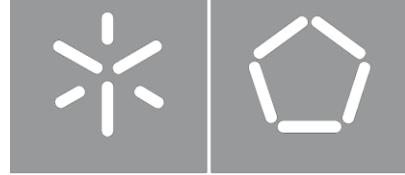




Universidade do Minho
Escola de Engenharia

Ricardo André Rodrigues Coelho

**Desenvolvimento de um Sistema de
Controlo Modular Descentralizado para
Conversores de Eletrónica de Potência em
Cascata**



Universidade do Minho
Escola de Engenharia

Ricardo André Rodrigues Coelho

**Desenvolvimento de um Sistema de
Controlo Modular Descentralizado para
Conversores de Eletrónica de Potência em
Cascata**

Dissertação de Mestrado
Engenharia Eletrónica Industrial e de Computadores
Eletrónica de Potência e Sistemas de Energia

Trabalho realizado sob orientação do
Professor Doutor José Gabriel Oliveira Pinto

DIREITOS DE AUTOR E CONDIÇÕES DE UTILIZAÇÃO DO TRABALHO POR TERCEIROS

Este é um trabalho académico que pode ser utilizado por terceiros desde que respeitadas as regras e boas práticas internacionalmente aceites, no que concerne aos direitos de autor e direitos conexos.

Assim, o presente trabalho pode ser utilizado nos termos previstos na licença abaixo indicada.

Caso o utilizador necessite de permissão para poder fazer um uso do trabalho em condições não previstas no licenciamento indicado, deverá contactar o autor, através do RepositóriUM da Universidade do Minho.

Licença concedida aos utilizadores deste trabalho



Atribuição-NãoComercial-SemDerivações
CC BY-NC-ND

<https://creativecommons.org/licenses/by-nc-nd/4.0/>

Agradecimentos

Esta dissertação é o culminar do meu percurso académico e não ficaria completa sem deixar os meus agradecimentos a todas as pessoas que contribuíram e me apoiaram durante estes anos. Agradeço a todos e em especial:

À minha família, à minha mãe Lídia, ao meu pai Ricardo, ao meu irmão Joca e a todos os que fazem parte dela, por toda a compreensão e apoio incondicional, fundamentais para o meu desenvolvimento.

À minha namorada Barbara Coelho por todo o carinho, paciência e confiança. Muito obrigado por acreditar sempre em mim e por teres estado ao meu lado durante os momentos de maior dificuldade.

Aos amigos de sempre, Marcelo Amaral, Gonçalo Saraiva, Rui Matos e Duarte Silva pela irmandade e companheirismo. Ao Rui Lacerda, Luís Varajão, Bruno Brasileiro, Mariana Morais, Inês Fernandes e Rita Sá pela amizade e por todos os bons momentos que passamos.

Aos meus caros amigos, João Marques, Armando Sarmento, Leonardo Sá, Simão Araújo e Bruno Ferreira pelo acompanhamento e pelas vivencias ao longo destes anos académicos.

À Universidade do Minho por me tornar melhor estudante, melhor pessoa, por me ter apresentado pessoas que levarei para o futuro, onde incluo professores e colegas, e por todas as experiências que fizeram destes os anos de maior evolução na minha vida pessoal.

Ao meu orientador Gabriel Pinto por todo o suporte dado ao longo deste projeto, pelas oportunidades concedidas, revisões da dissertação, sugestões e todo o conhecimento transmitido.

Por fim e não menos importante, um obrigado ao Luís Barros, pelos conhecimentos partilhados, pela disponibilidade e pela amizade. Obrigado por todo o esforço e dedicação, por fazer-me ser um melhor engenheiro e pelos conselhos que me guiaram ao longo deste caminho tão importante.

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração.

Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Resumo

Nas últimas décadas, tem-se assistido a uma crescente procura por novas fontes capazes de satisfazer as necessidades energéticas da sociedade. Essa demanda, aliada à evolução da eletrónica de potência, aumentou a necessidade de se fazer interface com aplicações de tensões e potências mais elevadas. Devido às limitações ainda existentes nas tensões de operação dos semicondutores, a utilização de conversores modulares surgiu como uma das principais soluções. Apesar das vantagens, estes conversores são operados com estratégias de controlo centralizadas, limitando a modularidade e expansibilidade do sistema devido às restrições relacionadas com o número periféricos disponíveis nos microcontroladores. Assim sendo, a aplicação de estratégias descentralizadas surge para eliminar essas limitações.

Na presente dissertação é descrito o Desenvolvimento de um Sistema de Controlo Modular Descentralizado para Conversores de Eletrónica de Potência em Cascata, bem como todas as fases relacionadas com o desenvolvimento deste projeto. Este sistema permite realizar a interface de barramentos CC com a rede elétrica. Para a sua criação, é apresentada uma topologia de três submódulos de eletrónica de potência ligados em cascata. Cada submódulo é constituído por um conversor CC-CA em ponte completa e um microcontrolador dedicado para o controlo do mesmo. Toda a gestão do sistema fica ao cargo de um microcontrolador externo. Para a realização deste projeto foi necessário o estudo, dimensionamento e implementação de todo o hardware que constitui o sistema, bem como a análise e desenvolvimento de todos os algoritmos de controlo e das topologias de comunicação utilizadas.

Ao longo do desenvolvimento desta dissertação, foi possível validar a modularidade do sistema de controlo descentralizado e a adaptabilidade dos algoritmos, bem como a capacidade do sistema injetar energia na rede elétrica, atingindo assim, os objetivos propostos.

Palavras-Chave: Controlo de Corrente, Conversores de Potência em Cascata, Rede Elétrica, Sistema de Controlo Descentralizado.

Abstract

In recent decades, there has been a growing demand for new sources of energy capable of meeting society's needs. This demand, combined with the evolution of electronic power, increase the need to interface with high power applications. Taking into consideration the restrictions of semiconductors operating voltage, the use of modular converters has emerged as one of the main solutions. Despite the advantages, these converters are operated with centralized control strategies, limiting the modularity and expandability of the system due to the restrictions related to the number of peripherals available in the microcontrollers. Therefore, the application of decentralized strategies appears to eliminate these limitations.

The present dissertation describes the Development of a Decentralized Modular Control System for Cascade Power Electronic Converters, as well as the phases related to the development of this project. This system allows the interface between DC buses and the electrical network. For its development, a topology of three power submodules connected in cascade is presented. Each submodule consists of a full-bridge DC-AC converter and a dedicated microcontroller. An external microcontroller is in charge of system management. To carry out this project, it was necessary to study, design and implement the hardware, as well as the analysis and development of control algorithms and communication topologies.

Throughout the development of this dissertation, it was possible to validate the modularity of the decentralized control system and the adaptability of the algorithms, as well as the capacity of the system to inject energy into the electrical network, thus achieving the proposed objectives.

Keywords: Current Control, Cascading Power Converters, Electric Grid, Decentralized Control System.

ÍNDICE

Agradecimentos	iv
Resumo	vi
Abstract.....	vii
Lista de Figuras.....	xi
Lista de Tabelas.....	xv
Acrónimos e Siglas	xvi
Nomenclatura	xix
Capítulo 1 Introdução.....	1
1.1 Evolução da Eletrónica de Potência	1
1.2 Principais Aplicações de MMCC	2
1.2.1 Ferrovias	2
1.2.2 Eólicas	3
1.2.3 Solares.....	3
1.3 Enquadramento e Motivações.....	4
1.4 Objetivos e Contribuições	5
1.5 Organização e Estrutura do Relatório	5
Capítulo 2 Configurações de MMCC e Sistemas de Controlo Descentralizados	7
2.1 Introdução.....	7
2.2 Conversores Modulares Multinível.....	8
2.3 Sistemas de Controlo Descentralizado.....	9
2.3.1 Meio de Transmissão	10
2.3.2 Topologias de Comunicação	12
2.3.3 Protocolos de Comunicação	14
2.4 Conclusões	16
Capítulo 3 Conversores de Eletrónica de Potência e Algoritmos de Controlo	18
3.1 Introdução.....	18
3.2 Topologias de Submódulos para MMCC	19
3.2.1 Meia Ponte.....	19
3.2.2 Ponte Completa	20
3.2.3 Neutral Point Clamped.....	21
3.2.4 T-type Neutral Point Clamped	22
3.2.5 Comparação das várias topologias	23
3.3 Técnicas de SPWM	24
3.3.1 Modulação SPWM com Deslocamento de Nível	25
3.3.2 Modulação SPWM com Deslocamento de Fase	26
3.4 Técnicas de Controlo de Corrente.....	27
3.4.1 Controlo de Corrente por Histerese	27

3.4.2	Controlo de Corrente por <i>Periodic Sampling</i>	28
3.4.3	Controlo de Corrente Proporcional Integral com SPWM.....	29
3.4.4	Controlo de Corrente Preditivo com SPWM	29
3.5	Algoritmo de Regulação do Barramento CC	31
3.6	Algoritmos de Sincronização com a Rede Elétrica	31
3.7	Conclusões	33
Capítulo 4	Simulações Computacionais da Topologia Utilizada.....	35
4.1	Introdução.....	35
4.2	Topologia Utilizada.....	35
4.3	Simulações Computacionais.....	37
4.3.1	Resultados de Simulação para Três Submódulos	39
4.3.2	Resultados de Simulação para Dois Submódulos	46
4.4	Conclusões	50
Capítulo 5	Implementação de um Protótipo Laboratorial	52
5.1	Introdução.....	52
5.2	Hardware de Potência.....	53
5.2.1	Conversores de Potência CC-CA.....	54
5.2.2	Círculo de Acoplamento com a Rede Elétrica e Proteções.....	56
5.3	Hardware de Controlo.....	59
5.3.1	Digital Signal Controller (DSC).....	60
5.3.2	<i>Docking Station</i> da LAUNCHXL-F28379D	61
5.3.3	Placa de <i>Driver</i>	67
5.3.4	Sensor de Tensão	69
5.3.5	Sensor de Corrente	71
5.4	Conclusões	73
Capítulo 6	Princípio de Funcionamento do Protótipo Laboratorial e Resultados Experimentais	75
6.1	Introdução.....	75
6.2	Princípio de Funcionamento do Protótipo Laboratorial	76
6.3	Resultados Experimentais	79
6.3.1	Algoritmo de Sincronização com a Rede Elétrica.....	79
6.3.2	Conversor com Três Submódulos	81
6.3.3	Conversor com Dois Submódulos	93
6.4	Conclusões	101
Capítulo 7	Conclusões e Sugestões de Trabalho Futuro	103
7.1	Conclusões	103
7.2	Sugestões de Trabalho Futuro	107
Lista de Referências	110	
Apêndice	120	
Apêndice A Conversor de Eletrónica de Potência	121	
Apêndice B <i>Docking Station</i> da LAUNCHXL-F28379D	123	
Apêndice C <i>Driver</i> ADUM3223	125	
Apêndice D <i>Driver</i> ADUM4135.....	127	
Apêndice E Código implementado para o <i>Master</i> (<i>main.c</i>)	129	

Apêndice F Código implementado para o <i>Master</i> (<i>Peripheral_Setup.h</i>)	140
Apêndice G Código implementado para o <i>Master</i> (<i>Peripheral_Setup.c</i>)	141
Apêndice H Código implementado para o <i>Master</i> (<i>Function_Definition.h</i>).....	149
Apêndice I Código implementado para o <i>Master</i> (<i>Function_Definition.c</i>).....	150
Apêndice J Código implementado para os <i>Slaves</i> (<i>main.c</i>).....	155
Apêndice K Código implementado para os <i>Slaves</i> (<i>Peripheral_Setup.h</i>)	164
Apêndice L Código implementado para os <i>Slaves</i> (<i>Peripheral_Setup.c</i>)	165
Apêndice M Código implementado para os <i>Slaves</i> (<i>Function_Definition.h</i>).....	173
Apêndice N Código implementado para os <i>Slaves</i> (<i>Function_Definition.c</i>)	174

Lista de Figuras

Figura 2.1 – Classificação de Conversores de Fonte de Tensão [34].....	9
Figura 2.2 – Esquema de blocos de uma rede com comunicação ligada em estrela.	13
Figura 2.3 – Esquema de blocos de uma rede de comunicação ligada em barramento.....	13
Figura 2.4 – Esquema de blocos de uma rede de comunicação ligada em anel.	14
Figura 3.1 – Esquema de um conversor CC-CA em meia ponte: (a) barramento dividido; (b) barramento único.	20
Figura 3.2 – Esquema de um conversor CC-CA em ponte completa.....	21
Figura 3.3 – Esquema de um conversor CC-CA em topologia NPC.	22
Figura 3.4 – Esquema de um conversor CC-CA em topologia T-NPC.	23
Figura 3.5 – Técnicas de SPWM com deslocamento de nível: (a) PD, (b) POD, (c) APOD.	26
Figura 3.6 – Técnica de SPWM com deslocamento de fase (PSC).....	26
Figura 3.7 – Esquema de blocos da técnica de controlo de corrente por histerese [92].....	28
Figura 3.8 – Esquema de blocos da técnica de controlo de corrente por <i>periodic sampling</i> [92].....	29
Figura 3.9 – Esquema de blocos de um controlador de corrente por PI com SPWM [92].....	29
Figura 3.10 – Esquema de blocos de um controlador de corrente preditivo com SPWM [92].....	30
Figura 3.11 – Modelo elétrico do sistema [92].	30
Figura 3.12 – Diagrama de blocos da PLL básica.	33
Figura 3.13 – Diagrama de blocos da PLL aprimorada.	33
Figura 4.1 – Diagrama de blocos da topologia utilizada.	36
Figura 4.2 – Modelo de simulação da parte de potência da topologia utilizada.....	37
Figura 4.3 – Modelo de simulação da parte de controlo da topologia utilizada.....	38
Figura 4.4 – Interface do PSIM para os parâmetros de uma forma de onda triangular.	40
Figura 4.5 – Resultado da comunicação em anel para um sistema com três submódulos ligados.	41
Figura 4.6 – Valores calculados para o desfasamento de 6 ondas portadoras.	41
Figura 4.7 – Formas de onda do sinal de entrada da EPLL (v_{rede}) dividido por 325 e sinal de saída (pll_{uni}).	42
Figura 4.8 – Formas de onda da corrente de referência (i_{ref}), do sinal unitário da EPLL (pll_{uni}) e da amplitude máxima (i_{amp}) da referência.	42
Figura 4.9 – Formas de onda da corrente de referência (i_{ref}) e da corrente sintetizada pelo inversor (i_{rede}).....	43
Figura 4.10 – Formas de onda da tensão à saída do inversor (v_{inv}), da tensão da rede elétrica (v_{rede}) e da corrente sintetizada pelo inversor (i_{rede}) multiplicada por 50.....	43
Figura 4.11 – Formas de onda da tensão do barramento CC no primeiro, segundo e terceiro submódulos (V_{cc1} , V_{cc2} e V_{cc3}) e da tensão à saída do inversor (v_{inv}).	44
Figura 4.12 – Formas de onda da potência no primeiro, segundo e terceiro submódulos (P_1 , P_2 , P_3) e da potência total do sistema (P_{rede}).	44
Figura 4.13 – Formas de onda: (a) corrente de referência (i_{ref}) e corrente sintetizada pelo inversor (i_{rede}); (b) tensão à saída do inversor (v_{inv}), tensão da rede elétrica (v_{rede}) e corrente sintetizada pelo inversor (i_{rede})	

multiplicada por 50; (c) potência no primeiro, segundo e terceiro submódulos (P_1 , P_2 , P_3) e potência total do sistema (P_{rede}).....	45
Figura 4.14 – Resultado da comunicação em anel para um sistema com dois submódulos ligados.....	46
Figura 4.15 – Valores calculados para o desfasamento de 4 ondas portadoras.....	47
Figura 4.16 – Formas de onda da corrente de referência (i_{ref}) e da corrente sintetizada pelo inversor (i_{rede}).....	47
Figura 4.17 – Formas de onda da tensão à saída do inversor (v_{inv}), da tensão da rede elétrica (v_{rede}), da corrente sintetizada pelo inversor (i_{rede}) multiplicada por 50 e da tensão do barramento CC no primeiro e segundo submódulos (V_{cc1} , V_{cc2}).	48
Figura 4.18 – Formas de onda da potência no primeiro e segundo submódulos (P_1 e P_2) e da potência total do sistema (P_{rede}).....	48
Figura 4.19 – Formas de onda: (a) corrente de referência (i_{ref}) e corrente sintetizada pelo inversor (i_{rede}); (b) tensão à saída do inversor (v_{inv}), tensão da rede elétrica (v_{rede}) e corrente sintetizada pelo inversor (i_{rede}) multiplicada por 50; (c) potência no primeiro, segundo e terceiro submódulos (P_1 , P_2) e potência total do sistema (P_{rede}).	49
Figura 5.1 – Protótipo laboratorial implementado.....	52
Figura 5.2 – Protótipo laboratorial legendado.....	53
Figura 5.3 – Placa do conversor CC-CA.....	54
Figura 5.4 – Circuitos de proteção implementados nos MOSFET.....	55
Figura 5.5 – Esquema elétrico do circuito de acoplamento com a rede elétrica.....	56
Figura 5.6 – Aparelhos de proteção do circuito de acoplamento com a rede elétrica.....	57
Figura 5.7 – Acionamento do contactor: (a) esquema elétrico; (b) botão de start e botoneira de emergência. .	58
Figura 5.8 – Resistências de pré-carga: (a) vista exterior da caixa que contém as resistências; (b) vista interior das resistências de $50\ \Omega$ em paralelo.....	58
Figura 5.9 – Bobina de acoplamento com a rede elétrica.	58
Figura 5.10 – Placa de distribuição das alimentações.....	59
Figura 5.11 – LAUNCHXL-F28379D da <i>Texas Instruments</i>	60
Figura 5.12 – <i>Docking Station</i> da LAUNCHXL-F28379D: (a) integração do <i>master</i> ; (b) integração do <i>slave</i>	61
Figura 5.13 – Esquema elétrico: (a) circuito de condicionamento de sinal; (b) circuito auxiliar.	63
Figura 5.14 – Esquema elétrico do circuito de deteção de erros.....	64
Figura 5.15 – Esquema elétrico do: (a) circuito de memorização de erros; (b) circuito de comando.	65
Figura 5.16 – Circuito de aplicação dos integrados MAX9122 e MAX9123.	66
Figura 5.17 – <i>Docking Station</i> com LAUNCHXL-F28379D integrada.	67
Figura 5.18 – Placa de <i>driver</i> desenvolvida.	68
Figura 5.19 – Esquema elétrico do circuito de configuração das resistências de <i>gate</i>	68
Figura 5.20 – Esquema elétrico do circuito de ligação recomendado para o sensor LV 25-P.....	69
Figura 5.21 – Placa de adaptação do sensor de tensão.	70
Figura 5.22 – Gráfico da linearidade do sensor de tensão, onde são comparados os valores reais medidos aos valores teóricos calculados.	71
Figura 5.23 – Esquema elétrico do circuito de ligação recomendado para o sensor LA 55-P.....	71
Figura 5.24 – Placa de adaptação do sensor de corrente.	72
Figura 5.25 – Gráfico da linearidade do sensor de corrente, onde são comparados os valores reais medidos aos valores teóricos calculados.	73

Figura 6.1 – Diagramas das máquinas de estados implementadas para o controlo de: (a) <i>master</i> ; (b) <i>slaves</i>	76
Figura 6.2 – Resultado experimental do sincronismo com a rede elétrica.	79
Figura 6.3 – Resultado experimental do sincronismo com a rede elétrica em regime permanente.	80
Figura 6.4 – Valor das expressões do <i>master</i> durante o estágio de contagem.	81
Figura 6.5 – Resultado experimental das comunicações em anel durante o estágio de contagem.	82
Figura 6.6 – Valor das expressões do <i>master</i> durante o estágio de configuração.	83
Figura 6.7 – Resultado experimental das comunicações em anel durante o estágio de configuração.	83
Figura 6.8 – Valor das expressões do <i>master</i> durante o estágio de recolha.	84
Figura 6.9 – Resultado experimental das comunicações em anel durante o estágio de configuração.	84
Figura 6.10 – Resultados experimentais da comunicação em barramento: (a) tempo de transmissão; (b) tempo de leitura.	85
Figura 6.11 – Resultados experimentais do desfasamento dos sinais de PWM: (a) para os primeiros quatro braços; (b) para os primeiro, segundo, quinto e sexto braços.	86
Figura 6.12 – Resultados experimentais da placa de <i>driver</i>	86
Figura 6.13 – Resultados experimentais das tensões nos barramentos e à saída do inversor.	87
Figura 6.14 – Resultados experimentais do controlo de corrente preditivo para diferentes amplitudes: (a) 1 A; (b) 2 A; (c) 3 A; (d) 4 A.	88
Figura 6.15 – Resultado experimental do controlo de corrente preditivo para os sinais de saída do DAC com uma amplitude de corrente de 2 A.	89
Figura 6.16 – Resultados experimentais do controlo de corrente preditivo na transição ascendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.	89
Figura 6.17 – Resultados experimentais do controlo de corrente preditivo na transição descendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.	90
Figura 6.18 – Resultados experimentais do controlo de corrente preditivo para a potência no: (a) primeiro submódulo; (b) segundo submódulo.	90
Figura 6.19 – Resultados experimentais do controlo de corrente PI para diferentes amplitudes: (a) 1 A; (b) 2 A; (c) 3 A; (d) 4 A.	91
Figura 6.20 – Resultados experimentais do controlo de corrente preditivo na transição ascendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.	92
Figura 6.21 – Resultados experimentais do controlo de corrente preditivo na transição descendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.	92
Figura 6.22 – Resultados experimentais do controlo de corrente PI para a potência no: (a) primeiro submódulos; (b) segundo submódulo.	93
Figura 6.23 – Valor das expressões do <i>master</i> durante o estágio de contagem.	94
Figura 6.24 – Resultado experimental das comunicações em anel durante o estágio de contagem.	94
Figura 6.25 – Valor das expressões do <i>master</i> durante o estágio de configuração.	95
Figura 6.26 – Resultado experimental das comunicações em anel durante o estágio de configuração.	95
Figura 6.27 – Valor das expressões do <i>master</i> durante o estágio de recolha.	96
Figura 6.28 – Resultado experimental das comunicações em anel durante o estágio de configuração.	96
Figura 6.29 – Resultados experimentais do desfasamento dos sinais de PWM para os quatro braços do sistema.	97
Figura 6.30 – Resultados experimentais das tensões nos barramentos e à saída do inversor.	98

Figura 6.31 – Resultados experimentais do controlo de corrente preditivo para diferentes amplitudes: (a) 1 A; (b) 2 A; (c) 3 A; (d) 4 A.....	99
Figura 6.32 – Resultado experimental do controlo de corrente preditivo para os sinais de saída do DAC com uma amplitude de corrente de 2 A.....	99
Figura 6.33 – Resultados experimentais do controlo de corrente preditivo para a potência no: (a) primeiro submódulo; (b) segundo submódulo.....	100
Figura 6.34 – Resultados experimentais do controlo de corrente preditivo na transição ascendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.....	100
Figura 6.35 – Resultados experimentais do controlo de corrente preditivo na transição descendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.....	101
Figura 7.1 – Placa de <i>driver</i> desenvolvida para o ADUM4135.	109

Lista de Tabelas

Tabela 1.1 – Dispositivos semicondutores de potência [6].....	2
Tabela 3.1 – Estados de operação num conversor CC-CA em meia ponte com ponto médio.....	20
Tabela 3.2 – Estados de operação num conversor CC-CA em meia ponte sem ponto médio.	20
Tabela 3.3 – Estados de operação num conversor CC-CA em ponte completa.	21
Tabela 3.4 – Estados de operação num conversor CC-CA na topologia NPC.	22
Tabela 3.5 – Estados de operação num conversor CC-CA na topologia TNPC.	23
Tabela 3.6 – Comparação das diferentes topologias [67].	24
Tabela 4.1 – Características nominais do conversor modular multinível em cascata.	39
Tabela 5.1 – Valores das resistências de entrada e medida, utilizadas nos sensores de tensão.	70
Tabela 6.1 – Estrutura das tramas de comunicação.	77

Acrónimos e Siglas

Acrónimo / Sigla	Significado
ADC	<i>Analog to Digital Converter</i> Conversor Analógico-Digital
APOD	<i>Alternate Phase Opposition Disposition</i> Disposição de Oposição de Fase Alternada
CA	<i>Alternating Current</i> Corrente Alternada
CAN	<i>Controller Area Network</i> Rede de Área do Controlador
CC	<i>Constant Current</i> Corrente Contínua
CLA	<i>Control Law Accelerators</i> Aceleradores de Leis de Controle
CPU	<i>Central Processing Unit</i> Unidade de Processamento Central
DAC	<i>Digital to Analog Converter</i> Conversor Digital-Analógico
DSC	<i>Digital Signal Controller</i> Controlador Digital de Sinais
EPLL	<i>Enhanced Phase-Locked Loop</i> Malha de Deteção de Fase Aprimorada

FC	<i>Flying Capacitor</i> Condensador Flutuante
GEPE	<i>Power Electronics and Energy Group</i> Grupo de Eletrónica de Potência e Energia
GPIO	<i>General Purpose Input/Output</i> Entradas/Saídas de Uso Geral
HVDC	<i>High Voltage Direct Current</i> Corrente Contínua de Alta Tensão
I2C	<i>Inter-Integrated Circuit</i> Circuito Inter-integrado
IGBT	<i>Insulated Gate Bipolar Transistor</i> Transístor Bipolar de Porta Isolada
LS-PWM	<i>Level-Shift PWM</i> PWM com Deslocamento de Nível
MMC	<i>Modular Multilevel Converter</i> Conversor Modular Multinível
MMCC	<i>Modular Multilevel Cascade Converter</i> Conversor Modular Multinível em Cascata
MOSFET	<i>Metal-Oxide-Semiconductor Field-Effect Transistor</i> Transístor de Efeito de Campo Óxido-Metal Semicondutor
NPC	<i>Neutral Point Clamped</i> Ponto Neutro Fixo
PCB	<i>Printed Circuit Board</i> Placa de Circuito Impresso
PD	<i>Phase Disposition</i> Disposição de Fase
PI	<i>Proportional Integral</i> Proporcional Integral

PLL	<i>Phase Locked Loop</i> Malha de Deteção de Fase
POD	<i>Phase Opposition Disposition</i> Disposição de Oposição de Fase
PSC	<i>Phase-Shift Carrier</i> PWM com Deslocamento de Fase
PSIM	<i>Power Simulation</i> Simulação de Potência
PWM	<i>Pulse Width Modulation</i> Modulação por Largura de Pulso
SPWM	<i>Sinusoidal Pulse Width Modulation</i> Modulação Sinusoidal de Largura de Pulso
THD	<i>Total Harmonic Distortion</i> Distorção Harmónica Total
TNPC	<i>T-type Neutral Point Clamped</i> Ponto Neutro Fixo do Tipo T
VSC	<i>Voltage Source Converter</i> Conversor Fonte de Tensão
ZOH	<i>Zero-Order Hold</i> Retentor de Ordem Zero

Nomenclatura

i_{amp}	Amplitude de corrente de saída desejada	A
i_{rede}	Corrente de saída do conversor	A
i_{ref}	Corrente de referência	A
i_1	Corrente na fonte de alimentação do primeiro submódulo	A
i_2	Corrente na fonte de alimentação do segundo submódulo	A
i_{d_ref}	Tensão do DAC que representa a corrente de referência	V
i_{d_rede}	Tensão do DAC que representa a corrente de saída	V
v_{rede}	Tensão da rede elétrica	V
v_{inv}	Tensão de saída do conversor	V
V_{cc1}	Tensão no barramento CC do primeiro submódulo	V
V_{cc2}	Tensão no barramento CC do segundo submódulo	V
V_{cc3}	Tensão no barramento CC do terceiro submódulo	V
$v_{trigger}$	Tensão do sinal de deteção da transição descendente de amplitude	V
v_{dac}	Tensão da rede elétrica à saída do DAC	V
v_{pll}	Tensão gerada pela EPLL à saída do DAC	V
P_{rede}	Potência ativa injetada na rede elétrica	W
P_1	Potência ativa do primeiro submódulo	W
P_2	Potência ativa do segundo submódulo	W
P_3	Potência ativa do terceiro submódulo	W
f_a	Frequência de amostragem	Hz
f_c	Frequência de comutação	Hz
$THD\%$	Taxa de distorção harmónica percentual total da amplitude fundamental do sinal	%
com_b	Ligações da topologia de comunicação em barramento	-
com_p	Ligações da topologia de comunicação em anel	-
k_p	Ganho proporcional	-
k_i	Ganho integral	-

Capítulo 1

Introdução

1.1 Evolução da Eletrónica de Potência

A eletrónica de potência é uma área tecnológica que recorre a dispositivos semicondutores e componentes elétricos para controlar o fluxo de energia entre as fontes de energia elétrica e as cargas. Esse controlo do fluxo de energia é normalmente feito de corrente alternada (CA) para corrente contínua (CC) ou vice-versa, onde os parâmetros controlados são tensão, corrente e frequência.

O retificador de vapor de mercúrio, inventado em 1902 por Peter Cooper Hewitt, marcou o início da eletrónica de potência, permitindo converter corrente alternada em corrente contínua [1]. Durante décadas a pesquisa sobre válvulas de mercúrio e transmissão de energia continuou. Até que, em 1947 Walter H. Brattain e John Bardeen, sob a direção de William Shockley nos *Bell Labs*, apresentaram o primeiro semicondutor amplificador, chamado de transístor [2]. Uns meses depois, em 1948, William Shockley inventa o transístor baseado na junção p-n, esta que foi descoberta por Russell Ohl em 1940 [3, 4]. Na década de 50 o primeiro retificador controlado de silício foi proposto pela *Bell Labs* e produzido comercialmente pela *General Electric* [5]. Estes acontecimentos revolucionaram a eletrónica de potência. Desde então, diversos tipos de dispositivos semicondutores e técnicas de conversão foram e continuam a ser desenvolvidos.

A grande parte das topologias de conversores de energia está disponível sob a forma de produtos padrão para tensões entre os 2,3 kV e os 13,8 kV [6]. Para a operação em alta tensão é necessário utilizar um transformador ou dispositivos semicondutores de alta tensão. O uso do transformador para elevar a tensão aumenta o custo e tamanho do sistema. A outra solução não aumenta o tamanho do conversor, porém está limitada quanto à disponibilidade de dispositivos semicondutores de alta tensão. Alguns dos semicondutores disponíveis e as suas classificações de tensão e corrente são mostrados na Tabela 1.1 [6, 7].

Tabela 1.1 – Dispositivos semicondutores de potência [6].

Parâmetro	Díodo de Potência	Tiristor	GTO	SGCT	IGBT
Tensão	8,5 kV @ 1,2 kA	12 kV @ 1,5 kA	6 kV @ 6 kA	10 kV @ 1,7 kA	6,5 kV @ 0,75 kA
Corrente	9,6 kA @ 1,8 kV	5 kA @ 0,4 kV	6 kA @ 6 kV	5 kA @ 4,5 kV	2,4 kA @ 1,7 kV

Uma alternativa passa por conectar os semicondutores de média tensão em série para aumentar a tensão operacional dos conversores de energia. Esta solução permite combater as desvantagens da anterior, contudo novos problemas são criados. Tanto os semicondutores conectados em série como os seus *gate drivers* podem não apresentar desempenho estático e dinâmico semelhante, e a tensão total do sistema pode não ser igualmente partilhada pelos semicondutores. Isto significa que é necessário um circuito adicional para equilibrar a tensão durante o modo de bloqueio, aumentando as perdas de energia no conversor. Além disso, a conexão em serie não melhora a qualidade da forma de onda da corrente e a tensão de saída [8]. Para combater todos as limitações dos semicondutores e da conexão em série, uma abordagem modular foi desenvolvida. Nesta abordagem, denominada de conversor modular multinível em cascata (MMCC - *Modular Multilevel Cascade Converter*), são conectados submódulos de baixa tensão idênticos em cascata para obter uma tensão de operação superior [9].

1.2 Principais Aplicações de MMCC

Algumas das aplicações dos conversores MMCC encontradas baseiam-se sobretudo nas ferrovias e energias renováveis, devido a serem aplicações para elevados níveis de potência. Para além disso, são áreas de grande interesse no que diz respeito à gestão e produção dos recursos energéticos, tentando dessa forma mitigar a falta de energia e os problemas da poluição ambiental. Dessa forma, são apresentadas, com mais detalhe, as principais aplicações dos MMC.

1.2.1 Ferrovias

As ferrovias representam um dos mais populares e usados transportes públicos da atualidade. Devido à sua importância, tem se investigado sobre a utilização de novas técnicas e tecnologias para a construção das linhas ferroviárias [10]. Assim sendo, o MMCC é apresentado como uma solução para a interface entre as catenárias, permitindo realizar uma

conexão direta sem a necessidade de transformadores complicados e com um melhor desempenho [11]. Estes atuam como um condicionador ferroviário de potência, designado na nomenclatura como *Railway Power Conditioner* (RPC), que realiza o controlo ativo do fluxo de energia [12]. Dessa forma, equilibra as correntes de alimentação do transformador e compensa as correntes harmónicas de carga [13].

1.2.2 Eólicas

A produção de energia a partir do vento, energia eólica, é uma das tecnologias de fontes renováveis com maior maturidade e desenvolvimento. Com a sua evolução, as dimensões dos parques eólicos tornaram-se cada vez maiores [14, 15]. Portanto, a tendência para o futuro consiste na energia eólica de longa distância e grande capacidade, como é o exemplo dos parques eólicos *offshore* [16].

A transmissão de alta tensão de longa distância entre *offshore* e *onshore* é um novo desafio no sistema de energia *offshore*. Assim, é fundamental um projeto capaz de reduzir o peso e o tamanho dos componentes, os custos e tempos de manutenção e as perdas de energia na turbina. Logo, os MMCC com controlo descentralizado são uma boa solução para estes requisitos, como apresentado em [17 - 20].

1.2.3 Solares

Ao longo dos últimos anos, assistiu-se ao crescente investimento e desenvolvimento da energia solar fotovoltaica. Isto permitiu o aumento da eficiência dos painéis, bem como reduzir o custo da produção de energia, tornando o sistema fotovoltaico numa das principais fontes renováveis do mundo [21].

O sistema fotovoltaico baseado em MMCC é uma das novas configurações para integração de centrais fotovoltaicas de grande escala com a rede elétrica [22]. Esta configuração elimina os grandes transformadores de interface com a rede, possui menores requisitos de filtragem, fornece melhor qualidade de energia, reduz os custos de produção e melhora a fiabilidade devido à sua modularidade [23, 24]. Em [21 – 24] são apresentados exemplos da utilização de MMCC em sistemas fotovoltaicos.

1.3 Enquadramento e Motivações

A energia representa atualmente um bem essencial para a vida humana, sendo um suporte indispensável à atividade económica e ao conforto da população em geral. Todavia, o aumento das necessidades energéticas aliado à utilização de fontes de energia não renováveis, como é o caso da utilização de combustíveis fosseis, tem provocado, com o passar do tempo, sérios problemas ambientais. Por essa razão, a procura por novas fontes não poluentes e tecnologias que permitam o máximo aproveitamento das fontes já existentes é de extrema importância.

Cada vez mais, existe a necessidade de fazer interface com aplicações de tensões e potências elevadas. Contudo, há limitações nas tensões de operação que os semicondutores conseguem suportar. Assim sendo, é necessário utilizar soluções modulares, ou seja, dividir o sistema em vários submódulos, tornando-o mais fiável e robusto [25]. Para além disso, outro problema está inerente à solução anterior. Nos sistemas de controlo, a nível de microcontroladores, existem limitações relacionadas com o número de GPIO (*General Purpose Input/Output*) disponíveis. Ou seja, a utilização de um controlo centralizado, baseado apenas num microcontrolador, não é viável.

Os conversores modulares multinível com controlo descentralizado são, portanto, conversores que permitem descentralizar o controlo do sistema, isto é, cada submódulo passa a ter o seu controlo dedicado. Por essa razão, são apresentados como solução para a interface com sistemas de muito alta tensão, como é o caso das fontes de energia renovável e dos sistemas ferroviários.

Estes tipos de conversores caracterizam-se por terem alta frequências de comutação, o que permite desenvolver eletrónica de potência mais compacta, considerando os níveis de potência e de tensão que são tratados. Em caso de falha de um dos submódulos, o conversor pode operar com um nível de energia reduzido e, além disso, é fácil localizar a falha no sistema. Não obstante, é possível adicionar submódulos auxiliares que entram em funcionamento em caso de falha de algum dos outros submódulos, criando mecanismos redundantes de proteção. Com esta topologia, a potência total é igualmente dividida pelos submódulos, o que reduz a tensão e, consequentemente, o preço de cada componente, minimizando o custo total do sistema [17, 25].

1.4 Objetivos e Contribuições

Esta dissertação tem como principal objetivo o desenvolvimento de um sistema de controlo modular descentralizado para conversores de eletrónica de potência em cascata. Para tal, pretende-se construir um protótipo laboratorial composto por três submódulos, sendo que, cada um deverá possuir inteligência própria. Assim, os objetivos da presente dissertação são:

- Levantamento do estado da arte sobre conversores modulares multinível em cascata, sistemas de controlo descentralizado, topologias de submódulos, técnicas de modulação PWM (*Pulse Width Modulation*) e de controlo de corrente e algoritmos de regulação do barramento CC e de sincronização com a rede elétrica;
- Estudo, com recurso a simulações computacionais utilizando o software PSIM (*Power Simulation*), das topologias e algoritmos de controlo a implementar no sistema proposto;
- Desenvolvimento do hardware de potência inerente a três submódulos que constituem o conversor modular multinível. Cada submódulo composto por um barramento CC e um conversor CC-CA.
- Desenvolvimento de um sistema de controlo modular descentralizado. Onde cada submódulo tem um microcontrolador dedicado (*slave*) e a gestão dos vários submódulos fica a cargo de um outro microcontrolador (*master*).
- Desenvolvimento de um sistema de comunicação que possibilita a troca de informações entre os microcontroladores (*master* e *slaves*).
- Integração de todos os objetivos anteriormente referidos e realização de ensaios experimentais de forma a verificar e validar o desempenho do sistema proposto.

1.5 Organização e Estrutura do Relatório

Referente à escrita desta dissertação, esta encontra-se dividida em sete capítulos, sendo estes descritos em maior detalhe nos parágrafos seguintes.

No Capítulo 1 é apresentada uma breve história sobre a evolução da eletrónica de potência e são apresentadas as principais aplicações dos MMC. Além disso, é feita uma pequena abordagem ao problema em questão, enquadrando brevemente os sistemas descentralizados

aplicados aos conversores modulares multinível, e são apresentados os objetivos e estrutura da dissertação.

No Capítulo 2 é feita uma breve introdução aos conversores modulares multinível, sendo procedido de uma análise dos sistemas de controlo descentralizado, em particular, dos meios de transmissão, topologias e protocolos utilizados para comunicações.

No Capítulo 3 é realizado um estudo do estado da arte sobre os sistemas constituintes dos conversores modulares multinível em cascata, nomeadamente, as topologias de submódulos existentes, as técnicas de PWM necessárias para controlar a tensão de saída do conversor, as técnicas de controlo de corrente utilizadas para a sintetização de uma corrente sinusoidal e injeção de energia na rede elétrica, os algoritmos de regulação dos barramentos CC e os algoritmos de sincronização com a rede elétrica mais comuns.

No Capítulo 4 é apresentada a topologia utilizada, sendo feita uma breve descrição da mesma. Além disso, são mostrados os modelos e resultados das simulações computacionais, realizadas com recurso ao software PSIM, permitindo analisar e validar a topologia e os algoritmos de controlo selecionados.

No Capítulo 5 é detalhado todo o dimensionamento e implementação do hardware desenvolvido para o protótipo laboratorial, tanto do sistema de potência, como do sistema de controlo.

No Capítulo 6 são mostrados os resultados experimentais do hardware e software, validando a topologia e os algoritmos de controlo propostos.

Por fim, no Capítulo 7 são apresentadas as conclusões obtidas ao longo da realização deste trabalho. Neste capítulo, são ainda descritas algumas sugestões para trabalhos futuros, com o intuito de aperfeiçoar e complementar o trabalho desenvolvido.

Capítulo 2

Configurações de MMCC e Sistemas de Controlo Descentralizados

2.1 Introdução

Tanto na indústria, como no meio académico, os conversores de eletrónica de potência são uma das escolhas favoritas para sistemas de conversão de energia elétrica de alta eficiência. Esses conversores sustentam uma ampla gama de aplicações industriais, como bombas, compressores, ventiladores, moinhos, transportes, transmissão de corrente contínua de alta tensão (HVDC – *High Voltage Direct Current*), compensação de energia reativa e sistemas de conversão de energia eólica [26, 27].

Este capítulo trata da análise do estado da arte e dos avanços recentes em topologias de conversores multinível. São abordadas as principais configurações dos conversores multinível e os requisitos necessários para o desenvolvimento de um sistema de controlo descentralizado, considerando: meios de transmissão, topologias e protocolos de comunicação.

Inicialmente, são discutidas as topologias dos conversores fonte de tensão (VSC – *Voltage Source Converter*), nomeadamente, conversores de dois níveis de tensão e conversores multinível, apresentando as respetivas vantagens e desvantagens quando relacionadas. De todos os conversores multinível apresentados, é atribuído um maior foco para os conversores modulares multinível em cascata.

Posteriormente, é realizado um paralelismo entre os sistemas centralizados e descentralizados, onde são expostas as razões que levam à utilização de estratégias descentralizadas para MMCC. Nas subsecções seguintes, são discutidos os requisitos necessários para o desenvolvimento de um sistema com uma estratégia de controlo descentralizada. Primeiramente, são analisados os meios de transmissão aplicados às diferentes ligações de comunicação, seguido da descrição das diversas topologias de

comunicação, com foco nas topologias em estrela, barramento e anel. Por último, foi feito um estudo dos protocolos de comunicação existentes para as topologias mencionadas, com uma explicação mais detalhada do envio e receção das tramas de comunicação para os protocolos mais relevantes.

2.2 Conversores Modulares Multinível

Os conversores multinível são um novo conceito de VSC desenvolvido para aplicações de média e alta tensão, como sistemas ferroviários, sistemas de distribuição de energia, parques eólicos e parques solares fotovoltaicos. Estes sistemas permitem superar a limitação das tensões de bloqueio dos semicondutores. Além disso, têm a capacidade de sintetizar a forma de onda da saída a partir de um conjunto de barramentos CC.

Estes conversores, quando comparados com conversores de dois níveis, apresentam vantagens significativas, tais como [28 - 31]:

- Facilmente extensíveis devido à sua construção modular;
- Capacidade de operar em casos de falha, através da criação de estados de funcionamentos redundantes, tornando o sistema mais robusto;
- Redução do dv/dt dos semicondutores de potência, permitindo diminuir problemas relacionados com interferências eletromagnéticas;
- Melhor conteúdo harmônico e diminuição do volume dos filtros passivos, minimizando as perdas do sistema e os custos de implementação. Quanto maior o número de níveis, melhor é a qualidade da onda de saída;
- Possibilidade de redução das frequências de comutação, minimizando as perdas de comutação;
- Integração fácil de fontes de energia renovável e sistemas de armazenamento de energia.

Nos últimos anos foram estabelecidas várias topologias de conversor multinível, conforme mostra Figura 2.1, contudo poucas tiveram sucesso na indústria. Conversores com ponto neutro fixo (NPC – *Neutral Point Clamped*), ponto neutro fixo do tipo T (TNPC – *T-type Neutral Point Clamped*) e condensadores flutuantes (FC – *Flying Capacitors*) podem suportar apenas uma tensão de 2,3 kV a 4,16 kV [32]. Esses conversores obrigam a modificações significativas

para aumentar a tensão operacional, não sendo uma solução económica. Além disso, devem ser desligados durante falhas, levando a uma perda significativa de produção nos processos industriais [33].

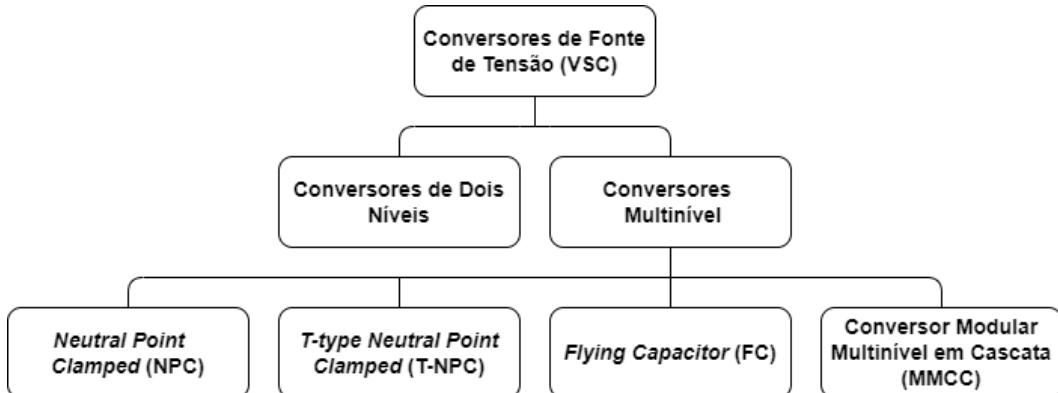


Figura 2.1 – Classificação de Conversores de Fonte de Tensão [34].

Com o intuito de resolver os problemas acima descritos, topologias de conversor modular multinível em cascata (MMCC – *Modular Multilevel Cascade Converter*) foram desenvolvidas. Essas topologias são construídas com uma conexão em cascata de submódulos de baixa potência com uma fonte CC isolada. Essas topologias são de construção modular, podendo atingir tensões de operação até 13,8 kV [32], e operar com capacidade reduzida em caso de falhas. Para atingir uma tensão operacional mais alta, o número de submódulos pode ser aumentado. No entanto, estas topologias requerem um transformador de mudança de fase com vários enrolamentos secundários para gerar fontes CC isoladas. O transformador aumenta o tamanho geral e o custo do conversor [32, 33].

Além destes, foram desenvolvidos conversores modulares multinível que não requerem fontes CC isoladas. O transformador de deslocamento de fase pode ser eliminado, minimizando o custo e o tamanho do conversor [34]. Esta característica única é explorada e usada com sucesso em várias aplicações industriais. Por outro lado, estes MMCC também requerem um controlo mais complexo [35].

2.3 Sistemas de Controlo Descentralizado

Os conversores modulares multinível discutidos na literatura existente são operados com estratégias de controlo centralizadas. Estas estratégias têm diversos objetivos de controlo, tais como: regulação de corrente ou tensão de saída, regulação do barramento CC do

submódulo e injeção de energia na rede elétrica [36, 37]. O principal problema deste controlo está relacionado com a limitação da modularidade e expansibilidade do sistema, devido às restrições relacionadas com o número de GPIO disponíveis nos microcontroladores. Além disso, a carga computacional é pesada e o tempo de execução pode não ser suficiente, especialmente para um MMCC com um grande número de submódulos, podendo levar a problemas de confiabilidade [38].

Assim sendo, é benéfico utilizar uma estratégia de controlo descentralizado para sistemas MMCC. Esta estratégia de controlo traz inúmeras vantagens para o sistema. A modularidade é melhorada em termos de desenvolvimento de software, a capacidade de processamento é distribuída entre os diversos controladores, a implementação torna-se mais fácil e a confiabilidade do sistema aumenta [39, 40]. Estas vantagens levaram ao aumento do interesse nas arquiteturas de controlo descentralizado para os sistemas MMCC, especialmente para os com grande número de submódulos [41]. Porém, a tarefa de controlar, montar e fazer a manutenção de um elevado número de submódulos não é algo trivial. Vários requisitos são levantados no desenvolvimento de um sistema de controlo descentralizado. Questões como o meio de transmissão utilizado, a topologia de comunicação escolhida ou os protocolos de comunicação usados são discutidas até hoje.

2.3.1 Meio de Transmissão

Na literatura existente são utilizados dois tipos de meio de transmissão de dados: o meio elétrico e o meio ótico. Contudo, a utilização de fibra ótica em conversores modulares multinível é recorrentemente considerada preferível, mas alguma forma de isolamento ótico é de facto obrigatória: nenhuma outra solução permite transferir dados com a necessária baixa latência, enquanto proporciona o isolamento galvânico necessário para uma operação segura [42]. Além do tipo de meio de transmissão, a transferência de dados pode estar localizada em dois sítios distintos. A primeira é a ligação de comunicação entre os submódulos, tanto do *master* para os *slaves* como entre os próprios *slaves*. A segunda é entre os microcontroladores e os gate-drivers dentro dos submódulos, cuja função é enviar os sinais de PWM.

A grande maioria dos artigos considera o meio ótico como uma melhor opção para a transmissão dos dados nos conversores modulares multinível. Em [43 - 45] mencionam o ambiente com ruido eletromagnético (EMI - *Electromagnetic Interference*) como uma razão para favorecer o meio ótico, uma vez que este apresenta maior imunidade a esse ruido. Já o artigo [46], invoca tanto o isolamento galvânico dos submódulos, como a tolerância ao ruido eletromagnético para preferir a comunicação ótica. Assim como em [47], onde os *drivers* de IGBT (*Insulated Gate Bipolar Transistor*) são conectados com cabos de fibra ótica à placa do microcontrolador para garantir o isolamento galvânico entre as partes de potência e controle. Em [48] é analisada a fiabilidade do sistema, sendo que cada submódulo tem acesso às tensões de cada barramento CC dos restantes submódulos. Deste estudo, foi concluído que, sem um excelente isolamento elétrico destes sistemas, não é possível garantir que não existe interferência do sistema de eletrónica de potência com o sistema de controlo. Todavia, outros autores discutem estes argumentos.

Em relação ao argumento das interferências eletromagnéticas é referido que na grande parte dos MMCC, os submódulos têm uma tensão de bloqueio de alguns quilovolts e produzem dv/dt e di/dt semelhantes aos de outros conversores de baixa/média tensão. Além disso, por motivos de operação e segurança, é necessário fornecer as distâncias mínimas entre os submódulos. Essas distâncias podem atingir a magnitude de metros, o que reduz o acoplamento parasita no circuito de potência e consequentemente a interferência eletromagnética. Em relação ao argumento da confiabilidade, se ocorrer uma falha e centenas de quilovolts forem aplicados a um submódulo projetado para suportar apenas alguns, o dano será catastrófico, independentemente de o *master* também sofrer um risco. Assim, isolar os submódulos é irrelevante neste contexto [42].

Exemplos da utilização de ambos meios de transmissão são apresentados em [42], [49] e [50]. Em [49] são utilizados os dois meios de transmissão para um sistema de controlo com protocolo em barramento para sistemas de conversores modulares. [42] menciona que entre os microcontroladores e os *gate-drivers*, o meio ótico é opcional e argumenta que utilização dos dois meios de transmissão é possível, desde que o hardware dos submódulos esteja ao mesmo potencial. Neste caso, a estratégia de controlo também é importante, pois o hardware de controlo está no nível de terra, portanto, se as tensões dos condensadores tiverem de ser medidas, não é viável colocar sensores isolados em todos os submódulos. Em [50] é utilizada

uma estratégia para estimar as tensões dos condensadores, eliminando os sensores nos submódulos e justificando o uso de acoplamento ótico para os sinais PWM e meios elétricos para a ligação de comunicação.

2.3.2 Topologias de Comunicação

As topologias de comunicação desempenham um papel muito importante no controlo descentralizado do MMCC, uma vez que, os comandos do controlo e algumas medições precisam ser trocadas entre as diferentes partes do sistema. Estas têm um grande impacto no número e tamanho das conexões, no *delay* das comunicações e na tolerância a falhas.

Na maior parte dos casos, o microcontrolador principal (*master*) determina a tensão que cada submódulo deve ter a cada instante, mas as ligações diretas entre o *master* e os microcontroladores em cada módulo (*slaves*), conhecidas como topologia em estrela, levam a uma quantidade enorme de conexões. Para reduzi-las, alguns autores propuseram e analisaram o uso de uma rede interna de comunicação digital para o controle de MMCC. Tal rede permite compartilhar ligações de comunicação entre submódulos, levando a topologias mais simples, por exemplo, barramento ou anel, simplificando a montagem e manutenção [51]. Como tal, foram estudadas três topologias para que o *master* comunique com os *slaves* de forma a garantir que a troca de dados entre eles esteja assegurada.

2.3.2.1 Estrela

A rede de comunicação em estrela é uma topologia em que os dados são trocados entre pares, neste caso os pares seriam formados pelo *master* e um *slave*, como representado na Figura 2.2. Este tipo de rede de comunicação confere algumas vantagens relativamente aos restantes, nomeadamente uma alta eficiência e confiabilidade, uma vez que o *master* comunica diretamente com o *slave*. Contudo, não apresenta grandes benefícios para o MMCC com controlo descentralizado. Com o aumento do número de *slaves*, o controlo e comunicação pode tornar-se algo complexo de se gerir [52]. Não obstante, o número de GPIO exigidos ao *master* torna esta solução inviável para sistemas com uma grande quantidade de submódulos.

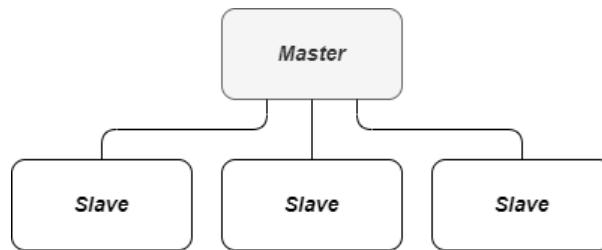


Figura 2.2 – Esquema de blocos de uma rede com comunicação ligada em estrela.

2.3.2.2 Barramento

A rede de comunicação em barramento é uma topologia em que todos os pontos de interceção estão ligados ao mesmo nó. Nesta topologia os dispositivos são chamados de estação. Cada estação recebe todo o tráfego da rede de comunicação. Contudo, apenas uma estação pode escrever no barramento num dado momento e, enquanto isso, todas as outras leem os dados. Para que todos compartilhem o barramento de forma eficaz existem técnicas de controlo de acesso ao barramento [53].

Para os sistemas com controlo descentralizado esta topologia pode ser uma grande aliada pois permite uma comunicação quase simultânea do *master* para todos os *slaves*. Possibilitando enviar sinais de *enable* ou atualizações do sistema de uma forma mais rápida e sem atrasos entre os submódulos. Além do mais, tem como vantagens a sua fácil implementação, o simples funcionamento em pequenas redes de comunicação e os reduzidos custos. Por outro lado, podem ocorrer colisões, o que resulta na perda de dados, e é difícil isolar falhas que ocorram na rede de comunicação.

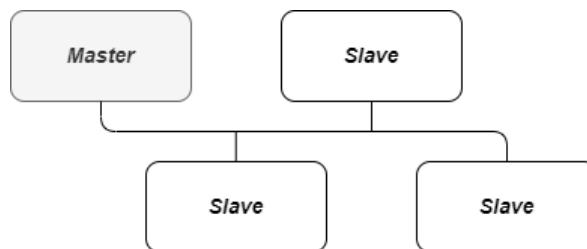


Figura 2.3 – Esquema de blocos de uma rede de comunicação ligada em barramento.

2.3.2.3 Anel

A rede de comunicação em anel é uma topologia em que cada dispositivo (*master* ou *slave*) se conecta apenas a outros dois, sendo que, os dados vão passando de dispositivo em

dispositivo até que todos os recebam, como podemos notar na Figura 2.4. Além disso, a comunicação pode ser feita de forma unidirecional ou bidirecional.

Este tipo de topologia não requer um nó central para gerir as conectividades e possui um desempenho melhor que a topologia em barramento. Se um dos *slaves* tiver algum problema a cadeia de transmissão de dados será quebrada [54]. Para este sistema com controlo descentralizado isto também pode ser interpretado como uma vantagem, uma vez que é possível detetar uma anomalia. Contudo, quanto maior for o número de *slaves* ligados à rede de comunicação, maior será o tempo de atraso da comunicação.

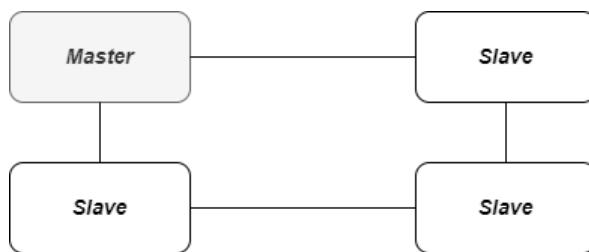


Figura 2.4 – Esquema de blocos de uma rede de comunicação ligada em anel.

2.3.3 Protocolos de Comunicação

Existem inúmeros protocolos de comunicação disponíveis, porém para o correto funcionamento do MMCC com controlo descentralizado, o tempo de resposta e a velocidade de comunicação são cruciais. Nesse sentido, as tecnologias de comunicação *non-real-time* não podem ser consideradas.

Como referido anteriormente, entre as topologias de comunicação consideradas, as que têm maior potencial para conversores modulares multinível são as topologias em barramento e anel. Enquanto a topologia em barramento é menos tolerante a falhas e com maior número de conexões, mas apresenta um atraso menor; a topologia em anel implica maiores atrasos, mas com as vantagens de reduzir o tamanho das conexões e possibilitar comunicar com um módulo por dois caminhos separados, o que simplifica a manutenção e aumenta a tolerância a falhas.

Dentro dos protocolos de comunicação em tempo real existentes para as topologias em barramento, uma das possíveis escolhas é o protocolo I2C (*Inter-Integrated Circuit*), sendo que em [55, 56] os autores apresentam esse protocolo para MMCC. Em [55] a comunicação serie

unidirecional assíncrona um para um é usada para transmissão de dados com uma taxa de 2 Mbit/s. Em [56] a arquitetura proposta consiste num controlador *master* e seis controladores do braço (*slaves*) a comunicar através do protocolo I2C. Além disso, são apresentadas as tramas de comunicação usadas para enviar e receber os dados dos *slaves*. O protocolo CAN (*Controller Area Network*) também pode ser usado para o controlo descentralizado de um MMCC [47, 57]. É referido em [47] que este protocolo carece de velocidade usando apenas, no máximo, 1 Mbit/s com tamanho máximo da ligação de comunicação de 40 m, porém é suficiente para sistemas com um baixo número de submódulos. De forma semelhante, em [57] este protocolo é usado para controlar um MMCC, com uma precisão de sincronização de $\pm 20 \mu\text{s}$. Também é feita uma descrição detalhada do protocolo bem como as suas vantagens e limitações.

Para a topologia em anel, protocolos como EtherCAT (*Ethernet for Control Automation Technology*), SPI (*Serial Peripheral Interface*) e RS232 (*Recommended Standard 232*) são apresentados em [51] como soluções para MMCC com controlo descentralizado. No artigo [58] é proposto um novo protocolo baseado na norma Ethernet IEEE802.3, o protocolo proposto com a nova estrutura de quadro estático mostrou funcionar de forma síncrona na faixa de $\pm 4 \text{ ns}$. Em [49] é desenvolvido um protocolo baseado em Ethernet, otimizado para transmissão de dados entre vários *slaves*, alcançando uma sincronização para $\pm 5\text{ns}$ com um *jitter* durante a operação abaixo de $\pm 600 \text{ ps}$.

Dentro destes protocolos, o EtherCAT é o de maior interesse para o controlo descentralizado de MMCC devido à sua alta velocidade de comunicação e à capacidade de conectar um grande número de *slaves*. O EtherCAT é amplamente utilizado na indústria, pois grandes empresas, como *ABB*, *OMRON* e *Bechhoff Automation*, usam ou fabricam produtos EtherCAT [59]. Cada *slave* EtherCAT lê e escreve dados na posição que lhe é atribuída na trama. Quando o *master* envia uma trama, ela vai para o primeiro *slave* que processa os dados e depois envia a trama para o próximo *slave*. Este processo continua até que o último *slave* seja alcançado. O último *slave* envia a mensagem de volta ao *master*. A trama será atrasada pelo atraso da propagação do meio de transmissão e pelo atraso do processamento introduzido pelos *slaves* [60].

Além das aplicações industriais, muitos autores adotaram este protocolo para sistemas de conversores modulares multinível. A técnica de sincronização de PWM para controladores

locais baseados no mecanismo de *clock* distribuído EtherCAT é abordada em [39]. Em [61] é realizada uma explicação detalhada sobre a configuração do sistema MMCC BTB (*Back-to-Back*) com 31 níveis e operação de todo o sistema sincronizando os *masters* e *slaves*, usando o protocolo de comunicação EtherCAT. O desenvolvimento de um demonstrador de média tensão com 350 kVA foi descrito em [62], onde a topologia MMCC meia ponte é combinada com o sistema de comunicação EtherCAT. No artigo [63] é descrito um algoritmo de controlo tolerante a falhas, baseado no protocolo de comunicação em tempo real EtherCAT, para MMCC com estrutura de submódulos em ponte completa.

2.4 Conclusões

Neste capítulo foi realizada uma revisão dos avanços recentes em topologias de conversores multinível e dos requisitos necessários para o desenvolvimento de um sistema de controlo descentralizado.

Inicialmente, foram apresentados os desenvolvimentos mais recentes dos conversores fonte de tensão, os conversores multinível. Estes conversores possuem inúmeras vantagens que são aplicadas com sucesso em diversas aplicações industriais. Concluiu-se que, dentro dos conversores multinível as topologias de conversores modulares multinível são as que apresentam maior interesse para aplicações de média e alta tensão, uma vez que estas topologias permitem aumentar a tensão e a potência nominal do conversor adicionando módulos em série e operar em caso de falhas.

Posteriormente, abordaram-se os benefícios da utilização da estratégia de controlo descentralizado para sistemas MMCC, tais como, modularidade melhorada, carga de computação distribuída, implementação simplificada e confiabilidade do sistema aumentada. Além disso, foram apresentadas questões como o meio de transmissão utilizado, a topologia de comunicação escolhida e os protocolos de comunicação usados.

Posto isto, foram discutidos dois tipos de meio de transmissão de dados: o meio elétrico e o meio ótico. Analisando a literatura, conclui-se que a utilização de fibra ótica em conversores modulares multinível é a mais utilizada. Contrariamente às outras soluções, a fibra ótica permite transferir dados com uma baixa latência, proporcionando igualmente o isolamento galvânico necessário para uma operação segura.

Após uma breve análise das principais topologias de comunicação existentes e fazendo uma fusão com as necessidades do sistema estudado, concluiu-se que será necessário o uso de duas topologias: anel e barramento. Relativamente à topologia em anel, esta topologia permite ao *master* detetar o número de submódulos que estão ligados à rede de comunicação e encontrar anomalias. Contudo, devido ao tempo de atraso da comunicação na topologia em anel, uma segunda é introduzida: topologia em barramento. Esta topologia garante a comunicação “simultânea” do *master* a todos os submódulos, sendo também responsável por transmitir o sinal de sincronismo, o valor de amplitude da corrente que é pretendido à saída do inversor e o número total de submódulos que estão conectados.

Por fim, foram apresentados diferentes protocolos de comunicação para as topologias mencionadas, fazendo uma descrição mais detalhada do princípio de funcionamento do protocolo EtherCAT. Este é de maior interesse para o controlo descentralizado de conversores modulares multinível devido à sua alta velocidade de comunicação e à capacidade de conectar um grande número de *slaves*.

