.
#include "init_timers.h"//Rotinas de inicializa??o dos timers internos.
#include "init_PWM.h"
#include "delay.h"//Biblioteca de rotinas de atraso
#include "traps.h"//Biblioteca de manipula??o de exce??es de hardware
_FGS(GWRP_OFF & GSS_OFF);
_FOSCSEL(FNOSC_PRIPLL);
_FOSC(FCKSM_CSDCMD & OSCIOFNC_OFF & POSCMD_XT);
_FWDT(FWDTEN_OFF);

////////////////////////////////////

///  Vari?veis Globais do programa                                     ///
////////////////////////////////////

int Sinal_IBAT=0;//AN9
int Sinal_VG=0;//AN10
int Sinal_VO=0;//AN11
int Sinal_ILB=0;//AN12
int Sinal_VEXT=0;//AN29
int Sinal_VBAT=0;//AN30
int Sinal_ILG=0;//AN31

```

```

#include "isr_DMAC.h"//Rotina de atendimento de interrup??o de DMA.
#include "isr_timers.h"//Rotina de atendimento de interrup dos timers internos.

//Vari?veis para c?lculos matem?ticos de fase e quadratura:.....
unsigned int ADCData[DMA_BUFFER_SIZE * 2] __attribute__((space(dma)));
// Buffer localizado na DPRAM para manter os dados de ADC entrantes.
// Como o modo Ping-Pong foi selecionado, o tamanho total do buffer ?
//"DMA_BUFFER_SIZE * 2".
//Flag indicando que dados v?lidos foram colocados nas vari?veis "fIn" e "fQn".
volatile unsigned short SampleReady;

//Placa de bot?es:
unsigned int estBotao=0;//,evBotao;
int botaoliga= 1;

#define botaoliga PORTGbits.RG12

////////////////////////////////////
///  Prot?tipos de fun??es do programa                                     ///
////////////////////////////////////
void aposReset(void);
void controle(void);

////////////////////////////////////
///  Fun??o Main                                                         ///
///
////////////////////////////////////
int main(void)
{
    //Realizando inicializa??es globais do programa:.....
    aposReset();
    //Entrando no la?o de execu??o cont?nua do programa:.....
    while(1)
    {int var;

```

```

////////////////////////////////////
////////LED INDICANDO QUE A VARI?VEL conversorligado esta comutando////////
if(conversorligado==1){PORTBbits.RB3=1;}//IO11//PIN46//LED2
else          {PORTBbits.RB3=0;}
////////////////////////////////////

// Chamar o controle
if ((SampleReady==1)// & (chamoucontrole==0))
{
    if(chamoucontrole==0)
    {
        chamoucontrole = 1;
        controle();
    }
}

/*
        if(evBotao==1)
        {
            //Acende todos os leds:
            PORTEbits.RE5=1;//Falha_Bateria
            PORTEbits.RE6=1;//B_1
            PORTEbits.RE7=1;//B_2
            PORTCbits.RC1=1;//B_3
            PORTCbits.RC2=1;//B_4
            PORTCbits.RC3=1;//C_1
            PORTCbits.RC4=1;//C_2
            PORTGbits.RG6=1;//C_3
            PORTGbits.RG7=1;//C_4
            PORTGbits.RG8=1;//Falha_Sobrecarga
            PORTBbits.RB3=1;//Operacao_Normal
            PORTBbits.RB2=1;//Falha_Rede
        }
    else

```

```

    {
        //Apaga todos os leds:
        PORTEbits.RE5=0;//Falha_Bateria
        PORTEbits.RE6=0;//B_1
        PORTEbits.RE7=0;//B_2
        PORTCbits.RC1=0;//B_3
        PORTCbits.RC2=0;//B_4
        PORTCbits.RC3=0;//C_1
        PORTCbits.RC4=0;//C_2
        PORTGbits.RG6=0;//C_3
        PORTGbits.RG7=0;//C_4
        PORTGbits.RG8=0;//Falha_Sobrecarga
        PORTBbits.RB3=0;//Operacao_Normal
        PORTBbits.RB2=0;//Falha_Rede
    }
    */
} //Fim do while(1)

//.....

} //Fim da fun??o "main"

////////////////////////////////////
/// Fun??o Inicializa??o Geral do Sistema                                     ///
////////////////////////////////////

void aposReset(void)
{
    //Configurando Oscilador Interno:.....
    //Clock Prim?rio da Explorer 16: 8MHz (OSC1 e OSC2).
    //Clock Secund?rio da Explorer 16: 32KHz (SOSCO e SOSCI)
    //Modo de oscilador prim?rio utilizado: XT com PLL ativado (objetivo 40 MIPS
    // a partir de um cristal de 8MHz)
    //Clock usado pelo WDT (Watchdog Timer): oscilador secund?rio
    //LPRC (Low-Power RC) com frequencia de 32,768KHz
    //Configura??o de PLL interno:

```



```

//          PLLPRE<4:0>: configura o valor N1, de prescale de PLL. Faixa de
//sa?da do prescale do PLL: 0,8MHz a 8MHz.

