

**UNIVERSIDADE FEDERAL DE SANTA CATARINA  
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA  
ELÉTRICA**

Paulo Ricardo Cechelero Villa

**PLATAFORMA DE DESENVOLVIMENTO BASEADA EM  
TECNOLOGIA RECONFIGURÁVEL PARA CONVERSORES  
DE ENERGIA INTELIGENTES**

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina para a obtenção do Grau de Mestre em Engenharia Elétrica.

Orientador:

Prof. Dr. Eduardo Augusto Bezerra

Coorientador:

Prof. Dr. Samir Ahmad Mussa

Florianópolis  
2013



Ficha de identificação da obra elaborada pelo autor,  
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Villa, Paulo Ricardo Cechelero  
Plataforma de Desenvolvimento Baseada em Tecnologia  
Reconfigurável para Conversores de Energia Inteligentes  
[dissertação] / Paulo Ricardo Cechelero Villa ; orientador,  
Eduardo Augusto Bezerra ; co-orientador, Samir Ahmad  
Mussa. - Florianópolis, SC, 2013.  
127 p. ; 21cm

Dissertação (mestrado) - Universidade Federal de Santa  
Catarina, Centro Tecnológico. Programa de Pós-Graduação em  
Engenharia Elétrica.

Inclui referências

1. Engenharia Elétrica. 2. FPGA. 3. Plataforma de  
Eletrônica de Potência. 4. Reconfiguração Dinâmica. 5. Redes  
Inteligentes. I. Bezerra, Eduardo Augusto. II. Mussa,  
Samir Ahmad. III. Universidade Federal de Santa Catarina.  
Programa de Pós-Graduação em Engenharia Elétrica. IV. Título.



Paulo Ricardo Cechelero Villa

**PLATAFORMA DE DESENVOLVIMENTO BASEADA EM  
TECNOLOGIA RECONFIGURÁVEL PARA CONVERSORES  
DE ENERGIA INTELIGENTES**

Esta Dissertação foi julgada adequada para obtenção do Título de Mestre em Engenharia Elétrica, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina.

Florianópolis, 25 de fevereiro de 2013.

---

Prof. Patrick Kuo Peng, Dr.  
Coordenador do Curso

**Banca Examinadora:**

---

Prof. Eduardo Augusto Bezerra, Dr.  
Presidente (Orientador)  
Universidade Federal de Santa Catarina

---

Prof. Eduardo Todt, Dr.  
Universidade Federal do Paraná

---

Prof. Djones Vinicius Lettnin, Dr.  
Universidade Federal de Santa Catarina

---

Prof. Márcio Silveira Ortmann, Dr.  
Universidade Federal de Santa Catarina



*Dedico esse trabalho ao meu Avô  
Oscar Cechelero, por sempre estar  
presente.*





## AGRADECIMENTOS

Antes de tudo, gostaria de agradecer ao Prof. Eduardo Augusto Bezerra pelo apoio e orientação durante meu percurso na UFSC, auxílio sem o qual não existiria tal Dissertação.

Ao Prof. Samir Ahmad Mussa, pela orientação e ajuda no decorrer desse trabalho. Ao Doutorando (em breve Doutor) Tiago Kommers Jappe, pelas diversas horas investidas no meu trabalho.

Aos meus pais, avós, irmãos, tios, a minha namorada e toda minha família que deram apoio durante todo o mestrado e também forças para continuar a seguir esse caminho. Em especial minha Mãe e Avó, que conseguem segurar a saudade devido à distância, sempre me apoiando nos momentos mais difíceis.

Agradeço aos meus amigos e colegas que convivem e fazem parte da minha vida, que me ajudam dia após dia a me tornar uma pessoa melhor.

A UFSC, CNPq, funcionários e integrantes do INEP e LCS por fornecerem os recursos e auxílio financeiro necessário para o desenvolvimento do trabalho.

E por fim, agradeço a Deus por ter me concedido a benção da vida, e a oportunidade de realizar tais feitos.



*“A good scientist is a person with original ideas. A good engineer is a person who makes a design that works with as few original ideas as possible. There are no prima donnas in engineering.”*

Freeman Dyson



## RESUMO

A busca constante da sociedade por um ambiente com maior sustentabilidade é uma tendência mundial. Novas abordagens na maneira e eficiência em que os recursos naturais são utilizados estão, cada vez mais, no foco de pesquisas acadêmicas. Foco que inclui também a energia elétrica, causando uma mudança nos processos de geração, transmissão, distribuição e utilização da energia disponível. A abordagem atual apresenta uma característica unidirecional e suscetível a falhas, sendo este o ponto em que as redes inteligentes de energia (*Smart Grids*) têm o objetivo de preencher parte dos problemas. Ao mesmo tempo, o controle de conversão de energia é migrado do domínio analógico para o discreto. Ambiente em que microprocessadores e DSPs são utilizados com elevada frequência, e – mais recentemente – o uso de circuitos reconfiguráveis (como FPGAs) que demonstram seu valor devido a diversos fatores. Entretanto, a escolha do controlador DSP ainda é predominante devido ao fato de implementações em software não alarmarem os desenvolvedores, além do fato de dispor de uma maior disponibilidade de recursos humanos capacitados. O presente trabalho visa o desenvolvimento de uma plataforma para a utilização de FPGAs no controle de eletrônica de potência, podendo ser empregada em aplicações de redes inteligentes. A plataforma auxilia na construção de um sistema embarcado completo com possibilidade de reprogramação remota, e monitoramento via rede. Além disso, fornece suporte para simulações do hardware, entradas e saídas pré-definidas e de fácil acesso ao hardware de controle desejado, fluxo detalhado e acesso as ferramentas de desenvolvimento necessárias para a tecnologia FPGA. Esta plataforma, com essas especificações, visa auxiliar engenheiros de eletrônica de potência no desenvolvimento de controladores baseados em FPGA, cobrindo a área de sistemas embarcados e tecnologia da informação com uma abordagem simplificada. Visando demonstrar o processo de desenvolvimento proposto, como estudo de caso foi implementado um Retificador Boost. Esse estudo de caso foi utilizado para realizar os testes de aplicabilidade da plataforma.

**Palavras-chave:** FPGA. Plataforma de Eletrônica de Potência. Reconfiguração Dinâmica. Redes Inteligentes.



## ABSTRACT

The search for a more sustainable environment is a global trend. New approaches in the way and efficiency that natural resources are used have been an important target of academic research. This academic effort also includes electric energy – leading to a change in the process of power generation, transmission, distribution and use. The present approach has a unidirectional feature and it is susceptible to faults. This is a major concern which Smart Grids are intended to cover. At the same time, the energy conversion control is migrated from the analogue the discrete domain. The discrete domain is where microprocessor and DSPs are frequently used, and – more recently – reconfigurable circuits (such as FPGAs) also demonstrate their values due to several reasons. However, the choice of DSP controllers remains prevalent due to the fact that software implementations do not frighten the developers, besides there is a greater availability of trained human resources. The present work aims the development of a platform to assist the use of FPGAs in the control of power electronics which can be used in intelligent network applications. The platform helps in the construction of a complete embedded system, with the possibility of remote reprogramming and network monitoring. Additionally, the platform provides hardware simulation facilities, pre-defined and easy access to the desired control hardware input/output, detailed design flow and support to the development tools needed by FPGA technology. This platform, with these specifications, aims to assist power electronic engineers in the development of FPGA based controllers, covering the area of embedded systems and information technology with a simplified approach. In order to present the proposed process, a Boost Rectifier has been implemented as a case study. This case study was used to perform the platform applicability tests.

**Keywords:** FPGA. Power Electronics Platform. Dynamic Reconfiguration. Smart Grids.





## LISTA DE FIGURAS

Figura 1 – Quantidade de artigos na base IEEE Xplore classificados por ano de publicação usando como termo de busca “renewable energy”.....	25
Figura 2 – Tempos de execução entre (a) microcontrolador de uso geral, (b) DSP e (c) FPGA. ....	28
Figura 3 – Arquitetura de um FPGA genérico. ....	32
Figura 4 – Camadas de configuração e aplicação de um FPGA. ....	33
Figura 5 – Exemplos de aplicações para FPGAs com reconfiguração parcial, (a) redução do tamanho do dispositivo, (b) troca do hardware de codificação do protocolo.....	34
Figura 6 – FPGA com reconfiguração parcial. ....	35
Figura 7 – Fluxo genérico para desenvolvimento de hardware em FPGAs. ....	38
Figura 8 – Fluxo de desenvolvimento para FPGA com reconfiguração parcial. ....	39
Figura 9 – Conversor de potência genérico.....	40
Figura 10 – Definição de <i>Smart Grid</i> de acordo com o departamento de energia dos EUA. ....	42
Figura 11 – Comparação entre o modelo conceitual IEEE de <i>Smart Grid</i> e o sistema atual de energia. ....	43
Figura 12 – Exemplo de <i>MicroGrid</i> .....	44
Figura 13 – Blocos básicos de uma plataforma genérica de controle para eletrônica de potência. ....	45
Figura 14 – Visão geral da plataforma de desenvolvimento. ....	47
Figura 15 – Arquitetura da plataforma proposta. ....	48
Figura 16 – Periférico de controle do conversor. ....	54
Figura 17 – Representação das portas de entrada e saída da área de controle do conversor. ....	55
Figura 18 – Sequência de inicialização do software embarcado. ....	57
Figura 19 – Captura da janela do software de monitoramento em execução. ....	59
Figura 20 – Diagrama do fluxo de desenvolvimento proposto. ....	62
Figura 21 – Visão geral do controle do retificador boost aplicado na plataforma. ....	64
Figura 22 – Retificador boost. ....	65
Figura 23 – (a) Modelo equivalente do retificador boost, (b) interruptor em condução e (c) interruptor aberto. ....	66
Figura 24 – Gráfico da corrente no indutor boost em função do tempo. ....	68
Figura 25 – Circuito da placa de potência.....	70
Figura 26 – Circuito de condicionamento dos sinais de entrada. ....	72
Figura 27 – Circuito de condicionamento dos sinais de saída. ....	73
Figura 28 – Estrutura interna do conversor AD7367. ....	75
Figura 29 – Principais componentes da plataforma XUPV5 (FPGA).....	76
Figura 30 – Principais componentes da placa de instrumentação. ....	76
Figura 31 – Principais componentes da placa de potência. ....	77
Figura 32 – Representação da implementação do controle em malha aberta. ....	78

Figura 33 – Diagrama de blocos da implementação do controle da malha de corrente.....	80
Figura 34 – Diagrama de blocos da implementação do controle com a malha de tensão.....	81
Figura 35 – Simulação funcional de um hardware para a plataforma.....	84
Figura 36 – Sinais de entrada e saída do módulo PWM.....	85
Figura 37 – Sinais de saída do módulo PWM com valor fixo de referência.....	86
Figura 38 – Sinais de saída do módulo PWM com referência de uma rampa.....	87
Figura 39 – Sinais de entrada e saída do módulo AD7367_SPI.....	88
Figura 40 – Monitoramento da requisição de conversão – sinais <i>convst</i> , <i>busy</i> , <i>cs</i> e <i>sclk</i> .....	89
Figura 41 – Monitoramento da comunicação SPI com o conversor AD – sinais <i>cs</i> , <i>sclk</i> e <i>douta</i> .....	89
Figura 42 – Diagrama de blocos do controlador PI.....	90
Figura 43 – Sinais de entrada e saída do módulo PI.....	91
Figura 44 – Captura do efeito de <i>inrush</i> .....	92
Figura 45 – Conversor boost em malha aberta ( $D=0.5$ ).....	93
Figura 46 – Conversor boost em malha aberta ( $D=0.25$ ).....	94
Figura 47 – Formas de onda da tensão de entrada, corrente e tensão de saída sem atuação do conversor boost.....	95
Figura 48 – Conversor boost com malha de controle da corrente.....	95
Figura 49 – Conversor boost com malha de controle da tensão, alteração na tensão de entrada.....	96
Figura 50 – Conversor boost com malha de controle da tensão, elevação do valor de referência da tensão de saída.....	97
Figura 51 – Conversor boost com malha de controle da tensão, redução do valor de referência da tensão de saída.....	98
Figura 52 – Diagrama de blocos da implementação de [57].....	100
Figura 53 – Particionamento da técnica de controle vetorial utilizado em [58].	101
Figura 54 – Diagrama de blocos da implementação de [59].....	103
Figura 55 – Diagrama de blocos da implementação de [61].....	105
Figura 56 – Comparação entre as abordagens utilizadas para a implementação da plataforma.....	112

**LISTA DE TABELAS**

Tabela 1 – Comparação entre a abordagem de geração de energia existente e inteligente. ....	26
Tabela 2 – Modos de operação do módulo PWM. ....	85
Tabela 3 – Comparação entre os trabalhos relacionados e a plataforma proposta. ....	109



## LISTA DE ABREVIATURAS E SIGLAS

A	Ampère
AD	Analógico-Digital
ADC	Analog-Digital Converter
API	Application Programming Interface
ASIC	Application Specific Integrated Circuit
BL	Bloco Lógico
BRAM	Block RAM
BSB	Base System Builder
C.A.	Corrente Alternada
C.C.	Corrente Contínua
CAN	Controller Area Network
CHP	Combined Heat and Power
CI	Circuito Integrado
CLB	Configurable Logic Block
CPLD	Complex Programmable Logic Device
CPU	Central Processing Unit
DDR	Double Data Rate
DOE	Department of Energy
DR	Dynamic Reconfiguration
DSP	Digital Signal Processor
E/S	Entrada/Saída
EDA	Electronic Design Automation
FPGA	Field Programmable Gate Array
FSL	Fast Simplex Link
HDL	Hardware Description Language
Hw	Hardware
Hz	Hertz
ICAP	Internal Configuration Access Port
IEEE	Institute of Electrical and Electronics Engineers
IGBT	Insulated-Gate Bipolar Transistor
IOB	Input/Output Boundary
IP	Internet Protocol
IPC	Inter-Process Communication
JTAG	Joint Test Action Group
kHz	kilo Hz
LED	Light Emitting Diode
LL TEMAC	Local Link TEMAC
MAC	Medium Access Control
MHz	Mega Hz

MMU	Memory Management Unit
MOSFET	Metal-Oxide-Silicon Field Effect Transistor
MPSoC	Multi-Processor SoC
MSPS	Mega Samples Per Second
PAL	Programmable Array Logic
PC	Personal Computer
PCIe	Peripheral Component Interconnect Express
PF	Power Factor
PFC	Power Factor Correction
PI	Proporcional-Integral
PID	Proporcional-Integral-Derivativo
PLB	Processor Local Bus
POSIX	Portable Operating System Interface
PPC	Power PC
PR	Partial Reconfiguration
PWM	Pulse Width Modulation
RAM	Random Access Memory
RISC	Reduced Instruction Set Computing
RMS	Root Mean Square
RP	Reconfigurable Partition
RTL	Register Transfer Level
SDK	Software Development Kit
SDRAM	Single Data RAM
SO	Sistema Operacional
SoC	System-on-Chip
SPI	Serial Peripheral Interface
SRAM	Static RAM
SVPWM	Space Vector PWM
Sw	Software
TCL	Tool Command Language
TEMAC	Tri-mode Ethernet MAC
TFTP	Trivial File Transfer Protocol
THD	Total Harmonic Distortion
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
V	Volt
VHDL	Very high speed integrated circuit HDL
XPS	Xilinx Platform Studio
XST	Xilinx Synthesis Technology
ZOH	Zero Order Hold

## SUMÁRIO

<b>1 INTRODUÇÃO .....</b>	<b>25</b>
1.1 OBJETIVOS .....	29
1.2 ESTRUTURA DO TEXTO .....	30
<b>2 TECNOLOGIAS ENVOLVIDAS.....</b>	<b>31</b>
2.1 CIRCUITOS RECONFIGURÁVEIS .....	31
2.2 RECONFIGURAÇÃO PARCIAL E DINÂMICA .....	34
2.3 FERRAMENTAS DE DESENVOLVIMENTO PARA FPGAS .....	36
2.4 CONVERSORES DE POTÊNCIA.....	40
2.5 SMARTGRIDS E MICROGRIDS .....	41
<b>3 PLATAFORMA DE DESENVOLVIMENTO .....</b>	<b>45</b>
3.1 MICROPROCESSADOR E PERIFÉRICOS .....	47
<b>3.1.1 Microblaze .....</b>	<b>48</b>
<b>3.1.2 MDM.....</b>	<b>49</b>
<b>3.1.3 Hw ICAP .....</b>	<b>49</b>
<b>3.1.4 SysAce.....</b>	<b>49</b>
<b>3.1.5 LL TEMAC .....</b>	<b>50</b>
<b>3.1.6 UART .....</b>	<b>50</b>
<b>3.1.7 MPMC .....</b>	<b>50</b>
<b>3.1.8 Power Electronics Control .....</b>	<b>50</b>
3.2 HARDWARE DE CONTROLE DO CONVERSOR .....	51
3.3 SOFTWARE DE GERÊNCIA EMBARCADO E MONITORAMENTO ..55	
<b>3.3.1 Software Embarcado de Gerência.....</b>	<b>56</b>
<b>3.3.2 Software de Monitoramento .....</b>	<b>58</b>
3.4 FLUXO PROPOSTO PARA DESENVOLVIMENTO .....	59
<b>4 APLICAÇÃO DA PLATAFORMA NO CONTROLE DO RETIFICADOR BOOST.....</b>	<b>63</b>
4.1 TOPOLOGIA E FUNCIONAMENTO DO RETIFICADOR BOOST.....	64
4.2 CONVERSOR ESTÁTICO DE POTÊNCIA .....	68
4.3 CIRCUITO DE CONDICIONAMENTO DOS SINAIS DE ENTRADA ..71	
4.4 CIRCUITO DE CONDICIONAMENTO DOS SINAIS DE SAÍDA .....	72
4.5 CONVERSORES ANALÓGICO-DIGITAL.....	74
4.6 VISÃO GERAL DA MONTAGEM DO SISTEMA .....	75
4.7 CONTROLE DO CONVERSOR BOOST .....	77
<b>4.7.1 Malha Aberta (Elevador de Tensão) .....</b>	<b>78</b>
<b>4.7.2 Malha de Controle da Corrente.....</b>	<b>79</b>
<b>4.7.3 Malha de Controle da Tensão .....</b>	<b>80</b>
4.8 UTILIZAÇÃO DA PLAFORMA.....	81
<b>5 RESULTADOS EXPERIMENTAIS .....</b>	<b>83</b>
5.1 SIMULAÇÃO COMPORTAMENTAL DO PERIFÉRICO DE CONTROLE DO CONVERSOR .....	83
5.2 PWM.....	84
5.3 CONVERSORES AD.....	87
5.4 CONTROLADOR PROPORCIONAL INTEGRAL (PI).....	89

5.5 EFEITO DE INRUSH .....	91
5.6 CONTROLE DO CONVERSOR BOOST .....	92
<b>5.6.1 Malha Aberta (Elevador de Tensão) .....</b>	<b>92</b>
<b>5.6.2 Malha de Controle da Corrente .....</b>	<b>94</b>
<b>5.6.3 Malha de Controle da Tensão.....</b>	<b>95</b>
<b>6 TRABALHOS RELACIONADOS .....</b>	<b>99</b>
6.1 FPGA-BASED REALIZATION OF SELF-OPTIMIZING DRIVE- CONTROLLERS .....	99
6.2 MPSOC DESIGN APPROACH OF FPGA BASED CONTROLLER FOR INDUCTION MOTOR DRIVE .....	100
6.3 DESIGN AND DEVELOPMENT OF A FLEXIBLE MULTI-PURPOSE CONTROLLER HARDWARE SYSTEM FOR POWER ELECTRONICS AND OTHER INDUSTRIAL APPLICATIONS .....	102
6.4 A UNIVERSAL DIGITAL PLATFORM AND SOFTWARE LIBRARY FOR POWER ELECTRONIC SYSTEMS INTEGRATION .....	104
6.5 POSICIONAMENTO E COMPARAÇÃO DOS TRABALHOS RELACIONADOS .....	106
<b>6.5.1 Uso da Plataforma de Maneira Genérica para Controle de Conversores de Potência .....</b>	<b>106</b>
<b>6.5.2 Implementação do Controle do Conversor em Hardware ou Software .....</b>	<b>107</b>
<b>6.5.3 Fluxo de Ferramentas de Desenvolvimento para Uso da Plataforma .....</b>	<b>107</b>
<b>6.5.4 Reconfiguração Parcial .....</b>	<b>108</b>
<b>6.5.5 Tabela Comparativa .....</b>	<b>108</b>
<b>7 CONCLUSÕES .....</b>	<b>111</b>
<b>REFERÊNCIAS .....</b>	<b>115</b>
<b>APÊNDICE A – EXEMPLO DE VHDL PARA O HARDWARE DO USUÁRIO .....</b>	<b>121</b>
<b>APÊNDICE B – TESTBENCH VHDL PARA O HARDWARE DO USUÁRIO .....</b>	<b>123</b>
<b>APÊNDICE C – TUTORIAL DO FLUXO DE DESENVOLVIMENTO DA PLATAFORMA .....</b>	<b>127</b>

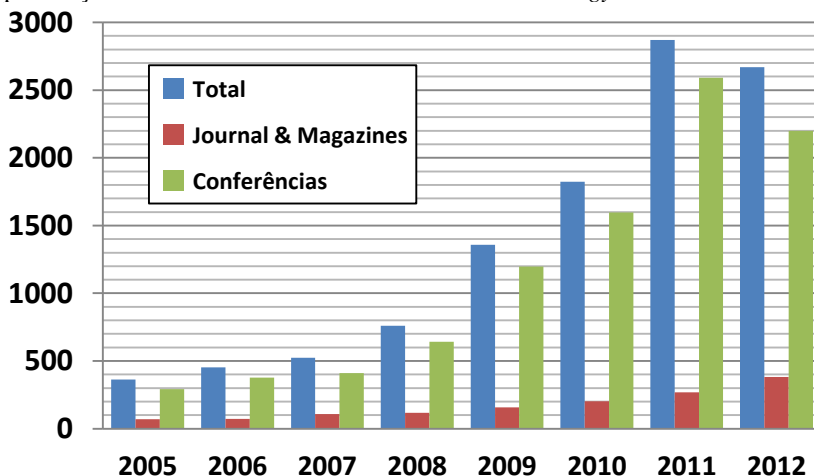


## 1 INTRODUÇÃO

A sociedade busca constantemente aprimoramentos e/ou descobertas científicas que possam melhorar o conforto e a qualidade de vida das pessoas como um todo. Porém, mais recentemente, existe uma crescente preocupação com o desenvolvimento sustentável. De acordo com [1], de um ponto de vista gerencial de recursos renováveis, existem dois itens que definem o desenvolvimento sustentável: 1 – as taxas de coleta dos recursos devem ser iguais às taxas de regeneração (rendimento sustentado), e 2 – as taxas de desperdício emitidas devem ser iguais a capacidade natural do ecossistema (onde esse desperdício é emitido) de assimilar esse desperdício. Na falha de um dos itens, não existe sustentabilidade. Estas novas abordagens fazem com que parte do foco das pesquisas se volte para maneiras mais eficientes de utilização dos recursos que o ambiente dispõe.

Para a energia elétrica este fato não é uma exceção. Pesquisas em fontes de energias renováveis, como geração eólica, solar, correntes marítimas, são alguns dos exemplos. Além de geração, a transmissão, conversão e utilização dessa energia estão inclusos nos fatores de interesse a serem controlados visando melhorias. Como exemplo mais palpável, a Figura 1 apresenta uma síntese dos artigos encontrados na base da IEEE Xplore em Janeiro de 2013 usando o termo de busca “renewable energy”, separado por ano de publicação.

Figura 1 – Quantidade de artigos na base IEEE Xplore classificados por ano de publicação usando como termo de busca “renewable energy”.



Ainda na Figura 1, é possível notar um aumento significativo no volume de artigos publicados nos últimos anos. Também, a quantidade total de artigos com a ocorrência de uma pequena queda no último ano pesquisado. Porém, as contagens de artigos aceitos em revistas e periódicos aumentando, fato que pode ser interpretado como um avanço na maturidade do assunto na academia.

O sistema de transmissão de energia elétrica apresenta uma característica unidirecional, ou seja, o fluxo de potência é da fonte para a carga. No processo de geração, transmissão e distribuição de energia elétrica, há perdas associadas a cada um destes processos. Aproximadamente oito por cento é perdido em linhas de transmissão, vinte por cento é utilizado para suprir a demanda em momentos de pico que representam cinco por cento do tempo [2]. Adicionalmente, a rede sofre de um efeito em cascata no caso de falhas [2]. Nos últimos anos, com o rápido desenvolvimento da economia, capacidade de produção e demanda da população mundial, falhas no fornecimento de energia se tornaram mais frequentes e impactantes.

Nesse contexto o investimento em melhorias da rede se torna mais evidente nas últimas décadas, podendo citar como exemplo o Brasil que – através da resolução normativa Nº 502 [3] – normalizou o sistema de medição de energia com uso de medidores inteligentes. Também, a exigência de uma mudança na abordagem desde a geração de energia até o consumidor final, novas técnicas de proteção, monitoramento e controle as quais irão oferecer uma oportunidade para reduzir a ocorrência de falhas no sistema elétrico de potência [4]. A Tabela 1 apresenta uma comparação entre alguns itens que estão defasados nas redes atuais e as propriedades desejadas.

Tabela 1 – Comparação entre a abordagem de geração de energia existente e inteligente.

<b>Rede Atual de Energia</b>	<b>Rede Inteligente de Energia</b>
Eletromecânica	Digital
Comunicação Unilateral	Comunicação Bilateral
Geração Centralizada	Geração Distribuída
Hierárquica	Rede
Poucos Sensores	Sensores Ubíquos
Restauração Manual	Restauração Autônoma
Verificação e Testes Manuais	Verificação e Testes Remotos
Controle Limitado	Controle Ubíquo

Adaptado de [2].

A próxima geração de arquitetura de rede de energia elétrica, conhecida como *Smart Grid*, tem a expectativa de cobrir os maiores empecilhos das redes atuais, em essência provendo para as companhias de fornecimento uma visibilidade e ubiquidade completa de seus serviços. Outro requisito da rede é ser capaz de recuperar-se automaticamente e ser flexível o bastante em relação a anomalias do sistema.

Seguindo essas tendências, o controle de conversão de energia é migrado da forma analógica para o domínio discreto, que é tradicionalmente implementado com o uso de microprocessadores de uso geral e processadores digitais de sinais (DSP, do inglês *Digital Signal Processor*). O uso de controladores digitais no campo da eletrônica de potência é bastante aceito e diversas implementações já foram alvo de discussão gerando diversos trabalhos [5–10]. Mas o fato do controle ser feito de maneira digital, facilita a possibilidade de adicionar inteligência no controlador para se tornar parte de um *Smart Grid*. Tipicamente microcontroladores e DSPs possuem circuitos dedicados à comunicação de rede (como Ethernet e CAN), memória para armazenamento de informações, entre outras funcionalidades. Esses recursos viabilizam a implementação de sub-rotinas que enviam os valores do conversor em determinado momento, ou recebem um dado para tomada de atitude.

Porém, mais recentemente o uso de circuitos reconfiguráveis, popularmente conhecidos como FPGAs (do inglês, *Field Programmable Gate Array*), tem demonstrado seu valor no controle de eletrônica de potência devido a diversos motivos [11–13]:

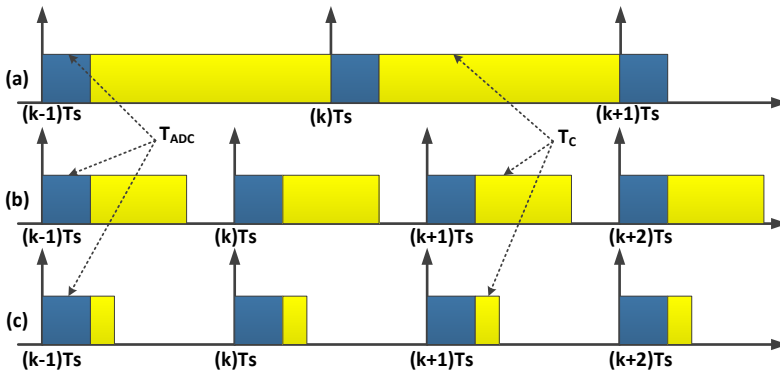
- Melhoria na qualidade e técnica de controle;
- Possibilidade de paralelização de operações aritméticas;
- Controle adaptável em tempo de execução;
- Baixo consumo de potência/redução de emissão térmica;
- Redução do tempo de lançamento do produto no mercado (*time to market*);

A Figura 2 apresenta uma comparação entre os tempos de execução de um algoritmo de controle de corrente para um inversor PWM (do inglês, *Pulse Width Modulation*) em plataformas distintas. Nesta figura,  $T_{ADC}$  representa o tempo de conversão analógico-digital (ADC, do inglês *Analog-to-Digital Converter*),  $T_C$  o tempo de execução do algoritmo de controle e  $T_S$  o período de amostragem, que de costume é fixo levando em conta o período de chaveamento do conversor ou metade disso. Na Figura 2(a) são apresentados os tempos para um

microcontrolador de uso geral, onde o período de amostragem teve que ser adaptado ao tempo de execução do algoritmo, fator limitante nesse caso, causando perdas na qualidade de controle e em alguns casos até a instabilidade da malha de controle fechada [11].

Na Figura 2(b) são mostrados os tempos para uma implementação em um controlador DSP, apresentando melhores resultados e, como consequência, a falta de velocidade de execução não é o fator limitante na malha fechada de controle, levando a uma frequência de amostragem tipicamente entre 10 a 20 kHz para sistemas de potência intermediários. Na Figura 2(c), o diagrama corresponde ao controlador implementado em FPGA. Devido a sua habilidade de transcrever na arquitetura de hardware todo o paralelismo do algoritmo de controle, este leva somente algumas frações do tempo de execução, sobrando tempo assim para implementações mais complexas que possam melhorar a qualidade do controle.

Figura 2 – Tempos de execução entre (a) microcontrolador de uso geral, (b) DSP e (c) FPGA.



Adaptado de [11].

Mesmo havendo diversas vantagens em implementar o controle de conversores de potência em FPGAs, a escolha do controlador DSP ainda é predominante, e conforme [11] isto ocorre devido a fatores históricos. Soluções desenvolvidas em software são mais antigas e não alarmam os desenvolvedores pelo fato de serem baseadas em programação procedural. Além disso, na indústria existe uma disponibilidade maior de recursos humanos capacitados no desenvolvimento de software (linguagem C para DSPs), quando comparado com a oferta de projetistas de hardware (linguagens de

descrição de hardware para FPGAs, fluxo de projeto de hardware, uso de ferramentas específicas, etc) [13,14].

