MODULAÇÃO PWM APLICADA A INVERSORES TRIFÁSICOS

Natan Lucas Rocha, Rodrigo Jose Piontkewicz, Arthur Garcia Bartsch
Instituto Federal de Santa Catarina
Câmpus Jaraguá do Sul – Rau – Curso de Bacharelado em Engenharia Elétrica
e-mail: natan.lr@aluno.ifsc.edu.br, rodrigo.piontkewicz@ifsc.edu.br, arthur.bartsch@ifsc.edu.br
Trabalho de Conclusão de Curso – 10/12/2023

***Resumo* – Considerando a relevância na participação em laboratórios no processo de aprimoramento do conhecimento, este trabalho de conclusão de curso visa desenvolver um inversor trifásico didático, destinado a facilitar os estudos das diversas técnicas de modulação aplicadas a conversores do tipo inversor. Para alcançar esse objetivo, foi construído um inversor trifásico 380V / 1,3kVA. O projeto obteve êxito na construção e na comparação de diferentes métodos de modulação, contribuindo para a compreensão prática desses dispositivos.**

***Palavras-chave* – Eletrônica de potência, conversores trifásicos, modulação PWM senoidal.**

PWM MODULATION APPLIED TO THREE-PHASE INVERTERS

***Abstract* – Considering the relevance of participation in laboratories in the process of improving knowledge, this undergraduate thesis aims to develop a didactic three-phase inverter, intended to facilitate the study of various modulation techniques applied to inverter-type converters. To achieve this goal, a three-phase inverter with a rating of 380V / 1.3kVA was constructed. The project succeeded in building and comparing different modulation methods, contributing to the practical understanding of these devices.**

***Keywords* – Power electronics, three-phase converters, sinusoidal PWM modulation.**

I. INTRODUÇÃO

A investigação do desempenho de inversores trifásicos com diferentes estratégias de Modulação PWM (*Pulse Width Modulation*) é uma questão relevante na área de eletrônica de potência [1], uma vez que a escolha da estratégia de modulação PWM pode impactar significativamente o desempenho desses dispositivos [2], [3].

Através da análise comparativa de diferentes técnicas de modulação PWM, é possível identificar as vantagens e desvantagens de cada método em termos de qualidade do sinal de saída, eficiência energética, resposta dinâmica, níveis de harmônicos, robustez, entre outros parâmetros relevantes.

Dada a necessidade da indústria, que está em constante busca por soluções mais eficientes, visando a economia do consumo de energia, redução de perdas e o aumento da vida útil dos equipamentos [4], a investigação comparativa das diferentes estratégias de acionamentos por PWM se faz presente na etapa de aprendizado dos discentes do curso de engenharia elétrica. Dentre elas, destacam-se o acionamento por condução a 120° e 180°, assim como a modulação senoidal de dois níveis.

Através de análises, foram avaliados parâmetros como qualidade do sinal de saída e níveis de harmônicos. Para tal, foram comparadas simulações, com o intuito de validar as principais diferenças entre as técnicas de modulação. Além disso, foi elaborado o protótipo de um inversor trifásico, servindo de base comparativa entre teoria e prática, contribuindo assim para o aprendizado dos discentes de engenharia elétrica no processo de aprendizagem de conversores do tipo inversores, bem como estratégias de modulação.

II. FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, serão explorados os principais conceitos básicos associados ao desenvolvimento do presente trabalho, fornecendo uma visão dos conceitos fundamentais de funcionamento e de projeto. A seguir, são apresentadas as subseções que compõem este capítulo:

A. Inversor

Inversores trifásicos desempenham um papel fundamental na área de eletrônica de potência, sendo amplamente utilizados em diversas aplicações industriais. Capazes de converterem corrente contínua (CC) em corrente alternada (CA), através do chaveamento coordenado dos MOSFETs ou IGBTs.

O conversor é representado pela Figura 1, este possuindo uma topologia retificadora de seis pulsos, seguida do filtro capacitivo. O inversor é composto pelas chaves S1 a S6, estas agrupadas em duas chaves em série por fase (normalmente nomeadas de fase U, V e W), de maneira que cada ramo de chaves, podem ser denominados de braços [2].

A organização da sequencia de acionamento, pode ser realizada por meio defasagem de pulsos, que é o acionamento coordenado das chaves sem variação da razão cíclica. Além deste método, a modulação PWM é uma das técnicas de acionamento que envolve a variação da largura dos pulsos de tensão para gerar uma forma de onda aproximadamente senoidal [3].

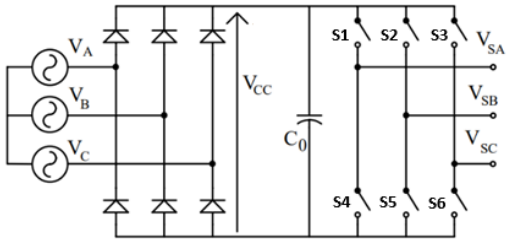


Fig. 1. Conversor trifásico. Modificado de [3].

Na Figura 1, pode-se observar que o os braços são conectados aos pontos positivo e negativo do barramento CC. Desse modo, o acionamento coordenado das chaves, resulta na variação de polaridade da tensão, gerando um sinal alternado a partir da tensão contínua.

B. Modulação PWM

A modulação PWM é amplamente utilizada nos inversores de frequência trifásicos devido à sua eficiência e precisão no controle da saída de energia, além de proporcionar um alto rendimento energético [3]. Entre os métodos de controle de tensão em conversores CC-CA, a modulação e a defasagem de sinais são técnicas frequentemente empregadas em circuitos inversores. Vale ressaltar, mesmo que a defasagem de pulso não inclua a modulação do sinal, o termo modulação PWM é comumente utilizado quando se discute métodos de controle de tensão em inversores [5].

Diversas estratégias de modulação PWM podem ser implementadas em inversores trifásicos, incluindo a modulação por defasagem, a modulação senoidal (em dois, três/múltiplos níveis) e a Modulação Vetorial. Todas essas estratégias têm como objetivo reproduzir uma forma de onda que se aproxime de uma senoide na saída do inversor, visando minimizar a presença de harmônicos [1], assegurando uma qualidade aprimorada do sinal [3]. Embora a modulação vetorial não seja abordada neste projeto, vale destacar que essa abordagem utiliza técnicas matemáticas mais avançadas, permitindo o ajuste preciso da magnitude e fase das tensões de saída [6], [7].

C. Modulação por defasagem a 180°

A modulação por defasagem a 180º, ocorre com o acionamento das chaves de um mesmo braço, operando de forma complementar. Portanto, no momento que a chave S1 está conduzindo, a chave S4 estará desliga, defasadas em 180º. Isto, tem o intuito de não ocorrer um curto-circuito no braço. As chaves dos braços em paralelo, são defasada 120º entre si, garantindo a defasagem em cada fase [8]. Para compreender melhor, a Figura 2 representa os estados das chaves do inversor. Já a tensão de saída, representada pela Figura 3.

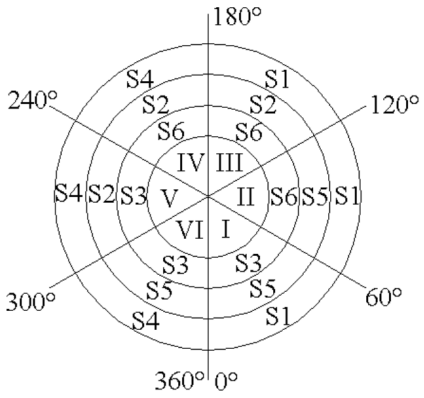


Fig. 2. Diagrama de estados das chaves a 180º [5].

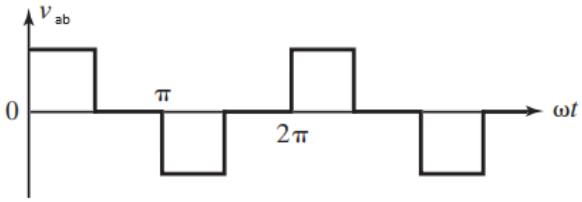


Fig. 3. Tensão de linha a 180º. Modificado de [8].

As vantagens primárias do método de defasagem de pulso a 180º envolvem uma operação mais eficiente das chaves, proporcionando um melhor aproveitamento. Adicionalmente, destaca-se a preservação das características do sinal de saída, que não é afetado pela natureza da carga [5] e [8].

D. Modulação por defasagem a 120°

A configuração de potência a 120º é semelhante à utilizada no método a 180º. No entanto, a condução ocorre por um terço do período (120º). Resultando no intervalo entre chaves de mesmo braço, em 60º durante o período de comutação [8].

Logo, pode-se observar através da Figura 4 a sequência de comutação das chaves, S1 conduz por 120°, durante 60º a mesma fica desligada e posteriormente a chave S4 entra em condução. Diferente do método a 180º, para cada ciclo, apenas duas chaves conduzem, ao invés de três. De maneira, que a tensão de saída para este método deverá ter um comportamento conforme a Figura 5.

As principais vantagens do método de condução a 120º, é tempo reduzido em que as chaves operam, gerando redução na perdas por condução. Também, se tornando uma boa alternativa para redução de tensão na carga sem alterar a tensão de entrada [5]. O ponto negativo, é que o formato da tensão de saída depende da carga, mais perceptíveis em cargas predominantemente indutivas.

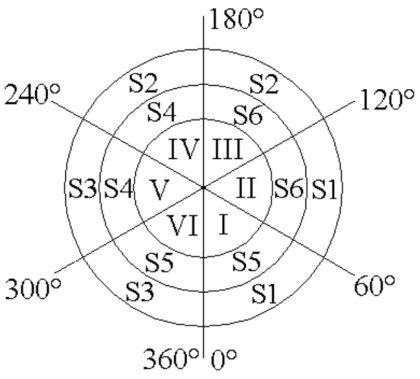


Fig. 4. Diagrama de estados das chaves a 120º [5].

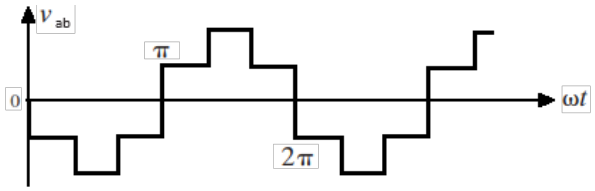


Fig. 5. Tensão de linha a 120º. Modificado de [8].

E. Modulação senoidal de dois níveis:

Na modulação senoidal de dois níveis, o sinal de comando das chaves é obtido através da comparação de duas formas de ondas, uma senoidal e uma triangular, respectivamente definidas como modulante e portadora.

Conforme ilustrado na Figura 6, a cada intervalo em que o sinal modulante cruza o sinal da onda portadora, ocorre uma alteração no estado da chave. Quando o sinal portador é inferior ao sinal modulante, a chave conduz; inversamente, quando o sinal portador é maior, a situação se inverte. Em relação ao tempo de condução, à medida que a magnitude da onda senoidal aumenta, a chave permanece conduzindo por mais tempo, enquanto uma menor magnitude resulta em um período maior em que a chave permanece desligada.

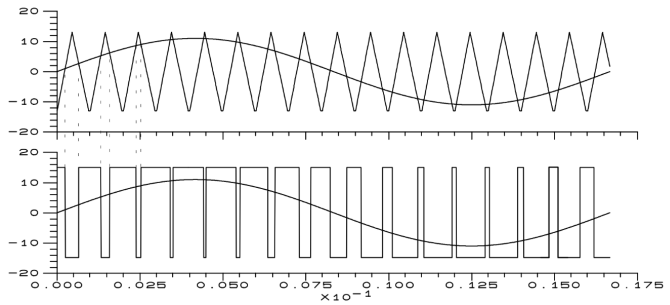


Fig. 6. Modulação senoidal a dois níveis [3].