//      PLLDIV<8:0>: configura o divisor de feedback do PLL
//(fator M que multiplica a entrada do VCO). Faixa de outVCO:100MHz a 200MHz
//          PLLPOST<1:0>: configura o postscaler que divide a frequencia de
//sa?da do VCO(fator N2).Faixa da sa?da do postscaler(Fosc):12,5MHz a 80MHz.

//      F?rmula final para c?lculo de Fosc:  $Fosc = FIN * (M / (N1 * N2))$ .

//      F?rmula final para c?lculo de Fcy
//(frequ?ncia do ciclo de instru??o):  $Fcy = Fosc / 2$ .

//Frequ?ncia de opera??o do dispositivo, ou frequencia de ciclo de
//instru??o interna, (Fcy):  $Fcy = Fosc / 2$ .

//Frequ?ncia de base de tempo de clock de perif?ricos (Fp):  $Fp = Fosc / 2$ 
//Configura??o definida para a Explorer 16:

//      CLKDIV=0: PLLPRE<4:0>=0 (escolhida frequ?ncia de entrada  $Fin/2$ , ou
N1=2)

//      PLLPOST<1:0>=0 (escolhida divis?o por 2 da sa?da do VCO, ou N2=2)
//      PLLFBD<8:0>=0x26=00100110b=38d, portanto PLLDIV<8:0>=(38+2)=40 e
M=40.

//      Resultado final:  $Fosc = 8MHz * (40 / (2+2)) = 80MHz$ ,  $Fcy = 80MHz / 2 = 40MHz$ 
//(ou 40 MIPS) e  $Tcy = 25ns = 0,025us$ .

PLLFBD = 0x0026;
CLKDIV = 0x0000;

//Configurando portas de I/O:.....

//Configurando porta A:
LATA = 0b0000000000000000;
TRISA = 0b1111111100111111;//A0      X

//Configurando porta B:
LATB = 0b0000000000000000;
TRISB = 0b1111111111110011;//B0      X

//Configurando porta C:
LATC = 0b0000000000000000;
TRISC = 0b1111111111100001;//C0

```

```

//Configurando porta D:
LATD = 0b0000000000000000;
TRISD = 0b1111101100000000;//D0      SaidaAnalogica0
                                     //Configurando porta E:

LATE = 0b0000000000000000;
TRISE = 0b1111111111111100;//E0      Falha2 (SD)

//Configurando porta F:
LATF = 0b0000000000000000;
TRISF = 0b1111110101111111;//F0X

//Configurando porta G:
LATG = 0b0000000000000000;
TRISG = 0b1001111000111100;//G0      Conexao (SD)

Delay(Delay_15mS_Cnt);
//Iniciando variáveis globais:.....
//
evBotao=0;
//Iniciando filtros, conversor AD:.....
Init_ADC();
//Configurando PWMs internos:.....
Init_PWMs();

} //Fim da função "apósReset"

void controle(void) {
    PORTBbits.RB2 = 1;

    // IFS2bits.T6IF = 0;
// RAMPA de INICIALIZAÇÃO
////////////////////////////////////
//////////BOTÃO START/STOP CONTROLADOR DO CONVERSOR//////////

```

```

if(PORTGbits.RG12==0)//PIN 33 TENS?O
{
    count++;
}
else if (count>0)
{
    count--;
}
if((count> (500000/tsamp)) & (conversorligado==0))
{
    //Evento gerado!
    conversorligado = 1;
    count = 0;
}

if ((count> (500000/tsamp)) & (conversorligado==1))
{
    //Evento gerado!
    conversorligado = 0;
    count = 0;
}

if((conversorligado==1) & (rampa<40000))
{
    rampa = rampa + (tsamp >> 7);
}

if((conversorligado==0) & (rampa>1))
{
    rampa = rampa - (tsamp >> 6);
}

//INICIO DO CONVERSOR EM A??O//
//REFERENCIA DO CONVERSOR//
ref = (rampa/10);//estava16

```

```

//ERRO DE REALIMENTA??O//
erro = ref - Sinal_ILB;
//Controlador PI
integral = integral + (long int) ((erro*tsamp)/2560);
if (integral >= 102400) // Maximo em 50% 102400
{
    integral = 102400;
}
else if (integral <= -10240) // Mimino em 1%
{
    integral = -10240;
}
saida = (erro >> 1) + (integral >> 6); // tava 6
//regeneracao
if(saida < 0){
    reg = 0;
} else {
    reg = 1600;
}
//Saturacao da saida//
if (saida >= 600)
{ // Maximo
    saida = 600;
}
else if (saida <= -600)
{ // Mimino
    saida = -600;
}