Nesse contexto, as dificuldades para adicionar aos novos conversores de potência com as funcionalidades desejadas das redes inteligentes de energia resultam em um trabalho multidisciplinar. O conhecimento apenas de técnicas de controle causa um empecilho para a integração entre os sistemas. Por exemplo, uma dificuldade encontrada pelo projetista do controle de eletrônica de potência é a comunicação com um componente de ADC. Como desenvolver o protocolo necessário para comunicação com o dispositivo utilizando um de FPGA como plataforma alvo? Ainda, como fazer a comunicação e troca de dados através de uma rede ou barramento de comunicação para possibilitar o acesso remoto ao controle? No caso de alguma melhoria na técnica de controle de um hardware já existente e em funcionamento, como realizar a atualização?

## 1.1 OBJETIVOS

O presente trabalho visa o desenvolvimento de uma plataforma para auxiliar a utilização de FPGAs no controle de eletrônica de potência, podendo ser empregada em aplicações de redes inteligentes. A plataforma auxilia na construção de um sistema embarcado completo com possibilidade de reprogramação remota e monitoramento via rede. Além disso, fornece suporte para simulações do hardware, entradas e saídas pré-definidas e de fácil acesso ao hardware de controle desejado, fluxo detalhado e acesso às ferramentas de desenvolvimento necessárias para a tecnologia FPGA. Esta plataforma, com essas especificações, visa auxiliar engenheiros de eletrônica de potência no desenvolvimento de controladores baseados em FPGA, cobrindo a área de sistemas embarcados e tecnologia da informação com uma abordagem simplificada.

Assim, o presente trabalho tem como **objetivo geral** o desenvolvimento da plataforma para controle de conversores de potência. Os objetivos específicos são:

- Apresentar uma proposta de fluxo para desenvolvimento de hardware de conversores para eletrônica de potência;
- Desenvolver uma ferramenta gráfica para monitoramento do estado do conversor, com fácil acesso do ponto de vista do usuário, de maneira remota;

- Disponibilizar o recurso de envio de novo hardware de controle de maneira remota, através de uma rede de comunicação;
- Realizar uma descrição detalhada do fluxo de projeto para FPGAs pré-definido para engenheiros de eletrônica de potência.

## 1.2 ESTRUTURA DO TEXTO

Os demais capítulos desta dissertação estão divididos da seguinte forma: no Capítulo 2 são apresentados conceitos básicos sobre circuitos reconfiguráveis, reconfiguração parcial e dinâmica, conversores de potência e *Smart Grids*, visando auxiliar o entendimento dos requisitos de projeto para a plataforma desenvolvida. No Capítulo 3 são apresentados os requisitos de uma plataforma genérica de desenvolvimento para eletrônica de potência, e também é descrita a plataforma desenvolvida com detalhes de implementação, ferramentas utilizadas, hardware, circuitos, entre outros; e a maneira proposta de utilização desta plataforma através de um fluxo de desenvolvimento padrão. No Capítulo 4 é apresentado o estudo e aplicação da plataforma, utilizando um conversor de potência conhecido como retificador boost. O Capítulo 5 contém os resultados experimentais obtidos com a implementação da plataforma e descrição dos módulos individuais desenvolvidos, os quais foram utilizados para validar a aplicabilidade da mesma.

Após os conceitos e definições necessárias para o entendimento de plataformas de desenvolvimento para eletrônica de potência serem devidamente elucidados, no Capítulo 6 é apresentada a revisão da literatura, sendo esta necessária para o desenvolvimento desse trabalho. Também é realizada uma comparação com os trabalhos relacionados e o aqui proposto, fornecendo uma visão de inserção dos trabalhos no estado-da-arte. Por fim, as conclusões e considerações finais no Capítulo 7.

## 2 TECNOLOGIAS ENVOLVIDAS

Este capítulo tem como objetivo introduzir as tecnologias utilizadas no decorrer do trabalho. Por se tratar de uma plataforma de desenvolvimento, são apresentados alguns conceitos básicos sobre circuitos reconfiguráveis, reconfiguração dinâmica e fluxo de desenvolvimento para hardware. Também são discutidos conceitos básicos de conversores de potência e as novas tecnologias sobre geração, controle, transmissão e uso de energia elétrica.

### 2.1 CIRCUITOS RECONFIGURÁVEIS

Circuitos reconfiguráveis fazem parte de um tipo especial de circuitos integrados os quais possibilitam que os usuários alterem suas funcionalidades lógicas após a etapa de fabricação do chip. A maneira mais comum de fazer essa mudança é utilizar uma linguagem de descrição de hardware (HDL, do inglês *Hardware Description Language*). Os FPGAs são os principais circuitos dessa categoria, mas também existem componentes programáveis como os CPLDs (do inglês, *Complex Programmable Logic Device*) e PALs (do inglês, *Programmable Array Logic*) [15,16]. Os FPGAs podem ser usados para implementar qualquer lógica digital implementada por um ASIC (do inglês *Application Specific Integrated Circuit*), o que faz estes dispositivos obterem uma aceitação crescente nas últimas décadas. A arquitetura de um FPGA genérico, apresentada na Figura 3, é composta basicamente por três componentes:

- Blocos Lógicos (BL) – Implementam tabelas verdade, e possuem também recursos de armazenamento. São os componentes principais dos FPGAs responsáveis pelo processamento. A quantidade destes elementos em um FPGA representa o tamanho ou quantidade de lógica que o FPGA consegue implementar.
- Blocos de Entrada e Saída (IOB, do inglês *Input/Output Block*) – Fazem a interface do circuito com o mundo externo, podendo ser configurados como entradas, saídas ou bidirecionais.
- Interconexões Programáveis (R) – Responsáveis pelas interconexões entre BLs, entre IOBs, e entre BLs e IOBs, e agem como uma rede de roteadores contando segmentos de fios dentro do FPGA.

Figura 3 – Arquitetura de um FPGA genérico.

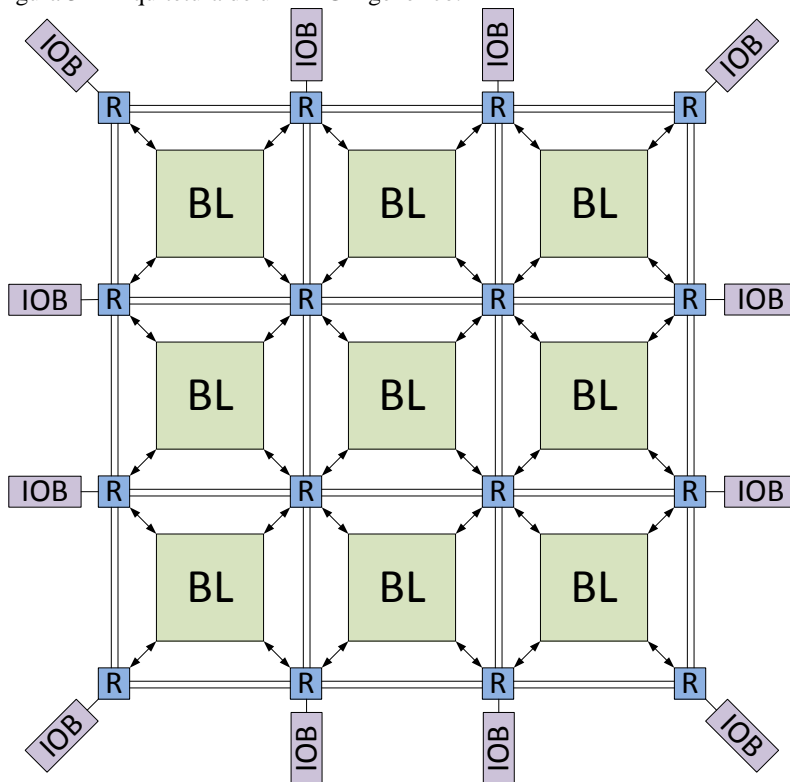


Figura do autor.

Mais recentemente, com a rápida evolução da densidade de portas lógicas em circuitos integrados, é possível notar a adição de blocos com funções específicas que se conectam ao hardware reconfigurável. Como blocos de memória RAM (ou BRAMS), aceleradores de processamento digital de sinais (multiplicadores em hardware capazes de fazer operações aritméticas em um ciclo de relógio), transceptores com padrões industriais (*e.g.* 10-Gb Ethernet) ou configuráveis pelo usuário, processadores integrados no silício [17] (como o FPGA Zynq da empresa Xilinx que embarca um processador ARM Cortex A9 de dois núcleos), entre outros. Melhorando a aceitação e escolha por FPGAs no mercado [13].

Para configurar o FPGA com a lógica desejável, primeiramente é feita a descrição do hardware desejado através de uma HDL ou utilizando um diagrama de portas lógicas (esquemático ou máquina de estados). Após, uma ferramenta de EDA (do inglês *Electronic Design*



*Automation*) faz a tradução e mapeamento para a tecnologia alvo (específica de cada modelo de FPGA e fabricante) gerando um arquivo de configuração (*bitstream*) para o FPGA. O *bitstream* é carregado no FPGA, configurando assim a lógica interna do dispositivo. O processo para gerar o hardware desejado é discutido em mais detalhes na Seção 2.3.

Para implementar a função lógica definida pelo *bitstream* gerado, o FPGA possui uma memória especial que armazena essas informações, podendo ser visto como duas camadas: uma para a memória de configuração, onde o *bitstream* é gravado, e uma camada de aplicação onde estão as tabelas verdades do FPGA. Uma vez que a memória de configuração é gravada, as interconexões do FPGA agem de forma a implementar o hardware definido. A Figura 4 apresenta uma abstração das camadas do FPGA sem estar configurado (estado inicial), à esquerda, e com a configuração desejada pelo usuário, à direita. São duas as maneiras mais tradicionais que esses dispositivos usam para armazenar suas configurações: FPGAs baseados em *antifuse* que uma vez configurados não possibilitam a reconfiguração, e os FPGAs baseados em SRAM (do inglês *Static RAM*) que necessitam de uma nova configuração sempre que alimentado, perdendo a configuração do usuário quando não estiver energizado.

Figura 4 – Camadas de configuração e aplicação de um FPGA.

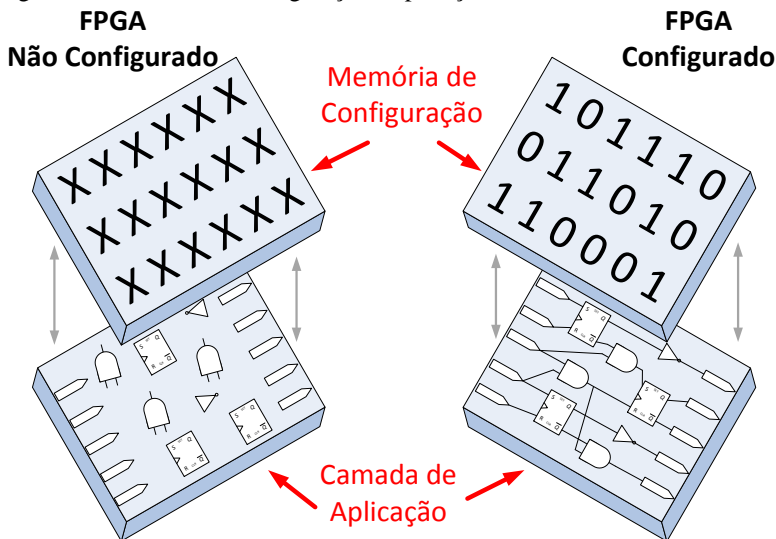


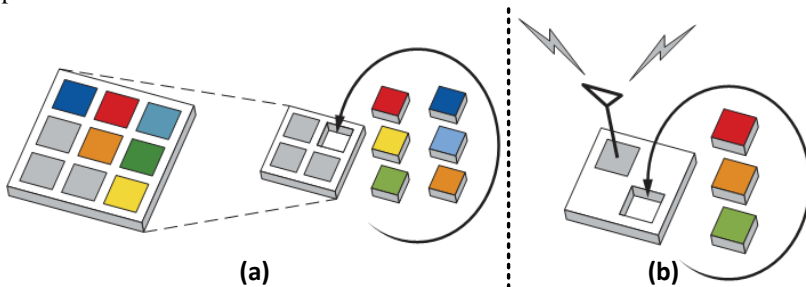
Figura do autor.

## 2.2 RECONFIGURAÇÃO PARCIAL E DINÂMICA

Com a evolução do uso dos FPGAs, a possibilidade de mudar parte do circuito implementado pode ser de utilidade para o usuário. Esta alteração de parte do circuito é conhecida como reconfiguração parcial (PR, do inglês *Partial Reconfiguration*) onde somente uma porção pré-determinada, denominada partição reconfigurável (RP, do inglês *Reconfigurable Partition*), do dispositivo sofre uma alteração na sua lógica e interconexão. Uma vantagem importante relacionada a possibilidade de realizar alterações de porções lógicas do FPGA é [18] a redução de custo e tamanho de placa, sendo possível revezar entre as lógicas necessárias em um determinado momento pelo circuito, o que resulta em menores *bitstreams* e FPGAs com menor capacidade. Uma outra vantagem importante é a redução do consumo de potência, uma vez que menos lógica estática é utilizada.

Na Figura 5 são apresentados exemplos de uso genéricos da reconfiguração parcial em FPGAs. Onde (a) apresenta um caso que possibilita a redução da capacidade do dispositivo através da multiplexação do hardware que não é utilizado, e em (b) a troca do hardware responsável pela codificação do protocolo de acordo com a necessidade.

Figura 5 – Exemplos de aplicações para FPGAs com reconfiguração parcial, (a) redução do tamanho do dispositivo, (b) troca do hardware de codificação do protocolo.



Adaptado de [18].

Um exemplo típico de aplicação que se beneficia da reconfiguração parcial é a comunicação com um PC através da interface *PCI Express* (PCIe). Quando o computador é inicializado, os periféricos PCIe conectados a este devem sinalizar sua presença em um prazo máximo, de acordo com o protocolo, de até 100 milissegundos [19,20]. Como os FPGAs conseguem implementar cada vez mais lógica e o

tempo de configuração total (tempo levado para gravar a memória de configuração) é proporcional a essa capacidade, se torna possível utilizar um FPGA com grande capacidade para essa aplicação iniciando a configuração pelo bloco responsável em realizar a comunicação PCIe. Após feito isso, o restante do FPGA é configurado com a aplicação do usuário [19].

A reconfiguração parcial pode ser realizada através de um agente externo (e.g. um microcontrolador) ou internamente pelo próprio FPGA. Neste caso, para o fabricante Xilinx, é utilizada a porta interna de acesso a configuração (ICAP, do inglês *Internal Configuration Access Port*). Para ser possível realizar a reconfiguração parcial, uma ou mais RPs devem ser definidas no projeto inicial, ou seja, uma (ou mais) parte(s) do circuito será interpretada como uma caixa preta sendo definidas somente as entradas e saídas. Assim toda a parte que não é reconfigurável é conhecida como lógica estática. A partir da lógica estática, as lógicas reconfiguráveis podem ser geradas, também chamadas de módulos reconfiguráveis (RM, do inglês *Reconfigurable Module*).

Na Figura 6 é demonstrado um FPGA que pode ser configurado com o *bistream* completo, o qual contempla a lógica estática e mais uma lógica reconfigurável, ou somente a parte da lógica reconfigurável através de *bistreams* parciais.

Figura 6 – FPGA com reconfiguração parcial.

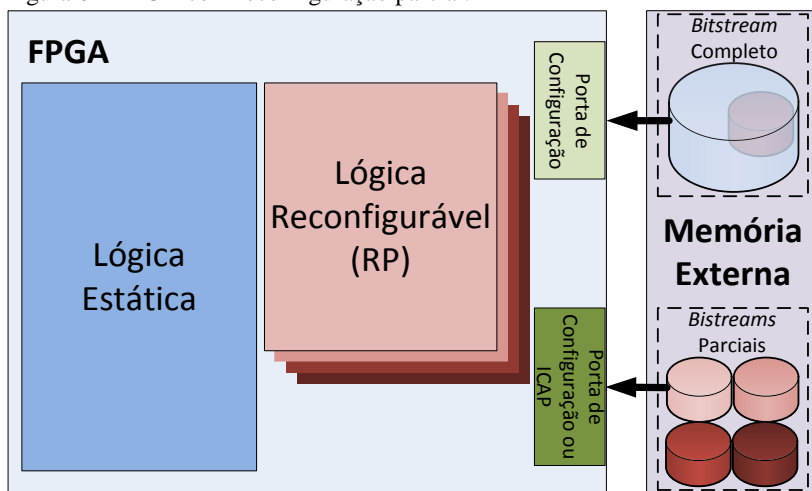


Figura do autor.

Outra flexibilidade disponível é a possibilidade de fazer essa reconfiguração parcial sem que a parte estática (a parte que não pode ser reconfigurada sem fazer uma configuração completa do dispositivo) seja desativada. Esta técnica é conhecida como reconfiguração dinâmica (DR, do inglês *Dynamic Reconfiguration*). Dessa maneira, se desejado, é possível integrar um microcontrolador dentro da lógica do FPGA para gerenciar as possíveis múltiplas configurações parciais, evitando o uso de outro circuito integrado (CI) para essa função. Neste caso o microcontrolador é localizado na área estática do FPGA.

É importante ressaltar que, para que seja possível tirar proveito das técnicas de reconfiguração parcial, a(s) RP(s) deve(m) ser definida(s) ainda na etapa de requisitos de projeto. Um hardware para FPGA que não possui uma RP definida, não se enquadra nas descrições aqui fornecidas de reconfiguração parcial.

### 2.3 FERRAMENTAS DE DESENVOLVIMENTO PARA FPGAS

São quatro as etapas básicas para o desenvolvimento de um hardware para FPGA. O fluxo inclui também a parte de validação para conferir se o hardware que está sendo criado corresponde aos requisitos de projeto. Ambos os fluxos são apresentados na Figura 7, e as etapas de desenvolvimento são descritas como segue:

1. Partindo de uma especificação do projeto é realizada a entrada no fluxo através de uma HDL, *e.g.* Verilog ou VHDL (do inglês *Very High Speed Integrated Circuit Hardware Description Language*), ou um esquemático lógico (normalmente composto de blocos de portas e funções lógicas predefinidas).
2. Após, na fase da síntese lógica, o código é verificado para erros de sintaxe e é realizada uma análise se a hierarquia de componentes está correta. Ainda nessa fase, a descrição é convertida em uma lista de conexões (do inglês, *netlist*) que contém as interconexões entre os componentes no nível de transferência entre registradores (RTL, do inglês. *Register Transfer Level*). Esse processo também é conhecido como sintetização.
3. Na etapa de implementação é realizado o mapeamento dos componentes descritos na *netlist* para os recursos disponíveis no FPGA (ou se não disponível inteiramente, a conversão para um hardware equivalente), que após são

dispostos em seus devidos locais físicos dentro do FPGA juntamente com as devidas conexões para comunicação.

4. O resultado é um *netlist* utilizado para gerar um arquivo de *bitstream* que é empregado na configuração o FPGA alvo.

No fluxo de validação o projeto pode ser analisado de forma comportamental utilizando as entradas do projeto e a saída da síntese lógica. Nesta etapa não são considerados os atrasos das portas lógicas, ou seja, o circuito lógico é considerado ideal. Nas simulações funcional e temporal são considerados os atrasos provenientes da implementação em hardware do circuito. Tradicionalmente, as simulações são feitas utilizando um arquivo *testbench* que aplica estímulos na entrada do circuito observando as saídas para conferência. É possível também fazer a verificação dentro do próprio hardware (*In-Circuit*), incluindo um hardware de monitoramento adicional para extrair tais informações, finalizando o fluxo de validação do sistema.

Figura 7 – Fluxo genérico para desenvolvimento de hardware em FPGAs.

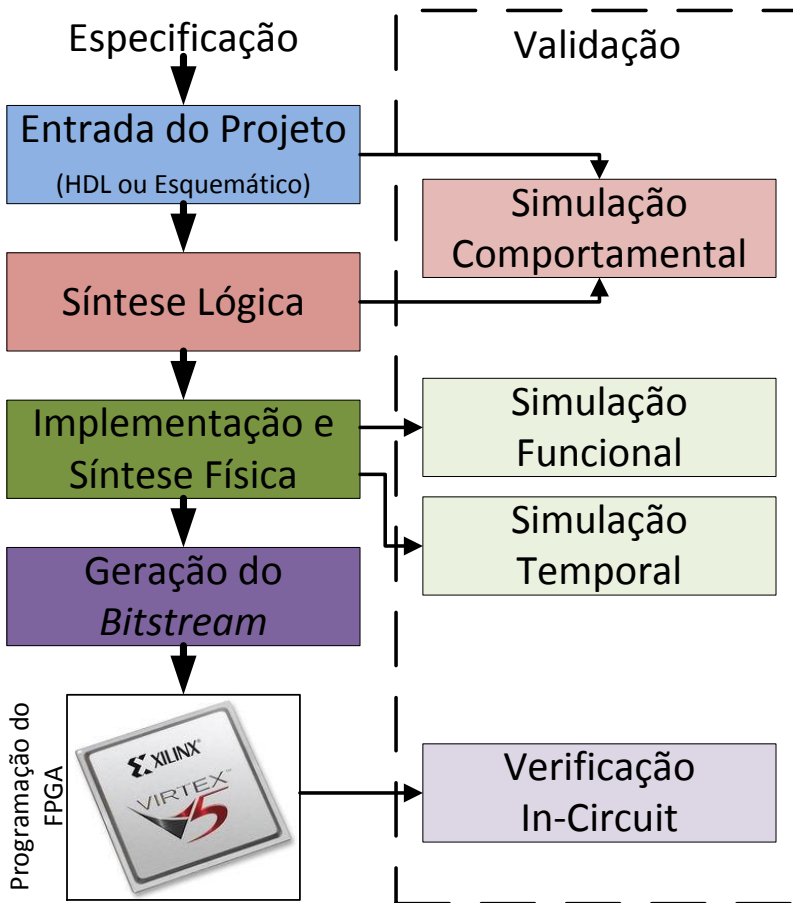


Figura do autor.

Para o desenvolvimento de hardware para FPGAs utilizando a reconfiguração parcial, o fluxo é semelhante, mas devido ao fato das partes reconfiguráveis serem diferentes, estas devem ser sintetizadas separadamente. Porém, a parte estática, comum a todos os RMs, pode ser reaproveitada somente importando a implementação e síntese física. Isso se torna possível, pois todos os RMs são gerados com a mesma interface de entrada e saída. Portanto, na fase de especificação do projeto, deve estar previsto uma área reconfigurável, com sua periferia definida. Caso seja necessário alterar a conexão entre a parte estática e a área reconfigurável, o fluxo completo de projeto deve ser seguido, e o

FPGA reconfigurado com o *bitstream* novo gerado, para ser compatível com os RMs.

Na Figura 8 pode ser verificado o fluxo de desenvolvimento para gerar o hardware desejado, resultando em *bitstreams* completos e parciais. A divisão à esquerda da figura representa a primeira execução do fluxo para a configuração RM\_A, onde ainda não existe a implementação e síntese física da parte estática. E na divisão à direita da figura o fluxo para gerar os *bitstreams* completo e parcial para uma segunda configuração parcial RM\_B. Portanto, fluxos para futuras configurações, como uma RM\_C, utilizarão da mesma implementação e síntese física geradas no primeiro fluxo executado.

Figura 8 – Fluxo de desenvolvimento para FPGA com reconfiguração parcial.

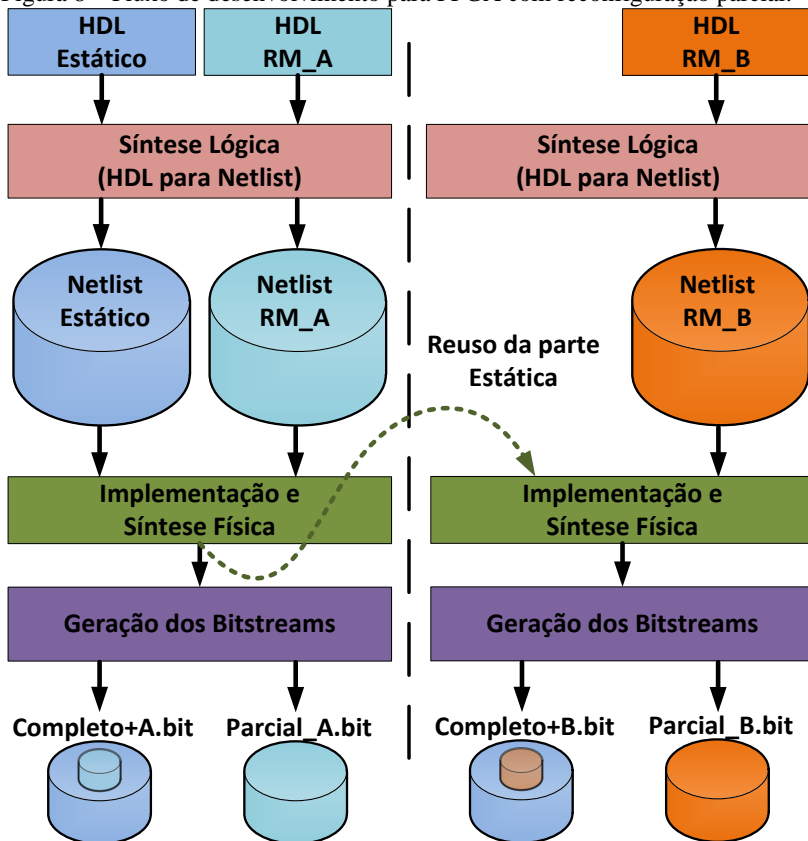
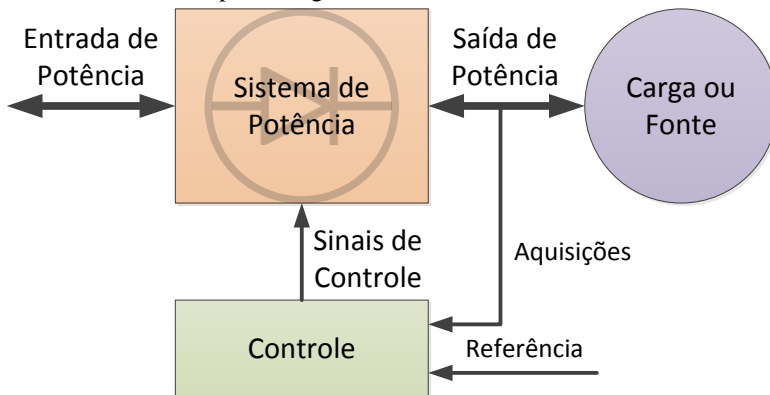


Figura do autor.

## 2.4 CONVERSORES DE POTÊNCIA

Em termos gerais, a funcionalidade de um conversor de potência consiste no processamento e controle do fluxo de energia elétrica de forma otimizada para a carga do usuário [21]. Esta atividade pode ser simples como uma fonte linear composta por um retificador passivo com filtro capacitivo até conversores mais complexos, como filtros ativos trifásicos e conversores matriciais. Na Figura 9 é demonstrado um conversor de potência genérico, onde existe uma entrada de potência que passa por um sistema de potência – ou conversor estático – que atua de acordo com o controle. Utilizando como base valor(es) de referência e medida(s) adquirida(s) do sistema, o controle faz um cálculo a partir de sua lei de controle, que atua na parte de potência através de interruptores estáticos, como por exemplo transistores bipolares, MOSFETs, IGBTs, etc.

Figura 9 – Conversor de potência genérico.



Adaptado de [21].

Tradicionalmente, o controle de conversores de potência é estudado e realizado através de componentes analógicos. Porém, nas últimas décadas esta abordagem vem mudando. O avanço da capacidade de processamento e tecnologia de construção de circuitos integrados em silício tornou possível realizar o controle através de equipamentos discretos, ou seja, o controle realizado sobre o fluxo de energia entre a fonte e a carga não é contínuo no tempo e sim em tempos de amostragem e chaveamento regulares. Um transformador planar, por exemplo, que trabalha na frequência de 250kHz tem menos de um por cento do volume de um transformador tradicional trabalhando na frequência de 50Hz com a mesma classificação de potência [22].



Dentre as vantagens da implementação do controle digital, podemos citar: maior imunidade a ruídos pois há elementos analógicos; flexibilidade da implementação da lei de controle; redução da quantidade de componentes passivos (capacitores, indutores, etc); entre outros [23]. Alguns dos valores e propriedades de sistemas de potência com controle digital são: Fator de potência (PF, do inglês *Power Factor*), eficiência de conversão de energia ( $\eta$ ), tensão de Ripple, e distorção harmônica total (THD, do inglês *Total Harmonic Distortion*) [21].

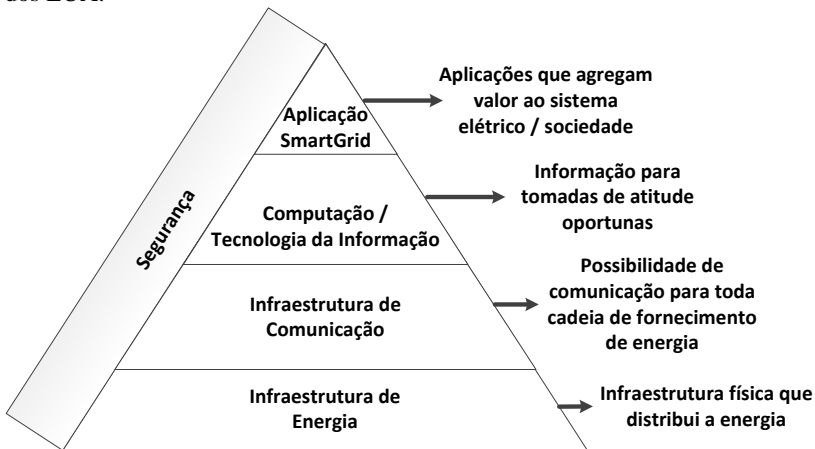
Para realizar o controle de sistemas de conversão de maneira discreta são, tradicionalmente, utilizados processadores de sinais digitais devido, principalmente, à sua capacidade de processamento matemático. Porém, com o aumento da complexidade das topologias dos conversores, o uso de DSPs nesse tipo de aplicação pode ficar limitado a sua natureza de processamento sequencial, outro fator que dificulta o uso desses processadores é a quantidade de pinos de entrada e saída. Nesse contexto os FPGAs se destacam, sendo uma opção mais viável para os novos tipos de controle de conversores de potência.