Na operação em dois níveis, observam-se algumas semelhanças com a condução a 180º. Nesse método, os sinais modulantes, aplicados no mesmo braço (S1 e S4), são defasados em 180º, assegurando a prevenção de curto-circuito no braço. Quanto à tensão, ela é controlada ajustando a largura dos pulsos entre valores máximo e mínimo, resultando em um sinal quadrado. Esse sinal varia sua razão cíclica de modo a aproximar o sinal médio a uma onda senoidal [9].

Comparado à modulação por defasagem, esse método produz um sinal de saída com menor conteúdo harmônico nas proximidades da frequência fundamental, deslocando-o para a frequência de comutação, sendo essa sua principal vantagem. No entanto, é importante notar que as perdas por comutação são mais elevadas em relação a outros métodos.

III. METODOLOGIA

A. Especificações de projeto

De acordo com a ANEEL(Agência Nacional de Energia Elétrica), a tensão a ser contratada nos pontos de conexão com tensão nominal de operação inferior a 230 kV deverá situar-se entre 95 % (noventa e cinco por cento) e 105 % (cento e cinco por cento) da tensão nominal de operação do sistema no ponto de conexão [10]. De tal forma, foram feitas as seguintes considerações:

$$V_{\text{Linha}} = 220\sqrt{3} = 380\,\text{V} \tag{1}$$

Considerando a variação mínima prevista, obtêm-se:

$$V_{\text{LinhaMin}} = V_{\text{Linha}} \cdot 95\,\% = 361\,\text{V} \tag{2}$$

Considerando a variação máxima prevista, obtêm-se:

$$V_{\text{LinhaMax}} = V_{\text{Linha}} \cdot 105\,\% = 399\,\text{V} \tag{3}$$

Para o correto dimensionamento inversor e cálculo dos esforços dos semicondutores, é necessário, definir a potência de operação. Para o projeto, foi definido uma potência aparente de 1300 VA.

B. Retificador

Para dimensionar um retificador, necessita-se definir a tensão de saída após a retificação. De acordo com [11] a tensão em um retificador de 6 pulsos é dado por:

$$Vcc = V_{\text{Linha}} \cdot \sqrt{2} = 537{,}40\,\text{Vcc} \tag{4}$$

Para tensão mínima:

$$Vcc_{Min} = V_{\text{LinhaMin}} \cdot \sqrt{2} = 510{,}53\,\text{Vcc} \tag{5}$$

Por fim, para a tensão máxima:

$$Vcc_{Max} = V_{\text{LinhaMax}} \cdot \sqrt{2} = 564{,}27\,\text{Vcc} \tag{6}$$

Com a tensão retificada, pode-se determinar través da tensão mínima (5) e da potência aparente (1300 VA) a corrente máxima:

$$Icc_{\text{Max}} = \frac{S}{Vcc_{Min}} = 2{,}55\,\text{A} \tag{7}$$

Dado a corrente máxima (7), determina-se a corrente média nos diodos retificadores [11]:

$$ID_{\text{Med}} = \frac{Icc_{\text{Max}}}{3} = 0,849\,\text{A} \tag{8}$$

Na sequência, a corrente RMS nos diodos [11]:

$$ID_{\text{RMS}} = \frac{Icc_{\text{Max}}}{\sqrt{3}} = 1,47\,\text{A} \tag{9}$$

De acordo com o resultado obtido em (9), defini-se um diodo que suporte uma corrente RMS maior que 1,47 A . De tal forma, foi selecionado seis unidades do diodo 6A10R-6, capazes de suportar 6 A RMS/400 A pico e 700 V RMS/1000 V pico.

C. Dimensionamento do link-CC

Para dimensionamento do capacitor do filtro do retificador é necessário definir a variação de tensão. Foi estipulado que o *ripple* máximo, será de até 3 % de variação da tensão nominal. Para determinar a capacitância necessária, pode-se utilizar o método de análise do ábaco [11]. Para tal, define-se que a resistência de carga é encontrada através de (5) e (7):

$$R_L = \frac{Vcc_{Min}}{Icc_{\text{Max}}} = 200,20 \approx 200\,\Omega \tag{10}$$

De posse da resistência de carga, é definido ωRC para aplicar ao ábaco, logo:

$$\omega RC = 2 \cdot \pi \cdot 60 \cdot 470\mu \cdot 200 \approx 35,5 \tag{11}$$

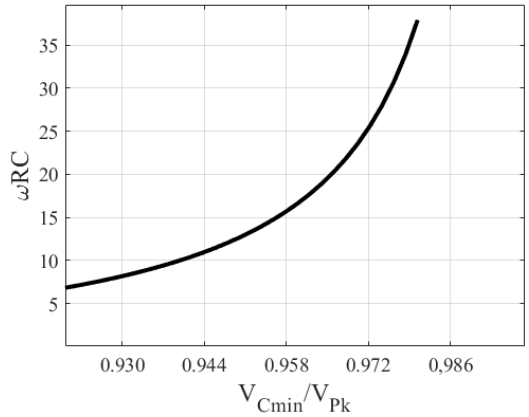


Fig. 7. Ábaco retificador trifásico. Modificado de [11]

Aplicando o valor 35,5 de (11) na curva da Figura 7, assim, obtem-se aproximadamente 97,78% de V_{Cmin}/V_{PK}. Logo o ripple obtido é de 2,22%, menor que 3% inicialmente desejado.

Em seguida, é necessário determinar a corrente do banco de capacitores do filtro capacitivo.

Novamente aplicando (11) a curva da Figura 8, o resultado obtido é aproximadamente 2,1. Portanto, a corrente RMS do banco de capacitores pode ser definida isolando IC_{RMS} [11]:

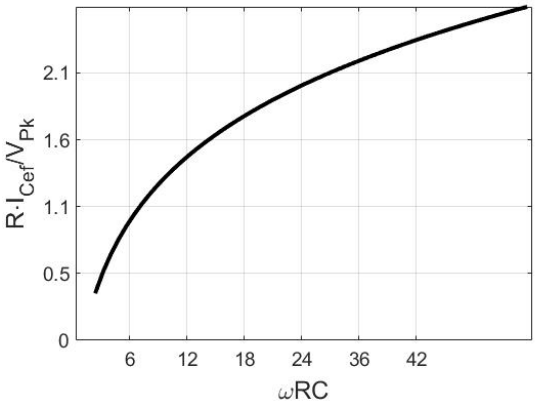


Fig. 8. Ábaco corrente capacitor. Modificado de [11]

$$IC_{\text{RMS}} = \frac{2,1 \cdot Vcc_{Min}}{RL} = 5,36\,\text{A} \tag{12}$$

Portanto, a corrente de pico:

$$IC_{\text{Pico}} = 2 \cdot \pi \cdot f \cdot C \cdot \sqrt{(Vcc_{Min})^2 - (Vcc_{Min} \cdot 97,78\%)^2} \tag{13a}$$

$$IC_{Pico} = 18,95A \tag{13b}$$

Foi definido o capacitor B43501-A5477-M, com as característica do fabricante expressa pela Tabela I.

TABELA I
Dados do capacitor

C	470 µF
$Vmax$	450 V
$Imax$ (100 Hz - 40 ℃)	5,69 A
$Imax$ (100 Hz - 85 ℃)	2,58 A

Para atender as características de tensão e corrente calculadas, optou-se por associar quatro capacitores em série/paralelo. Deste modo, garantindo uma tensão máxima de 900 V e o dobro de corrente suportável por capacitor. Para garantir que a corrente se distribua entre os ramos de forma igual, foram utilizados resistores de balanço de 110 kΩ.

D. Dimensionamento dos MOSFETs

Para determinar os esforços nas chaves, utiliza-se a potência aparente de 1300 VA definida anteriormente. A tensão da carga é avaliada conforme a equação (2), considerando o pior caso para uma tensão nominal de 380 V. Com base na potência aparente e na tensão miníma na carga, torna-se possível determinar a corrente nominal máxima, logo:

$$InMax_{380\text{V}} = \frac{S}{V_{\text{LinhaMin}} \cdot \sqrt{3}} = 2,08\,\text{A} \tag{14}$$

Corrente de pico em 380 V:

$$IP_{380\,\mathrm{V}} = InMax_{380\mathrm{V}} \cdot \sqrt{2} = 2{,}94\,\mathrm{A} \tag{15}$$

Também espera-se poder utilizar cargas em 220 V, desde que a corrente requerida não exceda o valor da corrente calculada em 380 V. Portanto, assumindo que $InMax_{220\mathrm{V}}$ é $InMax_{380\mathrm{V}}$ tem-se:

$$InMax_{220\mathrm{V}} = InMax_{380\mathrm{V}} = 2{,}08\,\mathrm{A} \tag{16}$$

De mesma forma, em 220 V considera-se o pior caso de tensão que é 95% da tensão nominal, obtendo a potência aparente em 220 V:

$$S_{220\,\mathrm{V}} = 220V \cdot 95\% \cdot \sqrt{3} \cdot InMax_{220\mathrm{V}} \approx 750\,\mathrm{VA} \tag{17}$$

Com base nos dados deduzidos, determina-se que as chaves devem suportar uma corrente nominal de 2,08 A (14), uma corrente de pico de 2,94 A (15) e uma tensão máxima de 564,27 V (6). Portanto, o MOSFET 2SK2847 foi escolhido, pois atende aos requisitos do projeto e foi modelo comercialmente mais próximo, de maneira que os dados do fabricantes estão expostos pela Tabela II.

TABELA II
Dados do MOSFET

Vds	$900V$
Id	$8A$
Tr	$10ns$
Tj	$5ns$
R_{jc}	1,47 ºC/W
R_{ja}	41.6 ºC/W
R_{DSon}	$1{,}4\,\Omega$
T_{Jmax}	150 ºC/W

E. Dimensionamento dos dissipadores do MOSFETs

Para que o conversor possa operar com carga em uma temperatura de trabalho ideal, é necessário o uso de dissipadores que auxiliem na dissipação térmica.

TABELA III
Considerações para dimensionamento do dissipador

T_a	40 ºC
R_{cd}	1 ºC/W
I_D	2,08 A
f_s	10 kHz

De posse das considerações da Tabela III é possível calcular as perdas. De acordo com [12] as perda por condução são expressas por:

$$P_{cond} = R_{DSON} \cdot {I_D}^2 = 6{,}057W \tag{18}$$

De forma simplificada, as perdas no chaveamento de um MOSFET, são representadas pela perda no momento em que o interruptor entra em condução (Pon) e e perda durante o bloqueio do interruptor (Poff) [12] e podem ser obtidas respectivamente por:

$$P_{ON} = \frac{1}{2} \cdot Vcc_{max} \cdot I_D \cdot T_r \cdot F_s = 0{,}147\,\mathrm{W} \tag{19}$$

$$P_{Off} = \frac{1}{2} \cdot Vcc_{max} \cdot I_D \cdot T_f \cdot F_s = 0{,}117W \tag{20}$$

De posse das perdas supracitadas, tona-se possível a soma das perdas, encontrando a perda total:

$$P_d = P_{cond} + P_{ON} + P_{Off} = 6{,}321W \tag{21}$$

A partir de (21), define-se a temperatura de junção do MOSFET:

$$T_j = T_a + P_d \cdot R_{ja} = 302{,}95\,^{\circ}C \tag{22}$$

Como observado a temperatura calculado do MOSFET atingiu 302,95 ºC sem o uso de dissipador, para que não exceda os 150 ºC suportado, deve-se calcular a resistência térmica mínima entre junção e ambiente:

$$R_{jaMax} = \frac{T_{jMax} - T_a}{P_d} = 17{,}402\,^{\circ}C/W \tag{23}$$

De posse de (23), pode-se definir a resistência térmica mínima para o dissipador de calor:

$$R_{da} = R_{jaMax} - R_{jc} - R_{cd} = 14{,}932\,^{\circ}C/W \tag{24}$$

Dado o exposto, foi selecionado o dissipador HS 4225 - 50 mm, que possui uma resistência térmica de 4,38 ºC/W, inferior a (24). De tal modo, atendendo a resistência mínima exigida para atingir temperatura suportada no MOSFET. Para manter a temperatura de trabalho mais próxima da temperatura ambiente, utilizou-se ventilação forçada visando uma redução térmica mais eficaz.