////////////////////////////////////

// Malha fechada
if(ref2pol==0) {
    ref2 = 2075 + (vetorseno[ponteiroseño]/4);
    // (vetorseno[ponteiroseño] >> 3) o idela era 2048
} else {

```

```

    ref2 = 2075 - (vetorseno[ponteiro seno]/4);
}
//Atraso de ponteiro//
atrasoponteiro++; // Para diminuir a frequencia da referencia 2 (senoide)
if (atrasoponteiro >= 4)//estava100// ESSTAVA 4
{
    atrasoponteiro = 0;
    ponteiroseno++;
    if (ponteiroseno >= 25)
    {
        ponteiroseno = 0;
        if(ref2pol == 0)
        {
            ref2pol = 1;
        } else
        {
            ref2pol = 0;
        }
    }
}
erro2 = ref2 - Sinal_VO;
//Controle P + Ressonante da malha externa de tens?o da sa?da do inversor:
vaux = erro2*15;
vaux1 = 2000*ck_1;
vaux2 = -1000*ck_2;
ck = (int) (vaux+vaux1+vaux2)/1015;
ck_2 = ck_1;
ck_1 = ck;
saida2 = (erro2 >> 1) + (ck>>1);
// Controle da polaridade da saida //
if(saida2 > 0){
    pol = 0;
} else {
    pol = 1600;
}

```

```

}
//Satura??o saida2//
if (saida2 >= 800)
{ // Maximo
    saida2 = 800;
}
else if (saida2 <= -800)
{ // Mimino
    saida2 = -800;
}
OC1RS = fabs(saida);//conversorligado*1000; //saida;//800;
OC2RS = fabs(saida) + fabs(saida2) + 20;
//Escreve o valor de Duty Cycle para o pr?ximo ciclo de PWM. (D1 + D2)
    OC3RS = pol;
//Escreve o valor de Duty Cycle para o pr?ximo ciclo de PWM. (POLARIDADE)
OC4RS = 1600;
//Escreve o valor de Duty Cyle para o pr?ximo ciclo de PWM. SEPIC
    OC5RS = reg; // Duty Cyle para o pr?ximo ciclo de PWM. REGENERACAO
//DEIXE VALOR 1600 PARA CURTO
    OC6RS = (Sinal_VO >> 2); PWM. SEPIC
PORTBbits.RB2=0;
}

```

ANEXO A – ESQUEMÁTICO DA PLACA DE CONTROLE



ANEXO B - ADC

```

/*****
*   Autores:      Andre M Soares e Rene A Capitaniao
*   Companhia:    Hiperon Engenharia Ltda
*   Arquivo:      init_ADC.h
*   Date:         07/2006
*   File Version: 1.00
*****/

////////////////////////////////////
// Função de Configuração do Módulo de AD                      //
////////////////////////////////////
// Initialize the ADC module to 12-bit mode, sampling only channel 8
// and conversion triggered by TMR 3. Output is to be in 16-bit signed
// fractional format.
// Also initialise the DMA controller Channel 0 to transfer DMA_BUFFER_SIZE
// samples from the ADC to the DPRAM before generating an interrupt.

void Init_ADC( void )
{
//Configurando módulo ADC1:.....
AD1PCFGLbits.PCFG9 = 0;//Configura a entrada AN9  ADC1.  PIN 13
AD1PCFGLbits.PCFG10 = 0;//Configura a entrada AN10  ADC1. PIN 11
AD1PCFGLbits.PCFG11 = 0;//Configura a entrada AN11  ADC1. PIN 9
AD1PCFGLbits.PCFG12 = 0;//Configura a entrada AN12  ADC1. PIN 1 e 2

AD1PCFGHbits.PCFG29 = 0;//Configura a entrada AN29  ADC1. PIN 26
AD1PCFGHbits.PCFG30 = 0;//Configura a entrada AN30  ADC1. PIN 28
AD1PCFGHbits.PCFG31 = 0;//Configura a entrada AN31  ADC1. PIN 30