## 2.5 SMARTGRIDS E MICROGRIDS

As redes inteligentes de energia (do inglês, *Smart Grid*) são um conceito recente em termos de transmissão e utilização de energia elétrica. Dentre as definições [24–27] pode-se afirmar que seria a aplicação extremamente integrada da tecnologia e processamento da informação digital no sistema elétrico de potência tendo como objetivo principal melhorias no aproveitamento e disponibilidade da rede [24]. Este processo envolve a medição e atuação de parâmetros da rede, como corrente e tensão, para monitoramento e tomada de atitude, autônoma ou não [2,25]. O termo carece de uma definição universal, mas diversas agências e forças tarefas existem com esse objetivo [26,27].

De acordo com o Departamento de Energia dos Estados Unidos da América (DOE, do inglês *Department of Energy*), uma definição em camadas de um *Smart Grid* pode ser visto na Figura 10, onde a base é formada pela infraestrutura física que distribui a energia. A camada de comunicação é definida no topo da infraestrutura física de energia para toda a cadeia de fornecimento. A tecnologia de informação vem logo acima da comunicação para tomada de decisão, e aplicações para o *Smart Grid* estão no topo para agregar valor ao sistema elétrico e sociedade. A segurança está em uma dimensão diferente cobrindo todas as camadas.

Figura 10 – Definição de *Smart Grid* de acordo com o departamento de energia dos EUA.

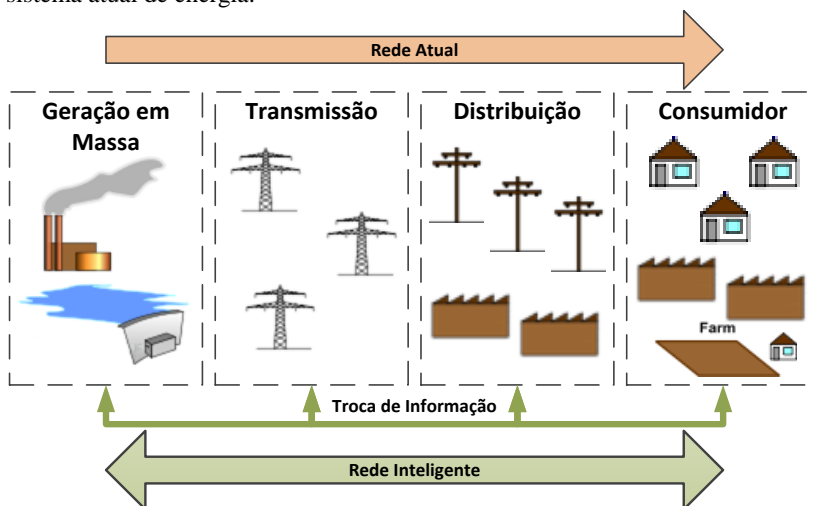


Adaptado de [28].

Uma comparação entre o modelo conceitual da IEEE [29] e a rede atual de energia elétrica é apresentada na Figura 11. A rede de energia atual – representada pela flecha superior na figura – possui somente um sentido, ou seja, saindo da geração da energia, passando pela transmissão e distribuição, chegando as indústrias e consumidores finais. A rede inteligente, por outro lado, oferece a possibilidade da rede ser bidirecional, e todos os pontos estão interconectados, através de uma infraestrutura de comunicação (distribuída ou centralizada), permitindo a troca de informação para tomada de decisão na rede.

Com a diversificação dos tipos disponíveis de fontes de energias – como geradores eólicos, geradores combinados de calor e energia (CHP, do inglês *Combined Heat and Power*), painéis solares, entre outros – a necessidade de um gerenciamento mais eficiente e a independência parcial ou total das redes de distribuição de energia principais se tornam uma necessidade. Como exemplo, uma região rural com geradores eólicos locais. Esta região pode consumir a energia desses geradores, sendo necessário isolar, ou ilhar, essa região da rede principal. Enquanto for possível suprir essa necessidade energética local através dos geradores eólicos, o usuário final (consumidor) não é afetado. Formando assim o conceito de *microgrids* – redes inteligentes de menor capacidade, porém parcialmente autossustentáveis [24].

Figura 11 – Comparação entre o modelo conceitual IEEE de *Smart Grid* e o sistema atual de energia.



Adaptado de [29].

De acordo com o autor [30]: “Microredes ou *microgrids* são sistemas de distribuição, tipicamente em baixa tensão, com alta inserção de fontes de energia distribuídas e dispositivos que fazem o armazenamento de energia, e que operam com uma conexão à rede convencional.”. Na Figura 12 é apresentada um exemplo de aplicação onde um *Microgrid* é utilizado para oferecer a opção de isolar as cargas locais da rede principal. Nesta situação, caso ocorra falta de energia na rede principal e as demais fontes locais possuírem a energia necessária para manter o subsistema, o consumidor final não sofrerá essa perda.

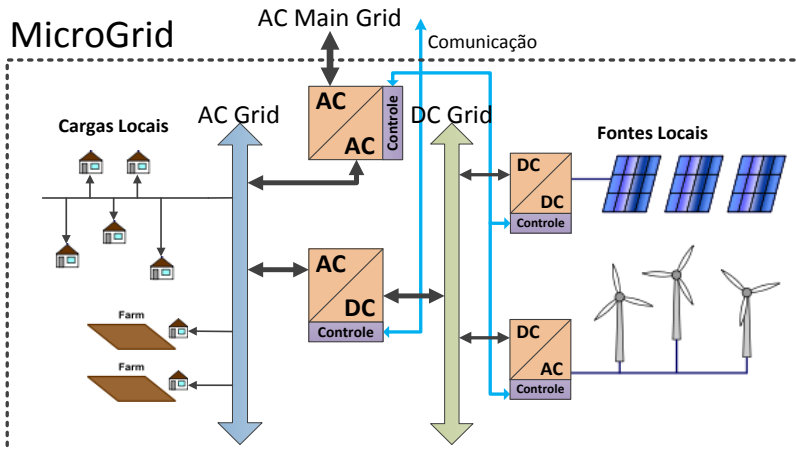
Figura 12 – Exemplo de *MicroGrid*.

Figura do autor.

Isso se torna possível pelo fato dos conversores de energia estarem interconectados por uma rede de comunicação. E trocando informações é possível detectar uma falha, agindo nos barramentos de forma a redirecionar a energia para onde esta é necessária. Também é importante destacar que a topologia da rede de energia está diretamente relacionada com os recursos disponíveis no local de implantação da micro rede.

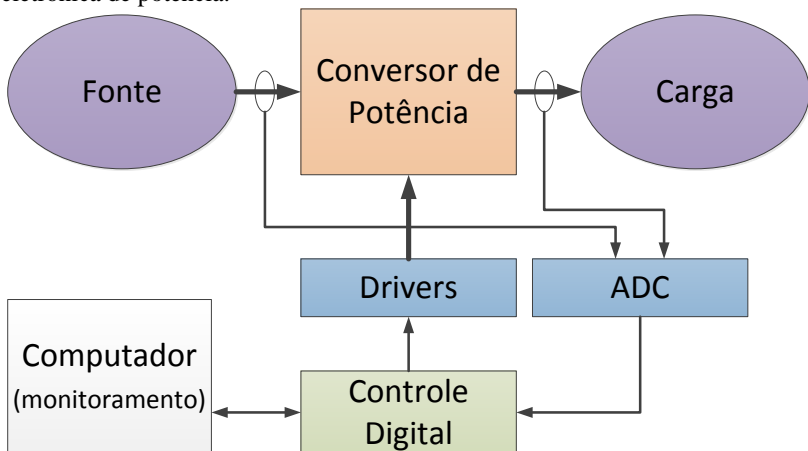
Porém, o diferencial necessário para implantação de redes inteligentes se encontra no controle do conversor. Para atingir os pré-requisitos que estão acima da camada de infraestrutura de energia (Figura 10), o controle do conversor deve ser capaz de ter uma interface para rede de comunicação, bem como a disponibilidade de informações sobre seu estado de funcionamento para suprir as aplicações que agregam valor ao sistema.

Nesse contexto o uso de controladores digitais é essencial, sendo possível utilizar um processador de uso geral, DSP, FPGA, ou até mesmo uma combinação destes, conforme a complexidade do sistema. Além disso, todos os conversores da rede devem ser capazes de se comunicar entre si ou com um sistema centralizado, para tomada de decisão.

### 3 PLATAFORMA DE DESENVOLVIMENTO

Pela perspectiva de aplicação, uma plataforma genérica de um conversor de potência com controle digital é composta por uma fonte, uma carga, o conversor estático de potência e o controlador digital com os módulos correspondentes de suas interfaces (como *drivers* de potência e conversores analógico-digital). Existe também uma conexão opcional para uma interface com um computador, se desejado realizar um monitoramento ou alteração de algum parâmetro no conversor. A Figura 13 representa os blocos básicos encontrados em um conversor de potência genérico. Como exemplo de uma aplicação típica, a fonte é a rede principal de energia, o conversor é um inversor PWM e a carga é um motor de indução. O controle digital desse exemplo é implementado usando um DSP e/ou FPGA [12].

Figura 13 – Blocos básicos de uma plataforma genérica de controle para eletrônica de potência.



Adaptado de [12].

As principais limitações em termos de dinâmica do sistema (largura de banda) são encontradas, de um ponto de vista [12], no laço digital, *i.e.* tempo de conversão do AD, atuação do PWM digital e tempo de processamento (controle). Mas por outro lado, a limitação pode se dar devido ao tempo de chaveamento imposto pelas perdas de chaveamento.

Tendo os componentes básicos requeridos, apresentados na Figura 13, foi possível gerar a especificação básica para uma plataforma para validar a sua aplicabilidade. É importante ressaltar também que, como citado anteriormente, o uso de DSP para controle de eletrônica de

potência é bastante aceito na academia e, portanto, não estudado neste trabalho, sendo o foco o uso de FPGA.

A plataforma proposta nesse trabalho está dividida em três grandes partes: o microprocessador juntamente com seus periféricos, o hardware de controle do conversor de potência e o software de gerência. O conjunto de desenvolvimento de FPGA utilizado é a XUPV5-LX110T [31] fabricado pela empresa Digilent, com as seguintes características:

- FPGA Xilinx Virtex-5 modelo XC5VLX110T;
- Controlador System ACET<sup>TM</sup> para acesso de cartão de memória CompactFlash;
- Memória DDR2 com largura de 64 *bits*;
- Interface PHY Ethernet compatível com *tri-mode* 10/100/1000Mbps;
- USB *Host* e periférico;
- Gerador de relógio programável;
- Interface Serial RS-232, Display de caracteres 16x2;
- Interface de barra de pinos para entradas e saídas;
- Botões, chaves e LEDs.

Pelo fato desta placa ser uma plataforma genérica para desenvolvimento de aplicações de ensino e pesquisa, ela contém diversas outras interfaces que não foram utilizadas neste projeto. No entanto a escolha deste equipamento se deu devido ao domínio das ferramentas e características já utilizadas no decorrer do mestrado. Outras placas de desenvolvimento podem ser utilizadas, somente tomando o cuidado de conter as interfaces e componentes necessários para este projeto.

No desenvolvimento desta plataforma foi utilizado o fluxo padrão fornecido junto do Microblaze, onde é definida uma base do sistema com interfaces, memória e demais configurações para o FPGA. Um periférico personalizado para esse sistema base foi utilizado no modo escravo para implementar o hardware de controle do conversor. Também foi desenvolvido um software embarcado responsável por gerenciar e monitorar o sistema em funcionamento. Para tornar possível o desenvolvimento de diferentes controladores de conversores em hardware no FPGA, o fluxo de reconfiguração parcial da Xilinx foi adotado. As três divisões da plataforma são explicadas, em detalhes, nas seções a seguir. O diagrama de blocos com uma visão geral da plataforma desenvolvida é apresentado na Figura 14.

Figura 14 – Visão geral da plataforma de desenvolvimento.

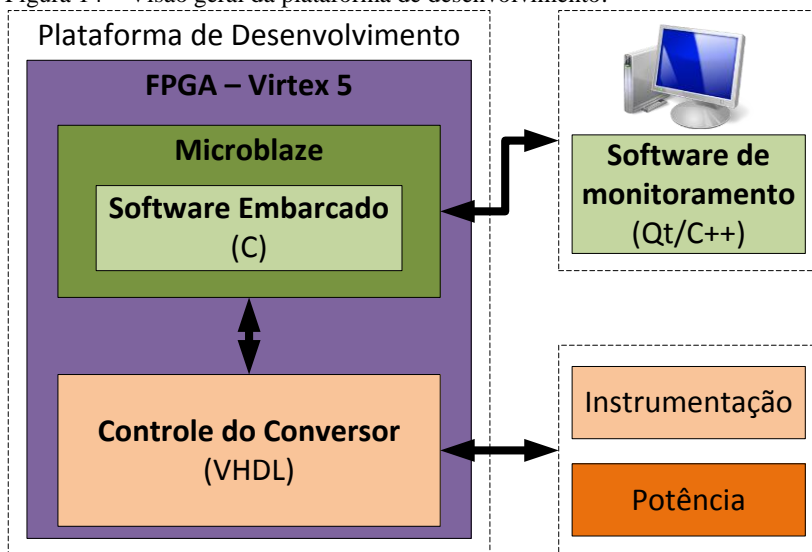


Figura do autor.

### 3.1 MICROPROCESSADOR E PERIFÉRICOS

A ferramenta *Xilinx Platform Studio* (XPS) [32] foi utilizada para configurar o sistema base. Este aplicativo permite configurar graficamente um sistema com um microprocessador e seus periféricos, bem como a geração de modelos para periféricos personalizáveis pelo usuário. No caso da implementação da presente plataforma foi utilizado o processador Microblaze. A Figura 15 demonstra a conexão do Microblaze através de seu barramento local com os demais periféricos utilizados no sistema. Em destaque está o periférico localizado na parte inferior da figura, o qual contém a lógica de controle de potência. Nas seções que seguem, os periféricos utilizados no sistema são detalhados.

Figura 15 – Arquitetura da plataforma proposta.

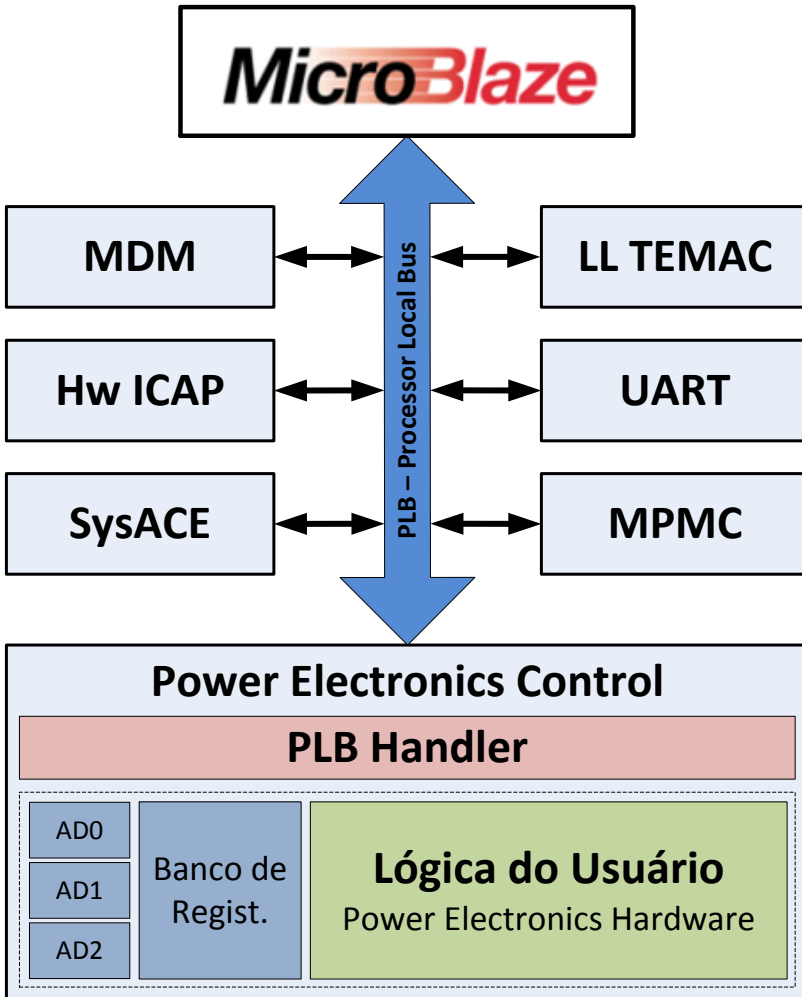


Figura do autor.

### 3.1.1 Microblaze

O Microblaze [33] é um *softprocessor*, *i.e.* uma descrição de hardware sintetizável, de 32 bits RISC com arquitetura Harvard. O processador possui as seguintes características:



- Memória local para programa de execução e armazenamento de dados;
- Unidade de gerenciamento de memória (do inglês *memory management unit* – MMU);
- Memória cache de dados e de instruções otimizada para aplicações embarcadas;
- Controlador de interrupção;
- Suporte ao desenvolvimento de aplicação embarcada (*firmware*) ou utilização de um sistema operacional, e.g. FreeRTOS.

Este processador utiliza a interface de comunicação local denominada *Processor Local Bus* (PLB) [34]. Esta interface é usada para realizar a comunicação com os periféricos utilizados pela Microblaze no modo mestre/escravo, onde os periféricos são acessados através de mapeamento em memória.

### 3.1.2 MDM

Acrônimo para *Microblaze Debug Module* [35], este periférico é utilizado para a comunicação em modo de *debug* com o computador de desenvolvimento, através do ambiente de desenvolvimento integrado. Para realizar tais funcionalidades, o periférico se comunica com o mundo externo através da conexão JTAG, e também disponibiliza uma porta de comunicação UART virtual, visando facilitar a comunicação com o processador.

### 3.1.3 Hw ICAP

O Hw ICAP [36] é uma propriedade intelectual que realiza a comunicação com a porta de acesso a configuração interna (do inglês *Internal Configuration Access Port* – ICAP) do FPGA em tempo de execução. Utilizando esse periférico conectado ao barramento local da Microblaze, é possível executar um código no processador o qual faz o acesso para realizar a modificação da memória de configuração do FPGA, caracterizando uma reconfiguração parcial dinâmica.

### 3.1.4 SysAce

O periférico SysAce [37] faz a comunicação entre o processador e o System ACE (*Advanced Configuration Environment*). O System

ACE é utilizado para realizar leitura e escrita em um dispositivo de Compact Flash. Este periférico é utilizado para armazenar arquivos de maneira local para a placa de desenvolvimento. Também pode ser configurado externamente para configurar o FPGA com um *bistream* no momento em que o sistema é alimentado. Para a plataforma ele é utilizado como dispositivo de armazenamento dos arquivos de configuração.

### 3.1.5 LL TEMAC

O *LocalLink Tri-Mode Ethernet MAC* [38] é um componente que pode estar localizado fisicamente no FPGA ou ser uma descrição de hardware sintetizável, distribuído no formato de um módulo de propriedade intelectual. Tem a função de realizar a comunicação entre a interface física (PHY) e o processador, para troca de pacotes em redes locais Ethernet.

### 3.1.6 UART

Acrônimo do inglês *Universal Asynchronous Receiver Transmitter*, componente utilizando pela Microblaze denominada UART lite [39], o qual faz a comunicação entre o processador e uma interface serial assíncrona para transferência de dados. Pode ser usado como uma interface de modo terminal (interface texto) para o programa executado pelo processador, bem como para comunicação com outros dispositivos que possuam a mesma interface.

### 3.1.7 MPMC

O *Multi-Port Memory Controller* [40] faz a interface entre o processador e memórias de acesso DDR SDRAM disponíveis na placa de desenvolvimento. É utilizado pelo processador como memória cache de instruções e dados, e também pelo LL TEMAC para realizar o armazenamento de pacotes de entrada e saída da rede.

### 3.1.8 Power Electronics Control

Periférico no qual o usuário da plataforma pode desenvolver a lógica de controle para eletrônica de potência. Este periférico disponibiliza componentes para comunicação com conversores analógico-digital externos, o quais podem ser utilizados para medir as

grandezas necessárias para realizar a técnica de controle. Existe também um banco de registradores responsável pela troca de informações dentro do periférico, bem como a monitoria e controle através da interface com a Microblaze.

A comunicação com o barramento local do processador Microblaze é feita através da instância *PLB Handler*, o qual faz a leitura e escrita no barramento e troca informações com o banco de registradores e ADs. O espaço denominado de *Power Electronics Hardware* é uma entidade para ser descrita em VHDL pelo usuário da plataforma, sendo que as portas de entrada e saída disponíveis para o desenvolvimento são pré-definidas, facilitando o uso.

### 3.2 HARDWARE DE CONTROLE DO CONVERSOR

O Hardware de Controle do Conversor, denominado de *Power Electronics Control*, é um periférico do tipo escravo que se conecta ao barramento local da Microblaze. Este periférico utiliza como base os periféricos personalizáveis da Microblaze e é dividido em duas partes:

- Área de lógica estática, que possui as conexões entre o barramento, banco de registradores e interface para os conversores AD;
- Área de lógica reconfigurável, que pode ser visto como uma caixa preta (do inglês, *Blackbox*), denominada de *Power Electronics Hardware*, espaço dedicado a implementação do hardware de controle do usuário.

O hardware programável, de maneira análoga a um conversor dedicado, age de forma independente do resto do sistema, trocando informação com o sistema gerencial, *i.e.* Microblaze, através do banco de registradores. Este hardware também é responsável pela requisição de dados dos ADs, sendo assim, possibilita fazer o ciclo de aquisição de dados analógicos, realizar os cálculos inerentes da técnica de controle implementada e após gerar as saídas para o sistema controlado.

O banco de registradores possui uma quantidade de registradores fixa e foi definido com dezesseis registradores. Esta escolha foi tomada para ser possível implementar algoritmos de controle com poucas variáveis, caso seja utilizado algoritmos de maior complexidade deve-se alterar a área de lógica estática para comportar mais posições no banco. O banco de registradores está organizado da seguinte maneira:

- O registrador zero é usado para configurações do hardware, como, por exemplo, habilitação das entradas e saídas da parte reconfigurável.
- Os registradores de um até sete podem ser escritos somente pelo microprocessador, *i.e.* entradas do hardware de controle, e podem ser utilizados para parâmetros configuráveis do sistema de controle, como ganhos, constantes, etc.
- Os registradores de oito até doze são do modo de somente leitura do microprocessador, *i.e.* saídas do hardware de controle, e podem ser utilizados para monitoramento de valores internos do conversor implementado.
- Os registradores 13 até 15 estão conectados diretamente aos valores de conversão de AD, também para monitoramento.

O hardware de controle possui oito ainda saídas para o mundo externo, podendo assim implementar, por exemplo, o controle de chaves através de um PWM digital. Também estão inclusos no periférico três componentes que realizam a interface com os CIs de conversão AD, sendo disponibilizado para o usuário a aquisição dos conversores através de uma interface simplificada, sem a necessidade de implementação de protocolos de comunicação adicional. Na Figura 16 pode ser visto em detalhes os principais componentes e as interconexões do periférico.

Do ponto de vista do usuário da plataforma, para gerar o hardware do controlador desejado, é necessário entender como as conexões do sistema devem ser utilizadas. Sendo assim um conjunto pré-definido de portas de entrada e saída (E/S) deve ser respeitado, bem como o controle correto dos sinais enviados e recebidos dessas E/S. No Apêndice A é fornecido um código exemplo na linguagem VHDL, podendo ser utilizado como referência para projeto de controladores.

A Figura 17 representa as portas de E/S que devem ser seguidas e os detalhes de utilização de cada porta é apresentado a seguir:

- `clk`, `rst`, `en`:  
Sinais de entrada do bloco, em ordem: ciclo de relógio do sistema, sinal de reinício (*reset*) e sinal de habilitação. Estes sinais são usados para implementação do circuito sequencial, como máquina de estados e registradores do controle do conversor.
- `ADn_req`:

Sinais de saída para requisição de conversão dos conversores analógico-digital, onde  $n$  é o índice do AD desejado, sendo os valores possíveis entre 0 e 2, ativo alto.

- **AD $n$ \_rdy:**  
Sinal de entrada indicando que o dado de conversão do AD de índice  $n$  (de 0 a 2) está disponível, ativo alto.
- **AD $n$ DAT $m$ :**  
Dado de entrada com 14 bits de largura representando o valor, em complemento de dois, da aquisição do AD  $n$  (de 0 a 2), sendo  $m$  a porta do AD indo de 0 a 1.
- **reg $n$ :**  
Sinais do banco de registradores com 32 bits de largura. Para valores de  $n$  entre 1 e 7, dados de entrada (somente leitura), podendo ser utilizado para variáveis controladas pelo usuário, como ponto de operação do controlador, etc. Para valores de  $n$  entre 8 e 12, informações de saída (somente escrita), para utilização de informação de valores internos do controle implementado, como valor RMS da tensão de entrada, etc.
- **switch $n$ :**  
Sinais de saída de um bit para o conversor estático, podendo ser utilizado, por exemplo, para a saída do PWM.

Figura 16 – Periférico de controle do conversor.

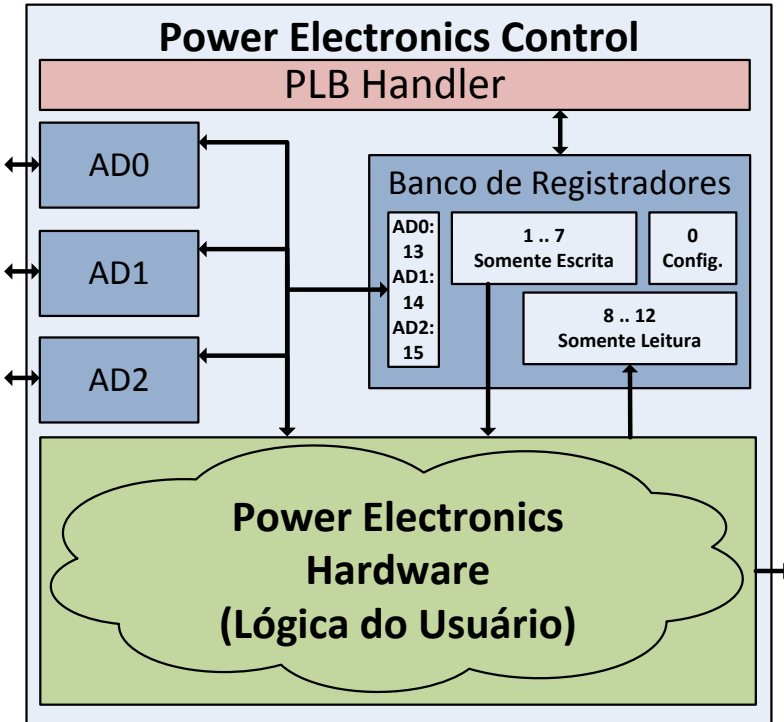


Figura do autor.

Figura 17 – Representação das portas de entrada e saída da área de controle do conversor.

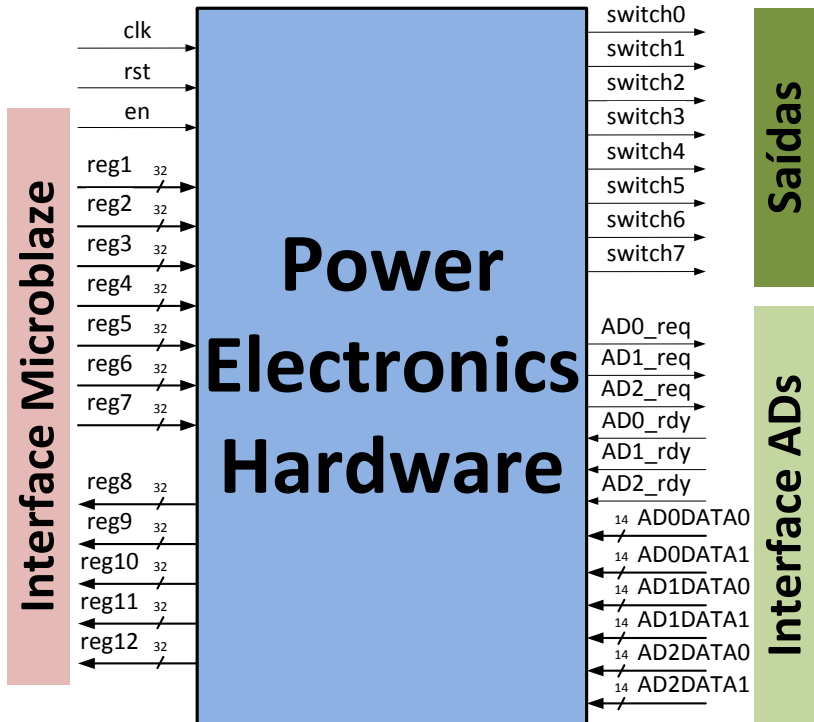


Figura do autor.

### 3.3 SOFTWARE DE GERÊNCIA EMBARCADO E MONITORAMENTO

Uma vez definido o hardware estático que é programado no FPGA, é necessário desenvolver a rotina que o microprocessador irá executar para monitorar e gerenciar as informações. Para isso, inicialmente deve-se gerar um conjunto de bibliotecas e instruções específicas desse hardware, como tipo de processador, quantidade de memória do sistema, mapeamento em memória dos registradores do sistema, periféricos disponíveis, dentre outros. Este conjunto de informações é denominado de *Base System Builder* (BSB). O BSB para essa plataforma é gerado a partir das informações do sistema criado pela ferramenta XPS e serve como base para compilar e gerar o código que se deseja utilizar.

O software de gerência embarcado e o software de monitoramento são executados em plataformas distintas, trocando informação através da rede. No caso, o software de gerência embarcado é executado pelo processador Microblaze no FPGA, e o software de monitoramento é um aplicativo que necessita de um PC x86 e um Sistema Operacional (S.O.) (Windows ou Linux). Ambos são descritos em maiores detalhes nas seções que seguem.

### 3.3.1 Software Embarcado de Gerência

O software embarcado do presente trabalho foi desenvolvido na linguagem C utilizando a ferramenta da empresa Xilinx denominada *Software Development Kit* (SDK) [32]. Este aplicativo possui diversas facilidades de uso como edição de código fonte, compilação automática, bibliotecas padrão, *drivers* de dispositivos (*device drivers*), etc. Como base de desenvolvimento foi utilizado o micro-*kernel* de tempo real disponibilizado pela Xilinx, chamado de Xilkernel [41], que implementa uma API (do inglês, *Application Programming Interface*) POSIX (do inglês, *Portable Operating System Interface*) [42], facilitando o desenvolvimento de código, devido a interoperabilidade imposta para sistemas padrão UNIX.