F. Dimensionamento Gate Driver

Para acionar um MOSFET, necessita-se aplicar uma tensão no *gate* do transistor. O MOSFET 2SK2847 exige pelo menos 12 V de gate, porém a tensão gerada pelo PWM do DSP é de apenas 3,3 V. Logo, o dimensionamento do *gate driver* é fundamental para adequar os requisitos mínimos de acionamento da chave.

De tal modo, optou-se por utilizar optoacopladores, estes que garantem a isolação entre o sistema de comando e o sistema de potência, bem como ajustam a tensão de saída do PWM para 15 V.

A Figura 9 ilustra o circuito de drive adotado, onde sinal PWM gerado pelo DSP, aciona o fototransistor, que por sua vez aciona transistores de saída, que direcionam a tensão de 15 V para resistência de *gate*. Para aplicação, foi selecionado o optoacoplador HCPL3120. Para sua alimentação foram selecionadas fontes alimentação isoladas de 15 V do modelo

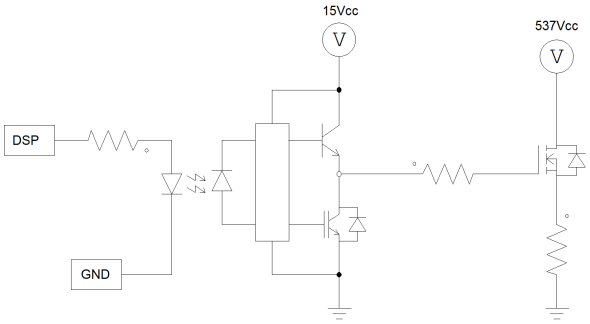


Fig. 9. Circuito *gate driver*.

B1215S-2WR3. Já para a resistência de gate, foi adotado um valor de 47 Ω.

G. Simulações

A etapa de simulação desempenha um papel fundamental na validação dos cálculos e das topologias empregadas. Para este propósito, foram utilizados os *softwares* LTSPICE e PSIM. Respectivamente, empregado na validação do circuito retificador/filtro e o funcionamento do circuito chaveado, incluindo a modulação PWM.

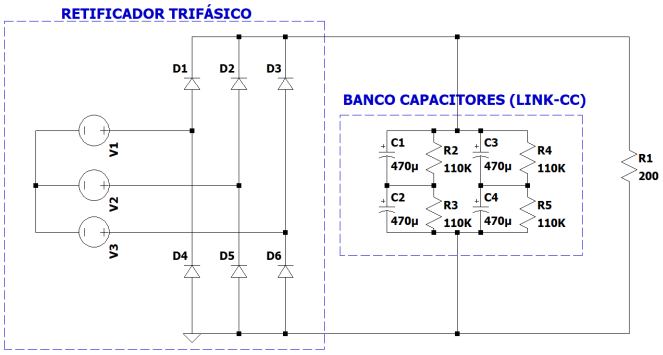


Fig. 10. Retificador trifásico de seis pulsos - Simulação.

TABELA IV
Simulação retificador

Grandeza	Calculado	Simulado
Vcc	$510,53V$	$508,25V$
$Ripple$	$2,22\%$	$2,35\%$
IC_{RMS}	$5,65A$	$4,27A$
IC_{Pico}	$18,95A$	$10,34A$

A Tabela IV ilustra a comparação entre valores calculados e simulados. Em relação a tensão de saída do retificador, conforme ilustra a Figura 11, o ripple obtido foi de aproximadamente 2,35%, permanece dentro do limite estipulado de 3% e próximo do calculado. A corrente de RMS e de Pico, foram as que apresentaram a maior discrepância. Tal fato, pode ser explicado pelo a impedância de rede limitar a corrente, já que foram feitas considerações no simulador spice.

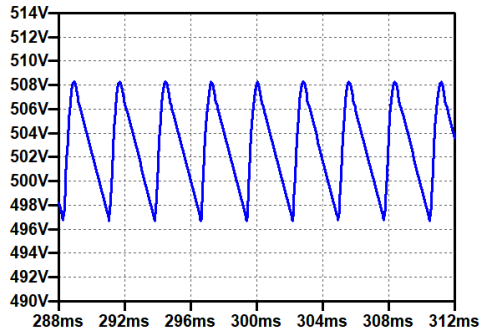


Fig. 11. Tensão retificada e filtrada - Simulação.

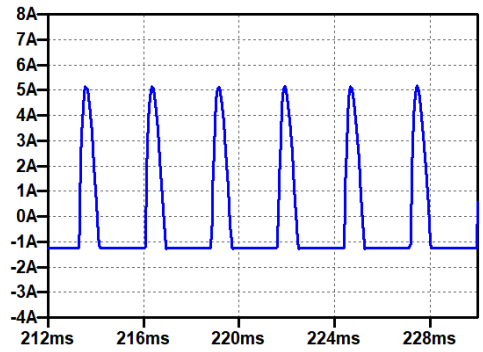


Fig. 12. Corrente do capacitor - Simulação.

Para as modulações, foram realizados as simulações no inversor com acionamento por defasagem a 180º, em 120º e modulação senoidal dois níveis. O circuito geral é representado pela Figura 13.

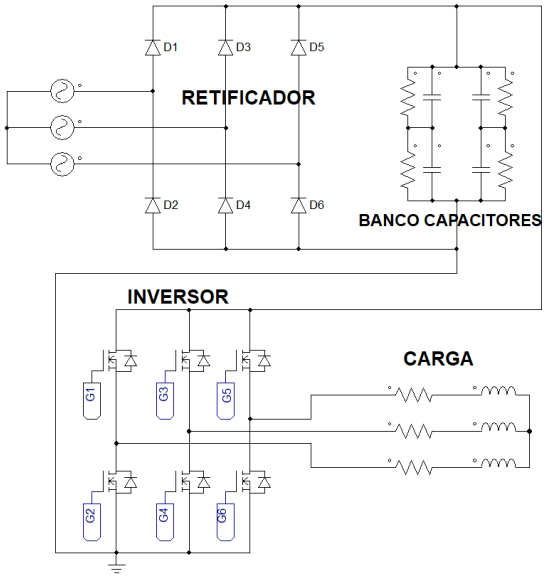


Fig. 13. Inversor trifásico - Simulação.

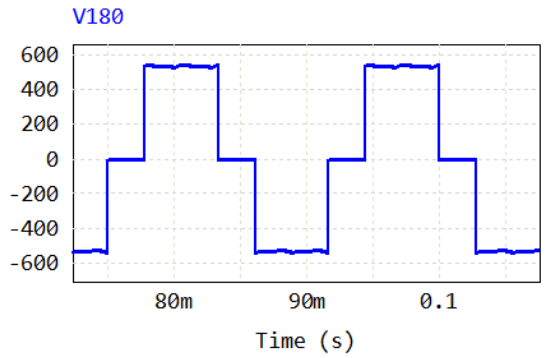


Fig. 14. Tensão de linha a 180º - Simulação.

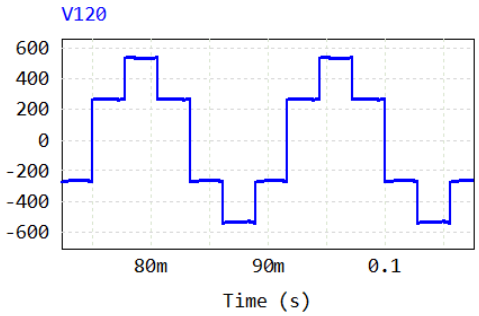


Fig. 15. Tensão de linha a 120º- Simulação.

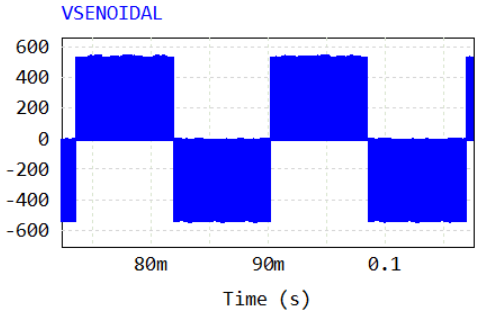


Fig. 16. Tensão de linha em modulaçlão senoidal - Simulação.

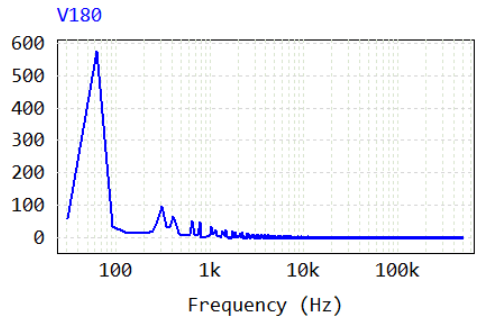


Fig. 17. FFT da tensão de linha a 180º- Simulação.

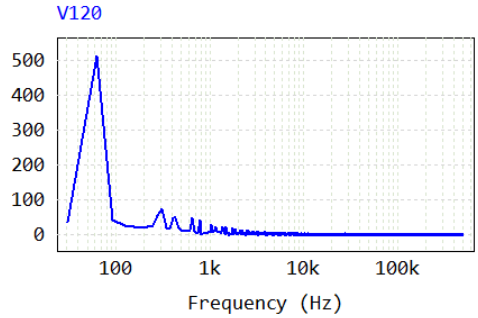


Fig. 18. FFT da tensão de linha a 120º- Simulação.

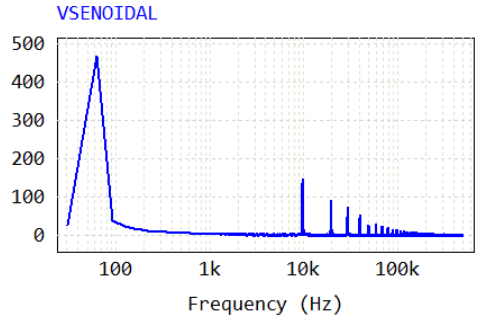


Fig. 19. FFT da tensão de linha em modulação senoidal - Simulação.

Após simulações obtidas, obtém os parâmetro expresso pela Tabela V.

TABELA V
Simulação modulações

Grandeza	180º	120ª	Senoidal dois níveis
Vcc	$537,4V$	$537,4V$	$537,4V$
f_s	60 Hz	60 Hz	10 kHz
$V_{\text{Linha(RMS)}}$	$435,59V$	$378,10V$	$395,533V$
THD_V	$31,16\,\%$	$31,12\,\%$	$68,81\,\%$
$V_{\text{Linha(60Hz)}}$	$547,88V$	$485,70V$	$441,72V$
$V_{\text{Linha(120Hz)}}$	$12,57V$	$18,94V$	$13,10V$
$V_{\text{Linha(300Hz)}}$	$80,40V$	$66,87V$	$7,70V$
$V_{\text{Linha(420Hz)}}$	$55,72V$	$47,18V$	$5,45V$
$V_{\text{Linha(10kHz)}}$	$1,74V$	$1,18V$	$99,03V$
$V_{\text{Linha(20kHz)}}$	$0,16V$	$0,27V$	$22,31V$

As principais discrepâncias entre os métodos de modulação tornam-se evidentes ao considerar a tensão $V_{\text{Linha(RMS)}}$, a distorção harmônica total de tensão THD_V e a frequência operacional de cada abordagem. No método de acionamento a 180º, a saída do inversor alcança 81% da tensão de link CC, enquanto no método a 120º, essa tensão é equivalente a 70% da tensão de link CC. Este resultado é previsível, uma vez que no método de modulação a 120º, as chaves ficam desligadas por um terço do período [13].