Capítulo 3

Conversores de Eletrónica de Potência e Algoritmos de Controlo

3.1 Introdução

A topologia do conversor modular multinível é uma solução promissora para aplicações industriais de alta tensão e potência. Este capítulo trata dos seus aspectos gerais, incluindo configuração dos submódulos e algoritmos de controlo.

Inicialmente, são discutidas diferentes configurações de submódulo do MMCC, particularmente, as configurações de meia ponte, ponte completa, NPC e TNPC. No final desta secção é realizada uma comparação entre elas, analisando diferentes características, como o número de semicondutores usados, o número de condensadores no barramento CC, os níveis de tensão que cada uma permite obter à saída e os graus de complexidade no que diz respeito ao design e algoritmos de controlo dos submódulos.

Posteriormente, são apresentadas técnicas de PWM utilizadas de forma a controlar o valor da tensão a ser gerado pelo conversor, para que se obtenha a corrente de saída desejada. Em seguida, é descrito o princípio de funcionamento das técnicas de PWM com deslocamento de nível e de fase e são apresentados exemplos da utilização das mesmas em conversores modulares multinível.

De seguida, são estudadas as técnicas de controlo de corrente complementares às técnicas PWM, expondo os comportamentos de cada técnica e as respetivas vantagens e desvantagens.

Uma vez concluída a apresentação das técnicas de controlo, são analisados algoritmos de regulação do barramento CC. Essa análise consiste não só na explicação e descrição das vantagens e desvantagens do algoritmo de classificação, mas também na menção de novas técnicas capazes de superar as suas limitações.

Por último, são expostos algoritmos de sincronização com a rede elétrica, dando um maior destaque para o algoritmo de sincronização *phase locked loop* (PLL). Sendo este o algoritmo mais utilizado, é mostrado um diagrama de blocos tanto da PLL básica como o de uma PLL melhorada.

3.2 Topologias de Submódulos para MMCC

Um submódulo é um circuito simples de conversão de energia DC-AC. Normalmente, dispositivos semicondutores de baixa tensão e condensadores CC são usados para configurar um submódulo. As topologias de submódulo habitualmente usadas num conversor modular multinível são [64, 65]:

- Submódulo de meia ponte (HB – *Half Bridge*);
- Submódulo de ponte completa (FB – *Full Bridge*);
- Submódulo de ponto neutro fixo (NPC);
- Submódulo de ponto neutro fixo do tipo T (TNPC).

3.2.1 Meia Ponte

Existem duas configurações diferentes para a topologia em meia ponte. A primeira é composta por um barramento CC dividido, para que seja fornecido um ponto neutro; e por um braço constituído por dois semicondutores de potência totalmente controlados, como pode ser observado na Figura 3.1(a). Na segunda o barramento CC é composto por uma única fonte de tensão, Figura 3.1(b) [66].

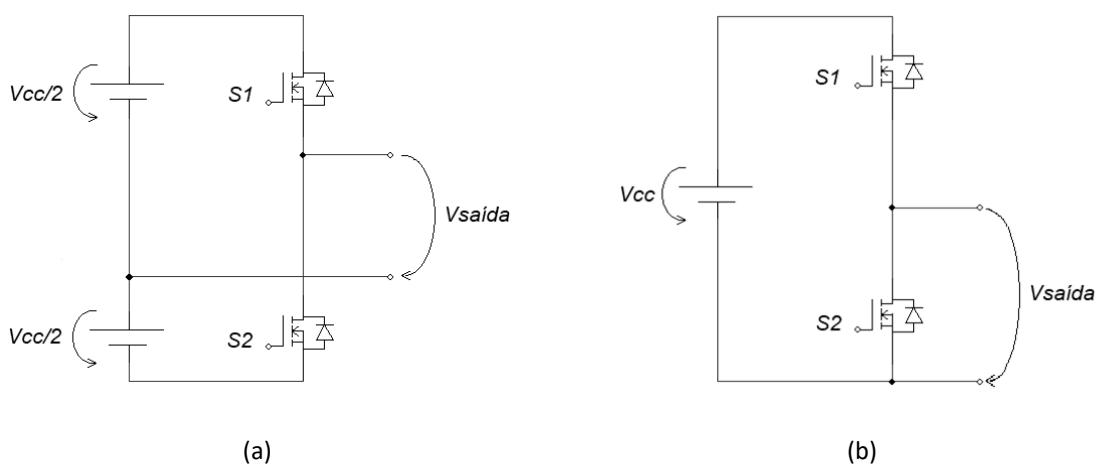


Figura 3.1 – Esquema de um conversor CC-CA em meia ponte: (a) barramento dividido; (b) barramento único.

No primeiro caso é possível obter à saída dois níveis de tensão. Sendo o barramento dividido, os níveis têm apenas metade da tensão V_{cc} . Na Tabela 3.1 são apresentados os diversos estados de operação dos semicondutores [66, 67]. Apesar da sua simplicidade em termos de hardware, esta não é uma configuração de submódulo habitualmente usada para aplicações de MMCC [68]. No entanto, é possível encontrar a integração desta configuração em soluções multinível híbridas [69].

Tabela 3.1 – Estados de operação num conversor CC-CA em meia ponte com ponto médio.

Estado	$S1$	$S2$	$V_{saída}$
1	ON	OFF	$+V_{cc}/2$
2	OFF	ON	$-V_{cc}/2$

No segundo caso, a tensão de saída é igual à tensão da fonte ou zero, dependendo dos estados de operação dos semicondutores. Deste modo, esta topologia necessita de um submódulo auxiliar para fazer interface com aplicações CA. Um submódulo é responsável por sintetizar o semiciclo positivo e outro o semiciclo negativo [67, 70]. Os estados de operação são apresentados na Tabela 3.2. Em [71, 72] são apresentadas aplicações desta topologia em MMCC.

Tabela 3.2 – Estados de operação num conversor CC-CA em meia ponte sem ponto médio.

Estado	$S1$	$S2$	$V_{saída}$
1	ON	OFF	$+V_{cc}$
2	OFF	ON	0

3.2.2 Ponte Completa

Um conversor de ponte completa é constituído por uma fonte de tensão no barramento CC e por dois braços formados por dois semicondutores totalmente controlados. A carga é conectada entre os pontos médios de cada braço dos semicondutores, como representado na Figura 3.2 [73].

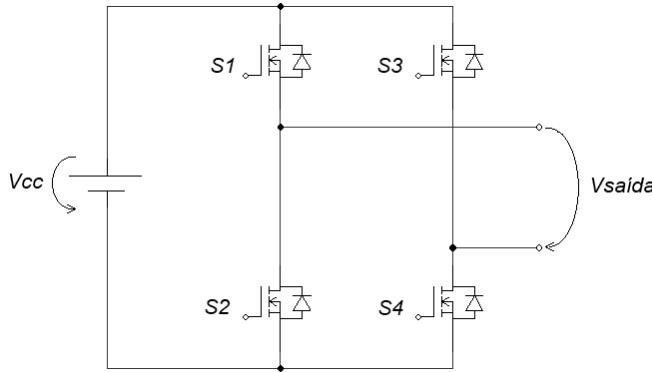


Figura 3.2 – Esquema de um conversor CC-CA em ponte completa.

Esta topologia permite que sejam obtidos três níveis de tensão na saída do conversor, isto tendo em conta os vários estados dos semicondutores, como mostra a Tabela 3.3. Uma vez que o número de semicondutores na ponte completa é o dobro do número de semicondutores na meia ponte, as perdas e custo do MMCC com submódulos em ponte completa são significantemente maiores [64]. Exemplos da topologia em ponte completa para MMCC são expostos em [74 - 76].

Tabela 3.3 – Estados de operação num conversor CC-CA em ponte completa.

Estado	S1	S2	S3	S4	Vsaída
1	ON	OFF	OFF	ON	+Vcc
2	ON	OFF	ON	OFF	0
3	OFF	ON	ON	OFF	- Vcc
4	OFF	ON	OFF	ON	0

3.2.3 Neutral Point Clamped

Em 1981, foi apresentada por Nabae, Takakashi e Akagi uma nova topologia de inversor com um ponto neutro [66]. A nova topologia tem por base o uso de mais dois semicondutores de potência comparando com a topologia de ponte completa. Dessa forma, é constituída por um braço com quatro semicondutores totalmente controlados e dois díodos de fixação de acordo com a montagem da Figura 3.3.

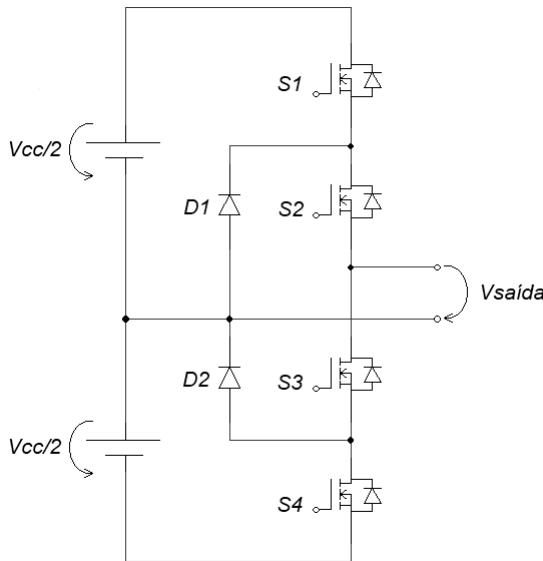


Figura 3.3 – Esquema de um conversor CC-CA em topologia NPC.

Esta topologia permite que se obtenha à saída do conversor três níveis de tensão distintos, como mostra a Tabela 3.4. Um dos principais benefícios de utilizar o NPC é a baixa taxa de distorção harmónica (THD – *Total Harmonic Distortion*) da corrente, fazendo com que o esforço de filtragem seja menor [77]. O MMCC com submódulos NPC tem perdas nos semicondutores maiores que o MMCC com submódulos meia ponte e menores que o MMCC com submódulos ponte completa [64]. Em [78, 79] são apresentados exemplos da topologia para MMCC.

Tabela 3.4 – Estados de operação num conversor CC-CA na topologia NPC.

Estado	S1	S2	S3	S4	Vsaída
1	OFF	OFF	ON	ON	-Vcc/2
2	OFF	ON	ON	OFF	0
3	ON	ON	OFF	OFF	+Vcc/2

3.2.4 T-type Neutral Point Clamped

Esta topologia apresenta como principal benefício relativamente ao NPC a forma de onda da tensão de saída, sobretudo se não existirem restrições no esquema de comutações. Além disso, tal como a topologia anterior, produz formas de onda de corrente com baixo THD [77].

Este tipo de topologia consiste apenas em quatro semicondutores organizados de acordo com a configuração representada na Figura 3.4.

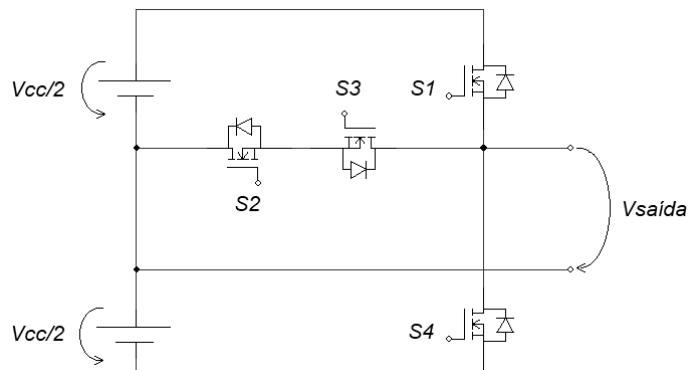


Figura 3.4 – Esquema de um conversor CC-CA em topologia T-NPC.

Esta topologia permite, tal como as anteriores, que nos terminais de saída se obtenham três níveis de tensão diferentes, como demonstrado na Tabela 3.5. O modelo desta topologia para MMCC pode ser observado em [80].

Tabela 3.5 – Estados de operação num conversor CC-CA na topologia TNPC.

Estado	S1	S2	S3	S4	$V_{saída}$
1	ON	OFF	OFF	OFF	$+V_{cc}/2$
3	OFF	ON	ON	OFF	0
5	OFF	OFF	OFF	ON	$-V_{cc}/2$

3.2.5 Comparação das várias topologias

Uma vez apresentadas algumas das topologias existentes para os submódulos dos MMCC, foi feita uma comparação entre elas. As características analisadas foram: o número de semicondutores utilizados (díodos e MOSFET – *Metal-Oxide-Semiconductor Field-Effect Transistor*), o número de condensadores necessários no barramento CC, os níveis de tensão que cada uma permite obter à sua saída e os graus de complexidade no que diz respeito ao design e algoritmos de controlo dos submódulos.

Tabela 3.6 – Comparação das diferentes topologias [67].

Topologia	Número de MOSFET	Número de diódos	Número de condensadores	Níveis de tensão à saída	Grau de complexidade no design	Grau de complexidade no controlo
Meia-ponte (a)	2	0	2	2	Baixo	Médio
Meia-ponte (b)	2	0	1	2	Baixo	Médio
Ponte completa	4	0	1	3	Médio	Baixo
NPC	4	2	2	3	Alto	Alto
TNPC	4	0	2	3	Médio	Alto

Após analisar a Tabela 3.6, pode concluir-se que quanto maior o número de semicondutores totalmente controlados, maior é o número de níveis de tensão de saída que podem ser sintetizados. A comparação entre a topologia em meia ponte e as restantes é o exemplo disso. Com dois dispositivos semicondutores totalmente controlados, a topologia em meia ponte é capaz de gerar dois níveis de tensão, enquanto as restantes topologias são capazes de gerar três níveis de tensão usando quatro. Todavia, o aumento dos semicondutores pode aumentar o preço e as perdas de potência no MMCC.

Verifica-se ainda que para o mesmo nível de tensão no barramento CC, apenas a topologia de ponte completa consegue sintetizar $+V_{cc}$ e $-V_{cc}$. Nas restantes, o nível de tensão sintetizado é apenas metade, ou seja, o nível de tensão das topologias com barramento dividido terá de ser o dobro para criar o mesmo valor de tensão de saída. Além disso, na topologia de meia ponte com um único barramento é necessário adicionar um submódulo auxiliar para aplicações AC. Além de encarecer o sistema, a divisão do barramento e a duplicação dos componentes acrescentam complexidade aos algoritmos de controlo e dificultam a implementação dos conversores modulares multinível.

3.3 Técnicas de SPWM

A modulação por largura de pulso (PWM – *Pulse-Width Modulation*) é recorrentemente utilizada para controlar a tensão de saída AC de um conversor. A tensão de saída é obtida através da comutação dos dispositivos semicondutores. As técnicas de modulação são

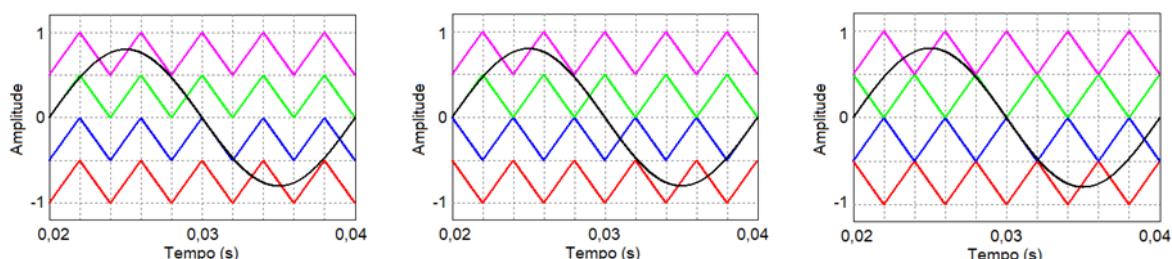
projetadas para reduzir a distorção harmônica e aumentar a magnitude da tensão de saída para uma determinada frequência de comutação. Além disso, as técnicas de modulação possuem outros objetivos de controlo, como a minimização das frequências de comutação, redução das perdas de potência e melhoria da qualidade da tensão/corrente de saída [81]. Várias técnicas de PWM, baseadas no uso de uma única onda de referência, foram desenvolvidas para MMCC. Uma das técnicas mais utilizadas é a modulação sinusoidal por largura de pulso (SPWM – *Sinusoidal Pulse Width Modulation*), onde as ondas portadoras são comparadas com uma onda moduladora sinusoidal.

3.3.1 Modulação SPWM com Deslocamento de Nível

As técnicas de SPWM com deslocamento de nível (LS-PWM – *Level-Shift Pulse-Width Modulation*) foram propostas principalmente para serem aplicadas nos conversores multinível NPC e FC. Estas estratégias de modulação não são adequadas para conversores em cascata, pois produzem uma distribuição desigual de potência entre os submódulos e injetam harmónicos na rede elétrica [67, 82]. No entanto, outras estratégias foram propostas para mitigar este problema.

Com base na relação de fase entre as portadoras, o LS-PWM é ainda categorizado em Disposição de Fase (PD – *Phase Disposition*), Disposição de Oposição de Fase (POD – *Phase Opposition Disposition*) e Disposição de Oposição de Fase Alternada (APOD – *Alternate Phase Opposition Disposition*).[83] As portadoras têm a mesma amplitude e frequência, mas são colocadas em diferentes níveis (*offsets*) [84].

Conforme apresentado na Figura 3.5(a), as portadoras no método PD possuem o mesmo ângulo de fase. No método POD, as duas primeiras portadoras estão 180° desfasadas das próximas duas, conforme mostrado na Figura 3.5(b). Todas as portadoras no método APOD estão alternadamente em oposição de fase, tal como apresentado na Figura 3.5(c).



(a)

(b)

(c)

Figura 3.5 – Técnicas de SPWM com deslocamento de nível: (a) PD, (b) POD, (c) APOD.

A técnica POD para um MMCC de cinco níveis com a configuração do submódulo em ponte completa é apresentada em [85], com o objetivo de operar um motor de indução trifásico. Em [86] a técnica de modulação PD é usada num MMCC, cujo intuito se concentrou na investigação da influência de diferentes técnicas de modulação na eficiência, harmónicos e temperatura do MMCC. No artigo [87] adotam um MMCC de cinco níveis para comparar diferentes técnicas de PWM com deslocamento de nível. Esta comparação mostrou que o perfil de tensão e o conteúdo harmónico das técnicas PD, POD, APOD têm quase a mesma qualidade de forma de onda, a qual pode ser melhorada usando filtros passivos na saída.

3.3.2 Modulação SPWM com Deslocamento de Fase

A técnica de SPWM com deslocamento de fase (PSC – *Phase-Shift Carrier*) é das mais utilizadas. Alguns microcontroladores possuem registos próprios para o controlo do ângulo de fase e sincronização das portadoras, o que facilita a implementação desta técnica [67]. Todas as portadoras têm a mesma amplitude, frequência e deslocamento, mas defasadas em $2\pi/n$, conforme mostrado na Figura 3.6, onde n representa o número de braços do MMCC e, consequentemente, o número de portadoras. Esta técnica é atrativa para MMCC, pois fornece um equilíbrio na tensão dos condensadores dos submódulos em altas frequências de comutação, distribui uniformemente a potência em cada submódulo e minimiza o *ripple* da corrente do barramento CC, diminuindo o conteúdo harmónico injetado na rede elétrica [88, 89].

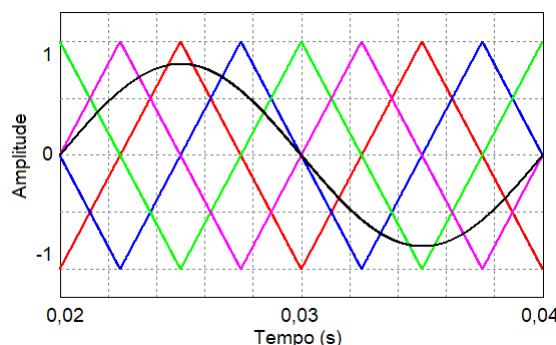


Figura 3.6 – Técnica de SPWM com deslocamento de fase (PSC).

Em [86], a técnica de modulação PSC é usada num MMCC. O trabalho concentrou-se na investigação da influência de diferentes técnicas de modulação e concluiu que o MMCC com modulação PSC tem melhor desempenho harmónico e maior eficiência que as restantes, a PD e a modulação de nível mais próximo (NLM – *Nearest Level Modulation*). No artigo [90], a modulação PSC foi combinada com um algoritmo de baixa frequência de comutação, que foi capaz de resolver o problema do equilíbrio nas tensões dos condensadores dos submódulos e reduzir as perdas nos semicondutores. Em [91], as modulações PSC e PD são aplicadas a um MMCC para acionamento ferroviários, unindo vantagens de ambas as técnicas.

3.4 Técnicas de Controlo de Corrente

Apesar das técnicas de SPWM apresentadas influenciarem o valor da tensão de saída do conversor CC-CA, atuando os semicondutores, não são suficientes para controlar a corrente de saída do inversor. Dessa forma, é necessário utilizar técnicas de controlo de corrente que complementam as anteriores.

As técnicas de controlo de corrente permitem determinar o valor da tensão a ser gerada pelo conversor para que se obtenha a corrente de saída desejada. Sendo que, quanto mais próximo do valor de referência for o valor da corrente sintetizada, melhor qualidade terá a corrente de saída do conversor. Estas são características necessárias para que se injete energia na rede elétrica tal como pretendido, resultando na necessidade do estudo das técnicas de controlo abordadas a seguir.

3.4.1 Controlo de Corrente por Histerese

A técnica de controlo de corrente por histerese tem por base a comparação da corrente que sai do conversor, $i_{saída}$, com a corrente de referência, i_{ref} , atribuindo uma margem de histerese, $\pm H$, ao resultado da comparação [92].

Esta técnica é implementada através da seguinte lógica: se a corrente não ultrapassar o valor de referência mais a histerese, os semicondutores S_1 e S_4 mantêm-se fechados e S_2 e S_3 abertos. Por outro lado, se o valor da corrente não ultrapassar o valor de referência menos a histerese, S_1 e S_4 mantêm-se abertos e S_2 e S_3 devem estar fechados. No caso intermédio os semicondutores devem permanecer no estado de condução em que se encontram [93]. A

lógica é baseada na análise da Figura 3.2, correspondente à topologia da ponte completa. Na Figura 3.7 é apresentado o esquema de blocos da técnica de controlo de corrente por histerese.

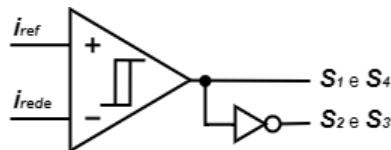
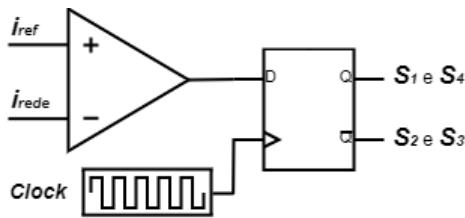


Figura 3.7 – Esquema de blocos da técnica de controlo de corrente por histerese [92].

Uma das desvantagens desta técnica deve-se à elevada gama de variação de frequências de comutação dos semicondutores, podendo causar ressonâncias indesejadas com a rede elétrica. Este problema pode danificar os próprios semicondutores de potência, devido a elevadas frequências de comutação. Além disso, faz com que seja difícil dimensionar filtros passivos necessários para filtrar os harmónicos de corrente causados pelas comutações [94]. Por outro lado, este método é caracterizado por ter um algoritmo relativamente simples, pequena ondulação na corrente de saída e uma boa robustez, sendo apresentadas estas e mais características em [95].

3.4.2 Controlo de Corrente por *Periodic Sampling*

A técnica de controlo de corrente por *periodic sampling* (amostragem periódica) é implementada através da comparação da corrente de referência com a corrente de saída do conversor. O valor que resulta dessa comparação entra num flip-flop do tipo D que irá funcionar como um circuito de *sampling and hold* (amostragem e retenção) com frequência fixa [92]. Este circuito permite controlar a frequência máxima a que os semicondutores podem comutar, evitando que seja ultrapassada a frequência que estes suportam [93]. Em [96] é apresentada uma proposta de solução para MMCC usando esta técnica de controlo de corrente. Na Figura 3.8 é apresentado o esquema de blocos da técnica de controlo de corrente por *periodic sampling*.

Figura 3.8 – Esquema de blocos da técnica de controlo de corrente por *periodic sampling* [92].

3.4.3 Controlo de Corrente Proporcional Integral com SPWM

A técnica de controlo de corrente proporcional integral (PI) com SPWM usa o erro entre a corrente de saída do conversor, $i_{saída}$, e a corrente de referência, i_{ref} , como variável de entrada do controlador PI. O controlador gera um sinal, $v_{controlo}$, que é usado para sintetizar os sinais de comando para o controlo dos semicondutores através da técnica de SPWM, como demonstrado na Figura 3.9 [92]

Esta técnica apresenta um grau de complexidade de implementação superior às técnicas apresentadas anteriormente e permite que se obtenha uma frequência de comutação fixa. Contudo existe um atraso entre a corrente de saída e a corrente de referência devido aos valores das constantes k_p e k_i [93]. Um exemplo da utilização desta técnica para MMCC é apresentado em [97].

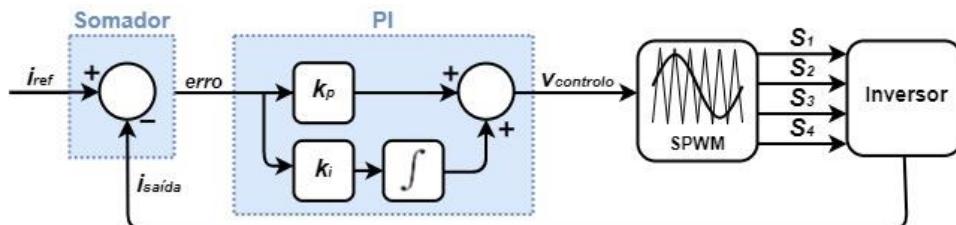


Figura 3.9 – Esquema de blocos de um controlador de corrente por PI com SPWM [92].

3.4.4 Controlo de Corrente Preditivo com SPWM

O funcionamento desta técnica de controlo é similar à anterior. Como pode ser observado na Figura 3.10, o controlador preditivo gera um sinal, $v_{controlo}$, que é usado para sintetizar os sinais de comando através da técnica de SPWM. Contudo, a técnica de controlo de corrente preditivo apoia-se no modelo elétrico do sistema para calcular $v_{controlo}$, antecipando o comportamento das variáveis a controlar. Dessa forma, a eficiência do controlo é diretamente

influenciada pelo modelo utilizado, melhorando com a aproximação do modelo elétrico à realidade [92].

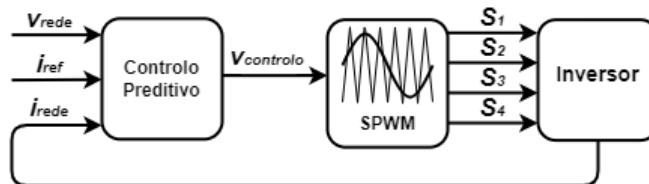


Figura 3.10 – Esquema de blocos de um controlador de corrente preditivo com SPWM [92].

Na Figura 3.11 é apresentado o modelo elétrico do sistema, a partir do qual as equações para o algoritmo do controlo preditivo são encontradas, através da aplicação da lei das malhas.

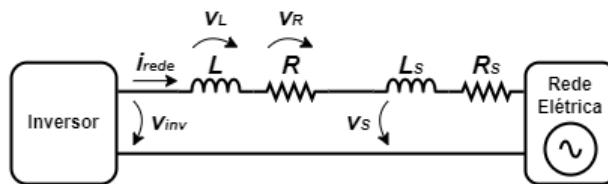


Figura 3.11 – Modelo elétrico do sistema [92].

Em [93] é demonstrada a dedução das equações, obtendo a equação (3.1), onde a corrente de erro corresponde à diferença entre a corrente de referência e a corrente de saída.

	$v_{inv} = v_s + L \frac{di_{ref}}{dt} + L \frac{di_{erro}}{dt}$	(3.1)
--	--	---------

Além disso, é necessário converter a equação anterior para o domínio discreto, permitindo que esta seja utilizada no microcontrolador. O resultado da conversão é a equação (3.2), onde T_a é o período de amostragem e considerando que a variação da corrente de erro (Δi_{erro}) é constante para um intervalo de amostragem curto (d_t).

	$v_{inv}[k] = v_s[k] + \frac{L}{T_a} (2i_{ref}[k] - i_{ref}[k - 1] - i_{saída}[k])$	(3.2)
--	---	---------

Apesar de implicar cálculos mais complexos, aumentando o tempo e atrasos de processamento, esta técnica dispensa o ajuste de ganhos, apresentando uma enorme vantagem quando comparada com a técnica de controlo PI. Dessa forma, o desempenho do sistema pode ser melhorado [98, 99]. Em [100] é proposto um modelo de controlo preditivo melhorado para MMCC, que permite reduzir a carga computacional. No artigo [101] é apresentado um método de controlo tolerante a falhas baseado no controlo preditivo para MMCC sem submódulos redundantes, melhorando a confiabilidade de operação do sistema.

3.5 Algoritmo de Regulação do Barramento CC

Um dos principais problemas dos MMCC é o equilíbrio das tensões dos condensadores dos submódulos. Este problema torna-se mais complexo com o aumento do número de submódulos do sistema. O controlo da tensão do condensador do submódulo é um requisito fundamental para conseguir um bom desempenho no controlo da corrente, melhorar a qualidade da corrente de saída e fornecer uma operação estável e controlável [102].

A técnica mais usada para realizar a tarefa de regular a tensão dos condensadores é baseada num algoritmo de classificação. Este algoritmo de classificação da tensão do condensador do módulo é apresentado para MMCC em [103 - 105]. Nesta técnica as tensões dos condensadores são medidas, usando sensores de tensão, e classificadas. Se a corrente do braço for positiva, os submódulos com tensão mais baixa são ligados e vice-versa. Apesar do método de classificação garantir o equilíbrio das tensões dos condensadores, ele produz transições de comutação desnecessárias, resultando no aumento da frequência de comutação e, posteriormente, perdas de energia, que são indesejáveis, especificamente para sistemas de alta potência [64].

Com o intuito de resolver este problema, vários autores propuseram e avaliaram novas técnicas. Em [106] é proposta uma estratégia de regulação de tensão usando SPWM com deslocamento de fase. A regulação da tensão do condensador é obtida enviando pulsos PWM apropriados aos submódulos. Esta estratégia não requer a medição das correntes no condensador, o que aumenta a simplicidade do controlo e reduz o número de sensores. É desenvolvida uma estratégia preditiva para o controlo de um MMCC em [107], na qual as tensões dos condensadores são equilibradas com base numa função predefinida. Em [108] é introduzido o método de regulação da tensão do condensador com a modulação PSC. A regulação dinâmica da tensão é obtida em [109] pela SPWM com deslocamento de fase para o conversor modular multinível.

3.6 Algoritmos de Sincronização com a Rede Elétrica

A sincronização dos conversores com a rede elétrica é um requisito muito importante nos sistemas de eletrónica de potência para garantir uma operação estável e continua. A situação torna-se mais crítica quando a rede está sujeita a problemas de qualidade de energia elétrica.

Esses problemas podem provocar desequilíbrios nas tensões de alimentação e harmónicos de tensão afetando os sinais de referência gerados pelos sistemas de controlo. Deste modo, são necessários algoritmos de sincronização para obter sinais de referência sinusoidais e em fase com a componente fundamental da tensão da rede elétrica [110, 111].

A rápida proliferação de sistemas de eletrónica de potência conectados à rede elétrica deu origem a uma série de métodos de sincronização. Um dos primeiros métodos propostos foi a deteção da passagem por zero (ZCD – *Zero Crossing Detection*) [112]. Este método é um dos mais simples para obter informações da rede. No entanto, o ponto de passagem por zero só é detetável a cada meio período do sinal, tornando esta deteção insuficiente, pois o desempenho dinâmico não será considerado [113]. O filtro de Kalman é uma das técnicas de processamento de sinal aplicadas à medição da frequência de sinais dos sistemas de potência. Contudo, é computacionalmente exigente [114]. Outro método é a transformada discreta de Fourier (DFT – *Discrete Fourier Transform*), que na verdade é uma das primeiras abordagens para detetar harmónicos e frequência [115]. Uma técnica alternativa para estimar a frequência é o método do erro mínimo quadrado (NLS – *Nonlinear Least-Square*), onde o objetivo é minimizar o erro quadrado entre o sinal modelado e o sinal medido [116]. Algumas outras técnicas como o método de sincronização de algoritmo baseado em *adaptive notch filtering* (ANF) e o *frequency-locked loop* (FLL) também foram apresentados na literatura para analisar o ângulo de fase da tensão da rede [117, 118].

A *Phase-Locked Loop* (PLL) é a técnica de sincronização mais utilizada, devido à sua simplicidade, eficácia e robustez em diversas condições da rede. É um sistema de controlo não linear de malha fechada que sincroniza o seu sinal de saída com o sinal de entrada em frequência e fase [119]. Conforme mostrado na Figura 3.12, a estrutura básica da PLL é composta por três blocos principais: o detetor de fase, o filtro (LF – *Loop Filter*) e o oscilador controlado por tensão (VCO – *Voltage-Controlled Oscillator*). Primeiro, o detetor de fase compara o sinal de entrada e de saída. O sinal de erro, resultante da comparação, é filtrado e passa pelo oscilador, gerando o sinal de saída. Este processo continua até que o erro de fase entre os sinais (sinal de referência e saída) atinja o valor mínimo [120].



Figura 3.12 – Diagrama de blocos da PLL básica.

Embora a PLL básica referida tenha sido muito bem-sucedida, ela apresenta dois problemas que dificultam o seu uso em sistemas de potência e controlo. O primeiro é que o *loop* tem erros de frequência dupla, mesmo no caso mais ideal onde o sinal de entrada é uma sinusóide. O segundo é que a magnitude do sinal de saída não tem relação com a do sinal de entrada. Em [120, 121] uma PLL aprimorada (EPLL – *Enhanced Phase-Locked Loop*) é apresentada. O diagrama de blocos da EPLL é mostrado na Figura 3.13.

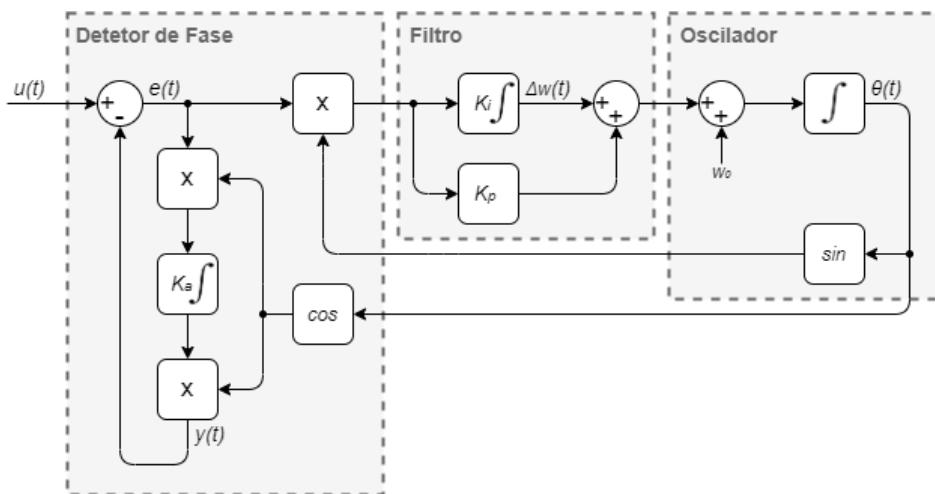


Figura 3.13 – Diagrama de blocos da PLL aprimorada.

3.7 Conclusões

Neste capítulo foi realizada uma revisão dos aspetos gerais dos conversores modulares multinível, abrangendo aspetos como: configurações dos submódulos, técnicas de SPWM, algoritmos de regulação do barramento CC e mecanismos de sincronização com a rede elétrica.

Inicialmente, foram apresentadas e comparadas diferentes topologias de submódulos usados em MMCC, tais como, topologias de conversores em meia ponte, ponte completa, NPC e T-NPC. Concluiu-se que a topologia mais adequada para a implementação da solução pretendida é o conversor em ponte completa. Isto deve-se ao facto de permitir sintetizar três

níveis de tensão utilizando poucos semicondutores, ser a única topologia capaz de gerar $\pm V_{CC}$ a partir de um único barramento CC e possuir uma complexidade de controlo reduzida. De salientar que a estrutura com submódulos de ponte completa ligados em cascata é uma estrutura modular que permite aumentar a tensão total com alta confiabilidade, apenas aumentando o número de submódulos usados. Por essa razão, é frequentemente considerada como a solução mais adequada para este tipo de problemas.

Posteriormente, abordaram-se técnicas de SPWM com desfasamento de nível e fase, aplicadas no controlo da tensão de saída do MMCC, e as técnicas de controlo complementares para o controlo da corrente. Após essa análise, concluiu-se que ambas as técnicas de controlo de corrente, PI e preditivo, aliadas à SPWM com desfasamento de fase, apresentam as características mais adequadas para a aplicação em causa. Mais concretamente, na SPWM com desfasamento de fase, é possível observar que, a onda moduladora interceta com mais frequência as ondas portadoras. Dessa forma, cada submódulo contribui de forma semelhante na produção da forma de onda de saída, dividindo, igualmente, a potência de operação entre os submódulos. Além disso, a ondulação de tensão dos barramentos CC é idêntica em aplicações em cascata [67]. Por consequência, as características desta técnica reforçam o conceito de modularidade e permitem alcançar os objetivos pretendidos de uma forma simples e robusta. De salientar que, em relação aos controlos de corrente referidos, teoricamente ambos apresentam ótimas respostas estáticas e dinâmicas, porém, no controlo preditivo não existe a necessidade de ajustar ganhos, podendo ser uma grande vantagem, principalmente em aplicações multinível.

De seguida, analisaram-se algoritmos de regulação do barramento CC. Foi possível constatar que o método de classificação, apesar ser um dos mais utilizados, resulta no aumento da frequência de comutação e, posteriormente, em perdas de energia. Por essa razão, vários algoritmos propostos com o intuito de superar essas limitações foram referenciados.

Por fim, foram expostos algoritmos de sincronização com a rede elétrica. De todos os apresentados, o mais utilizado é o algoritmo de sincronização PLL, apesar dos seus erros de frequência dupla e da falta de relação entre as magnitudes dos sinais de saída e entrada. Esses erros podem ser superados com a melhoria do algoritmo, sendo, consequentemente, apresentado o diagrama de blocos da EPLL.

Capítulo 4

Simulações Computacionais da Topologia Utilizada

4.1 Introdução

No estudo e desenvolvimento de sistemas de eletrónica de potência é fundamental a utilização de ferramentas que possibilitem estudar o comportamento e desempenho dos mesmos. Para tal, softwares como o PSIM, que permitem simular computacionalmente esses sistemas, são um recurso indispensável.

Esta ferramenta de simulação computacional tem como objetivo orientar corretamente o desenvolvimento dos sistemas. Além disso, permite a implementação de algoritmos de controlo codificados em linguagem de programação C, através de blocos de controlo e processamento de sinais digitais, facilitando a migração do código implementado para o sistema de controlo real. Adicionalmente, possibilita a análise dos resultados de simulação obtidos, a partir da observação gráfica da evolução das variáveis do sistema, e ajuda no dimensionamento de componentes, na previsão de falhas e na realização de melhorias no sistema, reduzindo o risco dos ensaios experimentais. Para tal, é essencial utilizar um modelo de simulação semelhante ao modelo real, aproximando os resultados de simulação aos resultados experimentais.

Neste capítulo, é apresentada a topologia do sistema de controlo modular descentralizado para conversores de eletrónica de potência ligados em cascata. De seguida, são comentados os constituintes gerais de cada submódulo e as escolhas adotadas para o controlo do sistema. Por fim, são expostos os modelos implementados e resultados das simulações computacionais.

4.2 Topologia Utilizada

Na Figura 4.1 está representada a topologia do sistema de controlo modular descentralizado para conversores de eletrónica de potência ligados em cascata. Esta topologia é composta por

um microcontrolador (DSC – *Digital Signal Controllers*), chamado de *master*, três submódulos de eletrónica de potência ligados em cascata e uma bobina de acoplamento com a rede elétrica. Cada submódulo é formado por um microcontrolador, denominado *slave*, um barramento CC e um conversor CC-CA.

Utilizaram-se três módulos ligados em cascata de forma a validar o conceito de modularidade. Com o intuito de descentralizar o sistema, cada submódulo tem um microcontrolador dedicado e a gestão de todo o sistema fica a cargo do *master*.

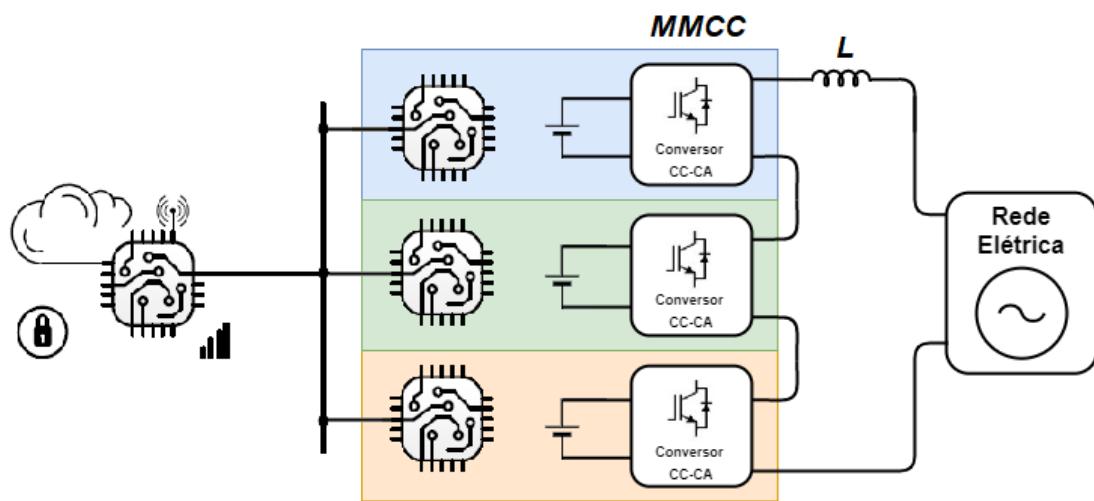


Figura 4.1 – Diagrama de blocos da topologia utilizada.

A nível de hardware, cada submódulo é constituído por um inversor VSI (*Voltage Source Inverter*) de ponte completa e um barramento CC. O barramento CC desses módulos é composto por uma fonte isolada, visto que o principal foco do trabalho é a validação dos algoritmos de controlo e comunicação e não a regulação desse barramento, como seria necessário com a utilização de condensadores.

Ao nível do controlo, diversos requisitos tiveram de ser estudados. Após a análise dos capítulos anteriores, concluiu-se que a técnica SPWM com desfasamento de fase é a técnica de modulação que melhor se adequa à topologia utilizada. Os principais motivos são a sua facilidade de implementação nos DSC e a semelhante divisão da potência entre os submódulos. Além disso, apesar do algoritmo de controlo de corrente preditivo apresentar uma resposta dinâmica rápida sem necessidade de ajuste de ganhos, optou-se por, numa primeira fase, utilizar o controlo de corrente PI pela maior experiência em projetos anteriores. Já no que diz respeito aos algoritmos de sintonização com a rede, de forma a superar as

limitações da PLL convencional, optou-se por implementar a EPLL apresentada previamente. Também se observou a necessidade de utilizar as topologias de comunicação em anel e barramento, permitindo, respetivamente, detetar o número de submódulos e garantir a comunicação simultânea do *master* para todos os *slaves*.

4.3 Simulações Computacionais

O modelo de simulação do sistema de controlo modular descentralizado para conversores de eletrónica de potência em cascata, desenvolvido no software PSIM, encontra-se representado na Figura 4.2 e Figura 4.3. Para permitir uma melhor visualização, o modelo está separado em duas figuras.

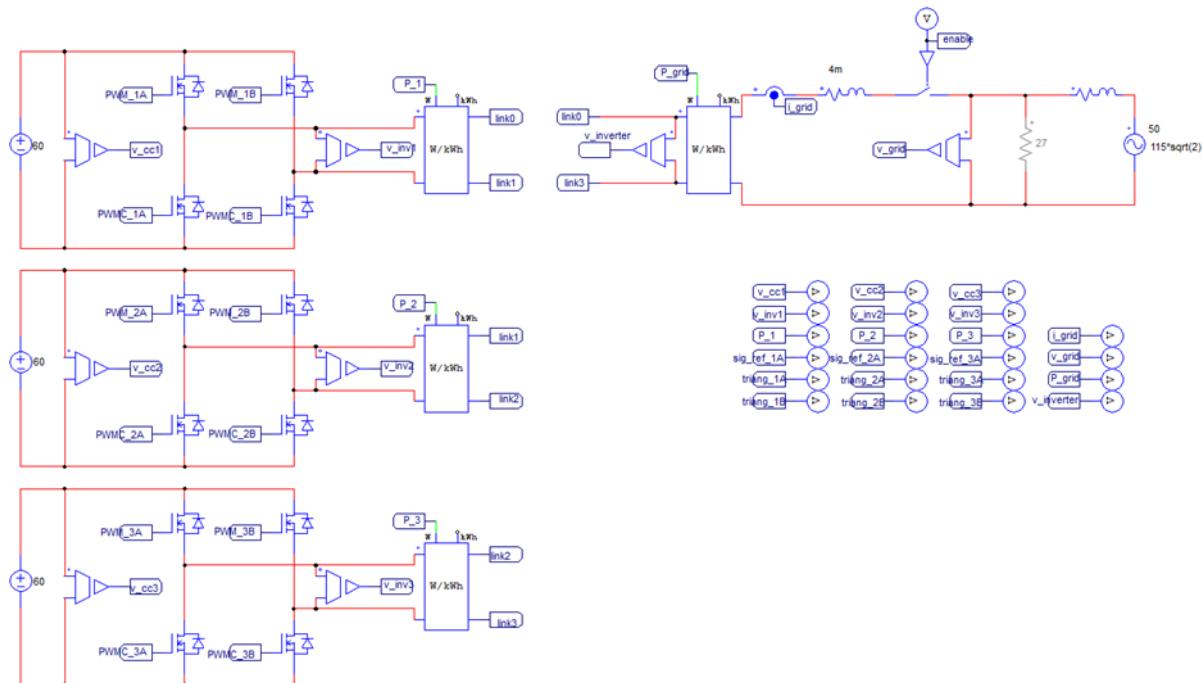


Figura 4.2 – Modelo de simulação da parte de potência da topologia utilizada.

Na Figura 4.2 é representado o andar de potência do sistema, composto pelos conversores em ponte completa, os barramentos CC e a rede elétrica. Os três conversores ligados em cascata representam o MMCC com controlo descentralizado, realizando a interfase dos barramentos CC com a rede elétrica. Os barramentos CC são constituídos por fontes de tensão constantes e a rede elétrica é representada por uma fonte de tensão sinusoidal presente no lado direito da imagem. O modelo também possui um interruptor que permite isolar o sistema da rede, comandado por o sinal de *enable* proveniente do *master*. Além disso, são visíveis os

vários sensores de tensão e de corrente usados para realizar a medição dos parâmetros necessários ao controlo do sistema.

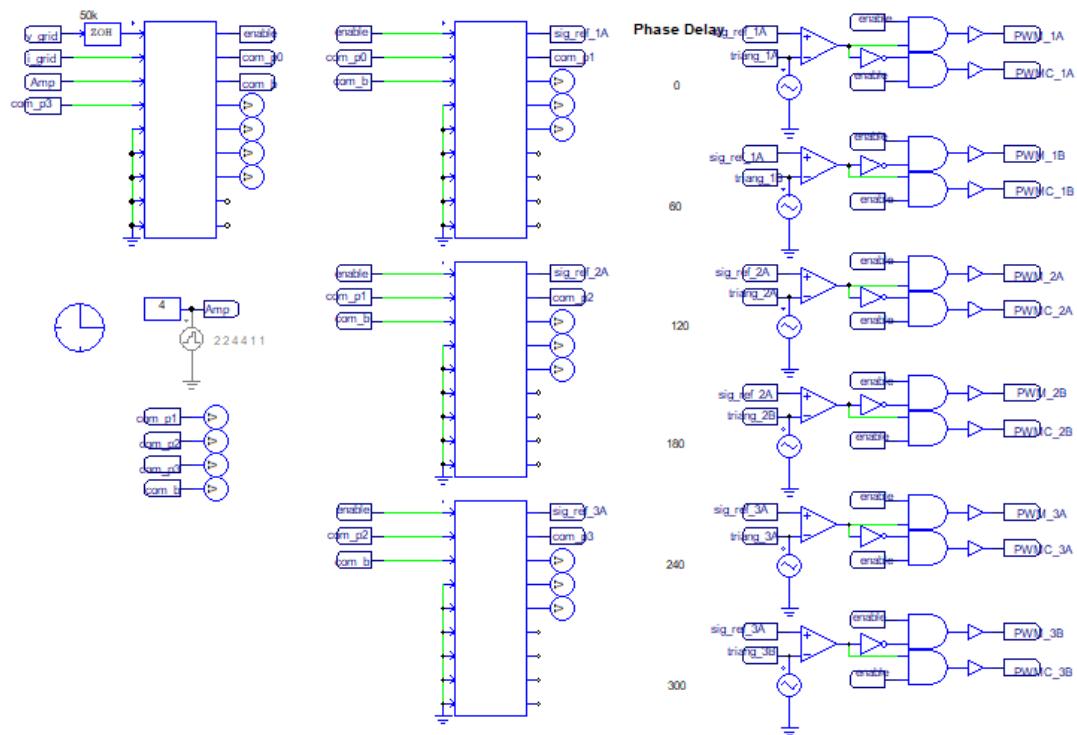


Figura 4.3 – Modelo de simulação da parte de controlo da topologia utilizada.

O sistema de controlo do modelo desenvolvido foi realizado através de blocos de processamento, *C block*, observados na Figura 4.3. Estes representam os microcontroladores e permitem implementar os algoritmos em linguagem C, facilitando a migração dos algoritmos simulados para o sistema real.

O bloco no canto superior esquerdo representa o *master*, possui como entradas as leituras dos sensores de tensão e corrente, o valor da amplitude da corrente desejado e a ligação da comunicação em anel. As saídas lógicas correspondem às ligações das comunicações, nomeadamente, ligação em anel e barramento, e ainda à *flag* responsável pela iniciação das comutações e ligação do sistema à rede, denominada de *enable*. A frequência de amostragem (f_a) foi também definida neste bloco, através do *Zero-Order Hold* (ZOH), estabelecendo-se o valor de 50 kHz.

Na parte central da figura pode observar-se mais três C block que representam os *slaves*. Como entradas possuem o sinal de *enable* proveniente do *master* e as ligações de ambas as comunicações. As saídas lógicas correspondem aos sinais de referência para as comutações

dos semicondutores, ao sistema de comunicação em anel e aos valores do desfasamento das ondas portadoras.

À direita na imagem, encontra-se a implementação da técnica de modulação PSC aplicada no MMCC, onde foi estabelecida uma frequência de comutação (f_c) para os semicondutores de 10 kHz. Cada circuito é composto por um comparador de tensão que compara a onda moduladora com a portadora desfasada, gerando o PWM para acionar os respectivos semicondutores; uma porta lógica NOT responsável por produzir o PWM complementar; e duas portas lógicas AND responsáveis por garantir que os sinais apenas são enviados após o sinal de *enable*.

As principais especificações do sistema implementado encontram-se apresentadas na Tabela 4.1. O MMCC foi dimensionado para trabalhar com tensões fase-neutro 230 V e com uma corrente máxima de, aproximadamente, 4 A. Cada barramento é composto por uma fonte de tensão com 125 V, o que perfaz uma potência nominal do sistema de aproximadamente, 1050 VA, 350 VA por módulo.

Tabela 4.1 – Características nominais do conversor modular multinível em cascata.

Parâmetro	Valor	Unidade
Valor eficaz da tensão da rede	230	V
Frequência da tensão da rede	50	Hz
Potência nominal do MMCC	1500	VA
Amplitude da corrente do MMCC	4	A
Tensão nominal do barramento CC	125	V
Frequência de comutação (f_c)	10	kHz
Frequência de amostragem (f_a)	50	kHz
Bobina de acoplamento (L)	2,7	mH

4.3.1 Resultados de Simulação para Três Submódulos

No controlo do sistema é utilizada a técnica de SPWM com deslocamento de fase, onde todas as portadoras têm a mesma amplitude e frequência, mas defasadas em $2\pi/n$. Em simulação,

os desfasamentos são previamente calculados e atribuídos aos parâmetros das portadoras, como pode ser observado na Figura 4.4.

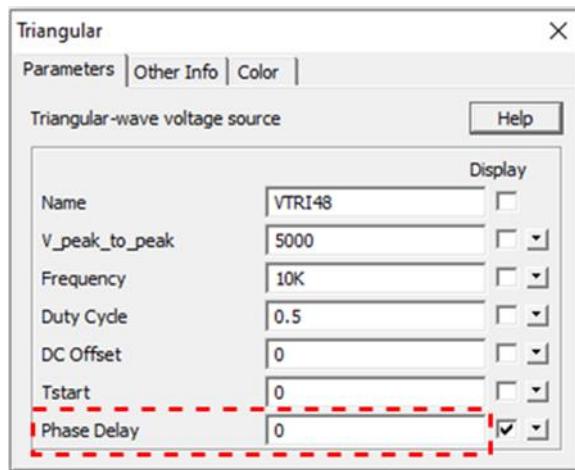


Figura 4.4 – Interface do PSIM para os parâmetros de uma forma de onda triangular.

O protótipo laboratorial deve possuir um mecanismo que permite identificar os submódulos ligados, além de calcular e atribuir os respetivos desfasamentos de forma automática. Dessa forma, a topologia de comunicação em anel foi emulada, permitindo testar e validar os algoritmos. De notar que, os parâmetros das portadoras foram adequados aos valores usados no microcontrolador para uma frequência de 10 KHz, aproximando a simulação ao sistema real.

A topologia é constituída por uma série de submódulos ligados através de um meio físico. Essas ligações são representadas pelas *labels* com_p , sendo que, com_{p0} corresponde à ligação entre o *master* e o primeiro *slave*, com_{p1} corresponde à ligação entre o primeiro e segundo *slave* e assim sucessivamente até conectar novamente ao *master*. Na Figura 4.5, são apresentados os resultados da comunicação em anel para um sistema com três submódulos ligados. Como pode ser observado, o *master* envia a trama de comunicação vazia (com valor 0), esta é recebida pelo primeiro *slave* que a lê, incrementa o valor e volta a enviar para o *slave* seguinte. No final da comunicação, o *master* volta a receber a trama com o valor total dos submódulos ligados.

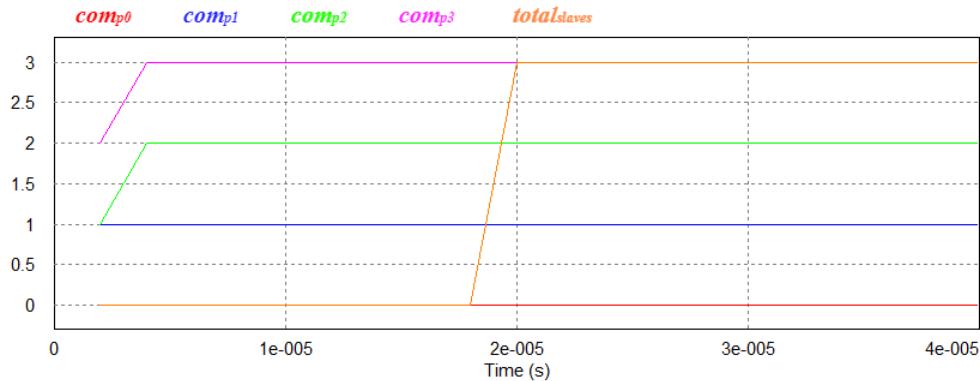


Figura 4.5 – Resultado da comunicação em anel para um sistema com três submódulos ligados.

Através do total de submódulos ligados o desfasamento é calculado e enviado para os respetivos *slaves*. Como referido anteriormente, as ondas portadoras estão desfasadas $2\pi/n$, onde n representa o número de braços do MMCC. Visto que cada submódulo é constituído por uma ponte completa (dois braços), o número total de braços do sistema é 6. Os resultados obtidos são apresentados na Figura 4.6

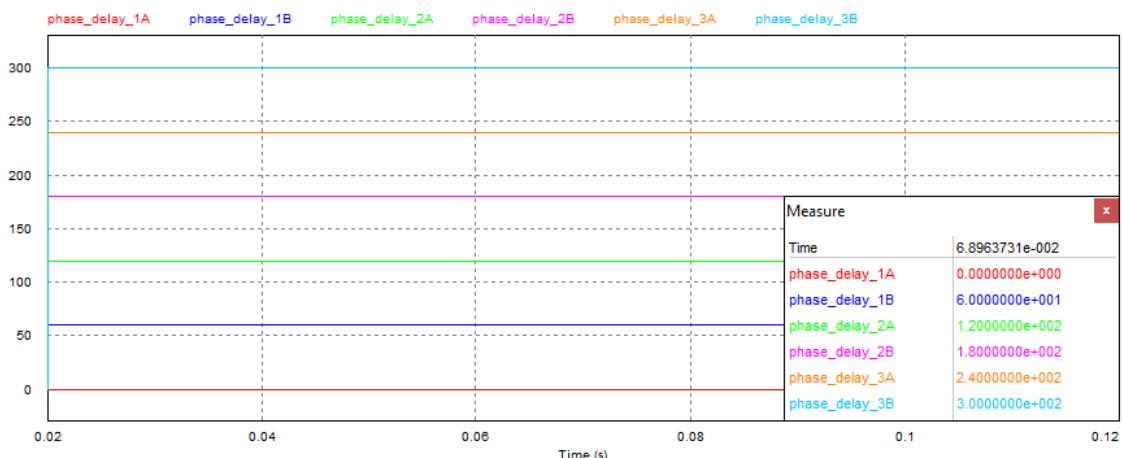


Figura 4.6 – Valores calculados para o desfasamento de 6 ondas portadoras.

A outra topologia de comunicação utilizada foi a topologia em barramento, como referido nos capítulos anteriores. Ainda que em simulação estes atrasos sejam praticamente inexistentes, foi criada uma comunicação em barramento para testar e validar o seu princípio de funcionamento. No sistema essa comunicação é representada pelas *labels* com_b , que são responsáveis por enviar a forma de onda modulador calculada pelo algoritmo de controlo de corrente. No controlador PI utilizado, o cálculo é feito através das correntes de referência e saída.

Para obter a forma de onda da corrente de referência é necessário que o sistema de controlo contenha um mecanismo de sincronização com a tensão da rede elétrica. A sincronização, tal como referido anteriormente, é feita recorrendo à técnica EPLL. Esta é responsável por gerar uma onda sinusoidal unitária em fase com um sinal de entrada, neste caso, a tensão da rede elétrica. Os sinais de entrada e saída da EPLL, v_{rede} e pll_{uni} , estão representados na Figura 4.7, onde se comprova o bom funcionamento da mesma, sendo os valores de THD%f de 2,93% e 0,18%, respetivamente. Como se pode constatar, o algoritmo de sincronismo adquire a frequência e a fase da componente fundamental quase instantaneamente. De salientar que, apesar da sincronização da rede elétrica ser feita a partir do primeiro momento da simulação, o sistema só é conectado com a rede aos 0,02 s, dando tempo para completar a sincronização e as comunicações de configuração.

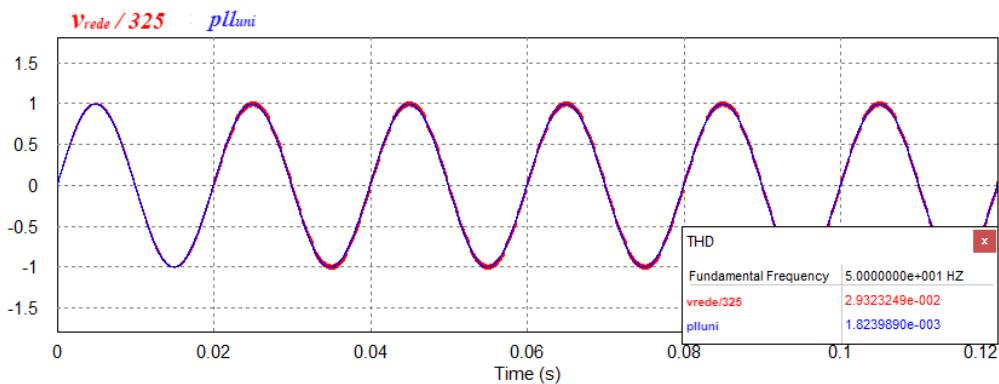


Figura 4.7 – Formas de onda do sinal de entrada da EPLL (v_{rede}) dividido por 325 e sinal de saída (pll_{uni}).

A referência de corrente, i_{ref} , utilizada na técnica de controlo é calculada multiplicando o sinal unitário da EPLL, pll_{uni} , com o valor de amplitude desejada, i_{amp} . Na Figura 4.8 pode observar-se a corrente de referência calculada para uma amplitude máxima de 4 A.

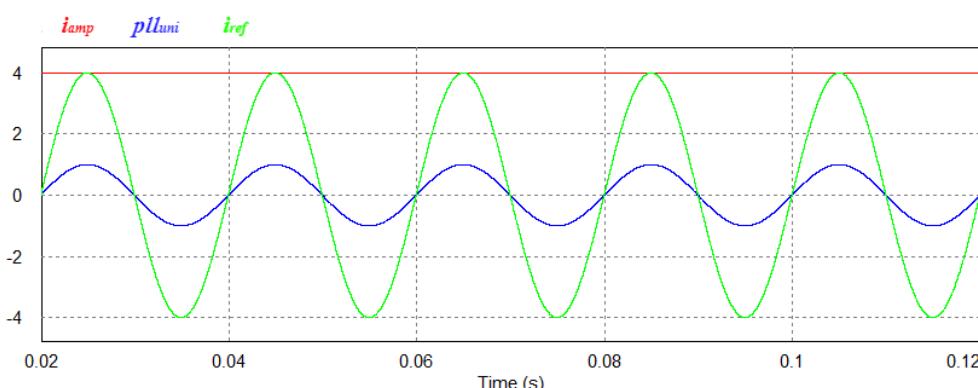


Figura 4.8 – Formas de onda da corrente de referência (i_{ref}), do sinal unitário da EPLL (pll_{uni}) e da amplitude máxima (i_{amp}) da referência.

A corrente sintetizada pelo conversor, i_{rede} , e a corrente de referência são apresentadas na Figura 4.9. Estes resultados foram obtidos utilizando o controlador PI com ganhos $k_p = 1000$ e $k_i = 40$, encontrados empiricamente. Tal como mencionado no terceiro capítulo, é possível reparar que a corrente de saída está atrasada em relação à referência. De notar que, o THD% da forma de onda da corrente de saída é 0,58%.