O software embarcado de gerência têm as seguintes funcionalidades principais:

- Gerenciar, receber, armazenar e apagar os arquivos de configuração disponíveis;
- Fazer, a partir da memória de armazenamento, a reconfiguração do FPGA;
- Servir como interface para comunicar com a rede;
- Disponibilizar informações sobre o estado atual do conversor para fins de monitoramento;
- Modificar valores de referência utilizados no hardware de controle.

Para realizar tais funcionalidades, a rotina de execução do software funciona da seguinte maneira: após o sistema ser ligado são executadas as rotinas de inicialização do sistema, sendo elas a inicialização da conexão com a rede, habilitação do hardware do usuário, inicialização do componente SysAce, e inicialização do componente Hw ICAP.

Após todos os componentes serem inicializados, o software é dividido em três funções que são executados em paralelo, através da



programação com *threads*. A primeira *thread* é um serviço de *telnet* que provê uma interface de acesso ao conversor no modo texto ao usuário através da rede. Sendo possível se conectar ao conversor através de um terminal e modificar os parâmetros desejados sem auxílio de um software adicional no PC. A segunda *thread* é responsável pelo envio e recebimento dos arquivos de *bitstream* que o conversor pode utilizar. Esta *thread* utiliza para a transferência de dados o protocolo TFTP (do inglês, *Trivial File Transfer Protocol*) [43], sendo a sua escolha devido à simplicidade de implementação.

A terceira *thread* é uma implementação análoga ao serviço de *telnet*, denominado de *frontend*, direcionado para comunicação específica com uma interface gráfica. Através desse processo o Software de Monitoramento gerencia o conversor de potência pela rede com uma interface amigável ao usuário, que deve executar um aplicativo específico no PC. A Figura 18 demonstra a sequência de execução das tarefas do software embarcado.

Figura 18 – Sequência de inicialização do software embarcado.

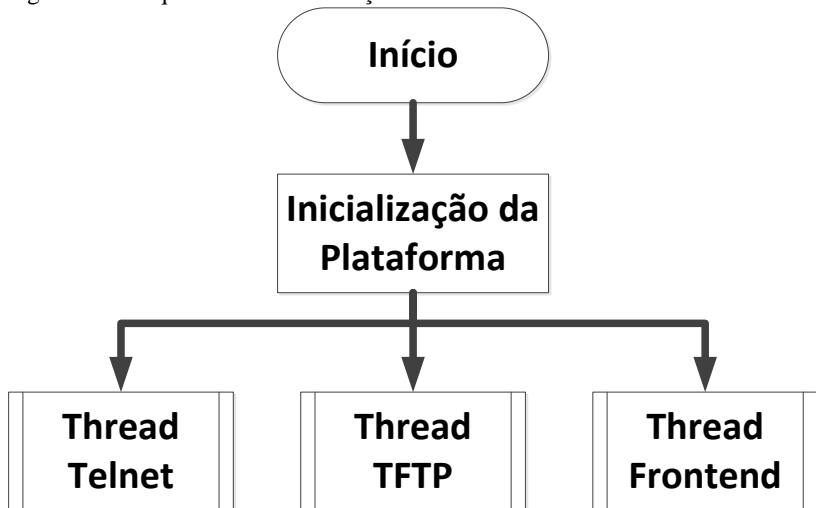


Figura do autor.

As *threads* de *telnet* e *frontend* possuem basicamente as mesmas funcionalidades, sendo os responsáveis por visualizar e modificar os valores do banco de registradores, habilitar ou desabilitar o hardware implementado no conversor e executar a rotina de reconfiguração parcial do hardware. Como já mencionado, o hardware de controle do conversor age de maneira independente do sistema embarcado. O

software embarcado pode somente enviar valores ao controle através da interface do banco de registradores e alterar os sinais de habilitação do hardware.

Quando o usuário deseja alterar o hardware de controle – seja através da interface gráfica ou texto – o software embarcado recebe o comando via rede e desabilita o conversor (através do sinal de *en*), para após realizar a leitura do arquivo *bitstream* desejado, que deve estar localizado na memória de armazenamento (gerenciada pelo periférico SysAce). Após ter acesso ao arquivo de configuração, o *bitstream* é enviado ao periférico Hw ICAP que realiza a reconfiguração da memória do FPGA. Ao final, o hardware de controle do conversor é habilitado novamente, voltando ao estado de funcionamento normal.

### 3.3.2 Software de Monitoramento

Atuando juntamente com o software de gerência desenvolvido para a plataforma, uma interface gráfica foi desenvolvida para facilitar a utilização da plataforma por parte do usuário final. Essa interface foi escrita na linguagem C++ utilizando a API Qt, que é proprietária, porém gratuita, da empresa Digia [44]. A escolha desse conjunto de linguagem e bibliotecas se deu, principalmente, pelo fato de serem disponibilizados compiladores para as plataformas Windows e Linux, sem necessidade de modificações nos códigos fonte do software.

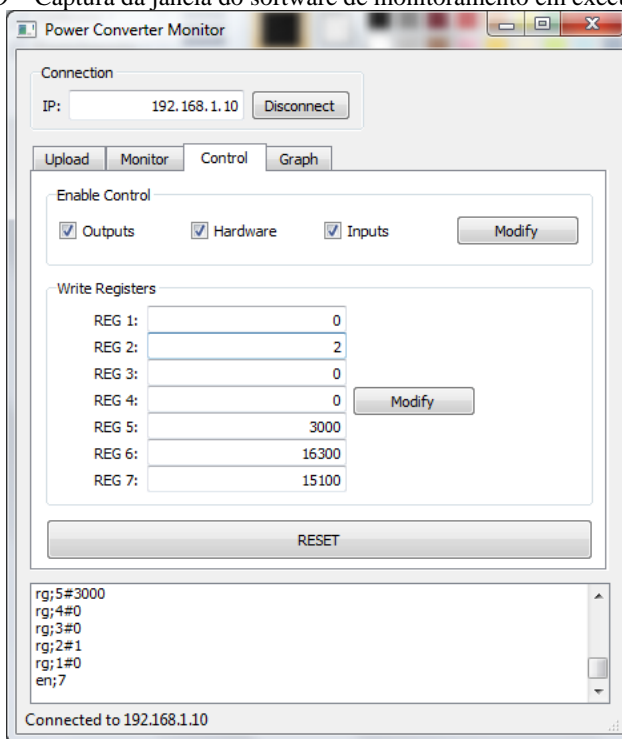
O software de monitoramento do conversor tem as seguintes funcionalidades:

- Mostrar para o usuário os arquivos de configuração disponíveis para realizar a reconfiguração;
- Enviar o arquivo *bitstream* para reconfiguração parcial do FPGA;
- Enviar o comando para a reconfiguração do FPGA;
- Monitorar os valores do banco de registradores;
- Enviar os valores para escrita nos registradores desejados;
- Enviar o comando de habilitação das entradas, saídas e hardware de controle do conversor;
- Reinicializar o controle do conversor;

Para ser possível conectar a plataforma desenvolvida através do software de monitoramento, o PC que executa o software e a plataforma do conversor devem estar na mesma rede, com endereços de IP conhecidos. Assim, basta digitar o endereço IP da plataforma no

software de monitoramento e, após conectados, a comunicação ocorre através de troca de mensagens via *sockets* POSIX. Essa troca de mensagens é feita no modo texto, sendo que nenhum protocolo específico foi utilizado, visando a facilidade e rapidez de implementação para validar os resultados do sistema como um todo. Uma tela do software de monitoramento sendo utilizado para o controle de um conversor pode ser vista na Figura 19.

Figura 19 – Captura da janela do software de monitoramento em execução.



### 3.4 FLUXO PROPOSTO PARA DESENVOLVIMENTO

Esta seção descreve a metodologia que deve ser adotada para geração de descrições de hardware compatíveis com a plataforma desenvolvida. Como descrito nas seções anteriores, a plataforma possui uma área de hardware estática que deve ser compatível com o hardware reconfigurável. O fluxo proposto para esse trabalho é dividido em duas etapas, semelhante ao fluxo descrito na Seção 2.3.

Na primeira parte, deve ser gerada uma descrição de hardware, e.g. VHDL, com a interface de entrada e saída compatível. A partir dessa descrição de hardware, a etapa de síntese lógica deve ser realizada para obter um *netlist*. A diferença entre a síntese lógica para a plataforma e para um FPGA tradicional é que a opção de não incluir na *netlist* os IOBs deve estar habilitada. Por se tratar de um hardware reconfigurável, os IOBs já estão presentes no hardware estático.

Para realizar este passo foi utilizada a ferramenta ISE (*Integrated Synthesis Environment*) [45] da empresa Xilinx, a qual é composta por diversos aplicativos que, juntos, realizam o fluxo de desenvolvimento para FPGAs. Porém, somente a etapa de síntese lógica dessa ferramenta é utilizada, mais especificamente o software XST (*Xilinx Synthesis Technology*) [46] é necessário. Sendo assim, pode-se utilizar da interface gráfica do ISE ou através de linha de comando da ferramenta XST.

Como um dos requisitos da plataforma é facilitar o desenvolvimento de conversores de potência por usuários não muito familiarizados com as ferramentas de EDA para FPGA, é disponibilizado também um *script* (para Windows no formato de arquivo em lotes, i.e. extensão bat, ou Unix um *shell script*, i.e. extensão sh) que faz a chamada do software XST, gerando a *netlist* a partir do(s) arquivo(s) de entrada(s).

Na segunda parte do fluxo, o arquivo *netlist* com o hardware desejado deve passar pela implementação e síntese física para gerar os *bitstreams* completo e parcial que configuram o FPGA. Para essa etapa a ferramenta PlanAhead [47] da empresa Xilinx deve ser utilizada. Esta ferramenta é um ambiente de desenvolvimento para FPGAs, semelhante ao ISE, porém mais avançado, que oferece suporte ao fluxo de desenvolvimento de reconfiguração parcial.

Logo, um projeto do PlanAhead previamente configurado é disponibilizado, no qual uma nova configuração (*Design Run*) deve ser criada, informando que os arquivos da parte estática serão reutilizados (ou importados). Feito isso, é somente necessário executar o novo *Design Run* que os arquivos *bitstream* serão criados. Novamente, com foco na facilidade de uso da plataforma, é disponibilizado um *script* na linguagem Tcl para o usuário, sendo somente necessário abrir o arquivo (executar o *script*) a partir do software PlanAhead que os arquivos *bitstream* são gerados.

Portanto, o usuário que utiliza a plataforma somente necessita desenvolver sua descrição de hardware seguindo as especificações de entrada e saída da plataforma, e após executar dois *scripts* que fornecem

os arquivos de configuração do FPGA. Feito isso o FPGA pode ser reconfigurado através da rede pelo software de monitoramento.

A Figura 20 apresenta o diagrama do fluxo de desenvolvimento proposto. O fluxo é iniciado com a codificação RTL (por exemplo, descrição do hardware usando VHDL). Após é necessário verificar se a entidade topo do hardware é compatível com a interface da plataforma. Feito isso, é gerado a *netlist* através do fluxo da ferramenta ISE. Sendo o arquivo *netlist* (com extensão *ngc*), utilizado como entrada no projeto base da plataforma, sendo executado o fluxo do software PlanAhead. A saída resultante são os dois *bitstreams* para o FPGA. O completo é a configuração da parte estática juntamente com o novo hardware de usuário desenvolvido, podendo ser utilizado como a inicialização padrão da plataforma. O *bitstream* parcial é utilizado para substituir o hardware já funcionando na plataforma, podendo ser enviado pelo software de monitoramento via rede. No Apêndice C é disponibilizado um tutorial passo a passo de como realizar esse processo utilizando os *scripts* para as ferramentas.

Figura 20 – Diagrama do fluxo de desenvolvimento proposto.

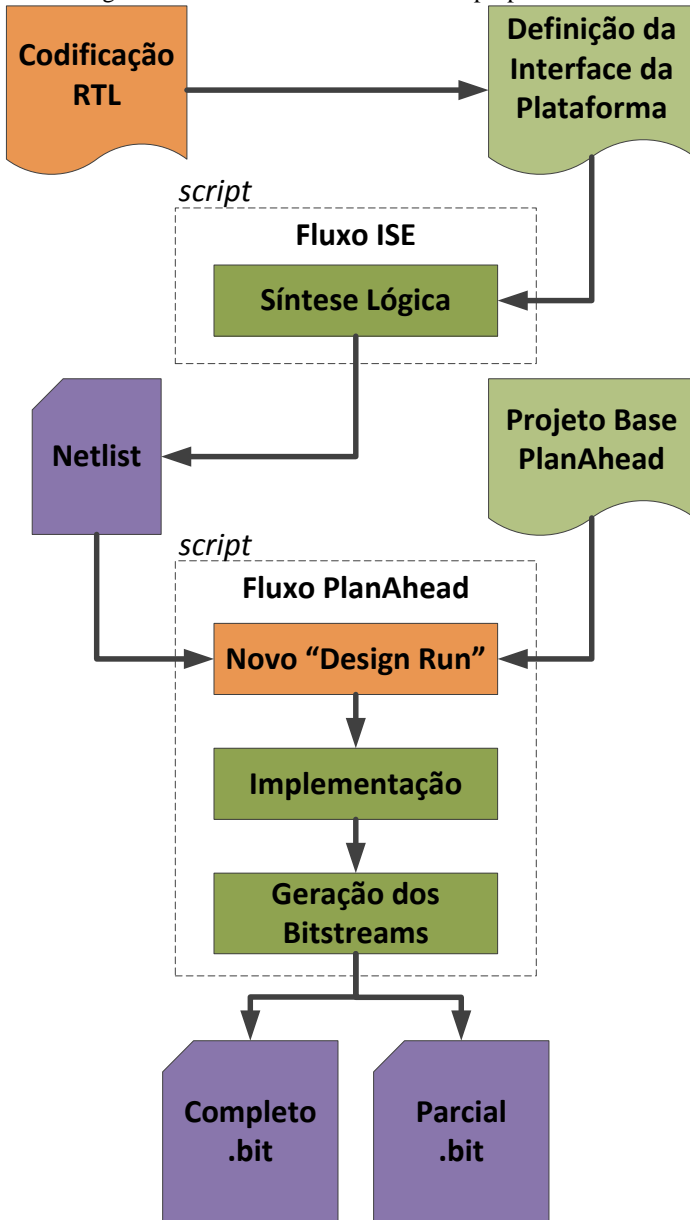


Figura do autor.

## 4 APLICAÇÃO DA PLATAFORMA NO CONTROLE DO RETIFICADOR BOOST

Para ser possível realizar os testes e validação da plataforma de desenvolvimento proposta neste trabalho, foi utilizado como estudo de caso a implementação do controle de um retificador boost usando a plataforma desenvolvida como base. Como o foco de desenvolvimento da plataforma não se concentrou no estudo da(s) técnica(s) de controle e topologias dos circuitos de eletrônica de potência, o conversor de energia foi compartilhado do trabalho resultante da dissertação de mestrado do PPGEEL/UFSC intitulada “Análise do retificador boost sob interrupções instantâneas da tensão de alimentação” [48].

A escolha de utilizar esse trabalho como estudo de caso se deu por dois motivos: primeiramente ao fato da topologia do conversor de potência já ter sido confeccionada e também validada através de uma implementação em FPGA, minimizando o desenvolvimento de circuitos de para eletrônica de potência, sendo possível fazer o uso da mesma placa com o sistema de potência. E também a existência de documentação adequada com a descrição, em detalhes, da implementação discreta do controle do conversor de potência.

Para realizar a implementação do controle do conversor, duas placas de circuito impresso foram utilizadas, uma contendo o sistema de potência (já feita pelo trabalho anterior), e outra responsável pela instrumentação desenvolvida durante o presente trabalho de mestrado, devido à incompatibilidade de pinos entre a placa de instrumentação pré-existente. Entretanto a placa de instrumentação manteve o mesmo esquemático do trabalho anterior, sendo apenas realizado um novo desenho e roteamento.

Na Figura 21 é apresentado uma visão geral do controle do retificador boost que foi implementado para a plataforma de desenvolvimento aqui proposta. Na parte superior da Figura 21 são apresentadas as conexões da placa de potência, sendo possível identificar os pontos de conexão como a entrada e saída de potência, locais onde são adquiridos níveis de corrente e tensão, e onde o controle atua. No centro da figura os módulos principais que representam a placa de instrumentação, e por fim, na parte inferior pode ser visto a representação do FPGA, componente principal da plataforma, bem como os módulos indicando o controle e monitoramento.

Figura 21 – Visão geral do controle do retificador boost aplicado na plataforma.

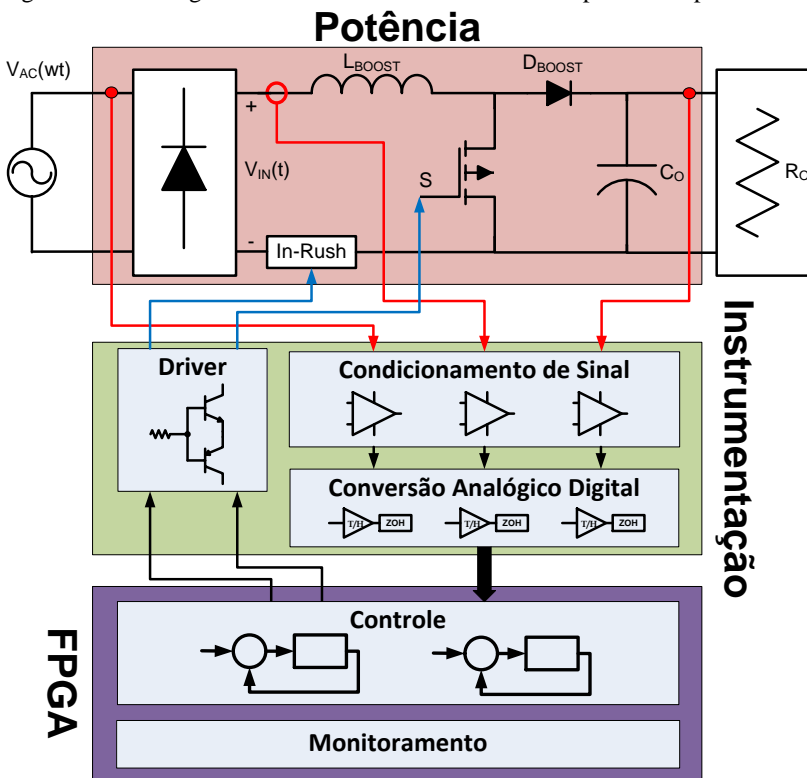


Figura do autor.

Nas seções que seguem são apresentados em detalhes a topologia do retificador boost bem como uma breve explicação de seu funcionamento. Também os esquemáticos dos circuitos de instrumentação responsáveis pelo condicionamento de sinal no sentido conversor para controle, *i.e.* aquisição das tensões e correntes para o FPGA, e controle para o conversor, *i.e.* pulsos modulados e sinais de controle do FPGA para o conversor. E por fim uma exemplificação do cenário de uso da plataforma.

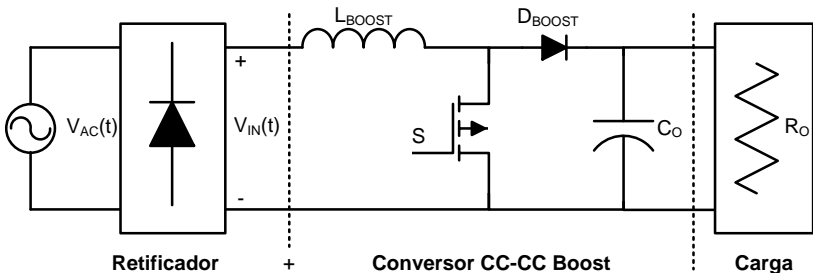
#### 4.1 TOPOLOGIA E FUNCIONAMENTO DO RETIFICADOR BOOST

O retificador boost é composto pela junção de um retificador de onda completa em cascata com o conversor C.C.-C.C. boost, resultando em um retificador boost unidirecional como demonstrada na Figura 22. É possível afirmar que esta topologia está bastante consolidada em



aplicações de correção de fator de potência (em inglês, *Power Factor Correction – PFC*) em sistemas C.A-C.C., como consequência de um volume elevado de trabalhos desenvolvidos [6,49–52]. O conversor boost também é conhecido como elevador de tensão (em inglês, *step-up converter*), pois tem a característica de sempre oferecer à carga uma tensão maior que a tensão de entrada do conversor. Como especificação de projeto de [48], o desenvolvimento do conversor é analisado somente para o caso de modo de condução contínua. O modo de condução descontínua é desaconselhado para conversores com potência nominal acima de 600 watts, devido aos altos valores de corrente de pico no interruptor [23,48].

Figura 22 – Retificador boost.



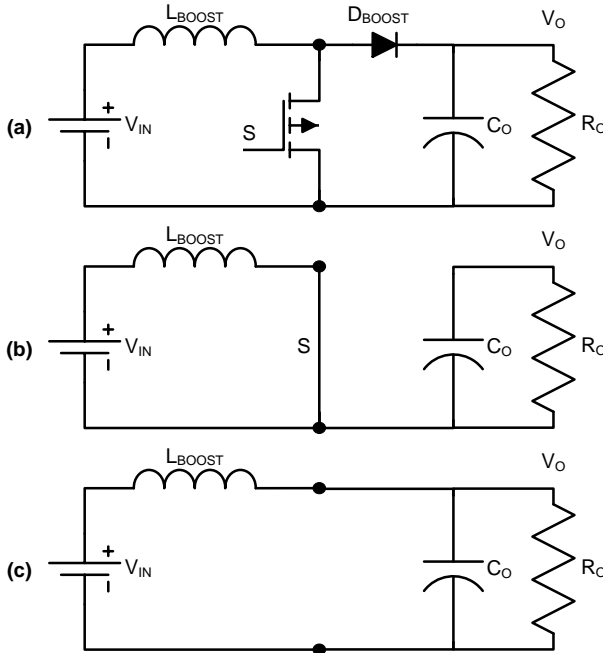
Adaptado de [48].

Para fazer uma análise do funcionamento e determinar um modelo equivalente do conversor, as seguintes hipóteses simplificativas, retiradas de [48], foram adotadas:

- Em função da frequência de comutação do conversor ser maior que a frequência da tensão de alimentação ( $f_s \gg f_{rede}$ ) então, a tensão de entrada do conversor boost é definida como constante em cada intervalo de comutação.
- Os dispositivos semicondutores são considerados ideais, ou seja, apresentam impedância infinita quando em estado de bloqueio e impedância nula quando em condução.
- As resistências intrínsecas dos componentes, assim como as resistências parasitas, estão sendo rejeitadas e desprezadas nesta abordagem inicial.
- A tensão no capacitor  $C_O$ , de saída do conversor, é considerada constante e isenta de ondulações.

Sendo assim é possível obter os circuitos equivalentes apresentados na Figura 23.

Figura 23 – (a) Modelo equivalente do retificador boost, (b) interruptor em condução e (c) interruptor aberto.



Adaptado de [48].

A característica fundamental de funcionamento do conversor boost está na tendência dos indutores resistirem a mudanças de corrente. O princípio básico de funcionamento consiste em dois estados distintos: 1 – No estado ligado, apresentado pela Figura 23(b), a chave está no modo de condução, resultando em um aumento da corrente no indutor ( $L_{BOOST}$ ); 2 – No estado desligado, Figura 23(c), a chave está aberta e o único caminho disponível para a corrente do indutor é através do capacitor ( $C_O$ ) e a carga ( $R_O$ ). Resultando na transferência da energia acumulada durante o estado ligado para o capacitor.

Se a chave for comutada em um tempo suficientemente rápido, o indutor não perderá totalmente sua energia entre as trocas de estado e a carga sempre verá uma tensão maior que a tensão de entrada. Quando a chave é aberta, o capacitor em paralelo com a carga é carregado com essa energia.

O emprego de conversores estáticos de potência com correção ativa do fator de potência (PFC) é uma solução frequentemente utilizada no processo de conversão de energia de modalidade C.A. para C.C.. Comparativamente aos retificadores não controlados à diodos, conversores com a correção ativa do fator de potência viabilizam: drenar correntes senoidais com reduzido conteúdo harmônico, desfasamento praticamente nulo com a tensão de suprimento e controle da tensão de saída [48,53].

Em aplicações de baixa e média potência monofásicas, o retificador boost é uma topologia geralmente empregada como pré-regulador de fontes de alimentação. Esta topologia, demonstrada na Figura 22, apresenta as seguintes características:

- Necessita de apenas um diodo com comutação rápida ( $D_{\text{boost}}$ );
- Possui apenas um interruptor comandado (S);
- O sinal de comando para o interruptor e os pontos de medição da tensão de entrada retificada, da tensão de saída e da corrente no indutor podem ser todos referenciados ao terminal negativo da saída do conversor;
- Pode operar com estratégias simples de modulação e controle, frequentemente utilizando para isto circuitos integrados de baixo custo;
- Apresenta baixo nível de ruído de modo comum se comparada a estruturas mais complexas.

A Figura 23(b) demonstra que no período em que o interruptor está em condução, a corrente no indutor cresce de acordo com a seguinte equação:

$$\frac{di_L(t)}{dt} = \frac{V_i}{L}$$

Enquanto que na hipótese do interruptor bloqueado, apresentado na Figura 23(c), a corrente decresce com a seguinte relação:

$$\frac{di_L(t)}{dt} = \frac{V_i - V_o}{L}$$

Onde, nas equações apresentadas, temos:

$V_o$  – Tensão de saída.

$V_i$  – Tensão de entrada.

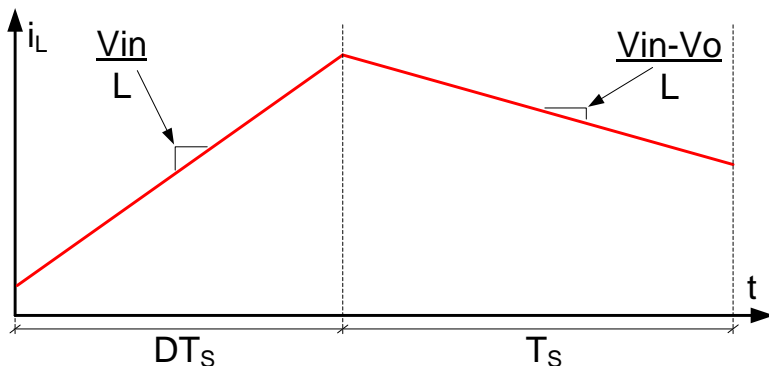
$L$  – Indutor boost.

$i_L$  – Corrente no indutor boost.

Sendo assim, a Figura 24 representa graficamente a corrente no indutor em função do tempo, nos dois momentos distintos de

funcionamento, ou seja, quando o interruptor está comandado ( $DT_S$ ), e após ser aberto ( $T_S$ ).

Figura 24 – Gráfico da corrente no indutor boost em função do tempo.



Adaptado de [53].

Nestas estruturas, a corrente de entrada é controlada ativamente através da imposição de diferentes níveis de tensão sobre o indutor (indutor boost), utilizando-se semicondutores de potência comandados (interruptores) e esquemas de modulação diversos. A cada período de comutação, a tensão de entrada do conversor é considerada constante assim como a tensão de saída. A estratégia de controle impõe diferentes tempos de condução ao interruptor estático com o objetivo que seja possível sintetizar, em baixa frequência, a corrente senoidal com os objetivos requeridos. Neste sentido, este controle ativo da corrente de entrada visa a obtenção de um fator de potência elevado e a redução do conteúdo harmônico da corrente injetada na rede.

## 4.2 CONVERSOR ESTÁTICO DE POTÊNCIA

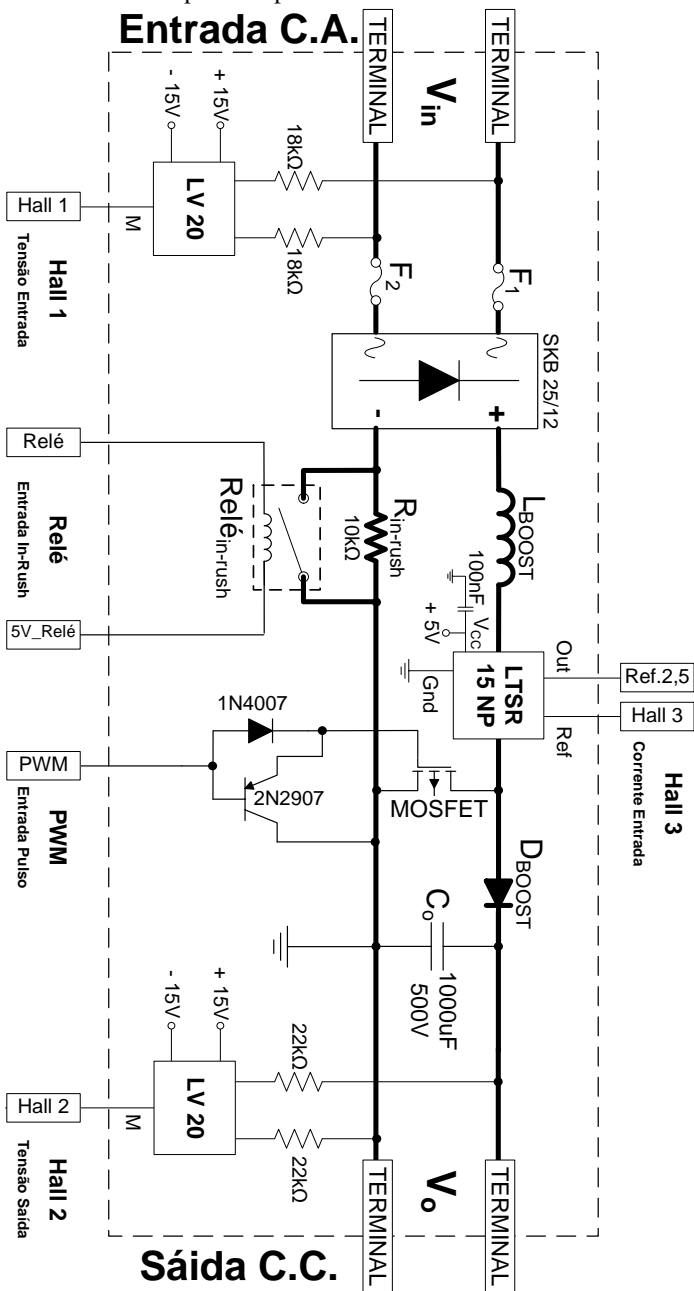
O circuito que representa a topologia do retificador boost, o qual é responsável pela parte de potência do conversor, é apresentado em detalhes na Figura 25. Esta placa e seus componentes são os mesmos que foram utilizados em [48]. Além do interruptor boost, indutor boost, capacitor de saída e ponte retificadora a quatro diodos – os quais compõem o retificador boost – componentes adicionais são necessários para o conversor poder ser controlado da maneira desejada. O circuito possui dois terminais onde a entrada de C.A. é conectada, neste caso a rede elétrica, e dois terminais para a saída de C.C. onde a carga do conversor é conectada. Além dos terminais de entrada e saída de

potência, o circuito possui cinco pontos de conexão com a placa de instrumentação.