A distorção harmônica total de tensão THD_V entre os

métodos de defasagem permanece praticamente inalterada. Em contraste, a modulação senoidal a dois níveis exibe a maior THD_V, embora as harmônicas estejam deslocadas para a frequência de comutação e distante da fundamental.

H. Microprocessador de sinais

O micro processador de sinais utilizado é o kit de desenvolvimento STM32F103F8T6 da *STMicroelectronics*. Essa escolha foi motivada pelas características do kit e pela facilidade de acionamento e geração de sinais PWM.

O STM32F103F8T6 apresenta uma gama de portas PWM que desempenham um papel fundamental na geração dos sinais de modulação. Essas portas PWM proporcionam soluções de geração de sinais complementares de forma nativa, através de comparadores internos.

Para facilitar o desenvolvimento, optou-se pela utilização da IDE *CubeIDE*, que oferece uma interface intuitiva. Essa ferramenta não apenas simplifica o processo de programação, mas também inclui funcionalidades de fluxogramas para o acionamento das portas e direcionamento de frequência de *clocks*. Essa abordagem visa proporcionar uma visão mais clara e compreensível da implementação prática da forma de acionamento no contexto do DSP STM32F103F8T6. Detalhes de cálculo e geração e sinais, estão detalhados no apêndice A ao D.

I. Protótipo

O desenvolvimento do protótipo foi realizado através do *software Altium Designer*, elaborado com a implementação de técnicas de layout que permitiram o distanciamento das trilhas das entradas de potência R, S, T em relação às entradas de controle (DSP). De tal modo, mitigando interferências do circuito chaveado nas demais áreas da PCI. O circuito de proteção, baseado em fusíveis, foi posicionado próximo aos bornes de entrada. Quanto ao circuito de potência, optou-se pelo método de polígono, melhorando a impedância e ampliando significativamente a distribuição de corrente.

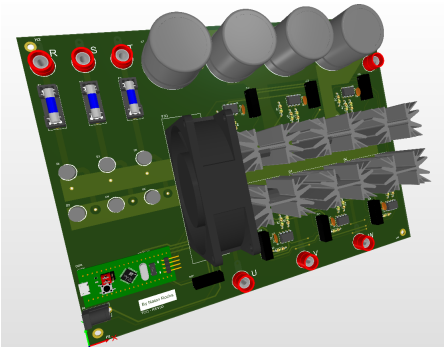


Fig. 20. Protótipo do inversor - Modelo 3D

Apos o recebimento da PCI, confeccionada por um fabricante externo, foi iniciada a montagem da placa com a inserção dos componentes e solda dos mesmos resultando no protótipo apresentado na Figura 21.

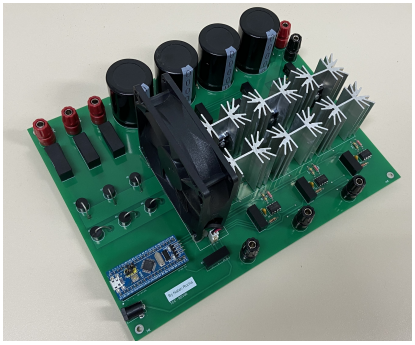


Fig. 21. Protótipo montado - Inversor.

Características de projeto, esquemático e layout, estão demonstradas no apêndice E.

IV. RESULTADOS

A. Validação dos sinais de modulação oriundos do DSP

Para analisar os resultados obtidos nos ensaios, foram exportados os dados do osciloscópio para o software PSIM, que possibilita mais ferramentas de análise. Para validação dos métodos de modulação, foram realizadas medições de tensão diretamente na saída do DSP.

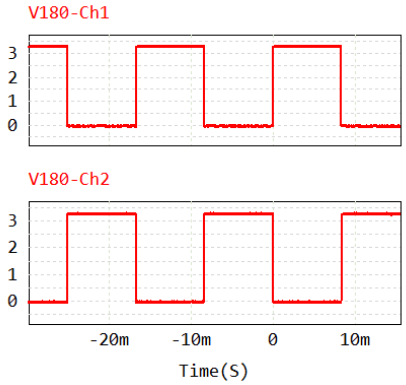


Fig. 22. Defasagem e pulsos a 180º.

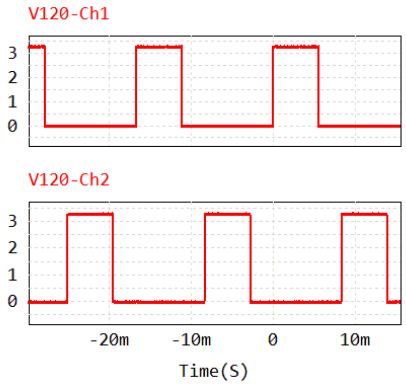


Fig. 23. Defasagem e pulsos a 120º.

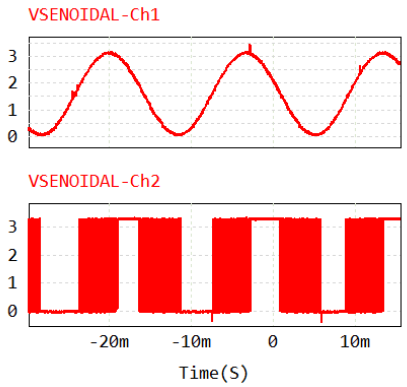


Fig. 24. Modulação senoidal.

Conforme esperado, a Figura 22 expressa o sinal entre os MOSFETs de um mesmo braço (S1 e S4), resultando apresentando uma defasagem de 180º entre si. De acordo com esperado para o método de modulação a 180º. Para a modulação a 120°, espera-se que os sinais de um mesmo braço, haja um um intervalo de 60º sem condução, entre as chaves. A Figura 23 expressa o resultado obtido com êxito.

Para garantir que não haja um curto-circuito entre os mosfets de um mesmo braço,utilizou-se um tempo porto de 1μ segundo. De tal forma. inibindo que as chaves comutem de forma simultânea.

Para a modulação senoidal de dois níveis, representado pela Figura 24, foi obtido o resultado com a variação da razão cíclica de forma senoidal. De maneira que foi utilizado para validação um filtro RC de 1ª ordem com frequência de corte em 100 Hz, este acoplado em paralelo com a saída do sinal do DSP.Sinal representado pela curva VSenoidal - Ch1, bem como o sinal não filtrado, representado pela curva VSenoidal - Ch2.

B. Ensaios

Para os ensaios do inversor, foram utilizadas lâmpadas halógenas, 220 V e 170 W cada unidade. Estas, dispostas duas unidades em série por fase, ligadas em estrela, conforme Figura 25.

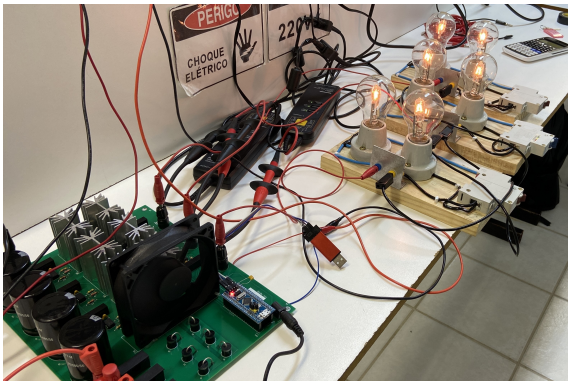


Fig. 25. Ensaio com lâmpadas.

Para todos os métodos de modulação, foi aplicado tensão na entrada do inversor, até que se obtive-se 220 V de linha na carga. As tensões de linha representadas nas figuras 26, 27 e 28, correspondem a tensão referente a fase U, sendo respectivamente correspondente a modulação de 180, 120 e senoidal 2 níveis. As demais fases foram desconsideradas para análise, já que a diferença entre as mesma, será apenas o angulo de defasem de 120º.

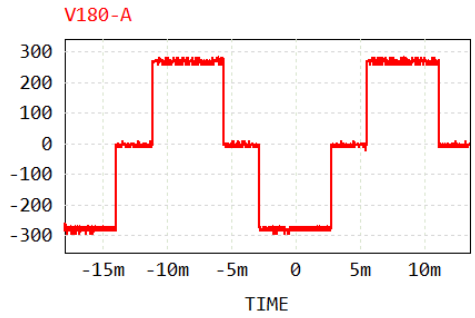


Fig. 26. Tensão de linha por defasagem de pulso a 180º .

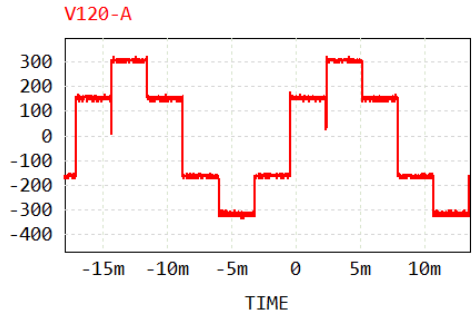


Fig. 27. Tensão de linha por defasagem de pulso a 120º . Autoria própria

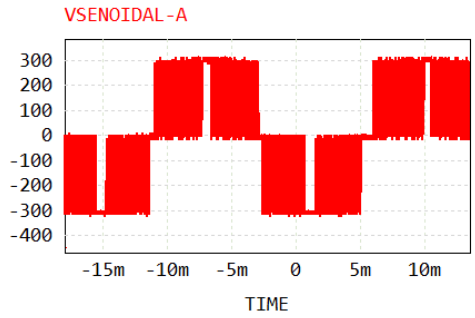


Fig. 28. Tensões de linha com modulação senoidal de dois níveis.

Para as medições no domínio da frequência, as tensões de linha são representadas nas figuras 29, 30 e 31, correspondem a tensão referente a fase U, sendo respectivamente correspondente a modulação de 180, 120 e senoidal 2 níveis.

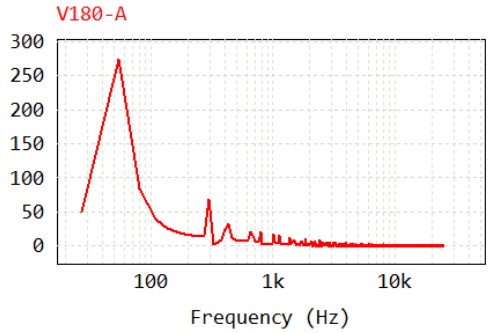


Fig. 29. FFT tensão de linha a defasagem de pulso 180°.

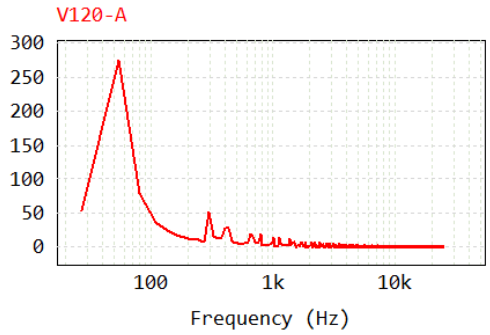


Fig. 30. FFT tensão de linha a defasagem de pulso 120°.

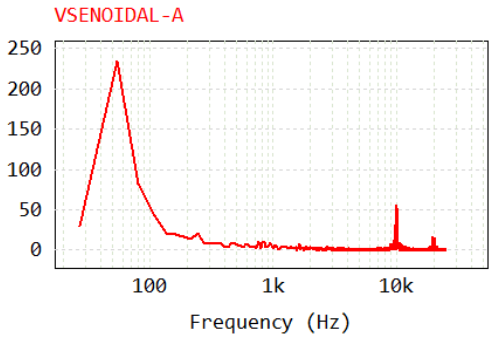


Fig. 31. FFT tensão de linha a modulação senoidal a dois níveis.