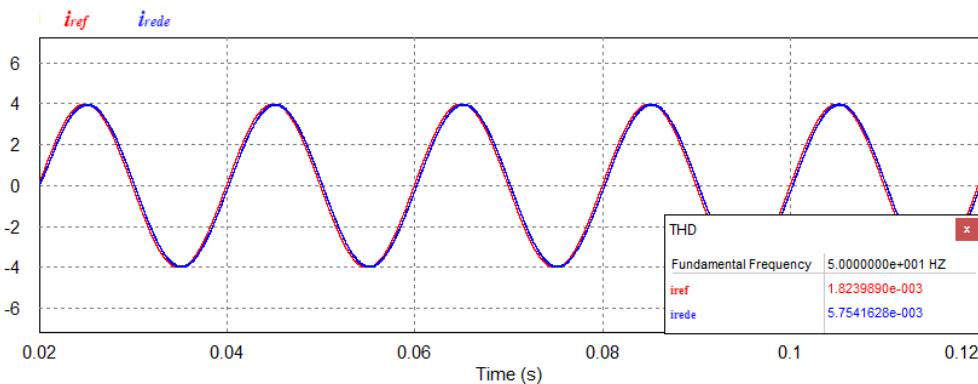


Figura 4.9 – Formas de onda da corrente de referência (i_{ref}) e da corrente sintetizada pelo inversor (i_{rede}).

Na Figura 4.10 são expostos os valores da tensão à saída do inversor, da tensão da rede e da corrente sintetizada pelo inversor. Visto que o sentido positivo da corrente é do inversor para a rede, a injeção de energia na rede elétrica é realizada quando a corrente está em fase com a tensão. Dessa forma, é possível observar o correto funcionamento da solução proposta. Além disso, na forma de onda da tensão de saída do inversor são visíveis os diferentes níveis característicos dos conversores MMCC. Como esperado, a forma de onda apresenta sete níveis de tensão correspondendo aos três submódulos ligados em cascata.

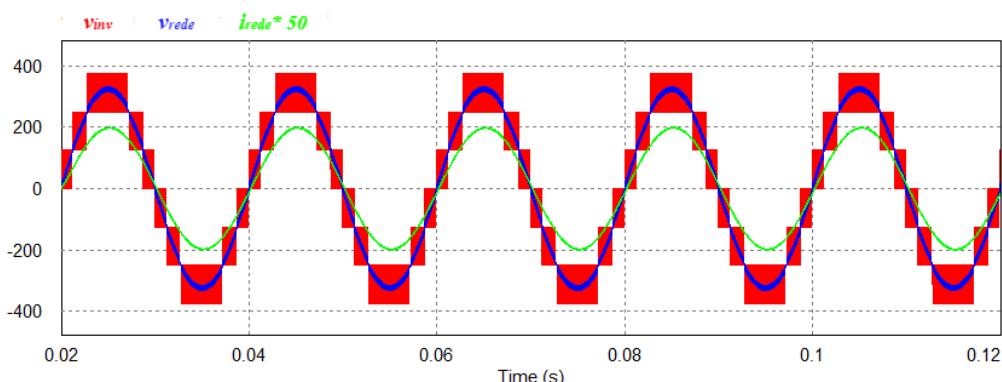


Figura 4.10 – Formas de onda da tensão à saída do inversor (v_{inv}), da tensão da rede elétrica (v_{rede}) e da corrente sintetizada pelo inversor (i_{rede}) multiplicada por 50.

Com o intuito de comprovar que os diferentes níveis de tensão apresentados correspondem as combinações das tensões dos barramentos CC, são apresentadas as tensões nos barramentos e a tensão de saída do inversor na Figura 4.11. De notar que, a tensão dos barramentos é de 125 V e que ao valor máximo da tensão de saída é de aproximadamente 375 V.

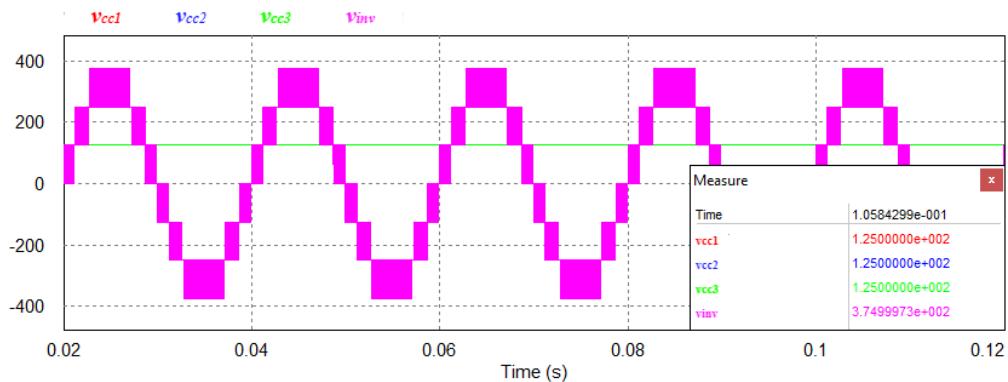


Figura 4.11 – Formas de onda da tensão do barramento CC no primeiro, segundo e terceiro submódulos (V_{cc1} , V_{cc2} e V_{cc3}) e da tensão à saída do inversor (V_{inv}).

Por fim, na Figura 4.12 são apresentadas as potências em cada submódulo e a potência total do sistema. Mais uma vez, é possível comprovar o correto funcionamento do sistema, visto que, a potência está igualmente dividida entre os três submódulos.

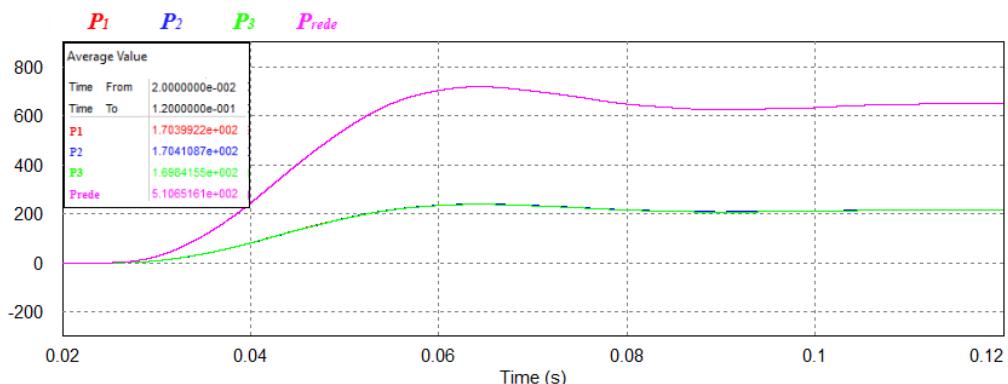


Figura 4.12 – Formas de onda da potência no primeiro, segundo e terceiro submódulos (P_1 , P_2 , P_3) e da potência total do sistema (P_{rede}).

Além dos resultados apresentados, testou-se o comportamento dinâmico do sistema. Para isso, fez-se variar a amplitude de pico desejada para a corrente de saída do inversor da seguinte forma: 2 A pico de 0,02 s a 0,04 s; 4 A pico de 0,06 s a 0,08 s; 1 A pico de 0,1 s a 0,12 s. Os resultados para essa simulação são apresentados na Figura 4.13.

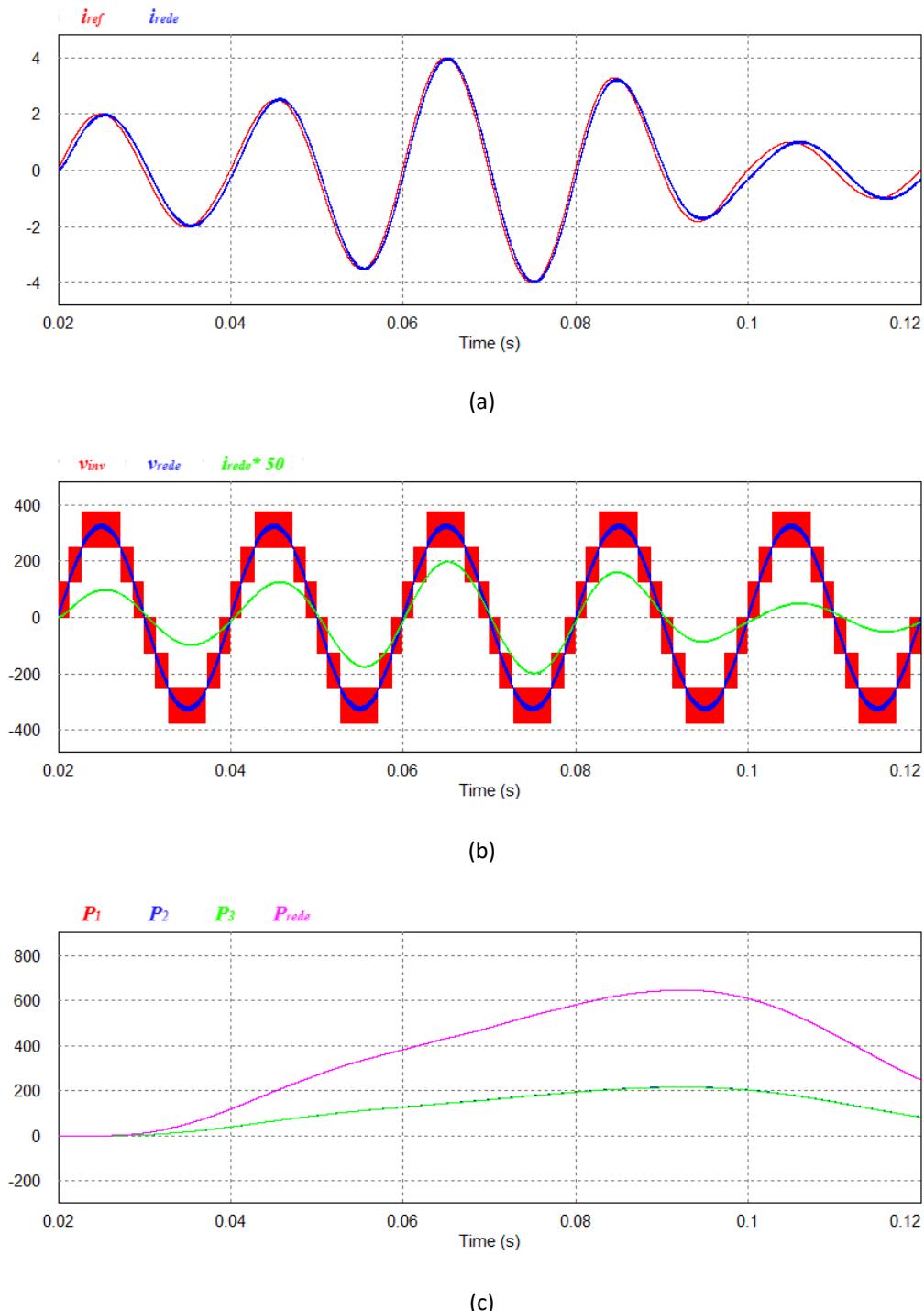


Figura 4.13 – Formas de onda: (a) corrente de referência (i_{ref}) e corrente sintetizada pelo inversor (i_{rede}); (b) tensão à saída do inversor (v_{inv}), tensão da rede elétrica (v_{rede}) e corrente sintetizada pelo inversor (i_{rede}) multiplicada por 50; (c) potência no primeiro, segundo e terceiro submódulos (P_1, P_2, P_3) e potência total do sistema ($Prede$).

A corrente de referência, i_{ref} , e a corrente sintetizada pelo inversor, i_{rede} , são apresentadas na Figura 4.13 (a), sendo que, o atraso da corrente em relação à referência aumenta com a diminuição da amplitude. Na Figura 4.13 (b) são expostos os valores da tensão à saída do inversor, v_{inv} , da tensão da rede, v_{rede} , e da corrente sintetizada pelo inversor, i_{rede} , multiplicada

50 vezes. As formas de onda da potência em cada submódulo (P_1, P_2, P_3) e a potência total do sistema, P_{rede} , são mostradas na Figura 4.13 (c). Desta forma, é possível observar o correto funcionamento da solução proposta para variações da amplitude de corrente.

4.3.2 Resultados de Simulação para Dois Submódulos

Como referido anteriormente, a topologia utilizada é constituída por três submódulos ligados em cascata. No entanto, de forma a mostrar a modularidade do sistema e a adaptabilidade dos algoritmos, foram adquiridos resultados para dois submódulos ligados. Para isso, foram feitas algumas modificações no modelo de simulação, nomeadamente: a alteração da tensão dos barramentos CC para 187,5 V, mantendo os 375 V na soma das tensões; desconectaram-se as comunicações e o hardware do terceiro submódulo através das *labels* de configuração e ajustaram-se os desfasamentos das ondas portadoras.

De forma semelhante aos resultados anteriores, na Figura 4.14 são apresentados os resultados da comunicação em anel para um sistema com dois submódulos ligados. O princípio de funcionamento continua a ser o mesmo e pode ser observado que o número total de submódulos ligados é ajustado automaticamente.

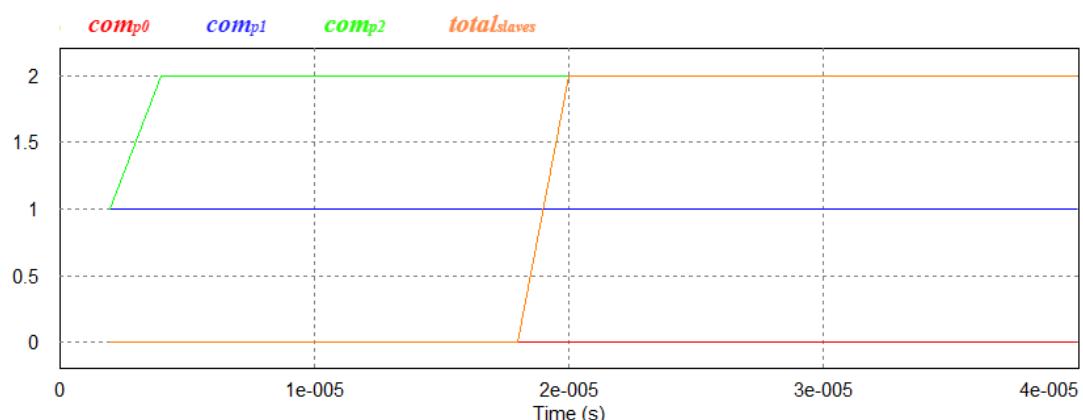


Figura 4.14 – Resultado da comunicação em anel para um sistema com dois submódulos ligados.

Neste caso, o sistema passa a ser constituído por quatro braços, duas pontes completas ligadas em cascata, obtendo os desfasamentos apresentados na Figura 4.15. Como as ondas portadoras estão desfasadas $2\pi/n$, onde n representa o número de braços do sistema, o desfasamento entre as portadoras é de 90°.

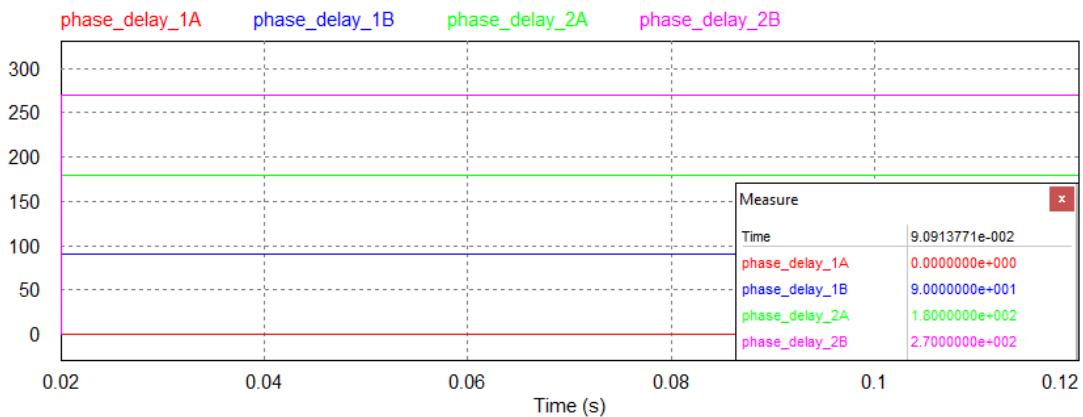
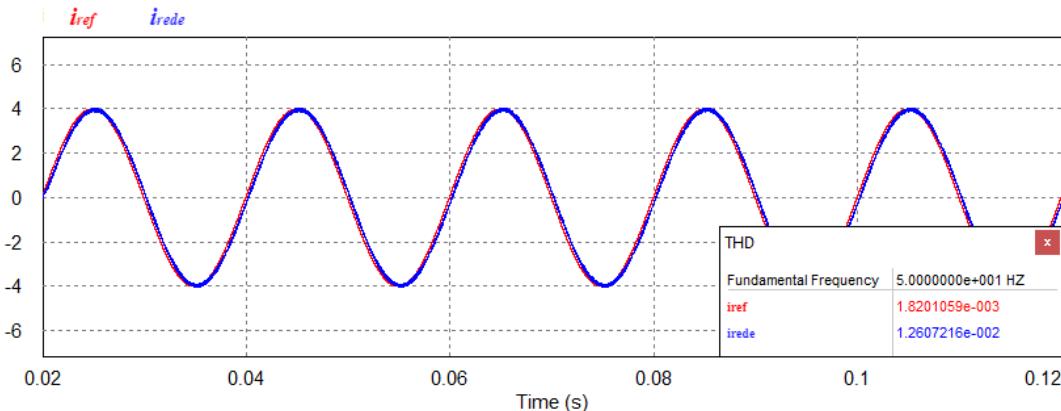


Figura 4.15 – Valores calculados para o desfasamento de 4 ondas portadoras.

Uma vez que, tanto o mecanismo de sincronização com a rede elétrica, como o cálculo da corrente de referência utilizada na técnica de controlo são independentes do número de submódulos, os resultados apresentados na Figura 4.7e Figura 4.8 não se alteram.

A corrente sintetizada pelo conversor e a corrente de referência são mostradas na Figura 4.16. Estas foram obtidas utilizando o controlador PI com os ganhos das simulações anteriores, $k_p = 1000$ e $k_i = 40$. De notar que, o THD%f da forma de onda da corrente aumenta para 1,26%.

Figura 4.16 – Formas de onda da corrente de referência (i_{ref}) e da corrente sintetizada pelo inversor (i_{rede}).

Na Figura 4.17 são expostos os valores da tensão à saída do inversor, da tensão da rede, da corrente sintetizada pelo inversor e da tensão dos barramentos CC dos submódulos. De salientar que, na forma de onda da tensão de saída do inversor são visíveis os diferentes níveis característicos dos conversores MMCC. Como esperado, a forma de onda apresenta cinco níveis de tensão correspondendo aos dois submódulos ligados em cascata. Além disso, é também apresentada a tensão do barramento do primeiro submódulo e a soma dos

barramentos, comprovando que os diferentes níveis de tensão apresentados correspondem as combinações das tensões dos barramentos.

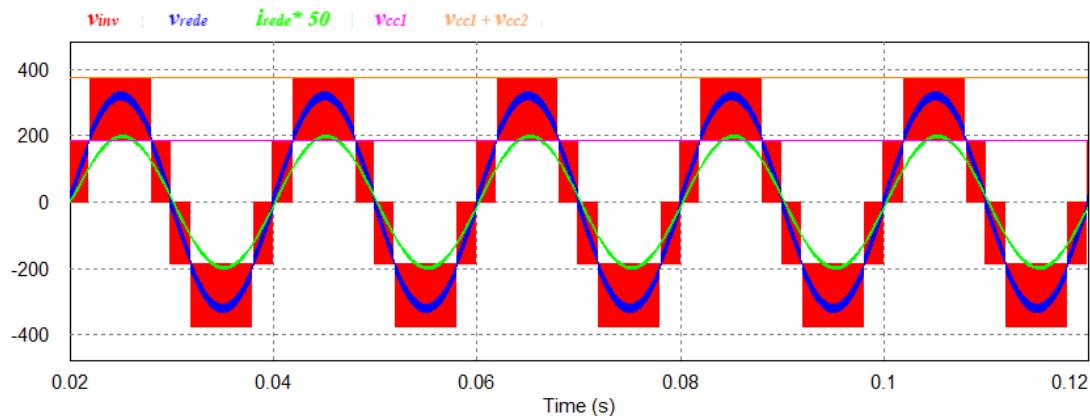


Figura 4.17 – Formas de onda da tensão à saída do inversor (v_{inv}), da tensão da rede elétrica (v_{rede}), da corrente sintetizada pelo inversor (i_{rede}) multiplicada por 50 e da tensão do barramento CC no primeiro e segundo submódulos (V_{cc1}, V_{cc2}).

Por fim, na Figura 4.18 são apresentadas as potências em cada submódulo e a potência total do sistema, sendo possível comprovar o correto funcionamento do sistema, visto que, a potência está igualmente dividida entre os dois submódulos.

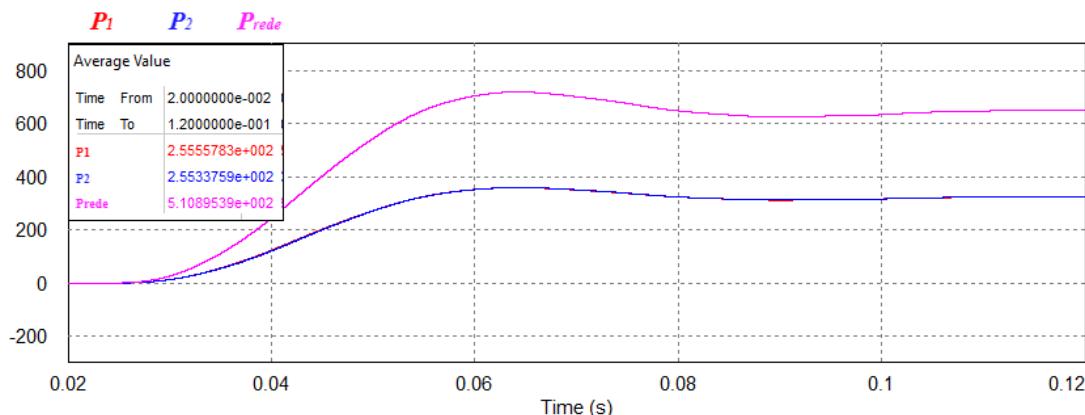


Figura 4.18 – Formas de onda da potência no primeiro e segundo submódulos (P_1 e P_2) e da potência total do sistema (P_{rede}).

Mais uma vez, testou-se o comportamento dinâmico do sistema. Para isso, fez-se variar a amplitude de pico desejada para a corrente de saída do inversor da seguinte forma: 2 A pico de 0,02 s a 0,04 s; 4 A pico de 0,06 s a 0,08 s; 1 A pico de 0,1 s a 0,12 s. Os resultados para essa simulação são apresentados na Figura 4.19.

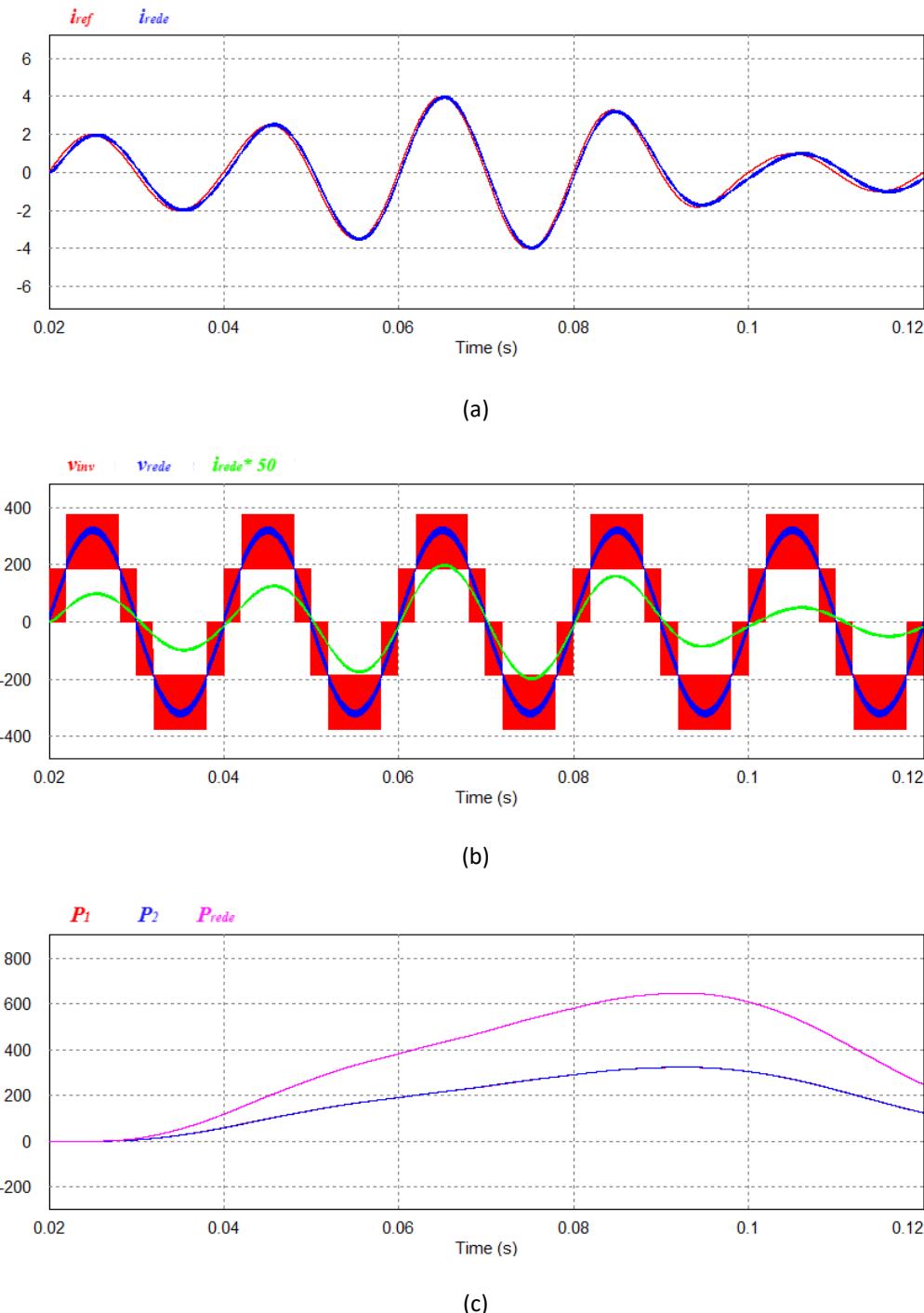


Figura 4.19 – Formas de onda: (a) corrente de referência (i_{ref}) e corrente sintetizada pelo inversor (i_{rede}); (b) tensão à saída do inversor (v_{inv}), tensão da rede elétrica (v_{rede}) e corrente sintetizada pelo inversor (i_{rede}) multiplicada por 50; (c) potência no primeiro, segundo e terceiro submódulos (P_1, P_2) e potência total do sistema (P_{rede}).

A corrente de referência, i_{ref} , e a corrente sintetizada pelo inversor, i_{rede} , são apresentadas na Figura 4.19(a), sendo que, o atraso da corrente em relação à referência aumenta com a diminuição da amplitude. Na Figura 4.19(b) são expostos os valores da tensão à saída do inversor, v_{inv} , da tensão da rede, v_{rede} , e da corrente sintetizada pelo inversor, i_{rede} , multiplicada

50 vezes. As formas de onda da potência em cada submódulo, P_1 e P_2 , e a potência total do sistema, P_{rede} , são mostradas na Figura 4.19(c). Desta forma, é possível observar o correto funcionamento da solução proposta para variações de amplitude da corrente.

4.4 Conclusões

Neste capítulo foi apresentada a topologia do sistema de controlo modular descentralizado para conversores de eletrónica de potência ligados em cascata, desde o hardware que constitui o sistema, até às escolhas adotadas para o controlo do mesmo. De uma forma complementar, realizaram-se simulações computacionais com o objetivo de validar a topologia utilizada, ajudando a prever o comportamento do sistema e assim, melhorar a sua resposta e o dimensionamento dos componentes.

Inicialmente, foi realizada uma descrição detalhada do modelo de simulação e apresentadas as principais características do sistema. Posteriormente, foram mostrados os resultados de simulação, a partir dos quais foi possível comprovar que todo o sistema, incluindo os algoritmos de controlo implementados, funciona de acordo com o esperado.

Os resultados foram obtidos a partir de dois modelos de simulação distintos, um correspondente à topologia utilizada com três submodulos ligados em cascata e outro modelo com apenas dois submodulos, permitindo mostrar a modularidade do sistema e a adaptabilidade dos algoritmos. Além disso, foram feitos testes dinâmicos para ambos os modelos, apresentando resultados com a variação da amplitude na corrente sintetizada e comprovando, novamente, o correto funcionamento do sistema.

As principais diferenças entre os resultados dos dois modelos foram o número de níveis na tensão de saída do inversor (três submódulos apresenta sete níveis e dois submódulos apenas cinco) e a THD%f da forma de onda da corrente de saída (três submódulos THD%f = 0,57% e dois submódulos THD%f = 1,26%). Estes dois fenómenos estão relacionados, pois quanto maior o número de submódulos, maior é o número de níveis de tensão de saída sintetizados. Com o aumento do número de níveis, maior é a semelhança com uma sinusoide, melhorando o conteúdo harmônico.

Confirmou-se ainda o correto funcionamento do mecanismo de sincronização com a rede elétrica; a igual divisão da potência pelos submódulos e a adaptabilidade dos algoritmos que calcularam automaticamente o número de submódulos ligados e os respetivos desfasamentos.

Capítulo 5

Implementação de um Protótipo Laboratorial

5.1 Introdução

Uma vez aprovada a operação da topologia utilizada em simulação computacional, é realizada a descrição da implementação do protótipo laboratorial do sistema de controlo modular descentralizado para conversores de eletrónica de potência em cascata. Este capítulo está dividido em duas partes: hardware de potência e controlo.

Inicialmente, são apresentados os elementos que constituem o andar de potência, nomeadamente, os conversores CC-CA em ponte completa que formam o MMCC e todos os componentes necessários para o acoplamento do com a rede elétrica. Posteriormente, é mostrado o sistema de controlo usado, abrangendo os microcontroladores, os sensores de corrente e tensão e as placas de circuito impresso (PCB – *Printed Circuit Board*) desenvolvidas para o condicionamento de sinal e deteção de erros, comando e *driver* dos semicondutores de potência e comunicações. Posto isto, o protótipo do sistema implementado está representado e legendado na Figura 5.1 e na Figura 5.2, respetivamente.



Figura 5.1 – Protótipo laboratorial implementado.

De notar que todas as PCB foram desenvolvidas no software Altium Designer. Esta ferramenta foi escolhida pela sua facilidade de utilização e vasto material de suporte, ampla utilização em contexto empresarial e visualização 3D das placas de forma individual e integrada. Alguns exemplos das PCB desenvolvidas neste software são apresentados nas secções seguintes e os seus ficheiros encontram-se no apêndice. Salientar também que os componentes existentes no GEPE (Grupo de Eletrónica de Potência e Energia) foram sempre preferencialmente considerados devido à facilidade de aquisição, experiência de utilização e custos de compra de novos componentes.

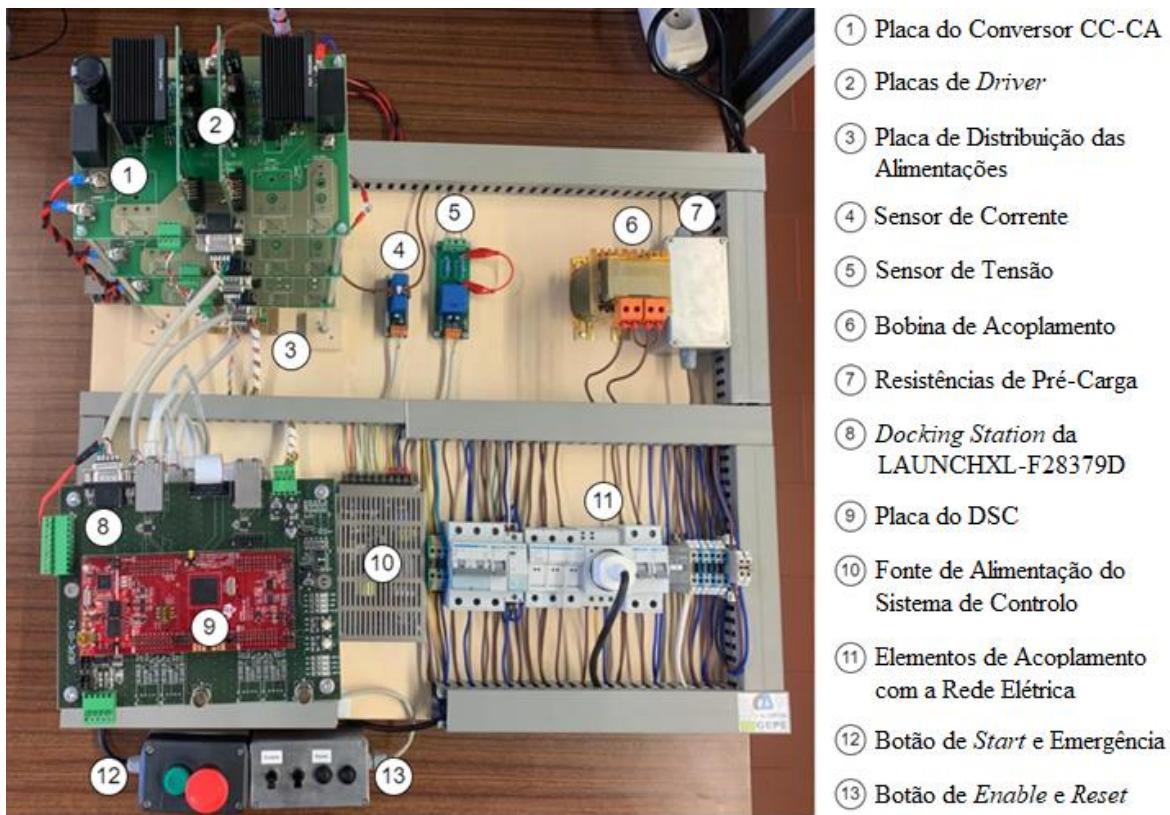


Figura 5.2 – Protótipo laboratorial legendado.

5.2 Hardware de Potência

Ao longo desta secção, é analisado o hardware de potência utilizado no protótipo laboratorial, principalmente os conversores CC-CA em ponte completa que constituem os submódulos e os sistemas de acoplamento com a rede elétrica, bem como as respetivas proteções. Estes conversores foram dimensionados com o intuito de cumprir os principais requisitos do sistema, nomeadamente a injeção de energia na rede elétrica. Neste sentido, para o correto dimensionamento dos componentes foi necessário especificar as características de operação,

sendo estas apresentadas na Tabela 4.1, mantendo as características das simulações computacionais.

5.2.1 Conversores de Potência CC-CA

A topologia utilizada é constituída por três submódulos de conversores CC-CA em ponte completa ligados em cascata. Ou seja, seria necessário desenvolver uma PCB capaz de ser replicada e que cobrisse todos os requisitos dos submódulos. Além disso, foi definido o objetivo de criar uma PCB modular que pudesse ser reutilizada em projetos futuros. Para isso, esta tem de incluir todos os componentes necessários para fazer face a diferentes problemas.

Dessa forma, optou-se por colocar, numa só placa, os seguintes componentes:

- Condensadores eletrolíticos e de polipropileno à entrada e saída do conversor;
- Quatro semicondutores que formam a ponte completa, permitindo usar diferentes gamas de semicondutores desde que estes possuam encapsulamento TO-247;
- Dissipadores de calor para os semicondutores;
- Circuitos de proteção de *gate* e *snubber*;
- Sensores de tensão à entrada e saída do conversor e sensor de corrente;
- Conectores de potência, de alimentação e para receber/enviar sinais.

Na Figura 5.3, encontra-se a placa de circuito impresso desenvolvida para a implementação do inversor que constitui os submódulos.

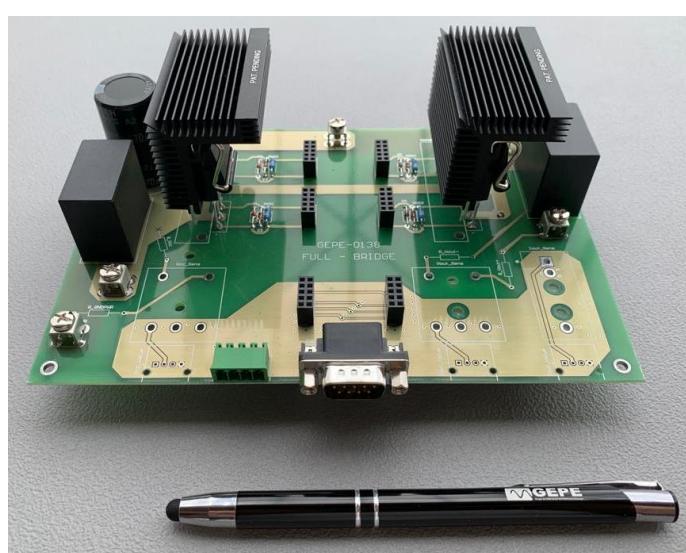


Figura 5.3 – Placa do conversor CC-CA.

À entrada do conversor, escolheu-se combinar dois tipos de condensadores, um condensador eletrolítico de $560 \mu\text{F}$ (canto superior esquerdo da Figura 5.3) e um de polipropileno de $10 \mu\text{F}$, associando a grande capacidade do condensador eletrolítico à rapidez de resposta a variações de dv/dt do condensador de polipropileno. Já para a saída optou-se por usar apenas um condensador de polipropileno de $10 \mu\text{F}$.

Neste protótipo utilizaram-se os MOSFET de potência da *Littelfuse*, cuja referência do fabricante é IXFH42N50P2. Estes suportam uma tensão de 500 V e uma corrente 42 A . Além disso, são populares para aplicações de “*hard switching*” e modo ressonante, oferecendo uma baixa resistência de condução e uma excelente robustez com um diodo rápido interno [122, 123]. Os dissipadores de calor utilizados foram da *Ohmite*, de referência VRA-55-101E. Estes foram projetados para dois semicondutores com encapsulamento TO-247, usam um sistema de clip e podem ser soldados em PCB, tornando-os ideais para soluções compactas [124]. Circuitos de proteção contra sobretensões entre o *drain* e *source* também foram implementados para cada MOSFET, assim como circuitos de proteção de *gate*.

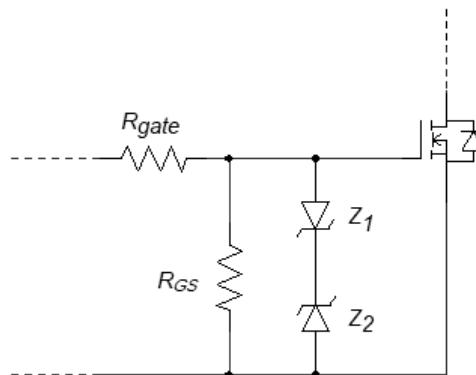


Figura 5.4 – Circuitos de proteção implementados nos MOSFET.

Como se pode observar na Figura 5.4, as resistências R_{gate} e R_{GS} e os diodos de zener Z_1 e Z_2 constituem o circuito de proteção de *gate*. Apesar da R_{gate} não se encontrar presente na placa de potência, ela é adicionada à placa de *driver*, servindo para limitar os picos de corrente na *gate*. Os restantes componentes são responsáveis por proteger a *gate* dos picos de tensão. Além disso, o circuito de *snubber* é composto por um condensador de polipropileno de 100 nF em paralelo com cada braço do conversor [125]. Estes circuitos de proteção foram implementados o mais próximo possível dos MOSFET com o intuito de diminuir as impedâncias no circuito e melhorar o funcionamento.

Podem observar-se vários conectores de potência na Figura 5.3, tais como: conectores de entrada do conversor que se encontram no canto inferior esquerdo; um conector referente ao ponto médio do primeiro braço que se situa na parte superior ao centro, possibilitando usar esta PCB como meia ponte; e conectores de saída do conversor que se encontram na parte superior do lado direito. Além destes, foram colocados seis conectores centrais para alimentar e integrar as placas de *driver*, um conector DB9 para receber os sinais de PWM e sinais de *enable/disable* e foram projetados conectores mini XLR para alimentar os sensores e enviar o sinal medido.

Apesar de ser possível acoplar os sensores de corrente e tensão à placa de potência, optou-se por usar duas placas de sensores externos: um de corrente e outro de tensão, desenvolvidas no GEPE. Os sensores seriam semelhantes, com o mesmo princípio de funcionamento, porém, desta forma, foi possível minimizar o custo do projeto reutilizando material existente.

5.2.2 Circuito de Acoplamento com a Rede Elétrica e Proteções

O esquema elétrico do circuito de acoplamento com a rede elétrica está ilustrado na Figura 5.5. O circuito de potência incorpora os elementos de acoplamento da rede elétrica com o conversor CC-CA, as fontes de tensão dos barramentos CC e a fonte de alimentação *TRACO* que alimenta o circuito de controlo.

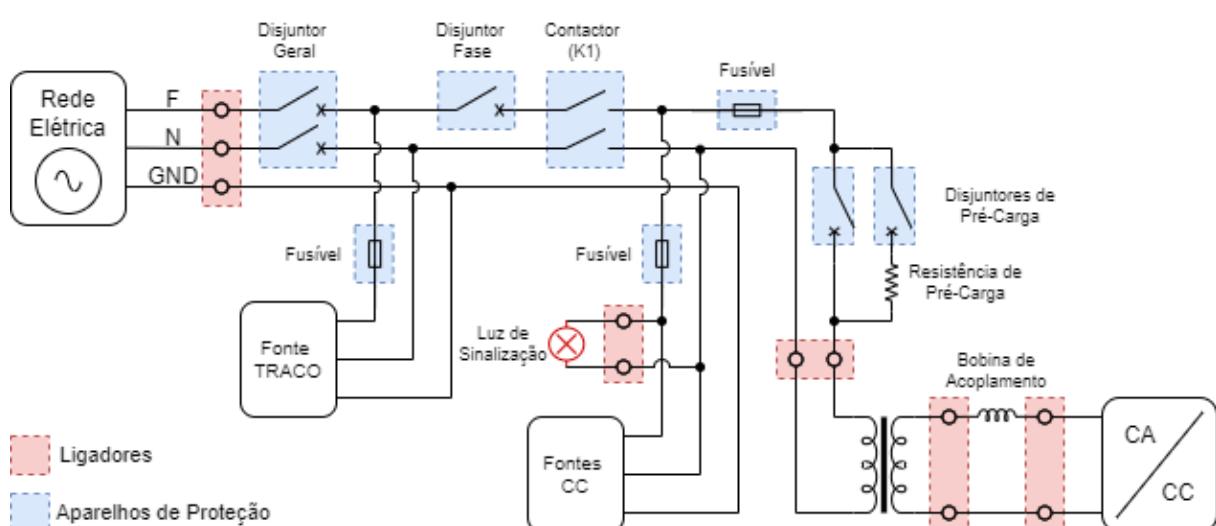


Figura 5.5 – Esquema elétrico do circuito de acoplamento com a rede elétrica.

Os aparelhos de proteção responsáveis pela ligação da rede elétrica aos restantes constituintes do sistema são apresentados na Figura 5.6.

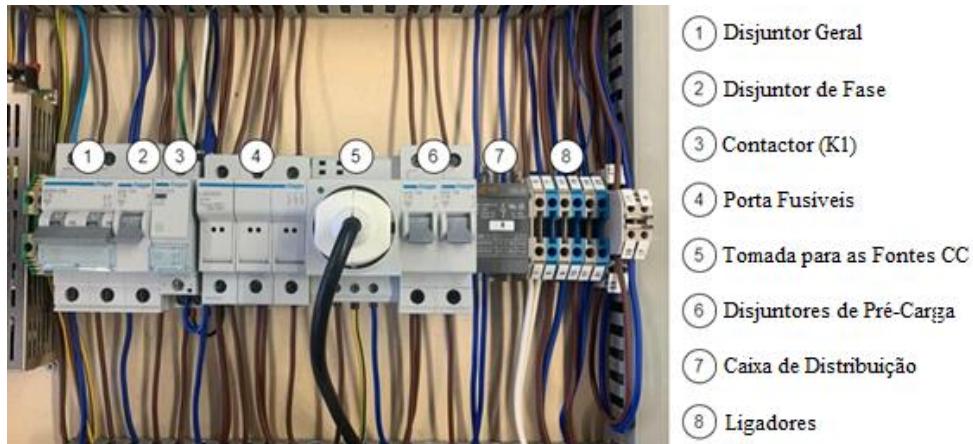


Figura 5.6 – Aparelhos de proteção do circuito de acoplamento com a rede elétrica.

O disjuntor geral do sistema destina-se a ligar ou desligar toda a conexão com a rede elétrica e o disjuntor de fase serve para isolar o hardware de potência e controlo. Ou seja, se o disjuntor geral estiver ligado e o disjuntor de fase desligado, a parte de potência encontra-se desconectada enquanto a fonte *TRACO* alimenta a parte de controlo. Isto permite realizar alterações de código e testes com maior segurança, como a verificação das comunicações, das leituras dos canais ADC (*Analog to Digital Converter*) e dos sinais de comando.

Um mecanismo de proteção por hardware é efetuado através do contactor *K1* e está representado na Figura 5.7(a). Mesmo com ambos disjuntores ativos, o contactor garante que o sistema só é ligado quando o botão de *start* é pressionado. Além disso, este mecanismo de proteção permite fazer uma paragem de emergência através de uma botoneira com encravamento mecânico, se esta for pressionada, o contactor *K1* abre, desconectando o hardware de potência da rede elétrica e desligando os barramentos CC.

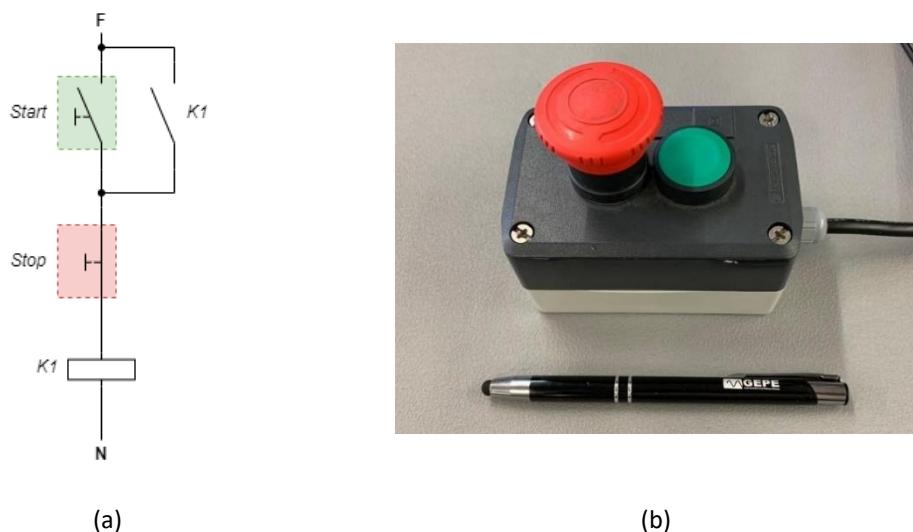


Figura 5.7 – Acionamento do contactor: (a) esquema elétrico; (b) botão de start e botoneira de emergência.

Outros mecanismos de proteção são empregues no sistema. Além dos fusíveis dimensionados para proteger contra curto-circuitos, também são utilizadas duas resistências de pré-carga de $50\ \Omega$ ligadas em paralelo, mostradas na Figura 5.8(b). Esta resistência é fundamental, pois evita picos de corrente indesejados que ocorrem no instante em que o conversor é ligado à rede elétrica. Uma vez concluída a pré-carga, é realizado o *bypass* às resistências.

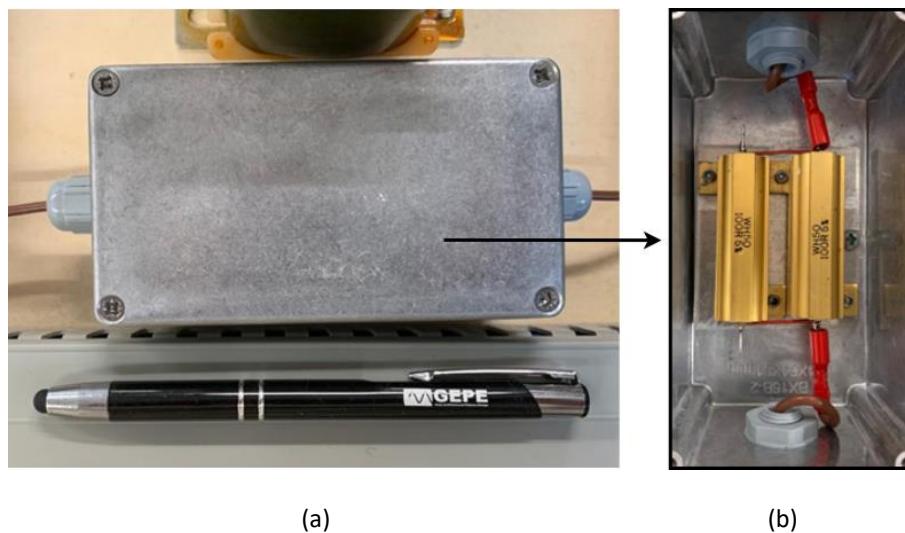


Figura 5.8 – Resistências de pré-carga: (a) vista exterior da caixa que contém as resistências; (b) vista interior das resistências de $50\ \Omega$ em paralelo.

Por último, para o acoplamento com a rede elétrica foi utilizada uma bobina de acoplamento mútuo disponibilizada pelo laboratório do GEPE. Assim sendo, na Figura 5.9 está representada a bobina utilizada. Com os dois enrolamentos ligados em série, esta possui um valor de indutância total de cerca de $2,76\ mH$.



Figura 5.9 – Bobina de acoplamento com a rede elétrica.

5.3 Hardware de Controlo

Ao longo desta secção, é analisado o hardware de controlo utilizado no protótipo laboratorial. O sistema de controlo é constituído por sensores de tensão e corrente, responsáveis pela aquisição dos sinais que, posteriormente, são convertidos pelos circuitos de condicionamento de sinal e, de seguida, processados pelo DSC. Após o processamento dos sinais adquiridos, são enviados sinais de PWM para as placas de *driver* dos MOSFET. Para além disso, é realizada uma deteção de erros das grandezas físicas lidas e, visto que se trata de um sistema de controlo modular descentralizado, a comunicação entre os microcontroladores é indispensável. Através dos canais de DAC (*Digital-to-Analog Converter*) disponíveis no microcontrolador é também possível visualizar as variáveis do sistema no osciloscópio.

Como referido anteriormente, é possível isolar os sistemas de potência e controlo. Este isolamento galvânico, aliado a um dimensionamento adequado do sistema de controlo, garante o correto funcionamento, mantendo a segurança durante a realização de testes experimentais e evitando danos irreversíveis ao próprio equipamento. Para isso, todo o hardware de controlo é alimentado pela fonte isolada *TRACO POWER* de referência TXL 060-0533TI. A fonte de alimentação inclui um circuito de proteção contra curtos-circuitos e sobretensões, opera com uma potência nominal de 60 W e fornece níveis de tensão de saída de ± 15 V e +5 V [126]. As alimentações provenientes da fonte são distribuídas para as placas do sistema de controlo através da PCB exposta na Figura 5.10, desenvolvida no GEPE.

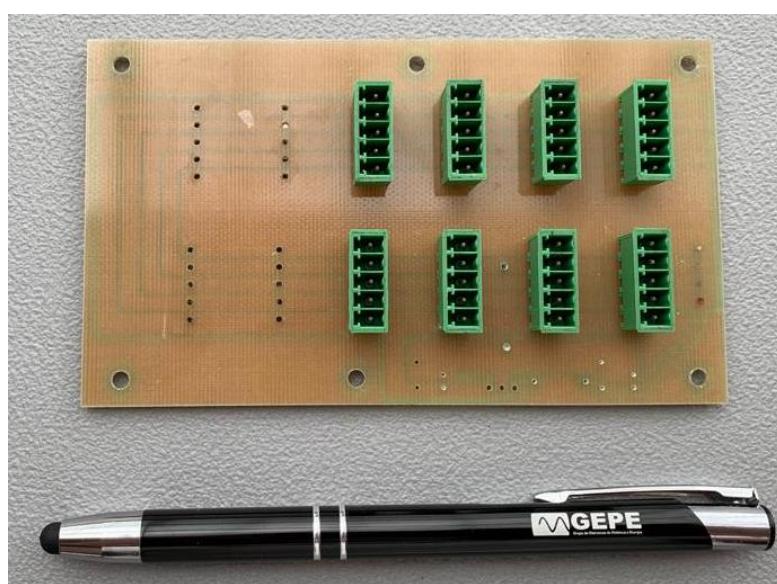


Figura 5.10 – Placa de distribuição das alimentações.

5.3.1 Digital Signal Controller (DSC)

Para a implementação dos algoritmos de controlo digital foi utilizado o microcontrolador TMS320F28379D da *Texas Instruments* incluído na placa de desenvolvimento LAUNCHXL--F28379D, apresentada na Figura 5.11. A sua arquitetura e funcionalidades dedicadas para aplicações de eletrónica de potência foram alguns dos principais motivos que levaram à escolha desta família de microcontroladores. As principais características desta placa de desenvolvimento são [127, 128]:

- 2 CPU (*Central Processing Unit*) de 32-bits e 2 CLAs (*Control Law Accelerators*) programáveis com frequência de relógio de 200 MHz;
- 4 canais independentes de ADC (*Analog-to-Digital Converter*) de 12 bits ou 16 bits, permitindo leitura no modo individual ou diferencial, respetivamente;
- 2 módulos de DAC (*Digital-to-Analog Converter*) de 12 bits;
- 16 canais de PWM com alta resolução (HRPWM – *High-Resolution Pulse Width Modulator*);
- Periféricos de comunicação como: SCI (*Serial Communications Interface*), SPI (*Serial Peripheral Interface*), I2C (*Inter-Integrated Circuit*), CAN (*Controller Area Network*), McBSP (*Multichannel Buffered Serial Port*) e USB (*Universal Serial Bus*).



Figura 5.11 – LAUNCHXL-F28379D da *Texas Instruments*.

O sistema modular descentralizado foi implementado em linguagem de programação C, suportada pela DSC recorrendo ao software Code Composer Studio v9.1.0, disponibilizado pela *Texas Instruments*. Em apêndice encontra-se o código implementado.

5.3.2 Docking Station da LAUNCHXL-F28379D

Para simplificar a implementação do sistema modular descentralizado foi concebida uma placa onde estão incorporados todos os sistemas que fazem conexão direta com a DSC. O principal objetivo passou pela criação de uma estação que inclui a integração da DSC, condicionamento de sinal e deteção de erros, comunicações, circuito de comando e todos os conectores necessários para fazer interface com os restantes elementos do sistema. Esta PCB foi pensada de forma a ser capaz de responder a todos requisitos, para o microcontrolador a funcionar como *master* ou *slave*. As diferenças dos periféricos usados nos dois casos estão expostas na Figura 5.12, em (a) a placa para integração do *master* e em (b) a placa para integração do *slave*. A placa é alimentada com +15 V e -15 V provenientes da fonte *TRACO*, fornecendo localmente 3,3 V de alimentação para o DSC e 5 V para os restantes circuitos.

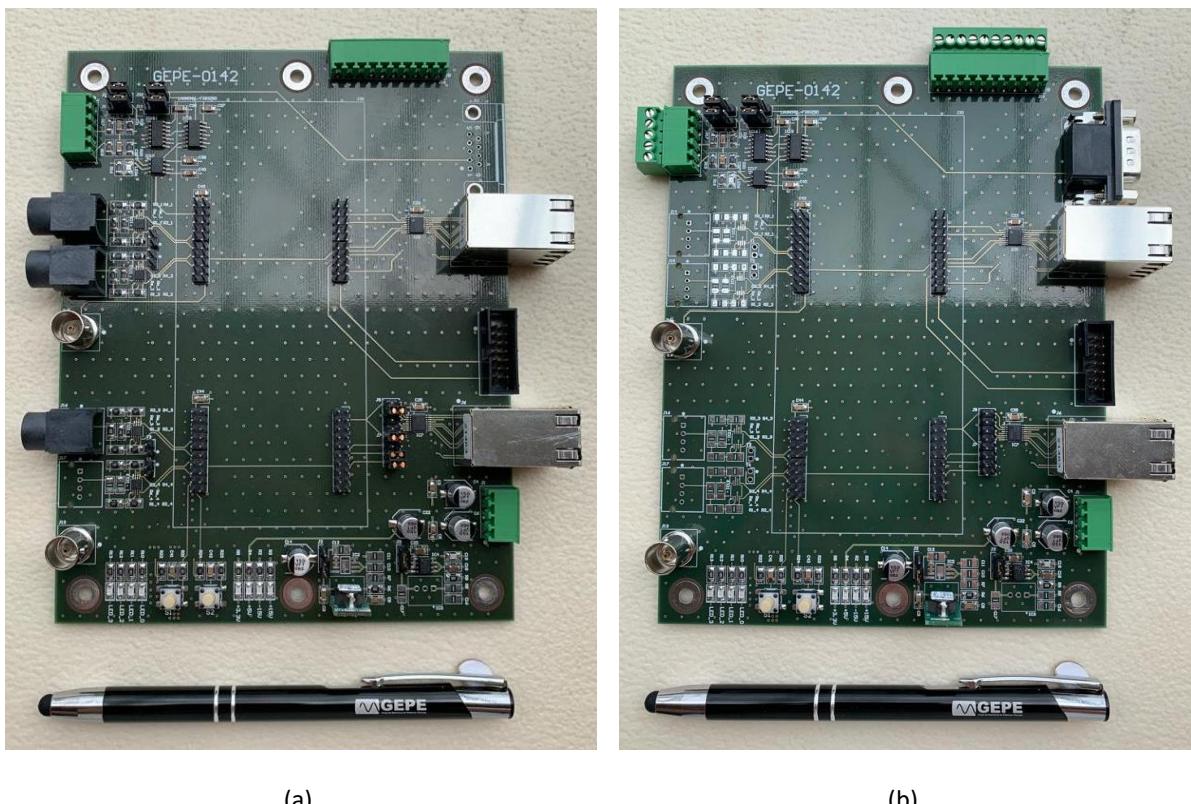


Figura 5.12 – Docking Station da LAUNCHXL-F28379D: (a) integração do *master*; (b) integração do *slave*.

Relativamente à sua construção, existe uma grande distinção entre a *docking station* e as restantes placas desenvolvidas, visto que, esta é composta por quatro camadas. Apesar de elevar o custo e a complexidade do design, o aumento das camadas e o correto empilhamento das mesmas melhoram a performance do sistema. Optou-se por dividir as camadas da seguinte forma: ambas as camadas exteriores (*Top e Bottom Layer*) são camadas de sinal, a

segunda (*Inner Layer 1*) é um plano de massa e a terceira (*Inner Layer 2*) é um plano de potência onde são distribuídas as alimentações. Com este empilhamento, ambas as camadas de sinal estão perto dos planos de massa ou potência, portanto a corrente de retorno pode fluir para os planos adjacentes. O *loop* criado pelo fluxo de corrente é reduzido, minimizando a indutância do caminho de retorno. Dessa forma, o desempenho do sistema com interferências eletromagnéticas e ruído é melhorado [129, 130].

Em cada placa foi projetado o condicionamento de sinal e deteção de erros para quatro ADC, porém estes circuitos só foram montados na placa de integração do *master*. Como os sensores de tensão e corrente são externos à *docking station*, foi necessário colocar quatro conectores mini XLR de 4 pinos. Ou seja, cada ADC possui um conector na borda da placa que fornece as alimentações para o sensor (+15 V, -15 V e GND) e recebe o sinal medido. Além disso, foram usados os canais de DAC do DSC para visualizar o comportamento de determinadas variáveis internas, facilitando a percepção do comportamento do sistema.

5.3.2.1 Condicionamento de Sinal

Uma vez que os sensores utilizados são sensores de efeito de Hall, é fundamental ter um circuito de condicionamento de sinal que converta o valor da corrente, proveniente do sensor, num valor de tensão que seja adequado aos valores do ADC. Para isso, foi escolhido o ADC interno do microcontrolador que aceita valores de tensão entre 0 V e 3 V. Como referido anteriormente, o microcontrolador permite leitura em modo individual ou diferencial. Enquanto o modo individual gera o seu resultado a partir de um sinal de tensão referenciado à massa, o modo diferencial produz o resultado a partir da diferença de dois sinais de tensão, um sinal invertido (v_{in-}) e um não invertido (v_{in+}). Apesar de reduzir o número de ADC disponíveis no microcontrolador, por precisar de dois canais, o modo diferencial tem como grande vantagem a imunidade ao ruído de modo comum, uma vez que, por estar acoplado a ambas as entradas, o ruído é rejeitado.

Na Figura 5.13 está representado o circuito de condicionamento de sinal utilizado e o circuito auxiliar para criar uma referência de tensão constante. A resistência R_m é uma resistência de medida usada para converter a saída de corrente do sensor numa saída de tensão, entre -2,5 V e +2,5 V. Foi escolhido o valor máximo de 2,5 V para garantir uma margem de segurança

em relação ao valor máximo de tensão que o ADC suporta. Neste caso, R_s funciona como uma resistência auxiliar, colocada em série com R_m , para garantir que a resistência de medida está compreendida entre os valores estipulados pelo fabricante dos sensores. O integrado LTC1992 é um amplificador totalmente diferencial que permite converter e amplificar um sinal individual em diferencial [131]. As resistências R_1 , R_2 , R_3 e R_4 foram dimensionadas, todas com valor de $10\text{ k}\Omega$, para o ganho de amplificação ser 1 e as resistências R_5 e R_6 funcionam como divisor resistivo, criando a referência de tensão para o *offset* necessário. Desta forma, a saída é ajustada para uma gama de valores compatíveis com os de leitura do ADC.

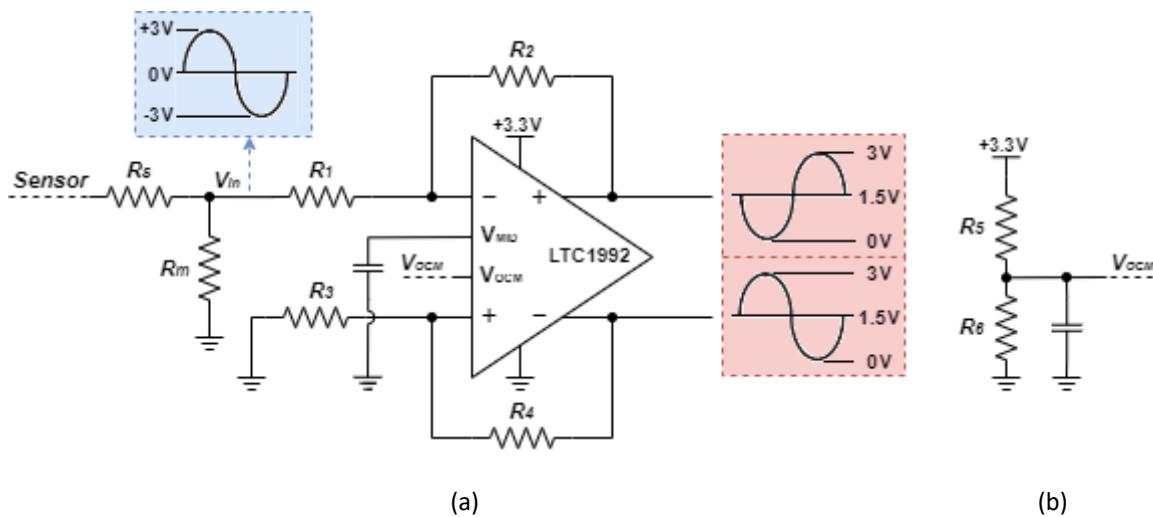


Figura 5.13 – Esquema elétrico: (a) circuito de condicionamento de sinal; (b) circuito auxiliar.

