UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS–GRADUAÇÃO EM
ENGENHARIA ELÉTRICA

Marcos Paulo Moccelini

REAL-TIME SIMULATION OF POWER ELECTRONICS SYSTEMS AND HARDWARE-IN-THE-LOOP APPLICATIONS

Florianópolis

2018

Marcos Paulo Moccelini

Real-time simulation of power electronics systems and hardware-in-the-loop applications

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.

Supervisor: Prof. Marcelo Lobo Heldwein, Dr. sc. ETH
Co-supervisor: Prof. André Luís Kirsten, Dr. Eng.

Florianópolis

2018

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

Moccelini, Marcos Paulo
Real-time simulation of power electronics systems and hardware-in-the-loop applications / Marcos Paulo Moccelini ; orientador, Marcelo Lobo Heldwein, coorientador, André Luís Kirsten, 2018.
156 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós Graduação em Engenharia Elétrica, Florianópolis, 2018.

Inclui referências.

1. Engenharia Elétrica. 2. hardware-in-the-loop. 3. simulação em tempo real. 4. OPAL-RT. I. Heldwein, Marcelo Lobo. II. Kirsten, André Luís. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Engenharia Elétrica. IV. Título.

Marcos Paulo Moccelini

Real-time simulation of power electronics systems and hardware-in-the-loop applications

Esta dissertação foi julgada adequada para obtenção do título de Mestre em Engenharia Elétrica, na área de concentração em Eletrônica de Potência e Acionamento Elétrico, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina.

Florianópolis, 2018.

Prof. Bartolomeu F. Uchôa-Filho, Ph.D.
Coordenador do Programa de Pós–Graduação em Engenharia Elétrica

Banca Examinadora:

Prof. Marcelo Lobo Heldwein, Dr. sc. ETH
Orientador
Universidade Federal de Santa Catarina – UFSC

Prof. Denizar Cruz Martins, Dr. Ing.
Universidade Federal de Santa Catarina – UFSC

Prof. Márcio Silveira Ortmann, Dr.
Instituto Federal de Santa Catarina – IFSC

Victor Maryama, Me. (Videoconferência)
Typhoon HIL, Inc.

Acknowledgements

First of all, I would like to specifically thank my supervisors Marcelo Lobo Heldwein and André Luiz Kirsten for the vote of confidence and the opportunity to make this work a reality; my INEP colleagues André de Bastiani Lange and Marcos José Jacoboski for the precious tips given for the PCB design; and Dr. Christian Dufour from OPAL-RT for promptly elucidating parts of the theory when it was requested.

My gratitude to all INEP members for the occasional technical contributions and the companionship (this group sometimes feel like one big family), with a special mention to Lúcio Steckling, Jéssika Melo de Andrade, Edhuardo Francisco Celli Grabovski, Gustavo Pereira, Gean Jacques Maia de Sousa, Thiago Antonio Pereira, Márcio Moura Bridon Júnior, and Jacson Luis de Oliveira.

I want to thank my wife for the love, support, for being always there – in the good and bad days – and for taking such good care of our babies for the many hours I was away. One huge thanks goes to my two beautiful daughters; I find it amazing how such little sprouts are able to teach valuable life lessons in the most graceful manners.

Last but not least, I acknowledge the brazilian people. Despite the injustices and uncertainties many of them go through every day, part of their scarce, hard-earned money is what financed my master's course. I sincerely hope my acquired expertise allows me to repay fairly.

"Have you ever had a dream, Neo, that you were so sure was real? What if you were unable to wake from that dream? How would you know the difference between the dream world and the real world?"

Morpheus – The Matrix (1999)

Abstract

Once available exclusively for slow transient models such as those of power systems, the real-time simulation of electric circuits was improved with the growth of the available computing power, and is nowadays a crucial part of the design process of new electrical engineering solutions in several areas – especially when in a hardware-in-the-loop configuration. In this master's thesis some of the fundamental theoretical aspects of general circuit simulation softwares are initially outlined: the graph representation, the state-variable analysis and the nodal analysis. Afterwards, important characteristics and restrictions of real-time operation oriented methods are discussed. Finally, the evaluation of a state-of-the-art hardware-in-the-loop testing capable system, the CPU/FPGA-based OPAL-RT OP5700 real-time simulator, is presented in terms of speed and accuracy.