Sendo três destas conexões as saídas da placa de potência para a placa de instrumentação, com os valores obtidos dos sensores de efeito hall da tensão de entrada, tensão de saída e corrente de entrada. Identificados na figura como Hall 1, Hall 2 e Hall 3. Os outros dois pontos de conexão são entradas provenientes da placa de instrumentação para a placa de potência, com o pulso de comando que controla a chave do conversor boost e o conjunto resistor/relé para evitar o efeito de *inrush*, representados pelas conexões PWM e RELÉ, ainda na Figura 25.

O efeito de *inrush* ocorre no estado inicial do conversor, ou seja, quando este está desconectado da rede e não há valores de corrente e tensão no circuito. Como um capacitor descarregado possui uma característica inicial de um curto circuito, e o interruptor do conversor inicia no modo aberto, o circuito é semelhante ao apresentado na Figura 23(c). Esse conjunto de fatores causa um curto circuito momentâneo na fonte, e para dissipar o pico de corrente, um resistor em série é utilizado na placa. Após, o relé realiza a função de *bypass*, pois está conectado em paralelo com este resistor, devendo ser acionado logo após os primeiros instantes de alimentação do conversor, deixando o conversor boost em modo de funcionamento.

Figura 25 – Circuito da placa de potência.



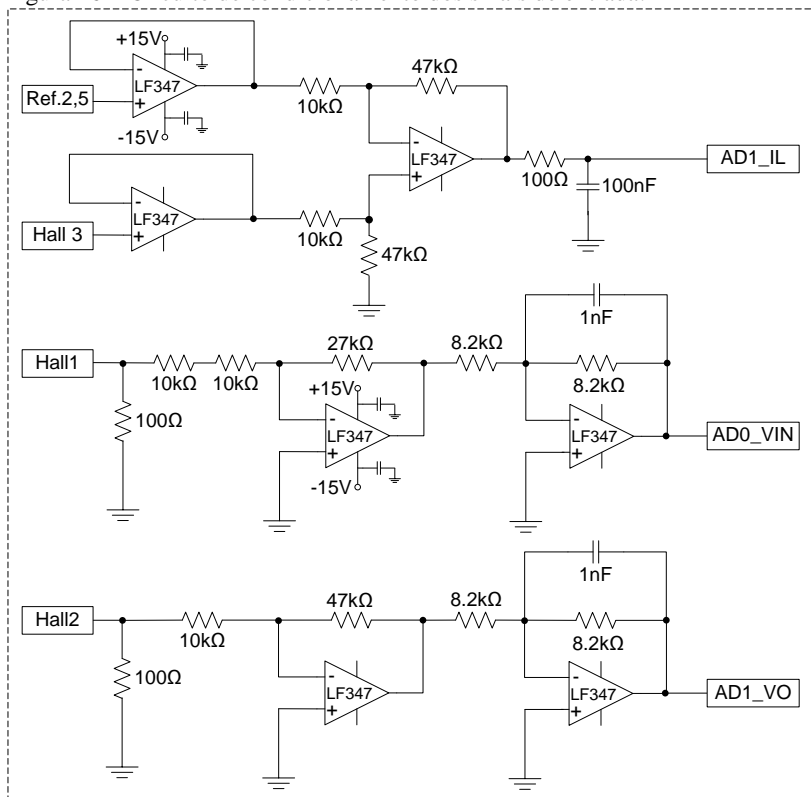
### 4.3 CIRCUITO DE CONDICIONAMENTO DOS SINAIS DE ENTRADA

O condicionamento de sinais de entrada faz parte da primeira de três subdivisões da placa de instrumentação. Estas operações são realizadas por amplificadores operacionais e componentes discretos, tendo a função de aplicar ganhos e realizar o filtro de *anti-aliasing*, deixando o sinal dentro dos limites aceitos pelos conversores AD. Os sinais de corrente de entrada e tensão de saída possuem sempre valores positivos e, portanto, são condicionados a ficarem entre 0V e +10V. Já a tensão de entrada é uma senoide que atinge valores negativos, por isso o sinal é condicionado para ficar entre -5V e +5V.

O transdutor hall do sinal da corrente fornece um valor de referência de 2,5V que deve ser subtraído do valor medido. Na Figura 26, o condicionamento do sinal da corrente é representado pelo circuito do topo contendo três amplificadores operacionais, sendo os dois primeiros atuando como *buffers* e o terceiro realiza o cálculo da diferença entre o sinal de referencia e o valor obtido pelo sensor Hall 3, por fim um filtro *anti-aliasing* deixa o sinal de corrente pronto para ser adquirido.

Os transdutores de tensão podem ser vistos como uma fonte de corrente, portanto nos dois circuitos seguintes de condicionamento, um resistor é usado para medir tensão equivalente, sendo a entrada do primeiro amplificador operacional, que aplica um ganho no sinal e um segundo amplificador operacional realiza a operação de filtro *anti-aliasing*.

Figura 26 – Circuito de condicionamento dos sinais de entrada.

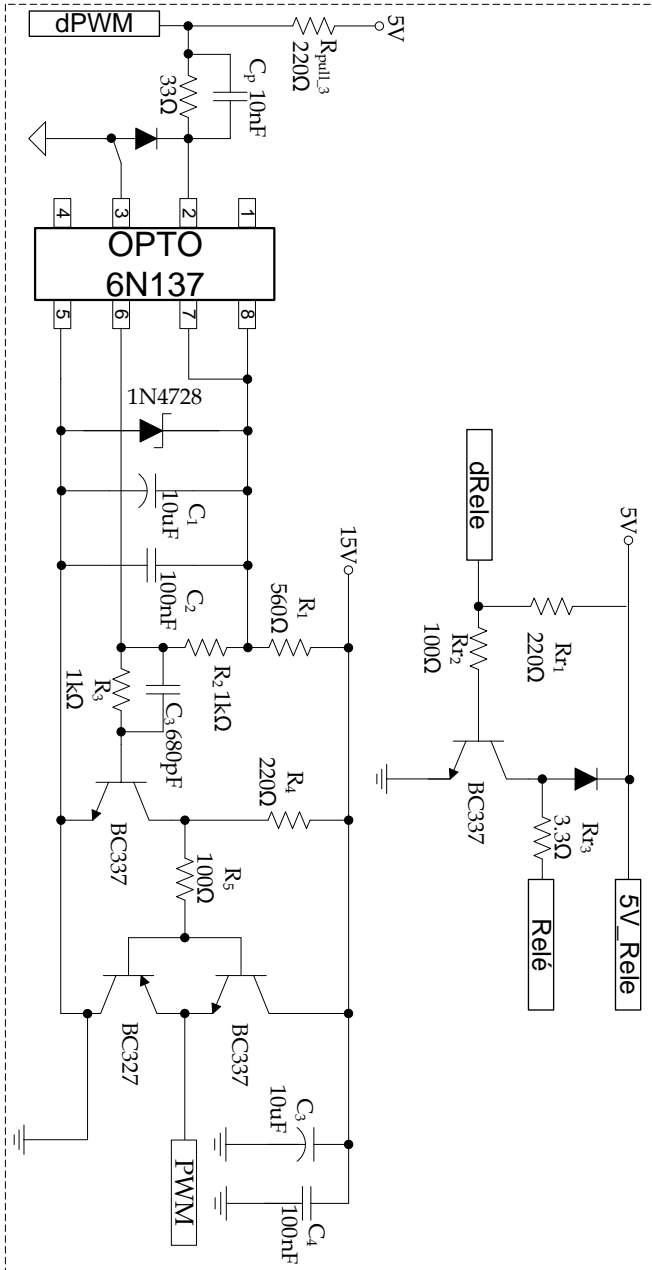


#### 4.4 CIRCUITO DE CONDICIONAMENTO DOS SINAIS DE SAÍDA

Nesta topologia, dois sinais são necessários para realizar a atuação no sistema de potência do retificador boost, que são os sinais de PWM e ativação do relé de *in-rush*. Para cada um deles é utilizado um circuito que fornece a potência necessária para a placa de potência. Ambos os circuitos são apresentados, em detalhes, na Figura 27.



Figura 27 – Circuito de condicionamento dos sinais de saída.



O primeiro, e mais extenso, circuito tem a função de ativação do PWM, este circuito utiliza como componente principal um opto acoplador, devido à necessidade de isolamento da chave do conversor. Dessa maneira o hardware do FPGA fica protegido do circuito da parte de potência, melhorando também a resistência a interferências provenientes da comutação da chave. O segundo circuito tem a função de ativar o relé de *in-rush* localizado na placa de potência. Este esquemático é, tradicionalmente, utilizado em controladores digitais. Quando as saídas necessitam ser conectadas a componentes que precisam de uma corrente maior que o pino pode fornecer, um circuito de potência deve ser utilizado.

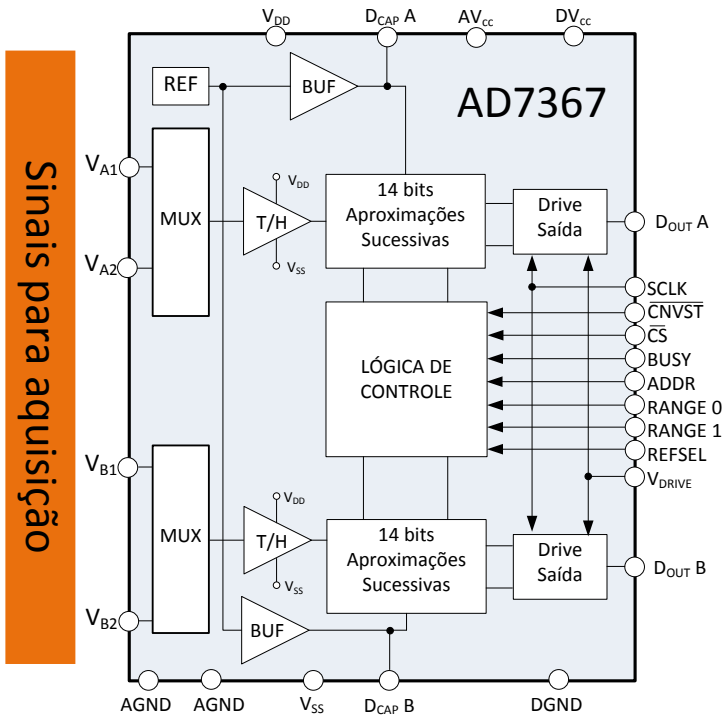
#### 4.5 CONVERSORES ANALÓGICO-DIGITAL

A placa de instrumentação possui ainda os componentes responsáveis pela conversão dos sinais analógicos para digital. Para isso são utilizados dois circuitos integrados modelo AD7367 [54] da empresa Analog Devices. Este componente tem as seguintes características principais:

- Taxa máxima de conversão de 1MSPS;
- Resolução de 14 bits;
- Comunicação através do protocolo SPI (do inglês, *Serial Peripheral Interface*);
- 2 *sample-hold* internos e independentes dispostos em 2 canais multiplexados para cada *sample-hold*;
- Leitura de tensões configurável de  $\pm 5V$ ,  $\pm 10V$  ou  $0V$  até  $10V$ .

Como a especificação do retificador boost requer três valores de aquisição, foi necessário utilizar dois componentes para realizar as três medições no mesmo instante de tempo. Na Figura 28 é apresentada a estrutura interna do dispositivo e também os pinos de entrada e saída que devem ser conectados. À esquerda da figura são identificados os pinos de entrada de sinal analógico os quais o AD faz a conversão. Na parte superior e inferior da figura, os pinos de alimentação são demonstrados. À direita da Figura 28 os pinos que devem ser conectados ao FPGA com a comunicação SPI.

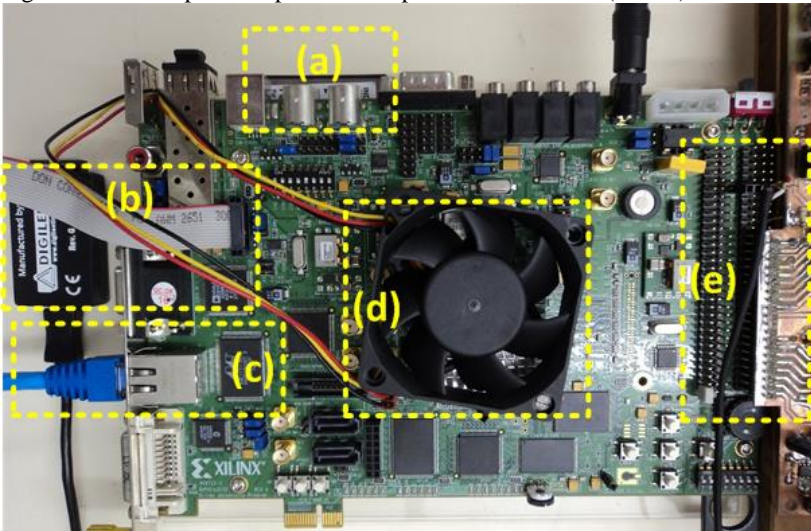
Figura 28 – Estrutura interna do conversor AD7367.



#### 4.6 VISÃO GERAL DA MONTAGEM DO SISTEMA

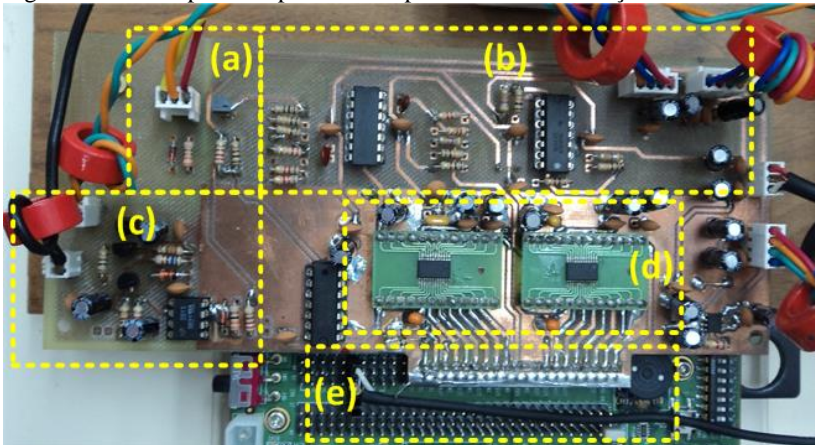
Com o intuito de facilitar o entendimento da maneira em que as placas de potência, instrumentação e o FPGA estão conectadas, a seguir são apresentadas imagens do sistema, destacando os principais componentes em cada parte. Na Figura 29 é apresentada o kit de desenvolvimento XUPV5, em que a plataforma foi baseada, em (a) está localizado, na parte traseira da placa, o cartão de memória CompactFlash responsável pelo armazenamento de arquivos locais. Em (b) é destacada a conexão através da interface JTAG, e em (c) a conexão com a rede Ethernet. Ainda na figura, (d) apresenta a localização do FPGA Virtex-5, e por final em (e) os pinos de uso geral onde é feita a conexão com a placa de instrumentação.

Figura 29 – Principais componentes da plataforma XUPV5 (FPGA).



Na Figura 30 são apresentados os componentes principais da placa de instrumentação. Em (a) o circuito de condicionamento para o relé, (b) os circuitos integrados e componentes discretos que formam o circuito de condicionamento dos sinais de entrada. (c) destaca o circuito de condicionamento para o comando da chave do retificador, em (d) os dois conversores AD e a conexão com o FPGA destacada em (e).

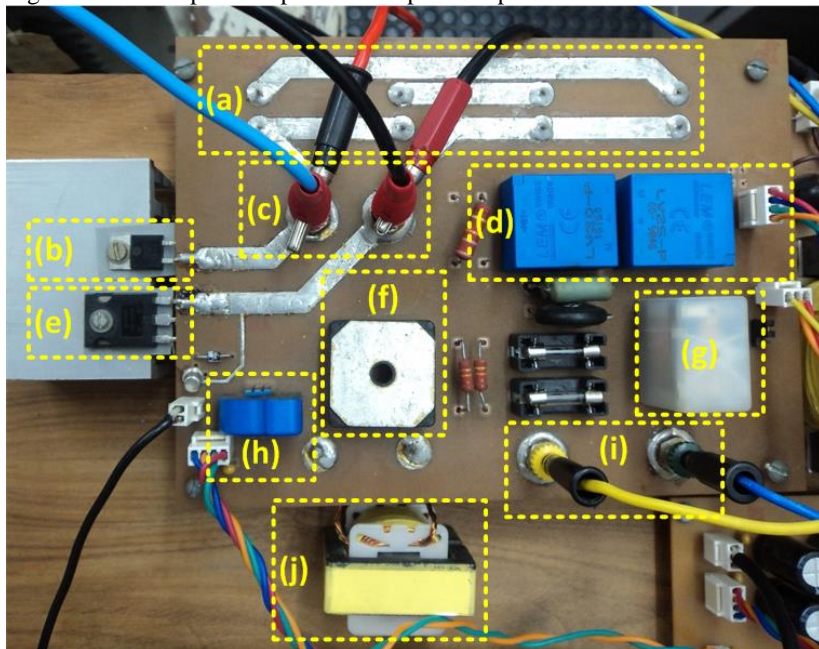
Figura 30 – Principais componentes da placa de instrumentação.



E por fim, na Figura 31 é apresentada a placa do conversor estático de potência, em (a) estão as conexões dos capacitores de saída,

localizados na parte inferior da placa. (b) destaca o diodo boost, (c) os terminais de saída C.C., (d) os transdutores de tensão de entrada e saída do conversor, (e) a chave de comutação (mosfet) do retificador boost, (f) apresenta a ponte de diodos. Ainda na figura, (g) representa o relé do circuito de *inrush*, (h) o transdutor de corrente no indutor, (i) os terminais de entrada de tensão C.A. e por fim o indutor boost em (j).

Figura 31 – Principais componentes da placa de potência.



#### 4.7 CONTROLE DO CONVERSOR BOOST

Nas seções que seguem, as técnicas de controle utilizadas para a topologia do conversor boost são apresentadas, iniciando por um modelo em malha aberta para demonstrar a funcionalidade do conversor trabalhando no modo C.C. – C.C., e após as técnicas que realizam o casamento da correção de fator de potência e controle da tensão de saída do barramento.

### 4.7.1 Malha Aberta (Elevador de Tensão)

Quando considerado somente a simplificação apresentada na Seção 4.1, referente ao conversor boost – isto é, um conversor C.C.-C.C. onde a entrada é uma bateria – é possível operá-lo no modo de malha aberta. Sem a existência da realimentação do controle, este modo de funcionamento é conhecido como elevador de tensão, mas não oferece nenhum tipo de correção de fator de potência ou controle das tensões por meio de referência. As deduções e desenvolvimento das equações não serão demonstradas aqui por não ser o foco do trabalho, maiores detalhes sobre a parte matemática do controle pode ser encontrada em [48,51].

A equação que rege o conversor boost, em modo de condução contínua é dada por:

$$\frac{V_o}{V_i} = \frac{1}{1 - D}$$

Ou ainda em função da razão cíclica:

$$D = 1 - \frac{V_i}{V_o}$$

Onde temos:

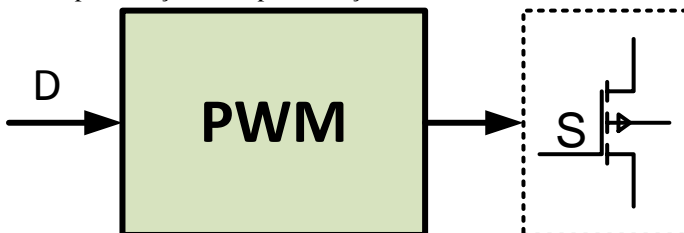
$V_o$  – Tensão de saída.

$V_i$  – Tensão de entrada.

$D$  – Razão cíclica da chave (variando entre 0 e 1).

De tal modo, é possível notar que, para dobrar a tensão de saída é necessário aplicar uma razão cíclica de 50% ( $D/2$ ) na chave do conversor. Este tipo de utilização do conversor boost é bastante utilizado em situações que a bateria do sistema não pode fornecer o nível de tensão necessário, ou para reduzir a quantidade de baterias em série (e por consequência o custo) necessárias no sistema. A Figura 32 apresenta o diagrama de blocos para implementação descrita, onde é aplicado um valor na entrada  $D$ , representando a razão cíclica desejada, fazendo com que a chave receba o pulso de comando equivalente.

Figura 32 – Representação da implementação do controle em malha aberta.

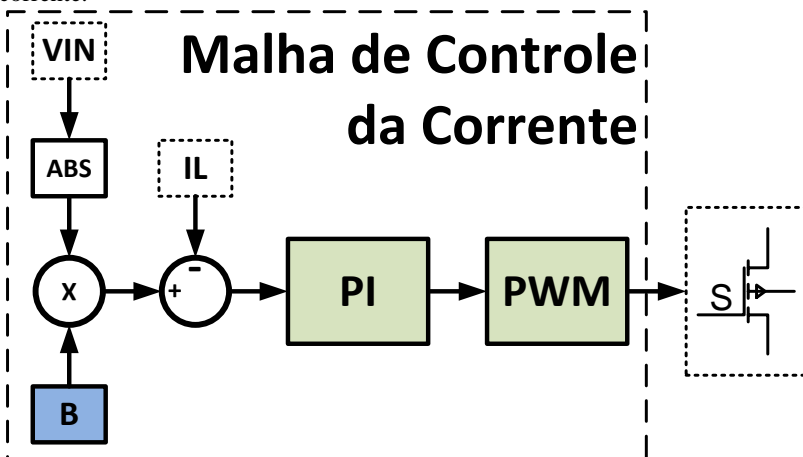


## 4.7.2 Malha de Controle da Corrente

O controle da corrente do conversor boost é feito em uma malha interna de controle, a qual é responsável por impor uma corrente conforme a referência. É utilizada, para esta implementação, a técnica de controle por valores médios instantâneos da corrente no indutor, de acordo com o trabalho desenvolvido em [48]. O funcionamento básico da técnica consiste no monitoramento da corrente no indutor e o controle da comutação em alta frequência do interruptor, fazendo com que esta corrente siga uma referência senoidal com um pequeno de erro.

Esta malha de controle apresenta uma dinâmica rápida, sendo que o controlador deve possuir a capacidade de sintetizar a corrente no indutor conforme a referência senoidal. Para isso a implementação pode ser feita como apresentado na Figura 33, onde as grandezas utilizadas são: tensão de entrada ( $V_{IN}$ ) para realizar a referência senoidal, e a corrente no indutor ( $I_L$ ) para gerar o erro do controlador. A diferença (subtração) entre a corrente no indutor e o resultado da multiplicação do valor absoluto (operação do bloco ABS) da tensão de entrada, por uma constante (representada por  $B$ ), entra no controlador proporcional-integral, o qual gera um sinal de controle para o bloco PWM, que por fim determina o intervalo de tempo para condução do interruptor durante um período de comutação. A constante  $B$  é substituída, na próxima seção, para o controle da tensão de saída. Os detalhes e cálculos dessa técnica podem ser vistos em detalhes nos trabalhos [48,53]

Figura 33 – Diagrama de blocos da implementação do controle da malha de corrente.



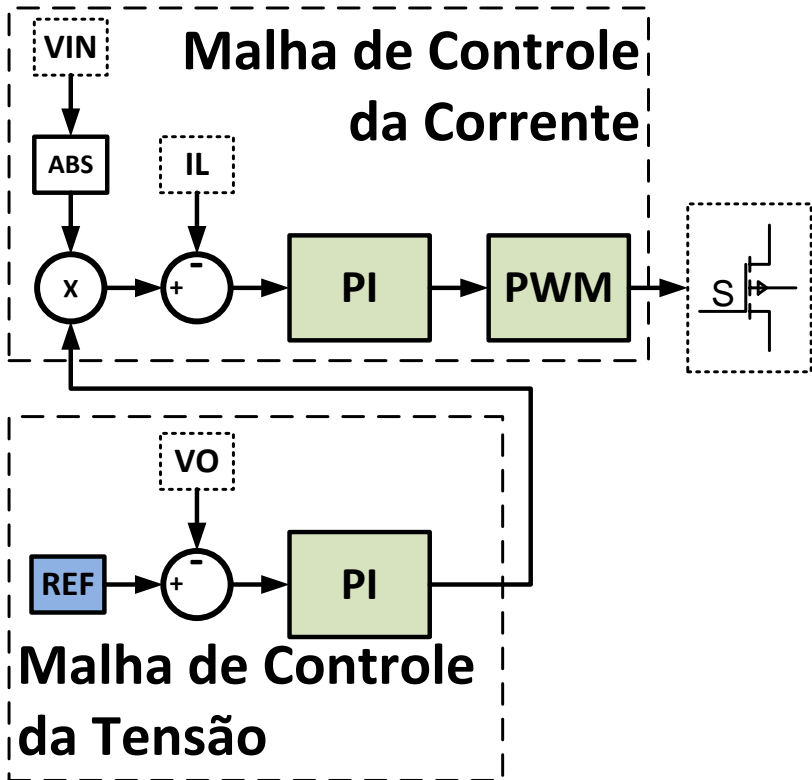
#### 4.7.3 Malha de Controle da Tensão

A malha de controle da tensão de saída atua na amplitude de referência da corrente, controlando a energia transferida entre a fonte de alimentação na entrada e a carga conectada a saída da topologia. Por sua vez, possui uma dinâmica suficientemente lenta com intuito de minimizar as ondulações na referência senoidal [48].

A Figura 34 apresenta uma implementação que compreende as duas malhas de controle, abordagem a qual foi utilizada neste trabalho. Sendo que, o valor que multiplica a tensão de entrada na malha da corrente é o resultado do controle da malha de tensão de saída. O erro utilizado na entrada do controlador proporcional-integral da tensão é a diferença entre uma referência (bloco REF na Figura 34) que representa a magnitude desejada na saída, e o valor lido da tensão de saída.



Figura 34 – Diagrama de blocos da implementação do controle com a malha de tensão.



#### 4.8 UTILIZAÇÃO DA PLAFORMA

Com todas as frentes de desenvolvimento da plataforma elucidadas individualmente, esta seção tem como objetivo exemplificar como se faz o uso de todas as ferramentas da plataforma em um cenário prático. Partindo do princípio que as placas do conversor de potência, instrumentação e o FPGA estão interconectados corretamente, o sistema pode ser energizado e conectado a rede de comunicação. Após o FPGA iniciar, este realiza uma procura em sua memória local (lendo o cartão CF através da interface SysAce) pelo arquivo *bitstream* que contém a configuração completa do dispositivo. Caso tais requisitos não sejam satisfeitos, o usuário deve utilizar a interface JTAG do FPGA para

realizar a configuração através de um computador com o software de programação do FPGA e o *bitstream* da plataforma.

Em um segundo momento, com a parte estática da plataforma gravada, o processador Microblaze inicia a execução do Software de Gerência Embarcado, fazendo a inicialização da plataforma e entrando no estado de funcionamento normal. Novamente, caso este programa não esteja disponível de maneira local ao sistema, deve-se conectar na interface MDM da Microblaze para fazer o carregamento e dar início ao software. É importante enfatizar que esse arquivo *bitstream* contendo a configuração completa, pode (opcionalmente) ter um hardware do usuário para controle do conversor, e depois de feita a inicialização do Software de Gerência Embarcado, o hardware é ativado realizando o controle do conversor.

Para interagir com a plataforma durante seu funcionamento, o usuário executa o Software de Monitoramento em um computador conectado a rede, digita o endereço IP designado à plataforma, podendo realizar a conexão. A partir desse ponto, é possível fazer o monitoramento e alteração das variáveis mapeadas em registradores da Microblaze. Caso o usuário necessite fazer a alteração da técnica utilizada de controle, após gerar o arquivo de configuração parcial usando o fluxo de desenvolvimento proposto, é necessário enviar o *bitstream* para o conversor via protocolo TFTP. Após o arquivo ser completamente transferido para a memória local da plataforma, pode-se fazer a seleção do mesmo, e enviar o comando de reconfiguração. Ao receber o comando a Microblaze interrompe a execução do controle atual para alterar a parte reconfigurável do FPGA, e após habilita novamente o hardware do usuário contendo o novo controle.

## 5 RESULTADOS EXPERIMENTAIS

Este capítulo apresenta os resultados experimentais obtidos durante o desenvolvimento da plataforma. Os módulos que compõem o sistema de controle do conversor são apresentados inicialmente com suas validações e testes individuais. Após são detalhados os passos para implementação do sistema completo, com demonstração dos resultados do controle do retificador boost.

### 5.1 SIMULAÇÃO COMPORTAMENTAL DO PERIFÉRICO DE CONTROLE DO CONVERSOR

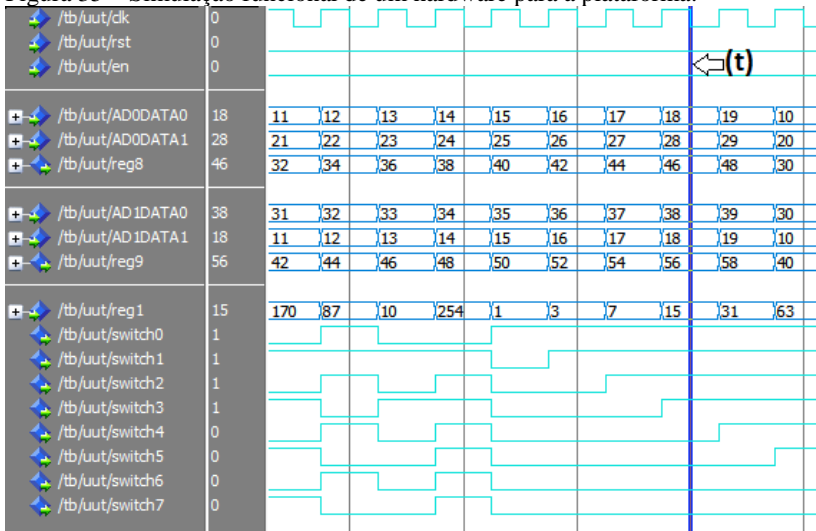
Com o intuito de facilitar os testes do hardware de controle para a plataforma, um arquivo de *testbench* é fornecido (no Apêndice B) para o usuário realizar os testes funcionais do sistema. Esse *script* é um código em VHDL o qual faz um encapsulamento do hardware do usuário, aplicando estímulos nas portas de entrada e fornecendo valores de saída. Para aplicar os estímulos de entrada, quando simulado, o *testbench* lê valores de um arquivo texto possibilitando assim simular as magnitudes resultantes das conversões dos ADs e visualizar os dados referentes à interface do banco de registradores.