5.3.2.2 Detecção de Erros

O circuito de deteção de erros está representado na Figura 5.14. Este circuito é recomendado por motivos de segurança, garantindo que o sistema apenas opera dentro dos valores nominais estipulados. Para cada ADC são utilizados dois comparadores e dois pares de resistências. As resistências R_{p1} e R_{p2} são dimensionadas para configurar o limite máximo, enquanto R_{n1} e R_{n2} o limite mínimo. Se o sinal do sensor for superior ao limite máximo ou inferior ao limite mínimo, a saída dos comparadores vem a 0, originando um sinal de erro. Esse sinal é recebido tanto pelo DSC como pelo circuito de comando, desativando as comutações por hardware e software. Desta forma, o sistema fica protegido contra estados de operação que podem danificar o conversor.

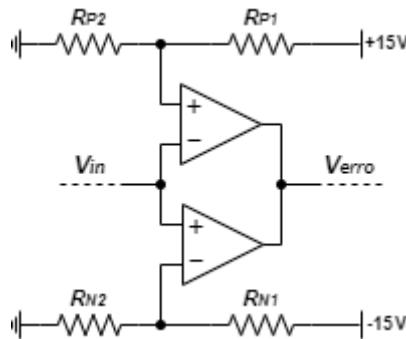


Figura 5.14 – Esquema elétrico do circuito de deteção de erros.

O dimensionamento das resistências é baseado na equação (5.1), onde V_{erro} representa o valor de comparação pretendido, máximo ou mínimo, e V_{in} o valor de alimentação do respetivo braço, +15 V ou -15 V. Como todos os sinais estão compreendidos entre $\pm 2,5$ V, estipulou-se $\pm 2,7$ V para os valores máximos de comparação, admitindo uma margem de erro para pequenas variações. Baseado no resultado da equação, usou-se resistências de 40 k Ω para R_{p1} e R_{n1} e resistências de 10 k Ω para R_{p2} e R_{n2} .

	$V_{erro} = \frac{R_{x2}}{R_{x2} + R_{x1}} \times V_{in}$	(5.1)
--	---	---------

5.3.2.3 Memorização de Erros e Circuito de Comando

O circuito de memorização de erros é mostrado na Figura 5.15(a). O principal componente deste circuito é o integrado NE555 da *Texas Instruments*, que permite memorizar o erro quando está a funcionar no modo biestável [132]. No estado normal de operação do sistema, a saída do NE555 é +5 V. Contudo, quando um erro é detetado, o nível lógico no *Reset* do integrado vem a 0 e, consequentemente, a saída do NE555 é 0 V. Mesmo que o erro deixe de existir, a saída mantém-se com um nível lógico baixo até receber um sinal de *Trigger*.

De salientar que, os amplificadores operacionais que constituem o circuito de deteção de erros são de coletor aberto. Esta particularidade possibilita conectar todas as saídas dos comparadores e permite que um erro seja suficiente para mudar o nível lógico do *Reset*. Desta forma, é apenas necessário utilizar um circuito de memorização.

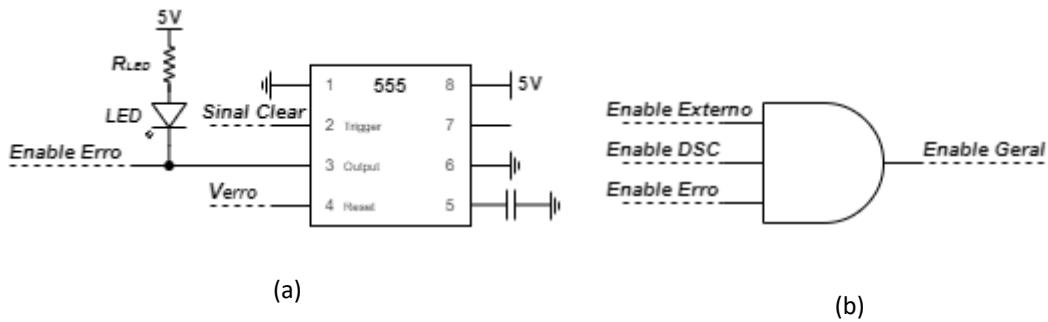


Figura 5.15 – Esquema elétrico do: (a) circuito de memorização de erros; (b) circuito de comando.

Na Figura 5.15(b), está representado o circuito lógico de comando responsável por calcular o sinal que, após ser enviado para a placa de *driver*, habilita ou desabilita os PWM. Este circuito tem como entradas três sinais de *enable* distintos (*Enable Externo*, *Enable DSC* e *Enable Erro*). O *Enable Externo* é um sinal externo que pode ser proveniente do *master*, se for uma placa *slave*, ou de um botão disponível para o utilizador, se for a placa *master*. O *Enable DSC* é um sinal dado pela DSC integrada na *docking station* e é apenas ativado quando todas as configurações iniciais são realizadas. O *Enable Erro* está ligado à saída do circuito de memorização e só é desativado quando existe um erro no sistema. Se algum destes sinais estiver a nível lógico baixo (não ativo) os PWM são desabilitados pelo *driver*.

5.3.2.4 Comunicações

Como referido nas conclusões do Capítulo 2, existe a necessidade de utilizar duas topologias de comunicação, uma em barramento e outra em anel. Para a topologia em barramento escolheu-se usar 13 GPIO (*General Purpose Input/Output*) disponíveis no microcontrolador, onde 12 servem para enviar os dados e o décimo terceiro tem a função de *clock*. A ligação topologia é realizada diretamente entre as placas, sem recorrer a circuitos adicionais, através de uma *flat cable*.

Para a topologia em anel, optou-se por utilizar o protocolo de comunicação SPI. Com o objetivo de melhorar a performance do sistema, foram dimensionados circuitos de sinal diferenciais de baixa tensão (LVDS – *Low Voltage Differential Signaling*) para os 4 canais de SPI (*MOSI – Master Out Slave In*, *MISO – Master In Slave Out*, *SS – Slave Select* e *SCLK – Serial Clock*). O uso de sinais diferenciais apresenta inúmeros benefícios, tais como: inexistência de corrente de retorno, resistência e redução de interferências eletromagnéticas ou *crosstalk*, operação com tensões mais baixas e maior precisão do nível lógico do sinal [133]. Para isso,

foram escolhidos integrados da *Maxim Integrated*, que convertem sinais individuais em diferenciais e vice-versa. Como se pode analisar na Figura 5.16, os integrados MAX9122 e MAX9123 funcionam, respetivamente, como receptor e transmissor, ambos permitindo converter 4 sinais em simultâneo [134, 135]. A ligação entre as placas é feita a partir de cabos Ethernet, uma vez que, estes possuem 4 pares entrançados, tronando-os ideais para a comunicação SPI com sinais diferenciais.

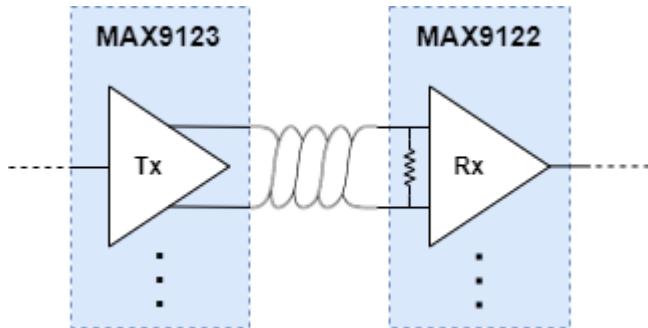


Figura 5.16 – Circuito de aplicação dos integrados MAX9122 e MAX9123.

5.3.2.5 Conectores

Nas Figuras 5.13 (a) e (b) podem observar-se os conectores que compõem as PCB. De destacar que, todos os conectores são *plug in* e foram colocados na periferia da placa, facilitando as conexões e evitando intersecções entre cabos. Além disso, com o propósito de auxiliar na medição de algumas variáveis, incluíram-se pontos de teste em zonas de maior interesse da PCB. Os conectores projetados foram: um conector horizontal de 5 pinos, no canto superior do lado esquerdo, para conectar os sinais de comando entre as diferentes *docking stations*; quatro conectores mini XLR, no lado esquerdo, com a função de enviar a alimentação aos sensores e receber o sinal medido; dois conectores BNC verticais, no lado esquerdo, conectados aos canais de DAC; um conector horizontal de 10 pinos, no canto superior ao centro, com periféricos auxiliares disponíveis caso algum se danifique; quatro conectores *header* de 20 pinos, no centro, para conectar a LAUNCHXL-F28379D; um conector DB9, no canto superior do lado direito, para transmitir os sinais de PWM e o respetivo sinal de habilitação; dois conectores duplos Ethernet, no lado direito, para a comunicação em anel; um conector de 14 pinos, no lado direito, para a flat cable da comunicação em barramento; e um conector horizontal de 4 pinos, no canto inferior do lado direito, para receber as alimentações.

Por fim, da Figura 5.17 é apresentado o resultado da integração da placa *docking station* com um DSC LAUNCHXL-F28379D.

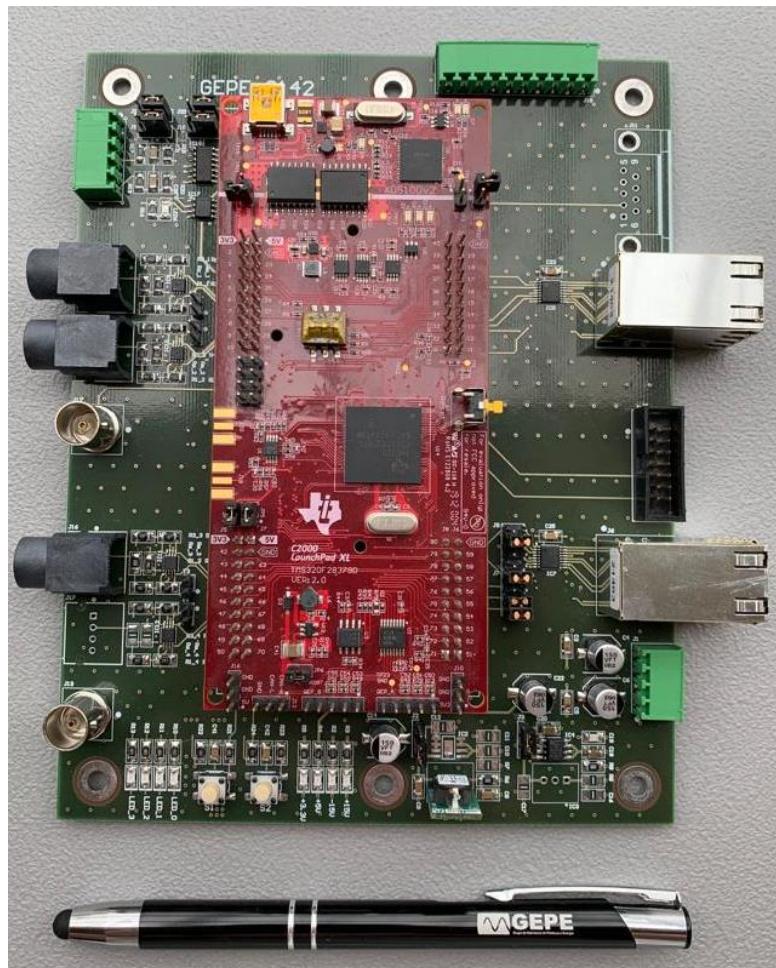


Figura 5.17 – Docking Station com LAUNCHXL-F28379D integrada.

5.3.3 Placa de *Driver*

Os circuitos de *driver* são responsáveis pela interface entre o sistema de controlo e o sistema de potência, podendo permitir isolamento galvânico entre os sistemas. A placa de *driver* foi desenvolvida com o objetivo de adequar os sinais de PWM à gama de tensões de *gate* dos semicondutores.

O *driver* escolhido foi o integrado ADUM3223 da *Analog Devices*. Este suporta frequências de operação até 1 MHz e fornece dois canais isolados independentes, permitindo acionar 2 semicondutores de potência em simultâneo [136]. Dessa forma, em cada placa de *driver* apenas foi usado um ADUM3223. A alimentação isolada do lado secundário é providenciada através das fontes CC-CC isoladas da *Murata Power Solutions* com a referência MEV1D1515SC

[137]. Estas são alimentadas com +15 V, obtendo à saída ± 15 V. Na Figura 5.18 é apresentada a placa de *driver* desenvolvida para o ADUM3223.

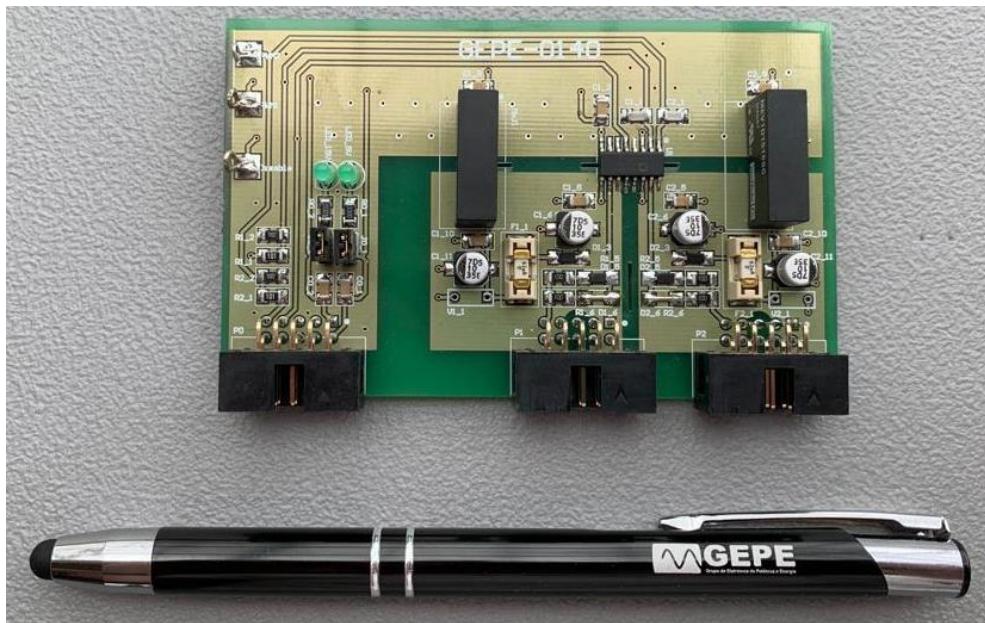


Figura 5.18 – Placa de *driver* desenvolvida.

Relativamente à sua construção, esta placa foi desenvolvida de forma a ser acoplada verticalmente na placa de potência, sendo utilizados três conectores de 10 pinos com um ângulo reto. Adicionalmente, para aumentar o isolamento do lado primário e dos lados secundários, os respetivos componentes foram agrupados e os planos de massa isolados, mantendo o espaçamento definido pelo *drive*. Além disso, foram realizados pequenos recortes na PCB, em pontos críticos como: *driver*, fontes isoladas e entre os lados secundários.

Por fim, foram implementados circuitos de proteção contra curto-circuitos e sobretensões para as fontes de tensão e adicionadas, à saída, duas resistências e dois diodos em paralelo, como representado na Figura 5.19. Esta configuração permite selecionar diferentes resistências de *gate* para ligar ou desligar o semicondutor [138].

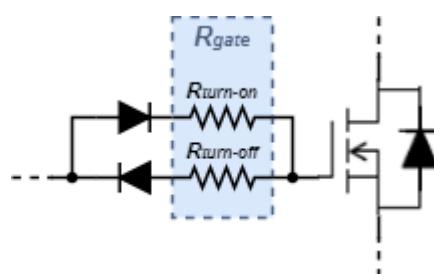


Figura 5.19 – Esquema elétrico do circuito de configuração das resistências de *gate*.

5.3.4 Sensor de Tensão

A medição das tensões é essencial para o correto funcionamento de todo o sistema, nomeadamente no sincronismo com a rede elétrica e no controlo da corrente. Assim, para medir as tensões, foram utilizados os sensores de tensão de efeito de *Hall* disponibilizados pelo GEPE, de referência LV 25-P fabricado pela *LEM USA Inc* [139]. Este sensor apresenta uma relação de transformação de 2500:1000 entre o primário e o secundário; garante isolamento galvânico até 2,5 kV; permite medir uma tensão nominal até 500 V; possui uma corrente nominal no lado primário de 10 mA e uma precisão de $\pm 0,8\%$, com temperatura de 25 °C e tensão de alimentação de ± 15 V. Na Figura 5.20 é apresentado o circuito de ligação recomendado.

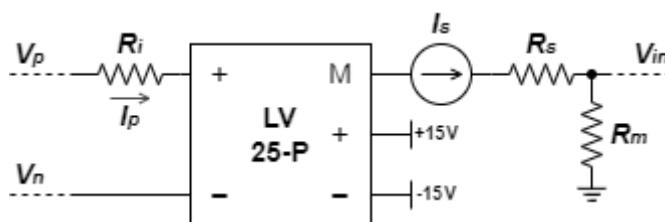


Figura 5.20 – Esquema elétrico do circuito de ligação recomendado para o sensor LV 25-P.

Como referido anteriormente, a saída de um sensor de *Hall* é em corrente, sendo necessário utilizar uma resistência de medida (R_m), para converter este sinal em tensão e, dessa forma, poder ser lido pelo ADC. A resistência R_m foi colocada na placa de condicionamento de sinal e a sua gama de valores devem ser entre 100Ω e 350Ω , valores recomendados pelo fabricante quando o sensor é alimentado com ± 15 V e com uma corrente máxima de 10 mA [139]. A equação (5.2) permite dimensionar a resistência R_m , sendo I_s a corrente de saída de 25 mA e V_{ADC_max} a tensão de leitura máxima do ADC. A corrente de saída foi calculada a partir da multiplicação da corrente de entrada, 10 mA, pela razão de transformação do sensor, 2,5.

$$R_m = \frac{V_{ADC_max}}{I_s} \quad (5.2)$$

A partir do esquema de ligações, também se pode verificar que é necessário utilizar uma resistência de entrada R_i , cuja função é garantir que a corrente do lado primário nunca ultrapasse a corrente nominal. Esta é dimensionada a partir da equação (5.3), onde V_{max} corresponde ao valor de tensão máximo que se pretende medir e I_p ao valor da corrente de entrada do sensor.

$$R_i = \frac{V_{max}}{I_p} \quad (5.3)$$

Os valores das resistências utilizadas para cada sensor de tensão estão representados na Tabela 5.1.

Tabela 5.1 – Valores das resistências de entrada e medida, utilizadas nos sensores de tensão.

Local de Medição	V_{ADC_Max}	$R_m (\Omega)$	$V_{Max} (V)$	$R_i (\Omega)$
Barramento CC	3 V	120 Ω	100 V	10 kΩ
Rede Elétrica	3 V	120 Ω	325 V	30 kΩ (2 x 15 kΩ)

Um exemplo de uma placa de adaptação do sensor LV 25-P é apresentado na Figura 5.21. Foram inseridos condensadores no lado secundário para filtrar a alimentação, um conector verde onde é recebido o sinal a ser medido e um conector laranja onde é realizada a alimentação do sensor e enviado o sinal de saída. Apesar desta placa ser desenvolvida no GEPE, foi necessário dimensionar e soldar as resistências R_i .



Figura 5.21 – Placa de adaptação do sensor de tensão.

Por fim, realizaram-se testes experimentais com o objetivo de analisar a linearidade do sensor e do sinal adquirido pelo DSC. Neste ensaio, variou-se a tensão de entrada do sensor, através de uma fonte de alimentação, sendo adquirido o valor conversão do ADC e o valor da tensão no porto do ADC. O gráfico da Figura 5.22 apresenta os resultados obtidos no teste realizado, sendo utilizado o sensor da rede elétrica.

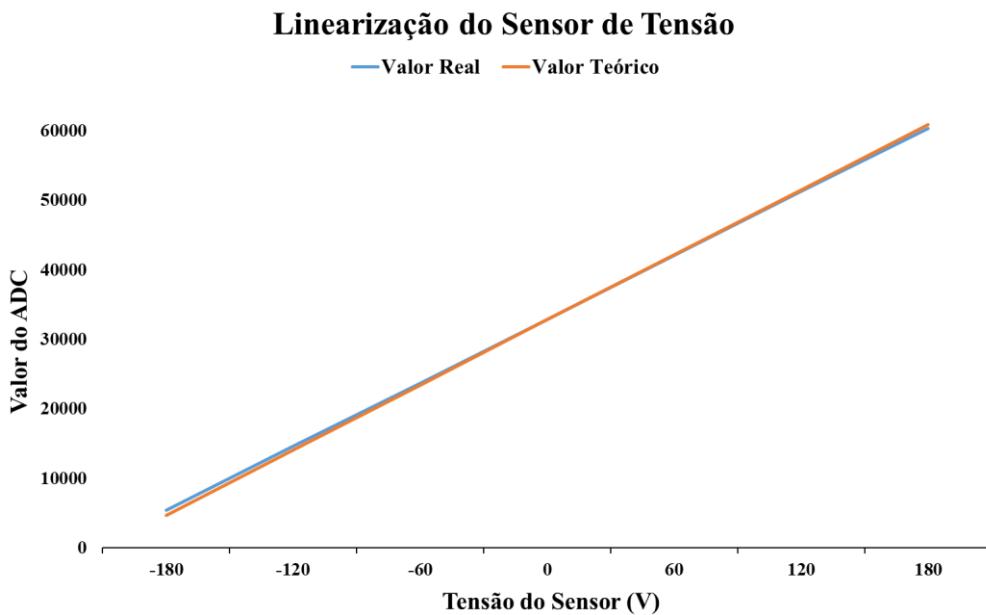


Figura 5.22 – Gráfico da linearidade do sensor de tensão, onde são comparados os valores reais medidos aos valores teóricos calculados.

5.3.5 Sensor de Corrente

A medição da corrente é essencial para o correto funcionamento de todo o sistema, especialmente no controlo da corrente. Assim, para medir a corrente, foi utilizado o sensor de corrente de efeito de Hall disponibilizado pelo GEPE, de referência LA 55-P fabricado pela *LEM USA Inc* [140]. Este sensor apresenta uma relação de transformação de 1:1000 entre o primário e o secundário; garante isolamento galvânico até 2,5 kV; possui uma gama de medição de ± 70 A; permite medir uma corrente nominal no lado primário de 50 A com uma precisão de $\pm 0,65\%$, com temperatura de 25 °C e tensão de alimentação de ± 15 V. Na Figura 5.23 é apresentado o esquema de ligação recomendado.

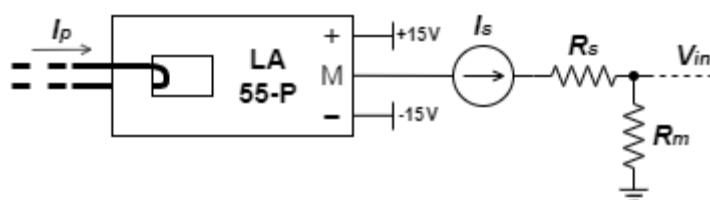


Figura 5.23 – Esquema elétrico do circuito de ligação recomendado para o sensor LA 55-P.

A corrente é medida pelo lado primário, passando um fio condutor pelo orifício do sensor. Este fio é atravessado várias vezes pelo orifício, aumentando a resolução da corrente medida. Ou seja, o valor medido é igual ao valor real multiplicado pelo número de vezes que o fio passa

pelo orifício, aproximando o valor medido ao valor nominal do sensor. Tal como no sensor de tensão, no lado secundário, é necessário utilizar uma resistência de medida (R_m) para converter o sinal de saída em tensão. A resistência R_m foi colocada na placa de condicionamento de sinal e a sua gama de valores dever ser entre $50\ \Omega$ e $160\ \Omega$, valores recomendados pelo fabricante quando o sensor é alimentado com $\pm 15\text{ V}$, a uma temperatura máxima de $70\text{ }^{\circ}\text{C}$ e apresenta uma corrente nominal de 50 A [140].

Tendo isto em consideração e estipulando que o valor eficaz máximo de corrente à saída do conversor é de 10 A , foram utilizadas 3 espiras à volta do sensor. Ou seja, multiplicado o número de espiras pelo valor de pico, aproximadamente 14 A , obtém-se uma gama de medição de $\pm 42\text{ A}$.

Um exemplo de uma placa de adaptação do sensor LA 55-P é apresentado na Figura 5.24. Esta placa foi desenvolvida no GEPE e através do conector laranja é realizada a alimentação do sensor e enviado o sinal medido. Adicionalmente, foram colocados condensadores no lado secundário para filtrar a alimentação.



Figura 5.24 – Placa de adaptação do sensor de corrente.

Por fim, também se realizaram testes experimentais para analisar a linearidade do sensor e do sinal adquirido pelo DSC. Neste ensaio, variou-se a corrente de entrada do sensor, sendo adquirido o valor de conversão do ADC e o valor da tensão no porto do ADC. O gráfico da Figura 5.25 apresenta os resultados obtidos no teste realizado, sendo utilizado o sensor da rede elétrica.

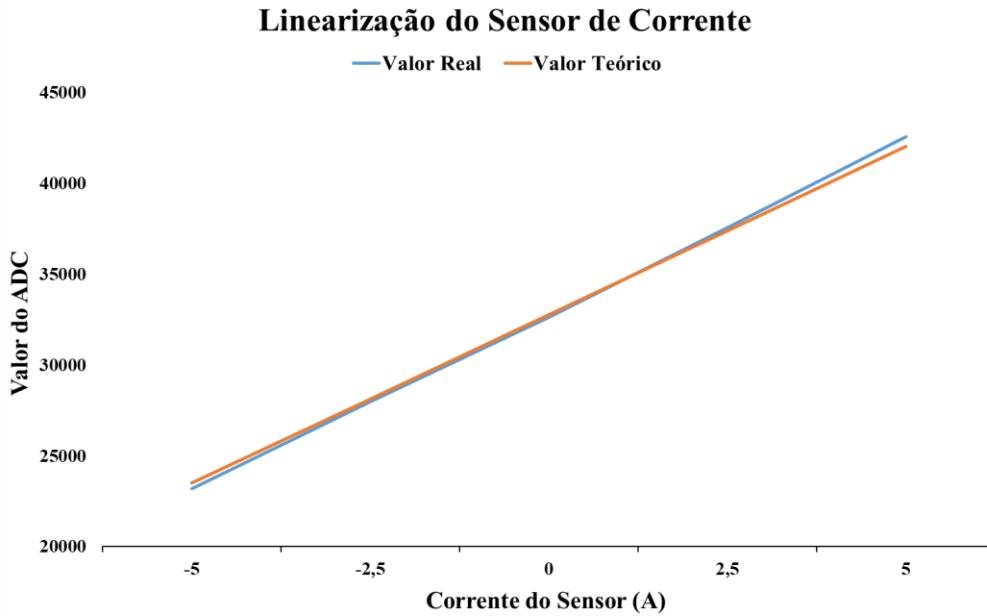


Figura 5.25 – Gráfico da linearidade do sensor de corrente, onde são comparados os valores reais medidos aos valores teóricos calculados.

5.4 Conclusões

Ao longo deste capítulo foi apresentado em detalhe todo o hardware utilizado na implementação do protótipo laboratorial para o sistema proposto. Em termos organizacionais, o capítulo foi dividido em duas partes, mostrando, inicialmente, o hardware que constitui os circuitos de potência e, de seguida, o hardware que integra o sistema de controlo.

Relativamente ao circuito de potência, foram descritos todos os componentes que constituem os conversores de potência CC-CA utilizados nos submódulos, como semicondutores e respetivos circuitos de proteção, dissipadores de calor, condensadores e conectores de potência. De seguida, foi apresentado o circuito de acoplamento com a rede elétrica e proteções, mostrando o esquema elétrico utilizado, os aparelhos e mecanismos de proteção e a bobina responsável por realizar o acoplamento com a rede elétrica.

Após a análise do andar de potência, foram descritos todos os elementos necessários ao controlo do sistema proposto. Inicialmente, é apresentada a fonte de alimentação isolada e a placa de distribuição das alimentações, responsáveis por alimentar todo o sistema de controlo. De seguida, foi mostrado o microcontrolador utilizado, TMS320F28379D, para a implementação dos algoritmos de controlo, e as razões que levaram para a escolha do mesmo.

Além disso, foi desenvolvida uma placa que permite fazer interface entre o microcontrolador e os restantes constituintes do sistema, denominada de *docking station*. Posteriormente, foi apresentada a placa de *driver* responsável pela interface entre os sistemas de controlo e potência, adequando os sinais de PWM à gama de tensões de *gate* dos semicondutores. Por fim, foram descritas as placas dos sensores de tensão e de corrente utilizadas para realizar a leitura das variáveis do sistema.

De realçar que, os componentes existentes no GEPE foram sempre considerados e utilizados quando assim o justificaram, devido à facilidade de aquisição e custo associado. Adicionalmente, todo o hardware foi utilizado com o objetivo principal de validar experimentalmente a topologia utilizada, contudo foi desenvolvido de forma a permitir o seu reaproveitamento para projetos futuros. Consequentemente, todas as PCB foram testadas individualmente, sendo aferido o seu correto funcionamento.

Capítulo 6

Princípio de Funcionamento do Protótipo Laboratorial e Resultados Experimentais

6.1 Introdução

Neste capítulo é explicado o princípio de funcionamento do controlo implementado, particularmente, as tramas usadas para as configurações do sistema, e são apresentados os resultados obtidos nos ensaios experimentais do protótipo laboratorial. Os algoritmos elaborados para o controlo do sistema são baseados nos diagramas das duas máquinas de estados, uma usada para o *master* e outra para os *slaves*. Dessa forma, são descritas, de uma maneira geral, as funcionalidades de cada estado e é explicada a progressão entre os eles, visto que, apesar de distintos, os diagramas são dependentes.

Relativamente aos ensaios experimentais, inicialmente são expostos os resultados para o sincronismo com a rede elétrica, utilizando o algoritmo de EPLL. Posteriormente, os resultados estão divididos em duas fases: a primeira fase visa a validação do sistema com três conversores de eletrónica de potência ligados em cascata, enquanto a segunda fase retrata os resultados obtidos para o sistema com dois conversores de eletrónica de potência ligados em cascata. Assim sendo, em ambas as fases, focou-se inicialmente nos testes às topologias de comunicação de forma a validar corretamente as tramas de configuração. Depois, foram validados os sinais de PWM de cada braço e testados os algoritmos de corrente com uma carga resistiva de modo a perceber o comportamento do conversor. Finalmente, procedeu-se à injeção de energia na rede elétrica, sendo apresentados os resultados adquiridos.

De referir que, de forma a manter a segurança e a integridade do sistema, todos os constituintes, como placas desenvolvidas, configurações de periféricos e algoritmos concebidos, foram testados individualmente e os valores de tensão e corrente utilizados foram sempre aumentados de forma gradual até aos valores apresentados nesta secção.

6.2 Princípio de Funcionamento do Protótipo Laboratorial

Para uma melhor compreensão do trabalho realizado e dos resultados experimentais obtidos, é necessário explicar o princípio de funcionamento do protótipo laboratorial implementado. Dessa forma, são apresentados na Figura 6.1 os diagramas das máquinas de estados que serviram como base para o funcionamento do sistema e para a implementação dos algoritmos, tanto no *master* como nos *slaves*. Na Figura 6.1(a) é mostrado o diagrama da máquina de estados do microcontrolador *master* e na Figura 6.1(b) o diagrama dos microcontroladores *slaves*. De notar que, tal como referido anteriormente, todos os *slaves* possuem os mesmos algoritmos, tornando o sistema o mais modular e autónomo possível.

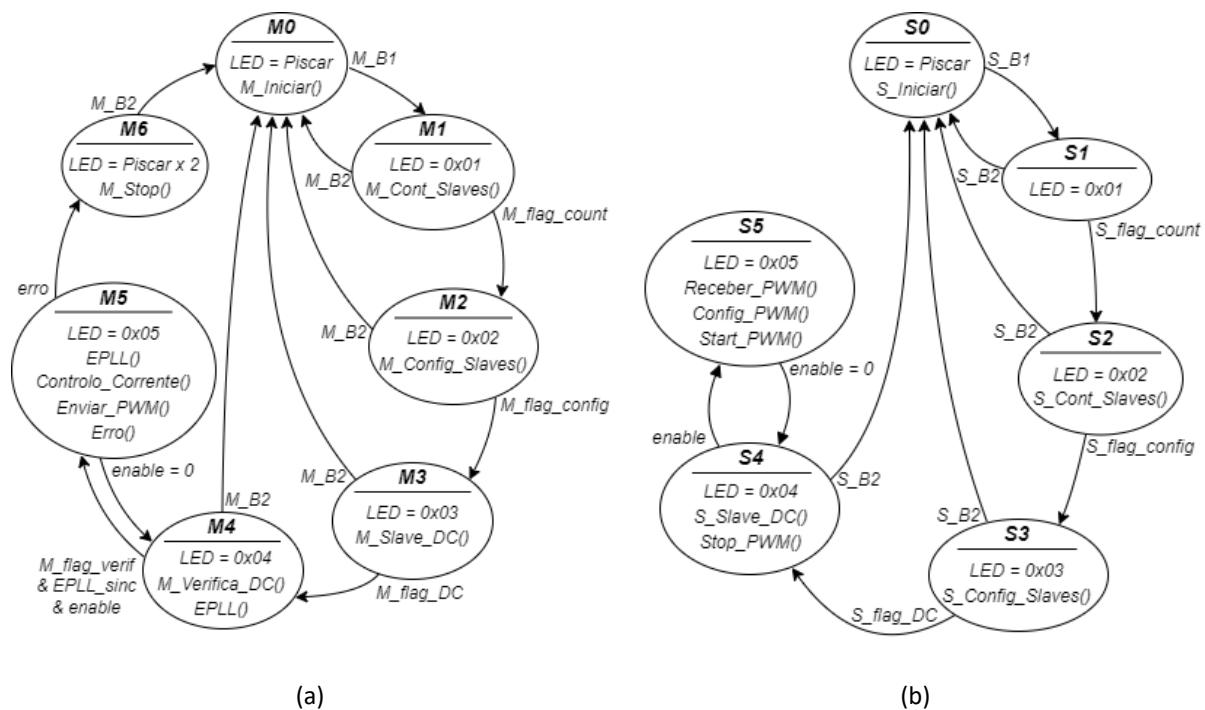


Figura 6.1 – Diagramas das máquinas de estados implementadas para o controlo de: (a) *master*; (b) *slaves*.

De uma forma geral, o *master* começa por um estado inicial, *M0*, onde o microcontrolador aguarda por um sinal dado pelo utilizador quando todos os submódulos estão ligados. Os próximos 4 estados, de *M1* a *M4*, são estados de configuração, nos quais são trocadas as tramas de comunicação que permitem identificar os *slaves* ligados, calcular e atribuir os desfasamentos das portadoras, verificar os barramentos CC e fazer a sincronização com a rede elétrica. O estado seguinte, *M5*, é o estado de ligação do conversor, no qual é ativado o controlo de corrente e enviado o valor da onda moduladora. Por fim, existe um estado de

erro, $M6$, que interrompe todo o sistema e espera por um sinal proveniente do utilizador para regressar ao estado inicial.

No diagrama dos slaves a divisão dos estados é idêntica, começando pelo estado inicial $S0$, que aguarda o sinal do utilizador. Os próximos quatro estados, de $S1$ a $S4$, também são de configuração, tendo as mesmas funções que os estados de configuração do master. Por fim, o estado $S5$ é o responsável por ativar as comunicações e enviar o PWM para a placa de *driver*.

Como é possível observar, apesar de diferentes, os diagramas possuem algumas similaridades, isto é, ambos têm um estado inicial de espera, um conjunto de estados de configuração e um estado de ligação do conversor. Além disso, eles são dependentes, pois a progressão entre os estados de um microcontrolador é determinada pelos restantes, contudo isto será explicado mais detalhadamente nos parágrafos seguintes.

Como referido anteriormente, ambos os diagramas começam por um estado inicial, $M0$ e $S0$. Após o sinal do utilizador, proveniente de um botão mecânico $B1$ (botão existente nas placas docking station para o master e slaves), os dois diagramas avançam para o estado 1, $M1$ e $S1$. A partir daqui todo o processo de configuração é realizado de forma automática através das tramas da comunicação em anel, cuja estrutura é apresentada na Tabela 6.1. Além disso, o processo de configuração pode ser dividido em quatro estágios: contagem, configuração, recolha e verificação/sincronização.

Tabela 6.1 – Estrutura das tramas de comunicação.

Endereço	Função	Dados	Stop
8 bits	8 bits	$N \times 8$ bits	8 bits

No estágio de contagem, o *master* envia a primeira trama de comunicação que serve para contar o número de submódulos ligados. O *slave* ligado ao *master*, estando no estado $S1$, aguarda pela receção da primeira trama. Assim que a recebe, a flag S_flag_count é ativa e o microcontrolador avança para o estado $S2$. Neste estado, os dados da trama são incrementados com o número correspondente (neste caso número 1) e a trama é enviada para o próximo *slave*, repetindo o processo até chegar novamente ao *master*. Ao receber a trama de contagem, a flag M_flag_count é ativa e o *master* avança para o estado $M2$, iniciando o estágio de configuração.

No estágio de *configuração*, o *master* calcula os desfasamentos das portadoras e envia a segunda trama preenchida com os respetivos valores. Ao receber a trama, *S_flag_config* é ativa e o *slave* avança para o estado *S3*. Neste estado, o microcontrolador lê o valor do desfasamento escrito na posição correspondente e envia a trama para o próximo *slave* que repete o processo até chegar novamente ao *master*. Ao receber a trama de configuração, *M_flag_config* é ativa e o *master* avança para o estado *M3*, iniciando o estágio de recolha.

No estágio de recolha, o *master* envia uma trama vazia, ou seja, os dados são preenchidos com o valor 0. Ao receber a trama, *S_flag_DC* é ativa e o *slave* avança para o estado *S4*. Neste estado, o microcontrolador escreve, na posição correspondente, o valor da tensão do barramento no seu submódulo e envia a trama para o próximo *slave*, repetindo o processo até chegar novamente ao *master*. Ao receber a trama de recolha, *M_flag_DC* é ativa e o *master* avança para o estado *M4*, iniciando o estágio de verificação/sincronização.

No estágio verificação/sincronização, o *master* verifica os valores dos barramentos recebidos e começa a sincronização com a rede elétrica. Assim que estas tarefas são concluídas o processo de configuração termina. De notar que, tal como referido anteriormente, não se vai realizar a regulação dos barramentos CC dos submódulos, porém esta funcionalidade foi adicionada para trabalhos futuros.

Após a configuração, o sistema fica em espera até receber o sinal de *enable* do utilizador. Quando o sinal de *enable* é ativo, os barramentos estão corretos (*M_flag_verif* = 1) e a sincronização é realizada (*EPLL_sinc* = 1), tanto o *master* como os *slaves* passam para o estado seguinte, *M5* e *S5*, respetivamente. Neste estado, o *master* continua o algoritmo de EPLL, inicia o controlo de corrente e começa a enviar o valor da onda moduladora através da comunicação em barramento. O *slave* recebe esse valor, ativas as comutações dos semicondutores e envia os sinais de PWM gerados para as placas de *driver*. Os microcontroladores apenas saem destes estados se o sinal de *enable* do utilizador é desligado ou se ocorre um erro no sistema. No primeiro caso, ambos regressam para o estado anterior *M4* e *S4*, desativando as comutações dos semicondutores e parando a comunicação em barramento, podendo avançar novamente com a ativação do *enable*. No segundo caso, o *master* passa para o estado *M6* e o *slave* regressa ao estado *S4*, desativando as comutações e parando o sistema. Após a paragem, o *master* aguarda por um sinal do utilizador *B2* (segundo

botão existente nas placas *docking station*) que inicia todo o sistema, colocando tanto o *master* como os *slaves* nos estados iniciais *M0* e *S0*. Realçar que, durante todo o processo de configuração, este pode ser interrompido pelo utilizador através do botão *B2* existente na *docking station* do *master* ou dos *slaves*, *M_B2* e *S_B2*.

6.3 Resultados Experimentais

Após a análise do princípio de funcionamento do protótipo laboratorial são apresentados os resultados experimentais do mesmo, nomeadamente, dos algoritmos de sincronização com a rede elétrica, do protótipo laboratorial com três submódulos ligados em cascata e do protótipo laboratorial com dois sumódulos. Além dos resultados apresentados nas secções seguintes, todas as placas desenvolvidas foram testadas individualmente, comprovando o seu correto funcionamento.

6.3.1 Algoritmo de Sincronização com a Rede Elétrica

Como referido nos capítulos anteriores, é fundamental gerar um sinal sincronizado com a rede elétrica para o correto funcionamento do controlo implementado, principalmente, para a injeção de energia na rede elétrica. Dessa forma, as ondas de referência utilizadas pelo controlador não são afetadas pelo conteúdo harmônico presente nas tensões da rede elétrica, adquirindo apenas a frequência e a fase da componente fundamental. Assim sendo, baseado no algoritmo usado nas simulações computacionais, é apresentado o sincronismo do sinal da EPLL com a rede elétrica na Figura 6.2.

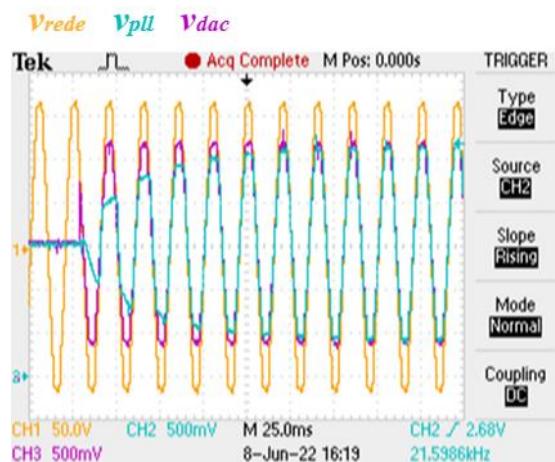


Figura 6.2 – Resultado experimental do sincronismo com a rede elétrica.

A primeira forma de onda apresentada, v_{rede} , representa a tensão da rede elétrica à saída do transformador com valor eficaz de 115 V (163 V de pico). Adicionalmente, as formas de onda v_{pll} e v_{dac} correspondem aos sinais gerados pelo DAC, estando compreendidos entre 0 V e +3 V, com valor médio de 1,5 V. Estes resultam, respetivamente, do algoritmo de EPLL e da própria tensão aos terminais do transformador.

Como é possível observar, quando o algoritmo é iniciado, aproximadamente aos 25 ms, a frequência e a fase da componente fundamental são adquiridas rapidamente, mas a amplitude é ajustada gradualmente, demorando cerca de seis ciclos da rede elétrica. De notar que, foram apresentados os resultados gerados pelo DAC para colocar ambas formas de onda na mesma ordem de grandeza, permitindo visualizar o tempo de ajuste da amplitude. Após esta análise, é possível concluir que o algoritmo da EPLL funciona corretamente, visto que é capaz de gerar um sinal de referência sinusoidal em fase com a componente fundamental da tensão da rede. Além disso, como o conversor só inicia a injeção de energia na rede elétrica após as configurações iniciais, o tempo de seis ciclos para a sincronização com a rede elétrica é irrelevante. Na Figura 6.3 é apresentado o sincronismo do sinal da EPLL com a rede elétrica após os seis ciclos necessários para entrar em regime permanente.

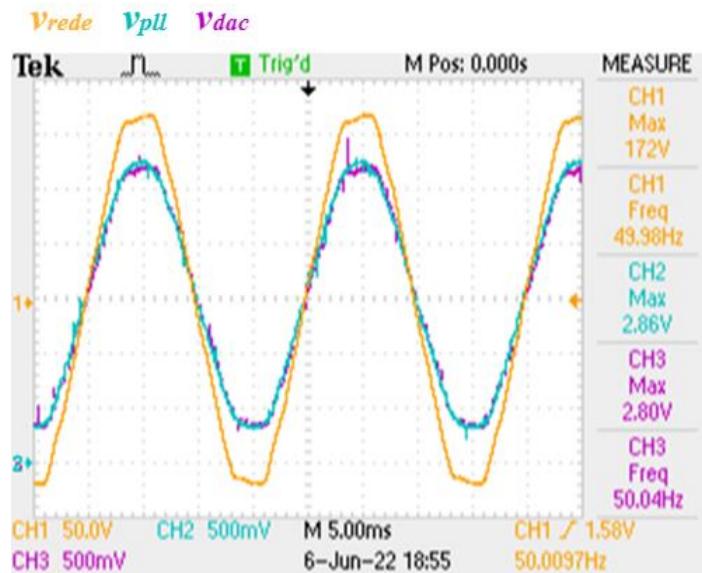


Figura 6.3 – Resultado experimental do sincronismo com a rede elétrica em regime permanente.

6.3.2 Conversor com Três Submódulos

Numa primeira fase, foram realizados testes experimentais à topologia utilizada, nomeadamente, ao sistema de controlo modular descentralizado para três conversores de eletrónica de potência ligados em cascata. Para tal, começou-se por testar e validar os algoritmos da comunicação em anel, medindo os sinais elétricos e utilizando a ferramenta de *debug* do microcontrolador a funcionar como *master*. Na Figura 6.4 é mostrado o painel que permite visualizar o valor das expressões durante o *debug* no primeiro estágio de configuração. É possível observar que o *master* envia uma trama inicial (*send_array*) com apenas três valores. Isto acontece porque no estágio de contagem a trama é enviada sem dados, com um valor para o endereço, outro para a função e o último de stop. No final, é recebida a trama completa (*receive_array*) com o número dos respetivos submódulos ligados, comprovando o correto funcionamento do estágio de contagem.

Expression	Type	Value	Address
<code>(*)= control_state</code>	unsigned int	1	0x0000D080@...
<code>(*)= n_slaves</code>	unsigned int	3	0x0000D08C@...
<code>(*)= flag_receive_counter</code>	unsigned int	1	0x0000D086@...
<code>(*)= flag_receive_config</code>	unsigned int	0	0x0000D087@...
<code>(*)= flag_dc_bus</code>	unsigned int	0	0x0000D088@...
<code>(*)= send_array</code>	unsigned int[3]	[0,1,36]	0x0000D038@...
<code>(*)[0]</code>	unsigned int	0	0x0000D038@...
<code>(*)[1]</code>	unsigned int	1	0x0000D039@...
<code>(*)[2]</code>	unsigned int	36	0x0000D03A@...
<code>(*)= receive_array</code>	unsigned int[50]	[0,1,2,3,...]	0x0000D000@...
<code>(*)[0]</code>	unsigned int	0	0x0000D000@...
<code>(*)[1]</code>	unsigned int	1	0x0000D001@...
<code>(*)[2]</code>	unsigned int	1	0x0000D002@...
<code>(*)[3]</code>	unsigned int	2	0x0000D003@...
<code>(*)[4]</code>	unsigned int	3	0x0000D004@...
<code>(*)[5]</code>	unsigned int	36	0x0000D005@...
<code>(*)[6]</code>	unsigned int	0	0x0000D006@...
<code>(*)[7]</code>	unsigned int	0	0x0000D007@...

Figura 6.4 – Valor das expressões do master durante o estágio de contagem.

Importante referir a utilidade das restantes variáveis apresentadas: *control_state* indica o estado de operação referente à máquina de estados; *n_slaves* guarda o número total de submódulos ligados; *flag_receive_counter*, *flag_receive_config* e *flag_dc_bus* são as *flags* que permitem o avanço dos estados. Além disso, as tramas são preenchidas da seguinte forma:

- No endereço é sempre utilizado o valor 0 que significa *broadcast*, ou seja, é uma informação direcionada a todos os *slaves*. Apesar de não terem sido utilizadas tramas com endereços diferentes optou-se por esta estrutura para possibilitar outras funcionalidades em trabalhos futuros.

- Na função são utilizados três valores diferentes dependendo da função da trama: 1 corresponde à função de contar os submódulos, 2 à função de configurar os desfasamentos e 3 à função de verificar os barramentos.
- No stop é utilizado o carácter “\$” que em decimal corresponde ao valor 36.

Na Figura 6.5 são apresentados os sinais elétricos à saída da comunicação em anel de cada microcontrolador (com_{p0} saída no *master*, com_{p1} saída no primeiro *slave*, com_{p2} saída no segundo *slave* e com_{p3} saída no terceiro *slave*). Como esperado, o tamanho da trama vai aumentando à medida que é recebida pelos *slaves*, começando com dois pulsos que correspondem à função e ao stop (o endereço tem valor 0) e terminando com cinco pulsos da adição dos três submódulos. O tempo total do estágio de contagem é de aproximadamente 500 μ s.

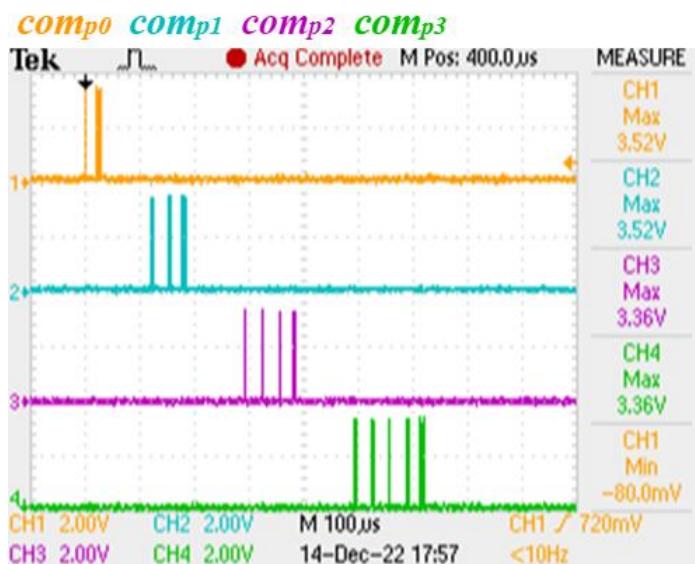


Figura 6.5 – Resultado experimental das comunicações em anel durante o estágio de contagem.

O painel com o valor das expressões do segundo estágio de configuração é mostrado na Figura 6.6. É possível observar que a trama enviada pelo master (*send_array*) é a mesma que a recebida (*receive_array*), sendo apenas de lida pelos *slaves*. Uma vez que existem seis braços no sistema o desfasamento entre as ondas portadoras é de 60°, obtendo os seguintes valores: {0, 60, 120, 180, 240, 300}. Com o intuito de diminuir o tamanho das tramas e, consequentemente, o tempo de configuração, optou-se por enviar apenas o valor do desfasamento do primeiro baço de cada submódulo, sendo o segundo calculado no *slave*.

Desta forma, foram validados dois mecanismos de cálculo dos desfasamentos, um feito no *master* e enviado para os restantes *slaves* e outro feito localmente no próprio *slave*.

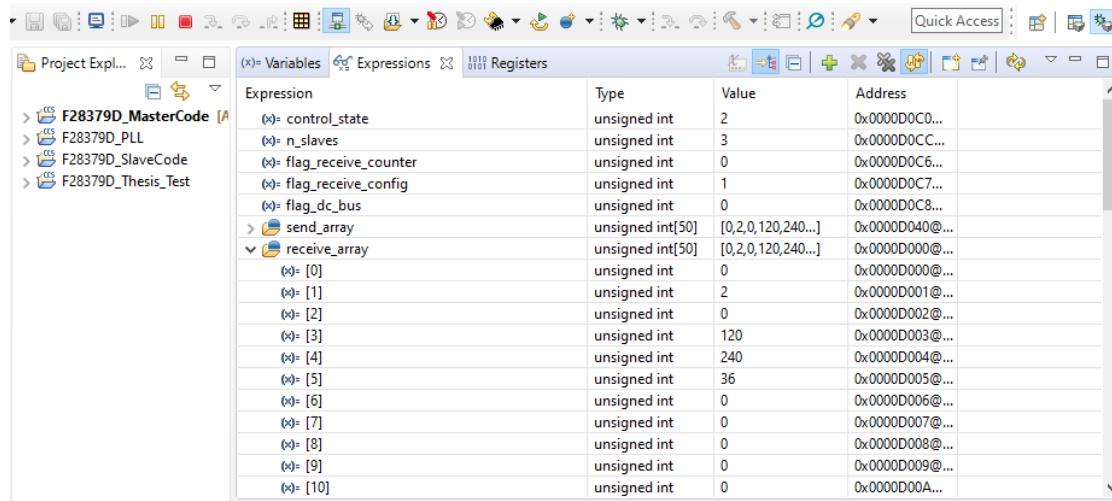


Figura 6.6 – Valor das expressões do master durante o estágio de configuração.

Na Figura 6.7 são apresentados os sinais elétricos à saída da comunicação em anel de cada microcontrolador. Como esperado, à medida que a trama é recebida pelos *slaves* o seu tamanho mantém-se o mesmo. O tempo total do estágio de configuração é de aproximadamente 700 µs.

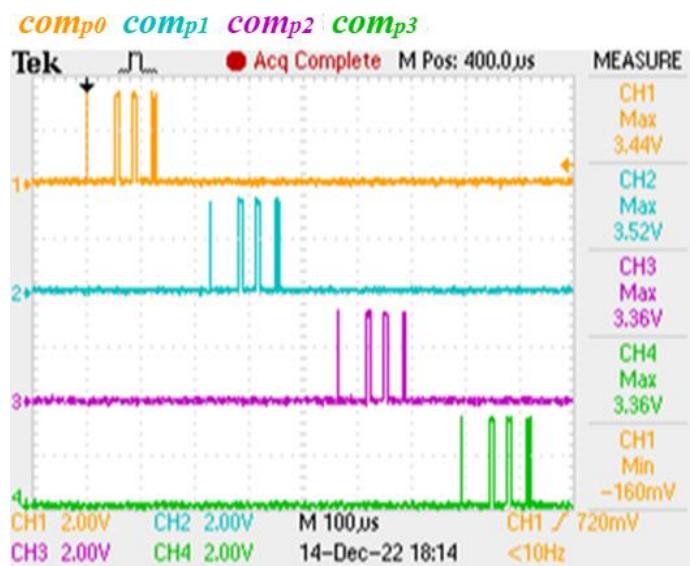


Figura 6.7 – Resultado experimental das comunicações em anel durante o estágio de configuração.

O painel com o valor das expressões do terceiro estágio de configuração é mostrado na Figura 6.8. É possível observar que a trama enviada pelo master (*send_array*) tem o mesmo tamanho que a recebida (*receive_array*), mas os dados são alterados. Isto acontece porque

no estágio de recolha cada *slave* preenche a respetiva posição da trama com o valor do barramento CC do seu submódulo. Uma vez que a regulação dos barramentos não é realizada, estes não são medidos para economizar no número sensores utilizados e os dados da trama são preenchidos com o valor 60, permitindo validar os algoritmos. Este valor é proveniente de uma variável interna previamente atribuída aos *slaves*.

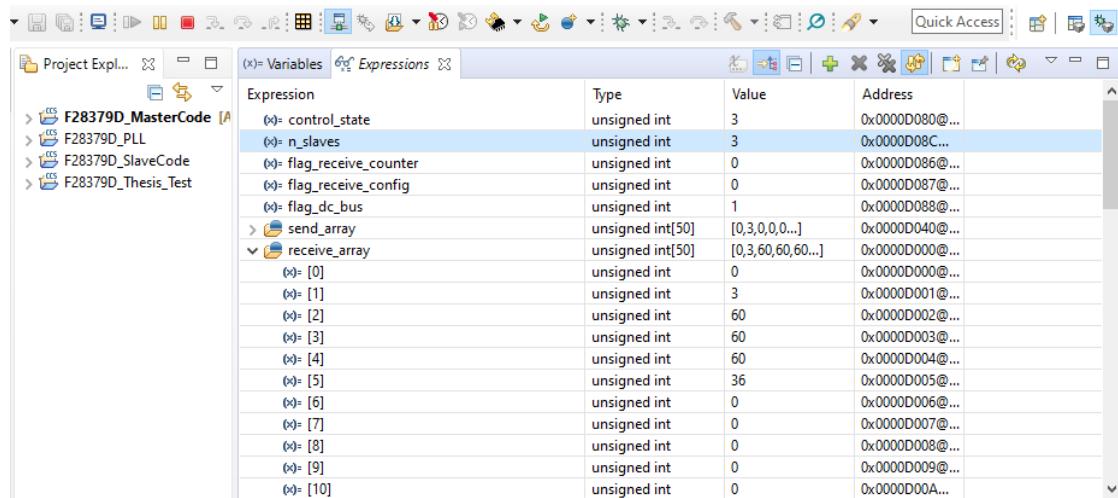


Figura 6.8 – Valor das expressões do *master* durante o estágio de recolha.

Na Figura 6.9 são apresentados os sinais elétricos à saída da comunicação em anel de cada microcontrolador. Como esperado, a trama vai sendo preenchida à medida que é recebida pelos *slaves*, sem alterar o tamanho total. O tempo do estágio de recolha é de aproximadamente 670 µs.

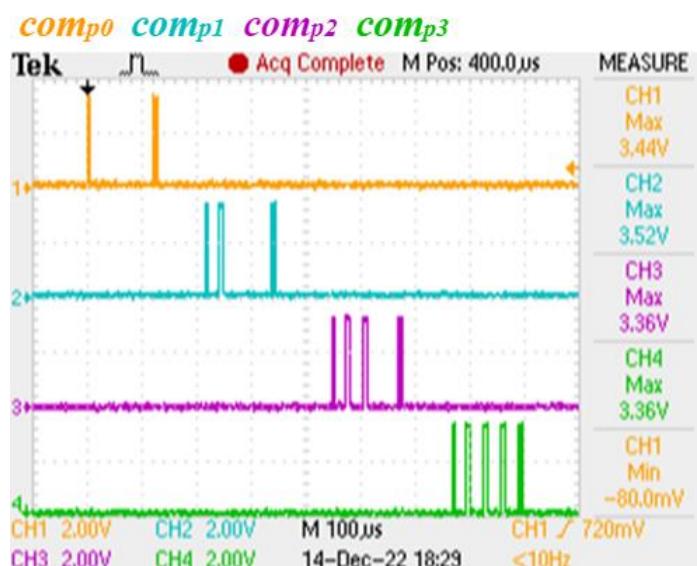


Figura 6.9 – Resultado experimental das comunicações em anel durante o estágio de configuração.

Além da topologia em anel, na Figura 6.10 são exibidos os resultados experimentais para a topologia em barramento. Em (a) foi medido um dos GPIO da comunicação em barramento à saída do *master* (com_{b0}) e à entrada dos três *slaves* (com_{b1} , com_{b2} , com_{b3}). Desta forma, pretendeu-se avaliar o tempo de transmissão da comunicação, comprovando que o atraso é praticamente inexistente. Em (b) são apresentados dois sinais, o primeiro ($gpio_m$) corresponde a uma saída do *master* e o segundo ($gpio_s$) a uma saída de um *slave*. Neste teste, foi ligado um dos GPIO da comunicação em barramento e mediou-se o tempo que o *slave* demora a ler essa informação, ativando um pino de saída. Apesar do tempo de transmissão ser praticamente inexistente, foi possível observar que o tempo de leitura é de aproximadamente de 240 ns, sendo um atraso irrelevante para a aplicação em questão.

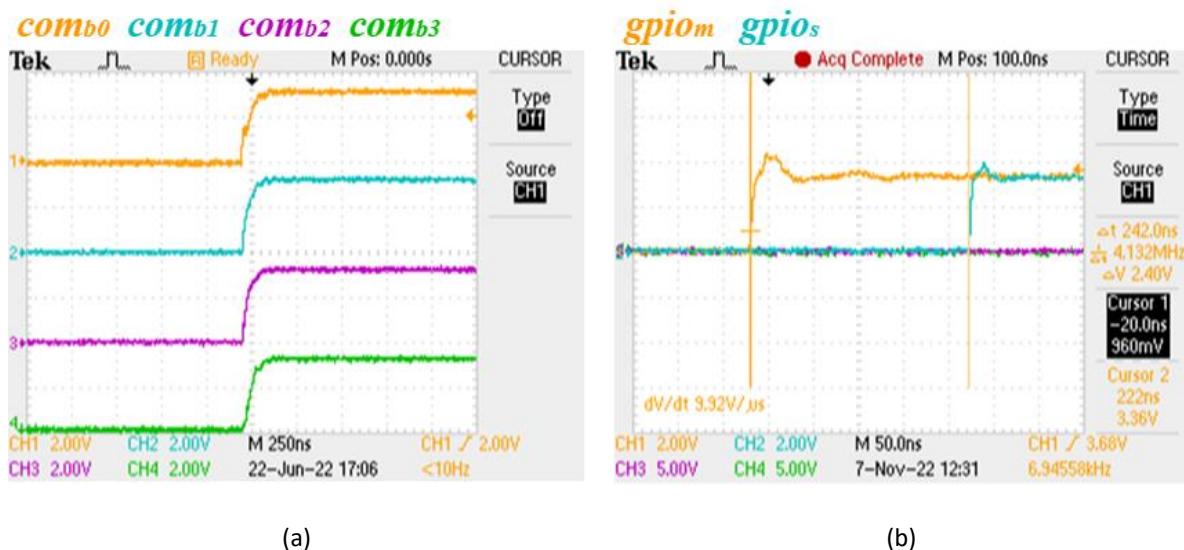


Figura 6.10 – Resultados experimentais da comunicação em barramento: (a) tempo de transmissão; (b) tempo de leitura.

Após testar e validar as comunicações, verificaram-se os sinais de PWM, nomeadamente, os desfasamentos e a saída das placas de *driver*. Na Figura 6.11 são apresentados os sinais de PWM em cada braço do sistema, com um *duty cycle* de 10% e uma frequência de 10 kHz. Em (a) são mostrados os sinais do primeiro ao quarto braço onde é possível observar o desfasamento de 60°, cerca de 16,7 µs para um período de 100 µs. Em (b) os dois primeiros sinais são novamente os PWM dos dois primeiros braços, porém os outros dois são os sinais do quinto e sexto braço, desfasados 240° (66,7 µs) e 300° (83,3 µs) do primeiro. Dessa forma, é comprovado o correto funcionamento do sistema.