AD1CON1bits.ADDMABM = 1;//ADDMABM (ou DMA Buffer Build Mode bit)=1, ou seja,
// os buffers de DMA são escritos na ordem de conversão. O ADC1
// irá prover um endereço para o canal de DMA que é o
// mesmo que o endereço usado para um buffer stand-alone não de
// DMA.
AD1CON1bits.AD12B = 1;//O conversor AD1 irá operar em modo de 12 bits
// 1 (ou seja, sem amostragem simultânea de 4 canais).
AD1CON1bits.FORM = 0;//Para valor=3: define o formato dos bits de dados de
// conversão AD como sendo 3 (ou 11b), indicando formato fracionário
// sinalizado. Dout= sddd dddd dddd 0000, onde s=not(d<11>)
// Para valor=0: define o formato dos bits de dados de conversão AD como
// sendo 0 (ou 0b), indicando formato fracionário sinalizado.
// Dout= 0000 dddd dddd dddd)
AD1CON1bits.SSRC = 0b000;//Escolhendo a fonte (o Timer) de gatilho do ADC1.
// Com o valor 010b, a comparação com valor de referência do timer
// (Timer3 para ADC1, Timer5 para ADC2) finaliza a amostragem e inicia a
// conversão AD.
AD1CON1bits.ASAM = 1;//Configurando o bit de Auto-Início de Amostragem
// de ADC. Bit ASAM=1 indica que a amostragem começa imediatamente depois
// da última conversão. O bit SAMP é automaticamente colocado em 1
// indicando de que os amplificadores S/H estão amostrando.
AD1CON1bits.DONE = 1;

```



```

AD1CON2 = 0x0000; //Configurando VCFG<2:0> (bits 13 a 15 do ADCON2)
//configuram refer?ncia do ADC. Os bits VCFG<2:0> iguais a 0: representam
//Vref+=AVdd e Vref-=AVss
AD1CON3bits.SAMC = 2; //Definindo bits 12-8 do registrador AD1CON3
// (ou SAMC<4:0>). Definindo um tempo de aquisi??o autom?tico do ADC
//igual a 2 Tad.
AD1CON3bits.ADCS = 5; //Definindo a fonte de clock do conversor AD e com
//isso o tempo de Tad. A equa??o ? a seguinte TAD = TCY ? (ADCS<7:0> + 1),
// neste caso TAD = 0,025us * 6 = 0,15us = 150ns.
AD1CSSL = 0x0000; //Selecionando quais canais ser?o varridos pelo scan
//autom?tico. Todos os bits em 0 indicam nenhuma entrada foi escolhida
//no conjunto AN0 a AN15
AD1CON4bits.DMABL = 0b000; //Configurando os bits 2-0 (DMABL<2:0>) do AD1CON4
// Seleciona o n?mero de localiza??es de buffer de DMA por entrada anal?gica
//Alocada 1 palavra do buffer de DMA para cada entrada anal?gica.
//Configurando m?dulo de DMA:.....
// clear the status register
DMA0CS0 = 0; //Limpando o registrador de status de DMA contendo bits
//indicadores de colis?o de escrita de perif?ricos para cada canal de DMA.
//Definindo os endere?os de in?cio A e B de dois buffers para opera??o
//do DMA em modo Ping-Pong.
DMA0STA = __builtin_dmaoffset(&ADCData[0]); //O buffer A de DMA come?a
//no endere?o " __builtin_dmaoffset(&ADCData[0])"
//DMA0STB = DMA0STA + DMA_BUFFER_SIZE * 2; //O buffer B de DMA come?a no
//endere?o "DMA0STA + DMA_BUFFER_SIZE", a partir do fim do buffer A de DMA.
DMA0PAD = (volatile unsigned int) &ADC1BUF0; //Definindo o endere?o do
//perif?rico que trabalhar? com o DMA.
//Este endere?o ? definido como o endere?o do buffer do perif?rico que ser?
// lido pelo DMAC.
//DMA0CNT = DMA_BUFFER_SIZE - 1; //Definindo o n?mero de itens (bytes) a
//serem transferidos pelo DMAC entre a RMA e o buffer de perif?rico.
//O n?mero de transfer?ncias de DMA = CNT<9:0> + 1.
DMA0CNT = 0;
DMA0CON = 0x0002; //Configurando o registrador de controle do canal 0 de DMA,
// definindo os bits descritos a seguir:
DMA0REQ = 13; //Configura o registrador de sele??o de IRQ do Canal 0 de DMA.
IFS0bits.DMA0IF = 0; //Limpando flag de interrup??o de DMA.
IEC0bits.DMA0IE = 1; //Habilitando interrup??o de DMA.
IPC1bits.DMA0IP = 4; //Definindo a interrup??o de DMA como tendo prioridade 4
//Ativando m?dulos configurados:.....
// turn on ADC modules
AD1CON1bits.ADON = 1; //Ativa opera??o do m?dulo ADC1.
DMA0CONbits.CHEN = 1; //Habilita o canal 0 do m?dulo de DMA.
//.....

} //Fim da fun??o "Init_ADC"

```