Na Figura 35 são apresentadas as formas de onda resultantes de uma simulação com o *testbench*. O software utilizado, nesse caso, para realizar a simulação é o *ModelSim* [55] da empresa Mentor Graphics. Também é fornecido um script para automatizar o processo de simulação nessa ferramenta, tornando possível executar um comando e obter as formas de onda da simulação comportamental do novo hardware de controle desenvolvido pelo usuário. Entretanto, nada impede de utilizar outra ferramenta para simulação de HDL uma vez que o *testbench* não utiliza nenhuma biblioteca específica do *ModelSim*.

O hardware do usuário utilizado aqui realiza três operações básicas para demonstrar a funcionalidade da simulação. Na Figura 35, os registradores *reg8* e *reg9*, que são registradores de saída do hardware, armazenam o resultado da soma entre os valores do *AD0DATA0* e *AD0DATA1*, e entre *AD1DATA0* e *AD1DATA1*, respectivamente. As saídas do hardware – *switch0* até *switch8* – estão conectadas aos oito bits menos significativos do *reg1*, o qual é um registrador de entrada. Portanto é possível notar no momento indicado por (t) na Figura 35 que o resultado da soma entre os valores 18 e 28 aparece no registrador *reg8*. Analogamente é possível notar o resultado da soma dos ADs no registrador *reg9*. Observando as saídas do hardware, podemos notar que

nesse mesmo instante os sinais switch0 até switch3 estão em nível lógico alto, pelo fato do registrador de entrada reg1 ter o valor 15, sendo representado em binário como 00001111.

Figura 35 – Simulação funcional de um hardware para a plataforma.



## 5.2 PWM

O módulo responsável por fornecer os pulsos modulados que comandam o interruptor do conversor boost é denominado de PWM. Este módulo foi desenvolvido em VHDL para ser compatível com a implementação do controlador e possíveis implementações futuras de outros conversores. O PWM funciona através da comparação entre um contador – ou onda portadora – e um sinal de referência. Sempre que o sinal de referência for menor que o contador, o sinal de saída vai para nível lógico baixo, caso contrário, permanece em nível lógico alto. Nessa implementação foi utilizado um contador no modo de subida e descida, gerando assim uma forma de onda triangular. Outra opção seria contar até um valor e iniciar a contagem novamente, gerando uma forma de onda dente de serra.

A Figura 36 apresenta os sinais de entrada e saída do módulo PWM. Os sinais de *clk*, *rst* e *en* representam relógio, *reset* e habilitação, respectivamente. O sinal *suppress* tem a funcionalidade de suprimir o sinal de pulso modulado, sem deixar que o módulo pare de funcionar. Os sinais *ref*, *vmin* e *vmax* são os sinais que definem a largura do pulso e

a amplitude de contagem. Ou seja, o contador interno inicia com o valor de  $vmin$ , sendo incrementado até atingir o valor de  $vmax$  e após faz o decremento até o valor de  $vmin$ , repetindo o ciclo. É importante ressaltar que o valor de contagem é incrementado/decrementado a cada ciclo de relógio implicando na relação direta entre o valor de contagem e a frequência em que o módulo trabalha, por exemplo, caso o relógio seja 100MHz e se deseje um pulso com frequência de 50kHz, o contador tem uma excursão de 2000 unidades de contagem, dessa maneira, o valor de  $vmin$  pode ser zero e  $vmax$  ter 1000, levando em consideração o fato de ser uma portadora triangular.

Ainda nessa implementação, existem quatro modos de operação, controlados pelo sinal de entrada  $mode$ , utilizados para realizar a atualização do sinal de referência e gerar o sinal de saída  $done$ . A Tabela 2 resume os modos de operação, demonstrando o momento em que as funções são realizadas. Por fim, o sinal  $out_pwm$  é o resultado da comparação entre o valor de referência e o contador que gera o pulso modulado.

Figura 36 – Sinais de entrada e saída do módulo PWM.

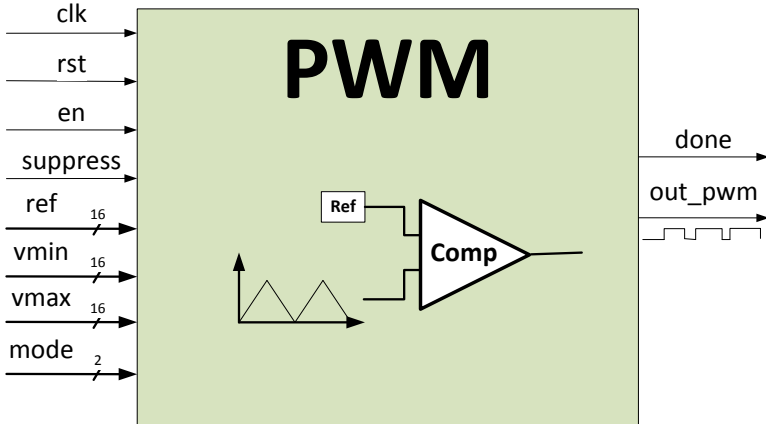


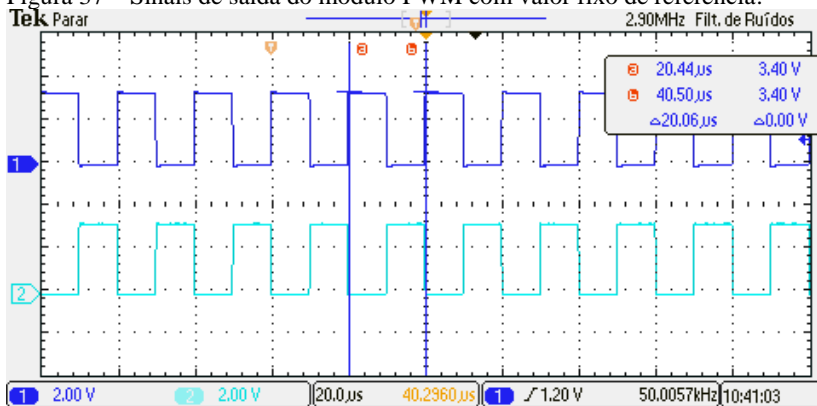
Tabela 2 – Modos de operação do módulo PWM.

Modo	Atualização	Done
00	Mínimo e Máximo	Mínimo e Máximo
01	Mínimo	Mínimo
10	Máximo	Máximo
11	Sempre	Nunca

Para realizar a validação, dois testes experimentais foram realizados com o módulo PWM desenvolvido. Foi desenvolvido uma

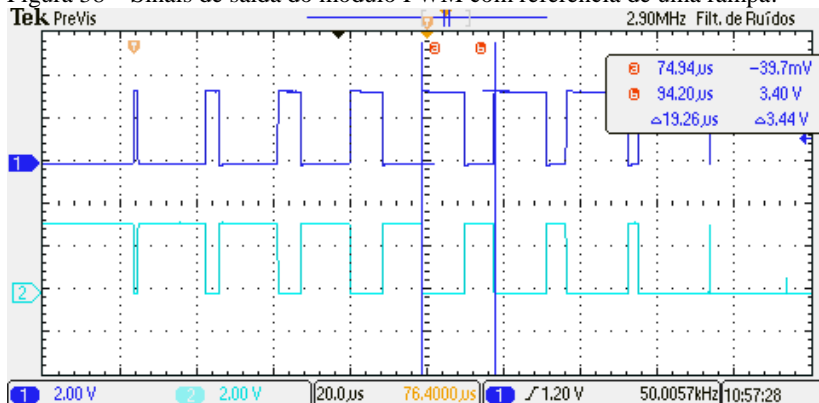
implementação para o FPGA alvo da plataforma, com valores de contagem entre zero e 1000 e se baseando no ciclo de relógio fornecido de 100MHz, gerando um pulso com frequência de 50kHz, ou 20 $\mu$ s. A saída do pulso modulado foi mapeada para um pino do FPGA e também o complemento do sinal (operação de negação) foi externado para validação. Os pinos mapeados do FPGA foram conectados a um osciloscópio para visualização das formas de onda. Na primeira experimentação, foi fixado o valor de referência na metade da contagem, gerando um a forma de onda quadrada com frequência fixa. Na Figura 37 as formas de onda de pulso modulado são apresentadas.

Figura 37 – Sinais de saída do módulo PWM com valor fixo de referência.



No segundo teste, a implementação foi modificada, adicionando um contador que incrementa seu valor a cada tempo pré-determinado, gerando uma rampa de contagem. Com isso a largura do pulso aumenta com o decorrer do tempo, e no caso do sinal complemento o inverso ocorre. A Figura 38 apresenta as formas de onda obtidas com essa implementação.

Figura 38 – Sinais de saída do módulo PWM com referência de uma rampa.



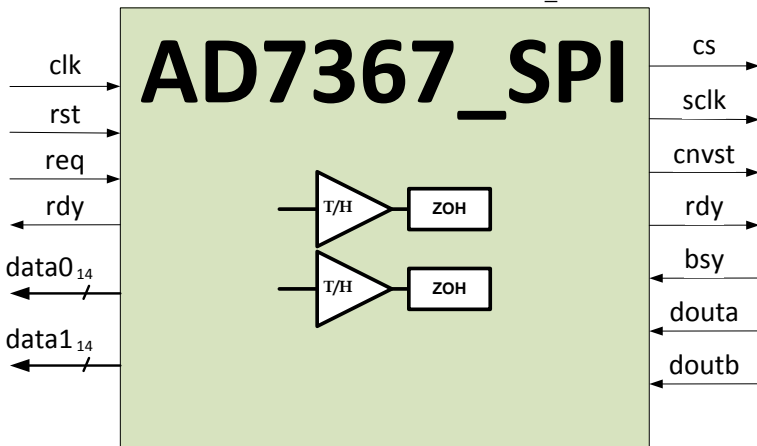
### 5.3 CONVERSORES AD

Para facilitar as aquisições dos sinais analógicos do ponto de vista do usuário da plataforma, foi desenvolvido um módulo em VHDL responsável por realizar a comunicação pelo protocolo SPI com o circuito integrado AD. Devido ao fato dos circuitos do conversor estático de potência e do sistema de instrumentação estarem fixados à topologia do conversor, esse módulo se encontra na parte estática do hardware da plataforma, isto é, não é possível modificar o módulo que faz a comunicação com os ADs. Porém, é importante ressaltar que, caso a plataforma seja utilizada em outro conversor e o sistema de instrumentação seja diferente, um novo módulo deve ser feito (ou adaptado) para suportar os novos dispositivos de AD.

Na Figura 39 os sinais de entrada e saída do módulo são apresentados. A esquerda da figura é possível ver a parte que se comunica com o hardware de controle interno, sendo os sinais *clk* e *rst* conectados diretamente ao relógio e reset da plataforma, e os sinais *req*, *rdy*, *data0* e *data1*, mapeados para o hardware de controle do usuário. Para realizar a comunicação com o módulo, o usuário deve, primeiramente, colocar o sinal *req* em nível lógico alto, ativando assim a máquina de estados do componente. Após o usuário deve monitorar o sinal *rdy*, pois quando este estiver em nível lógico alto, indica que a conversão da última requisição está disponível nos sinais *data0* e *data1*. Para realizar uma nova aquisição, o sinal *req* deve ser colocado em nível lógico baixo e após em nível alto, repetindo o processo.

A leitura dos valores dos circuitos de AD é feita pelo protocolo SPI. O barramento SPI é um protocolo serial síncrono que trabalha no modo mestre/escravo, sendo bastante utilizado na indústria. Os sinais apresentados à direita da Figura 39 são os pinos que devem ser conectados ao circuito na placa de instrumentação.

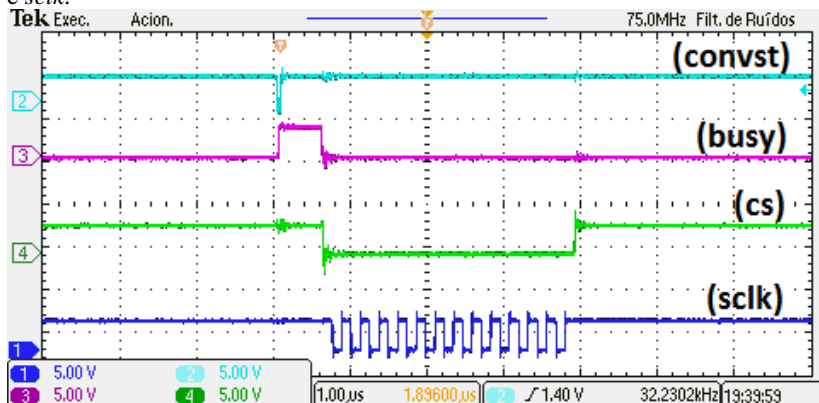
Figura 39 – Sinais de entrada e saída do módulo AD7367\_SPI.



Antes de realizar a leitura dos valores de conversão, é necessário fazer uma requisição para o circuito AD7367 através do pino *cnvst* (ativo baixo), o que causa o pino *bsy* ir para nível lógico alto, indicando que existe uma atividade de conversão em andamento (no caso do CI AD7367 especificamente, tem a duração aproximada de 620ns), após o pino *bsy* ir para nível lógico baixo é possível realizar a leitura do resultado da conversão pelo protocolo SPI. A Figura 40 apresenta uma captura, com auxílio de um osciloscópio, dos sinais fornecidos pelo componente AD7367\_SPI e o circuito integrado de AD no momento da requisição da conversão.

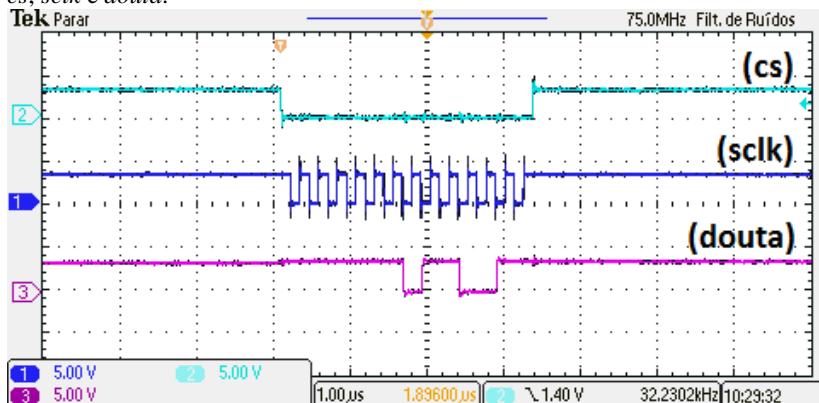


Figura 40 – Monitoramento da requisição de conversão – sinais *convst*, *busy*, *cs* e *sclk*.



Como o CI fornece os valores de conversão em um número representado por 14 bits, o módulo AD7367\_SPI faz a operação de deslocamento dos sinais de entrada *douta* e *doutb*, fornecendo os sinais paralelos nas saídas *data0* e *data1*, respectivamente. Na Figura 41 são apresentados os sinais *cs*, *sclk* e *douta* monitorados com um osciloscópio, sendo o valor lido em binário 1111110110011, que em complemento de dois representa o valor -77.

Figura 41 – Monitoramento da comunicação SPI com o conversor AD – sinais *cs*, *sclk* e *douta*.



#### 5.4 CONTROLADOR PROPORCIONAL INTEGRAL (PI)

Outro componente utilizado para o desenvolvimento da implementação do conversor boost é o bloco responsável pelo

controlador proporcional integral (PI). Este elemento tem a função de ajustar o processo de controle baseado em uma diferença entre um valor de referência e o valor medido do processo [56]. Para realizar tal controle, o componente faz uma operação discreta no tempo, de acordo com a seguinte fórmula:

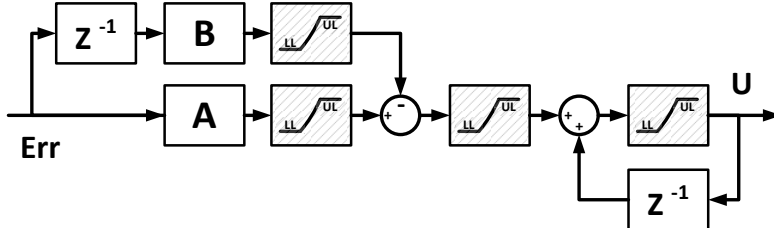
$$u[k] = u[k - 1] + A * err[k] - B * err[k - 1]$$

Onde:

- u – Saída do controlador.
- err – Sinal de erro (entrada do controlador).
- k – Índice do valor amostrado.
- A – Ganho proporcional.
- B – Ganho integral.

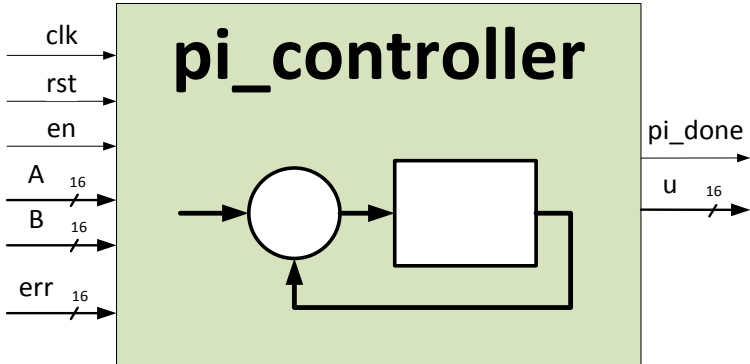
Sendo assim, a Figura 42 apresenta o diagrama de blocos da implementação discreta do controlador PI, onde os blocos indicados por  $Z^{-1}$  representam a operação de atraso unitário. Os blocos hachurados representam limites numéricos, fazendo a ação de *anti-windup* do controlador [48,56]. Os blocos A e B fazem a multiplicação pelos ganhos do controlador, e os blocos representados pelos círculos realizam as operações de soma e diferença.

Figura 42 – Diagrama de blocos do controlador PI.



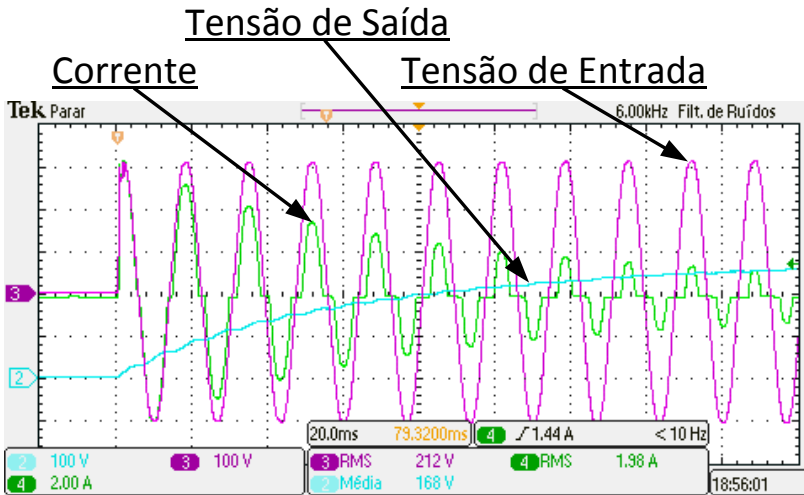
O componente descrito em VHDL possui a interface para utilização de acordo com a Figura 43, onde na parte à esquerda da figura os sinais *clk*, *rst* e *en* representam o sinal de relógio, reset e habilitação, respectivamente. Os sinais *A* e *B* de 16 bits de largura são utilizados para definir os ganhos proporcional e integral do controlador, o sinal *err*, também de 16 bits, é a entrada do sinal de erro do controlador. À direita da figura o sinal *pi\_done* é um bit que apresenta quando o resultado do sinal de controle está pronto para ser utilizado, sendo disponibilizado na saída do componente indicada por *u*.

Figura 43 – Sinais de entrada e saída do módulo PI.



## 5.5 EFEITO DE INRUSH

Para demonstrar o efeito de *inrush* (explicado na Seção 4.2) no momento em que o circuito é ligado, a captura com o osciloscópio das magnitudes da corrente e tensões de entrada e saída é apresentada na Figura 44. Como pode ser visto na figura, a corrente para uma tensão RMS de entrada de 210V chega a atingir um pico de 6A, mesmo com o resistor para limitar esse valor. Conforme os capacitores são carregados (nesse caso aproximadamente 100ms), e a tensão de saída aumenta, o valor da corrente se normaliza, sendo possível habilitar o relé de *bypass*, deixando o circuito pronto para ser controlado.

Figura 44 – Captura do efeito de *inrush*.

## 5.6 CONTROLE DO CONVERSOR BOOST

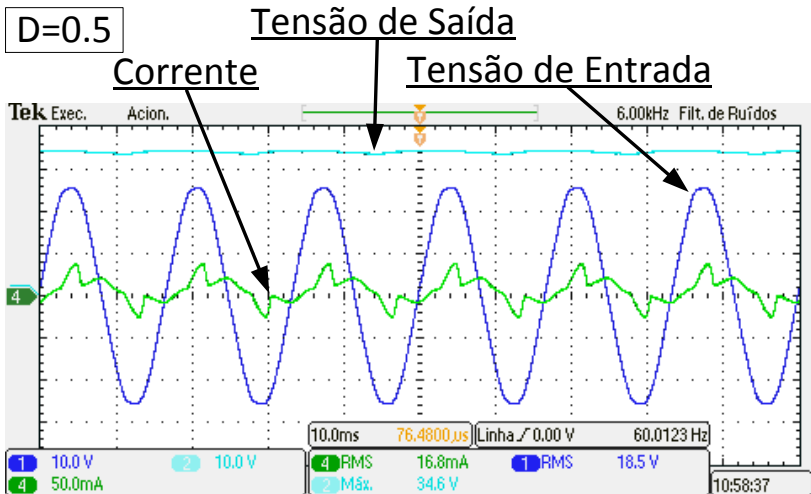
As seções que seguem apresentam os resultados experimentais obtidos com a implementação do controle do retificador boost. De acordo com a sequência desenvolvida na Seção 4.7, inicialmente são apresentados os resultados para o controle em malha aberta para em seguida serem mostrados os resultados com as implementações da malha de controle da corrente e controle da tensão de saída. É importante ressaltar que os controles aqui desenvolvidos foram inseridos como hardware de usuário na plataforma no intuito de realizar uma validação de maneira mais funcional do trabalho.

### 5.6.1 Malha Aberta (Elevador de Tensão)

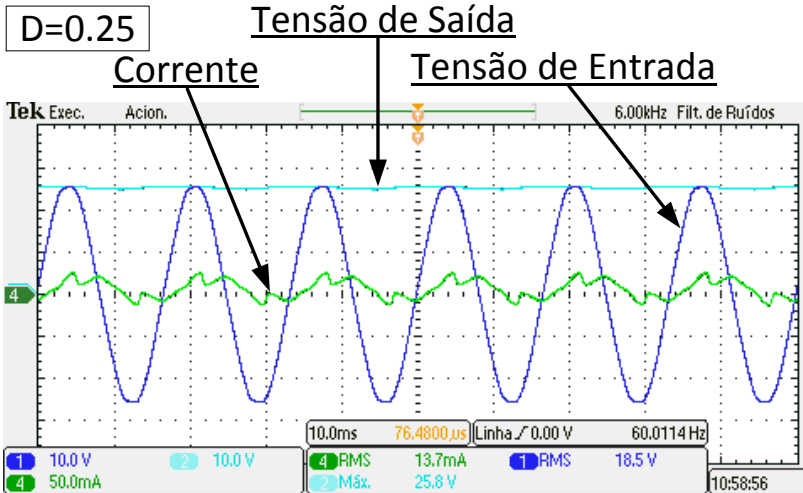
Com a operação em malha aberta, o controle do conversor boost pode realizar a elevação da tensão de saída em função do período de comutação da chave. De acordo com as equações apresentadas na Seção 4.7.1, para dobrar a tensão na saída é necessário aplicar uma razão cíclica de 50% na chave. Entretanto, no caso apresentado na Figura 45 onde a tensão de entrada (RMS) é 18,5V, a tensão de saída deve ficar na faixa dos 37V, porém devido as simplificações realizadas para o experimento, o valor obtido para essa comutação ficou em 34,6V, um

erro próximo a 7%, que pode ser também atribuído a imprecisões e perdas na aquisição experimental.

Figura 45 – Conversor boost em malha aberta ( $D=0.5$ ).



De maneira semelhante, aplicando uma razão cíclica na chave de 25%, a tensão de saída deve ficar 33% acima da tensão de entrada. A Figura 46 apresenta o experimento com essa comutação e uma tensão de entrada de 18,5V. Foi obtida uma tensão de saída de 25,8V – quando o esperado teórico deveria ser 24,66V – implicando em um erro de aproximadamente 5%, também bastante aceitável para tal simplificação.

Figura 46 – Conversor boost em malha aberta ( $D=0.25$ ).

### 5.6.2 Malha de Controle da Corrente

Para validar a funcionalidade da malha de controle da corrente, foi necessário analisar o formato das ondas de tensão de entrada e corrente. Quando estas estão em fase e tem o mesmo formato, o fator de potência é dito unitário ( $\cos(\phi)=1$ ), ou seja, do ponto de vista da fonte de entrada, a carga (incluindo o conversor) tem o valor de potência aparente igual ao da potência real sem nenhum componente na potência reativa, sendo este o objetivo desta implementação.

Para realizar uma comparação, a Figura 47 apresenta as formas de onda da tensão de entrada, corrente e tensão de saída quando o conversor boost não atua. Sendo possível notar a defasagem ( $\phi$ ) entre a senóide da tensão de entrada (referência) e a corrente. Em contraste a essa situação, na Figura 48 é possível visualizar o resultado experimental obtido, resultante da implementação do malha de controle da corrente (apresentado na Seção 4.7.2). Também nota-se a existência do ganho na magnitude de tensão de saída inerente do conversor boost.

Figura 47 – Formas de onda da tensão de entrada, corrente e tensão de saída sem atuação do conversor boost.

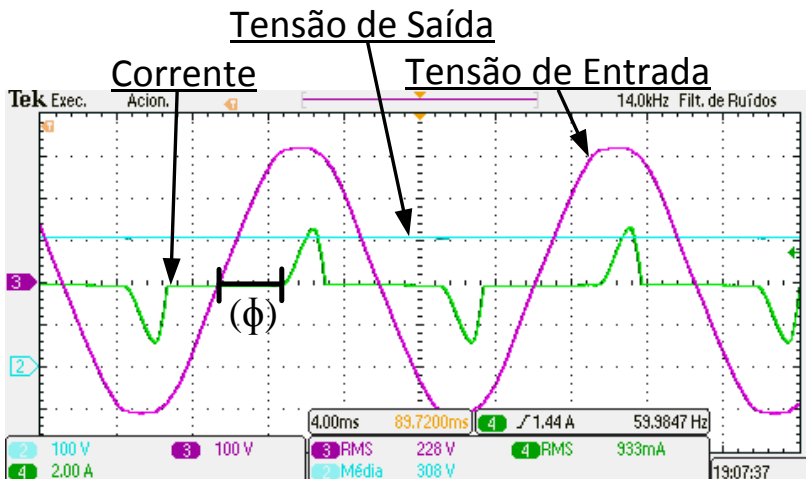
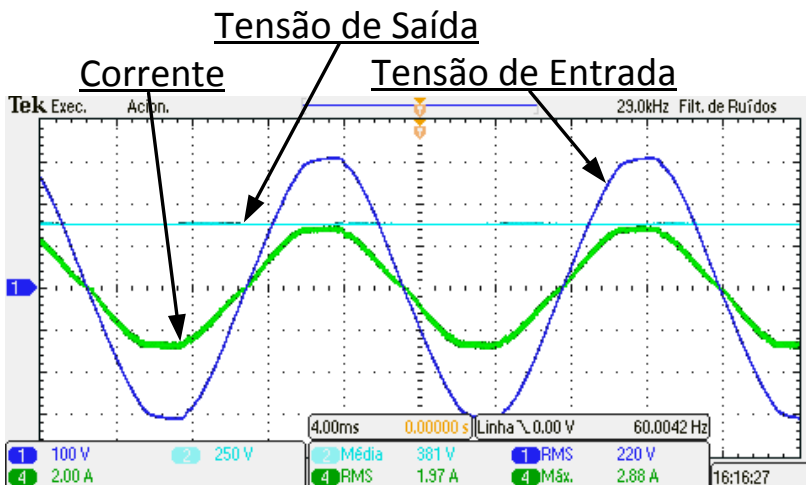


Figura 48 – Conversor boost com malha de controle da corrente.

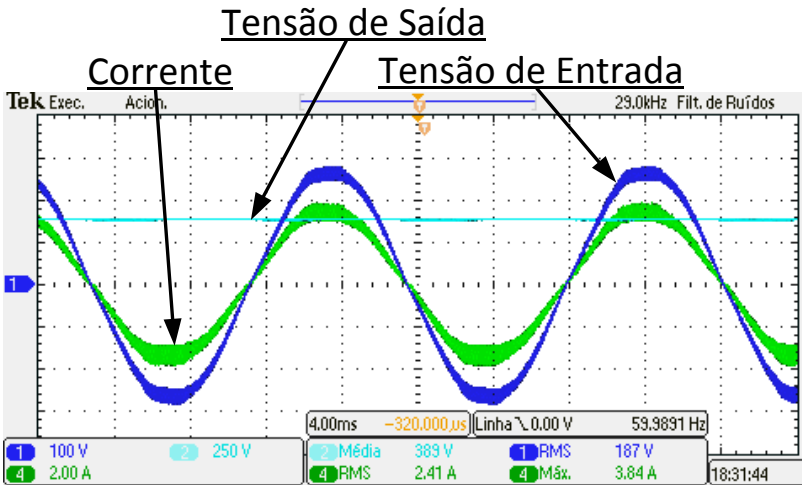


### 5.6.3 Malha de Controle da Tensão

Com a malha de controle da corrente atuando, foi possível implementar o controle da tensão de saída do conversor. E para visualizar a atuação do mesmo, dois cenários foram utilizados. Sendo

possível também, validar a funcionalidade de toda a técnica utilizada. Em um primeiro momento, foi observado a tensão de saída utilizando uma tensão de entrada RMS de 220V (Figura 48) e após, durante o funcionamento do conversor, a tensão de entrada reduzida em 15% (para 187V RMS). Na Figura 49 é apresentado a captura desse experimento, onde é possível notar que a tensão de saída continuou na faixa dos 380V, sendo medido 389V experimentais, diferença próxima a 3% que pode ser associado ao ruído gerado e imprecisões na aquisição experimental.