Keywords: hardware-in-the-loop; real-time simulation; OPAL-RT

Resumo

Uma vez disponível exclusivamente para modelos com transitórios lentos como os de sistemas de potência, a simulação em tempo-real de circuitos elétricos foi melhorada com o crescimento do poder computacional disponível, e é hoje parte crucial do processo de desenvolvimento de novas soluções de engenharia elétrica em diversas áreas – especialmente quando em uma configuração *hardware-in-the-loop*. Nesta dissertação, noções de alguns dos aspectos teóricos fundamentais de *softwares* de simulação de circuitos gerais são primeiramente mostradas: a representação por grafos, a análise por variáveis de estado e a análise nodal. Posteriormente, características e restrições importantes de métodos orientados à operação em tempo-real são discutidos. Finalmente, a avaliação de um sistema estado-da-arte apto a testes *hardware-in-the-loop*, o simulador em tempo-real baseado em CPU e FPGA OPAL-RT OP5700, é apresentada em termos de velocidade e exatidão.

Palavras-chave: hardware-in-the-loop; simulação em tempo real; OPAL-RT

Resumo expandido

A simulação por computador é parte importante e, muitas vezes, obrigatória no estágio de desenvolvimento de uma nova solução de engenharia. Essa afirmação é especialmente verdadeira na área da Engenharia Elétrica, onde as grandezas físicas envolvidas são capazes de causar danos ao ambiente e ao equipamento em frações de segundo e, em geral, não são diretamente observáveis. Simulações programadas a partir de modelos matemáticos criados para circuitos específicos podem apresentar maior velocidade e fidelidade de resultados, mas exigem um grande tempo de formulação de equações para cada circuito. Um *software* de simulação de circuitos gerais moderno sacrifica parte da exatidão e velocidade de simulação de modo a permitir uma modelagem muito rápida, onde elementos gráficos que representam os componentes são simplesmente adicionados e interligados.

Na simulação em tempo real (RTS) de circuitos elétricos, tensões e correntes do modelo respondem em sincronia com as mesmas grandezas do sistema real modelado. Quando há precisão suficiente e meios para serem realizadas conexões (como conversores analógicos/digitais e amplificadores), equipamentos reais podem ser adicionados como elementos da simulação, caracterizando a configuração *hardware-in-the-loop*. Dessa maneira, um controlador real, por exemplo, pode ser conectado à uma planta simulada. Esta técnica apresenta diversas vantagens, como a possibilidade de detecção de problemas de implementação em controladores antes da conexão ao dispositivo real, o que traz segurança tanto para o equipamento quando para o ambiente; redução de custos de operação caso o dispositivo real tenha alto consumo; possibilidade de testes de rotinas de segurança do controlador em casos extremos, como curto-circuito ou grandezas acima das faixas de operação; etc.

Objetivos

Esta dissertação possui dois objetivos principais, sendo um deles a avaliação de um sistema de simulação em tempo real da *OPAL-RT Technologies* adquirido para o laboratório do Instituto de Eletrônica de Potência da UFSC – o OP5700 – em termos de velocidade, exatidão e limitações, de modo a ser criado material de referência para futuros trabalhos com esse sistema. O segundo objetivo é prover uma visão introdutória concisa sobre a teoria por trás de programas simulação de circuitos gerais a não especialistas na área.

Metodologia

Testes comparativos entre resultados de simulação de métodos incluídos em pacotes proprietários da OPAL-RT e resultados do Simulink Simscape Power Systems – software já consolidado no mercado – foram conduzidos. Além disso, a capacidade do *hardware* do sistema de simulação, tanto em termos de poder de processamento quanto de operação adequada de entradas e saídas, foi avaliada. Para permitir a criação de testes significativos, um continuado e extenso estudo teórico sobre simulação de circuitos foi realizado.