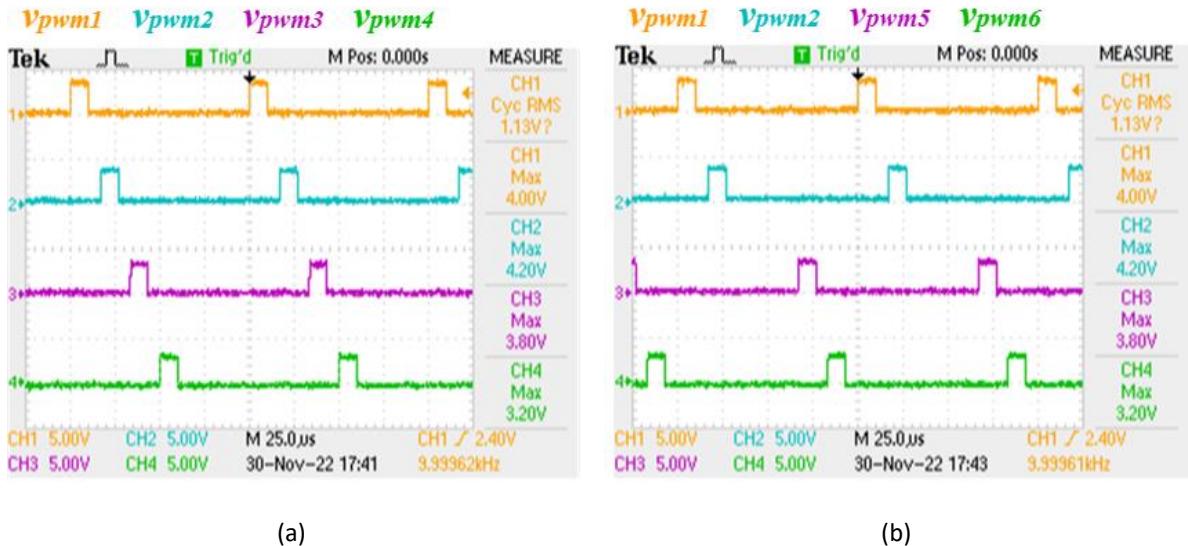


Figura 6.11 – Resultados experimentais do desfasamento dos sinais de PWM: (a) para os primeiros quatro braços; (b) para os primeiro, segundo, quinto e sexto braços.

Na Figura 6.12 são mostrados os sinais de PWM para os semicondutores do primeiro baço, sinal e o seu complementar, sendo v_{pwm1} e $v_{pwm_{c1}}$ medidos à entrada da placa de driver e v_{gpwm1} e $v_{gpwm_{c1}}$ medidos à saída (tensão aplicada na gate do MOSFET). O comportamento da placa é o previsto, a fase e frequência do sinal mantêm-se, mas a amplitude aumenta para os valores adequados à gate dos semicondutores utilizados.

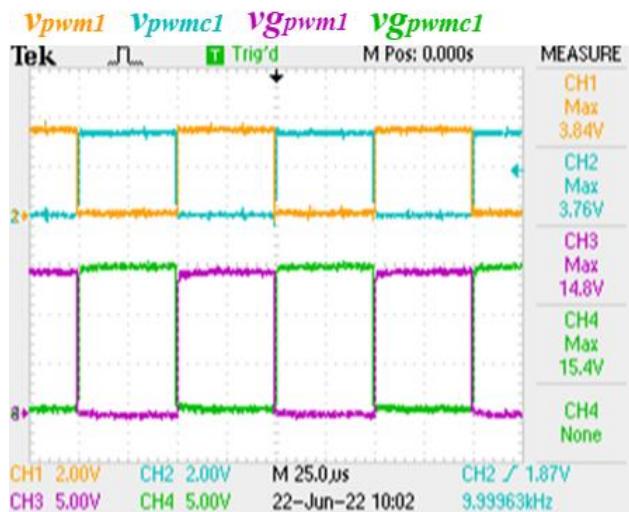


Figura 6.12 – Resultados experimentais da placa de driver.

Depois de validar os protocolos de comunicação e sinais de PWM, foram realizados os ensaios experimentais à totalidade do sistema proposto. Apesar do protótipo laboratorial estar sobredimensionado, optou-se por reduzir a potência máxima total do sistema para 300 VA, mantendo os 4 A para a amplitude máxima da corrente de saída e diminuindo a tensão dos

barramentos para 35 V. O conversor foi conectado à rede elétrica com um valor eficaz da tensão de 50 V com recurso a um transformador.

Antes dos testes com o sistema conectado à rede elétrica realizaram-se ensaios com uma carga resistiva de 53Ω e a bobina de acoplamento de 2,7 mH, sendo possível comprovar o correto funcionamento. Durante os ensaios foi utilizado o controlador PI com ganhos semelhantes aos de simulação ($k_p = 1000$ e $k_i = 40$). Na Figura 6.13 são mostrados os barramentos CC de cada submódulo (V_{cc1} , V_{cc2} , V_{cc3}) e a tensão de saída do inversor (v_{inv}) quando o sistema é conectado à rede, onde é possível observar os sete níveis de tensão.

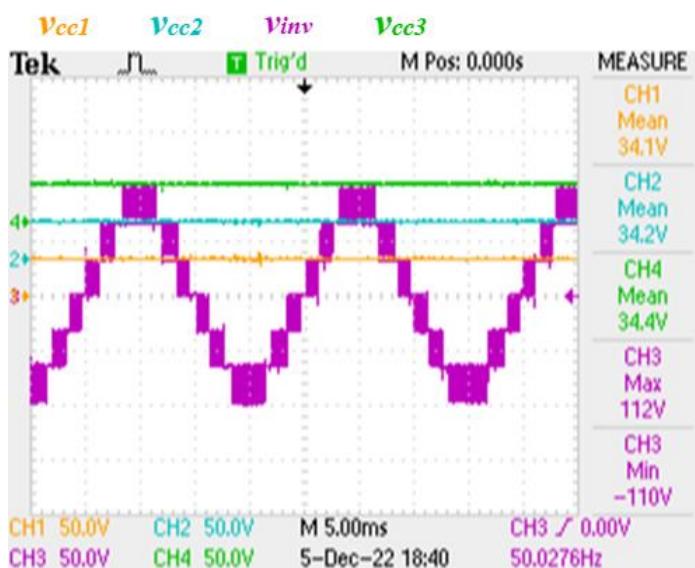


Figura 6.13 – Resultados experimentais das tensões nos barramentos e à saída do inversor.

Inicialmente, os resultados para injeção de energia na rede elétrica utilizando o controlo de corrente PI não foram satisfatórios. Para resolver esse problema foi implementado o controlo de corrente preditivo exposto no terceiro capítulo, obtendo-se os resultados apresentados na Figura 6.14, para diferentes amplitudes de corrente. Nesta imagem é mostrado o valor da tensão no barramento CC do primeiro submódulo (V_{cc1}), a corrente sintetizada pelo conversor (i_{rede}), a tensão à saída do conversor (v_{inv}) e a tensão da rede elétrica (v_{rede}). Em todos os resultados expostos, a corrente de saída encontra-se em fase e com a mesma frequência da tensão da rede, a amplitude atinge os valores desejados e os sete níveis da tensão de saída estão presentes, comprovando o correto funcionamento do sistema para o controlo de corrente preditivo. Além disso, como não há necessidade de ajustar ganhos não existem atrasos entre a corrente e a tensão. De notar que, tal como nas simulações computacionais, o

sentido positivo da corrente é do inversor para a rede, logo a injeção de energia na rede elétrica é realizada quando a corrente está em fase com a tensão.

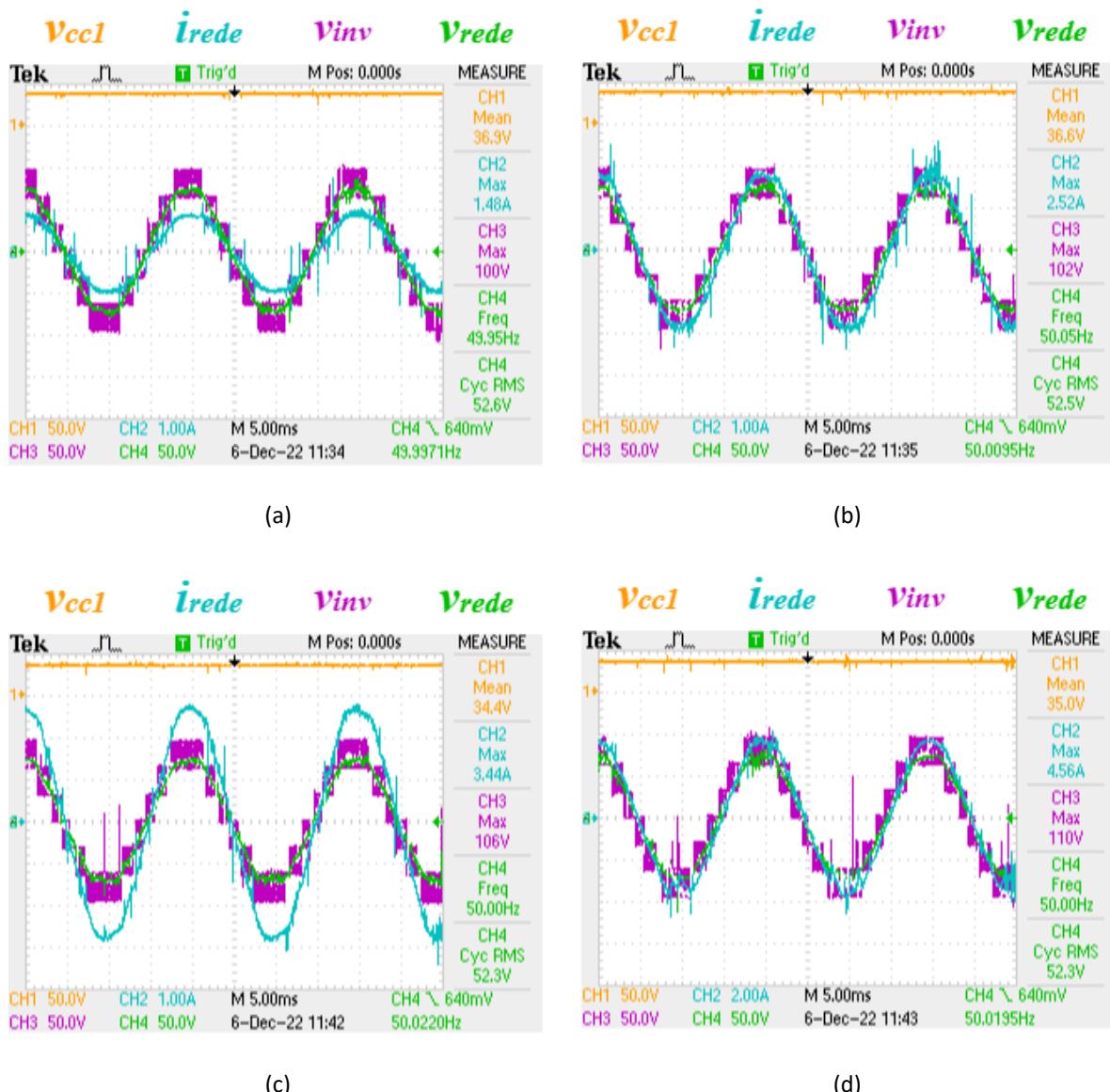


Figura 6.14 – Resultados experimentais do controlo de corrente preditivo para diferentes amplitudes: (a) 1 A; (b) 2 A; (c) 3 A; (d) 4 A.

Posteriormente, foram apresentados os sinais de saída do DAC para a corrente com amplitude de 2 A, permitindo analisar a evolução da corrente de saída em relação à corrente de referência. Na Figura 6.15 é mostrada a tensão gerada pelo DAC que representa a corrente de referência (i_{d_ref}), a corrente de saída do conversor (i_{rede}), a tensão gerada pelo DAC que representa a corrente de saída (i_{d_rede}) e a tensão da rede elétrica à saída do transformador (v_{rede}). Tal como previsto, a corrente de saída segue de perto a corrente de referência.

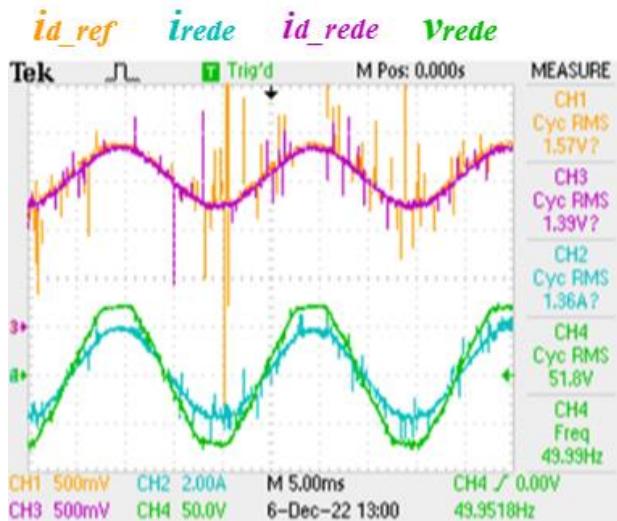


Figura 6.15 – Resultado experimental do controlo de corrente preditivo para os sinais de saída do DAC com uma amplitude de corrente de 2 A.

Além dos resultados apresentados, testou-se o comportamento dinâmico do sistema. Para isso, fez-se variar a amplitude da corrente de saída de 1 A para 3 A, sendo os resultados apresentados na Figura 6.16.

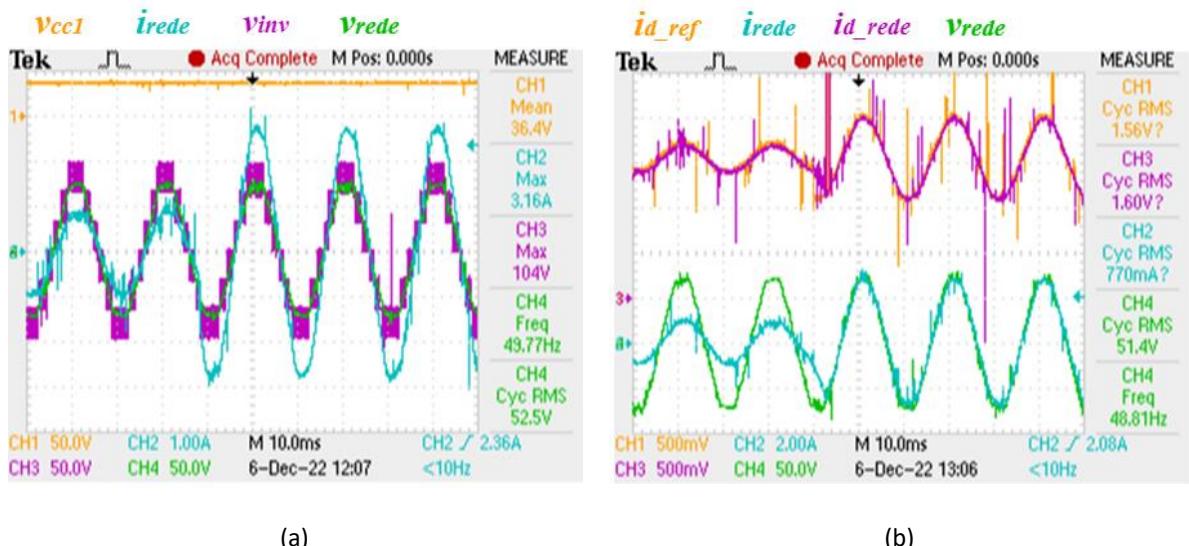


Figura 6.16 – Resultados experimentais do controlo de corrente preditivo na transição ascendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.

Na Figura 6.17 fez-se variar a amplitude da corrente de 3 A para 1 A, ou seja, transição descendente. Em ambos os casos, a corrente segue de perto a referência, atingindo rapidamente a amplitude desejada.

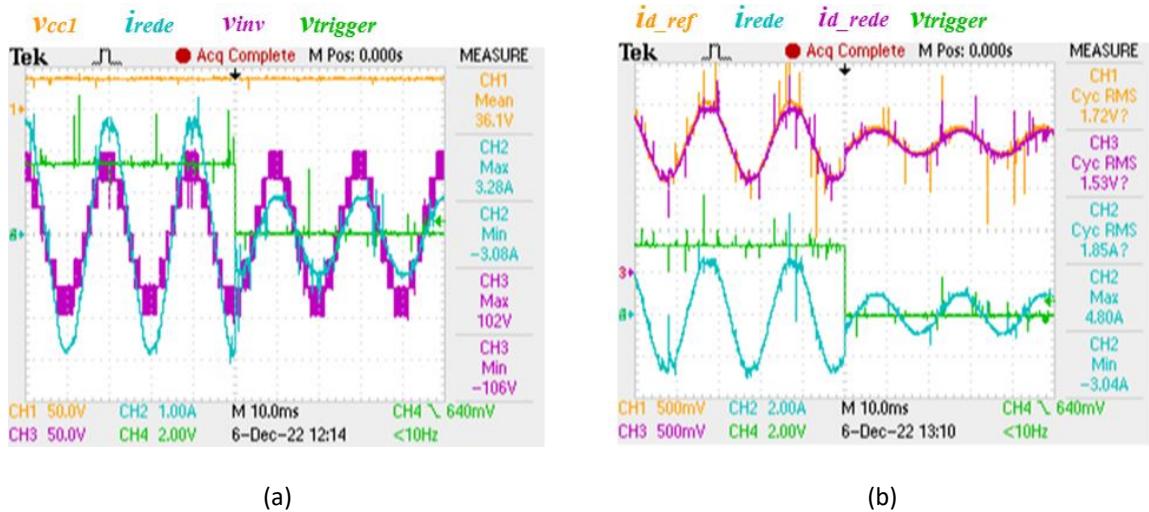


Figura 6.17 – Resultados experimentais do controlo de corrente preditivo na transição descendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.

Por fim, na Figura 6.18 são apresentadas as potências das fontes de alimentação para os barramentos CC, quando está a ser sintetizada uma corrente com 3 A de pico. Em (a) são mostradas as formas de onda da tensão e corrente à saída da fonte de alimentação do primeiro (V_{cc1} , i_1) e segundo submódulo (V_{cc2} e i_2) e a potência no primeiro submódulo (sinal de *MATCH*), resultante da multiplicação entre V_{cc1} e i_1 . Em (b) as tensão e correntes são as mesmas, contudo o sinal de *MATCH* resulta da multiplicação dos sinais V_{cc2} e i_2 , mostrando a potência no segundo submódulo. Visto que, a potência está igualmente dividida entre os submódulos, é comprovado o correto funcionamento do sistema. De notar que, apenas foram apresentados os valores para dois submódulos porque o osciloscópio somente permitia visualizar quatro sinais em simultâneo.

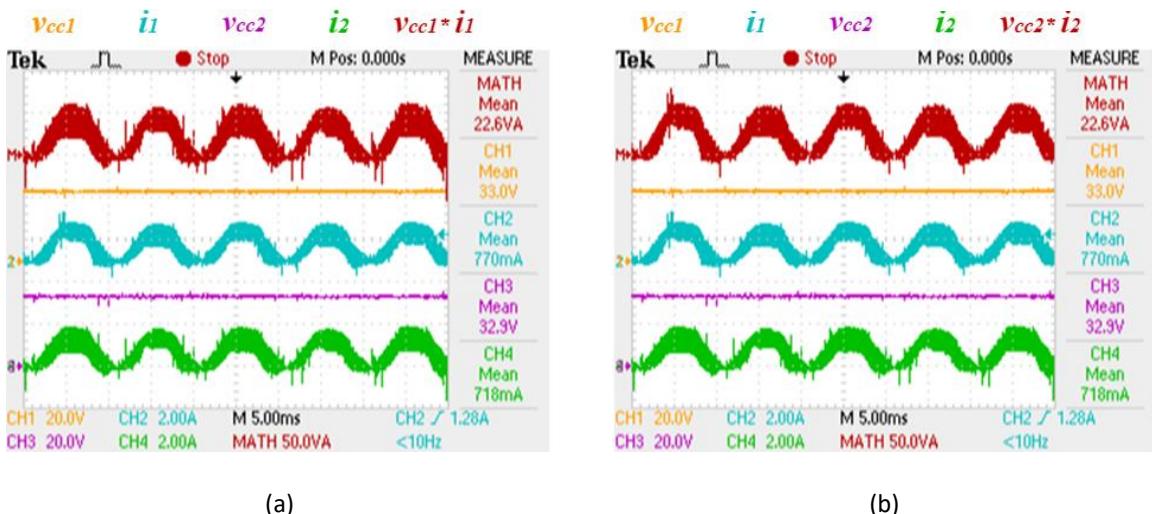


Figura 6.18 – Resultados experimentais do controlo de corrente preditivo para a potência no: (a) primeiro submódulo; (b) segundo submódulo.

Comprovado o correto funcionamento do sistema com o controlo de corrente preditivo, tentou-se, novamente, obter resultados satisfatórios com o controlo PI. Após um ajuste mais minuciosos dos ganhos k_p e k_i , conseguiram-se os resultados apresentados na Figura 6.19. Como nos resultados anteriores, é mostrado, para diferentes amplitudes de corrente, o valor da tensão no barramento CC do primeiro submódulo (V_{cc1}), a corrente sintetizada pelo conversor (i_{rede}), a tensão à saída do conversor (v_{inv}) e a tensão da rede elétrica (v_{rede}). Em todas as ocasiões a tensão de saída apresenta sete níveis e a corrente de saída alcança a amplitude desejada, porém, tal como esperado, esta encontra-se atrasada em relação à tensão da rede elétrica. Desta forma, é comprovando o correto funcionamento do sistema com o controlo de corrente PI.

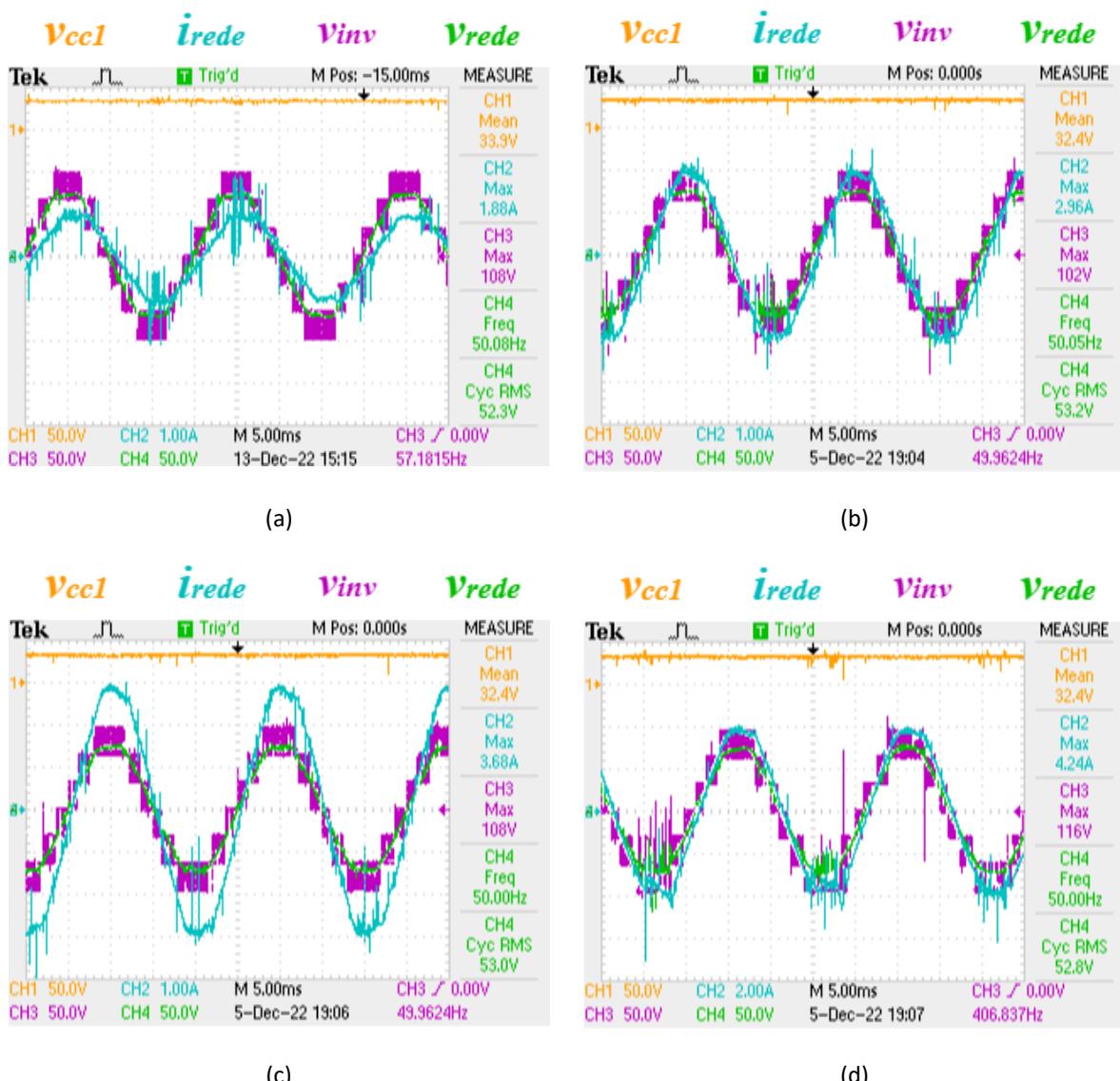


Figura 6.19 – Resultados experimentais do controlo de corrente PI para diferentes amplitudes: (a) 1 A; (b) 2 A; (c) 3 A; (d) 4 A.

Mais uma vez, foi testado o comportamento dinâmico do sistema. Para isso, fez-se variar a amplitude da corrente de saída de 1 A para 3 A na Figura 6.20, sendo possível observar um pico na corrente durante a transição ascendente.

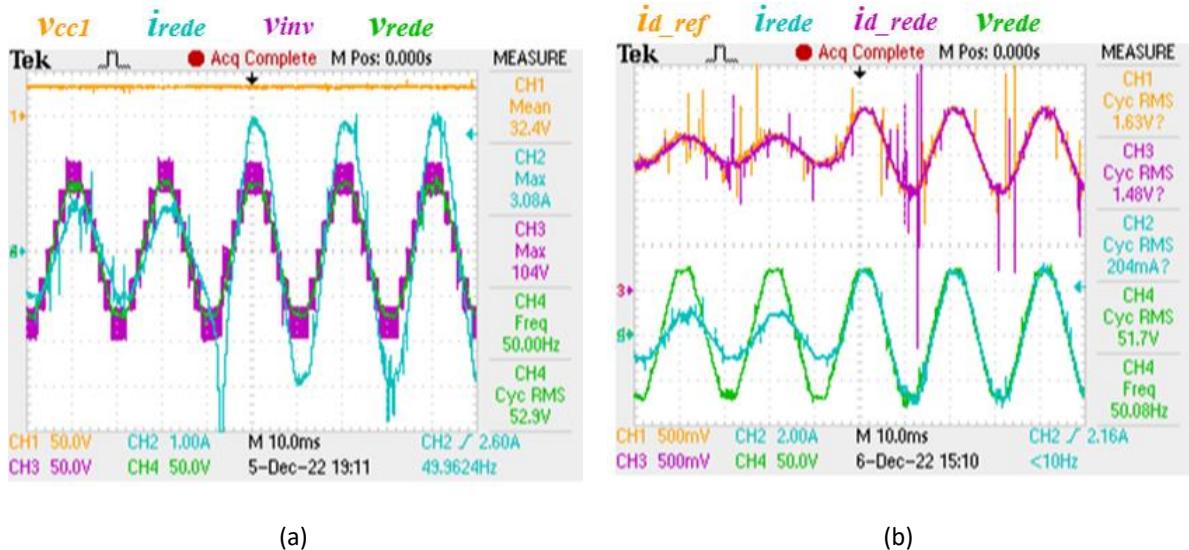


Figura 6.20 – Resultados experimentais do controlo de corrente preditivo na transição ascendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.

Na Figura 6.21 fez-se variar a amplitude da corrente de 3 A para 1 A, ou seja, transição descendente. Em ambos os casos, a corrente segue de perto a referência, atingindo rapidamente a amplitude desejada, e as formas de onda da tensão não apresentam alterações, mantendo os sete níveis característicos.

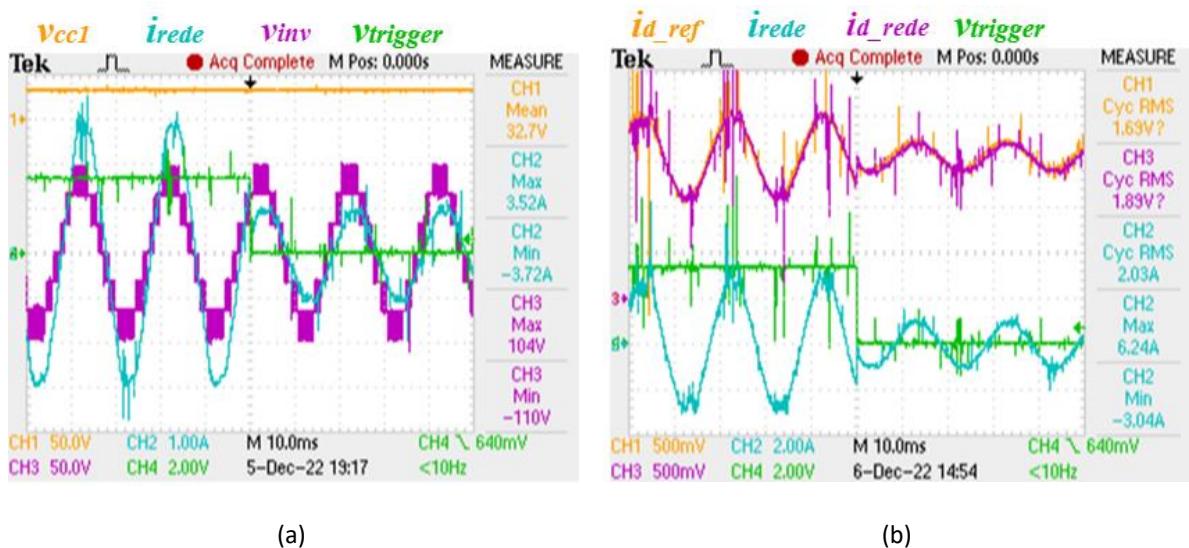


Figura 6.21 – Resultados experimentais do controlo de corrente preditivo na transição descendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.

Por fim, as potências das fontes de alimentação dos dois primeiros submódulos são apresentadas na Figura 6.22, para uma corrente sintetizada com 3 A de pico. Em (a) são mostradas as formas de onda da tensão e corrente do primeiro (V_{cc1} e i_1) e segundo (V_{cc2} e i_2) submódulos e a potência no primeiro submódulo (*MATCH*) que resulta da multiplicação entre os sinais V_{cc1} e i_1 . Em (b) as tensões e correntes mantêm-se, porém, o sinal de *MATCH* resulta da multiplicação dos sinais V_{cc2} e i_2 , mostrando a potência no segundo submódulo. Novamente, é possível comprovar o correto funcionamento do sistema, visto que, a potência está igualmente dividida entre os submódulos.

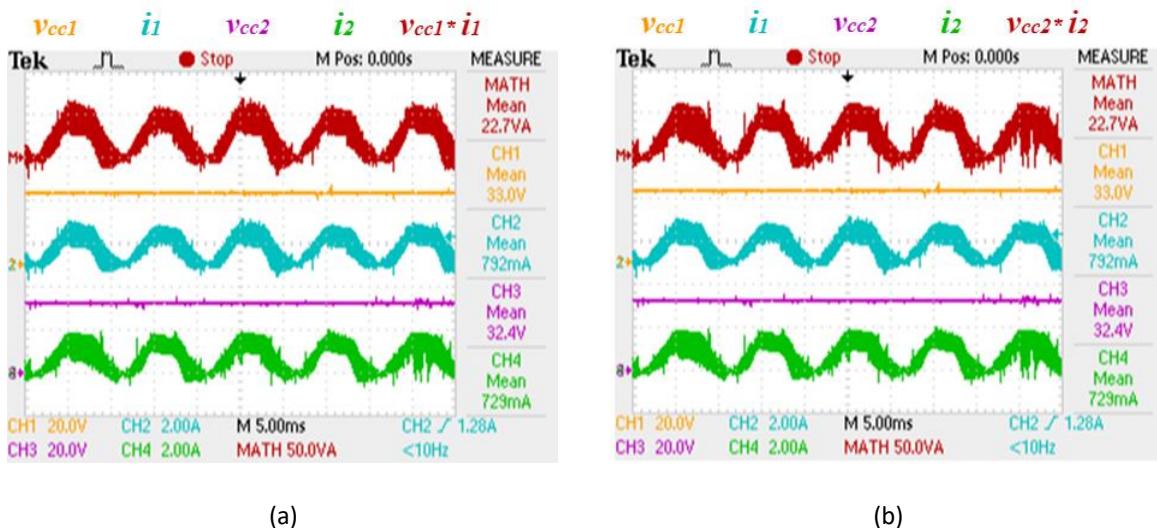


Figura 6.22 – Resultados experimentais do controlo de corrente PI para a potência no: (a) primeiro submódulos; (b) segundo submódulo.

6.3.3 Conversor com Dois Submódulos

Tal como nas simulações computacionais, com o intuito de testar a molaridade e adaptabilidade dos algoritmos implementados, realizaram-se ensaios experimentais para o sistema de controlo modular descentralizado para dois conversores de eletrónica de potência ligados em cascata. Para tal, foram feitas algumas modificações no protótipo, nomeadamente: desconectaram-se as comunicações e o hardware do terceiro submódulo e aumentou-se a tensão dos barramentos CC. De notar que, em nenhum momento os algoritmos foram alterados. Após isso, começou-se por testar e validar os algoritmos da comunicação em anel, medindo os sinais elétricos e utilizando a ferramenta de *debug* do microcontrolador a funcionar como *master*. Na Figura 6.23 é mostrado o painel que permite visualizar o valor das expressões durante o *debug* no primeiro estágio de configuração. Como esperado, que o

master envia uma trama inicial (*send_array*) com apenas três valores e recebe uma trama (*receive_array*) com o número dos respetivos submódulos ligados, comprovando o correto funcionamento do estágio de contagem.

Expression	Type	Value	Address
(0)= control_state	unsigned int	1	0x0000D080@...
(0)= n_slaves	unsigned int	2	0x0000D08C@...
(0)= flag_receive_counter	unsigned int	1	0x0000D086@...
(0)= flag_receive_config	unsigned int	0	0x0000D087@...
(0)= flag_dc_bus	unsigned int	0	0x0000D088@...
send_array			
(0)= [0]	unsigned int	0	0x0000D038@...
(0)= [1]	unsigned int	1	0x0000D039@...
(0)= [2]	unsigned int	36	0x0000D03A@...
receive_array			
(0)= [0]	unsigned int	0	0x0000D000@...
(0)= [1]	unsigned int	1	0x0000D001@...
(0)= [2]	unsigned int	1	0x0000D002@...
(0)= [3]	unsigned int	2	0x0000D003@...
(0)= [4]	unsigned int	36	0x0000D004@...
(0)= [5]	unsigned int	0	0x0000D005@...
(0)= [6]	unsigned int	0	0x0000D006@...
(0)= [7]	unsigned int	0	0x0000D007@...

Figura 6.23 – Valor das expressões do *master* durante o estágio de contagem.

Na Figura 6.24 são apresentados os sinais elétricos à saída da comunicação em anel de cada microcontrolador (com_{p0} saída no *master*, com_{p1} saída no primeiro *slave* e com_{p2} saída no segundo *slave*). Novamente, o tamanho da trama vai aumentando à medida que é recebida pelos *slaves*, começando com dois pulsos que correspondem à função e ao stop (o endereço tem valor 0) e terminando com quatro pulsos, consequentes da adição dos dois submódulos. Além disso, manteve-se a escala usada nos resultados apresentados anteriormente e foi mostrada a saída da comunicação no terceiro *slave* (com_p3), comprovando que esta não está ativa. O tempo total do estágio de contagem é de aproximadamente 300 μ s.

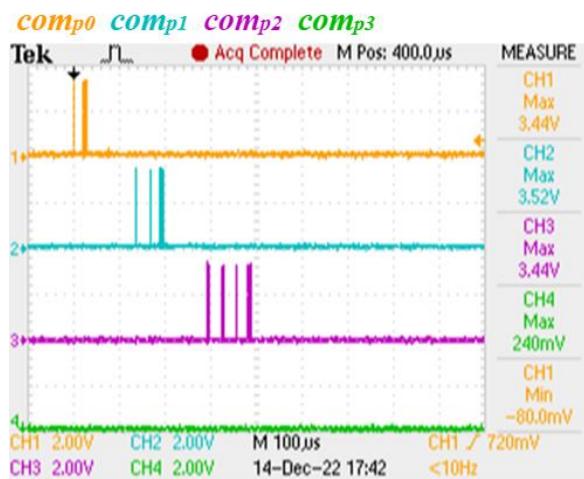


Figura 6.24 – Resultado experimental das comunicações em anel durante o estágio de contagem.

O painel com o valor das expressões do segundo estágio de configuração é mostrado na Figura 6.25, onde se verifica que a trama enviada pelo master é a mesma que a recebida. Agora existem quatro braços no sistema, logo o desfasamento entre as ondas portadoras é de 90° , obtendo-se os seguintes valores: $\{0, 90, 180, 270\}$. De relembrar que, optou-se por enviar apenas o valor do desfasamento do primeiro baço de cada submódulo, sendo o desfasamento do segundo calculado no slave.

Expression	Type	Value	Address
(*) control_state	unsigned int	2	0x0000D0C0...
(*) n_slaves	unsigned int	2	0x0000D0CC...
(*) flag_receive_counter	unsigned int	0	0x0000D0C6...
(*) flag_receive_config	unsigned int	1	0x0000D0C7...
(*) flag_dc_bus	unsigned int	0	0x0000D0C8...
(*) send_array	unsigned int[50]	[0,2,0,180,36...]	0x0000D040@...
(*) receive_array	unsigned int[50]	[0,2,0,180,36...]	0x0000D000@...
(*) [0]	unsigned int	0	0x0000D000@...
(*) [1]	unsigned int	2	0x0000D001@...
(*) [2]	unsigned int	0	0x0000D002@...
(*) [3]	unsigned int	180	0x0000D003@...
(*) [4]	unsigned int	36	0x0000D004@...
(*) [5]	unsigned int	0	0x0000D005@...
(*) [6]	unsigned int	0	0x0000D006@...
(*) [7]	unsigned int	0	0x0000D007@...
(*) [8]	unsigned int	0	0x0000D008@...
(*) [9]	unsigned int	0	0x0000D009@...
(*) [10]	unsigned int	0	0x0000D00A...

Figura 6.25 – Valor das expressões do *master* durante o estágio de configuração.

Na Figura 6.26 são apresentados os sinais elétricos à saída da comunicação em anel de cada microcontrolador. Como esperado, à medida que a trama é recebida pelos *slaves* o seu tamanho mantém-se o mesmo. O tempo total do estágio de configuração é de aproximadamente 390 μ s.

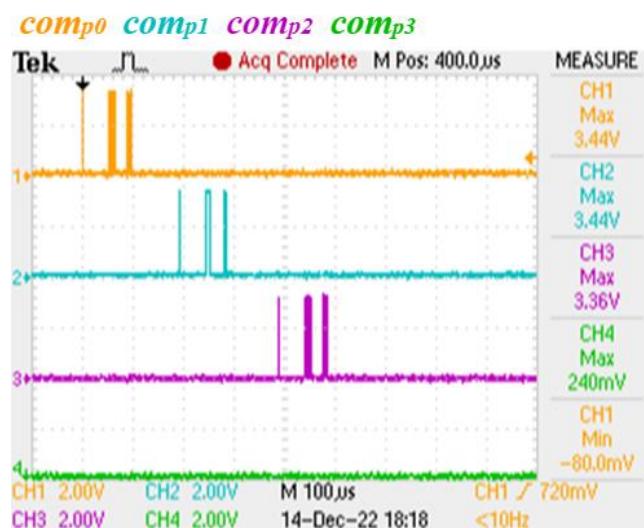


Figura 6.26 – Resultado experimental das comunicações em anel durante o estágio de configuração.

O painel com o valor das expressões do terceiro estágio de configuração é mostrado na Figura 6.27. Neste estágio, que a trama enviada pelo *master* tem o mesmo tamanho que a recebida, contudo os dados são alterados. Como no caso anterior, a trama vai sendo preenchida pelos *slaves*, na respetiva posição, com o valor 60.

Expression	Type	Value	Address
(*) control_state	unsigned int	3	0x0000D080@...
(*) n_slaves	unsigned int	2	0x0000D08C@...
(*) flag_receive_counter	unsigned int	0	0x0000D08E@...
(*) flag_receive_config	unsigned int	0	0x0000D08F@...
(*) flag_dc_bus	unsigned int	1	0x0000D088@...
(*) send_array	unsigned int[50]	[0,3,0,36...]	0x0000D040@...
(*) receive_array	unsigned int[50]	[0,3,60,60,36...]	0x0000D000@...
(*) [0]	unsigned int	0	0x0000D000@...
(*) [1]	unsigned int	3	0x0000D001@...
(*) [2]	unsigned int	60	0x0000D002@...
(*) [3]	unsigned int	60	0x0000D003@...
(*) [4]	unsigned int	36	0x0000D004@...
(*) [5]	unsigned int	0	0x0000D005@...
(*) [6]	unsigned int	0	0x0000D006@...
(*) [7]	unsigned int	0	0x0000D007@...
(*) [8]	unsigned int	0	0x0000D008@...
(*) [9]	unsigned int	0	0x0000D009@...
(*) [10]	unsigned int	0	0x0000D00A@...

Figura 6.27 – Valor das expressões do *master* durante o estágio de recolha.

Na Figura 6.28 são apresentados os sinais elétricos à saída da comunicação em anel de cada microcontrolador. Como esperado, a trama vai sendo preenchida à medida que é recebida pelos *slaves*, sendo o tempo do estágio de recolha de aproximadamente 380 µs.

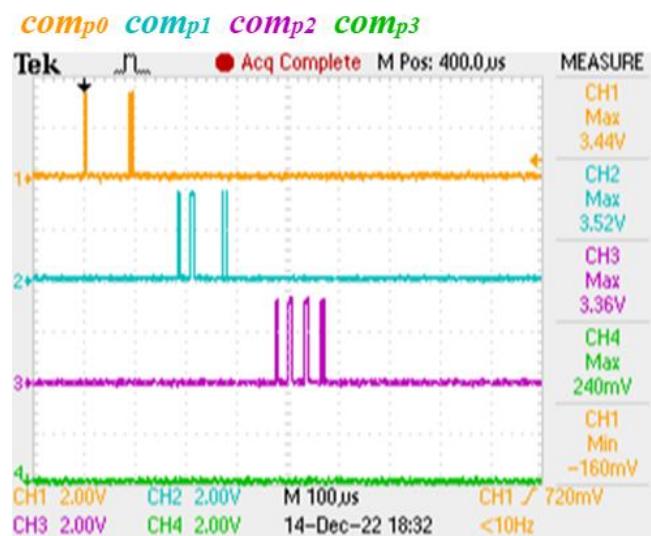


Figura 6.28 – Resultado experimental das comunicações em anel durante o estágio de configuração.

Além da topologia em anel, foi utilizada a topologia em barramento, permitindo enviar o valor da onda moduladora calculada pelo algoritmo de controlo de correte. Para este ensaio podem

considerar-se os resultados apresentados na Figura 6.10, visto que, a alteração do número de submódulos não influencia esta topologia.

Após testar e validar as comunicações, verificaram-se os sinais de PWM, nomeadamente, os desfasamentos e a saída das placas de *driver*. Na Figura 6.29 são apresentados os sinais de PWM em cada braço do sistema, com um *duty cycle* de 10% e uma frequência de 10 kHz. São mostrados os sinais dos quatro braços, onde é possível observar o desfasamento de 90° (25 µs para um período de 100 µs) entre as formas de onda.

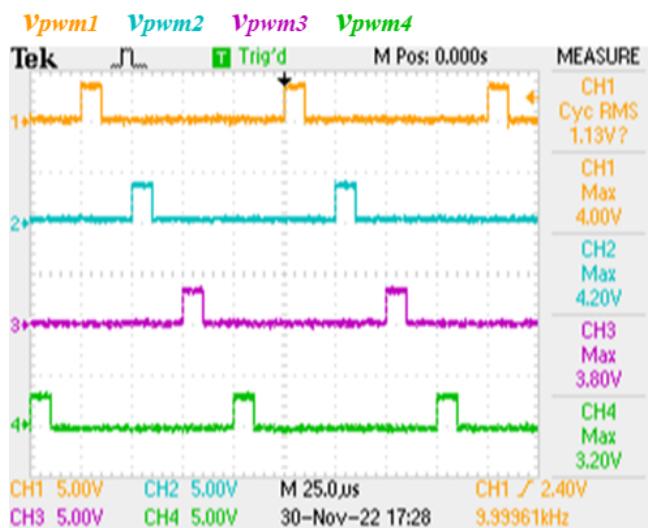


Figura 6.29 – Resultados experimentais do desfasamento dos sinais de PWM para os quatro braços do sistema.

De seguida, foram realizados os ensaios experimentais à totalidade do sistema proposto, aumentando a tensão dos barramentos CC para 53 V e mantendo a potência total do sistema em 300 VA, consequentemente, o valor eficaz da corrente é de 2,83 A (4 A de pico). O conversor foi conectado à rede elétrica com um valor eficaz da tensão de 50 V com recurso a um transformador. Na Figura 6.30 são mostrados os barramentos CC de cada submódulo (V_{cc1} e V_{cc2}), a tensão de saída do inversor (v_{inv}) quando o sistema é conectado à rede e a tensão da rede elétrica (v_{rede}). Além disso, foi utilizada a funcionalidade *MATCH* do osciloscópio para somar as formas de onda dos dois barramentos ($V_{cc1} + V_{cc2}$). Como previsto, é possível observar os cinco níveis na tensão à saída do inversor.

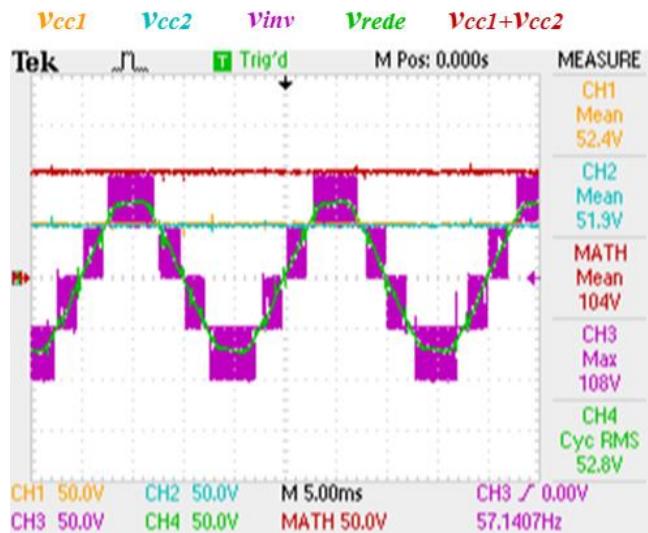
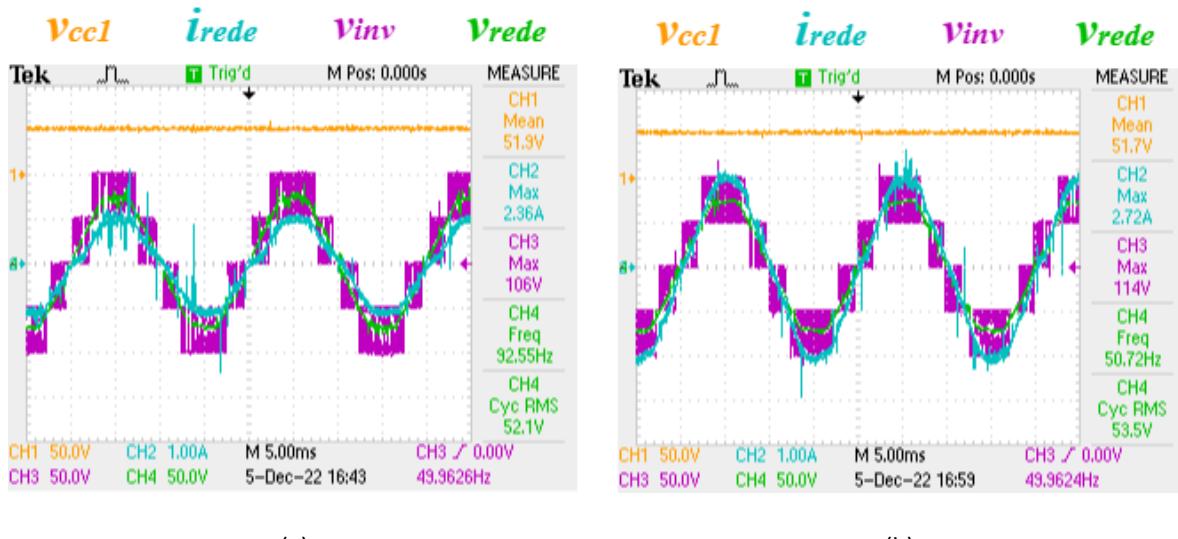


Figura 6.30 – Resultados experimentais das tensões nos barramentos e à saída do inversor.

Para o sistema com dois submódulos, foi apenas usada o controlo de corrente preditivo, visto que, não é necessidade ajustar ganhos e, nos ensaios anteriores, alcançam-se melhores resultados. Na Figura 6.31 são apresentados os resultados experimentais para diferentes amplitudes de corrente. Nesta imagem é mostrado o valor da tensão no barramento CC do primeiro submódulo (V_{cc1}), a corrente sintetizada pelo conversor (i_{rede}), a tensão à saída do conversor (v_{invo}) e a tensão da rede elétrica (v_{rede}). Em todos os resultados expostos, a corrente de saída encontra-se em fase e com a mesma frequência da tensão da rede, a amplitude atinge os valores desejados e os cinco níveis da tensão de saída são alcançados, comprovando o correto funcionamento do sistema.



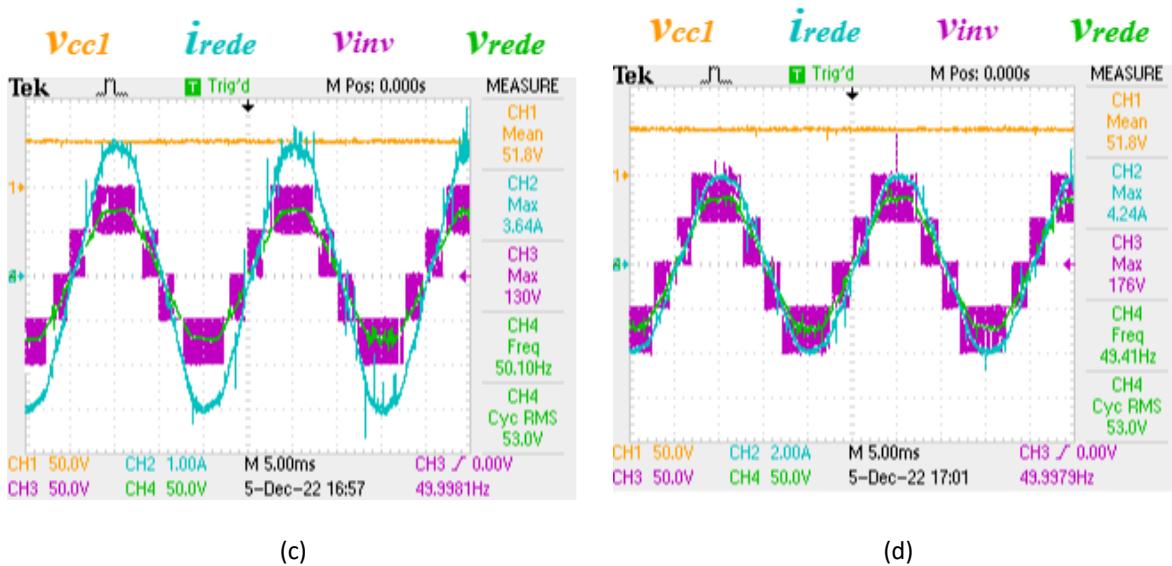


Figura 6.31 – Resultados experimentais do controlo de corrente preditivo para diferentes amplitudes: (a) 1 A; (b) 2 A; (c) 3 A; (d) 4 A.

Posteriormente, foram apresentados os sinais de saída do DAC para a corrente com amplitude de 2 A, permitindo analisar a evolução da corrente de saída em relação à corrente de referência. Na Figura 6.32 é mostrada a tensão gerada pelo DAC que representa a corrente de referência (i_{d_ref}), a corrente de saída do conversor (i_{rede}), a tensão gerada pelo DAC que representa a corrente de saída (i_{d_rede}) e a tensão da rede elétrica à saída do transformador (v_{rede}). Tal como previsto, a corrente de saída segue de perto a corrente de referência.

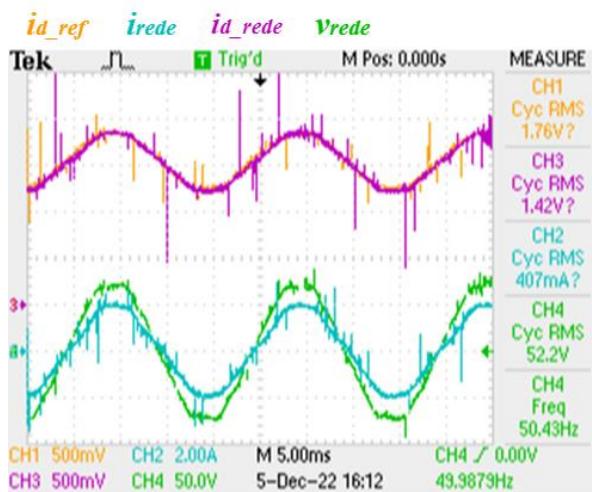


Figura 6.32 – Resultado experimental do controlo de corrente preditivo para os sinais de saída do DAC com uma amplitude de corrente de 2 A.

Na Figura 6.33 são apresentadas as potências das fontes de alimentação dos barramentos CC, quando está a ser sintetizada uma corrente com 2 A de pico. Em (a) são mostradas as formas de onda da tensão e corrente para o primeiro (V_{cc1} e i_1) e segundo submódulo (V_{cc2} e i_2) e a

potência no primeiro submódulo (sinal de *MATCH*), resultante da multiplicação entre V_{cc1} e i_1 . Em (b) as tensão e correntes são as mesmas, contudo o sinal de *MATCH* resulta da multiplicação dos sinais V_{cc2} e i_2 , mostrando a potência no segundo submódulo. Mais uma vez, é possível comprovar o correto funcionamento do sistema, visto que, a potência está igualmente dividida entre os submódulos.

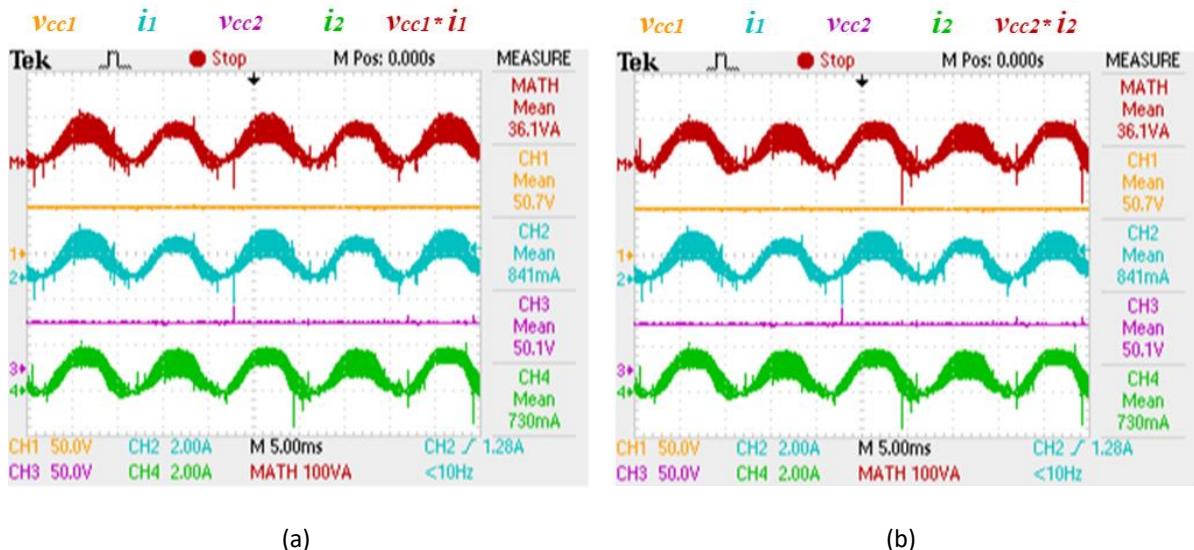


Figura 6.33 – Resultados experimentais do controlo de corrente preditivo para a potência no: (a) primeiro submódulo; (b) segundo submódulo.

Por fim, testou-se o comportamento dinâmico do sistema. Para isso, fez-se variar a amplitude da corrente de saída de 1 A para 3 A, sendo os resultados apresentados na Figura 6.34.

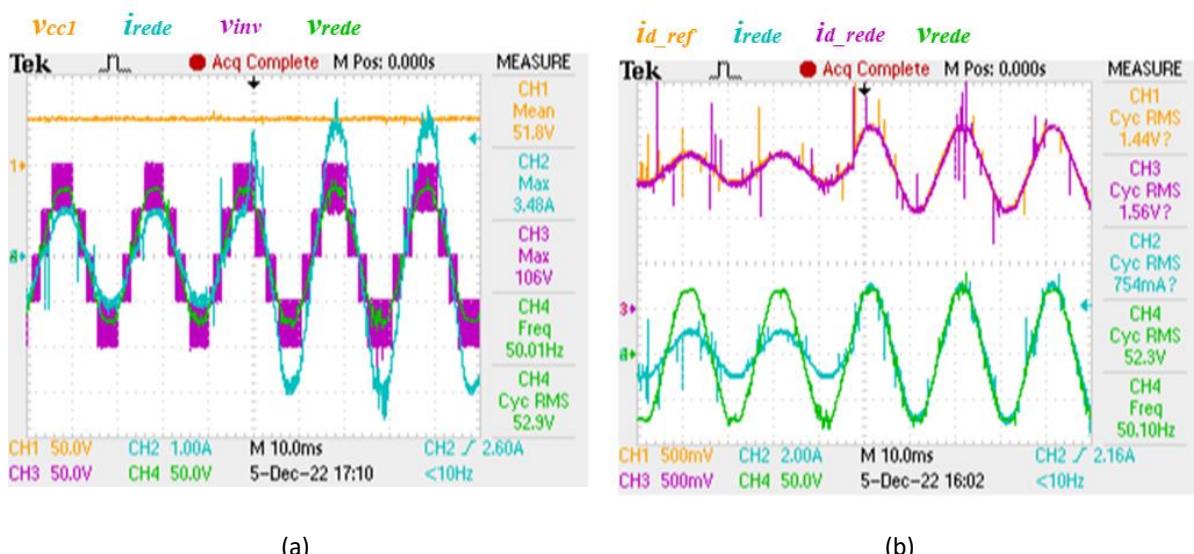


Figura 6.34 – Resultados experimentais do controlo de corrente preditivo na transição ascendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.

Na Figura 6.35 fez-se variar a amplitude da corrente de 3 A para 1 A, ou seja, transição descendente. Em ambos os casos, a corrente segue de perto a referência, atingindo rapidamente a amplitude desejada.

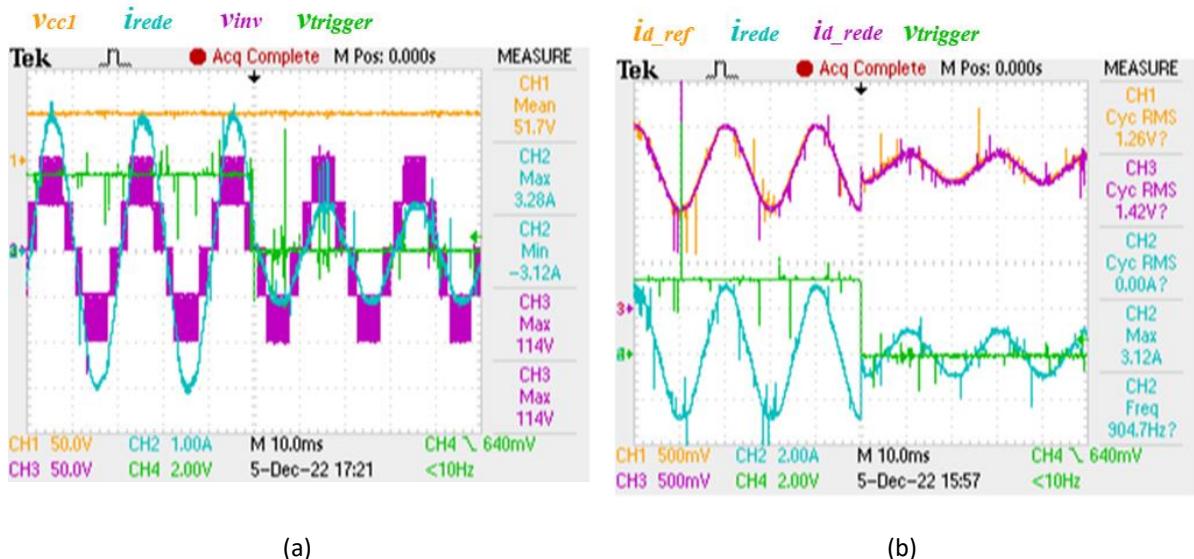


Figura 6.35 – Resultados experimentais do controlo de corrente preditivo na transição descendente: (a) corrente e tensão sintetizadas; (b) saídas do DAC.

6.4 Conclusões

Ao longo deste capítulo foi explicado o princípio de funcionamento do controlo implementado e foram abordados os resultados experimentais obtidos para dois modos de operação diferentes, um com o sistema a operar com três submódulos ligados e outro com apenas dois submódulos. No início do capítulo, foram apresentados os diagramas das máquinas de estrados, que serviram de base para o desenvolvimento dos algoritmos, e a estrutura utilizada para as tramas da comunicação em anel, permitindo uma melhor compreensão do sistema de controlo.

De seguida, foram mostrados os resultados dos ensaios experimentais. Inicialmente foi apresentada a sincronização do algoritmo de EPLL com a rede elétrica, permitindo analisar o tempo de sincronismo e obtendo um resultado em regime permanente semelhante ao obtido em simulação computacional.

Posteriormente, foram realizados testes para validar as topologias de comunicação do sistema de controlo modular descentralizado para três conversores ligados em cascata. Ambas as

topologias foram bem conseguidas, a topologia em anel sendo capaz de identificar o número de submódulos e atribuir o desfasamento das portadoras e a topologia em barramento transmitindo informações críticas sem atrasos. Em relação à injeção de energia na rede elétrica, foram testadas duas técnicas de controlo de corrente distintas, o controlo preditivo e PI. A utilização do controlo preditivo deveu-se à dificuldade do ajuste de ganhos no controlo PI que, numa fase inicial, impossibilitou a aquisição de resultados satisfatórios. Comparando as duas técnicas, ambas apresentaram resultados similares: a forma de onda da corrente de saída segue de perto a referência, alcançando sempre a amplitude desejada e estando à mesma frequência que a tensão da rede; a tensão sintetizada apresenta os sete níveis característicos dos três submódulos ligados em cascata; e a potência do sistema é igualmente dividida pelos submódulos. Além disso, relativamente ao comportamento dinâmico do sistema, tanto nas transições ascendentes como descendentes, a corrente é rapidamente ajustada para os valores pretendidos, seguindo a referência. Apesar do correto funcionamento das duas técnicas, no controlo de corrente PI constatou-se a dificuldade acrescida da afinação dos ganhos, o atraso da corrente em relação à tensão da rede e uma maior suscetibilidade ao ruido, principalmente com o aumento da potência.