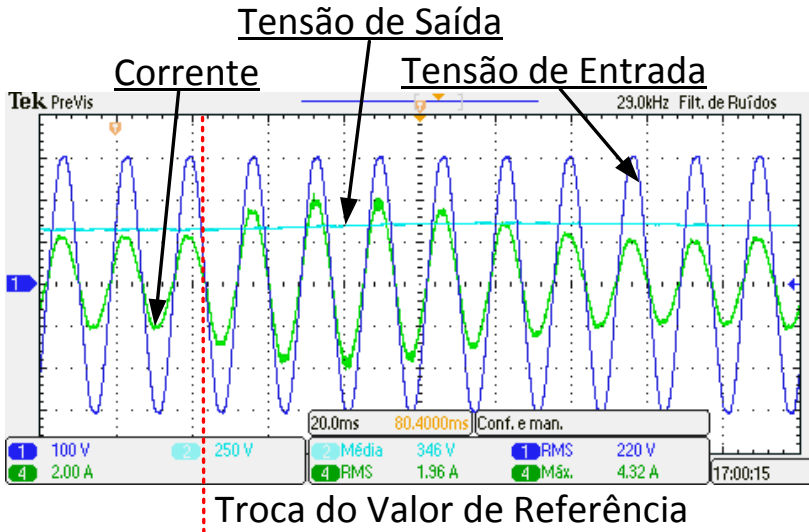
Figura 49 – Conversor boost com malha de controle da tensão, alteração na tensão de entrada.



No segundo teste de validação, foi alterado o valor de referência de tensão de saída através das ferramentas oferecidas pela plataforma de desenvolvimento. Duas abordagens para alteração do valor de referência foram utilizadas. Sendo a primeira alteração, a elevação da tensão de saída de 330V para 350V, e a captura das formas de onda exposta na Figura 50. É possível notar que após a troca do valor de referência do controle (representado pela linha pontilhada na figura), a corrente tem um aumento na amplitude para transferir a energia necessária para subir a tensão de saída, e após normalizada novamente quando o controle volta ao estado de regime.

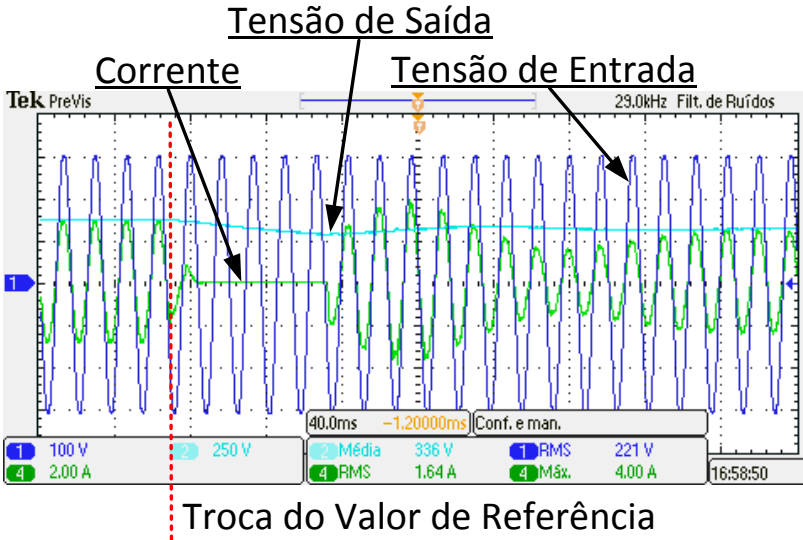


Figura 50 – Conversor boost com malha de controle da tensão, elevação do valor de referência da tensão de saída.



A segunda alteração do valor de referência de controle, foi a redução da tensão de saída desejada, de 350V para 330V. As formas de onda resultantes são exibidas na Figura 51. Nesta situação, o controle para de atuar (o interruptor boost fica aberto) por um pequeno intervalo de tempo (pouco menos que 80ms, após a linha pontilhada de indicação da alteração do valor de referência), fazendo com que a corrente seja nula nesse período, descarregando o indutor. No momento em que a tensão se aproxima do valor desejado, o controle volta a atuar, até que o ponto de operação se estabeleça, atingindo-se o estado de regime do controle do conversor.

Figura 51 – Conversor boost com malha de controle da tensão, redução do valor de referência da tensão de saída.



## 6 TRABALHOS RELACIONADOS

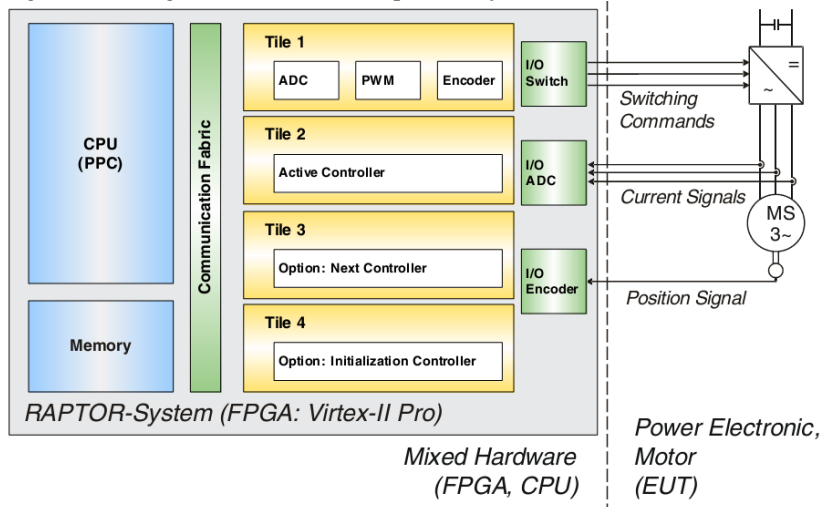
Neste capítulo são apresentados trabalhos relacionados à área de plataformas de desenvolvimento no campo do controle de conversores de eletrônica de potência. São discutidas as principais características de cada trabalho, as principais contribuições identificadas, e os pontos fortes e fracos, visando realizar uma comparação mais abrangente com a plataforma desenvolvida nessa dissertação.

### 6.1 FPGA-BASED REALIZATION OF SELF-OPTIMIZING DRIVE-CONTROLLERS

A referência [57] implementa um sistema auto otimizável, podendo ser composto de diversas possibilidades de configurações de hardware e software, para um motor de indução. Uma arquitetura SoC é apresentada, com a vantagem da troca *online* da técnica de controle, baseada entre FPGA e CPU tendo em vista a otimização da utilização dos recursos e qualidade do controle. Para realizar esta implementação foi utilizado uma placa de desenvolvimento, resultante de trabalho dos mesmos autores, equipada com um FPGA Virtex-II Pro da empresa Xilinx. Este FPGA possui uma arquitetura de suporte a reconfiguração parcial dinâmica baseada em quatro porções predeterminadas (denominadas *Tiles*), sem possibilidade de expansão devido ao limite físico da arquitetura do FPGA.

O estudo de caso deste trabalho utiliza um controlador de potência baseado na técnica de modulação vetorial para motor de indução. As porções predeterminadas do FPGA são utilizadas da seguinte forma: 1 – é utilizada para implementação comum do controlador que inclui o hardware do PWM digital, bloco de conversão analógico-digital e *encoder*; 2 – área para um controlador; 3 – segundo controlador (opcional) e 4 – controlador de inicialização (opcional). Na Figura 52 o diagrama de blocos do sistema utilizado é apresentado.

Figura 52 – Diagrama de blocos da implementação de [57].



Adaptado de [57].

Uma importante contribuição dos autores é o estudo e comparação entre os tempos de execução nos controles baseados em FPGA (hardware) e CPU (software). É possível notar que os tempos de execução na solução baseada em CPU impossibilitam a execução do algoritmo de controle. O trabalho valida em testes experimentais que é possível trocar a técnica de controle de um motor de indução, sem interrupção de funcionamento do equipamento eletromecânico.

Outra contribuição apresentada é o suporte de um fluxo da ferramenta utilizada com foco no desenvolvimento de controladores para FPGAs, tornando o desenvolvimento para engenheiros de outras áreas, que não a de hardware, uma tarefa mais factível e tolerante a erros.

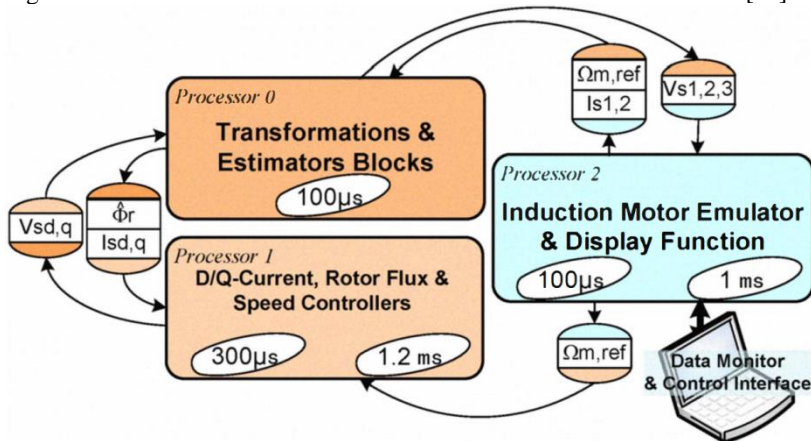
## 6.2 MPSOC DESIGN APPROACH OF FPGA BASED CONTROLLER FOR INDUCTION MOTOR DRIVE

Em [58], os autores desenvolveram uma plataforma SoC multi-processada (MPSoC, do inglês *Multi-Processor System-on-Chip*) com o objetivo de controlar um motor de indução. A plataforma é baseada em uma placa de desenvolvimento da empresa Xilinx, modelo ML310, que possui um FPGA Virtex II-Pro XC2VP30, com mais de 30.000 CLBs e dois processadores IBM PowerPC405 (PPC) físicos no FPGA. A placa

também oferece uma grande variedade de periféricos que auxiliam a interface com o mundo externo.

A abordagem de desenvolvimento proposta é realizar o controle do motor de indução através da técnica de comando vetorial. Para isso, a técnica foi analisada e particionada em três sub-rotinas, cada uma sendo executada em um processador do MPSoC, são elas: 1 – Blocos de transformadas e Conversões, 2 – Corrente D/Q, Fluxo do Rotor e Controlador de Velocidade, e 3 – Emulador do Motor de Indução e função de visualização. A Figura 53 apresenta os processadores do sistema e quais sub-rotinas cada um executa, juntamente das informações sobre o controle trocados entre eles.

Figura 53 – Particionamento da técnica de controle vetorial utilizado em [58].



Adaptado de [58].

Processadores Microblaze são utilizados no lugar dos processadores físicos presentes (PowerPC) pelo fato do trabalho utilizar mais de duas unidades para o processamento do algoritmo de controle. A principal contribuição do trabalho é a análise dos métodos de comunicação entre processadores, para o caso da Microblaze. São analisados os métodos de comunicação entre processos (IPC, do inglês *Inter-Process Communication*) através de barramento de memória compartilhada e conexão de memória multi-porta ponto-a-ponto (implementado usando BRAMs de duas portas). Outro método de IPC analisado é o de caixa de mensagens (do inglês *Mailbox*) através do barramento de comunicação da Microblaze denominado *Fast Simplex Link* (FSL).

Os autores utilizam uma solução híbrida dos métodos de comunicação estudados. Para evitar algumas das limitações de IPC as seguintes restrições foram estabelecidas: barramento compartilhado deve ser evitado, o FSL somente é utilizado quando as rotinas de software possuem a mesma taxa de troca de informações – evitando necessidade de sincronização dos dados. Assim a comunicação entre os processadores 0 e 2, que possuem o mesmo tempo de execução cíclica de 100 $\mu$ s, é feita utilizando o barramento FSL, e entre os processadores 0 e 1 é utilizado a técnica de memória compartilhada. Ressaltando que a troca de dados entre os processadores 1 e 2 utiliza o processador 0 como intermediário.

As conclusões exemplificadas neste trabalho são que a implementação de um sistema multi-processado com o objetivo de controle de potência para servo-motores e motores de indução é possível em um único FPGA. É demonstrado o fluxo de desenvolvimento para MPSoC bem como a escolha para cada plataforma, analisando e comparando os quesitos de diferentes arquiteturas. A técnica de controle vetorial é validada para tal arquitetura. Como trabalhos futuros para a plataforma, de acordo com os autores, ficam a adição e suporte de sensores, geração de pulsos PWM e emulação do estágio de potência.

### 6.3 DESIGN AND DEVELOPMENT OF A FLEXIBLE MULTI-PURPOSE CONTROLLER HARDWARE SYSTEM FOR POWER ELECTRONICS AND OTHER INDUSTRIAL APPLICATIONS

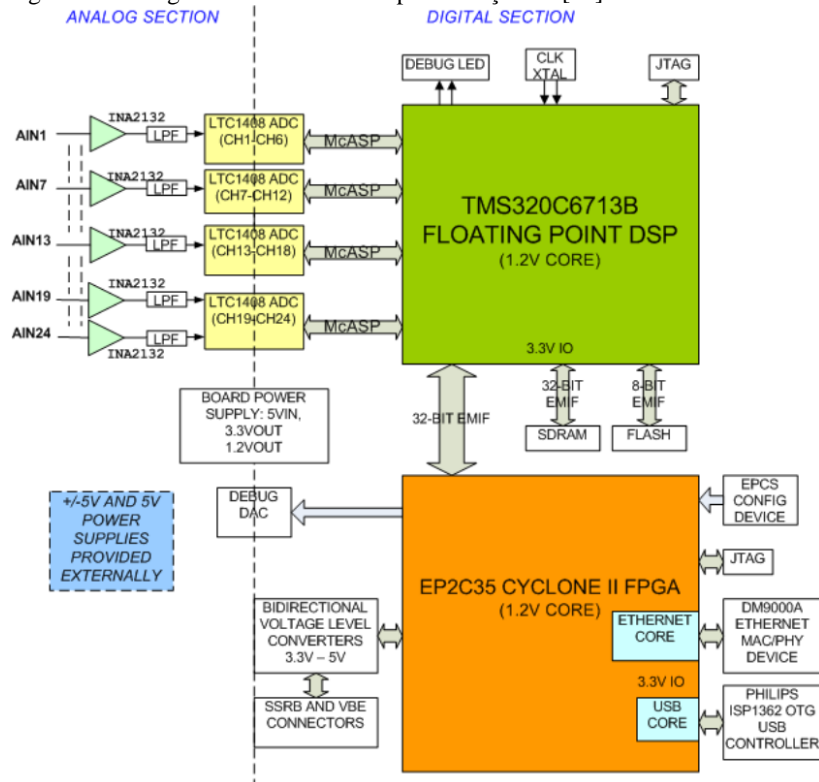
A referência [59] propõe o conceito de um controlador de hardware para múltiplos propósitos flexível para aplicações de eletrônica de potência. Com a presença de interfaces genéricas e esquemas de comunicação, o sistema pode ser usado em diversos cenários de controle de conversores de potência. Visando abranger tais cenários, uma plataforma híbrida entre FPGA e DSP foi construída para demonstrar o conceito de uma simulação de hardware de tempo real.

Os autores realizam um estudo dos requisitos de projeto para uma plataforma com foco no controle de eletrônica de potência, baseados em [60], sendo os pontos chaves: 1 – Processamento de sinais com ponto flutuante é essencial, e ponto fixo deve ser evitado a todo custo devido às perdas na precisão. 2 – A malha fechada de algoritmos de controle para tal fim requerem até 20.000 instruções para execução de um ciclo, e dado a amostragem de 20kHz, cerca de 40.000 instruções são executadas por ciclo. O período de amostragem dos ADs para que os sinais analógicos com uma frequência fundamental de 60Hz sejam

amostrados é de  $46\mu\text{s}$ , implicando na execução de um ciclo de controle nesse tempo. 3 – A habilidade de realizar a leitura de, no mínimo, 20 entradas analógicas. 4 – Resolução do sinal convertido de analógico para digital de 14 bits. 5 – Aproximadamente 128 sinais de E/S digitais, incluso os sinais para saídas de PWM. E 6 – Conectividade universal através de USB ou Ethernet.

O trabalho também realiza uma comparação entre outros sistemas acadêmicos similares. Dados os requisitos, a arquitetura de hardware utiliza um FPGA Cyclone II EP2C35 do fabricante Altera e o DSP TMS320C6713B do fabricante Texas Instruments. Para fazer a aquisição de dados externos analógicos, quatro ADCs modelo LTC1408 do fabricante Linear Technology são utilizados. O diagrama de blocos desta implementação é apresentado na Figura 54.

Figura 54 – Diagrama de blocos da implementação de [59].



Adaptado de [59].

A plataforma também possui conexão USB e Ethernet para desenvolvimento de uma interface homem-máquina. Como principais contribuições ficam o estudo, em detalhes, de como realizar a interação entre todos os blocos do sistema, desde a aquisição dos dados analógicos, condicionamento de sinal, comunicação com os entre ADC e DSP, comunicação entre DSP/FPGA e mapeamento de memória para implementação e troca de dados do controlador de potência implementado. Porém, somente a placa foi desenvolvida, nenhum resultado de simulação ou validação foi apresentado. Ficando como trabalhos futuros a implementação da interface homem-máquina via rede TCP/IP ou USB e adição de um *softprocessor* Nios-II da empresa Altera (equivalente ao *softprocessor* Microblaze).

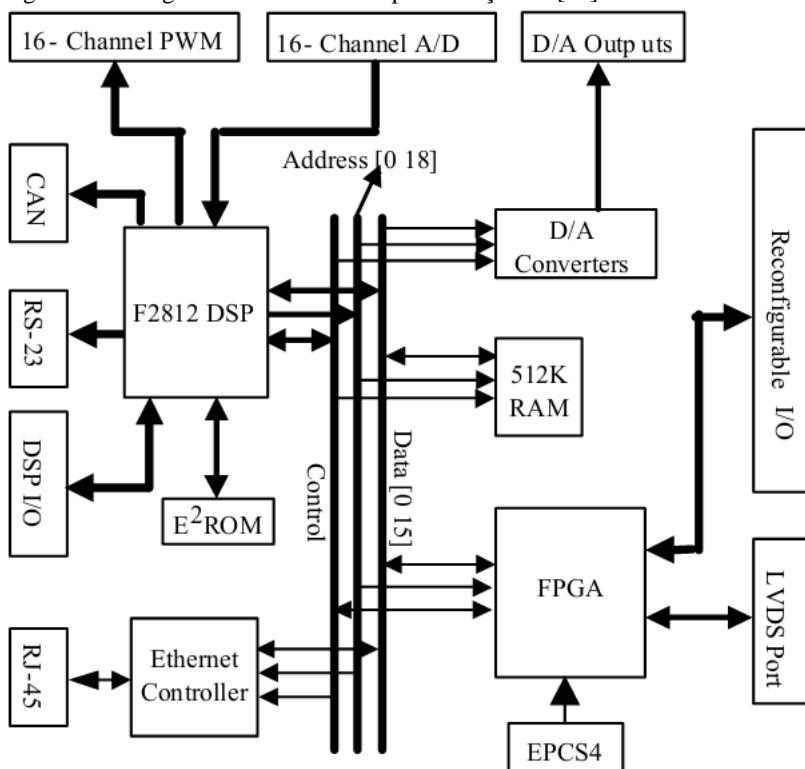
#### 6.4 A UNIVERSAL DIGITAL PLATFORM AND SOFTWARE LIBRARY FOR POWER ELECTRONIC SYSTEMS INTEGRATION

O trabalho em [61] apresenta uma plataforma digital com foco em eletrônica de potência, baseado em um DSP de ponto fixo. Incluso no desenvolvimento da plataforma, uma biblioteca de aplicação e integração de sistemas de eletrônica de potência foi implementada. Para obter uma plataforma com aplicabilidade ampla os requisitos de projeto definidos foram: 1 – Flexibilidade e versatilidade, 2 – Baixo custo, 3 – Alta capacidade computacional e 4 – Periféricos de comunicação em todos os níveis para integração do sistema.

A plataforma usa como DSP o CI TMS320F2812 do fabricante Texas Instruments e como FPGA, um Cyclone EP1C6 do fabricante Altera. Também estão presentes na placa desenvolvida interfaces de comunicação com rede (Ethernet e CAN), interface RS-232C para monitoramento e *debugging*, dois canais conversores Digital-Analógico (DA) de 12 bits, pinos de uso geral configuráveis e elementos de memória para armazenamento do programa e configuração o FPGA. O diagrama de blocos do sistema desenvolvido pode ser visto na Figura 55.



Figura 55 – Diagrama de blocos da implementação de [61].



Adaptado de [61].

Para melhorar o poder computacional do DSP de ponto fixo, unidades de ponto flutuante são implementadas no FPGA, e disponibilizadas para o usuário através de núcleos de propriedade intelectual (do inglês, *Intellectual Property* – IP). Três IPs são desenvolvidos para os cálculos, sendo eles, multiplicação, soma e divisão. Ainda no FPGA é disponibilizado um componente de PWM. Para desenvolvimento no DSP, bibliotecas em software de SVPWM de dois e três níveis, módulos de transformações (como Park e Clarke) para controle de potência, tabelas *lookup* – para funções como seno e cosseno – e módulo de PID são disponibilizados. Os módulos de FPGA são validados através de uma comparação entre a implementação de uma transformada de Fourier em software para DSP e utilizando o coprocessamento do FPGA, chegando a um tempo de execução cinco vezes menor quando o coprocessamento foi utilizado.

Embora seja uma plataforma para eletrônica de potência, nenhum resultado de implementação de controle foi apresentado, porém, os autores afirmam que os IPs e bibliotecas são utilizados em outros projetos para tal fim. Como contribuições identificadas, pode-se citar o desenvolvimento dos componentes padronizados de forma acessível ao usuário, bem como as bibliotecas implementadas para o DSP.

## 6.5 POSICIONAMENTO E COMPARAÇÃO DOS TRABALHOS RELACIONADOS

Para ser possível realizar a comparação entre os trabalhos relacionados e a plataforma aqui apresentada, as implementações são analisadas – nas seções que seguem – de acordo com quatro itens, onde foi possível fazer uma intersecção dos objetivos entre os mesmos.

### 6.5.1 Uso da Plataforma de Maneira Genérica para Controle de Conversores de Potência

Com um dos requisitos de projeto focado na utilização da plataforma de maneira genérica, as especificações da implementação são restringidas de tal maneira que as interfaces e componentes utilizados devem ser capazes de cobrir necessidades maiores que um controle de um conversor específico (como controle de motores), e que em grande parte das vezes causa um superdimensionamento dos equipamentos da plataforma.

No trabalho [59] esses requisitos foram analisados de maneira minuciosa, resultando em diversos conversores AD de alta taxa de amostragem e também em uma preocupação com interferências provenientes do ambiente ruidoso de potência, visto de um ponto de comunicação digital. Em [61] foram desenvolvidos blocos genéricos para diversas implementações de conversores, bem como uma placa para a plataforma com diversas interfaces para tal. Em ambos os trabalhos, a abordagem para processamento e cálculo do controle foi feito de maneira híbrida entre um FPGA e um DSP, aumentando a quantidade de componentes na placa, bem como necessidades de interfaces e softwares para programação de cada um deles.

Na implementação aqui apresentada, o uso de maneira genérica da plataforma é feito pelo usuário, existe a possibilidade de modificar a plataforma de acordo com os requisitos das placas de instrumentação e potência utilizadas. Pode-se afirmar que a plataforma é desenvolvida com o foco no âmbito digital, sendo necessário um casamento com as

demais interfaces, mas o uso do controle do conversor pode ser genérico para qualquer topologia de conversor. Os demais trabalhos não tem foco de implementação do controle de conversores variados, mas é possível especular que a implementação apresentada em [57] pode ser modificada para aceitar novas interfaces de potência, e em [58] não existe informação suficiente no trabalho apresentado para ser possível realizar tal comparação.

### **6.5.2 Implementação do Controle do Conversor em Hardware ou Software**

Outro ponto da vista do controle do conversor é a maneira em que os cálculos e a malha de controle são implementados, que para este item de avaliação é entendido da seguinte maneira: Hardware – uso de uma HDL para transcrição em lógica configurável, e Software – uso de um software procedural executado em um processador (de uso geral ou DSP). Os trabalhos [59] e [61] utilizam uma abordagem híbrida, ou seja, dependem de ambas implementações, porém sendo o DSP o principal componente responsável pelo controle, e o FPGA executa funções como um periférico (ou extensão de funcionalidade) do DSP. Em [58] é apresentada uma solução que utiliza somente software, porém particionada em mais de um processador, com o intuito de evitar problemas com restrições temporais do controle.

O trabalho desenvolvido em [57] oferece a possibilidade de realizar o controle somente em hardware ou somente software, sendo demonstradas as vantagens provenientes da implementação em hardware. Já a implementação dessa dissertação oferece somente a opção de implementação do controle em hardware. O processador utilizado tem a função de gerenciamento e monitoria do sistema. Pode-se concluir que a implementação mais vantajosa nesse item de avaliação é dada por [57] devido à possibilidade de escolha de acordo com a preferência do usuário.

### **6.5.3 Fluxo de Ferramentas de Desenvolvimento para Uso da Plataforma**

Um fator importante a ser considerado ao se propor uma plataforma é o suporte apropriado das ferramentas e fluxo de desenvolvimento. Esse item não foi possível identificar nos trabalhos apresentados em [59] e [61], fato bastante curioso por serem apontadas como plataformas genéricas de controle de conversores. A

implementação [58] é bastante focada na análise do controle para motores de indução, porém a reprodução do fluxo de desenvolvimento utilizada não foi apresentada.

[57] apresenta, em detalhes, o processo para uso da plataforma, desde a entrada da descrição de hardware até os arquivos *bitstream* que são gerados pela ferramenta, sendo possível reproduzir todo o processo. De maneira semelhante, na presente implementação, os passos necessários para utilizar a plataforma são apresentados em detalhes. Sendo estes apresentados como as escolhas mais apropriadas nesse quesito.

#### **6.5.4 Reconfiguração Parcial**

Por fim, um item que apresenta um diferencial no uso de controle de conversores é a possibilidade de atualização da técnica utilizada de controle sem realizar a parada de todo o conversor. Nos trabalhos [58], [59] e [61], por serem soluções baseadas em software procedural, o procedimento para mudança da técnica de controle é feito com a atualização do código executado pelo processador, que por consequência implica em desligar o conversor e fazer a reprogramação da memória de programa.

O trabalho apresentado em [57] oferece a possibilidade de atualização de uma parte do hardware do FPGA com a nova técnica de controle, sem afetar o funcionamento do conversor, e após feita a reconfiguração dessa parte, ambas técnicas de controle existentes nas áreas reconfiguráveis podem ser alternadas para realizar a troca em tempo real. A implementação apresentada nesse trabalho oferece a possibilidade de reconfiguração da área que contém o hardware de controle. Porém, devido ao fato de existir somente uma área para o hardware do usuário, o conversor deve ser parado por alguns segundos para haver a reconfiguração parcial do FPGA. A possibilidade de oferecer duas áreas para hardware de controle (uma ativa e uma para troca) é deixada como um possível trabalho futuro.

#### **6.5.5 Tabela Comparativa**

Depois de descritos os itens avaliados entre os trabalhos relacionados, a Tabela 3 tem como objetivo sintetizar a comparação entre os itens e os trabalhos apresentados. Os itens analisados estão nas linhas representados pela seção desse trabalho, e os trabalhos utilizados nas colunas da tabela utilizando o índice da referência bibliográfica.

Tabela 3 – Comparação entre os trabalhos relacionados e a plataforma proposta.

	[57]	[58]	[59]	[61]	<b>Presente Trabalho</b>
<b>6.5.1</b>	Não	Não	Sim	Sim	Sim
<b>6.5.2</b>	Hw ou Sw	Sw	Hw+Sw	Hw+Sw	Hw
<b>6.5.3</b>	Sim	Não	Não	Não	Sim
<b>6.5.4</b>	Sim	Não	Não	Não	Sim



## 7 CONCLUSÕES

Este trabalho apresentou o estudo e desenvolvimento de uma plataforma de desenvolvimento utilizando tecnologia reconfigurável para conversores de potência. O desenvolvimento teve como foco a facilidade de implementação do controle de conversores para engenheiros da área de eletrônica de potência e a possibilidade da inserção, de maneira simplificada, do conversor em uma rede de comunicação inteligente. Inicialmente, o trabalho apresentou a consolidação do uso da abordagem digital e FPGAs no controle de conversores de energia e eletrônica de potência. Foi demonstrada a tendência e necessidade de uma nova abordagem na geração, transmissão, distribuição e consumo de energia elétrica, sendo neste contexto posicionada a inserção da plataforma desenvolvida. Após foram abrangidas as tecnologias necessárias para o entendimento da implementação da plataforma, a qual foi detalhada e validada experimentalmente, e demonstrada sua aplicabilidade em um conversor de potência de C.A. para C.C, denominado conversor boost.

Durante o desenvolvimento do trabalho, foram disponibilizados e documentados módulos em descrição de hardware que podem ser utilizados em futuras implementações de conversores. O procedimento para uso e replicação da plataforma também é detalhado nesta dissertação, sendo uma das principais contribuições a possibilidade de uso em aplicações de conversão de energia por outros grupos de pesquisa e desenvolvimento. O software embarcado e de monitoramento para a plataforma foram desenvolvidos de acordo com os objetivos dessa dissertação, e também fornecidos para utilização e possível melhoria de funcionalidade.

Com a plataforma descrita, foi originada também uma comparação com outros trabalhos acadêmicos publicados que utilizam o controle de conversores de potência, com objetivo semelhante. Entretanto, com exceção dessa dissertação, nenhum dos trabalhos apresentou a possibilidade do uso em redes inteligentes, sendo esse mais um diferencial oferecido pelo sistema.

Como trabalhos futuros provenientes dessa dissertação pode-se citar o aperfeiçoamento do software de gerência embarcado, na forma de melhorias na implementação do sistema de tempo real, utilização de protocolos padrão para troca de informação em redes inteligentes (padronização a qual o Brasil carece até o presente momento), e inclusão de técnicas de segurança da informação. A possibilidade de adicionar uma segunda área reconfigurável na plataforma possui bastante utilidade

em conversores de energia (como o uso em controle de motores de indução), podendo assim, as implementações de controle do usuário, serem multiplexadas sem a interrupção do funcionamento do conversor, processo semelhante ao que é feito em [57].

A plataforma pode tirar proveito das novas arquiteturas de SoCs disponibilizados pelo fabricante Xilinx, sendo mais um trabalho futuro possível. A Figura 56 apresenta uma comparação entre a plataforma desenvolvida e uma mesma abordagem utilizando os recursos do SoC Zynq-7000 [17]. Este componente apresenta a integração entre um processador ARM Cortex A9 de dois núcleos, onde cada um deles pode executar um software em paralelo. A implementação da descrição de hardware do usuário pode ser implementada na área de lógica programável da mesma maneira que é feita em um FPGA, e a comunicação entre os blocos é realizada através de uma interface pré-definida.