Após medições obtidas, obtém-se os parâmetro expresso pela Tabela VI, os resultados medidos se mostraram similares a simulação. No método de acionamento a 180º, a saída do inversor alcança 82% da tensão de link CC, enquanto no método a 120º, essa tensão é equivalente a 73% da tensão de link CC. Na modulação senoidal, a tensão de saída é 71% da tensão de link CC.

A THD_V entre os métodos de defasagem, permaneceu semelhante ao simulado. Para modulação a 180º foi de 31,22%, a modulação a 120º apresentou um THD_V de 31,63%. Como esperado a modulação senoidal obteve um THD_V de 64,83%, de mesma forma, as harmônicas foram deslocada para a frequência de comutação.

TABELA VI
Medições modulações - 220 V

Grandeza	180º	120ª	Senoidal dois níveis
Vcc	$268\,V$	$300\,V$	$308\,V$
f_s	60 Hz	60 Hz	10 kHz
$V_{\text{Linha(RMS)}}$	$220\,V$	$220\,V$	$220\,V$
THD_V	$31,22\,\%$	$31,63\,\%$	$64,83\,\%$
$V_{\text{Linha(60Hz)}}$	$271,88\,V$	$272,64\,V$	$232,10\,V$
$V_{\text{Linha(120Hz)}}$	$32,68\,V$	$29,67\,V$	$32,00\,V$
$V_{\text{Linha(300Hz)}}$	$59,34\,V$	$47,10\,V$	$7,97\,V$
$V_{\text{Linha(420Hz)}}$	$28,41\,V$	$27,69\,V$	$2,69\,V$
$V_{\text{Linha(10kHz)}}$	$0,46\,V$	$0,50\,V$	$50,26\,V$
$V_{\text{Linha(20kHz)}}$	$0,23\,V$	$0,35\,V$	$11,81\,V$

Também foi ensaiado a carga com uma tensão de 538 V de link CC, juntamente com o método de modulação senoidal a dois níveis.

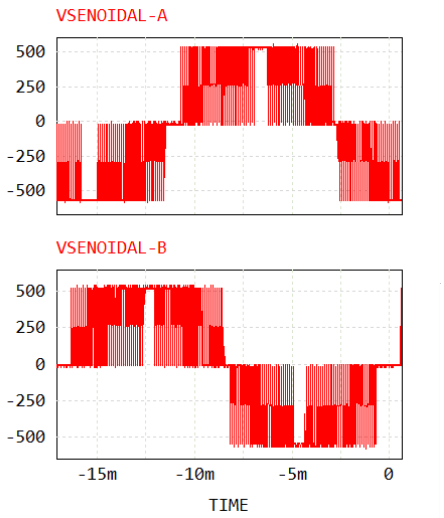


Fig. 32. Tensões de linha com modulação senoidal a dois níveis.

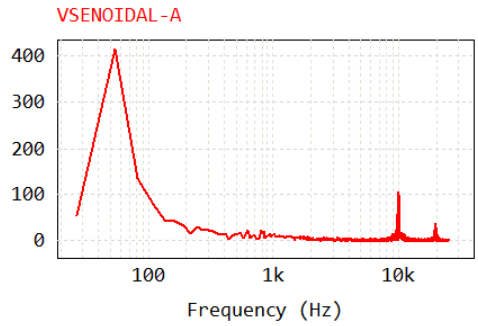


Fig. 33. FFT tensão de linha a modulação senoidal a dois níveis.

Para fins de ilustração a Figura 32 representa duas curvas, VSENOIDAL - A (fase U) e VSENOIDAL - B (fase V). Onde,

ambas estão defasadas em 120º e ambas possuem o valores de medidas semelhantes.

TABELA VII
Medições modulação - 380 V

Grandeza	Senoidal dois níveis
Vcc	$538\,V$
f_s	10 kHz
$V_{\text{Linha(RMS)}}$	$415\,V$
THD_V	64,69 %
$V_{\text{Linha(60Hz)}}$	$461\,V$
$V_{\text{Linha(120Hz)}}$	$9,80\,V$
$V_{\text{Linha(300Hz)}}$	$3,93\,V$
$V_{\text{Linha(420Hz)}}$	$2,74\,V$
$V_{\text{Linha(10kHz)}}$	$123,01\,V$
$V_{\text{Linha(20kHz)}}$	$33,21\,V$

É possível observar a Tabela VII as mesmas características obtidas anteriormente em 220 V, permanecem semelhantes, com a diferença da magnitude da tensão. Na modulação senoidal, a tensão de saída foi de 77 % da tensão de link CC, já THD_V foi de 64,69 %.

Vale destacar que foram conduzidos experimentos adicionais utilizando um motor de indução trifásico de 1/4 de CV, sob as mesmas condições de tensão de 380V e modulação senoidal de dois níveis. Os resultados obtidos com o motor assemelharam-se aos alcançados com as lâmpadas, evidenciando a consistência do método adotado. Durante os testes, confirmou-se o esperado funcionamento do motor, com a observação do eixo entrando em rotação de maneira coerente com as características típicas desse tipo de equipamento. Essa validação adicional, realizada em um cenário prático com um motor de indução, fortalece a coerência entres os resultados obtidos.

C. Considerações

De maneira geral, os testes realizados demonstram que as modulações a 180º e 120º possuem menor distorção harmônica da tensão. Isso devido a frequência de comutação, que nestes casos foram de 60 Hz. Entretanto, para a utilização de filtros de redução de harmônicas o processo é mais dificultoso, pois as harmônicas ficam muito próximas da frequência fundamental.

Entretanto, a modulação senoidal a dois níveis, apesar de possui uma distorção harmônica de tensão maior, é o método que apresenta maior vantagem. Neste método as harmônicas são deslocadas das proximidades da frequência fundamental 60 Hz, para a frequência de comutação, em 10 kHz. Facilitando a implementação de filtros, pois não há risco da atenuação de tensões no entorno da frequência fundamental.

V. CONCLUSÃO

O presente trabalho se concentrou na análise dos diferentes métodos de modulação PWM aplicados em inversores trifásicos. O projeto teve início por meio do cálculo e dimensionamento dos componentes, assegurando que atendessem plenamente às exigências do escopo. As simulações foram conduzidas para validar os cálculos obtidos, sendo esses resultados subsequentemente corroborados e validados por testes práticos.

Os métodos de modulação explorados revelaram-se de significativa importância. A aplicação prática permitiu uma avaliação mais aprofundada das diferenças entre os diversos métodos de modulação. Bem como análise harmônicos de diferentes níveis de tensão através dos métodos de acionamento.

O funcionamento do protótipo possibilita a utilização do mesmo em aplicações de unidade curriculares correlatas ao tema deste projeto. Uma causa que originou a implementação do presente projeto.

Destaca-se como dificuldade o desenvolvimento e fabricação de um inversor trifásico capaz de operar em níveis de tensão mais elevados. Considerações fundamentais, como o dimensionamento da placa e a seleção criteriosa de componentes, enriqueceram a experiência do discente. Da mesma forma, destaca-se a complexidade inerente à programação do DSP para um inversor de topologia trifásica.

De maneira abrangente, este trabalho possibilitou a aplicação e integração de conhecimentos adquiridos em diversas disciplinas, com ênfase em Eletrônica de Potência I e II, Mecânica dos Fluidos e Microcontroladores.

Quanto às perspectivas futuras, identificou-se oportunidades de aprimoramento, como a implementação de um circuito de pré-carga na entrada do inversor. Ademais, considera-se viável a implementação de um método de controle, implementando uma malha de controle que expanda as possibilidades de aplicações do protótipo.

REFERÊNCIAS

[1] A. F. de Souza, *Inversores quase-ressonantes modulados por largura de pulso*, Tese de Doutorado, Universidade Federal de Santa Catarina, 1992.

[2] F. A. B. Batista, *Modulação vetorial aplicada a retificadores trifásicos pwm unidirecionais*, Tese de Doutorado, Universidade Federal de Santa Catarina, 2006.

[3] A. J. Perin, *Modulação PWM*, Universidade Federal de Santa Catarina, 2006.

[4] I. Barbi, *Retificador boost bidirecional de onda completa com elevado fator de potência*, Universidade Federal de Santa Catarina, 2005.

[5] J. D. O. PACHECO, *Desenvolvimento de um sistema didático para ensino de conversores cc-ca com monitoramento por microcontroladores*, Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina, 2012.

[6] I. Barbi, *Modelagem e controle do retificador trifásico PWM bidirecional utilizando a transformada de park e modulação vetorial*, Universidade Federal de Santa Catarina, 2003.

[7] I. Barbi, *Princípios da modulação vetorial.*, Universidade Federal de Santa Catarina, 2003.

[8] M. H.Rashid, *Eletrônica de Potência: Dispositivos, Circuitos e Aplicações*, 4a ed., Pearson Universidades, São Paulo, 2014.

[9] S. K. Asuri, *Modelling and control of sparse converter fed induction motor drives*, Tennessee Tech University, 2005.

[10] Agência Nacional de Energia Elétrica - ANEEL, *"Módulo 8 - Qualidade da Energia Elétrica"*, 2017.

[11] I. Barbi, *Eletrônica de potência*, 8a ed., [s.n.], Florianópolis, 2012.

[12] I. Barbi, *Projeto de fonte chaveadas*, 2a ed., [s.n.], Florianópolis, 2007.

[13] D. C. Martins, *Introdução ao estudo dos conversores CC-CA*, 3a ed., [s.n.], Florianópolis, 2011.

Apêndice A: Configuração do DSP

Optou-se pelo uso do STM32F103F8T6, cujas características estão detalhadas na Tabela VIII.

TABELA VIII
Especificações do STM32F103C8T6

Modelo	STM32F103C8T6
Núcleo	ARM 32 Cortex-M3
Modo de Depuração	SWD
Frequência Máxima	72MHz
Memória do Programa	64kB
RAM de Dados	20kB
Resolução ADC	12 bits
Tensão de Trabalho	2,0VDC a 3,6VDC
Interface	CAN, I2C, SPI, USART, USB
Canais ADC	10
Contadores	3 Timers
Dimensões	5,3cm x 2,2cm

O ambiente de programação empregado é o STM32 Cube IDE, que possibilita a pré-configuração visual do dispositivo, otimizando o processo de programação. Nesse contexto, o primeiro passo consiste em determinar a frequência do clock interno:

$$f_{clockInterno} = 12\,\text{MHz} \tag{25}$$

Em seguida, é direcionado a frequência interna para os periféricos, como mostrado na Figura 34:

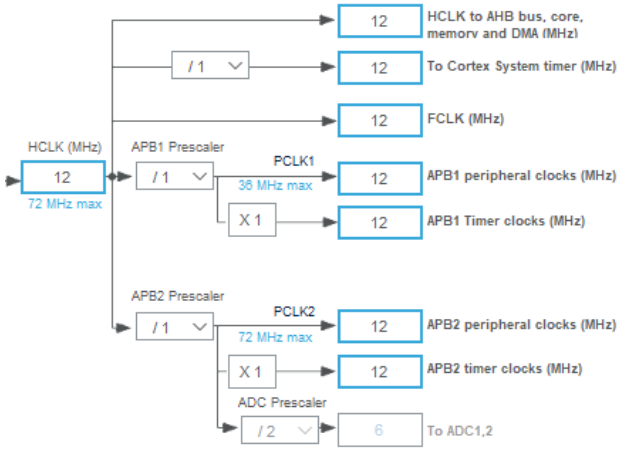


Fig. 34. Frequência clock interno - IDE.