Em relação aos resultados adquiridos para o sistema de controlo modular descentralizado para dois conversores ligados em cascata, o principal objetivo consistiu em testar a modularidade do protótipo, quer a nível de hardware, como de software. A nível de hardware a simplicidade na adição ou subtração de um submódulos foi conseguida, sendo apenas necessário alterar algumas ligações. A nível de software, comparando os resultados das tramas de configuração e analisando os sinais de PWM dos dois modos de operação, foi possível observar a adaptabilidade dos algoritmos implementados que calcularam corretamente todos os parâmetros de forma automática. Adicionalmente, foi demonstrada a injeção de energia na rede elétrica, obtendo resultados com qualidade similar aos da operação com três submódulos. Mais uma vez, foi comprovado o correto funcionamento do sistema com a forma de onda da corrente de saída a seguir de perto a referência, a potência do sistema a ser igualmente dividida pelos submódulos e a tensão sintetizada a apresentar os cinco níveis, característicos dos dois submódulos ligados em cascata.

Capítulo 7

Conclusões e Sugestões de Trabalho Futuro

7.1 Conclusões

Esta dissertação de mestrado teve como intuito o desenvolvimento de um sistema de controlo modelar descentralizado para conversores de eletrónica de potência em cascata. Pretendeu-se inovar com a criação do sistema de controlo descentralizado, algo nunca realizado no grupo de eletrónica de potência da Universidade do Minho.

No Capítulo 1 foi realizada uma pequena introdução dos conversores modulares multinível, com a apresentação da evolução da eletrónica de potência e das principais aplicações do sistema. Além disso, foram abordadas as temáticas de estudo desta dissertação, onde foi feito um breve enquadramento do problema. Uma vez introduzido o problema e identificadas as possíveis contribuições, nos próximos dois capítulos foram retratados os principais conteúdos tecnológicos identificados na literatura, relacionados com a eletrónica de potência, topologias de comunicação e algoritmos de controlo constituintes da arquitetura do sistema a desenvolvido.

No Capítulo 2 foi feita uma breve introdução aos conversores multinível, apresentando as principais vantagens quando relacionados com os conversores de dois níveis. Além disso, dentro desta topologia, foram abordados os conversores modulares multinível, especialmente os MMCC. Estes apresentaram elevado interesse para aplicações de média e alta tensão, permitindo aumentar a tensão e a potência nominal do conversor, adicionando módulos em série, e operar em caso de falhas. De seguida, foram abordados os benefícios da estratégia de controlo descentralizada para sistemas MMCC e comparadas as principais tecnologias utilizadas no desenvolvimento de um sistema de comunicação. Foi possível concluir que as melhorias na modularidade, confiabilidade e distribuição da carga computacional, características do controlo descentralizado, seriam conseguidas com um sistema de comunicação que permitisse a transmissão adequada de dados entre os

microcontroladores. Dessa forma, foi possível analisar que, para estes sistemas, a utilização de fibra ótica é normalmente preferível em relação ao meio elétrico, devido à baixa latência na transmissão de dados enquanto proporciona o isolamento galvânico indispensável. Adicionalmente, foi concluída a necessidade de utilizar duas topologias de comunicação, uma para detetar o número de submódulos ligados, topologia em anel, e outra para enviar dados críticos sem atrasos, topologia em barramento. Por fim, foram analisados os diferentes protocolos de comunicação usados, com especial foco no protocolo EtherCAT.

O estudo do estado da arte foi terminado no Capítulo 3. Neste capítulo realizou-se uma comparação das diferentes configurações existentes para os submódulos do MMCC, tais como: meia ponte, ponte completa, NPC e T-NPC. Desta comparação foi possível concluir que a topologia de conversor em ponte completa é a mais adequada, uma vez que permitir sintetizar três níveis de tensão utilizando poucos semicondutores, é capaz de gerar $\pm V_{cc}$ a partir de um único barramento CC e possui uma complexidade de controlo reduzida. De seguida, foram abordadas as técnicas de controlo aplicadas à tensão e corrente de saída do MMCC. Dentro do controlo aplicado à tensão, foram analisadas as técnicas de SPWM com desfasamento de nível e fase, sendo possível concluir que a técnica PSC é a que melhor se enquadra nesta aplicação, fazendo uma divisão semelhante da potência de operação entre os submódulos. Relativamente ao controlo de corrente, foram comparadas diversas técnicas tais como: controlo de corrente por histerese, por *periodic sampling*, proporcional integral com SPWM e preditivo com SPWM. Após essa análise, concluiu-se que as técnicas de controlo de corrente PI e preditivo, aliadas à SPWM com desfasamento de fase, apresentam as características mais adequadas para a aplicação em causa. Posteriormente, foi realizado um estudo aos algoritmos de regulação do barramento CC, fazendo referência a algoritmos propostos para combater as limitações do método de classificação. De realçar que, uma vez que foram usadas fontes para alimentar os barramentos CC, não foram utilizados os algoritmos de regulação citados. Por fim, foram apresentados os algoritmos de sincronização com a rede elétrica, onde se mostrou o algoritmo EPLL, capaz de superar os erros da PLL convencional.

Após o estudo teórico, no Capítulo 4 foi exposta a topologia utilizada e validada em simulação com recurso à ferramenta computacional PSIM. Inicialmente, foi realizada uma breve explicação da topologia, tanto a nível de hardware, como de software. Posteriormente, de

uma forma complementar, foi efetuada uma descrição detalhada do modelo de simulação utilizado e apresentadas as principais características do sistema. De seguida, foram apresentados os resultados obtidos para dois modelos de simulação distintos, um correspondente à topologia utilizada com três submodulos ligados em cascata e outro modelo com apenas dois submodulos, sendo utilizando o segundo modelo para validar a modularidade do sistema e a adaptabilidade dos algoritmos. Foram estudados, para ambos os modelos de simulação, os resultados das topologias de comunicação em anel; os valores calculados para os desfasamentos das portadoras; o algoritmo de sincronização com a rede elétrica através da EPLL; o cálculo da corrente de referência; a injeção da energia na rede elétrica, através das formas de onda da tensão e corrente de saída do MMCC; a igual distribuição da potência entre os submódulos e ainda o comportamento dinâmico do sistema. Com este estudo foi possível validar o correto funcionamento da topologia utilizada e a sua modularidade, ajudando a prever o comportamento do sistema e a dimensionar os componentes. Adicionalmente, foi possível concluir que o número de níveis de tensão sintetizados aumenta com a adição de submódulos, melhorando o conteúdo harmônico da corrente de saída.

No Capítulo 5 foi apresentado o protótipo laboratorial desenvolvido, sendo detalhados todos os sistemas utilizados e as placas produzidas. Inicialmente, foi descrito em detalhe todo o hardware que constitui o andar de potência, principalmente a placa de circuito impresso desenvolvida para os conversores CC-CA utilizados nos submódulos, onde foi conseguida uma adaptabilidade que permite a sua reutilização em projetos futuros. Além disso, foi apresentado o circuito de acoplamento com a rede elétrica e os mecanismos de proteção usados para garantir a integridade do sistema e a segurança do utilizador durante a operação. Após a descrição do andar de potência, foi exposto todo o hardware de controlo, desde equipamentos utilizados, como fontes de alimentação e microcontroladores, até às placas desenvolvidas, mostrando os respetivos circuitos elétricos. Quer as placas para o circuito de potência, quer as placas de *docking station* e de *driver* foram concebidas no âmbito desta dissertação. A placa de distribuição e as placas dos sensores foram disponibilizadas pelo GEPE, tendo sido necessário soldar e redimensionar alguns componentes.

Por fim, no Capítulo 6 foi explicado o princípio de funcionamento do controlo implementado e apresentados os resultados experimentais obtidos durante os ensaios efetuados ao sistema

desenvolvido. Primeiramente, foram realizados ensaios individuais as placas do sistema, com o intuito de avaliar individualmente o seu funcionamento. Uma vez validada cada placa, forma exibidos os resultados dos testes em bancada para dois modos de operação distintos, tanto para o MMCC a operar com três submódulos ligados, como para o MMCC com apenas dois submódulos. Inicialmente, foi comprovado o correto funcionamento do algoritmo de EPLL com a apresentação dos resultados da sincronização com a rede elétrica, tanto no início da sincronização, como em regime permanente. De seguida, foram validadas as topologias de comunicação do sistema de controlo modular descentralizado para três conversores em cascata, sendo a topologia em anel capaz de identificar o número de submódulos e atribuir o desfasamento das portadoras e a topologia em barramento não apresentando atrasos significativos nas transmissões. Posteriormente, foram apresentados os resultados obtidos para a injeção de energia na rede elétrica, sendo testadas duas técnicas de controlo de corrente distintas, o controlo preditivo e PI. Ambas as técnicas mostraram resultados similares, a forma de onda da corrente de saída seguiu de perto a referência, mesmo quando se estudou o comportamento dinâmico do sistema, a tensão sintetizada apresentou os sete níveis característicos dos três submódulos em cascata e a potência do sistema foi igualmente dividida pelos submódulos. Repetiu-se o mesmo processo para o sistema de controlo modular descentralizado para dois conversores em cascata, obtendo-se resultados semelhantes e sendo, mais uma vez, comprovado o seu correto funcionamento. As ondas portadoras encontravam-se corretamente desfasadas, a forma de onda da corrente de saída seguia de perto a referência, a tensão sintetizada apresentava os cinco níveis característicos dos dois submódulos em cascata e a potência do sistema estava igualmente dividida pelos submódulos.

Ao longo deste trabalho de dissertação foram surgindo várias dificuldades ao nível de software e hardware. Ao nível da programação, houve alguma dificuldade na configuração do ADC em modo diferencial devido à escassez de material de apoio para este tópico, fazendo com que os testes iniciais a esta funcionalidade demorassem mais tempo que o previsto. Além disso, um dos maiores desafios foi a sincronização de todas as DSC utilizadas, permitindo o correto desfasamento dos sinais de PWM. Para tal, foi utilizada uma funcionalidade da DSC, ajustada através do registo “SYNCOSEL”, que permite sincronizar os sinais de PWM através de um pulso externo. Ao nível de hardware surgiram algumas complexidades inesperadas associadas a

problemas de interferência eletromagnética, problema que piorava com o aumento da potência do sistema e, em alguns casos, levava à perda de comunicação entre o DSC e o computador. Este problema foi atenuado com as medidas adotadas ao longo da dissertação, desde o correto desenvolvimento das PCB, até à utilização de circuitos diferenciais e de pares entrançados para as ligações.

Pode concluir-se que todos os objetivos propostos foram alcançados, sendo desenvolvido com sucesso um sistema de controlo modular descentralizado para conversores de eletrónica de potência em cascata. Além de desafiante, este trabalho contribuiu para o enriquecimento de vários conhecimentos adquiridos ao longo do curso, nomeadamente na revisão bibliográfica de todos os elementos constituintes do sistema, realização de simulações computacionais, programação e estudo do DSC, desenho de placas PCB e desenvolvimento e teste de hardware.

7.2 Sugestões de Trabalho Futuro

O trabalho desenvolvido ao longo desta dissertação permitiu a validação do sistema de controlo modular descentralizado para conversores de eletrónica de potência em cascata com diferentes modos de operação. Todavia, são apresentadas algumas sugestões de trabalho futuro que visam melhorar diversos aspectos:

- Implementar um algoritmo de controlo que permita detetar a falha num submódulo e reconfigure o sistema. Adicionalmente, podem ser criados mecanismos redundantes de proteção com a adição de submódulos, entrando em funcionamento para substituir submódulos danificados.
- Aumentar a potência de operação até à potência nominal, adicionando submódulos ou aumentando a potência dos já existentes e registando a eficiência para uma análise comparativa;
- Substituir, na placa de potência, os sensores de tensão e de corrente por uns mais compactos e eficientes, de modo a ocupar o menor espaço possível e reduzir a dissipação térmica. Um exemplo pode ser o sensor de corrente ACS712 da *Allegro Microsystems*;

- Utilizar diferentes tecnologias de semicondutores, por exemplo os SiC (Silicon Carbide) MOSFET com o encapsulamento de 4 pinos que permitem reduzir as resistências e as perdas de comutação;
- Para além dos circuitos diferenciais implementados, isolar as comunicações utilizadas, aumentando a imunidade ao ruido eletromagnético e a proteção entre os microcontroladores;
- Estudar com maior detalhe os motivos que levaram ao ruido eletromagnético e implementar as respetivas melhorias, como o redimensionamento de componentes e a inclusão de filtros;
- Melhorar o sistema de comunicação, utilizando apenas uma topologia. Isto pode ser alcançado através da aquisição de placas de controlo Piggyback FB1111 da BECKHOFF que permitem comunicação EtherCAT;
- Utilização de diferentes *drivers* mais robustos, com circuitos de proteção integrados, e que permitam acionar semicondutores bipolares.

Relativamente ao último tópico abordado, antes da escolha do integrado ADUM3223, foi selecionado o *driver* ADUM4135 da *Analog Devices*. Estes permitem acionar semicondutores bipolares e unipolares, sendo otimizados para bipolares; oferecem isolamento entre o sinal de entrada e saída; incluem o sistema de *Miller Clamp* que reduz os picos de tensão provocados pelas capacidades de Miller, fornecendo o desligamento robusto do semicondutor; integram um circuito de deteção de dessaturação, proporcionando proteção contra operações do sistema em curto-círcito; facultam informações do estado do *driver* através de saídas dedicadas e permitem reiniciar o dispositivo após uma falha [141]. Apesar destas vantagens, cada *driver* aciona apenas um semicondutor. Foi ainda desenvolvida e testada uma PCB para o ADUM4135, apresentada na Figura 7.1. Apesar dos resultados positivos, as limitações do *stock* deste integrado impossibilitaram a sua aquisição.



Figura 7.1 – Placa de *driver* desenvolvida para o ADUM4135.

Lista de Referências

- [1] M. V. Guesdes, "O Rectificador de Vapor de Mercúrio," FEUP, 1995 [Online]. Available: https://paginas.fe.up.pt/maquel/RH/Res_Hist-24.pdf1 [Accessed: 28-Dec-2022]
- [2] ---, "1947: Invention of the Point-Contact Transistor," Computer History Museum, 2022 [Online]. Available: <https://www.computerhistory.org/siliconengine/invention-of-the-point-contact-transistor/> [Accessed: 28-Dec-2022]
- [3] ---, "1948: Conception of the Junction Transistor," Computer History Museum, 2022 [Online]. Available: <https://www.computerhistory.org/siliconengine/conception-of-the-junction-transistor/> [Accessed: 28-Dec-2022]
- [4] ---, "1940: Discovery of the *p-n* Junction," Computer History Museum, 2022 [Online]. Available: <https://www.computerhistory.org/siliconengine/discovery-of-the-p-n-junction/> [Accessed: 28-Dec-2022]
- [5] ---, "Module 1 Power Semiconductor Devices," version 2 EE IIT, Kharagpur [Online]. Available: [https://web.archive.org/web/20080920222959/http://nptel.iitm.ac.in/courses/Webcourse-contents/IIT%20Kharagpur/Power%20Electronics/PDF/L-1\(SSG\)\(PE\)%20\(\(EE\)NPTEL\).pdf](https://web.archive.org/web/20080920222959/http://nptel.iitm.ac.in/courses/Webcourse-contents/IIT%20Kharagpur/Power%20Electronics/PDF/L-1(SSG)(PE)%20((EE)NPTEL).pdf) [Accessed: 28-Dec-2022]
- [6] Sixing Du, Apparao Dekka, Bin Wu, and Navid Zargari, Modular multilevel converters : analysis, control, and applications. Hoboken: Wiley-ieee Press, 2018.
- [7] G. Pinto, H. Gonçalves, João L. Afonso, "Semicondutores de Potência," Universidade do Minho, Complementos de Eletrónica de Potência, 2020.
- [8] S. Kouro, J. Rodriguez, B. Wu, S. Bernet, and M. Perez, "Powering the Future of Industry: High-Power Adjustable Speed Drive Topologies," IEEE Industry Applications Magazine, vol. 18, no. 4, pp. 26–39, Jul. 2012, doi: 10.1109/mias.2012.2192231.
- [9] L. G. Franquelo, J. Rodriguez, J. I. Leon, S. Kouro, R. Portillo and M. A. M. Prats, "The age of multilevel converters arrives," in IEEE Industrial Electronics Magazine, vol. 2, no. 2, pp. 28-39, June 2008, doi: 10.1109/MIE.2008.923519.
- [10] Z. Zhang, B. Wu, J. Kang and L. Luo, "A Multi-Purpose Balanced Transformer for Railway Traction Applications," in IEEE Transactions on Power Delivery, vol. 24, no. 2, pp. 711-718, April 2009, doi: 10.1109/TPWRD.2008.2008491.
- [11] Shuguang Song, Jinjun Liu, Shaodi Ouyang and Xingxing Chen, "A modular multilevel converter based Railway Power Conditioner for power balance and harmonic compensation in Scott railway traction system," 2016 IEEE 8th International Power Electronics and Motion Control Conference (IPEMC-ECCE Asia), Hefei, China, 2016, pp. 2412-2416, doi: 10.1109/IPEMC.2016.7512675.
- [12] H. Morimoto et al., "Development of railway static power conditioner used at substation for Shinkansen," Proceedings of the Power Conversion Conference-Osaka 2002 (Cat. No.02TH8579), Osaka, Japan, 2002, pp. 1108-1111 vol.3, doi: 10.1109/PCC.2002.998127.
- [13] M. Hagiwara and H. Akagi, "Control and Experiment of Pulsewidth-Modulated Modular Multilevel Converters," in IEEE Transactions on Power Electronics, vol. 24, no. 7, pp. 1737-1746, July 2009, doi: 10.1109/TPEL.2009.2014236.
- [14] Y. Yue, L. Yang, H. Zhao and H. Wang, "A study on modular multilevel converter topology to

- inhibit DC voltage drop," 2017 IEEE International Conference on Mechatronics and Automation (ICMA), Takamatsu, Japan, 2017, pp. 13-17, doi: 10.1109/ICMA.2017.8015780.
- [15] Y. Jiang, J. W. Hu, and H. Wang, "A Precharging Control Strategy for Modular Multilevel Converter Based VSC-HVDC System," *Control Engineering of China*, vol. 21, no. 1, pp. 41-48, January 2014.
 - [16] X. F. Wang, X. H. Wei, and L. H. Ning, "Integration Techniques and Transmission Schemes for Offshore Wind Farms," *Proceedings of The CSEE*, vol. 34,no. 31, pp. 5459-5466, November 2014.
 - [17] N. Nityanand and A. K. Pandey, "HVDC-Based Multilevel Modular Power Converter for Offshore Wind Farms," 2018 International Conference on Power Energy, Environment and Intelligent Control (PEEIC), Greater Noida, India, 2018, pp. 540-544, doi: 10.1109/PEEIC.2018.8665408.
 - [18] A. Parastar, Y. C. Kang and J. -K. Seok, "Multilevel Modular DC/DC Power Converter for High-Voltage DC-Connected Offshore Wind Energy Applications," in *IEEE Transactions on Industrial Electronics*, vol. 62, no. 5, pp. 2879-2890, May 2015, doi: 10.1109/TIE.2014.2363818.
 - [19] A. Parastar and J. -K. Seok, "High power step-up modular resonant DC/DC converter for offshore wind energy systems," 2014 IEEE Energy Conversion Congress and Exposition (ECCE), Pittsburgh, PA, USA, 2014, pp. 3341-3348, doi: 10.1109/ECCE.2014.6953854.
 - [20] A. Gandomkar, A. Parastar and J. -K. Seok, "High-Power Multilevel Step-Up DC/DC Converter for Offshore Wind Energy Systems," in *IEEE Transactions on Industrial Electronics*, vol. 63, no. 12, pp. 7574-7585, Dec. 2016, doi: 10.1109/TIE.2016.2594050.
 - [21] H. Bayat and A. Yazdani, "A Hybrid MMC-Based Photovoltaic and Battery Energy Storage System," *IEEE Power and Energy Technology Systems Journal*, vol. 6, no. 1, pp. 32–40, Mar. 2019, doi: 10.1109/jpets.2019.2892418.
 - [22] H. Bayat and A. Yazdani, "A Power Mismatch Elimination Strategy for an MMC-Based Photovoltaic System," in *IEEE Transactions on Energy Conversion*, vol. 33, no. 3, pp. 1519-1528, Sept. 2018, doi: 10.1109/TEC.2018.2819982.
 - [23] M. Alsadah and F. Mancilla-David, "Modeling and control of grid-connected photovoltaic power plants utilizing a simplified model of the modular multilevel converter," 2014 North American Power Symposium (NAPS), Pullman, WA, USA, 2014, pp. 1-6, doi: 10.1109/NAPS.2014.6965433.
 - [24] J. Echeverría, S. Kouro, M. Pérez and H. Abu-rub, "Multi-modular cascaded DC-DC converter for HVDC grid connection of large-scale photovoltaic power systems," *IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society*, Vienna, Austria, 2013, pp. 6999-7005, doi: 10.1109/IECON.2013.6700293.
 - [25] J. Feng, W. Q. Chu, Z. Zhang and Z. Q. Zhu, "Power Electronic Transformer-Based Railway Traction Systems: Challenges and Opportunities," in *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 5, no. 3, pp. 1237-1253, Sept. 2017, doi: 10.1109/JESTPE.2017.2685464.
 - [26] H. Stemmler, "High-power industrial drives," in *Proceedings of the IEEE*, vol. 82, no. 8, pp. 1266-1286, Aug. 1994, doi: 10.1109/5.301688.
 - [27] L. M. Tolbert, Fang Zheng Peng and T. G. Habetler, "Multilevel converters for large electric drives," in *IEEE Transactions on Industry Applications*, vol. 35, no. 1, pp. 36-44, Jan.-Feb. 1999, doi: 10.1109/28.740843.
 - [28] J. Venkataramaiah, Y. Suresh, and A. K. Panda, "A review on symmetric, asymmetric, hybrid and single DC sources based multilevel inverter topologies," *Renewable and Sustainable Energy Reviews*, vol. 76, pp. 788–812, Sep. 2017, doi: 10.1016/j.rser.2017.03.066.
 - [29] M. A. Perez, S. Bernet, J. Rodriguez, S. Kouro and R. Lizana, "Circuit Topologies, Modeling,

- Control Schemes, and Applications of Modular Multilevel Converters," in IEEE Transactions on Power Electronics, vol. 30, no. 1, pp. 4-17, Jan. 2015, doi: 10.1109/TPEL.2014.2310127.
- [30] M. J. Mojibian and M. Tavakoli Bina, "Classification of multilevel converters with a modular reduced structure: implementing a prominent 31-level 5 kVA class B converter," IET Power Electronics, vol. 8, no. 1, pp. 20–32, Jan. 2015, doi: 10.1049/iet-pel.2013.0872.
- [31] Kamran Sharifabadi, Lennart Harnefors, Hans Peter Nee, Staffan Norrga, and Remus Teodorescu, Design, control, and application of modular multilevel converters for HVDC transmission systems. Chichester, West Sussex, United Kingdom: Ieee Press, Wiley, 2016.
- [32] J. Rodriguez, S. Bernet, B. Wu, J. O. Pontt and S. Kouro, "Multilevel Voltage-Source-Converter Topologies for Industrial Medium-Voltage Drives," in IEEE Transactions on Industrial Electronics, vol. 54, no. 6, pp. 2930-2945, Dec. 2007, doi: 10.1109/TIE.2007.907044.
- [33] H. Abu-Rub, J. Holtz, J. Rodriguez and G. Baoming, "Medium-Voltage Multilevel Converters—State of the Art, Challenges, and Requirements in Industrial Applications," in IEEE Transactions on Industrial Electronics, vol. 57, no. 8, pp. 2581-2596, Aug. 2010, doi: 10.1109/TIE.2010.2043039.
- [34] S. Kouro et al., "Recent Advances and Industrial Applications of Multilevel Converters," in IEEE Transactions on Industrial Electronics, vol. 57, no. 8, pp. 2553-2580, Aug. 2010, doi: 10.1109/TIE.2010.2049719.
- [35] M. Malinowski, K. Gopakumar, J. Rodriguez and M. A. Pérez, "A Survey on Cascaded Multilevel Inverters," in IEEE Transactions on Industrial Electronics, vol. 57, no. 7, pp. 2197-2206, July 2010, doi: 10.1109/TIE.2009.2030767.
- [36] L. Harnefors, A. Antonopoulos, S. Norrga, L. Angquist and H. -P. Nee, "Dynamic Analysis of Modular Multilevel Converters," in IEEE Transactions on Industrial Electronics, vol. 60, no. 7, pp. 2526-2537, July 2013, doi: 10.1109/TIE.2012.2194974.
- [37] Q. Song, W. Liu, X. Li, H. Rao, S. Xu and L. Li, "A Steady-State Analysis Method for a Modular Multilevel Converter," in IEEE Transactions on Power Electronics, vol. 28, no. 8, pp. 3702-3713, Aug. 2013, doi: 10.1109/TPEL.2012.2227818.
- [38] Y. -m. Park et al., "A Simple and Reliable PWM Synchronization & Phase-Shift Method for Cascaded H-Bridge Multilevel Inverters based on a Standard Serial Communication Protocol," Conference Record of the 2006 IEEE Industry Applications Conference Forty-First IAS Annual Meeting, Tampa, FL, USA, 2006, pp. 988-994, doi: 10.1109/IAS.2006.256645.
- [39] P. Dan Burlacu, L. Mathe and R. Teodorescu, "Synchronization of the distributed PWM carrier waves for modular multilevel converters," 2014 International Conference on Optimization of Electrical and Electronic Equipment (OPTIM), Bran, Romania, 2014, pp. 553-559, doi: 10.1109/OPTIM.2014.6851001.
- [40] H. -J. Knaak, "Modular multilevel converters and HVDC/FACTS: A success story," Proceedings of the 2011 14th European Conference on Power Electronics and Applications, Birmingham, UK, 2011, pp. 1-6.
- [41] U U. N. Gnanarathna, S. K. Chaudhary, A. M. Gole and R. Teodorescu, "Modular multi-level converter based HVDC system for grid connection of offshore wind power plant," 9th IET International Conference on AC and DC Power Transmission (ACDC 2010), London, 2010, pp. 1-5, doi: 10.1049/cp.2010.0984.
- [42] T. P. Corrêa, L. Almeida and F. J. Rodriguez, "Communication aspects in the distributed control architecture of a modular multilevel converter," 2018 IEEE International Conference on Industrial Technology (ICIT), Lyon, France, 2018, pp. 640-645, doi: 10.1109/ICIT.2018.8352253.

- [43] C. L. Toh and L. E. Norum, "A performance analysis of three potential control network for monitoring and control in Power Electronics converter," 2012 IEEE International Conference on Industrial Technology, Athens, Greece, 2012, pp. 224-229, doi: 10.1109/ICIT.2012.6209942.
- [44] C.C. L. Toh and L. E. Norum, "Implementation of high speed control network with fail-safe control and communication cable redundancy in modular multilevel converter," 2013 15th European Conference on Power Electronics and Applications (EPE), Lille, France, 2013, pp. 1-10, doi: 10.1109/EPE.2013.6631825.
- [45] C. L. Toh and L. E. Norum, "A high speed control network synchronization jitter evaluation for embedded monitoring and control in modular multilevel converter," 2013 IEEE Grenoble Conference, Grenoble, France, 2013, pp. 1-6, doi: 10.1109/PTC.2013.6652174.
- [46] T. Laakkonen, "Distributed control architecture of power electronics building-block-based frequency converters," Ph.D. dissertation, Lappeenranta University of Technology, 2010.
- [47] A. The, C. Bruening and S. Dieckerhoff, "CAN-based distributed control of a MMC optimized for low number of submodules," 2015 IEEE Energy Conversion Congress and Exposition (ECCE), Montreal, QC, Canada, 2015, pp. 1590-1594, doi: 10.1109/ECCE.2015.7309884.
- [48] S. Huang, R. Teodorescu and L. Mathe, "Analysis of communication based distributed control of MMC for HVDC," 2013 15th European Conference on Power Electronics and Applications (EPE), Lille, France, 2013, pp. 1-10, doi: 10.1109/EPE.2013.6634711.
- [49] C. Carstensen, R. Christen, H. Vollenweider, R. Stark and J. Biela, "A converter control field bus protocol for power electronic systems with a synchronization accuracy of $\pm 5\text{ns}$," 2015 17th European Conference on Power Electronics and Applications (EPE'15 ECCE-Europe), Geneva, Switzerland, 2015, pp. 1-10, doi: 10.1109/EPE.2015.7309146.
- [50] L. -A. Grégoire, Weihua Wang, S. I. Seleme and M. Fadel, "High reliability observers for modular multilevel converter capacitor voltage evaluation," 2016 IEEE 8th International Power Electronics and Motion Control Conference (IPEMC-ECCE Asia), Hefei, China, 2016, pp. 2332-2336, doi: 10.1109/IPEMC.2016.7512661.
- [51] S. Huang, "Distributed modulation and control of modular multilevel converter for HVDC application," Aalborg University, 2013.
- [52] R. Schollmeier, "A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications," Proceedings First International Conference on Peer-to-Peer Computing, Linköping, Sweden, 2001, pp. 101-102, doi: 10.1109/P2P.2001.990434.
- [53] M. Kubiš and P. Beňo, "Realization of communication via the CAN bus," Transportation Research Procedia, vol. 40, pp. 332–337, 2019, doi: 10.1016/j.trpro.2019.07.049.
- [54] F. Davik, M. Yilmaz, S. Gjessing and N. Uzun, "IEEE 802.17 resilient packet ring tutorial," in IEEE Communications Magazine, vol. 42, no. 3, pp. 112-118, March 2004, doi: 10.1109/MCOM.2004.1273782.
- [55] E. L. Talon, S. Gavin, D. Siemaszko, F. Biya-Motto, B. Z. Essimbi and M. Carpita, "Design and implementation of a multi-dsp based digital control system architecture for modular multilevel converters," 2016 IEEE International Power Electronics and Motion Control Conference (PEMC), Varna, Bulgaria, 2016, pp. 1182-1187, doi: 10.1109/EPEPEMC.2016.7752163.
- [56] B. Ciftci, J. Gross, S. Norrga and H. -P. Nee, "Simple Distributed Control for Modular Multilevel Converters," 2019 21st European Conference on Power Electronics and Applications (EPE '19 ECCE Europe), Genova, Italy, 2019, pp. P.1-P.10, doi: 10.23919/EPE.2019.8915488.
- [57] M. A. Parker, L. Ran and S. J. Finney, "Distributed Control of a Fault-Tolerant Modular Multilevel Inverter for Direct-Drive Wind Turbine Grid Interfacing," in IEEE Transactions on Industrial

- Electronics, vol. 60, no. 2, pp. 509-522, Feb. 2013, doi: 10.1109/TIE.2012.2186774.
- [58] S. Rietmann, S. Fuchs, A. Hillers and J. Biela, "Field Bus for Data Exchange and Control of Modular Power Electronic Systems with High Synchronisation Accuracy," 2018 International Power Electronics Conference (IPEC-Niigata 2018-ECCE Asia), Niigata, Japan, 2018, pp. 2301-2308, doi: 10.23919/IPEC.2018.8507631.
- [59] ---, "Hardware Datasheet: FB1111," Beckhoff, version 2.2, Dec. 2011 [Online]. Available: https://download.beckhoff.com/download/Document/io/ethercat-development-products/beckhoff_fb1111-014x_v22.pdf [Accessed: 28-Dec-2022]
- [60] Zhong Yubiao, and Xu Lianying. Realization of Synchronous Motion Control of Servo Gear Coupling Based on EtherCAT Bus Control[J]. Information Communication, 2018(3).
- [61] C. -H. Park, B. B. Negesse, J. -M. Kim and C. -K. Kim, "Back-to-Back 31 Level Modular Multilevel Converter with EtherCAT Communication," 2019 IEEE Energy Conversion Congress and Exposition (ECCE), Baltimore, MD, USA, 2019, pp. 1032-1039, doi: 10.1109/ECCE.2019.8913080.
- [62] J. -H. Fey, F. Hinrichsen, G. Carstens and R. Mallwitz, "Development of a Modular Multilevel Converter Demonstrator with EtherCAT Communication," 2019 IEEE 13th International Conference on Compatibility, Power Electronics and Power Engineering (CPE-POWERENG), Sonderborg, Denmark, 2019, pp. 1-6, doi: 10.1109/CPE.2019.8862424.
- [63] P. Dan Burlacu, L. Mathe, M. Rejas, H. Pereira, A. Sangwongwanich and R. Teodorescu, "Implementation of fault tolerant control for modular multilevel converter using EtherCAT communication," 2015 IEEE International Conference on Industrial Technology (ICIT), Seville, Spain, 2015, pp. 3064-3071, doi: 10.1109/ICIT.2015.7125551.
- [64] S. Debnath, J. Qin, B. Bahrani, M. Saeedifard and P. Barbosa, "Operation, Control, and Applications of the Modular Multilevel Converter: A Review," in IEEE Transactions on Power Electronics, vol. 30, no. 1, pp. 37-53, Jan. 2015, doi: 10.1109/TPEL.2014.2309937.
- [65] A. Nami, J. Liang, F. Dijkhuizen and G. D. Demetriadis, "Modular Multilevel Converters for HVDC Applications: Review on Converter Cells and Functionalities," in IEEE Transactions on Power Electronics, vol. 30, no. 1, pp. 18-36, Jan. 2015, doi: 10.1109/TPEL.2014.2327641.
- [66] M. H. Rashid, Power electronics handbook. Butterworth-Heinemann, 2017.
- [67] L. A. M. Barros, A. P. Martins, and J. G. Pinto, "A Comprehensive Review on Modular Multilevel Converters, Submodule Topologies, and Modulation Techniques," Energies, vol. 15, no. 3, p. 1078, Feb. 2022, doi: 10.3390/en15031078.
- [68] A. L. Batschauer, S. A. Mussa and M. L. Heldwein, "Three-Phase Hybrid Multilevel Inverter Based on Half-Bridge Modules," in IEEE Transactions on Industrial Electronics, vol. 59, no. 2, pp. 668-678, Feb. 2012, doi: 10.1109/TIE.2011.2158039.
- [69] H. Vahedi, K. Al-Haddad, P. -A. Labb   and S. Rahmani, "Cascaded multilevel inverter with multicarrier PWM technique and voltage balancing feature," 2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE), Istanbul, Turkey, 2014, pp. 2155-2160, doi: 10.1109/ISIE.2014.6864951.
- [70] R. Marquardt, "Modular Multilevel Converter: An universal concept for HVDC-Networks and extended DC-Bus-applications," The 2010 International Power Electronics Conference - ECCE ASIA -, Sapporo, Japan, 2010, pp. 502-507, doi: 10.1109/IPEC.2010.5544594.
- [71] M. L  pez, A. Rodr  iguez, E. Blanco, M. Saeed, A. Mart  nez, and F. Briz, "Design and implementation of the control of an MMCC-based solid state transformer," in 2015 IEEE 13th International Conference on Industrial Informatics (INDIN), 2015, pp. 1583-1590.

- [72] J. -J. Jung, S. Cui, J. -H. Lee and S. -K. Sul, "A New Topology of Multilevel VSC Converter for a Hybrid HVDC Transmission System," in IEEE Transactions on Power Electronics, vol. 32, no. 6, pp. 4199-4209, June 2017, doi: 10.1109/TPEL.2016.2598368.
- [73] M. Vijeh, M. Rezanejad, E. Samadai and K. Bertilsson, "A General Review of Multilevel Inverters Based on Main Submodules: Structural Point of View," in IEEE Transactions on Power Electronics, vol. 34, no. 10, pp. 9479-9502, Oct. 2019, doi: 10.1109/TPEL.2018.2890649.
- [74] Jingsheng Liao, K. Corzine and M. Ferdowsi, "A new control method for single-dc-source cascaded H-bridge multilevel converters using phase-shift modulation," 2008 Twenty-Third Annual IEEE Applied Power Electronics Conference and Exposition, Austin, TX, 2008, pp. 886-890, doi: 10.1109/APEC.2008.4522825.
- [75] W. Song and A. Q. Huang, "Fault-Tolerant Design and Control Strategy for Cascaded H-Bridge Multilevel Converter-Based STATCOM," in IEEE Transactions on Industrial Electronics, vol. 57, no. 8, pp. 2700-2708, Aug. 2010, doi: 10.1109/TIE.2009.2036019.
- [76] Y. S. Lai and F. S. Shyu, "New topology for hybrid multilevel inverter," 2002 International Conference on Power Electronics, Machines and Drives (Conf. Publ. No. 487), Santa Fe, NM, USA, 2002, pp. 211-216, doi: 10.1049/cp:20020116.
- [77] I. Staudt, "Application Note AN-11001: 3L NPC & TNPC Topology," Semikron, Set 2015.
- [78] G. Pereira da Silva, F. Trentini, V. T. Odaguiri, M. V. Bressan, M. L. Heldwein and A. L. Batschauer, "Hybrid three-phase multilevel inverter based ON NPC cascaded to half-bridge cells," 2013 Brazilian Power Electronics Conference, Gramado, Brazil, 2013, pp. 72-78, doi: 10.1109/COBEP.2013.6785097.
- [79] M. Carpaneto, M. Marchesoni and L. Vaccaro, "A New Cascaded Multilevel Converter Based on NPC Cells," 2007 IEEE International Symposium on Industrial Electronics, Vigo, Spain, 2007, pp. 1033-1038, doi: 10.1109/ISIE.2007.4374740.
- [80] A. Nami and J. Adabi, "A new T-type NPC-based submodule for modular multilevel cascaded converters," The 5th Annual International Power Electronics, Drive Systems and Technologies Conference (PEDSTC 2014), Tehran, Iran, 2014, pp. 137-142, doi: 10.1109/PEDSTC.2014.6799359.
- [81] J. Rodriguez, Jih-Sheng Lai and Fang Zheng Peng, "Multilevel inverters: a survey of topologies, controls, and applications," in IEEE Transactions on Industrial Electronics, vol. 49, no. 4, pp. 724-738, Aug. 2002, doi: 10.1109/TIE.2002.801052.
- [82] A. António-Ferreira, C. Collados-Rodríguez, and O. Gomis-Bellmunt, "Modulation techniques applied to medium voltage modular multilevel converters for renewable energy integration: A review," Electric Power Systems Research, vol. 155, pp. 21–39, Feb. 2018, doi: 10.1016/j.epsr.2017.08.015.
- [83] B. P. McGrath and D. G. Holmes, "Multicarrier PWM strategies for multilevel inverters," in IEEE Transactions on Industrial Electronics, vol. 49, no. 4, pp. 858-867, Aug. 2002, doi: 10.1109/TIE.2002.801073.
- [84] G. Carrara, S. Gardella, M. Marchesoni, R. Salutari and G. Scututto, "A new multilevel PWM method: a theoretical analysis," in IEEE Transactions on Power Electronics, vol. 7, no. 3, pp. 497-505, July 1992, doi: 10.1109/63.145137.
- [85] J. Ananthu and V. Srikanth, "Voltage balancing of modular multilevel converter for an induction motor drive," 2017 International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), Kerala, India, 2017, pp. 699-703, doi: 10.1109/ICICICT1.2017.8342649.

- [86] H. Liu, K. Ma and F. Blaabjerg, "Device loading and efficiency of Modular Multilevel Converter under various Modulation strategies," 2016 IEEE 7th International Symposium on Power Electronics for Distributed Generation Systems (PEDG), Vancouver, BC, Canada, 2016, pp. 1-7, doi: 10.1109/PEDG.2016.7527104.
- [87] A. U. Lawan and H. M. Abbas, "Level shifted PWMs comparison for a 5-level modular multilevel converter (MMC) topology inverter," 2015 IEEE Conference on Sustainable Utilization And Development In Engineering and Technology (CSUDET), Selangor, Malaysia, 2015, pp. 1-6, doi: 10.1109/CSUDET.2015.7446224.
- [88] F. Deng and Z. Chen, "Elimination of DC-Link Current Ripple for Modular Multilevel Converters With Capacitor Voltage-Balancing Pulse-Shifted Carrier PWM," in IEEE Transactions on Power Electronics, vol. 30, no. 1, pp. 284-296, Jan. 2015, doi: 10.1109/TPEL.2014.2322913.
- [89] J. Holtz, W. Lotzkat and K. . -H. Werner, "A high-power multitransistor-inverter uninterruptable power supply system," in IEEE Transactions on Power Electronics, vol. 3, no. 3, pp. 278-285, July 1988, doi: 10.1109/63.17945.
- [90] K. B. Shah and H. Chandwani, "Reduced switching-frequency voltage balancing technique for modular multilevel converters," 2017 International Conference on Intelligent Sustainable Systems (ICISS), Palladam, India, 2017, pp. 289-294, doi: 10.1109/ISS1.2017.8389417.
- [91] W. Liu, X. Chen, K. Zhang, and J. Xiong, "Simplified model and submodule capacitor voltage balancing of single-phase AC/AC modular multilevel converter for railway traction purpose," IET Power Electronics, vol. 9, no. 5, pp. 951–959, Apr. 2016, doi: 10.1049/iet-pel.2015.0120.
- [92] L. A. M. de Barros, "Desenvolvimento de um microinversor com armazenamento local de energia para aplicações solares fotovoltaicas," M.S. Thesis, Dep. of Industrial Electronics (DEI), Shc. of Engineering, Univ. Minho, 2016 [Online]. Available: <https://hdl.handle.net/1822/46592>
- [93] T. J. da C. Sousa, "Filtro ativo de potência paralelo monofásico com conversor CC-CC bidirecional para operação como UPS," M.S. Thesis, Dep. of Industrial Electronics (DEI), Shc. of Engineering, Univ. Minho, 2017 [Online]. Available: <https://hdl.handle.net/1822/54769>
- [94] S. Buso, L. Malesani and P. Mattavelli, "Comparison of current control techniques for active filter applications," in IEEE Transactions on Industrial Electronics, vol. 45, no. 5, pp. 722-729, Oct. 1998, doi: 10.1109/41.720328.
- [95] L. Yunbo, X. Yonghai and X. Yunfei, "A MMC hysteresis current control method based on current slope," IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society, Beijing, China, 2017, pp. 577-582, doi: 10.1109/IECON.2017.8216101.
- [96] E. Talon Louokdom et al., "Small-Scale Modular Multilevel Converter for Multi-Terminal DC Networks Applications: System Control Validation," Energies, vol. 11, no. 7, p. 1690, Jun. 2018, doi: 10.3390/en11071690.
- [97] M. Barghi Latran and A. Teke, "Investigation of multilevel multifunctional grid connected inverter topologies and control strategies used in photovoltaic systems," Renewable and Sustainable Energy Reviews, vol. 42, pp. 361–376, Feb. 2015, doi: 10.1016/j.rser.2014.10.030.
- [98] P. Cortes, M. P. Kazmierkowski, R. M. Kennel, D. E. Quevedo and J. Rodriguez, "Predictive Control in Power Electronics and Drives," in IEEE Transactions on Industrial Electronics, vol. 55, no. 12, pp. 4312-4324, Dec. 2008, doi: 10.1109/TIE.2008.2007480.
- [99] B. Exposto, R. Rodrigues, J. G. Pinto, V. Monteiro, D. Pedrosa and J. L. Afonso, "Predictive control of a current-source inverter for solar photovoltaic grid interface," 2015 9th International Conference on Compatibility and Power Electronics (CPE), Costa da Caparica, Portugal, 2015, pp. 113-118, doi: 10.1109/CPE.2015.7231058.

- [100] Y. Wang, W. Cong, M. Li, N. Li, M. Cao and W. Lei, "Model predictive control of modular multilevel converter with reduced computational load," 2014 IEEE Applied Power Electronics Conference and Exposition - APEC 2014, Fort Worth, TX, USA, 2014, pp. 1776-1779, doi: 10.1109/APEC.2014.6803546.
- [101] K. Xu, S. Xie, X. Wang, B. Zhang and S. Bian, "Model Predictive-based Fault-tolerant Control for Modular Multilevel Converters Without Redundant Submodules," 2018 IEEE International Power Electronics and Application Conference and Exposition (PEAC), Shenzhen, China, 2018, pp. 1-5, doi: 10.1109/PEAC.2018.8590218.
- [102] E. Solas, G. Abad, J. A. Barrena, S. Arunetxeta, A. Cárcar and L. Zajac, "Modular Multilevel Converter With Different Submodule Concepts—Part I: Capacitor Voltage Balancing Method," in IEEE Transactions on Industrial Electronics, vol. 60, no. 10, pp. 4525-4535, Oct. 2013, doi: 10.1109/TIE.2012.2210378.
- [103] S. Rohner, S. Bernet, M. Hiller and R. Sommer, "Modulation, Losses, and Semiconductor Requirements of Modular Multilevel Converters," in IEEE Transactions on Industrial Electronics, vol. 57, no. 8, pp. 2633-2642, Aug. 2010, doi: 10.1109/TIE.2009.2031187.
- [104] M. Saeedifard and R. Iravani, "Dynamic Performance of a Modular Multilevel Back-to-Back HVDC System," in IEEE Transactions on Power Delivery, vol. 25, no. 4, pp. 2903-2912, Oct. 2010, doi: 10.1109/TPWRD.2010.2050787.
- [105] S. Rohner, S. Bernet, M. Hiller and R. Sommer, "Modelling, simulation and analysis of a Modular Multilevel Converter for medium voltage applications," 2010 IEEE International Conference on Industrial Technology, Via del Mar, Chile, 2010, pp. 775-782, doi: 10.1109/ICIT.2010.5472634.
- [106] F. Deng and Z. Chen, "A Control Method for Voltage Balancing in Modular Multilevel Converters," in IEEE Transactions on Power Electronics, vol. 29, no. 1, pp. 66-76, Jan. 2014, doi: 10.1109/TPEL.2013.2251426.
- [107] J. Qin and M. Saeedifard, "Predictive Control of a Modular Multilevel Converter for a Back-to-Back HVDC System," in IEEE Transactions on Power Delivery, vol. 27, no. 3, pp. 1538-1547, July 2012, doi: 10.1109/TPWRD.2012.2191577.
- [108] B. Li, R. Yang, D. Xu, G. Wang, W. Wang and D. Xu, "Analysis of the Phase-Shifted Carrier Modulation for Modular Multilevel Converters," in IEEE Transactions on Power Electronics, vol. 30, no. 1, pp. 297-310, Jan. 2015, doi: 10.1109/TPEL.2014.2299802.
- [109] J. Mei, K. Shen, B. Xiao, L. M. Tolbert and J. Zheng, "A New Selective Loop Bias Mapping Phase Disposition PWM With Dynamic Voltage Balance Capability for Modular Multilevel Converter," in IEEE Transactions on Industrial Electronics, vol. 61, no. 2, pp. 798-807, Feb. 2014, doi: 10.1109/TIE.2013.2253069.
- [110] H. Akagi, E. H. Watanabe, and AredeS.M., Instantaneous power theory and applications to power conditioning. Hoboken, New Jersey: Wiley : IEEE Press, 2017.
- [111] Y. Han, M. Luo, X. Zhao, J. M. Guerrero and L. Xu, "Comparative Performance Evaluation of Orthogonal-Signal-Generators-Based Single-Phase PLL Algorithms—A Survey," in IEEE Transactions on Power Electronics, vol. 31, no. 5, pp. 3932-3944, May 2016, doi: 10.1109/TPEL.2015.2466631.
- [112] O. Vainio, S. J. Ovaska and M. Polla, "Adaptive filtering using multiplicative general parameters for zero-crossing detection," in IEEE Transactions on Industrial Electronics, vol. 50, no. 6, pp. 1340-1342, Dec. 2003, doi: 10.1109/TIE.2003.819565.
- [113] M. A. Perez, J. R. Espinoza, L. A. Moran, M. A. Torres and E. A. Araya, "A Robust Phase-Locked Loop Algorithm to Synchronize Static-Power Converters With Polluted AC Systems," in IEEE Transactions on Industrial Electronics, vol. 55, no. 5, pp. 2185-2192, May 2008, doi:

- 10.1109/TIE.2008.918638.
- [114] P. K. Dash, G. Panda, A. K. Pradhan, A. Routray and B. Duttagupta, "An extended complex Kalman filter for frequency measurement of distorted signals," 2000 IEEE Power Engineering Society Winter Meeting. Conference Proceedings (Cat. No.00CH37077), Singapore, 2000, pp. 1569-1574 vol.3, doi: 10.1109/PESW.2000.847576.
- [115] B. Chitti Babu, K. Sridharan, E. Rosolowski, and Z. Leonowicz, "Analysis of SDFT based phase detection system for grid synchronization of distributed generation systems," *Engineering Science and Technology, an International Journal*, vol. 17, no. 4, pp. 270–278, Dec. 2014, doi: 10.1016/j.jestch.2014.07.005.
- [116] G. Simon, R. Pintelon, L. Sujbert and J. Schoukens, "An efficient nonlinear least square multisine fitting algorithm," in *IEEE Transactions on Instrumentation and Measurement*, vol. 51, no. 4, pp. 750-755, Aug. 2002, doi: 10.1109/TIM.2002.803304.
- [117] G. Yin, L. Guo and X. Li, "An Amplitude Adaptive Notch Filter for Grid Signal Processing," in *IEEE Transactions on Power Electronics*, vol. 28, no. 6, pp. 2638-2641, June 2013, doi: 10.1109/TPEL.2012.2226752.
- [118] N. F. Guerrero-Rodríguez, A. B. Rey-Boué, L. C. Herrero-de Lucas, and F. Martinez-Rodrigo, "Control and synchronization algorithms for a grid-connected photovoltaic system under harmonic distortions, frequency variations and unbalances," *Renewable Energy*, vol. 80, pp. 380–395, Aug. 2015, doi: 10.1016/j.renene.2015.02.027.
- [119] S. Golestan, M. Monfared and F. D. Freijedo, "Design-Oriented Study of Advanced Synchronous Reference Frame Phase-Locked Loops," in *IEEE Transactions on Power Electronics*, vol. 28, no. 2, pp. 765-778, Feb. 2013, doi: 10.1109/TPEL.2012.2204276.
- [120] M. Karimi-Ghartemani, "Linear and Pseudolinear Enhanced Phased-Locked Loop (EPLL) Structures," in *IEEE Transactions on Industrial Electronics*, vol. 61, no. 3, pp. 1464-1474, March 2014, doi: 10.1109/TIE.2013.2261035.
- [121] Masoud Karimi-Ghartema, *Enhanced Phase-Locked Loop Structures for Power and Energy Applications*. John Wiley & Sons, 2014.
- [122] ---, "Datasheet: Polar2™ HiperFET™ IXFH42N50P2," *IXYS Corporation*, Jan. 2011 [Online]. Available: https://br.mouser.com/datasheet/2/240/Littelfuse_Discrete_MOSFETs_N-Channel_HiPerFETs_IX-1856071.pdf [Accessed: 28-Dec-2022]
- [123] ---, "N-Channel MOSFETs with Fast Intrinsic Diodes (HiPerFETs)," *Littelfuse*, 2022 [Online]. Available: <https://www.littelfuse.com/products/power-semiconductors/discrete-mosfets/n-channel-hiperfets.aspx> [Accessed: 28-Dec-2022]
- [124] ---, "Datasheet: VR Series Heatsink," *Ohmite*, 2022 [Online]. Available: https://www.mouser.com/datasheet/2/303/Ohmite_sink_vr-1928250.pdf [Accessed: 28-Dec-2022]
- [125] ---, "Application Note: Snubber circuit design methods," *ROHM Semiconductor*, 2020 [Online]. Available: https://fscdn.rohm.com/en/products/databook/applinote/discrete/sic/mosfet/sic-mos_snubber_circuit_design_an-e.pdf [Accessed: 28-Dec-2022]
- [126] ---, "Datasheet: TXL 060-070 Series, 50-70 Watt," *TracoPower*, 2021 [Online]. Available: https://www.tracopower.com/sites/default/files/products/datasheets/txl060_070_datasheet.pdf [Accessed: 28-Dec-2022]
- [127] ---, "User's Guide: LAUNCHXL-F28379D Overview," *Texas Instruments*, 2016 [Online]. Available: <https://www.ti.com/lit/ug/sprui77c/sprui77c.pdf?ts=1673333625198> [Accessed: 28-Dec-2022]

- [128] ---, "TMS320F2837xD Dual-Core Microcontrollers," *Texas Instruments*, 2013 [Online]. Available: https://www.ti.com/lit/ds/sprs880o/sprs880o.pdf?ts=1673357849240&ref_url=https%253A%252F%252Fwww.ti.com%252Ftool%252FLAUNCHXL-F28379D [Accessed: 28-Dec-2022]
- [129] ---, "2-Layer PCB vs. 4-Layer Printed Circuit Boards (PCB)," *GerberLabs*, 2020 [Online]. Available: <https://www.gerberlabs.com/guides/2-layer-vs-4-layer-printed-circuit-boards/> [Accessed: 28-Dec-2022]
- [130] S. Arar, "Common PCB Stackups for a Four-Layer Board," Technical Article, *All About Circuits*, 2019 [Online]. Available: <https://www.allaboutcircuits.com/technical-articles/common-layer-stack-ups-for-a-four-layer-board/> [Accessed: 28-Dec-2022]
- [131] ---, "Datasheet: LTC1992 Family (Fully Differential, Low Power Amplifiers)," *Linear Technology*, 2005 [Online]. Available: <https://eu.mouser.com/datasheet/2/609/1992fb-1270390.pdf> [Accessed: 28-Dec-2022]
- [132] ---, "Datasheet: NE555 Precision Timers," *Texas Instruments*, 2014 [Online]. Available: <https://www.ti.com/lit/ds/symlink/ne555.pdf> [Accessed: 28-Dec-2022]
- [133] C. Pinkle, "The Why and How of Differential Signaling," Technical Article, *All About Circuits*, 2016 [Online]. Available: <https://www.allaboutcircuits.com/technical-articles/the-why-and-how-of-differential-signaling/> [Accessed: 28-Dec-2022]
- [134] ---, "General Description: Quad LVDS Line Receivers with Integrated Termination and Flow-Through Pinout," *Maxim Integrated Products*, 2001 [Online]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/MAX9121-MAX9122.pdf> [Accessed: 28-Dec-2022]
- [135] ---, "General Description Quad LVDS Line Driver with Flow-Through Pinout," *Maxim Integrated Products*, 2001 [Online]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/MAX9123.pdf> [Accessed: 28-Dec-2022]
- [136] ---, "Datasheet: ADuM3223/ADuM4223 (Isolated Precision Half-Bridge Driver, 4 A Output)" *Analog Devices*, 2012 [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/adum3223_4223.pdf [Accessed: 28-Dec-2022]
- [137] ---, "Datasheet: MEV1 Series (3kVDC Isolated 1W Single & Dual Output DC-DC Converters)," *Murata Power Solutions*, 2020 [Online]. Available: https://eu.mouser.com/datasheet/2/281/1/kdc_mev-2940990.pdf [Accessed: 28-Dec-2022]
- [138] L. Balogh, "Application Report: Fundamentals of MOSFET and IGBT Gate Driver Circuits," *Texas Instruments*, 2017 [Online]. Available: <https://www.ti.com/lit/ml/slua618a/slua618a.pdf?ts=1673319846734> [Accessed: 28-Dec-2022]
- [139] ---, "Datasheet: Voltage Transducer LV 25-P," *LEM*, 2014 [Online]. Available: https://www.lem.com/sites/default/files/products_datasheets/lv_25-p.pdf [Accessed: 28-Dec-2022]
- [140] ---, "Datasheet: Current Transducer LA 55-P," *LEM*, 2018 [Online]. Available: https://www.lem.com/sites/default/files/products_datasheets/la_55-p_e.pdf [Accessed: 28-Dec-2022]
- [141] ---, "Datasheet: ADum4135 (Single-/Dual-Supply, High Voltage Isolated IGBT Gate Driver with Miller Clamp)," *Analog Devices*, 2015 [Online]. Available: <https://www.analog.com/media/en/technical-documentation/data-sheets/adum4135.pdf> [Accessed: 28-Dec-2022]

Apêndice

Apêndice A

Conversor de Eletrónica de Potência

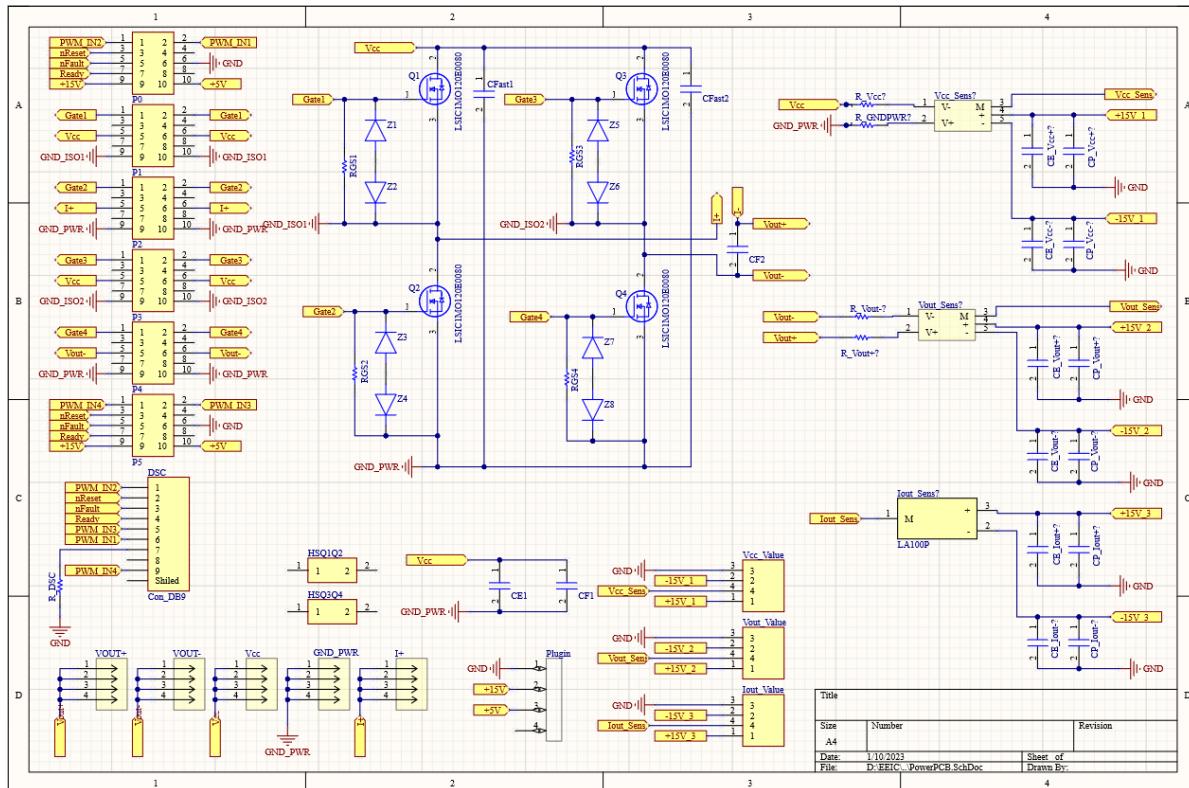


Figura A.1 – Esquemático da PCB para os conversores CC-CA.