Figura 56 – Comparação entre as abordagens utilizadas para a implementação da plataforma.

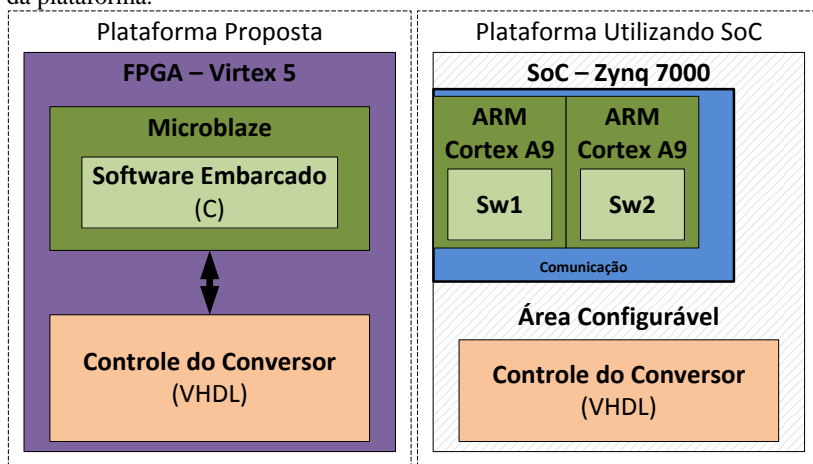


Figura do autor.

A implementação dessa plataforma poderá também ser utilizada para a validação do trabalho desenvolvido em [23], com a finalidade de obter resultados experimentais da técnica de controle do conversor proposta, uma vez que o artigo foi apresentado com simulações numéricas.

Outro item o qual não foi explorado nesse trabalho é a possibilidade do uso da plataforma em atividade de ensino para eletrônica de potência. Esta área possui grande aceitação na comunidade



acadêmica e diversos trabalhos estão relacionados [62–65]. Devido ao fato deste trabalho disponibilizar uma interface para a rede no padrão Ethernet, possibilidade de troca do hardware de controle, e suporte as ferramentas de desenvolvimento, poderia ser estudada a possibilidade de adaptar/focar o desenvolvimento em plataformas de ensino.



## REFERÊNCIAS

- [1] DALY, H. E. Toward some operational principles of sustainable development. *Ecological Economics*, v. 2, n. 1, p. 1-6, 1990.
- [2] FARHANGI, H. The path of the smart grid. *IEEE Power and Energy Magazine*, v. 8, n. 1, p. 18-28, doi:10.1109/MPE.2009.934876, 2010.
- [3] AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA (ANEEL). *RESOLUÇÃO NORMATIVA Nº 502, DE 7 DE AGOSTO DE 2012*. Disponível em: <<http://www.aneel.gov.br/cedoc/ren2012502.pdf>>. Acesso em: 30 jan. 2013.
- [4] HOROWITZ, S. H. e PHADKE, A. G. Boosting immunity to blackouts. *IEEE Power and Energy Magazine*, v. 1, n. 5, p. 47-53, doi:10.1109/MPAE.2003.1231691, 2003.
- [5] MUSSA, S. A. *Controle de um conversor CA-CC trifásico PWM de três níveis com fator de potência unitário utilizando DSP*. Tese (Doutorado) - Universidade Federal de Santa Catarina, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, SC, Brasil. 2003.
- [6] ALVES, A. *Estudo, projeto e implementação de unidades retificadoras de -48V/10A para telecomunicações utilizando circuitos de supervisão microcontrolados*. Dissertação (Mestrado) - Universidade Federal de Santa Catarina, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, SC, Brasil. 2002.
- [7] SUN, B. e GAO, Z. A DSP-Based Active Disturbance Rejection Control Design for a 1-kW H-Bridge DC–DC Power Converter. *IEEE Transactions on Industrial Electronics*, v. 52, n. 5, p. 1271-1277, doi:10.1109/TIE.2005.855679, 2005.
- [8] TEODORESCU, R. e BLAABJERG, F. Flexible Control of Small Wind Turbines With Grid Failure Detection Operating in Stand-Alone and Grid-Connected Mode. *IEEE Transactions on Power Electronics*, v. 19, n. 5, p. 1323-1332, doi:10.1109/TPEL.2004.833452, 2004.
- [9] HUA, C.; LIN, J. e SHEN, C. Implementation of a DSP-controlled photovoltaic system with peak power tracking. *IEEE Transactions on Industrial Electronics*, v. 45, n. 1, p. 99-107, doi:10.1109/41.661310, 1998.
- [10] ORTMANN, M. S. *Filtro Ativo Trifásico com Controle Vetorial Utilizando DSP: Projeto e Implementação*. Dissertação (Mestrado) - Universidade Federal de Santa Catarina, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, SC, Brasil. 2008.
- [11] MONMASSON, E. e CIRSTEA, M. N. FPGA Design Methodology for Industrial Control Systems—A Review. *IEEE*

- Transactions on Industrial Electronics*, v. 54, n. 4, p. 1824-1842, doi:10.1109/TIE.2007.898281, 2007.
- [12] MONMASSON, E.; IDKHAJINE, L. e NAOUAR, M. FPGA-based Controllers. *IEEE Industrial Electronics Magazine*, v. 5, n. 1, p. 14-26, doi:10.1109/MIE.2011.940250, 2011.
- [13] RODRIGUEZ-ANDINA, J. J.; MOURE, M. J. e VALDES, M. D. Features, Design Tools, and Application Domains of FPGAs. *IEEE Transactions on Industrial Electronics*, v. 54, n. 4, p. 1810-1823, doi:10.1109/TIE.2007.898279, 2007.
- [14] BEZERRA, E. . e GOUGH, M. . A guide to migrating from microprocessor to FPGA coping with the support tool limitations. *Microprocessors and Microsystems*, v. 23, n. 10, p. 561-572, doi:10.1016/S0141-9331(99)00063-0, 2000.
- [15] BROWN, S. FPGA architectural research: a survey. *IEEE Design & Test of Computers*, v. 13, n. 4, p. 9-15, doi:10.1109/54.544531, 1996.
- [16] BROWN, S. e ROSE, J. FPGA and CPLD architectures: a tutorial. *IEEE Design & Test of Computers*, v. 13, n. 2, p. 42-57, doi:10.1109/54.500200, 1996.
- [17] XILINX INC. *Zynq-7000 All Programmable SoC Overview*. Disponível em: <[http://www.xilinx.com/support/documentation/data\\_sheets/ds190-Zynq-7000-Overview.pdf](http://www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf)>. Acesso em: 30 jan. 2013.
- [18] XILINX INC. *Partial Reconfiguration of Xilinx FPGAs Using ISE Design Suite*. Disponível em: <[http://www.xilinx.com/support/documentation/white\\_papers/wp374\\_Partial\\_Reconfig\\_Xilinx\\_FPGAs.pdf](http://www.xilinx.com/support/documentation/white_papers/wp374_Partial_Reconfig_Xilinx_FPGAs.pdf)>. Acesso em: 30 jan. 2013.
- [19] XILINX INC. *Fast Configuration of PCI Express Technology through Partial Reconfiguration*. Disponível em: <[http://www.xilinx.com/support/documentation/application\\_notes/xapp883\\_Fast\\_Config\\_PCIE.pdf](http://www.xilinx.com/support/documentation/application_notes/xapp883_Fast_Config_PCIE.pdf)>. Acesso em: 30 jan. 2013.
- [20] PCI-SIG. *PCI Express Base Specification*. Disponível em: <<http://www.pcisig.com/specifications/pciexpress/>>. Acesso em: 30 jan. 2013.
- [21] MOHAN, N.; UNDELAND, T. M. e ROBBINS, W. P. *Power Electronics: Converters, Applications, and Design*. [S.l.]: John Wiley & Sons, 2002. p. 824
- [22] LUO, F. L.; YE, H. e RASHID, M. H. *Digital Power Electronics and Applications*. [S.l.]: Academic Press, 2005. p. 464
- [23] JAPPE, T. K. e MUSSA, S. A. Current technique applied in single phase PFC boost converter based on discrete-time One Cycle Control.

In: 2011 IEEE 33RD INTERNATIONAL TELECOMMUNICATIONS ENERGY CONFERENCE (INTELEC). *Anais...* [S.l.]: IEEE, 2011.

[24] MASSOUD AMIN, S. e WOLLENBERG, B. F. Toward a smart grid: power delivery for the 21st century. *IEEE Power and Energy Magazine*, v. 3, n. 5, p. 34-41, doi:10.1109/MPAE.2005.1507024, 2005.

[25] LASSETER, R. H. e PAIGI, P. Microgrid: a conceptual solution. In: 2004 IEEE 35TH ANNUAL POWER ELECTRONICS SPECIALISTS CONFERENCE (IEEE CAT. NO.04CH37551). *Anais...* [S.l.]: IEEE, 2004.

[26] IEEE. *IEEE & Smart Grid*. Disponível em:

<<http://smartgrid.ieee.org/ieee-smart-grid>>. Acesso em: 30 jan. 2013.

[27] ETP. *Smart Grids European Technology Platform*. Disponível em: <<http://www.smartgrids.eu/>>. Acesso em: 30 jan. 2013.

[28] GAO, J. et al. A survey of communication/networking in Smart Grids. *Future Generation Computer Systems*, v. 28, n. 2, p. 391-404, doi:10.1016/j.future.2011.04.014, 2012.

[29] IEEE. *Smart Grid Conceptual Model*. Disponível em:

<<http://smartgrid.ieee.org/ieee-smart-grid/smart-grid-conceptual-model>>. Acesso em: 30 jan. 2013.

[30] HELDWEIN, M. L. *Microredes em corrente contínua: qualidade de fornecimento eficiência em futuras redes de distribuição*.

Universidade Federal de Santa Catarina, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, SC, Brasil - [S.l.]. 2010.

[31] XILINX INC. *XUPV5-LX110T Development System*. Disponível em: <<http://www.xilinx.com/univ/xupv5-lx110t.htm>>. Acesso em: 30 jan. 2013.

[32] XILINX INC. *Embedded System Tools Reference Manual (EDK 13.2)*. Disponível em:

<[http://www.xilinx.com/support/documentation/sw\\_manuals/xilinx13\\_2/est\\_rm.pdf](http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_2/est_rm.pdf)>. Acesso em: 30 jan. 2013.

[33] XILINX INC. *LogiCORE IP MicroBlaze Micro Controller System*. Disponível em:

<[http://www.xilinx.com/support/documentation/sw\\_manuals/xilinx14\\_1/ds865\\_microblaze\\_mcs.pdf](http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_1/ds865_microblaze_mcs.pdf)>. Acesso em: 30 jan. 2013.

[34] XILINX INC. *LogiCORE IP Processor Local Bus (PLB) v4.6*. Disponível em:

<[http://www.xilinx.com/support/documentation/ip\\_documentation/plb\\_v46.pdf](http://www.xilinx.com/support/documentation/ip_documentation/plb_v46.pdf)>. Acesso em: 30 jan. 2013.

[35] XILINX INC. *MicroBlaze Debug Module (MDM)*. Disponível em:

<[http://www.xilinx.com/support/documentation/ip\\_documentation/mdm.pdf](http://www.xilinx.com/support/documentation/ip_documentation/mdm.pdf)>. Acesso em: 30 jan. 2013.

- [36] XILINX INC. *LogiCORE IP XPS HWICAP*. Disponível em: <[http://www.xilinx.com/support/documentation/ip\\_documentation/xps\\_hwicap.pdf](http://www.xilinx.com/support/documentation/ip_documentation/xps_hwicap.pdf)>. Acesso em: 30 jan. 2013.
- [37] XILINX INC. *XPS SYSACE (System ACE) Interface Controller*. Disponível em: <[http://www.xilinx.com/support/documentation/ip\\_documentation/xps\\_sysace.pdf](http://www.xilinx.com/support/documentation/ip_documentation/xps_sysace.pdf)>. Acesso em: 30 jan. 2013.
- [38] XILINX INC. *LogiCORE IP XPS LL TEMAC*. Disponível em: <[http://www.xilinx.com/support/documentation/ip\\_documentation/xps\\_1\\_1\\_temac.pdf](http://www.xilinx.com/support/documentation/ip_documentation/xps_1_1_temac.pdf)>. Acesso em: 30 jan. 2013.
- [39] XILINX INC. *XPS UART Lite*. Disponível em: <[http://www.xilinx.com/support/documentation/ip\\_documentation/xps\\_uartlite.pdf](http://www.xilinx.com/support/documentation/ip_documentation/xps_uartlite.pdf)>. Acesso em: 30 jan. 2013.
- [40] XILINX INC. *LogiCORE IP Multi-Port Memory Controller (MPMC)*. Disponível em: <[http://www.xilinx.com/support/documentation/ip\\_documentation/mpmc.pdf](http://www.xilinx.com/support/documentation/ip_documentation/mpmc.pdf)>. Acesso em: 30 jan. 2013.
- [41] XILINX INC. *Using EDK to Run Xilkernel on a MicroBlaze Processor*. Disponível em: <[http://www.xilinx.com/support/documentation/sw\\_manuals/xilinx13\\_2/ug758.pdf](http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_2/ug758.pdf)>. Acesso em: 30 jan. 2013.
- [42] IEEE. *POSIX.1-2008*. Disponível em: <<http://pubs.opengroup.org/onlinepubs/9699919799/>>. Acesso em: 30 jan. 2013.
- [43] SOLLINS, K. *IETF - RFC1350 - The TFTP Protocol (Revision 2)*. Disponível em: <<http://tools.ietf.org/html/rfc1350>>. Acesso em: 30 jan. 2013.
- [44] DIGIA. *Qt Documentation*. Disponível em: <<http://qt-project.org/doc/>>. Acesso em: 30 jan. 2013.
- [45] XILINX INC. *ISE In-Depth Tutorial*. Disponível em: <[http://www.xilinx.com/support/documentation/sw\\_manuals/xilinx13\\_2/ise\\_tutorial\\_ug695.pdf](http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_2/ise_tutorial_ug695.pdf)>. Acesso em: 30 jan. 2013.
- [46] XILINX INC. *XST User Guide*. Disponível em: <[http://www.xilinx.com/support/documentation/sw\\_manuals/xilinx13\\_2/xst.pdf](http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_2/xst.pdf)>. Acesso em: 30 jan. 2013.
- [47] XILINX INC. *PlanAhead User Guide*. Disponível em: <[http://www.xilinx.com/support/documentation/sw\\_manuals/xilinx13\\_3/PlanAhead\\_UserGuide.pdf](http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_3/PlanAhead_UserGuide.pdf)>. Acesso em: 30 jan. 2013.
- [48] JAPPE, T. K. *Análise do retificador boost monofásico sob interrupções instantâneas da tensão de alimentação*. Dissertação

(Mestrado) - Universidade Federal de Santa Catarina, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, SC, Brasil. 2009.

[49] TODD, P. C. e TEXAS INSTRUMENTS. UC3854 Controlled Power Factor Correction Circuit Design. *UNITRODE Application Note U-134*, 1996.

[50] TOMASELLI, L. C. *Controle de um Pré-Regulador com Alto Fator de Potência Utilizando o Controlador DSP TMS320F243*. Dissertação (Mestrado) - Universidade Federal de Santa Catarina, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, SC, Brasil. 2001.

[51] REMOR, J. P. *Autocontrole de corrente aplicado ao conversor boost monofásico, para correção do fator de potência*. Dissertação (Mestrado) - Universidade Federal de Santa Catarina, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, SC, Brasil. 2004.

[52] LARICO, H. R. E. *Conversor Boost Controlado em Corrente Aplicado ao Retificador Monofásico*. Dissertação (Mestrado) - Universidade Federal de Santa Catarina, Programa de Pós-Graduação em Engenharia Elétrica, Florianópolis, SC, Brasil. 2007.

[53] PRODIC, A.;; ERICKSON, R. W. e MAKSIMOVIC, D. Predictive digital current programmed control. *IEEE Transactions on Power Electronics*, v. 18, n. 1, p. 411-419, doi:10.1109/TPEL.2002.807140, 2003.

[54] ANALOG DEVICES. *AD7366/AD7367 Datasheet*. Disponível em: <[http://www.analog.com/static/imported-files/data\\_sheets/AD7366\\_7367.pdf](http://www.analog.com/static/imported-files/data_sheets/AD7366_7367.pdf)>. Acesso em: 30 jan. 2013.

[55] MENTOR GRAPHICS. *ModelSim*. Disponível em: <<http://model.com/>>. Acesso em: 30 jan. 2013.

[56] ASTRÖM, K. J. e MURRAY, R. M. *Feedback Systems: An Introduction for Scientists and Engineers*. [S.l.]: Princeton University Press, 2008. p. 424

[57] PAIZ, C. et al. FPGA-based realization of self-optimizing drive-controllers. *2009 35th Annual Conference of IEEE Industrial Electronics*, p. 2848-2853, doi:10.1109/IECON.2009.5415402, 2009.

[58] OTHMAN, S. BEN et al. MPSoC design approach of FPGA-based controller for induction motor drive. In: 2012 IEEE INTERNATIONAL CONFERENCE ON INDUSTRIAL TECHNOLOGY. *Anais...* [S.l.]: IEEE, 2012.

[59] GODBOLE, R. e BHATTACHARYA, S. Design and Development of a Flexible Multi-Purpose Controller Hardware System for Power Electronics and Other Industrial Applications. In: 2008 IEEE

## INDUSTRY APPLICATIONS SOCIETY ANNUAL MEETING.

*Anais...* [S.l.]: IEEE, 2008.

[60] ZHAONING, Y. *Digital Controller Design for Cascaded-Multilevel-Converter Based STATCOM Systems*. Graduate Faculty of North Carolina State University - [S.l.]. 2006.

[61] HU, H. et al. A Universal Digital Platform and Software Library for Power Electronic Systems Integration. In: 2006 CES/IEEE 5TH INTERNATIONAL POWER ELECTRONICS AND MOTION CONTROL CONFERENCE. *Anais...* [S.l.]: IEEE, 2006.

[62] WANG, S.-C. e LIU, Y.-H. Software-Reconfigurable e-Learning Platform for Power Electronics Courses. *IEEE Transactions on Industrial Electronics*, v. 55, n. 6, p. 2416-2424, doi:10.1109/TIE.2008.922592, 2008.

[63] SCHAF, F. M. e PEREIRA, C. E. Integrating Mixed-Reality Remote Experiments Into Virtual Learning Environments Using Interchangeable Components. *IEEE Transactions on Industrial Electronics*, v. 56, n. 12, p. 4776-4783, doi:10.1109/TIE.2009.2026369, 2009.

[64] DONG, H. e HUSSAIN, F. K. Focused Crawling for Automatic Service Discovery, Annotation, and Classification in Industrial Digital Ecosystems. *IEEE Transactions on Industrial Electronics*, v. 58, n. 6, p. 2106-2116, doi:10.1109/TIE.2010.2050754, 2011.

[65] DAI, Y. e YAO, Y. An internet-based e-Experiment system for automatic control education and research. In: 2010 SECOND INTERNATIONAL CONFERENCE ON COMMUNICATION SYSTEMS, NETWORKS AND APPLICATIONS. *Anais...* [S.l.]: IEEE, 2010.



## APÊNDICE A – EXEMPLO DE VHDL PARA O HARDWARE DO USUÁRIO

```

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
entity power_electronics_hardware is
generic (
    REG_WIDTH : integer := 32; SPI_DATA_WIDTH : integer := 14
);
port (
    clk, rst      : in std_logic;  -- clock and reset
    en            : in std_logic;  -- enable
    -- ADC information exchange
    AD0_req      : out std_logic;
    AD1_req      : out std_logic;
    AD2_req      : out std_logic;
    AD0_rdy      : in std_logic;
    AD1_rdy      : in std_logic;
    AD2_rdy      : in std_logic;
    AD0DATA0     : in std_logic_vector(SPI_DATA_WIDTH-1 downto 0);
    AD0DATA1     : in std_logic_vector(SPI_DATA_WIDTH-1 downto 0);
    AD1DATA0     : in std_logic_vector(SPI_DATA_WIDTH-1 downto 0);
    AD1DATA1     : in std_logic_vector(SPI_DATA_WIDTH-1 downto 0);
    AD2DATA0     : in std_logic_vector(SPI_DATA_WIDTH-1 downto 0);
    AD2DATA1     : in std_logic_vector(SPI_DATA_WIDTH-1 downto 0);
    -- Registers
    -- Read only
    reg1         : in std_logic_vector(REG_WIDTH-1 downto 0);
    reg2         : in std_logic_vector(REG_WIDTH-1 downto 0);
    reg3         : in std_logic_vector(REG_WIDTH-1 downto 0);
    reg4         : in std_logic_vector(REG_WIDTH-1 downto 0);
    reg5         : in std_logic_vector(REG_WIDTH-1 downto 0);
    reg6         : in std_logic_vector(REG_WIDTH-1 downto 0);
    reg7         : in std_logic_vector(REG_WIDTH-1 downto 0);
    -- Write only
    reg8         : out std_logic_vector(REG_WIDTH-1 downto 0);
    reg9         : out std_logic_vector(REG_WIDTH-1 downto 0);
    reg10        : out std_logic_vector(REG_WIDTH-1 downto 0);
    reg11        : out std_logic_vector(REG_WIDTH-1 downto 0);
    reg12        : out std_logic_vector(REG_WIDTH-1 downto 0);
    -- Switches
    switch0      : out std_logic;
    switch1      : out std_logic;
    switch2      : out std_logic;
    switch3      : out std_logic;
    switch4      : out std_logic;
    switch5      : out std_logic;
    switch6      : out std_logic;
    switch7      : out std_logic
);
end entity power_electronics_hardware;

architecture beh of power_electronics_hardware is
begin
    -- Add code here
end beh;

```



## APÊNDICE B – TESTBENCH VHDL PARA O HARDWARE DO USUÁRIO

```

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
use ieee.numeric_std.all;
LIBRARY std;
use std.textio.all;
ENTITY tb IS
END tb;
ARCHITECTURE behavior OF tb IS
    -- Component Declaration for the Unit Under Test (UUT)
COMPONENT power_electronics_hardware port (
    clk, rst      : in std_logic; -- clock and reset
    en           : in std_logic; -- enable
    AD0_req     : out std_logic;
    AD1_req     : out std_logic;
    AD2_req     : out std_logic;
    AD0_rdy     : in std_logic;
    AD1_rdy     : in std_logic;
    AD2_rdy     : in std_logic;
    AD0DATA0    : in std_logic_vector(13 downto 0);
    AD0DATA1    : in std_logic_vector(13 downto 0);
    AD1DATA0    : in std_logic_vector(13 downto 0);
    AD1DATA1    : in std_logic_vector(13 downto 0);
    AD2DATA0    : in std_logic_vector(13 downto 0);
    AD2DATA1    : in std_logic_vector(13 downto 0);
    reg1        : in std_logic_vector(31 downto 0);
    reg2        : in std_logic_vector(31 downto 0);
    reg3        : in std_logic_vector(31 downto 0);
    reg4        : in std_logic_vector(31 downto 0);
    reg5        : in std_logic_vector(31 downto 0);
    reg6        : in std_logic_vector(31 downto 0);
    reg7        : in std_logic_vector(31 downto 0);
    reg8        : out std_logic_vector(31 downto 0);
    reg9        : out std_logic_vector(31 downto 0);
    reg10       : out std_logic_vector(31 downto 0);
    reg11       : out std_logic_vector(31 downto 0);
    reg12       : out std_logic_vector(31 downto 0);
    switch0     : out std_logic;
    switch1     : out std_logic;
    switch2     : out std_logic;
    switch3     : out std_logic;
    switch4     : out std_logic;
    switch5     : out std_logic;
    switch6     : out std_logic;
    switch7     : out std_logic
); END COMPONENT;
signal clk : std_logic := '0';
signal rst : std_logic := '1';
signal en : std_logic := '0';
signal a0d0, a0d1, a1d0, a1d1, a2d0, a2d1 : std_logic_vector(13
downto 0) := (others=>'0');
signal reg1, reg2, reg3, reg4, reg5, reg6, reg7 : std_logic_vector(31
downto 0) := (others=>'0');
signal reg8, reg9, reg10, reg11, reg12 : std_logic_vector(31 downto
0);
signal sw0,sw1,sw2,sw3,sw4,sw5,sw6,sw7 : std_logic;
signal adreq : std_logic := '0';
signal adrdy, adr0, adr1, adr2 : std_logic := '0';
file StimCSV : text;

```

```

constant clk_period : time := 10 ns;
BEGIN
  ReadAndApply: process
    variable c: character;
    variable v0,v1,v2,v3,v4,v5,v6,v7,v8,v9,v10,v11,v12 : integer;
    variable L: line;
  begin
    wait for 200 ns;
    file_open(StimCSV, "inputs.csv", READ_MODE);
    readline(StimCSV, L);
    while not endfile(StimCSV) loop
      wait until(adreq'event and adreq = '1');
      readline(StimCSV, L); -- Get the next stimulus line
      -- extract the value (ex: 10,20,25,1,2, ...)
      read(L, v0);read(L, c);
      assert c = ',' report "error after, v0";
      read(L, v1);read(L, c);
      assert c = ',' report "error after, v1";
      read(L, v2);read(L, c);
      assert c = ',' report "error after, v2";
      read(L, v3);read(L, c);
      assert c = ',' report "error after, v3";
      read(L, v4);read(L, c);
      assert c = ',' report "error after, v4";
      read(L, v5);read(L, c);
      assert c = ',' report "error after, v5";
      read(L, v6);read(L, c);
      assert c = ',' report "error after, v6";
      read(L, v7);read(L, c);
      assert c = ',' report "error after, v7";
      read(L, v8);read(L, c);
      assert c = ',' report "error after, v8";
      read(L, v9);read(L, c);
      assert c = ',' report "error after, v9";
      read(L, v10);read(L, c);
      assert c = ',' report "error after, v10";
      read(L, v11);read(L, c);
      assert c = ',' report "error after, v11";
      read(L, v12);
      --apply stimulus
      a0d0 <= STD_LOGIC_VECTOR(TO_SIGNED(v0,14));
      a0d1 <= STD_LOGIC_VECTOR(TO_SIGNED(v1,14));
      a1d0 <= STD_LOGIC_VECTOR(TO_SIGNED(v2,14));
      a1d1 <= STD_LOGIC_VECTOR(TO_SIGNED(v3,14));
      a2d0 <= STD_LOGIC_VECTOR(TO_SIGNED(v4,14));
      a2d1 <= STD_LOGIC_VECTOR(TO_SIGNED(v5,14));
      reg1 <= STD_LOGIC_VECTOR(TO_SIGNED(v6,32));
      reg2 <= STD_LOGIC_VECTOR(TO_SIGNED(v7,32));
      reg3 <= STD_LOGIC_VECTOR(TO_SIGNED(v8,32));
      reg4 <= STD_LOGIC_VECTOR(TO_SIGNED(v9,32));
      reg5 <= STD_LOGIC_VECTOR(TO_SIGNED(v10,32));
      reg6 <= STD_LOGIC_VECTOR(TO_SIGNED(v11,32));
      reg7 <= STD_LOGIC_VECTOR(TO_SIGNED(v12,32));
    end loop;
    file_close(StimCSV);
    report "!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!EOF";
    wait until clk='1';
    wait;
  end process;
  adreq <= adr0 OR adr1 OR adr2;
  -- Instantiate the Unit Under Test (UUT)
  uut: power_electronics_hardware PORT MAP (

```

```

        clk => clk,
        rst => rst,
        en => en,
-- ADC information exchange
AD0_req    => adr0,
AD1_req    => adr1,
AD2_req    => adr2,
AD0_rdy    => adrdy,
AD1_rdy    => adrdy,
AD2_rdy    => adrdy,
AD0DATA0   => a0d0,
AD0DATA1   => a0d1,
AD1DATA0   => a1d0,
AD1DATA1   => a1d1,
AD2DATA0   => a2d0,
AD2DATA1   => a2d1,
-- Registers
-- Read only
reg1       => reg1,
reg2       => reg2,
reg3       => reg3,
reg4       => reg4,
reg5       => reg5,
reg6       => reg6,
reg7       => reg7,
-- Write only
reg8       => reg8,
reg9       => reg9,
reg10      => reg10,
reg11      => reg11,
reg12      => reg12,
-- Switches
switch0    => sw0,
switch1    => sw1,
switch2    => sw2,
switch3    => sw3,
switch4    => sw4,
switch5    => sw5,
switch6    => sw6,
switch7    => sw7
    );
-- Clock process definitions
clk_process :process
begin
    clk <= '0';
    wait for clk_period/2;
    clk <= '1';
    wait for clk_period/2;
end process;
-- Stimulus process
stim_proc: process
begin
    -- hold reset state for 10 ns.
    wait for 10 ns;
    rst <= '0';
    wait for 50 ns;
    adrdy <= '1';
    wait;
end process;
END;
```



## APÊNDICE C – TUTORIAL DO FLUXO DE DESENVOLVIMENTO DA PLATAFORMA

1. Preparação do hardware do usuário no Fluxo do ISE
  - a. Colocar todos os arquivos VHDL e/ou Verilog utilizados em um diretório conhecido.
  - b. Copiar para este diretório os arquivos referentes ao *script* do fluxo ISE: *peh.scr*, *peh.prj* e *doit.bat*.
  - c. Editar o arquivo texto *peh.scr*, e adicionar os arquivos *.vhd* ou *.v* utilizados, de maneira hierárquica.
  - d. Executar o arquivo *doit.bat*.
  - e. Caso não existam erros nos arquivos fonte VHDL, será gerado um arquivo *netlist* com o nome de *power\_electronics\_hardware.ngc*, contendo o hardware do usuário. Em caso de falha na execução do *script*, procurar no arquivo de relatório gerado em modo texto (*peh.srp*) os erros ocorridos.
2. Criação do *bitstream* parcial no Fluxo do PlanAhead
  - a. Copiar o arquivo *peh\_script.tcl* para o diretório onde está o projeto base do PlanAhead.
  - b. Editar o arquivo texto *peh\_script.tcl* e modificar as variáveis no início do arquivo para os diretórios e nomes de módulos obtidos.
  - c. Abrir o software PlanAhead
  - d. Executar o arquivo *peh\_script.tcl* usando o menu “Tools>Run Tcl Script...”.
  - e. Após finalizado, os arquivos *bitstreams* completo e parcial estarão localizados no subdiretório *project\_windows.runs/<nome da run criada>*.