O timer1 do dispositivo, possui saídas ADC que possui a funcionalidade de gerar sinais complementares com Dead Time, de forma nativa. Dado o exposto, foram selecionado os canais do timer1, conforme ilustrado pela Figura 35:

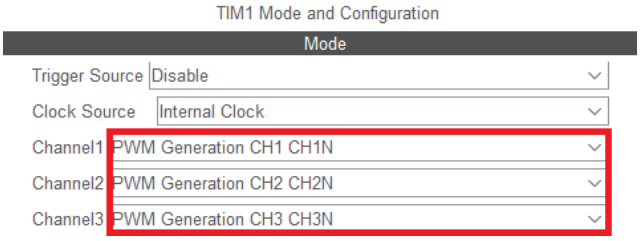


Fig. 35. Canais do timer 1 - IDE.

Em sequência, necessita-se definir o prescaler, assim, a frequência do clock interno é dividida:

$$Prescaler = \frac{f_{clockInterno}}{4} = 3\,\text{MHz} \tag{26}$$

Uma vez definido a frequência com prescaler (26), deve-se determinar a frequência que chegará no timer. Para tal, defini-se a frequência de estouro do timer:

$$f_{EstouroTimer} = \frac{Prescaler}{300} = 10\,\text{kHz} \tag{27}$$

Definido a frequência de chaveamento do inversor (27), deve-se determinar o período do timer e o tempo morto para o sinal complementar:

$$T_{Timer} = \frac{1}{f_{EstouroTimer}} = 100\,\mu\text{s} \tag{28}$$

Portando, defini-se 1 μs de tempo morto. Definindo o valor para aplicar ao software:

$$T_{Morto} = 1\,\mu\text{s} \cdot f_{clockInterno} = 12 \tag{29}$$

Na sequência, aplica-se as características supracitadas anteriormente, conforme Figura 36:

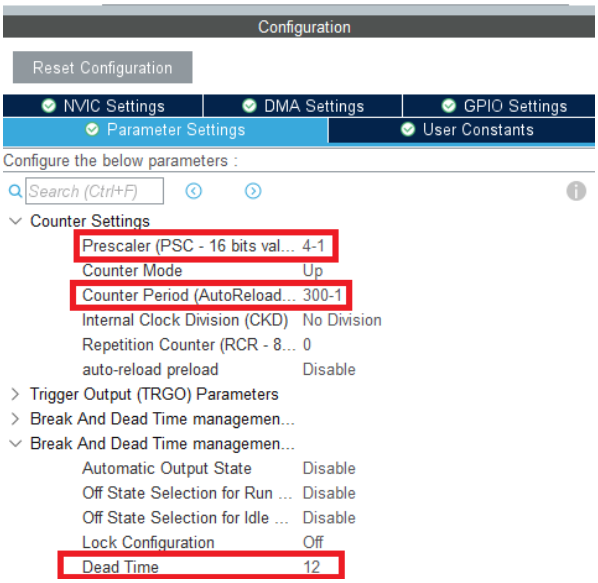


Fig. 36. Seleção de frequência pré-scaler - IDE.

Para gerar o sinal trifásico, foi utilizado três funções senoidais, nas quais, são composta por um array de 167 posições, determinado por:

$$TamVetor = \frac{f_{EstouroTimer}}{60\,\text{Hz}} \approx 167 \tag{30}$$

Apêndice B: Código modulação senoidal dois níveis

```
#include "main.h"

/* Private includes ----------------------------------------------------------*/
#include "math.h"
#define doispi 6.28

int vetor;
int timer;
timer=300;
vetor=167;

uint16_t senox[167];
uint16_t senoy[167];
uint16_t senoz[167];
uint8_t i;

/* USER CODE END Includes */

/* Private variables ---------------------------------------------------------*/
TIM_HandleTypeDef htim1;
DMA_HandleTypeDef hdma_tim1_ch1;
DMA_HandleTypeDef hdma_tim1_ch2;
DMA_HandleTypeDef hdma_tim1_ch3;

/* USER CODE END PV */
/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_DMA_Init(void);
static void MX_TIM1_Init(void);
/* USER CODE BEGIN PFP */

int main(void)
{

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* Configure the system clock */
  SystemClock_Config();


  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_DMA_Init();
  MX_TIM1_Init();
  /* USER CODE BEGIN 2 */

  HAL_TIM_PWM_Start_DMA(&htim1, TIM_CHANNEL_1, senox, vetor);
  HAL_TIMEx_PWMN_Start(&htim1, TIM_CHANNEL_1);
```

```
  HAL_TIM_PWM_Start_DMA(&htim1, TIM_CHANNEL_2, senoy, vetor);
  HAL_TIMEx_PWMN_Start(&htim1, TIM_CHANNEL_2);

  HAL_TIM_PWM_Start_DMA(&htim1, TIM_CHANNEL_3, senoz, vetor);
  HAL_TIMEx_PWMN_Start(&htim1, TIM_CHANNEL_3);

  for (i=0; i<vetor; i++)
  {
  senox[i] = ((sin(i * (doispi / vetor)))+ 1) * ((timer + 1) / 2);
  senoy[i] = ((sin((i - (vetor / 3)) * (doispi / vetor)))+1) * ((timer + 1) / 2);
  senoz[i] = ((sin((i - (2 * vetor / 3)) * (doispi / vetor)))+1) * ((timer + 1) / 2);
  }


  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}

void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.HSEPredivValue = RCC_HSE_PREDIV_DIV2;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
  RCC_OscInitStruct.PLL.PLLMUL = RCC_PLL_MUL3;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) != HAL_OK)
```

```
  {
    Error_Handler();
  }
}

/**

static void MX_TIM1_Init(void)
{

  /* USER CODE END TIM1_Init 0 */

  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};
  TIM_OC_InitTypeDef sConfigOC = {0};
  TIM_BreakDeadTimeConfigTypeDef sBreakDeadTimeConfig = {0};

  /* USER CODE END TIM1_Init 1 */
  htim1.Instance = TIM1;
  htim1.Init.Prescaler = 4-1;
  htim1.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim1.Init.Period = 300-1;
  htim1.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim1.Init.RepetitionCounter = 0;
  htim1.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim1) != HAL_OK)
  {
    Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim1, &sClockSourceConfig) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_TIM_PWM_Init(&htim1) != HAL_OK)
  {
    Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim1, &sMasterConfig) != HAL_OK)
  {
    Error_Handler();
  }
  sConfigOC.OCMode = TIM_OCMODE_PWM1;
  sConfigOC.Pulse = 150;
  sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
  sConfigOC.OCNPolarity = TIM_OCNPOLARITY_HIGH;
  sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
  sConfigOC.OCIdleState = TIM_OCIDLESTATE_RESET;
  sConfigOC.OCNIdleState = TIM_OCNIDLESTATE_RESET;
  if (HAL_TIM_PWM_ConfigChannel(&htim1, &sConfigOC, TIM_CHANNEL_1) != HAL_OK)
  {
    Error_Handler();
```

```
  }
  if (HAL_TIM_PWM_ConfigChannel(&htim1, &sConfigOC, TIM_CHANNEL_2) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_TIM_PWM_ConfigChannel(&htim1, &sConfigOC, TIM_CHANNEL_3) != HAL_OK)
  {
    Error_Handler();
  }
  sBreakDeadTimeConfig.OffStateRunMode = TIM_OSSR_DISABLE;
  sBreakDeadTimeConfig.OffStateIDLEMode = TIM_OSSI_DISABLE;
  sBreakDeadTimeConfig.LockLevel = TIM_LOCKLEVEL_OFF;
  sBreakDeadTimeConfig.DeadTime = 12;
  sBreakDeadTimeConfig.BreakState = TIM_BREAK_DISABLE;
  sBreakDeadTimeConfig.BreakPolarity = TIM_BREAKPOLARITY_HIGH;
  sBreakDeadTimeConfig.AutomaticOutput = TIM_AUTOMATICOUTPUT_DISABLE;
  if (HAL_TIMEx_ConfigBreakDeadTime(&htim1, &sBreakDeadTimeConfig) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN TIM1_Init 2 */

  /* USER CODE END TIM1_Init 2 */
  HAL_TIM_MspPostInit(&htim1);

}

/**
  * Enable DMA controller clock

static void MX_DMA_Init(void)
{

  /* DMA controller clock enable */
  __HAL_RCC_DMA1_CLK_ENABLE();

  /* DMA1_Channel2_IRQn interrupt configuration */
  HAL_NVIC_SetPriority(DMA1_Channel2_IRQn, 0, 0);
  HAL_NVIC_EnableIRQ(DMA1_Channel2_IRQn);
  /* DMA1_Channel3_IRQn interrupt configuration */
  HAL_NVIC_SetPriority(DMA1_Channel3_IRQn, 0, 0);
  HAL_NVIC_EnableIRQ(DMA1_Channel3_IRQn);
  /* DMA1_Channel6_IRQn interrupt configuration */
  HAL_NVIC_SetPriority(DMA1_Channel6_IRQn, 0, 0);
  HAL_NVIC_EnableIRQ(DMA1_Channel6_IRQn);

}

/**
  * @brief GPIO Initialization Function
  */
static void MX_GPIO_Init(void)
{
/* USER CODE BEGIN MX_GPIO_Init_1 */
```

```
/* USER CODE END MX_GPIO_Init_1 */

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOD_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOB_CLK_ENABLE();

/* USER CODE END MX_GPIO_Init_2 */
}

/**
  * @brief  This function is executed in case of error occurrence.
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}

#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number

  */
void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line number,
 ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```

Apêndice C: Código modulação a 180º

```
#include "main.h"

TIM_HandleTypeDef htim3;

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_TIM3_Init(void);

/* USER CODE BEGIN 0 */

void delay_us(uint16_t us) //Função para atraso em microssegundos
{
  __HAL_TIM_SET_COUNTER(&htim3, 0);  // Defina o valor do contador como 0
  while (__HAL_TIM_GET_COUNTER(&htim3) < us);  // Aguarde até que o contador atinja o valor desejado
}

/* USER CODE END 0 */

/**
  * @brief  The application entry point.

int main(void)
{

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM3_Init();
  /* USER CODE BEGIN 2 */
  HAL_TIM_Base_Start(&htim3);
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {

    /* USER CODE BEGIN 3 */

      //Primeiro quadrante (0-60°)
  delay_us(2774);
  //Segundo quadrante (60-120°)
  HAL_GPIO_TogglePin(S3_GPIO_Port,S3_Pin); //Desativa S3
  HAL_GPIO_TogglePin(S6_GPIO_Port,S6_Pin); //Aciona S6
  delay_us(2774);
  //Terceiro quadrante (120-180°)
  HAL_GPIO_TogglePin(S5_GPIO_Port,S5_Pin); //Desativa S5
  HAL_GPIO_TogglePin(S2_GPIO_Port,S2_Pin); //Aciona S2
  delay_us(2774);
  //Quarto quadrante (180-240°)
  HAL_GPIO_TogglePin(S1_GPIO_Port,S1_Pin); //Desativa S1
```

```
  HAL_GPIO_TogglePin(S4_GPIO_Port,S4_Pin); //Aciona S4
  delay_us(2774);
  //Quinto quadrante (240-300°)
  HAL_GPIO_TogglePin(S6_GPIO_Port,S6_Pin); //Desativa S6
  HAL_GPIO_TogglePin(S3_GPIO_Port,S3_Pin); //Aciona S3
  delay_us(2774);
  //Sexto quadrante (300-360°)
  HAL_GPIO_TogglePin(S2_GPIO_Port,S2_Pin); //Desativa S2
  HAL_GPIO_TogglePin(S5_GPIO_Port,S5_Pin); //Aciona S5
  delay_us(2774);
  HAL_GPIO_TogglePin(S4_GPIO_Port,S4_Pin); //Desativa S4
  HAL_GPIO_TogglePin(S1_GPIO_Port,S1_Pin); //Aciona S1
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration

void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI_DIV2;
  RCC_OscInitStruct.PLL.PLLMUL = RCC_PLL_MUL3;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) != HAL_OK)
  {
    Error_Handler();
  }
}