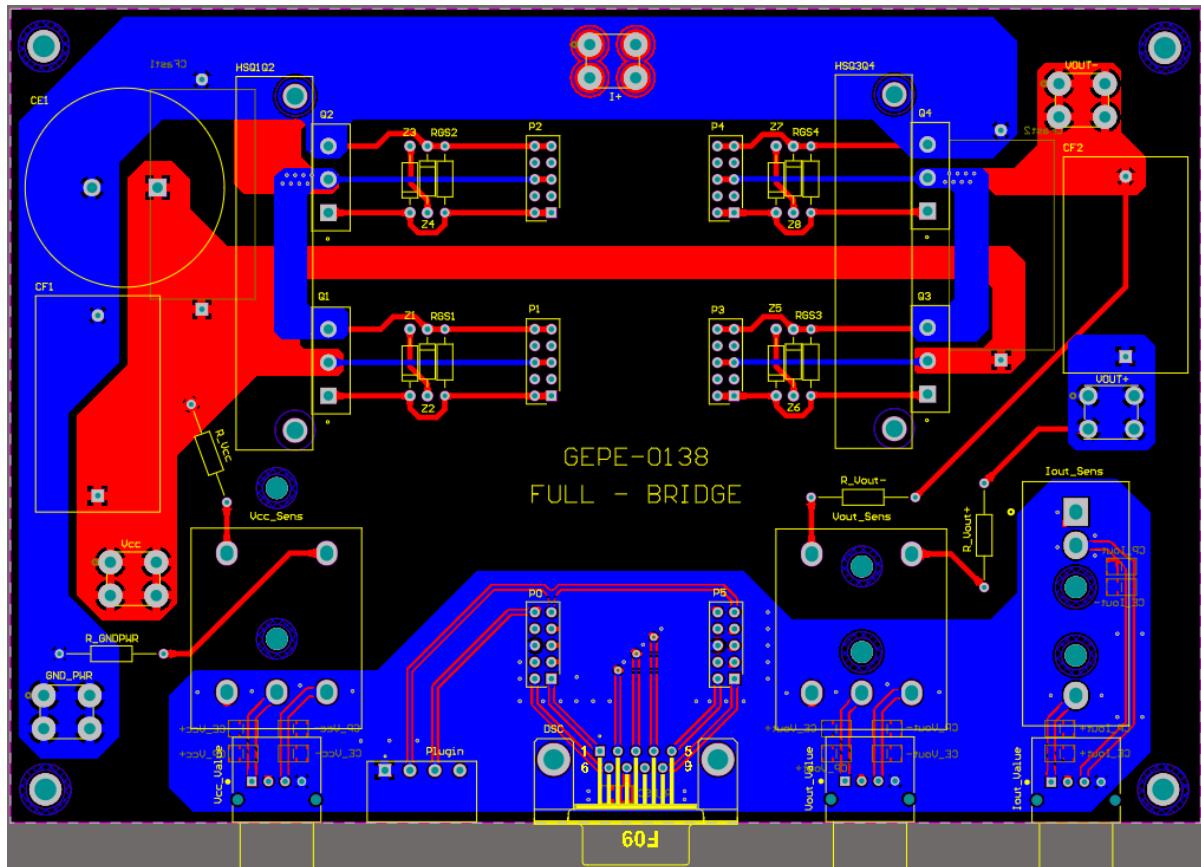


Figura A.2 – Layout da PCB para os conversores CC-CA-

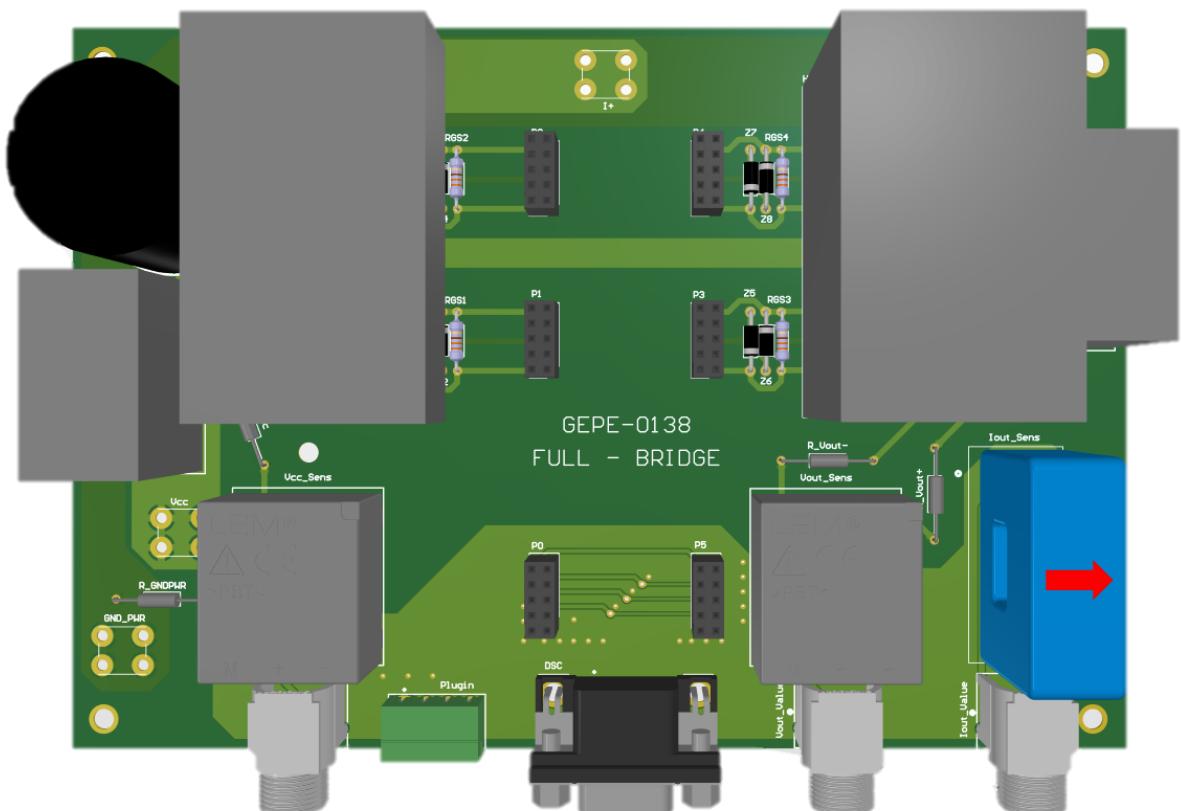


Figura A.3 – Vista 3D da PCB para os conversores CC-CA-

Apêndice B

Docking Station da LAUNCHXL-F28379D

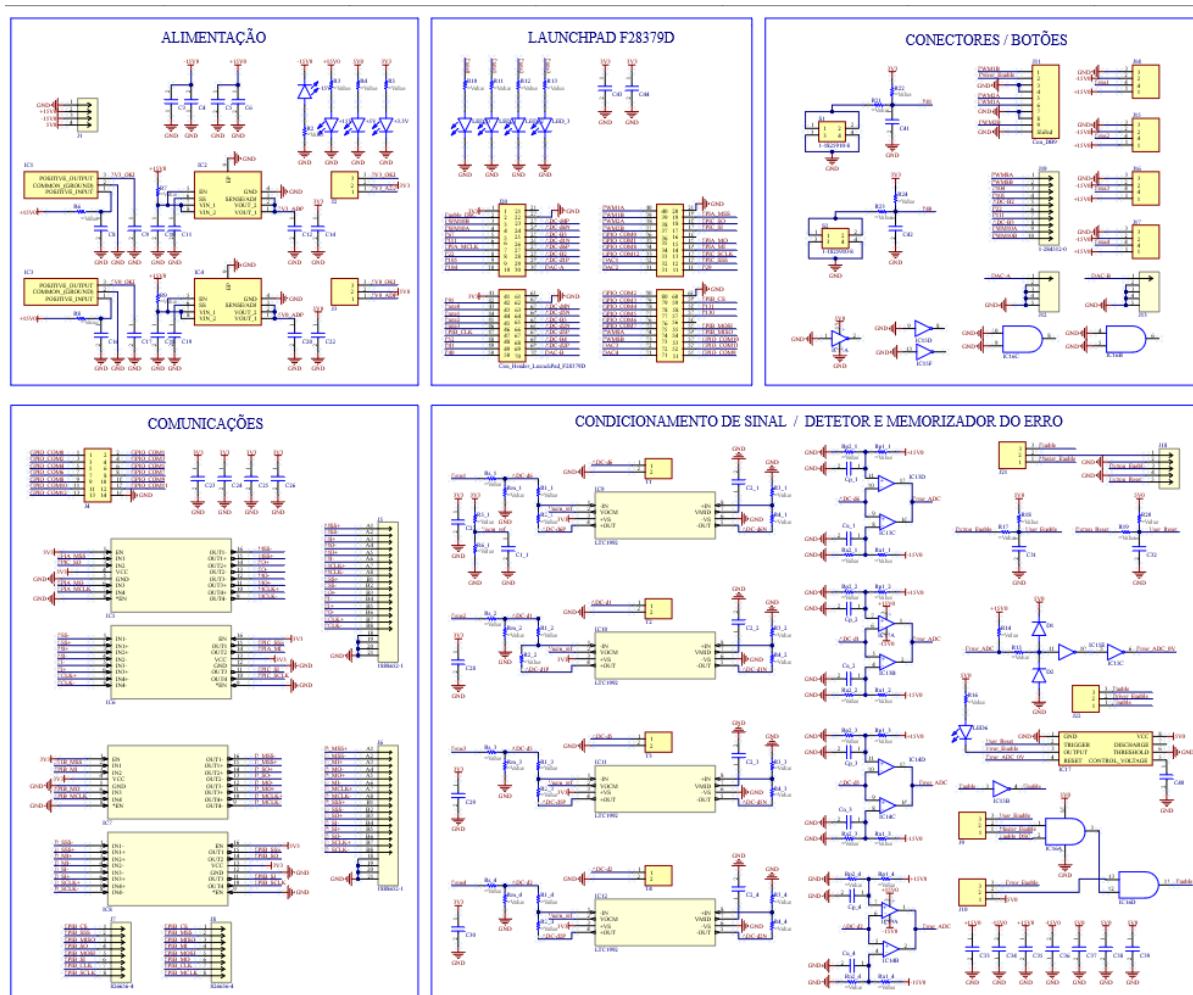


Figura B.4 – Esquemático da PCB Docking Station da LAUNCHXL-F28379D.

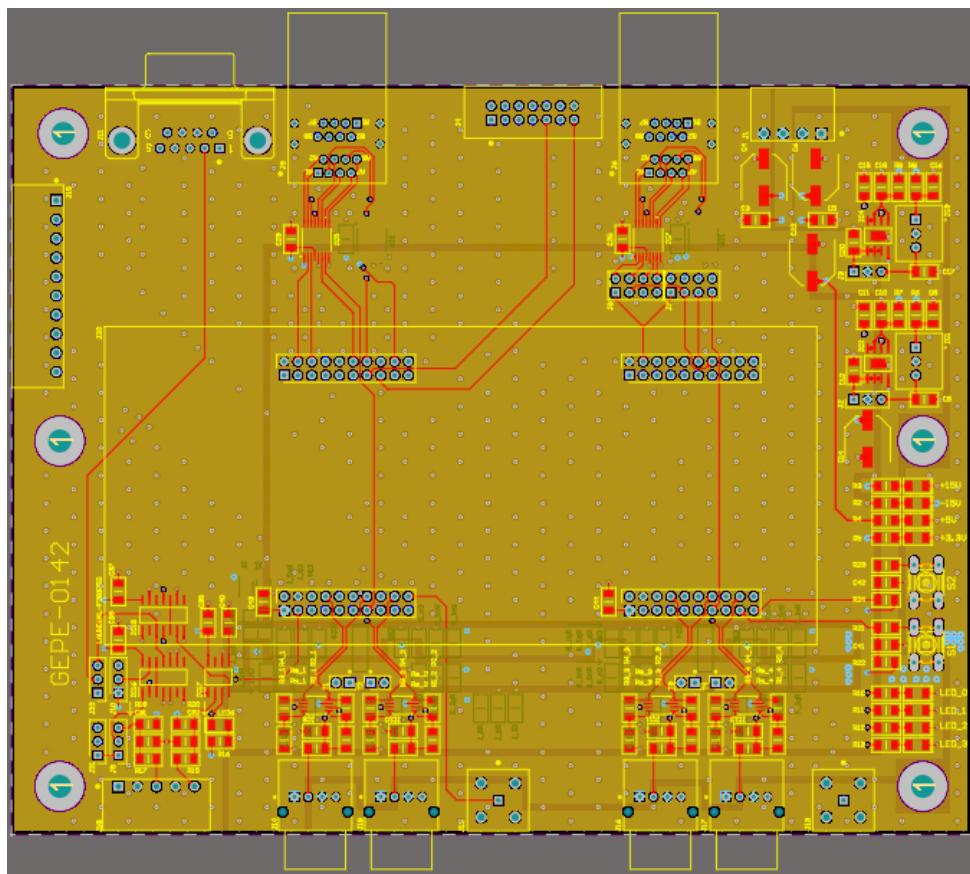


Figura B.5 – Layout da PCB Docking Station da LAUNCHXL-F28379D.

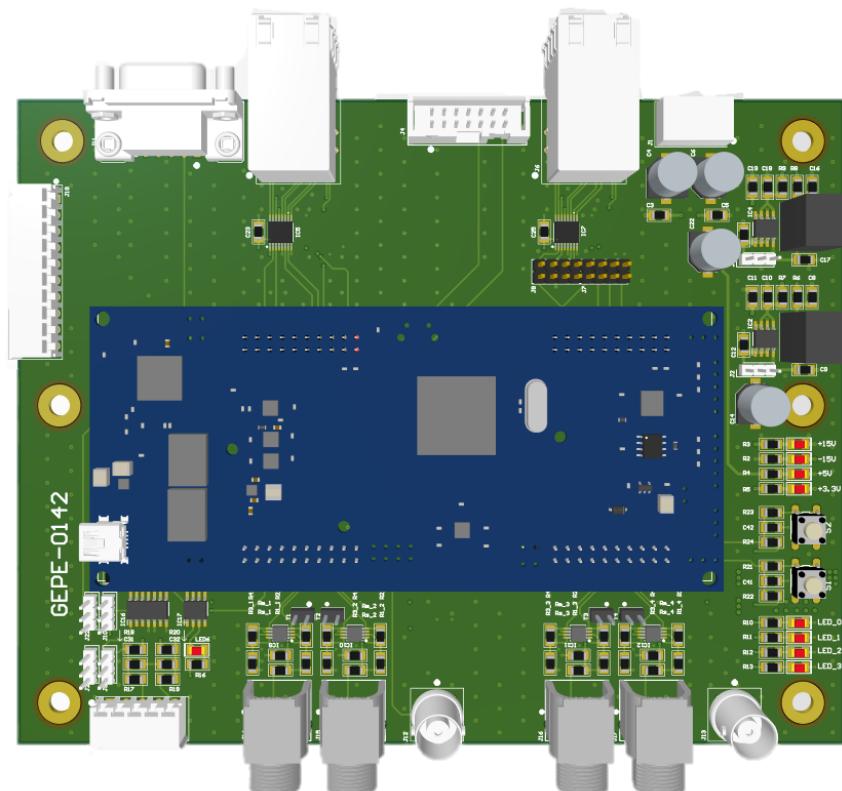


Figura B.6 – Vista 3D da PCB Docking Station da LAUNCHXL-F28379D.

Apêndice C

Driver ADUM3223

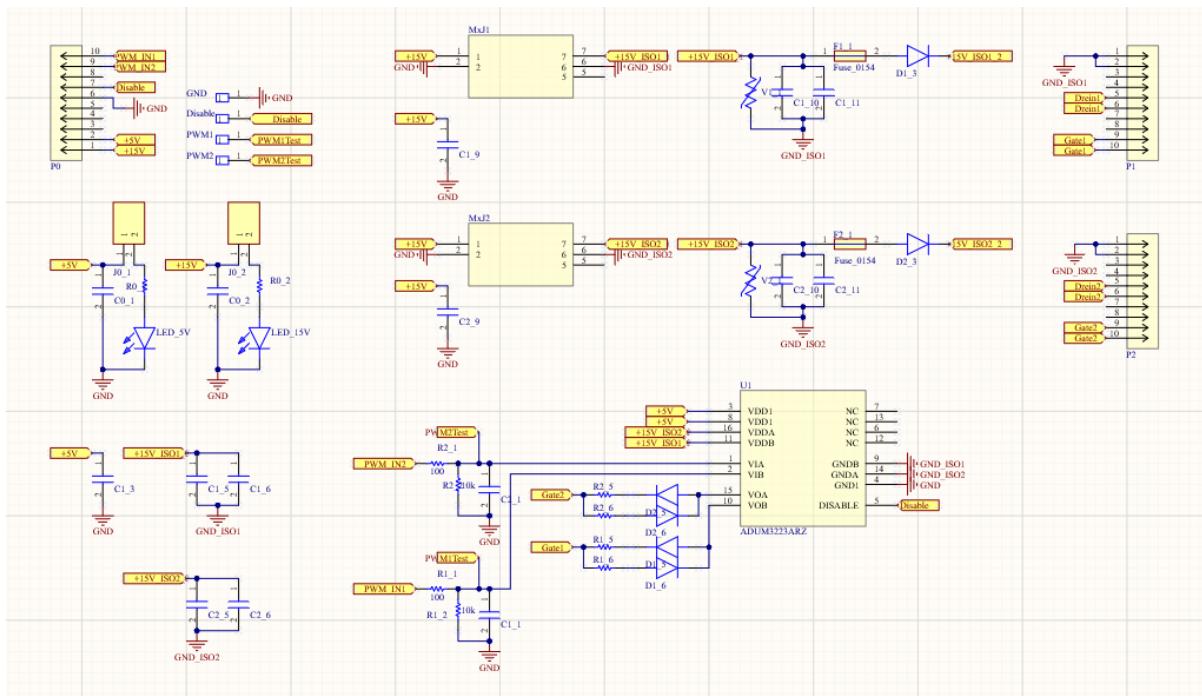
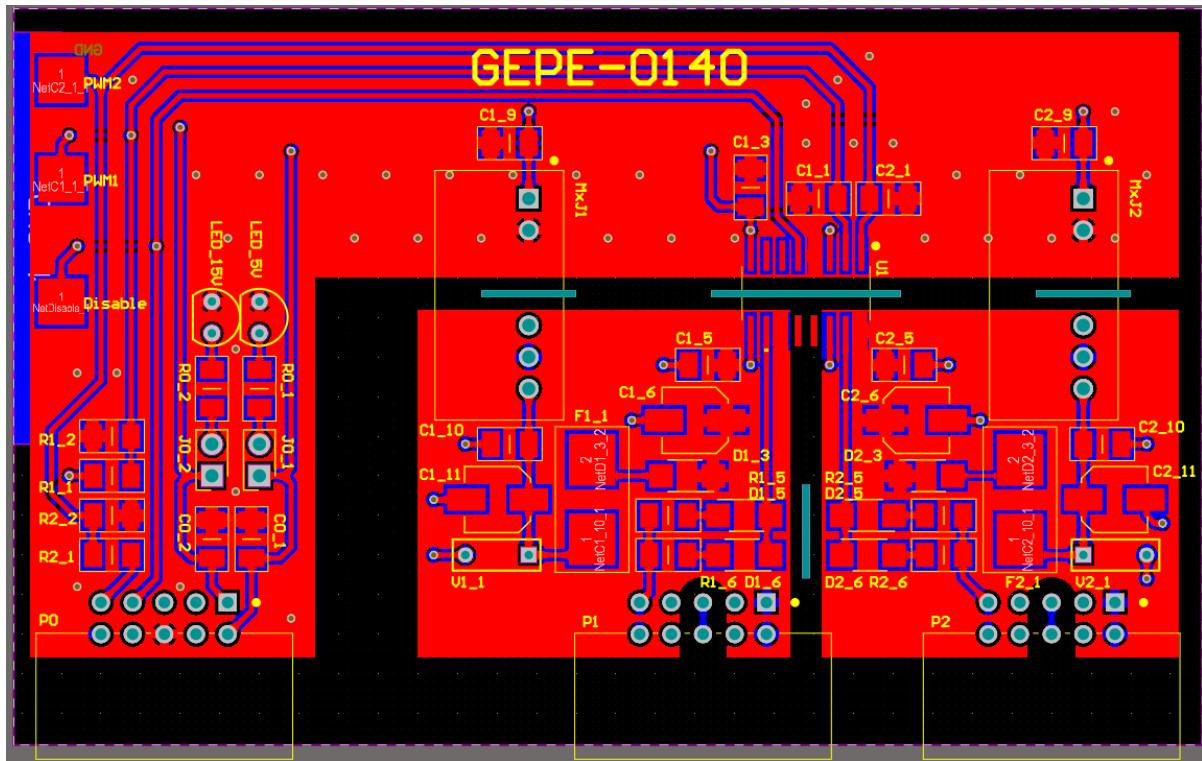
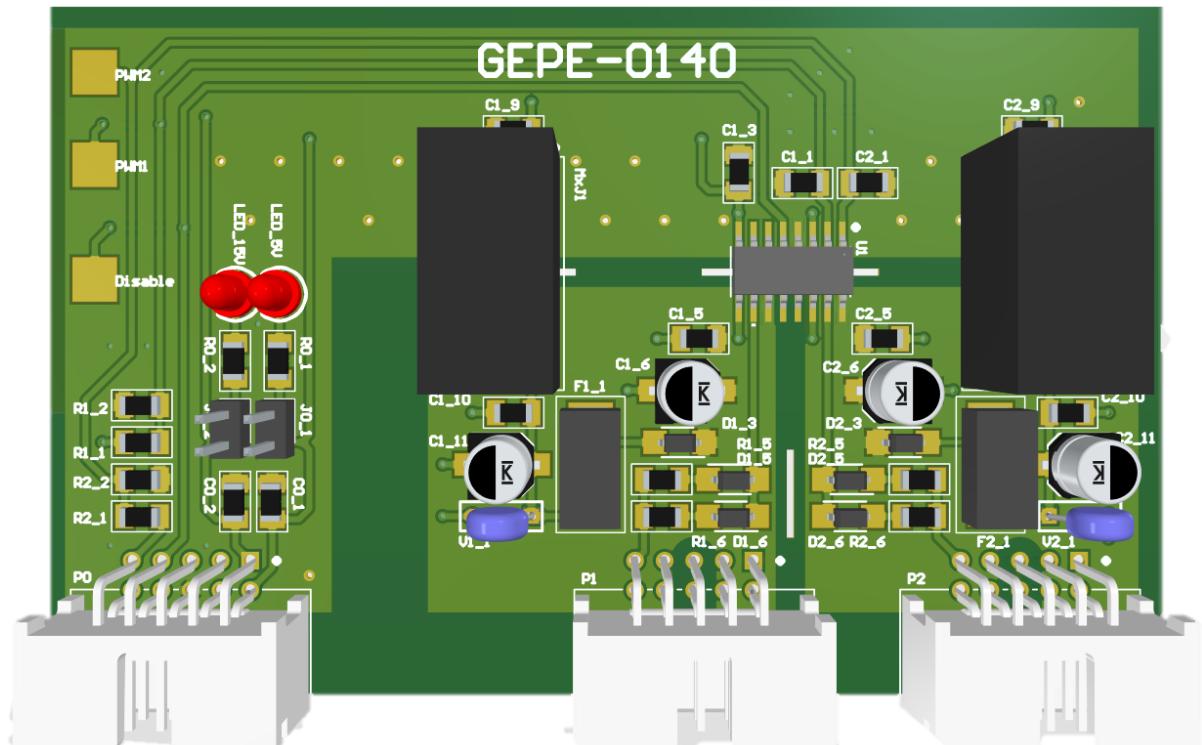


Figura C.7 – Esquemático da PCB de driver para o integrado ADUM3223.

Figura C.8 – Layout da PCB de *driver* para o integrado ADUM3223.Figura C.9 – Vista 3D da PCB de *driver* para o integrado ADUM3223.

Apêndice D

Driver ADUM4135

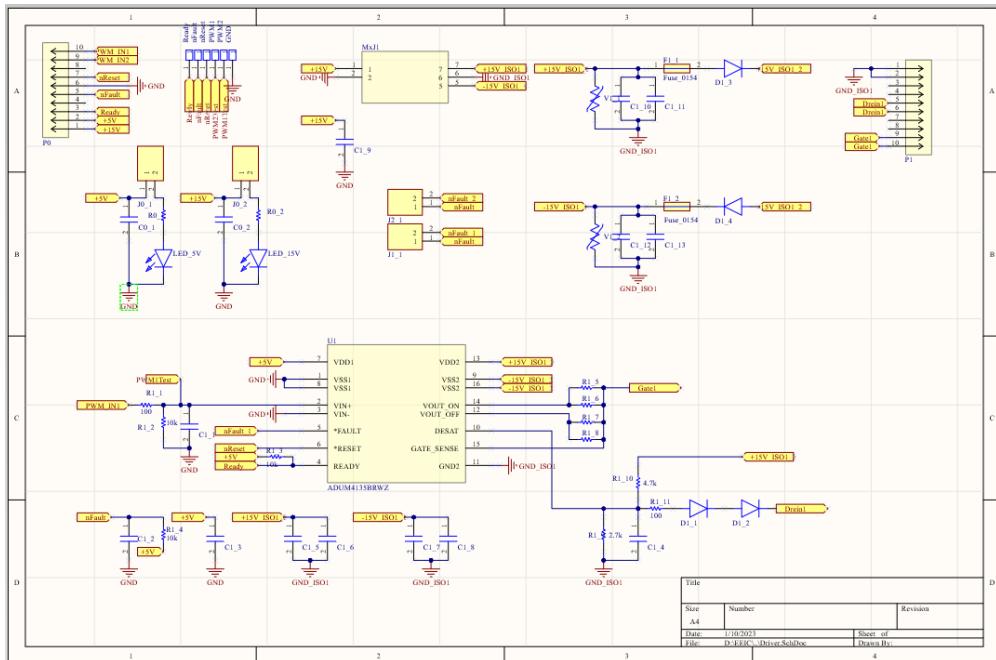


Figura D.10 – Primeiro esquemático da PCB de *driver* para o integrado ADUM4135.

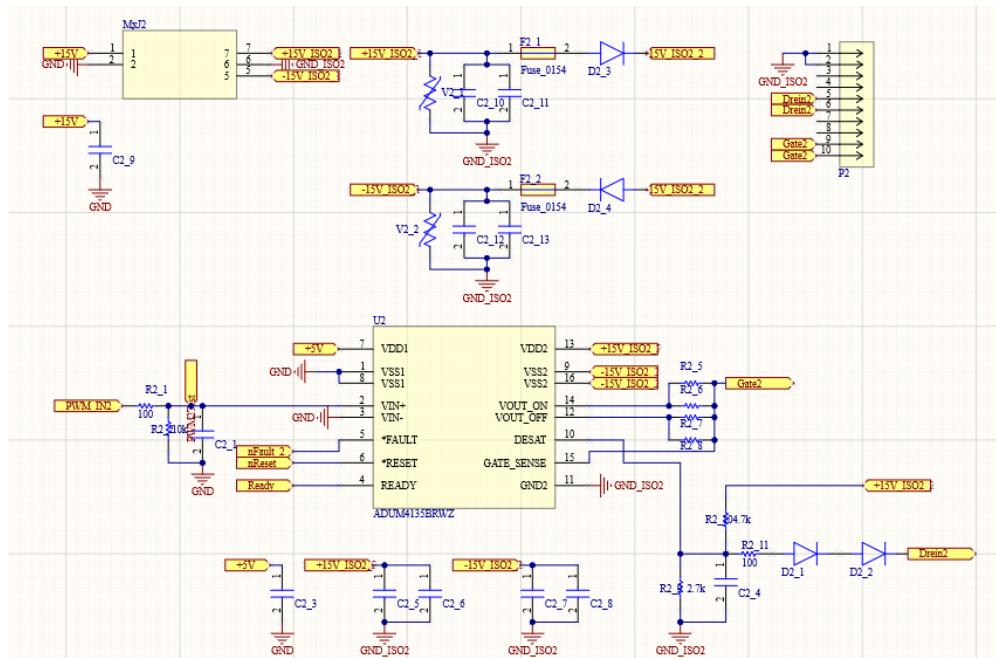


Figura D.11 – Segundo esquemático da PCB de *driver* para o integrado ADUM4135.

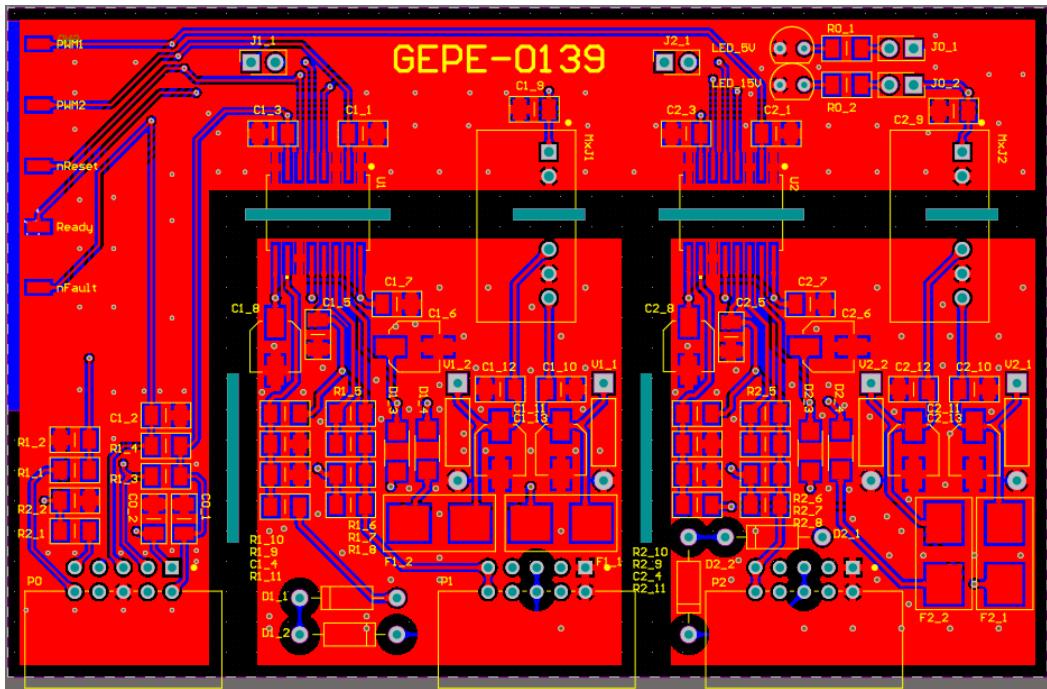


Figura D.12 – Layout da PCB de driver para o integrado ADUM4135.

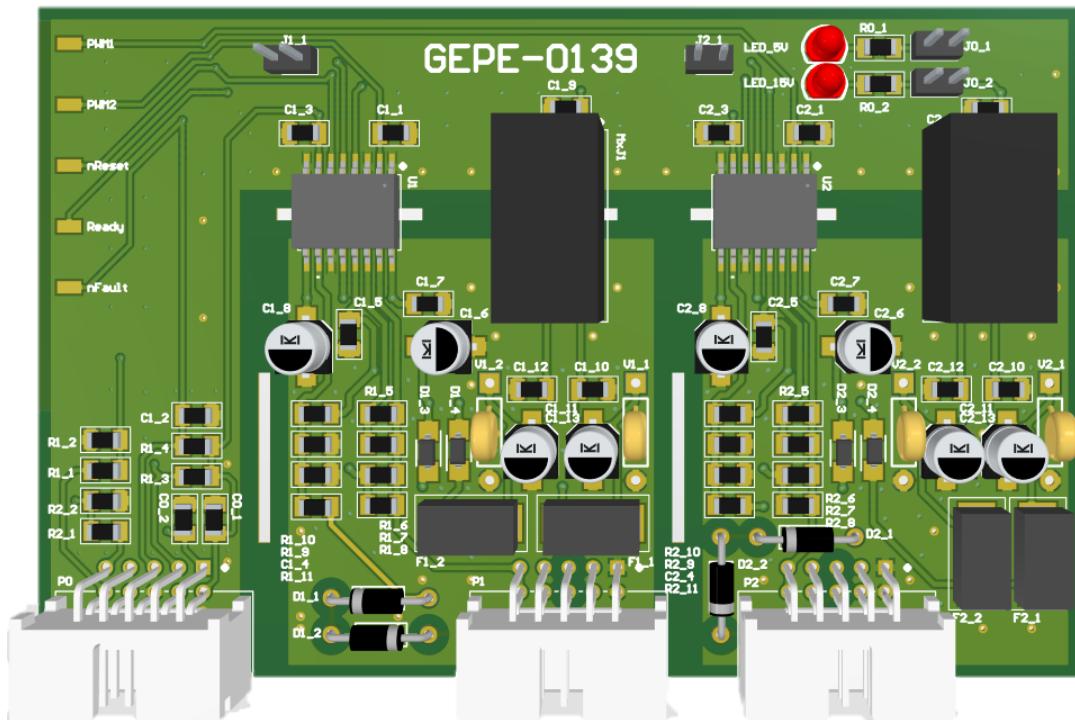


Figura D.4 – Vista 3D da PCB de driver para o integrado ADUM4135.

Apêndice E

Código implementado para o *Master* (*main.c*)

```
//#####
// FILE:    main.c
// TITLE:   Master Code
// CODE:    -
//          -
//          -
//#####
// AUTHOR: Ricardo Coelho
// DATE: 27_01_2022
// VERSION: v1.1
//#####

//*****
// Includes
//*****
#include "F28x_Project.h"
//C:\ti\controlSUITE\device_support\F2837xD\v210\F2837xD_common\include
#include "string.h"
#include "stdio.h"
#include "stdlib.h"
#include "math.h"
#include "time.h"
#include "Peripheral_Setup.h"
#include "Function_Definition.h"

//*****
// Defines
//*****
#define pi           3.14159
#define DOIS_PI_F    314.1592

//*****
// Global variables
//*****
// Control variables
Uint16 control_state;
Uint16 led_blink_counter;
Uint16 first_case;
Uint16 flag_xint1;
Uint16 flag_xint2;
Uint16 flag_timer0_isr;
Uint16 flag_receive_counter;
Uint16 flag_receive_config;
Uint16 flag_dc_bus;
Uint16 flag_dc_bus_check;
Uint16 user_enable;
```

```

Uint16 error;
Uint16 n_slaves;
Uint16 index_pwm;
Uint16 toggle_flag = 1;
Uint16 aux;
Uint16 aux2;
float32 aux_float;

// PLL variables
char start_pll;
float32 pll_uni;
float32 pll_amp;

// PI variables
char start_pi;
float32 amp_current;
float32 pi_ref;
float32 pi_ref2;

// ADC variables
float32 v_grid;
float32 i_grid;
//float32 i_grid2;
float32 v_grid_array[10] = {0};
float32 i_grid_array[10] = {0};
Uint16 average_index;
Uint16 loop;

// SPI variables
Uint16 index;
Uint16 receive_array[50] = {0};
Uint16 flag_receive_array;
Uint16 sdata_a; // send data
Uint16 sdata_c; // send data
Uint16 rdata_a; // received data
Uint16 rdata_c; // received data

//*****
// Prototype statements for functions
//*****

void initialize(void);
interrupt void adcaIsr(void);
interrupt void adccIsr(void);
interrupt void xint1_isr(void);
interrupt void xint2_isr(void);
interrupt void xint3_isr(void);
interrupt void timer0_isr(void);
interrupt void spiaRxFifoIsr(void);
interrupt void spicRxFifoIsr(void);

int main(void){

    //
    // Step 1. Initialize System Control:
    //
    // PLL, WatchDog, enable Peripheral Clocks
    // This example function is found in the F2837xD_SysCtrl.c file.
}

```

```

InitSysCtrl();

 $\text{// }$ 
 $\text{// Step 2. Initialize GPIO: }$ 
 $\text{// }$ 
 $\text{// This example function is found in the F2837xD_Gpio.c file and }$ 
 $\text{// illustrates how to set the GPIO to its default state. }$ 
 $\text{// Setup only the GP I/O only for SPI-A functionality }$ 
InitSpiaGpio();

 $\text{// }$ 
 $\text{// Step 3. Initialize PIE vector table: }$ 
 $\text{// }$ 
 $\text{// Disable and clear all CPU interrupts }$ 
DINT;
IER = 0x0000;
IFR = 0x0000;

 $\text{// Initialize PIE control registers to their default state: }$ 
 $\text{// This function is found in the F2837xD_PieCtrl.c file. }$ 
InitPieCtrl();

 $\text{// Initialize the PIE vector table with pointers to the shell Interrupt }$ 
 $\text{// Service Routines (ISR). }$ 
 $\text{// This will populate the entire table, even if the interrupt }$ 
 $\text{// is not used in this example. This is useful for debug purposes. }$ 
 $\text{// The shell ISR routines are found in F2837xD_DefaultIsr.c. }$ 
 $\text{// This function is found in F2837xD_PieVect.c. }$ 
InitPieVectTable();

 $\text{// Interrupts that are used in this example are re-mapped to }$ 
 $\text{// ISR functions found within this file. }$ 
EALLOW; // This is needed to write to EALLOW protected registers
CpuSysRegs.PCLKCR0.bit.CPUTIMER0 = 1; // Enable timer clock

PieVectTable.ADCA1_INT = &adcaIsr;
PieVectTable.ADCC1_INT = &adccIsr;
PieVectTable.XINT1_INT = &xint1_isr;
PieVectTable.XINT2_INT = &xint2_isr;
PieVectTable.XINT3_INT = &xint3_isr;
PieVectTable.TIMER0_INT = &timer0_isr;
PieVectTable.SPIA_RX_INT = &spiaRxFifoIsr;
PieVectTable.SPIC_RX_INT = &spicRxFifoIsr;
EDIS; // This is needed to disable write to EALLOW protected registers

 $\text{// }$ 
 $\text{// Step 4. Initialize the Device Peripherals: }$ 
 $\text{// }$ 
Setup_GPIO();
Setup_Timer();
Setup_ePWM();
Setup_SPI();
Setup_ADC();
Setup_DAC();
Setup_External_Interrupt();

 $\text{// }$ 
 $\text{// Step 5. User specific code, enable interrupts: }$ 
 $\text{// }$ 

```

```

// Initialize
initialize();

// Enable interrupts required for this example
PieCtrlRegs.PIECTRL.bit.ENPIE = 1;           // Enable the PIE block

PieCtrlRegs.PIEIER1.bit.INTx1 = 1;             // Enable PIE Group 1, INT 1 (ADCA1
interrupt)
PieCtrlRegs.PIEIER1.bit.INTx3 = 1;             // Enable PIE Group 1, INT 3 (ADCC1
interrupt)
PieCtrlRegs.PIEIER1.bit.INTx4 = 1;             // Enable PIE Group 1, INT 4 (XINT1
interrupt)
PieCtrlRegs.PIEIER1.bit.INTx5 = 1;             // Enable PIE Group 1, INT 5 (XINT2
interrupt)
PieCtrlRegs.PIEIER1.bit.INTx7 = 1;             // Enable PIE Group 1, INT 7 (TIMER0
interrupt)
PieCtrlRegs.PIEIER6.bit.INTx1 = 1;             // Enable PIE Group 6, INT 1 (SPIA_RX
interrupt)
PieCtrlRegs.PIEIER6.bit.INTx9 = 1;             // Enable PIE Group 6, INT 9 (SPIC_RX
interrupt)
PieCtrlRegs.PIEIER12.bit.INTx1 = 1;            // Enable PIE Group 12, INT 1 (XINT3
interrupt)
IER |= M_INT1;                                // Enable CPU INT1
IER |= M_INT6;                                // Enable CPU INT6
IER |= M_INT12;                               // Enable CPU INT12
EINT;                                         // Enable Global interrupt INTM
ERTM;                                         // Enable Global realtime interrupt

DBGM

CpuTimer0Regs.TCR.all = 0x4001;                // Start Timer0

//
// Step 6. IDLE loop. Just sit and loop forever (optional):
//
while(1){
    if(flag_timer0_isr == 1){
        switch(control_state){

            case 0:
                // Initial action
                if(first_case == 0){
                    initialize();
                    first_case = 1;
                }
                // Case action
                if(led_blink_counter > 25000){
                    GpioDataRegs.GPETOGGLE.bit.GPIO139 = 1;
                    GpioDataRegs.GPBTOGGLE.bit.GPIO56 = 1;
                    GpioDataRegs.GPDTOGGLE.bit.GPIO97 = 1;
                    GpioDataRegs.GPCTOGGLE.bit.GPIO94 = 1;
                    GpioDataRegs.GPADAT.bit.GPIO16 =
GpioDataRegs.GPCDAT.bit.GPIO94;
                    led_blink_counter = 0;
                }
                // Condition to switch cases
                if(flag_xint1 == 1){
                    control_state = 1;
                    first_case = 0;
                    flag_xint1 = 0;
                }
            }
        }
    }
}

```

```

        }

break;

case 1:
    // Initial action
    if(first_case == 0){
        GpioDataRegs.GPESET.bit.GPIO139 = 1;
        GpioDataRegs.GPBCLEAR.bit.GPIO56 = 1;
        GpioDataRegs.GPDCLEAR.bit.GPIO97 = 1;
        GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
        slave_counter();
        first_case = 1;
    }
    // Condition to switch cases
    if(flag_xint2 == 1){
        control_state = 0;
        first_case = 0;
        flag_xint2 = 0;
    }
    else if(flag_receive_counter == 1){
        control_state = 2;
        first_case = 0;
        flag_receive_counter = 0;
    }
break;

case 2:
    // Initial action
    if(first_case == 0){
        GpioDataRegs.GPECLEAR.bit.GPIO139 = 1;
        GpioDataRegs.GPBSET.bit.GPIO56 = 1;
        GpioDataRegs.GPDCLEAR.bit.GPIO97 = 1;
        GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
        slave_config(n_slaves);
        first_case = 1;
    }
    // Condition to switch cases
    if(flag_xint2 == 1){
        control_state = 0;
        first_case = 0;
        flag_xint2 = 0;
    }
    else if(flag_receive_config == 1 && aux == 1){
        control_state = 3;
        first_case = 0;
        flag_receive_config = 0;
    }
break;

case 3:
    // Initial action
    if(first_case == 0){
        GpioDataRegs.GPESET.bit.GPIO139 = 1;
        GpioDataRegs.GPBSET.bit.GPIO56 = 1;
        GpioDataRegs.GPDCLEAR.bit.GPIO97 = 1;
        GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
        slave_dc_bus(n_slaves);
    }

```

```

        first_case = 1;
    }
    // Condition to switch cases
    if(flag_xint2 == 1){
        control_state = 0;
        first_case = 0;
        flag_xint2 = 0;

    }
    /*else if(flag_dc_bus == 1){
        control_state = 4;
        first_case = 0;
        flag_dc_bus = 0;
    }*/
    break;

case 4:
    // Initial action
    if(first_case == 0){
        GpioDataRegs.GPECLEAR.bit.GPIO139 = 1;
        GpioDataRegs.GPBCLEAR.bit.GPIO56 = 1;
        GpioDataRegs.GPDSET.bit.GPIO97 = 1;
        GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
        GpioDataRegs.GPBCLEAR.bit.GPIO32 = 1;
        flag_dc_bus_check = dc_bus_check(receive_array, n_slaves);
        start_pll = 1;
        first_case = 1;
    }
    // Case action
    /*for(loop = 0; loop < 10; loop++) {
        v_grid = v_grid + v_grid_array[loop];
        i_grid = i_grid + i_grid_array[loop];
    }
    v_grid = v_grid / 10;
    i_grid = i_grid / 10;*/
    pll_amp = pll(v_grid, &pll_uni, start_pll);
    DacaRegs.DACVALS.bit.DACVALS = (UInt16)((pll_uni * pll_amp *
4096.0 / 420.0) + 2048.0 - 1.0);
    DacbRegs.DACVALS.bit.DACVALS = (UInt16)((v_grid * 4096.0 / 420.0)
+ 2048.0 - 1.0);
    // Condition to switch cases
    if(flag_xint2 == 1){
        control_state = 0;
        first_case = 0;
        flag_xint2 = 0;
    }
    else if(flag_dc_bus_check == 1 && user_enable == 1 /*&& pll_amp
>= 150*/){
        control_state = 5;
        first_case = 0;
        flag_dc_bus = 0;
    }
    break;

case 5:
    // Initial action
    if(first_case == 0){
        GpioDataRegs.GPESET.bit.GPIO139 = 1;
        GpioDataRegs.GPBCLEAR.bit.GPIO56 = 1;

```

```

        GpioDataRegs.GPDSET.bit.GPIO97 = 1;
        GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
        GpioDataRegs.GPBSET.bit.GPIO32 = 1;
        start_pll = 1;
        start_pi = 1;
        first_case = 1;
    }
    // Case action
    /*for(loop = 0; loop < 10; loop++) {
        v_grid = v_grid + v_grid_array[loop];
        i_grid = i_grid + i_grid_array[loop];
    }
    v_grid = v_grid / 10;
    i_grid = i_grid / 10;*/

    pll_amp = pll(v_grid, &pll_uni, start_pll);

    //pi_ref = current_control_pred(v_grid, i_grid, pll_uni,
amp_current, start_pi);
    pi_ref = current_control(i_grid, pll_uni, amp_current, start_pi);

    send_pwm_data(((UInt32)pi_ref));
    error = error_check();

    if(amp_current>=1){
        GpioDataRegs.GPASET.bit.GPIO22 = 1;
    }
    else GpioDataRegs.GPACLEAR.bit.GPIO22 = 1;
    //pi_ref2 = pi_ref/1.23;
    //DacbRegs.DACVALS.bit.DACVALS = (UInt16)pi_ref2;
    //DacbRegs.DACVALS.bit.DACVALS = (UInt16)(aux2 * 2);
    //DacbRegs.DACVALS.bit.DACVALS = (UInt16)((v_grid * 4096.0 /
420.0) + 2048.0 - 1.0);
    DacaRegs.DACVALS.bit.DACVALS = (UInt16)((pli_uni * amp_current) *
4096.0 / 20.0) + 2048.0 - 1.0);
    DacbRegs.DACVALS.bit.DACVALS = (UInt16)((i_grid * 4096.0 / 20.0) +
2048.0 - 1.0);

    // Condition to switch cases
    /*if (error == 0){
        control_state = 6;
        first_case = 0;
    }
    else if (user_enable == 0){
        control_state = 4;
        first_case = 0;
    }*/
    if (user_enable == 0){
        control_state = 4;
        first_case = 0;
    }
break;

case 6:
    // Initial action
    if(first_case == 0){
        initialize();
        GpioDataRegs.GPESET.bit.GPIO139 = 1;
        GpioDataRegs.GPDSET.bit.GPIO97 = 1;
    }
}

```

```

        }
        first_case = 1;
        // Case action
        if(led_blink_counter > 6250){
            GpioDataRegs.GPETOOGLE.bit.GPIO139 = 1;
            GpioDataRegs.GPBTOGGLE.bit.GPIO56 = 1;
            GpioDataRegs.GPDTOGGLE.bit.GPIO97 = 1;
            GpioDataRegs.GPCTOGGLE.bit.GPIO94 = 1;
            led_blink_counter = 0;
        }
        // Condition to switch cases
        if(flag_xint2 == 1){
            control_state = 0;
            first_case = 0;
            flag_xint2 = 0;
        }
        break;

    default:
        break;

    } //end switch(control_state)
    flag_timer0_isr = 0;
} //end if(flag_timer0_isr == 1)
} //end while(1)
} //end main()

//*****
// adcaIsr - ISR for ADC-A1
//*****
void initialize(void)
{
    GpioDataRegs.GPECLEAR.bit.GPIO139 = 1;
    GpioDataRegs.GPBCLEAR.bit.GPIO56 = 1;
    GpioDataRegs.GPDCLEAR.bit.GPIO97 = 1;
    GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
    GpioDataRegs.GPBCLEAR.bit.GPIO32 = 1;
    // Control variables
    //control_state      = 0;
    led_blink_counter = 0;
    //first_case         = 0;
    flag_xint1          = 0;
    flag_xint2          = 0;
    flag_timer0_isr     = 0;
    flag_receive_counter = 0;
    flag_receive_config = 0;
    flag_dc_bus         = 0;
    flag_dc_bus_check   = 0;
    //user_enable        = 0;
    //error              = 0;
    n_slaves             = 0;
    aux                  = 0;
    aux_float            = 0.0;
    // PLL variables
    start_pll            = 0;
    pll_uni              = 0.0;
    pll_amp = pll(v_grid, &pll_uni, start_pll);
    pll_amp              = 0.0;
}

```

```

// PI variables
start_pi          = 0;
amp_current       = 0.0;
pi_ref = current_control(i_grid, pll_uni, amp_current, start_pi);
pi_ref          = 0.0;
// ADC variables
v_grid           = 0.0;
i_grid           = 0.0;
// SPI variables
index            = 0;
memset(receive_array, 0, sizeof(receive_array));
flag_receive_array = 0;
sdata_a          = 0;
sdata_c          = 0;
rdata_a          = 0;
rdata_c          = 0;
}

//*****
// adcaIsr - ISR for ADC-A1
//*****
interrupt void adcaIsr(void)
{
    //aux_float = (float32)(-(AdcaResultRegs.ADCRESULT0 - 32768)) * 210 / 32768;
// == AdcaResultRegs.ADCRESULT1 (ADCINA2-ADCINA3)
    //adc1 = (float32)AdcaResultRegs.ADCRESULT2 * 3 / 65536;      // ==
AdcaResultRegs.ADCRESULT3 (ADCINA4-ADCINA5)
    //adc2 = (float32)AdcaResultRegs.ADCRESULT4 * 3 / 65536;      // ==
AdcaResultRegs.ADCRESULT5 (ADCINA14-ADCINA15)

    AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;      //clear INT1 flag
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

//*****
// adccIsr - ISR for ADC-C1
//*****
interrupt void adccIsr(void)
{
    /*v_grid_array[average_index] = (float32)(-(AdccResultRegs.ADCRESULT0 -
32768)) * 420 / 32768;      // == AdccResultRegs.ADCRESULT1 (ADCINC2-ADCINC3)
    i_grid_array[average_index] = (float32)(-(AdccResultRegs.ADCRESULT2 - 32768))
* 17 / 32768;      // == AdccResultRegs.ADCRESULT3 (ADCINC4-ADCINC5)
    average_index++;
    if(average_index >= 10) average_index = 0;
*/
    v_grid = (float32)(-(AdccResultRegs.ADCRESULT0 - 32768)) * 210 / 32768;      //
== AdccResultRegs.ADCRESULT1 (ADCINC2-ADCINC3)
    i_grid = (float32)(-(AdccResultRegs.ADCRESULT2 - 32768)) * 53 / 32768;      //
== AdccResultRegs.ADCRESULT3 (ADCINC4-ADCINC5)
    i_grid = i_grid/3;

    AdccRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;      //clear INT1 flag
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

```

```

}

//*****
// xint1_isr - External Interrupt 1 ISR
//*****
interrupt void xint1_isr(void)
{
    flag_xint1 = 1;

    // Acknowledge this interrupt to get more from group 1
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

//*****
// xint2_isr - External Interrupt 2 ISR
//*****
interrupt void xint2_isr(void)
{
    flag_xint2 = 1;

    // Acknowledge this interrupt to get more from group 1
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

//*****
// xint3_isr - External Interrupt 3 ISR
//*****
interrupt void xint3_isr(void)
{
    user_enable = GpioDataRegs.GPDDAT.bit.GPIO105;

    // Acknowledge this interrupt to get more from group 1
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

//*****
// timer0_isr - Timer0 ISR
//*****
interrupt void timer0_isr(void)
{
    led_blink_counter++;
    flag_timer0_isr = 1;
    toggle_flag++;

    if(control_state == 5 && toggle_flag == 5) GpioDataRegs.GPAToggle.bit.GPIO16 =
1;
    if(toggle_flag > 5) toggle_flag = 1;

    /* Parallel Communication Test*/
    /*if(control_state == 5 && toggle_flag == 5) GpioDataRegs.GPASET.bit.GPIO16 =
1;
    else GpioDataRegs.GPACLEAR.bit.GPIO16 = 1;
    if(toggle_flag > 5) toggle_flag = 1;*/
}

```

```

// Acknowledge this interrupt to receive more interrupts from group 1
PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

//*****
// spiaRx_fifoIsr - ISR for SPI-A receive FIFO
//*****
interrupt void spiaRx_fifoIsr(void)
{
    rdata_a=SpiaRegs.SPIRXBUF;           // Read data

    SpiaRegs.SPIFFRX.bit.RXFOVFCLR=1;   // Clear Overflow flag
    SpiaRegs.SPIFFRX.bit.RXFFINTCLR=1;  // Clear Interrupt flag
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

//*****
// spicRx_fifoIsr - ISR for SPI-C receive FIFO
//*****
interrupt void spicRx_fifoIsr(void)
{
    rdata_c = SpicRegs.SPIRXBUF;
    receive_array[index] = rdata_c;      // Read data
    index++;

    if(rdata_c == 0x0024){
        if(receive_array[1] == 1){
            flag_receive_counter = 1;
            n_slaves = receive_array[index-2];
        }
        else if(receive_array[1] == 2) flag_receive_config = 1;
        else if (receive_array[1] == 3) flag_dc_bus = 1;

        index = 0;
    }

    SpicRegs.SPIFFRX.bit.RXFOVFCLR=1;   // Clear Overflow flag
    SpicRegs.SPIFFRX.bit.RXFFINTCLR=1;  // Clear Interrupt flag
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

```

Apêndice F

Código implementado para o *Master* (*Peripheral_Setup.h*)

```
/*
 * Peripheral_Setup.h
 *
 * Created on: 23 de jul de 2020
 * Author: waner
 */

#ifndef PERIPHERAL_SETUP_H_
#define PERIPHERAL_SETUP_H_
#include "F28x_Project.h"

void Setup_GPIO(void);
void Setup_Timer(void);
void Setup_External_Interrupt(void);
void Setup_SPI(void);
void Setup_ePWM(void);
void Setup_ADC(void);
void Setup_DAC(void);

#endif /* PERIPHERAL_SETUP_H_ */
```

Apêndice G

Código implementado para o *Master*
(Peripheral_Setup.c)

```

/*
 * Peripheral_Setup.c
 *
 * Created on: 23 de jul de 2020
 * Author: waner
 */
#include "Peripheral_Setup.h"

//*****
***** *****
//*****
***** *****
***** *****
void Setup_GPIO(void)
{
    EALLOW;

    // ***** GPIO_COM ***** //
    GpioCtrlRegs.GPAMUX1.all = 0X0000;
    GpioCtrlRegs.GPAMUX2.all = 0X0000;
    GpioCtrlRegs.GPADIR.all = 0xFFFFFFFF; //1: Configures pin as output.
    GpioCtrlRegs.GPACSEL1.all = GPIO_MUX_CPU1; //GPIO is controlled by CPU1
    GpioCtrlRegs.GPACSEL2.all = GPIO_MUX_CPU1; //GPIO is controlled by CPU1

    // ***** SPI      ***** //
    // *** SPI-A ***
    // Enable internal pull-up for the selected pins
    GpioCtrlRegs.GPBPU.D.bit.GPIO58 = 0; // Enable pull-up on GPIO58 (SPISIMOA)
    GpioCtrlRegs.GPBPU.D.bit.GPIO59 = 0; // Enable pull-up on GPIO59(SPISOMIA)
    GpioCtrlRegs.GPBPU.D.bit.GPIO60 = 0; // Enable pull-up on GPIO60 (SPICLKA)
    GpioCtrlRegs.GPBPU.D.bit.GPIO61 = 0; // Enable pull-up on GPIO61 (SPLISTEA)
    // Set qualification for selected pins to asynch only
    GpioCtrlRegs.GPBQSEL2.D.bit.GPIO58 = 3; // Asynch input GPIO58 (SPISIMOA)
    GpioCtrlRegs.GPBQSEL2.D.bit.GPIO59 = 3; // Asynch input GPIO59 (SPISOMIA)
    GpioCtrlRegs.GPBQSEL2.D.bit.GPIO60 = 3; // Asynch input GPIO60 (SPICLKA)
    GpioCtrlRegs.GPBQSEL2.D.bit.GPIO61 = 3; // Asynch input GPIO61 (SPLISTEA)
    // Configure SPI-A pins using GPIO regs
    GpioCtrlRegs.GPBMUX2.D.bit.GPIO58 = 3; // Configure GPIO58 as SPISIMOA
    GpioCtrlRegs.GPBMUX2.D.bit.GPIO59 = 3; // Configure GPIO59 as SPISOMIA
    GpioCtrlRegs.GPBMUX2.D.bit.GPIO60 = 3; // Configure GPIO60 as SPICLKA
    GpioCtrlRegs.GPBMUX2.D.bit.GPIO61 = 3; // Configure GPIO61 as SPLISTEA
    GpioCtrlRegs.GPBGMUX2.D.bit.GPIO58 = 3; // Configure GPIO58 as SPISIMOA
    GpioCtrlRegs.GPBGMUX2.D.bit.GPIO59 = 3; // Configure GPIO59 as SPISOMIA
    GpioCtrlRegs.GPBGMUX2.D.bit.GPIO60 = 3; // Configure GPIO60 as SPICLKA
    GpioCtrlRegs.GPBGMUX2.D.bit.GPIO61 = 3; // Configure GPIO61 as SPLISTEA
}

```

```

// *** SPI-C ***
// Enable internal pull-up for the selected pins
GpioCtrlRegs.GPDPUD.bit.GPIO122 = 0;      // Enable pull-up on GPIO122
(SPISIMOC)
GpioCtrlRegs.GPDPUD.bit.GPIO123 = 0;      // Enable pull-up on GPIO123
(SPISOMIC)
GpioCtrlRegs.GPDPUD.bit.GPIO124 = 0;      // Enable pull-up on GPIO124
(SPICLKC)
GpioCtrlRegs.GPDPUD.bit.GPIO125 = 0;      // Enable pull-up on GPIO125
(SPISTEC)
// Set qualification for selected pins to asynch only
GpioCtrlRegs.GPDQSEL2.bit.GPIO122 = 3;    // Asynch input GPIO122 (SPISIMOC)
GpioCtrlRegs.GPDQSEL2.bit.GPIO123 = 3;    // Asynch input GPIO123 (SPISOMIC)
GpioCtrlRegs.GPDQSEL2.bit.GPIO124 = 3;    // Asynch input GPIO124 (SPICLKC)
GpioCtrlRegs.GPDQSEL2.bit.GPIO125 = 3;    // Asynch input GPIO125 (SPISTEC)
// Configure SPI-C pins using GPIO regs
GpioCtrlRegs.GPDMUX2.bit.GPIO122 = 2;     // Configure GPIO122 as SPISIMOC
GpioCtrlRegs.GPDMUX2.bit.GPIO123 = 2;     // Configure GPIO123 as SPISOMIC
GpioCtrlRegs.GPDMUX2.bit.GPIO124 = 2;     // Configure GPIO124 as SPICLKC
GpioCtrlRegs.GPDMUX2.bit.GPIO125 = 2;     // Configure GPIO125 as SPISTEC
GpioCtrlRegs.GPDGMUX2.bit.GPIO122 = 1;    // Configure GPIO122 as SPISIMOC
GpioCtrlRegs.GPDGMUX2.bit.GPIO123 = 1;    // Configure GPIO123 as SPISOMIC
GpioCtrlRegs.GPDGMUX2.bit.GPIO124 = 1;    // Configure GPIO124 as SPICLKC
GpioCtrlRegs.GPDGMUX2.bit.GPIO125 = 1;    // Configure GPIO125 as SPISTEC

// ***** GPIOs **** //
// Enable DSC (output)
GpioCtrlRegs.GPBMUX1.bit.GPIO32 = 0;
GpioCtrlRegs.GPBDIR.bit.GPIO32 = 1;
GpioCtrlRegs.GPBCSEL1.bit.GPIO32 = GPIO_MUX_CPU1; //GPIO1 is controlled by
CPU1
// LED - State 0 (output)
GpioCtrlRegs.GPEMUX1.bit.GPIO139 = 0;
GpioCtrlRegs.GPEDIR.bit.GPIO139 = 1;
GpioCtrlRegs.GPECSEL2.bit.GPIO139 = GPIO_MUX_CPU1; //GPIO1 is controlled by
CPU1
// LED - State 1 (output)
GpioCtrlRegs.GPBMUX2.bit.GPIO56 = 0;
GpioCtrlRegs.GPBDIR.bit.GPIO56 = 1;
GpioCtrlRegs.GPBQSEL2.bit.GPIO56 = GPIO_MUX_CPU1; //GPIO1 is controlled by
CPU1
// LED - State 2 (output)
GpioCtrlRegs.GPDMUX1.bit.GPIO97 = 0;
GpioCtrlRegs.GPDDIR.bit.GPIO97 = 1;
GpioCtrlRegs.GPDCSEL1.bit.GPIO97 = GPIO_MUX_CPU1; //GPIO1 is controlled by
CPU1
// LED - State 3 (output)
GpioCtrlRegs.GPCMUX2.bit.GPIO94 = 0;
GpioCtrlRegs.GPCDIR.bit.GPIO94 = 1;
GpioCtrlRegs.GPCQSEL2.bit.GPIO94 = GPIO_MUX_CPU1; //GPIO1 is controlled by
CPU1
// Button - S1 (input)
GpioCtrlRegs.GPBMUX1.bit.GPIO41 = 0;
GpioCtrlRegs.GPBDIR.bit.GPIO41 = 0;
GpioCtrlRegs.GPBCSEL2.bit.GPIO41 = GPIO_MUX_CPU1; //GPIO1 is controlled by
CPU1
// Button - S2 (input)
GpioCtrlRegs.GPBMUX1.bit.GPIO40 = 0;

```

```

GpioCtrlRegs.GPBDIR.bit.GPIO40 = 0;
GpioCtrlRegs.GPBCSEL2.bit.GPIO40 = GPIO_MUX_CPU1; //GPIO01 is controlled by
CPU1
// User Enable (input)
GpioCtrlRegs.GPDMUX1.bit.GPIO105 = 0;
GpioCtrlRegs.GPDDIR.bit.GPIO105 = 0;
GpioCtrlRegs.GPDCSEL2.bit.GPIO105 = GPIO_MUX_CPU1; //GPIO01 is controlled
by CPU1
// Error (input)
GpioCtrlRegs.GPDMUX1.bit.GPIO104 = 0;
GpioCtrlRegs.GPDDIR.bit.GPIO104 = 0;
GpioCtrlRegs.GPDCSEL2.bit.GPIO104 = GPIO_MUX_CPU1; //GPIO01 is controlled
by CPU1
EDIS;
//EXtra (output)
GpioCtrlRegs.GPAMUX2.bit.GPIO22 = 0;
GpioCtrlRegs.GPADIR.bit.GPIO22 = 1;
GpioCtrlRegs.GPACSEL3.bit.GPIO22 = GPIO_MUX_CPU1;
}

//*****
***** *****
//*****
***** *****
void Setup_Timer(void)
{
    // *** Timer 0 ***
    InitCpuTimers();
    ConfigCpuTimer(&CpuTimer0, 200, 20); // ConfigCpuTimer(struct CPUTIMER_VARS
*Timer, float Freq, float Period)

}

//*****
***** *****
//*****
***** *****
void Setup_External_Interrupt(void)
{
    GPIO_SetupXINT1Gpio(41);
    GPIO_SetupXINT2Gpio(40);
    GPIO_SetupXINT3Gpio(105);

    EALLOW;
    // Configure XINTx // 0 = Falling edge interrupt; 1 =
Rising edge interrupt; 3 = Rising and Fallind edge interrupt
    XintRegs.XINT1CR.bit.POLARITY = 0; // 0 = Falling edge interrupt
    XintRegs.XINT2CR.bit.POLARITY = 0; // 0 = Falling edge interrupt
    XintRegs.XINT3CR.bit.POLARITY = 3; // 3 = Rising and Fallind edge
interrupt
    // Enable XINTx
    XintRegs.XINT1CR.bit.ENABLE = 1; // Enable XINT1
    XintRegs.XINT2CR.bit.ENABLE = 1; // Enable XINT2
    XintRegs.XINT3CR.bit.ENABLE = 1; // Enable XINT3

    EDIS;
}

```

```

//*****
*****  

//*****  

*****  

void Setup_SPI()  

{  

    EALLOW;  

    // ***** SPI-A ***** //  

    // Initialize SPI FIFO registers  

    SpiaRegs.SPIFFTX.all = 0xE040;  

    SpiaRegs.SPIFFRX.all = 0x2061;  

    SpiaRegs.SPIFFCT.all = 0x0000;  

    // Set reset low before configuration changes  

    // Clock polarity (0 == rising, 1 == falling)  

    // 16-bit character  

    // Enable loop-back  

    SpiaRegs.SPICCR.bit.SPISWRESET = 0;  

    SpiaRegs.SPICCR.bit.CLKPOLARITY = 0;  

    SpiaRegs.SPICCR.bit.SPICHAR = (16 - 1);  

    SpiaRegs.SPICCR.bit.SPILBK = 0;  

    // Enable master (0 == slave, 1 == master)  

    // Enable transmission (Talk)  

    // Clock phase (0 == normal, 1 == delayed)  

    // SPI interrupts are disabled  

    SpiaRegs.SPICTL.bit.MASTER_SLAVE = 1;  

    SpiaRegs.SPICTL.bit.TALK = 1;  

    SpiaRegs.SPICTL.bit.CLK_PHASE = 0;  

    SpiaRegs.SPICTL.bit.SPIINTENA = 1;  

    // Set the baud rate  

    SpiaRegs.SPIBRR.bit.SPI_BIT_RATE = ((200E6 / 4) / 500E3) - 1;  

    // Set FREE bit  

    // Halting on a breakpoint will not halt the SPI  

    SpiaRegs.SPIPRI.bit.FREE = 1;  

    // Release the SPI from reset  

    SpiaRegs.SPICCR.bit.SPISWRESET = 1;  

    // ***** SPI-C ***** //  

    // Initialize SPI FIFO registers  

    SpicRegs.SPIFFTX.all = 0xE040;  

    SpicRegs.SPIFFRX.all = 0x2061;  

    SpicRegs.SPIFFCT.all = 0x0000;  

    // Set reset low before configuration changes  

    // Clock polarity (0 == rising, 1 == falling)  

    // 16-bit character  

    // Enable loop-back  

    SpicRegs.SPICCR.bit.SPISWRESET = 0;  

    SpicRegs.SPICCR.bit.CLKPOLARITY = 0;  

    SpicRegs.SPICCR.bit.SPICHAR = (16 - 1);  

    SpicRegs.SPICCR.bit.SPILBK = 0;  

    // Enable master (0 == slave, 1 == master)  

    // Enable transmission (Talk)  

    // Clock phase (0 == normal, 1 == delayed)  

    // SPI interrupts are disabled  

    SpicRegs.SPICTL.bit.MASTER_SLAVE = 0;  

    SpicRegs.SPICTL.bit.TALK = 1;  

    SpicRegs.SPICTL.bit.CLK_PHASE = 0;  

    SpicRegs.SPICTL.bit.SPIINTENA = 1;  

    // Set the baud rate