Resultados e discussão

O processo de análise de um circuito demanda a união de dois conjuntos de informações: as características de tensão-corrente (V-I) dos elementos que o compõem e as leis topológicas – a Lei de Kirchoff das Tensões (KVL) e a Lei de Kirchoff das Correntes (KCL) –, que são sempre válidas e dizem respeito apenas à geometria do circuito. As topologias são estudadas pela teoria matemática de grafos, que é apresentada no primeiro capítulo desta dissertação.

Nos Capítulos 2 e 3 são descritos os dois métodos de análise mais comumente utilizados por simuladores: a Análise por Variáveis de Estado (SVA) e a Análise Nodal. Os principais destaques dos métodos SVA são a utilização de passo de cálculo variável sem necessidade cálculos extras, completa integração com modelos de espaço de estados de controladores e a facilidade de representação de elementos não-lineares. Métodos nodais, por sua vez, calculam

grandes circuitos rapidamente, uma vez que as saídas são encontradas a partir de uma equação matricial algébrica, além de possuírem formulação simples.

Como pode ser visto no Capítulo 4, a validade da simulação em tempo real depende do atendimento de alguns requisitos. O mais importante é a sincronização dos passos de cálculo com um "relógio real", de forma a permitir uma resposta dinâmica do modelo equivalente àquela do circuito real. Além disso, todos os cálculos devem ser terminados antes do início do passo de cálculo seguinte; caso contrário, é caracterizado um *overrun* e os resultados perdem confiabilidade.

No Capítulo 5 são apresentados os resultados de testes experimentais. O método State-Space Nodal do pacote ARTEMiS da OPAL-RT, que permitiu passos de cálculo mínimos da ordem de 10 μs, foi 8 vezes mais rápido que o método SVA padrão do Simscape Power Systems na comparação realizada, que envolvia um circuito com 120 diodos. Além disso, os dois métodos apresentaram exatidão semelhante. O método de Pejovic, utilizado para a criação de modelos para processamento pelo FPGA, permitiu a simulação em tempo real de um conversor MMC de 120 interruptores com um passo de cálculo de 1,25 μs e conversores mais simples com passo de até 150 ns, aproximadamente. Os erros introduzidos, todavia, foram significativos e sensíveis à alteração de parâmetros do circuito. De qualquer forma, os resultados das simulações *hardware-in-the-loop* mostraram que esses erros não suficientes para invalidar as simulações realizadas em FPGA; o engenheiro deve, contudo, interpretar adequadamente os resultados levando em conta as características do modelo.

Considerações finais

Espera-se que, suportado pela informação apresentada nesta dissertação, um usuário novo do sistema OPAL-RT OP5700 ou de sistema equivalente possa discernir quais métodos e parâmetros melhor se encaixam na simulação de circuito pretendida e quais elementos podem ser ignorados de forma a ser diminuído o passo de cálculo, sem que sejam gerados *overruns* ou que ocorra perda grande de exatidão.

List of Figures

List of Tables

List of abbreviations and acronyms

ADC	Analog-to-Digital Converter
ARTEMiS	Advanced Real-time Electromagnetic Simulator
DAC	Digital-to-Analog Converter
DPL	Distributed Parameters Line
DSC	Digital Signal Controller
ECU	Electronic Control Units
eHS	Electric Hardware Solver
EMTP	Electromagnetic Transients Program
FPGA	Field-Programmable Gate Array
HIL	Hardware-in-the-loop
IDE	Integrated Development Environment
INEP	*Instituto de Eletrônica de Potência* – Power Electronics Institute
IPD	In-phase Disposition
KCL	Kirchoff's Current Law
KVL	Kirchoff's Voltage Law
LTI	Linear time-invariant
MMC	Modular Multilevel Converter