/**
```

```
  * @brief TIM3 Initialization Function

static void MX_TIM3_Init(void)
{
  /* USER CODE END TIM3_Init 0 */

  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};

  /* USER CODE END TIM3_Init 1 */
  htim3.Instance = TIM3;
  htim3.Init.Prescaler = 12-1;
  htim3.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim3.Init.Period = 0xffff-1;
  htim3.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim3.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim3) != HAL_OK)
  {
    Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim3, &sClockSourceConfig) != HAL_OK)
  {
    Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim3, &sMasterConfig) != HAL_OK)
  {
    Error_Handler();
  }
}

/**
  * @brief GPIO Initialization Function

static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOD_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOB_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOA, S4_Pin|S2_Pin, GPIO_PIN_RESET);

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(S5_GPIO_Port, S5_Pin, GPIO_PIN_SET);

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(S6_GPIO_Port, S6_Pin, GPIO_PIN_RESET);
```

```
  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOA, S1_Pin|S3_Pin, GPIO_PIN_SET);

  /*Configure GPIO pins : S4_Pin S1_Pin S2_Pin S3_Pin */
  GPIO_InitStruct.Pin = S4_Pin|S1_Pin|S2_Pin|S3_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

  /*Configure GPIO pins : S5_Pin S6_Pin */
  GPIO_InitStruct.Pin = S5_Pin|S6_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
  HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);

}

/**
  * @brief  This function is executed in case of error occurrence.
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}

#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number

void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```

Apêndice D: Código modulação a 120º

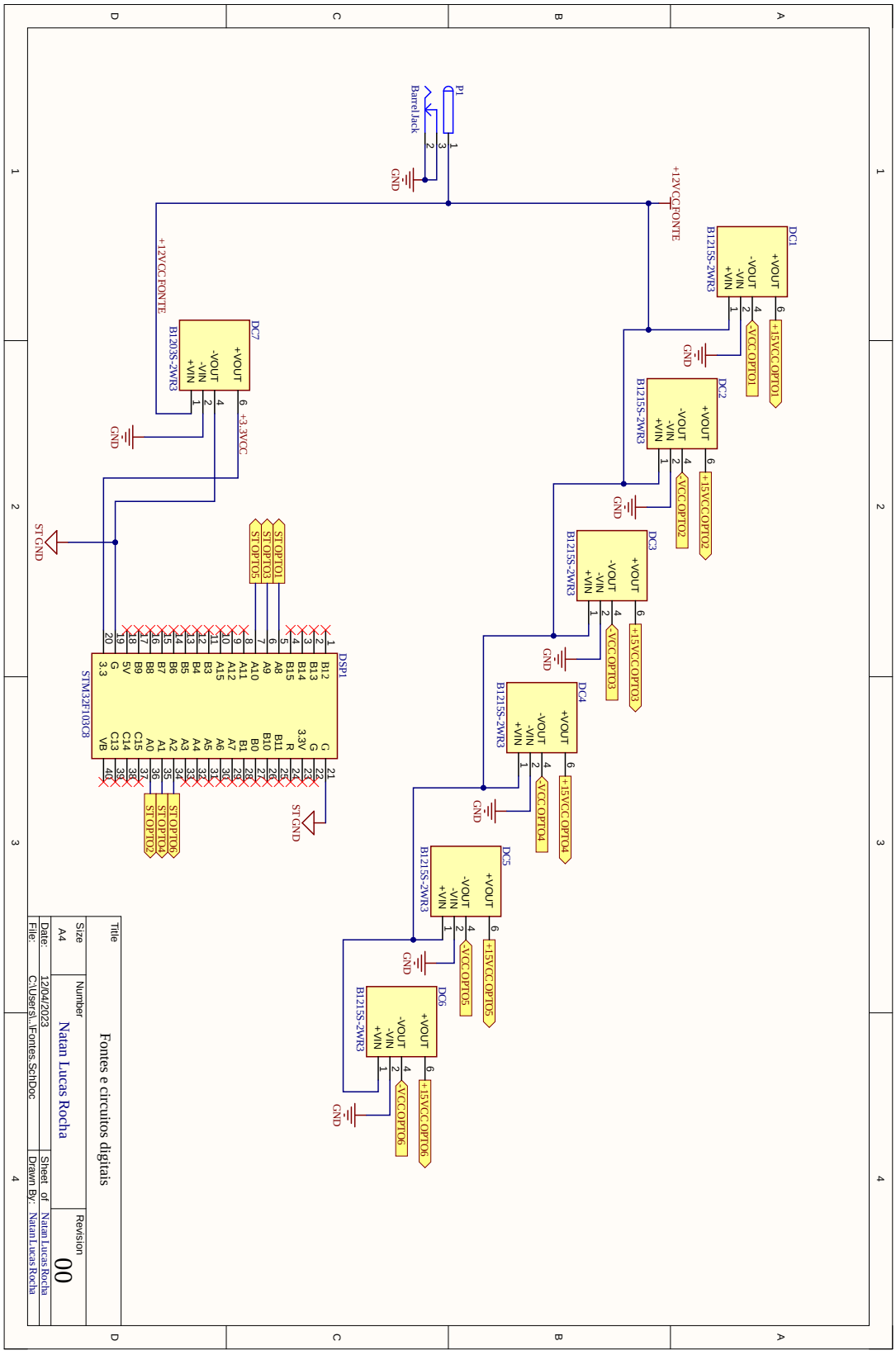
```
#include "main.h"

/* Private variables ---------------------------------------------------------*/
TIM_HandleTypeDef htim3;

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_TIM3_Init(void);

/* USER CODE BEGIN 0 */

void delay_us(uint16_t us) //Função para atraso em microssegundos
{
  __HAL_TIM_SET_COUNTER(&htim3, 0);  // Defina o valor do contador como 0
  while (__HAL_TIM_GET_COUNTER(&htim3) < us);  // Aguarde até que o contador atinja o valor desejado
}

/**
  * @brief  The application entry point.

int main(void)
{
   /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM3_Init();
  /* USER CODE BEGIN 2 */
  HAL_TIM_Base_Start(&htim3);
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */

            //Primeiro quadrante (0-60°)
  delay_us(2774);
  //Segundo quadrante (60-120°)
  HAL_GPIO_TogglePin(S5_GPIO_Port,S5_Pin); //Desativa S5
  HAL_GPIO_TogglePin(S6_GPIO_Port,S6_Pin); //Aciona S6
  delay_us(2774);
  //Terceiro quadrante (120-180°)
  HAL_GPIO_TogglePin(S1_GPIO_Port,S1_Pin); //Desativa S1
  HAL_GPIO_TogglePin(S2_GPIO_Port,S2_Pin); //Aciona S2
  delay_us(2774);
```

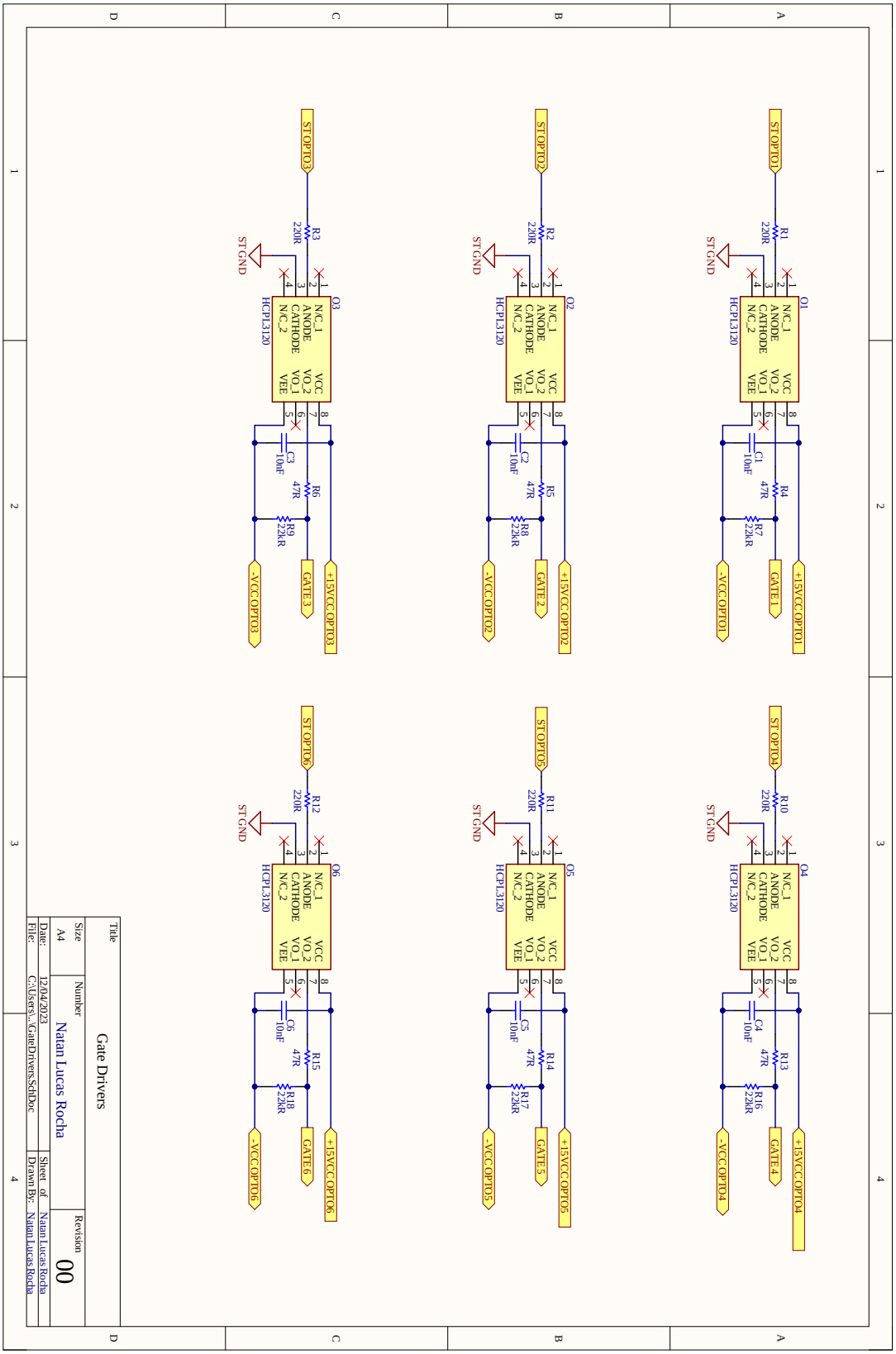
```
  //Quarto quadrante (180-240°)
  HAL_GPIO_TogglePin(S6_GPIO_Port,S6_Pin); //Desativa S6
  HAL_GPIO_TogglePin(S4_GPIO_Port,S4_Pin); //Aciona S4
  delay_us(2774);
  //Quinto quadrante (240-300°)
  HAL_GPIO_TogglePin(S2_GPIO_Port,S2_Pin); //Desativa S2
  HAL_GPIO_TogglePin(S3_GPIO_Port,S3_Pin); //Aciona S3
  delay_us(2774);
  //Sexto quadrante (300-360°)
  HAL_GPIO_TogglePin(S4_GPIO_Port,S4_Pin); //Desativa S4
  HAL_GPIO_TogglePin(S5_GPIO_Port,S5_Pin); //Aciona S5
  delay_us(2774);
  HAL_GPIO_TogglePin(S3_GPIO_Port,S3_Pin); //Desativa S3
  HAL_GPIO_TogglePin(S1_GPIO_Port,S1_Pin); //Aciona S1
  }
  /* USER CODE END 3 */
}

void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI_DIV2;
  RCC_OscInitStruct.PLL.PLLMUL = RCC_PLL_MUL3;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) != HAL_OK)
  {
    Error_Handler();
  }
}

/**
  * @brief TIM3 Initialization Function
```