```

```

// Calculate BRR: 7-bit baud rate register value
// SPI CLK freq = 500 kHz
// LSPCLK freq = CPU freq / 4 (by default)
// BRR = (LSPCLK freq / SPI CLK freq) - 1
SpicRegs.SPIBRR.bit.SPI_BIT_RATE = ((200E6 / 4) / 500E3) - 1;
// Set FREE bit
// Halting on a breakpoint will not halt the SPI
SpicRegs.SPIPRI.bit.FREE = 1;
// Release the SPI from reset
SpicRegs.SPICCR.bit.SPISWRESET = 1;

EDIS;
}

//*****
*****void Setup_ePWM(void)
{
    EALLOW;
    CpuSysRegs.PCLKCR0.bit.TBCLKSYNC = 0;
    CpuSysRegs.PCLKCR2.bit.EPWM8 = 1;

    // ***** PWM-8 ***** //
    EPwm8Regs.TBPRD = 2500;                                // Set timer period
    (((100Mhz/CLKDIV)/Fpwm)/2)
    EPwm8Regs.CMPA.bit.CMPA = 1250;                          // Setup compare
    EPwm8Regs.TBPHS.bit.TBPHS = 0;                            // Phase is 0
    EPwm8Regs.TBCTR = 0x0000;                                // Clear counter
    EPwm8Regs.TBCTL.bit.SYNCSEL = TB_CTR_ZERO;               // Sync down-stream module
    EPwm8Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN;          // Count up/down
    EPwm8Regs.TBCTL.bit.PHSEN = TB_DISABLE;                 // Disable phase loading
    EPwm8Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;                // Clock ratio to SYSCLKOUT
    EPwm8Regs.TBCTL.bit.CLKDIV = TB_DIV2;                   // Period autoreload
    EPwm8Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;            // Load registers every ZERO
    EPwm8Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO_PRD;
    EPwm8Regs.CMPCTL.bit.SHDBMODE = CC_SHADOW;              // Load registers every ZERO
    EPwm8Regs.CMPCTL.bit.LOADMODE = CC_CTR_ZERO_PRD;

    // Set actions
    EPwm8Regs.AQCTLA.bit.CAU = AQ_SET;
    EPwm8Regs.AQCTLA.bit.CAD = AQ_CLEAR;
    EPwm8Regs.AQCTLB.bit.CAU = AQ_CLEAR;
    EPwm8Regs.AQCTLB.bit.CAD = AQ_SET;
    // Dead time config
    EPwm8Regs.DBCTL.bit.OUT_MODE = DB_FULL_ENABLE;          // Enable Dead-band module
    EPwm8Regs.DBCTL.bit.POLSEL = DB_ACTV_HIC;               // Active High complementary
    EPwm8Regs.DBRED.all = 100;                               // Set 100 clock ticks of
    rising edge dead time ( x us)
    EPwm8Regs.DBFED.all = 100;                               // Set 100 clock ticks of
    falling edge dead time ( x us)
    CpuSysRegs.PCLKCR0.bit.TBCLKSYNC = 1;
    // Stop PWM
    EPwm8Regs.DBCTL.bit.POLSEL = DB_ACTV_HI;
    EPwm8Regs.AQCSFRC.bit.CSFA = 1;                         // Forces a continuous low on
    output A
}

```

```

EPwm8Regs.AQCSFRC.bit.CSFB = 1;                                // Forces a continuous low on
output B

CpuSysRegs.PCLKCR0.bit.TBCLKSYNC = 1;
EDIS;
}

//*****
***** *****
//*****
***** *****
***** *****
void Setup_ADC(void)
{
    Uint16 acqps;

    EALLOW;
    CpuSysRegs.PCLKCR13.bit.ADC_A = 1;
    CpuSysRegs.PCLKCR13.bit.ADC_C = 1;

    // ***** ADC-A ***** //
    AdcaRegs.ADCCTL2.bit.PRESCALE = 6;           // set ADCCLK divider to /4
    AdcSetMode(ADC_ADCA, ADC_RESOLUTION_16BIT, ADC_SIGNALMODE_DIFFERENTIAL);
    AdcaRegs.ADCCTL1.bit.INTPULSEPOS = 1;         // Set pulse um ciclo antes do
resultado
    AdcaRegs.ADCCTL1.bit.ADCPWDNZ = 1;           // power up the ADC
    DELAY_US(1000);                             // delay for 1ms to allow ADC time
to power up
    // Determine minimum acquisition window (in SYSCLKS) based on resolution
    if(ADC_RESOLUTION_12BIT == AdcaRegs.ADCCTL2.bit.RESOLUTION)
        acqps = 14;                            // 75ns
    else                                     // resolution is 16-bit
        acqps = 63;
    // SOC0 Config
    AdcaRegs.ADCSOC0CTL.bit.CHSEL = 2;          //sample window is 15 SYSCLK
    AdcaRegs.ADCSOC0CTL.bit.ACQPS = acqps;
cycles
    AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 0x01;      //trigger on ePWM7 SOCA
    // SOC1 Config
    AdcaRegs.ADCSOC1CTL.bit.CHSEL = 3;          //sample window is 15 SYSCLK
    AdcaRegs.ADCSOC1CTL.bit.ACQPS = acqps;
cycles
    AdcaRegs.ADCSOC1CTL.bit.TRIGSEL = 0x01;      //trigger on ePWM7 SOCA
    //ADC Interrupt 1 and 2 Selection Register
    AdcaRegs.ADCINTSEL1N2.bit.INT1SEL = 1;        // end of SOC1 will set INT1 flag
    AdcaRegs.ADCINTSEL1N2.bit.INT1E = 1;          // enable INT1 flag
    AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;        // make sure INT1 flag is cleared

    // ***** ADC-C ***** //
    AdccRegs.ADCCTL2.bit.PRESCALE = 6;           // set ADCCLK divider to /4
    AdcSetMode(ADC_ADCC, ADC_RESOLUTION_16BIT, ADC_SIGNALMODE_DIFFERENTIAL);
    AdccRegs.ADCCTL1.bit.INTPULSEPOS = 1;         // Set pulse um ciclo antes do
resultado
    AdccRegs.ADCCTL1.bit.ADCPWDNZ = 1;           // power up the ADC
    DELAY_US(1000);                             // delay for 1ms to allow ADC time
to power up
    // Determine minimum acquisition window (in SYSCLKS) based on resolution
    if(ADC_RESOLUTION_12BIT == AdccRegs.ADCCTL2.bit.RESOLUTION)
        acqps = 14;                            // 75ns
    else                                     // resolution is 16-bit

```

```

        acqps = 63;
// 320ns
// SOC0 Config
AdccRegs.ADCSOC0CTL.bit.CHSEL = 2;
AdccRegs.ADCSOC0CTL.bit.ACQPS = acqps;           //sample window is 15 SYSCLK
cycles
    AdccRegs.ADCSOC0CTL.bit.TRIGSEL = 0x01;         //trigger on ePWM7 SOCA
    // SOC1 Config
    AdccRegs.ADCSOC1CTL.bit.CHSEL = 3;
    AdccRegs.ADCSOC1CTL.bit.ACQPS = acqps;           //sample window is 15 SYSCLK
cycles
    AdccRegs.ADCSOC1CTL.bit.TRIGSEL = 0x01;         //trigger on ePWM7 SOCA
    // SOC2 Config
    AdccRegs.ADCSOC2CTL.bit.CHSEL = 4;
    AdccRegs.ADCSOC2CTL.bit.ACQPS = acqps;           //sample window is 15 SYSCLK
cycles
    AdccRegs.ADCSOC2CTL.bit.TRIGSEL = 0x01;         //trigger on ePWM7 SOCA
    // SOC3 Config
    AdccRegs.ADCSOC3CTL.bit.CHSEL = 5;
    AdccRegs.ADCSOC3CTL.bit.ACQPS = acqps;           //sample window is 15 SYSCLK
cycles
    AdccRegs.ADCSOC3CTL.bit.TRIGSEL = 0x01;         //trigger on ePWM7 SOCA
//ADC Interrupt 1 and 2 Selection Register
AdccRegs.ADCINTSEL1N2.bit.INT1SEL = 3;           // end of SOC1 will set INT1 flag
AdccRegs.ADCINTSEL1N2.bit.INT1E = 1;              // enable INT1 flag
AdccRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;            // make sure INT1 flag is cleared

EDIS;
}

//*****
*****void Setup_DAC(void)
{
    EALLOW;
    CpuSysRegs.PCLKCR16.bit.DAC_A = 1;
    CpuSysRegs.PCLKCR16.bit.DAC_B = 1;

    // ***** DAC-A ***** //
    DacaRegs.DACCTL.bit.SYNCSEL = 0x06;           // EPWM7SYNCPER signal will update
the DACVALA register
    DacaRegs.DACCTL.bit.LOADMODE = 0x01;          // Load on next EPWMSYNCPER
specified by SYNCSEL
    DacaRegs.DACCTL.bit.DACREFSEL = 0x01;          // ADC VREFHI/VSSA are the
reference voltages
    DacaRegs.DACVALS.bit.DACVALS = 0;
    DacaRegs.DACOUTEN.bit.DACOUTEN = 1;           // DAC output is enabled
    DacaRegs.DACLOCK.all = 0x00;

    // ***** DAC-B ***** //
    DacbRegs.DACCTL.bit.SYNCSEL = 0x06;           // EPWM7SYNCPER signal will update
the DACVALA register
    DacbRegs.DACCTL.bit.LOADMODE = 0x01;          // Load on next EPWMSYNCPER
specified by SYNCSEL
    DacbRegs.DACCTL.bit.DACREFSEL = 0x01;          // ADC VREFHI/VSSA are the
reference voltages
    DacbRegs.DACVALS.bit.DACVALS = 0;
}

```

```
DacbRegs.DACOUTEN.bit.DACOUTEN = 1;      // DAC output is enabled
DacbRegs.DACLOCK.all = 0x00;

EDIS;
}
```

Apêndice H

Código implementado para o *Master* (*Function_Definition.h*)

```
/*
 * Function_Definition.h
 *
 * Created on: 26/05/2022
 * Author: Ricardo Coelho
 */

#ifndef FUNCTION_DEFINITION_H_
#define FUNCTION_DEFINITION_H_
#include "F28x_Project.h"

void slave_counter(void);
void slave_config(Uint16 total_slaves);
void slave_dc_bus(Uint16 total_slaves);
Uint16 dc_bus_check(Uint16 send_array[], Uint16 total_slaves);
float32 pll(float32 v_grid, float32 * pll_uni, char start);
float32 current_control(float32 i_grid, float32 pll_uni, float32 amp_current, char start);
float32 current_control_pred(float32 v_grid, float32 i_grid, float32 pll_uni,
float32 amp_current, char start);
void send_pwm_data(Uint32 pi_ref);
Uint16 error_check(void);
void stop(void);

#endif /* FUNCTION_DEFINITION_H_ */
```

Apêndice I

Código implementado para o *Master* (*Function_Definition.c*)

```
/*
 * Function_Definition.c
 *
 * Created on: 26/05/2022
 * Author: Ricardo Coelho
 */

//*****
// Includes
//*****
#include "Function_Definition.h"
#include "math.h"

//*****
// Defines
//*****
#define DC_Bus_Max          65
#define DC_Bus_Min          55
#define PWM_Period_Max      (5000*0.9)
#define PWM_Period_Min      (5000*0.1)
#define fsampling            5000
#define fsignal              50

#define Pi                  3.14159265
#define w0                  (2 * Pi * fsignal)
#define Ts                  (1.0 / fsampling)
#define kp_p                100
#define ki_p                (50 * (1.0 / fsampling))
#define ki_a                (100 * (1.0 / fsampling))

float32 Kp = 1000;
float32 Ki = 40;
float32 auxL = 0.001;
/*
float32 Kp = 40;
float32 Ki = 3;
*/

//*****
*****
```

```

//*****
*****void slave_counter(void){
    int16 index = 0;
    Uint16 send_array[] = {0x0000, 0x0001, 0x0024};      // {broadcast, count slave,
$}

    while (index < 3){
        SpiaRegs.SPITXBUF = send_array[index];
        index++;
        //DELAY_US(1000);
    }
}

//*****
*****void slave_config(Uint16 total_slaves){
    int16 index = 0;
    Uint16 phase_delay = 0;
    Uint16 send_array[50] = {0};

    send_array[0]          = 0x0000;                  // {broadcast, config slave, $}
    send_array[1]          = 0x0002;
    send_array[total_slaves+2] = 0x0024;

    phase_delay = 360 / total_slaves;

    for(index = 0; index < total_slaves; index++){
        send_array[index+2] = phase_delay * index;
    }
    index = 0;

    while (index < total_slaves+3){
        SpiaRegs.SPITXBUF = send_array[index];
        index++;
        //DELAY_US(1000);
    }
}

//*****
*****void slave_dc_bus(Uint16 total_slaves){
    int16 index = 0;
    static Uint16 send_array[50] = {0};

    send_array[0]          = 0x0000;                  // {broadcast, config slave, $}
    send_array[1]          = 0x0003;
    send_array[total_slaves+2] = 0x0024;

    while (index < total_slaves+3){
        SpiaRegs.SPITXBUF = send_array[index];
        index++;
        //DELAY_US(1000);
    }
}

```



```

    return int_a;
}

//*****
***** 
//*****
***** 
float32 current_control(float32 i_grid, float32 pll_uni, float32 amp_current, char start){
    static float32 i_ref, i_error, i_error_sum, pi_ref;

    if(start){
        i_ref = pll_uni * amp_current;
        //i_grid = (float32)(-(AdccResultRegs.ADCRESULT2 - 32768)) * 17 / 32768;
        i_error = i_ref - i_grid;
        i_error_sum = i_error_sum + i_error;

        if(i_error_sum > 100000) i_error_sum = 100000;
        if(i_error_sum < -100000) i_error_sum = -100000;

        pi_ref = (i_error * Kp) + (i_error_sum * Ki) + 2500;
        if(pi_ref > PWM_Period_Max) pi_ref = PWM_Period_Max;
        else if(pi_ref < PWM_Period_Min) pi_ref = PWM_Period_Min;
    }
    else{
        i_error_sum = 0;
    }
    return pi_ref;
}

//*****
***** 
//*****
***** 
float32 current_control_pred(float32 v_grid, float32 i_grid, float32 pll_uni,
float32 amp_current, char start){
    static float32 i_ref, i_ref_ant, pi_ref;

    i_ref = pll_uni * amp_current;
    pi_ref = (v_grid+auxL*fsampling*(2*i_ref-i_ref_ant-i_grid))*(2500.0/(34.0*3.0-9.0))+2500;
    i_ref_ant = i_ref;

    return pi_ref;
}

void send_pwm_data(Uint32 pi_ref){
    Uint32 first_aux, second_aux, third_aux;
    Uint32 previous_clock;

    previous_clock = ((Uint32)GpioDataRegs.GPATD.bit.GPIO16)<<16;

    if(pi_ref >= PWM_Period_Max) pi_ref = PWM_Period_Max;
    if(pi_ref <= PWM_Period_Min) pi_ref = PWM_Period_Min;
}

```

```

pi_ref = pi_ref>>1;

first_aux = ((pi_ref<<4) & 0x00000FF0);
second_aux = ((pi_ref<<16) & 0x0F000000);
third_aux = first_aux | second_aux;
third_aux = third_aux | previous_clock;

GpioDataRegs.GPADAT.all = third_aux;
}

//*****
*****error_check(void){
    Uint16 error;

    error = GpioDataRegs.GPDDAT.bit.GPIO104;

    return error;
}

//*****
*****void stop(void){
}

```

Apêndice J

Código implementado para os *Slaves* (*main.c*)

```
//#####
// FILE: main.c
// TITLE: Master Code
// CODE: -
// -
//#####
// AUTHOR: Ricardo Coelho
// DATE: 27_01_2022
// VERSION: v1.1
//#####

//*****
// Includes
//*****
#include "F28x_Project.h"
//C:\ti\controlSUITE\device_support\F2837xD\v210\F2837xD_common\include
#include "string.h"
#include "stdio.h"
#include "stdlib.h"
#include "math.h"
#include "time.h"
#include "Peripheral_Setup.h"
#include "Function_Definition.h"

//*****
// Defines
//*****
#define pi 3.14159
#define DOIS_PI_F 314.1592

//*****
// Global variables
//*****
// Control variables
Uint16 control_state;
Uint16 led_blink_counter;
Uint16 flag_blink;
Uint16 first_case;
Uint16 flag_xint1;
Uint16 flag_xint3;
Uint16 flag_xint5;
Uint16 flag_timer0_isr;
Uint16 flag_receive_counter;
```

```

Uint16 flag_receive_config;
Uint16 flag_dc_bus;
Uint16 flag_dc_bus_check;
Uint16 master_enable;
Uint16 error;
Uint16 n_slaves;
Uint16 receive_index;
Uint16 phase_delay;
Uint16 counter_compare;
Uint32 received_parallel_com;
Uint16 aux;
float32 aux_float;

// SPI variables
Uint16 index;
Uint16 receive_array[50] = {0};
Uint16 flag_receive_array;
Uint16 sdata_a; // send data
Uint16 sdata_c; // send data
Uint16 rdata_a; // received data
Uint16 rdata_c; // received data

//*****
// Prototype statements for functions
//*****

void initialize(void);
interrupt void xint1_isr(void);
interrupt void xint3_isr(void);
interrupt void xint4_isr(void);
interrupt void xint5_isr(void);
interrupt void timer0_isr(void);
interrupt void spiaRxFifoIsr(void);
interrupt void spicRxFifoIsr(void);

int main(void){

    //
    // Step 1. Initialize System Control:
    //
    // PLL, WatchDog, enable Peripheral Clocks
    // This example function is found in the F2837xD_SysCtrl.c file.
    InitSysCtrl();

    //
    // Step 2. Initialize GPIO:
    //
    // This example function is found in the F2837xD_Gpio.c file and
    // illustrates how to set the GPIO to it's default state.
    // Setup only the GP I/O only for SPI-A functionality
    InitSpiaGpio();

    //
    // Step 3. Initialize PIE vector table:
    //
    // Disable and clear all CPU interrupts
    DINT;
    IER = 0x0000;
}

```

```

IFR = 0x0000;

// Initialize PIE control registers to their default state:
// This function is found in the F2837xD_PieCtrl.c file.
InitPieCtrl();

// Initialize the PIE vector table with pointers to the shell Interrupt
// Service Routines (ISR).
// This will populate the entire table, even if the interrupt
// is not used in this example. This is useful for debug purposes.
// The shell ISR routines are found in F2837xD_DefaultIsr.c.
// This function is found in F2837xD_PieVect.c.
InitPieVectTable();

// Interrupts that are used in this example are re-mapped to
// ISR functions found within this file.
EALLOW; // This is needed to write to EALLOW protected registers
CpuSysRegs.PCLKCR0.bit.CPUTIMER0 = 1; // Enable timer clock

PieVectTable.XINT1_INT = &xint1_isr;
PieVectTable.XINT3_INT = &xint3_isr;
PieVectTable.XINT4_INT = &xint4_isr;
PieVectTable.XINT5_INT = &xint5_isr;
PieVectTable.TIMER0_INT = &timer0_isr;
PieVectTable.SPIA_RX_INT = &spiaRxFifoIsr;
PieVectTable.SPIC_RX_INT = &spicRxFifoIsr;
EDIS; // This is needed to disable write to EALLOW protected registers

//
// Step 4. Initialize the Device Peripherals:
//
Setup_GPIO();
Setup_Timer();
Setup_ePWM();
Setup_SPI();
//Setup_DAC();
Setup_External_Interrupt();

//
// Step 5. User specific code, enable interrupts:
//
// Initialize
initialize();
control_state = 0x0000;
led_blink_counter = 0x0000;
sdata_a = 0x0000;
sdata_c = 0x0000;
//aux      = 0x0000;
flag_xint1    = 0x0000;
flag_xint3    = 0x0000;

// Enable interrupts required for this example
PieCtrlRegs.PIECTRL.bit.ENPIE = 1; // Enable the PIE block

PieCtrlRegs.PIEIER1.bit.INTx4 = 1; // Enable PIE Group 1, INT 4 (XINT1
interrupt)
PieCtrlRegs.PIEIER1.bit.INTx7 = 1; // Enable PIE Group 1, INT 7 (TIMER0
interrupt)

```

```

    PieCtrlRegs.PIEIER6.bit.INTx1 = 1;           // Enable PIE Group 6, INT 1 (SPIA_RX
interrupt)
    PieCtrlRegs.PIEIER6.bit.INTx9 = 1;           // Enable PIE Group 6, INT 9 (SPIC_RX
interrupt)
    PieCtrlRegs.PIEIER12.bit.INTx1 = 1;          // Enable PIE Group 12, INT 1 (XINT3
interrupt)
    PieCtrlRegs.PIEIER12.bit.INTx2 = 1;          // Enable PIE Group 12, INT 2 (XINT4
interrupt)
    PieCtrlRegs.PIEIER12.bit.INTx3 = 1;          // Enable PIE Group 12, INT 3 (XINT5
interrupt)
    IER |= M_INT1;                            // Enable CPU INT1
    IER |= M_INT6;                            // Enable CPU INT6
    IER |= M_INT12;                           // Enable CPU INT12
    EINT;                                    // Enable Global interrupt INTM
    ERTM;                                    // Enable Global realtime interrupt

DBGM

CpuTimer0Regs.TCR.all = 0x4001;             // Start Timer0

//  

// Step 6. IDLE loop. Just sit and loop forever (optional):
//  

while(1){
    if(flag_timer0_isr == 1){
        switch(control_state){

            case 0:
                // Initial action
                if(first_case == 0){
                    initialize();
                    first_case = 1;
                }
                // Case action
                if(flag_blink == 1){
                    GpioDataRegs.GPEDAT.bit.GPIO139 =
!(GpioDataRegs.GPADAT.bit.GPIO16);
                    GpioDataRegs.GPBDAT.bit.GPIO56 =
!(GpioDataRegs.GPADAT.bit.GPIO16);
                    GpioDataRegs.GPDDAT.bit.GPIO97 =
!(GpioDataRegs.GPADAT.bit.GPIO16);
                    GpioDataRegs.GPCDAT.bit.GPIO94 =
!(GpioDataRegs.GPADAT.bit.GPIO16);
                    flag_blink = 0;
                }
                else if(led_blink_counter > 25100){
                    GpioDataRegs.GPETOGGLE.bit.GPIO139 = 1;
                    GpioDataRegs.GPBTOGGLE.bit.GPIO56 = 1;
                    GpioDataRegs.GPDTOGGLE.bit.GPIO97 = 1;
                    GpioDataRegs.GPCTOGGLE.bit.GPIO94 = 1;
                    led_blink_counter = 0;
                }
                // Condition to switch cases
                if(flag_xint1 == 1){
                    control_state = 1;
                    first_case = 0;
                    flag_xint1 = 0;
                }
            break;
        }
    }
}

```

```

case 1:
    // Initial action
    if(first_case == 0){
        GpioDataRegs.GPESET.bit.GPIO139 = 1;
        GpioDataRegs.GPBCLEAR.bit.GPIO56 = 1;
        GpioDataRegs.GPDCLEAR.bit.GPIO97 = 1;
        GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
        first_case = 1;
    }
    // Condition to switch cases
    if(flag_xint3 == 1){
        control_state = 0;
        first_case = 0;
        flag_xint3 = 0;
    }
    else if(flag_receive_counter == 1){
        control_state = 2;
        first_case = 0;
        flag_receive_counter = 0;
    }
    break;

case 2:
    // Initial action
    if(first_case == 0){
        GpioDataRegs.GPECLEAR.bit.GPIO139 = 1;
        GpioDataRegs.GPBSET.bit.GPIO56 = 1;
        GpioDataRegs.GPDCLEAR.bit.GPIO97 = 1;
        GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
        slave_counter(receive_array, receive_index);
        first_case = 1;
    }
    // Condition to switch cases
    if(flag_xint3 == 1){
        control_state = 0;
        first_case = 0;
        flag_xint3 = 0;
    }
    else if(flag_receive_config == 1){
        control_state = 3;
        first_case = 0;
        flag_receive_config = 0;
    }
    break;

case 3:
    // Initial action
    if(first_case == 0){
        GpioDataRegs.GPESET.bit.GPIO139 = 1;
        GpioDataRegs.GPBSET.bit.GPIO56 = 1;
        GpioDataRegs.GPDCLEAR.bit.GPIO97 = 1;
        GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
        phase_delay = slave_config(receive_array, receive_index,
n_slaves);
        first_case = 1;
    }

```

```

// Condition to switch cases
if(flag_xint3 == 1){
    control_state = 0;
    first_case = 0;
    flag_xint3 = 0;

}
else if(flag_dc_bus == 1){
    control_state = 4;
    first_case = 0;
    flag_dc_bus = 0;
}
break;

case 4:
// Initial action
if(first_case == 0){
    GpioDataRegs.GPECLEAR.bit.GPIO139 = 1;
    GpioDataRegs.GPBCLEAR.bit.GPIO56 = 1;
    GpioDataRegs.GPDSET.bit.GPIO97 = 1;
    GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
    GpioDataRegs.GPBCLEAR.bit.GPIO32 = 1;
    slave_dc_bus(receive_array, receive_index, n_slaves);
    first_case = 1;
}
// Case action
if(flag_xint3 == 1){
    control_state = 0;
    first_case = 0;
    flag_xint3 = 0;
}
else if(master_enable == 1){
    control_state = 5;
    first_case = 0;
    flag_dc_bus = 0;
}
break;

case 5:
// Initial action
if(first_case == 0){
    GpioDataRegs.GPESET.bit.GPIO139 = 1;
    GpioDataRegs.GPBCLEAR.bit.GPIO56 = 1;
    GpioDataRegs.GPDSET.bit.GPIO97 = 1;
    GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
    GpioDataRegs.GPBSET.bit.GPIO32 = 1;
    start_pwm();
    first_case = 1;
}
// Case action
if(flag_xint5 == 1){
    counter_compare = receive_pwm_data(received_parallel_com);
    setup_pwm_phase(phase_delay, n_slaves);
    update_pwm(counter_compare);
    flag_xint5 = 0;
    //GpioDataRegs.GPACLEAR.bit.GPIO19 = 1;
}
// Condition to switch cases
if(master_enable == 0){

```

```

        control_state = 4;
        stop_pwm();
        first_case = 0;
    }
    break;

case 6:
    break;

default:
    break;

} //end switch(control_state)
flag_timer0_isr = 0;
} //end if(flag_timer0_isr == 1)
} //end while(1)
} //end main()

//*****
// adcaIsr - ISR for ADC-A1
//*****
void initialize(void)
{
    GpioDataRegs.GPECLEAR.bit.GPIO139 = 1;
    GpioDataRegs.GPBCLEAR.bit.GPIO56 = 1;
    GpioDataRegs.GPDCLEAR.bit.GPIO97 = 1;
    GpioDataRegs.GPCCLEAR.bit.GPIO94 = 1;
    GpioDataRegs.GPBCLEAR.bit.GPIO32 = 1;
    // Control variables
    //control_state      = 0;
    led_blink_counter   = 0;
    flag_blink          = 0;
    //first_case         = 0;
    flag_xint1          = 0;
    flag_xint3          = 0;
    flag_xint5          = 0;
    flag_timer0_isr     = 0;
    flag_receive_counter = 0;
    flag_receive_config = 0;
    flag_dc_bus         = 0;
    flag_dc_bus_check   = 0;
    //master_enable       = 0;
    //error               = 0;
    n_slaves             = 0;
    aux                  = 0;
    aux_float            = 0.0;
    // SPI variables
    index                = 0;
    memset(receive_array, 0, sizeof(receive_array));
    flag_receive_array   = 0;
    sdata_a              = 0;
    sdata_c              = 0;
    rdata_a              = 0;
    rdata_c              = 0;
    stop_pwm();
}

//*****

```

```

// xint1_isr - External Interrupt 1 ISR
//*****
interrupt void xint1_isr(void)
{
    flag_xint1 = 1;

    // Acknowledge this interrupt to get more from group 1
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

//*****
// xint3_isr - External Interrupt 3 ISR
//*****
interrupt void xint3_isr(void)
{
    flag_xint3 = 1;

    // Acknowledge this interrupt to get more from group 1
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

//*****
// xint4_isr - External Interrupt 4 ISR
//*****
interrupt void xint4_isr(void)
{
    master_enable = GpioDataRegs.GPDDAT.bit.GPIO105;

    // Acknowledge this interrupt to get more from group 1
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

//*****
// xint5_isr - External Interrupt 5 ISR
//*****
interrupt void xint5_isr(void)
{
    if(control_state == 5){
        received_parallel_com = GpioDataRegs.GPADAT.all;
        //GpioDataRegs.GPASET.bit.GPIO19 = 1;
        flag_xint5 = 1;
    }
    else{
        flag_blink = 1;
        led_blink_counter = 0;
    }

    // Acknowledge this interrupt to get more from group 1
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

```

```

//*****
// timer0_isr - Timer0 ISR
//*****
interrupt void timer0_isr(void)
{
    flag_timer0_isr = 1;
    led_blink_counter++;

    // Acknowledge this interrupt to receive more interrupts from group 1
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

//*****
// spiaRxFifoIsr - ISR for SPI-A receive FIFO
//*****
interrupt void spiaRxFifoIsr(void)
{
    rdata_a=SpiaRegs.SPIRXBUF;           // Read data

    SpiaRegs.SPIFFRX.bit.RXFFOVFCLR=1;   // Clear Overflow flag
    SpiaRegs.SPIFFRX.bit.RXFFINTCLR=1;   // Clear Interrupt flag
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

//*****
// spicRxFifoIsr - ISR for SPI-C receive FIFO
//*****
interrupt void spicRxFifoIsr(void)
{
    rdata_c = SpicRegs.SPIRXBUF;
    receive_array[index] = rdata_c;      // Read data
    index++;

    if(rdata_c == 0x0024){
        if(receive_array[1] == 1){
            flag_receive_counter = 1;
            receive_index = index - 1;
        }
        else if(receive_array[1] == 2){
            flag_receive_config = 1;
            n_slaves = index - 3;
        }
        else if (receive_array[1] == 3) flag_dc_bus = 1;
        index = 0;
    }

    SpicRegs.SPIFFRX.bit.RXFFOVFCLR=1;   // Clear Overflow flag
    SpicRegs.SPIFFRX.bit.RXFFINTCLR=1;   // Clear Interrupt flag
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP1;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP6;
    PieCtrlRegs.PIEACK.all |= PIEACK_GROUP12;
}

```

Apêndice K

Código implementado para os *Slaves* (*Peripheral_Setup.h*)

```
/*
 * Peripheral_Setup.h
 *
 * Created on: 23 de jul de 2020
 * Author: waner
 */

#ifndef PERIPHERAL_SETUP_H_
#define PERIPHERAL_SETUP_H_
#include "F28x_Project.h"

void Setup_GPIO(void);
void Setup_Timer(void);
void Setup_External_Interrupt(void);
void Setup_SPI(void);
void Setup_ePWM(void);
void Setup_DAC(void);

#endif /* PERIPHERAL_SETUP_H_ */
```

Apêndice L

Código implementado para os *Slaves* (*Peripheral_Setup.c*)

```
/*
 * Peripheral_Setup.c
 *
 * Created on: 23 de jul de 2020
 * Author: waner
 */

#include "Peripheral_Setup.h"

//*****
***** GpioCtrlRegs.GPAMUX1.all = 0X0000;
***** GpioCtrlRegs.GPAMUX2.all = 0X0000;
***** GpioCtrlRegs.GPDIR.all = 0x00000000; //1: Configures pin as input.
***** GpioCtrlRegs.GPACSEL1.all = GPIO_MUX_CPU1; //GPIO is controlled by CPU1
***** GpioCtrlRegs.GPACSEL2.all = GPIO_MUX_CPU1; //GPIO is controlled by CPU1
***** GpioCtrlRegs.GPACSEL3.all = GPIO_MUX_CPU1; //GPIO is controlled by CPU1
***** GpioCtrlRegs.GPACSEL4.all = GPIO_MUX_CPU1; //GPIO is controlled by CPU1
// GPIO14 CLOCK
GpioCtrlRegs.GPAGMUX2.bit.GPIO16 = 0;
GpioCtrlRegs.GPAMUX2.bit.GPIO16 = 0;
GpioCtrlRegs.GPAPUD.bit.GPIO16 = 1;
GpioCtrlRegs.GPADIR.bit.GPIO16 = 0;
GpioCtrlRegs.GPACSEL3.bit.GPIO16 = GPIO_MUX_CPU1;
GpioCtrlRegs.GPAQSEL2.bit.GPIO16 = 3;
InputXbarRegs.INPUT5SELECT = 16;

// ***** SPI *****
// *** SPI-A ***
// Enable internal pull-up for the selected pins
GpioCtrlRegs.GPBPU00.bit.GPIO58 = 0; // Enable pull-up on GPIO58 (SPISIMOA)
GpioCtrlRegs.GPBPU00.bit.GPIO59 = 0; // Enable pull-up on GPIO59(SPIISOMIA)
GpioCtrlRegs.GPBPU00.bit.GPIO60 = 0; // Enable pull-up on GPIO60 (SPICLKA)
GpioCtrlRegs.GPBPU00.bit.GPIO61 = 0; // Enable pull-up on GPIO61 (SPISTEA)
// Set qualification for selected pins to asynch only
GpioCtrlRegs.GPBQSEL2.bit.GPIO58 = 3; // Asynch input GPIO58 (SPISIMOA)
GpioCtrlRegs.GPBQSEL2.bit.GPIO59 = 3; // Asynch input GPIO59 (SPIISOMIA)
```

```

GpioCtrlRegs.GPBQSEL2.bit.GPIO60 = 3; // Asynch input GPIO60 (SPICLKA)
GpioCtrlRegs.GPBQSEL2.bit.GPIO61 = 3; // Asynch input GPIO61 (SPISTEA)
// Configure SPI-A pins using GPIO regs
GpioCtrlRegs.GPBMUX2.bit.GPIO58 = 3; // Configure GPIO58 as SPISIMOA
GpioCtrlRegs.GPBMUX2.bit.GPIO59 = 3; // Configure GPIO59 as SPISOMIA
GpioCtrlRegs.GPBMUX2.bit.GPIO60 = 3; // Configure GPIO60 as SPICLKA
GpioCtrlRegs.GPBMUX2.bit.GPIO61 = 3; // Configure GPIO61 as SPISTEA
GpioCtrlRegs.GPBGMUX2.bit.GPIO58 = 3; // Configure GPIO58 as SPISIMOA
GpioCtrlRegs.GPBGMUX2.bit.GPIO59 = 3; // Configure GPIO59 as SPISOMIA
GpioCtrlRegs.GPBGMUX2.bit.GPIO60 = 3; // Configure GPIO60 as SPICLKA
GpioCtrlRegs.GPBGMUX2.bit.GPIO61 = 3; // Configure GPIO61 as SPISTEA

// *** SPI-C ***
// Enable internal pull-up for the selected pins
GpioCtrlRegs.GPDPU.D.bit.GPIO122 = 0; // Enable pull-up on GPIO122
(SPISIMOC)
GpioCtrlRegs.GPDPU.D.bit.GPIO123 = 0; // Enable pull-up on GPIO123
(SPISOMIC)
GpioCtrlRegs.GPDPU.D.bit.GPIO124 = 0; // Enable pull-up on GPIO124
(SPICLKC)
GpioCtrlRegs.GPDPU.D.bit.GPIO125 = 0; // Enable pull-up on GPIO125
(SPISTEC)
// Set qualification for selected pins to asynch only
GpioCtrlRegs.GPDQSEL2.bit.GPIO122 = 3; // Asynch input GPIO122 (SPISIMOC)
GpioCtrlRegs.GPDQSEL2.bit.GPIO123 = 3; // Asynch input GPIO123 (SPISOMIC)
GpioCtrlRegs.GPDQSEL2.bit.GPIO124 = 3; // Asynch input GPIO124 (SPICLKC)
GpioCtrlRegs.GPDQSEL2.bit.GPIO125 = 3; // Asynch input GPIO125 (SPISTEC)
// Configure SPI-C pins using GPIO regs
GpioCtrlRegs.GPDMUX2.bit.GPIO122 = 2; // Configure GPIO122 as SPISIMOC
GpioCtrlRegs.GPDMUX2.bit.GPIO123 = 2; // Configure GPIO123 as SPISOMIC
GpioCtrlRegs.GPDMUX2.bit.GPIO124 = 2; // Configure GPIO124 as SPICLKC
GpioCtrlRegs.GPDMUX2.bit.GPIO125 = 2; // Configure GPIO125 as SPISTEC
GpioCtrlRegs.GPDGMUX2.bit.GPIO122 = 1; // Configure GPIO122 as SPISIMOC
GpioCtrlRegs.GPDGMUX2.bit.GPIO123 = 1; // Configure GPIO123 as SPISOMIC
GpioCtrlRegs.GPDGMUX2.bit.GPIO124 = 1; // Configure GPIO124 as SPICLKC
GpioCtrlRegs.GPDGMUX2.bit.GPIO125 = 1; // Configure GPIO125 as SPISTEC

// ***** PWM **** //
// PWM-1A
GpioCtrlRegs.GPAGMUX1.bit.GPIO0 = 0;
GpioCtrlRegs.GPAMUX1.bit.GPIO0 = 1;
GpioCtrlRegs.GPAPUD.bit.GPIO0 = 1;
// PWM-1B
GpioCtrlRegs.GPAGMUX1.bit.GPIO1 = 0;
GpioCtrlRegs.GPAMUX1.bit.GPIO1 = 1;
GpioCtrlRegs.GPAPUD.bit.GPIO1 = 1;
// PWM-2A
GpioCtrlRegs.GPAGMUX1.bit.GPIO2 = 0;
GpioCtrlRegs.GPAMUX1.bit.GPIO2 = 1;
GpioCtrlRegs.GPAPUD.bit.GPIO2 = 1;
// PWM-2B
GpioCtrlRegs.GPAGMUX1.bit.GPIO3 = 0;
GpioCtrlRegs.GPAMUX1.bit.GPIO3 = 1;
GpioCtrlRegs.GPAPUD.bit.GPIO3 = 1;
// PWM-8A
GpioCtrlRegs.GPAGMUX1.bit.GPIO14 = 0;
GpioCtrlRegs.GPAMUX1.bit.GPIO14 = 1;
GpioCtrlRegs.GPAPUD.bit.GPIO14 = 1;
// PWM-8B

```

```

GpioCtrlRegs.GPAGMUX1.bit.GPIO15      = 0;
GpioCtrlRegs.GPAMUX1.bit.GPIO15      = 1;
GpioCtrlRegs.GPAPUD.bit.GPIO15      = 1;

// ***** GPIOs **** //
// Enable DSC (output)
GpioCtrlRegs.GPBMUX1.bit.GPIO32      = 0;
GpioCtrlRegs.GPBDIR.bit.GPIO32      = 1;
GpioCtrlRegs.GPBCSEL1.bit.GPIO32      = GPIO_MUX_CPU1; //GPIO1 is controlled
by CPU1
    // LED - State 0 (output)
    GpioCtrlRegs.GPEMUX1.bit.GPIO139     = 0;
    GpioCtrlRegs.GPEDIR.bit.GPIO139     = 1;
    GpioCtrlRegs.GPECSEL2.bit.GPIO139     = GPIO_MUX_CPU1; //GPIO1 is controlled
by CPU1
    // LED - State 1 (output)
    GpioCtrlRegs.GPBMUX2.bit.GPIO56      = 0;
    GpioCtrlRegs.GPBDIR.bit.GPIO56      = 1;
    GpioCtrlRegs.GPBQSEL2.bit.GPIO56      = GPIO_MUX_CPU1; //GPIO1 is controlled
by CPU1
    // LED - State 2 (output)
    GpioCtrlRegs.GPDMUX1.bit.GPIO97      = 0;
    GpioCtrlRegs.GPDDIR.bit.GPIO97      = 1;
    GpioCtrlRegs.GPDCSEL1.bit.GPIO97      = GPIO_MUX_CPU1; //GPIO1 is controlled
by CPU1
    // LED - State 3 (output)
    GpioCtrlRegs.GPCMUX2.bit.GPIO94      = 0;
    GpioCtrlRegs.GPCDIR.bit.GPIO94      = 1;
    GpioCtrlRegs.GPCQSEL2.bit.GPIO94      = GPIO_MUX_CPU1; //GPIO1 is controlled
by CPU1
    // Button - S1 (input)
    GpioCtrlRegs.GPBMUX1.bit.GPIO41      = 0;
    GpioCtrlRegs.GPBDIR.bit.GPIO41      = 0;
    GpioCtrlRegs.GPBCSEL2.bit.GPIO41      = GPIO_MUX_CPU1; //GPIO1 is controlled
by CPU1
    // Button - S2 (input)
    GpioCtrlRegs.GPBMUX1.bit.GPIO40      = 0;
    GpioCtrlRegs.GPBDIR.bit.GPIO40      = 0;
    GpioCtrlRegs.GPBCSEL2.bit.GPIO40      = GPIO_MUX_CPU1; //GPIO1 is controlled
by CPU1
    // Master Enable (input)
    GpioCtrlRegs.GPDMUX1.bit.GPIO105      = 0;
    GpioCtrlRegs.GPDDIR.bit.GPIO105      = 0;
    GpioCtrlRegs.GPDCSEL2.bit.GPIO105      = GPIO_MUX_CPU1; //GPIO1 is controlled
by CPU1
    // Error (input)
    GpioCtrlRegs.GPDMUX1.bit.GPIO104      = 0;
    GpioCtrlRegs.GPDDIR.bit.GPIO104      = 0;
    GpioCtrlRegs.GPDCSEL2.bit.GPIO104      = GPIO_MUX_CPU1; //GPIO1 is controlled
by CPU1

    // Parallel Communication (output)
    GpioCtrlRegs.GPAMUX2.bit.GPIO19      = 0;
    GpioCtrlRegs.GPADIR.bit.GPIO19      = 1;
    GpioCtrlRegs.GPACSEL3.bit.GPIO19      = GPIO_MUX_CPU1; //GPIO1 is controlled
by CPU1

EDIS;
}

```

```

//*****
*****  

*****  

*****  

void Setup_Timer(void)  

{  

    // *** Timer 0 *** //  

    InitCpuTimers();  

    ConfigCpuTimer(&CpuTimer0, 200, 20); // ConfigCpuTimer(struct CPUTIMER_VARS  

*Timer, float Freq, float Period)  

}  

//*****  

*****  

*****  

*****  

void Setup_External_Interrupt(void)  

{  

    GPIO_SetupXINT1Gpio(41);  

    //GPIO_SetupXINT2Gpio(40);  

    GPIO_SetupXINT3Gpio(40);  

    GPIO_SetupXINT4Gpio(105);  

    GPIO_SetupXINT5Gpio(16);  

    EALLOW;  

    // Configure XINTx           // 0 = Falling edge interrupt; 1 =  

Rising edge interrupt; 3 = Rising and Fallind edge interrupt  

    XintRegs.XINT1CR.bit.POLARITY = 0;      // 0 = Falling edge interrupt  

    XintRegs.XINT3CR.bit.POLARITY = 0;      // 3 = Rising and Fallind edge  

interrupt  

    XintRegs.XINT4CR.bit.POLARITY = 3;      // 3 = Rising and Fallind edge  

interrupt  

    XintRegs.XINT5CR.bit.POLARITY = 3;      // 3 = Rising and Fallind edge  

interrupt  

    // Enable XINTx  

    XintRegs.XINT1CR.bit.ENABLE = 1;        // Enable XINT1  

    XintRegs.XINT3CR.bit.ENABLE = 1;        // Enable XINT3  

    XintRegs.XINT4CR.bit.ENABLE = 1;        // Enable XINT4  

    XintRegs.XINT5CR.bit.ENABLE = 1;        // Enable XINT5  

    EDIS;
}  

//*****  

*****  

*****  

*****  

void Setup_SPI()  

{  

    EALLOW;  

    // *****     SPI-A     ***** //  

    // Initialize SPI FIFO registers  

    SpiaRegs.SPIFFTX.all = 0xE040;  

    SpiaRegs.SPIFFRX.all = 0x2061;  

    SpiaRegs.SPIFFCT.all = 0x0000;
}

```

```

// Set reset low before configuration changes
// Clock polarity (0 == rising, 1 == falling)
// 16-bit character
// Enable loop-back
SpiaRegs.SPICCR.bit.SPISWRESET = 0;
SpiaRegs.SPICCR.bit.CLKPOLARITY = 0;
SpiaRegs.SPICCR.bit.SPICHR = (16 - 1);
SpiaRegs.SPICCR.bit.SPILBK = 0;
// Enable master (0 == slave, 1 == master)
// Enable transmission (Talk)
// Clock phase (0 == normal, 1 == delayed)
// SPI interrupts are disabled
SpiaRegs.SPICTL.bit.MASTER_SLAVE = 1;
SpiaRegs.SPICTL.bit.TALK = 1;
SpiaRegs.SPICTL.bit.CLK_PHASE = 0;
SpiaRegs.SPICTL.bit.SPIINTENA = 1;
// Set the baud rate
SpiaRegs.SPIBRR.bit.SPI_BIT_RATE = ((200E6 / 4) / 500E3) - 1;
// Set FREE bit
// Halting on a breakpoint will not halt the SPI
SpiaRegs.SIPPRI.bit.FREE = 1;
// Release the SPI from reset
SpiaRegs.SPICCR.bit.SPISWRESET = 1;

// ***** SPI-C *****
// Initialize SPI FIFO registers
SpicRegs.SPIFFTX.all = 0xE040;
SpicRegs.SPIFFRX.all = 0x2061;
SpicRegs.SPIFFCT.all = 0x0000;
// Set reset low before configuration changes
// Clock polarity (0 == rising, 1 == falling)
// 16-bit character
// Enable loop-back
SpicRegs.SPICCR.bit.SPISWRESET = 0;
SpicRegs.SPICCR.bit.CLKPOLARITY = 0;
SpicRegs.SPICCR.bit.SPICHR = (16 - 1);
SpicRegs.SPICCR.bit.SPILBK = 0;
// Enable master (0 == slave, 1 == master)
// Enable transmission (Talk)
// Clock phase (0 == normal, 1 == delayed)
// SPI interrupts are disabled
SpicRegs.SPICTL.bit.MASTER_SLAVE = 0;
SpicRegs.SPICTL.bit.TALK = 1;
SpicRegs.SPICTL.bit.CLK_PHASE = 0;
SpicRegs.SPICTL.bit.SPIINTENA = 1;
// Set the baud rate
// Calculate BRR: 7-bit baud rate register value
// SPI CLK freq = 500 kHz
// LSPCLK freq = CPU freq / 4 (by default)
// BRR = (LSPCLK freq / SPI CLK freq) - 1
SpicRegs.SPIBRR.bit.SPI_BIT_RATE = ((200E6 / 4) / 500E3) - 1;
// Set FREE bit
// Halting on a breakpoint will not halt the SPI
SpicRegs.SIPPRI.bit.FREE = 1;
// Release the SPI from reset
SpicRegs.SPICCR.bit.SPISWRESET = 1;

EDIS;
}

```

```

//*****
*****void Setup_ePWM(void)
{
    EALLOW;
    CpuSysRegs.PCLKCR0.bit.TBCLKSYNC = 0;
    CpuSysRegs.PCLKCR2.bit.EPWM1 = 1;
    CpuSysRegs.PCLKCR2.bit.EPWM2 = 1;
    CpuSysRegs.PCLKCR2.bit.EPWM8 = 1;

    // ***** PWM-1 *****
    EPwm1Regs.TBPRD = 5000;                                // Set timer period
    (((100Mhz/CLKDIV)/F pwm)/2)
    EPwm1Regs.CMPA.bit.CMPA = 0;                            // Setup compare
    EPwm1Regs.TBPHS.bit.TBPHS = 0;                          // Phase is 0
    EPwm1Regs.TBCTL.bit.SYNCSEL = TB_SYNC_IN;              // Sync down-stream module
    EPwm1Regs.TBCTR = 0x0000;                               // Clear counter
    EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN;        // Count up/down
    EPwm1Regs.TBCTL.bit.PHSEN = TB_ENABLE;                 // Disable phase loading
    EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;               // Clock ratio to SYSCLKOUT
    EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
    //EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;                // Period autoreload
    EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;           // Load registers every ZERO
    EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO_PRD;
    EPwm1Regs.CMPCTL.bit.SHDBMODE = CC_SHADOW;             // Load registers every ZERO
    EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO_PRD;

    // Set actions
    EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR;
    EPwm1Regs.AQCTLA.bit.CAD = AQ_SET;
    EPwm1Regs.AQCTLB.bit.CAU = AQ_SET;
    EPwm1Regs.AQCTLB.bit.CAD = AQ_CLEAR;
    // Dead time config
    EPwm1Regs.DBCTL.bit.OUT_MODE = DB_FULL_ENABLE;         // Enable Dead-band module
    EPwm1Regs.DBCTL.bit.POLSEL = DB_ACTV_HI;               // Active High complementary
    EPwm1Regs.DBRED.all = 10;                              // Set 100 clock ticks of
    rising edge dead time ( x us)
    EPwm1Regs.DBFED.all = 10;                             // Set 100 clock ticks of
    falling edge dead time ( x us)
    // Stop PWM
    EPwm1Regs.DBCTL.bit.POLSEL = DB_ACTV_HI;
    EPwm1Regs.AQCSFRC.bit.CSFA = 1;                      // Forces a continuous low on
    output A
    EPwm1Regs.AQCSFRC.bit.CSFB = 1;                      // Forces a continuous low on
    output B

    // ***** PWM-2 *****
    EPwm2Regs.TBPRD = 5000;                                // Set timer period
    (((100Mhz/CLKDIV)/F pwm)/2)
    EPwm2Regs.CMPA.bit.CMPA = 0;                            // Setup compare
    EPwm2Regs.TBPHS.bit.TBPHS = 0;                          // Phase is 0
    EPwm2Regs.TBCTL.bit.SYNCSEL = TB_SYNC_IN;              // Sync down-stream module
    EPwm2Regs.TBCTR = 0x0000;                               // Clear counter
    EPwm2Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN;        // Count up/down
    EPwm2Regs.TBCTL.bit.PHSEN = TB_ENABLE;                 // Disable phase loading
    EPwm2Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;               // Clock ratio to SYSCLKOUT
    EPwm2Regs.TBCTL.bit.CLKDIV = TB_DIV1;
}

```

```

//EPwm2Regs.TBCTL.bit.PRDLD = TB_SHADOW;           // Period autoreload
EPwm2Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;       // Load registers every ZERO
EPwm2Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO_PRD;
EPwm2Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;        // Load registers every ZERO
EPwm2Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO_PRD;
// Set actions
EPwm2Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm2Regs.AQCTLA.bit.CAD = AQ_CLEAR;
EPwm2Regs.AQCTLB.bit.CAU = AQ_CLEAR;
EPwm2Regs.AQCTLB.bit.CAD = AQ_SET;
// Dead time config
EPwm2Regs.DBCTL.bit.OUT_MODE = DB_FULL_ENABLE;    // Enable Dead-band module
EPwm2Regs.DBCTL.bit.POLSEL = DB_ACTV_HIC;          // Active High complementary
EPwm2Regs.DBRED.all = 10;                           // Set 100 clock ticks of
rising edge dead time ( x us)
EPwm2Regs.DBFED.all = 10;                           // Set 100 clock ticks of
falling edge dead time ( x us)
// Stop PWM
EPwm2Regs.DBCTL.bit.POLSEL = DB_ACTV_HI;
EPwm2Regs.AQCSFRC.bit.CSFA = 1;                    // Forces a continuous low on
output A
EPwm2Regs.AQCSFRC.bit.CSFB = 1;                    // Forces a continuous low on
output B

// ***** PWM-8 **** //
EPwm8Regs.TBPRD = 5000;                           // Set timer period
((100Mhz/CLKDIV)/Epwm)/2
EPwm8Regs.CMPA.bit.CMPA = 0;                       // Setup compare
EPwm8Regs.TBPHS.bit.TBPHS = 0;                     // Phase is 0
EPwm8Regs.TBCTL.bit.SYNCSEL = TB_SYNC_IN;          // Sync down-stream module
EPwm8Regs.TBCTR = 0x0000;                           // Clear counter
EPwm8Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN;    // Count up/down
EPwm8Regs.TBCTL.bit.PHSEN = TB_ENABLE;             // Disable phase loading
EPwm8Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;           // Clock ratio to SYSCLKOUT
EPwm8Regs.TBCTL.bit.CLKDIV = TB_DIV1;
//EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;           // Period autoreload
EPwm8Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;       // Load registers every ZERO
EPwm8Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO_PRD;
EPwm8Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;        // Load registers every ZERO
EPwm8Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO_PRD;
// Set actions
EPwm8Regs.AQCTLA.bit.CAU = AQ_CLEAR;
EPwm8Regs.AQCTLA.bit.CAD = AQ_SET;
EPwm8Regs.AQCTLB.bit.CAU = AQ_SET;
EPwm8Regs.AQCTLB.bit.CAD = AQ_CLEAR;
// Dead time config
EPwm8Regs.DBCTL.bit.OUT_MODE = DB_FULL_ENABLE;    // Enable Dead-band module
EPwm8Regs.DBCTL.bit.POLSEL = DB_ACTV_HIC;          // Active High complementary
EPwm8Regs.DBRED.all = 10;                           // Set 100 clock ticks of
rising edge dead time ( x us)
EPwm8Regs.DBFED.all = 10;                           // Set 100 clock ticks of
falling edge dead time ( x us)
// Stop PWM
EPwm8Regs.DBCTL.bit.POLSEL = DB_ACTV_HI;
EPwm8Regs.AQCSFRC.bit.CSFA = 1;                    // Forces a continuous low on
output A
EPwm8Regs.AQCSFRC.bit.CSFB = 1;                    // Forces a continuous low on
output B

```

```

CpuSysRegs.PCLKCR0.bit.TBCLKSYNC = 1;
EDIS;

}

//*****
***** *****
//*****
***** *****
***** *****
void Setup_DAC(void)
{
    EALLOW;
    CpuSysRegs.PCLKCR16.bit.DAC_A = 1;
    CpuSysRegs.PCLKCR16.bit.DAC_B = 1;

    // ***** DAC-A ***** //
    DacaRegs.DACCTL.bit.SYNCSEL = 0x06; // EPWM7SYNCPER signal will update
the DACVALA register
    DacaRegs.DACCTL.bit.LOADMODE = 0x01; // Load on next EPWMSYNCPER
specified by SYNCSEL
    DacaRegs.DACCTL.bit.DACREFSEL = 0x01; // ADC VREFHI/VSSA are the
reference voltages
    DacaRegs.DACVALS.bit.DACVALS = 0;
    DacaRegs.DACOUTEN.bit.DACOUTEN = 1; // DAC output is enabled
    DacaRegs.DACLOCK.all = 0x00;

    // ***** DAC-B ***** //
    DacbRegs.DACCTL.bit.SYNCSEL = 0x06; // EPWM7SYNCPER signal will update
the DACVALA register
    DacbRegs.DACCTL.bit.LOADMODE = 0x01; // Load on next EPWMSYNCPER
specified by SYNCSEL
    DacbRegs.DACCTL.bit.DACREFSEL = 0x01; // ADC VREFHI/VSSA are the
reference voltages
    DacbRegs.DACVALS.bit.DACVALS = 0;
    DacbRegs.DACOUTEN.bit.DACOUTEN = 1; // DAC output is enabled
    DacbRegs.DACLOCK.all = 0x00;

    EDIS;
}

```

Apêndice M

Código implementado para os *Slaves* (*Function_Definition.h*)

```
/*
 * Function_Definition.h
 *
 * Created on: 26/05/2022
 * Author: Ricardo Coelho
 */

#ifndef FUNCTION_DEFINITION_H_
#define FUNCTION_DEFINITION_H_
#include "F28x_Project.h"

void slave_counter(Uint16 array[], Uint16 last_index);
Uint16 slave_config(Uint16 array[], Uint16 last_index, Uint16 total_slaves);
void slave_dc_bus(Uint16 array[], Uint16 last_index, Uint16 total_slaves);
Uint16 receive_pwm_data(Uint32 data);
void setup_pwm_phase(Uint16 phase_delay, Uint16 total_slaves);
void update_pwm(Uint16 counter_compare);
void start_pwm(void);
void stop_pwm(void);
Uint16 error_check(void);
void stop(void);

#endif /* FUNCTION_DEFINITION_H_ */
```

Apêndice N

Código implementado para os *Slaves*
(Function_Definition.c)

```

/*
 * Function_Definition.c
 *
 * Created on: 26/05/2022
 * Author: Ricardo Coelho
 */

//*****
// Includes
//*****
#include "Function_Definition.h"
#include "math.h"

//*****
// Defines
//*****
#define DC_Bus 60

//*****
// *****
*****void slave_counter(Uint16 array[], Uint16 last_index){
    int16 index = 0;

    if(last_index == 2){
        array[2] = 1;
        array[3] = 0x0024;
    }
    else{
        array[last_index] = array[last_index-1] + 1;
        array[last_index+1] = 0x0024;
    }

    while (index < last_index+2){
        SpiaRegs.SPITXBUF = array[index];
        index++;
        //DELAY_US(1000);
    }
}

//*****

```

```

//*****
***** 
UInt16 slave_config(UInt16 array[], UInt16 last_index, UInt16 total_slaves){
    int16 index = 0;
    UInt16 phase_delay;

    phase_delay = array[last_index];

    while (index < total_slaves + 3){
        SpiaRegs.SPITXBUF = array[index];
        index++;
    }

    return phase_delay;
}

//*****
***** 
//*****
***** 
void slave_dc_bus(UInt16 array[], UInt16 last_index, UInt16 total_slaves){
    int16 index = 0;

    array[last_index] = DC_Bus;

    while (index < total_slaves + 3){
        SpiaRegs.SPITXBUF = array[index];
        index++;
        //DELAY_US(1000);
    }
}

//*****
***** 
//*****
***** 
UInt16 receive_pwm_data(UInt32 data){
    UInt32 first_aux, second_aux, third_aux;

    data = GpioDataRegs.GPADAT.all;

    first_aux = ((data>>4) & 0x000000FF);
    second_aux = ((data>>16) & 0x00000F00);
    third_aux = first_aux | second_aux;

    third_aux = third_aux<<1;

    return ((UInt16)third_aux);
}

//*****
***** 
//*****
***** 
void setup_pwm_phase(UInt16 phase_delay, UInt16 total_slaves){
    static float32 phase_delay1, phase_delay2;
    UInt32 aux;

    phase_delay1 = phase_delay;
}

```

```

phase_delay2 = phase_delay + (180.0 / total_slaves);

if( phase_delay1 <= 180 ){
    EPwm1Regs.TBCTL.bit.PHSDIR = 0;
    EPwm8Regs.TBCTL.bit.PHSDIR = 0;
    aux = (phase_delay1 * 500 / 18);
    EPwm1Regs.TBPHS.bit.TBPHS = (UInt16)aux;
    EPwm8Regs.TBPHS.bit.TBPHS = (UInt16)aux;
}
else{
    EPwm1Regs.TBCTL.bit.PHSDIR = 1;
    EPwm8Regs.TBCTL.bit.PHSDIR = 1;
    aux = ((360 - phase_delay1) * 500 / 18);
    EPwm1Regs.TBPHS.bit.TBPHS = (UInt16)aux;
    EPwm8Regs.TBPHS.bit.TBPHS = (UInt16)aux;
}

if(phase_delay2 <= 180){
    EPwm2Regs.TBCTL.bit.PHSDIR = 0;
    aux = (phase_delay2 * 500 / 18);
    EPwm2Regs.TBPHS.bit.TBPHS = (UInt16)aux;
    //EPwm2Regs.TBPHS.bit.TBPHS = 5000;
}
else{
    EPwm2Regs.TBCTL.bit.PHSDIR = 1;
    aux = ((360 - phase_delay2) * 500 / 18);
    EPwm2Regs.TBPHS.bit.TBPHS = (UInt16)aux;
}

}

//*****
*****  

//*****
*****  

*****  

void update_pwm(Uint16 counter_compare){
    EPwm1Regs.CMPA.bit.CMPA = counter_compare;
    EPwm8Regs.CMPA.bit.CMPA = counter_compare;
    EPwm2Regs.CMPA.bit.CMPA = counter_compare;

}

//*****
*****  

//*****
*****  

*****  

void start_pwm(void){
    EPwm1Regs.DBCTL.bit.POLSEL = DB_ACTV_HIC;
    EPwm1Regs.AQCSFRC.bit.CSFA = 0;           // Forces a continuous low on
output A
    EPwm1Regs.AQCSFRC.bit.CSFb = 0;           // Forces a continuous low on
output B
    EPwm8Regs.DBCTL.bit.POLSEL = DB_ACTV_HIC;
    EPwm8Regs.AQCSFRC.bit.CSFA = 0;           // Forces a continuous low on
output A
    EPwm8Regs.AQCSFRC.bit.CSFb = 0;           // Forces a continuous low on
output B
}

```

```

EPwm2Regs.DBCTL.bit.POLSEL = DB_ACTV_HIC;
EPwm2Regs.AQCSFRC.bit.CSFA = 0;                                // Forces a continuous low on
output A
EPwm2Regs.AQCSFRC.bit.CSFB = 0;                                // Forces a continuous low on
output B
}

//*****
***** *****
//*****
***** *****
void stop_pwm(void){
    EPwm1Regs.DBCTL.bit.POLSEL = DB_ACTV_HI;
    EPwm1Regs.AQCSFRC.bit.CSFA = 1;                                // Forces a continuous low on
output A
    EPwm1Regs.AQCSFRC.bit.CSFB = 1;                                // Forces a continuous low on
output B
    EPwm8Regs.DBCTL.bit.POLSEL = DB_ACTV_HI;
    EPwm8Regs.AQCSFRC.bit.CSFA = 1;                                // Forces a continuous low on
output A
    EPwm8Regs.AQCSFRC.bit.CSFB = 1;                                // Forces a continuous low on
output B

    EPwm2Regs.DBCTL.bit.POLSEL = DB_ACTV_HI;
    EPwm2Regs.AQCSFRC.bit.CSFA = 1;                                // Forces a continuous low on
output A
    EPwm2Regs.AQCSFRC.bit.CSFB = 1;                                // Forces a continuous low on
output B
}

//*****
***** *****
//*****
***** *****
Uint16 error_check(void){
    Uint16 error;

    error = GpioDataRegs.GPDDAT.bit.GPIO104;

    return error;
}

//*****
***** *****
//*****
***** *****
void stop(void){
}

```