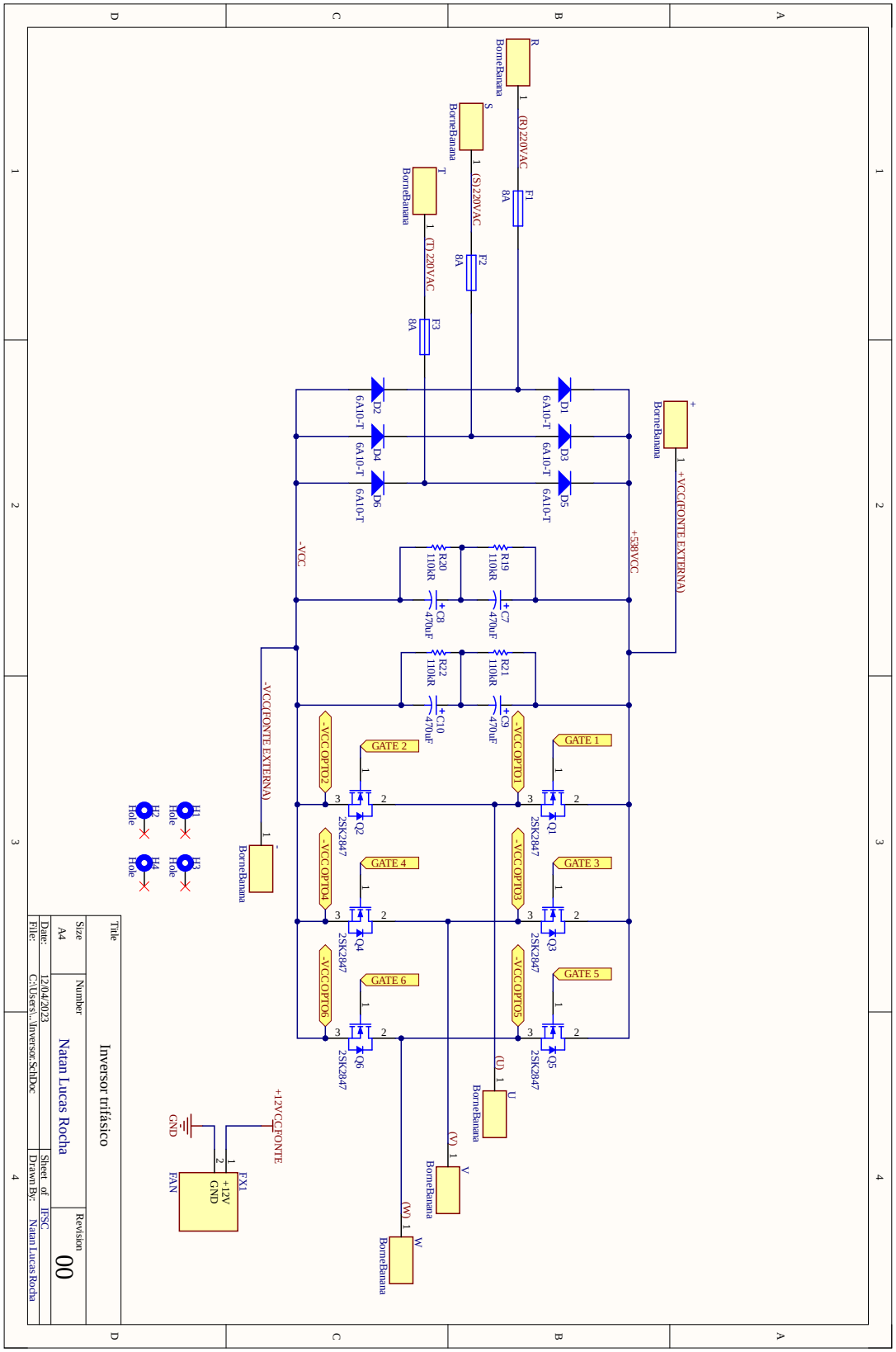
```
static void MX_TIM3_Init(void)
{
  /* USER CODE END TIM3_Init 0 */

  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};

  /* USER CODE END TIM3_Init 1 */
  htim3.Instance = TIM3;
  htim3.Init.Prescaler = 12-1;
  htim3.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim3.Init.Period = 0xffff-1;
  htim3.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim3.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim3) != HAL_OK)
  {
    Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim3, &sClockSourceConfig) != HAL_OK)
  {
    Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if (HAL_TIMEx_MasterConfigSynchronization(&htim3, &sMasterConfig) != HAL_OK)
  {
    Error_Handler();
  }

}

/**
  * @brief GPIO Initialization Function

static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};
/* USER CODE BEGIN MX_GPIO_Init_1 */
/* USER CODE END MX_GPIO_Init_1 */

  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOD_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOB_CLK_ENABLE();

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOA, S4_Pin|S2_Pin|S3_Pin, GPIO_PIN_RESET);

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(S5_GPIO_Port, S5_Pin, GPIO_PIN_SET);

  /*Configure GPIO pin Output Level */
```

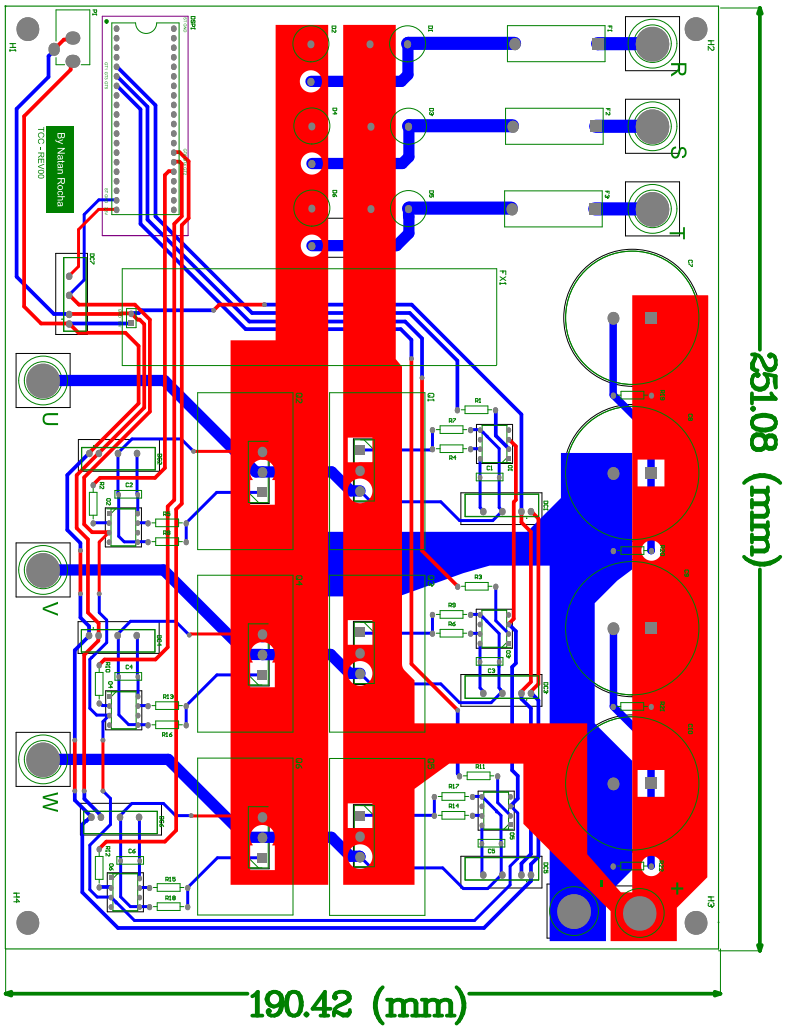
```
  HAL_GPIO_WritePin(S6_GPIO_Port, S6_Pin, GPIO_PIN_RESET);

  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(S1_GPIO_Port, S1_Pin, GPIO_PIN_SET);

  /*Configure GPIO pins : S4_Pin S1_Pin S2_Pin S3_Pin */
  GPIO_InitStruct.Pin = S4_Pin|S1_Pin|S2_Pin|S3_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

  /*Configure GPIO pins : S5_Pin S6_Pin */
  GPIO_InitStruct.Pin = S5_Pin|S6_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
  HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);

}

/**
  * @brief  This function is executed in case of error occurrence.

void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  while (1)
  {
  }

}

#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number

void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
}
#endif /* USE_FULL_ASSERT */
```

Apêndice E: Placa de circuito impresso - Protótipo




ST OPTO1
ST OPTO2
ST OPTO3
ST OPTO4
ST OPTO5
ST OPTO6
ST GND
R1 220R
R2 220R
R3 220R
R10 220R
R11 220R
R12 220R
O1
O2
O3
O4
O5
O6
HCPL3120
N.C_1
ANODE
CATHODE
N.C_2
VCC
VO_2
VO_1
VEE
C1 10nF
C2 10nF
C3 10nF
C4 10nF
C5 10nF
C6 10nF
R4 47R
R5 47R
R6 47R
R13 47R
R14 47R
R15 47R
R7 22kR
R8 22kR
R9 22kR
R16 22kR
R17 22kR
R18 22kR
-VCC OPTO1
-VCC OPTO2
-VCC OPTO3
-VCC OPTO4
-VCC OPTO5
-VCC OPTO6
GATE 1
GATE 2
GATE 3
GATE 4
GATE 5
GATE 6
+15VCC OPTO1
+15VCC OPTO2
+15VCC OPTO3
+15VCC OPTO4
+15VCC OPTO5
+15VCC OPTO6
Title
Gate Drivers
Size
A4
Number
Revision
00
Natan Lucas Rocha
Date:
12/04/2023
Sheet of
Natan Lucas Rocha
File:
C:\Users\...\GateDrivers.SchDoc
Drawn By:
Natan Lucas Rocha


R
S
T
(R) 220VAC
(S) 220VAC
(T) 220VAC
F1
F2
F3
8A
D1
D2
D3
D4
D5
D6
6A10-T
+VCC(FONTE EXTERNA)
-VCC
+538VCC
R19
R20
R21
R22
110RR
C7
C8
C9
C10
470uF
-VCC(FONTE EXTERNA)
GATE 1
GATE 2
GATE 3
GATE 4
GATE 5
GATE 6
-VCC OPTO1
-VCC OPTO2
-VCC OPTO3
-VCC OPTO4
-VCC OPTO5
-VCC OPTO6
Q1
Q2
Q3
Q4
Q5
Q6
25K2847
H1
H2
H3
H4
Hole
+12VCCFONTE
GND
FAN
FX1
+12V
GND
(U)
(V)
(W)
U
V
W
BorneBanana
Title
Inversor trifásico
Size
A4
Number
Natan Lucas Rocha
Revision
00
Date:
12/04/2023
Sheet of
IFSC
File:
C:\Users\...\Inversor.SchDoc
Drawn By:
Natan Lucas Rocha



Layer	Name	Material	Thickness	Constant	Gerber	Board Layer Stack
	Top Overlay					
1	Top Solder	Solder Resist	0.010mm	3.5	GTS	
	Top Layer		0.036mm		GTL	
	Dielectric 1	FR-4	1.60mm	4.8		
2	Bottom Layer		0.036mm		GBL	
	Bottom Solder	Solder Resist	0.010mm	3.5	GBS	
	Bottom Overlay				GBO	