MNA	Modified Nodal Analysis
OS	Operating System
PHIL	Power hardware-in-the-loop
PWM	Pulse Width Modulation
RTE	Real-time Events
RTS	Real-time Simulation
SFP	Small Form-factor Pluggable
SOM	Synchronous Optimum Modulation
SPS	Simscape Power Systems
SSN	State-Space Nodal
SVA	State Variable Analysis
SVM	Space Vector Modulation
TCRTS	Technical Committee on Real-Time Systems
UI	User interface
V-I	Voltage-current
VHDL	Very High Speed Integrated Circuits Hardware Description Language
VSI	Voltage Source Inverter

List of symbols

Ⓝ	Node identifier
-N-	Branch identifier
$\boldsymbol{A_a}$	Complete node-to-branch incidence matrix
$\boldsymbol{A}$	Incidence matrix
$\boldsymbol{B_a}$	Complete loop matrix
$\boldsymbol{B}$	Fundamental loop matrix
$\boldsymbol{U}$	Identity matrix
$\boldsymbol{D_a}$	Complete cutset matrix
$\boldsymbol{D}$	Fundamental cutset matrix
$\boldsymbol{\epsilon}$	Elementary matrix
$\square_k$	(Subscript) Branch-related
$\square_t$	(Subscript) Twig-related
$\square_l$	(Subscript) Link-related
$\hat{\boldsymbol{A}}$,$\hat{\boldsymbol{B}}$,$\hat{\boldsymbol{C}}$,$\hat{\boldsymbol{D}}$	State-space matrices
v_r	Node-to-reference voltage
$\boldsymbol{Y_r}$	Admittance matrix
$\boldsymbol{J_r}$	Equivalent nodal current source vector
y	Admittance matrix element
G	Conductance

C'	Capacitance per unit length
L'	Inductance per unit length
a	Wave propagation velocity
Δt	Time interval, time-step
T_s	Time-step
$\tilde{\boldsymbol{A}}$,$\tilde{\boldsymbol{B}}$	Discretized state-space matrices

Contents

Introduction

The computer simulation plays an important and almost mandatory role at the development stage of a new engineering solution nowadays. It is a procedure where a mathematical representation of the significant aspects of the system being studied is created, and a computer calculates the outputs given a set of initial conditions. This allows the engineer to analyze the system when operated with relevant inputs and to observe expected or even unexpected system behaviors more clearly, and then evaluate the need for project redesigns. This is especially true in the field of Electrical Engineering, where the involved physical quantities are able to cause damage to equipment or the environment in fractions of a second and are typically not naturally observable to the eye.

One possible approach to the simulation of a system is the initial analytical construction of a mathematical model by the engineer, followed by the conversion of this model to a computer program by means of a programming language, and subsequent processing by the computer. This method allows for a specifically tailored and optimized system model, with accuracy/speed compromise adjusted to the needs of the engineer. However, the analytical modeling of complex systems is often a difficult process and with restricted results, which means it must be redone every time there is a new system to be simulated.

A **general purpose** simulation software sacrifices some pre-

cision and speed of the calculations to favor a faster modeling process than with the previous approach. In modern programs, the user inputs the system components with the help of a graphical interface, configures the relevant parameters, selects the desired outputs and then starts the simulation. The needed mathematical model is created internally and automatically with advanced algorithms that have been developed and perfected for decades. Some examples of well known general purpose electric circuit simulation programs are PSPICE; PSCAD; PSIM; Multisim; SIMULINK and EMTP-RV.

In the **Real-time Simulation** (RTS) of electric circuits, voltages and currents of the simulated system show a "real clock" time response equivalent to those of the real system being modeled [1]. The first applications of RTS were in power systems transient analysis, and were performed with analog scaled down versions of real systems, such as the Transient Network Analyzer [2]. Although the first digital RTS's were performed with supercomputers, the immense evolution of processing power and reduction of costs allowed the transition to computer clusters, followed by PC-level processors and, nowadays, DSPs and FPGAs. The creation and refinement of specialized algorithms were also critical for the digital RTS of large networks and systems with fast transients, such as power electronics converters with switching frequencies of the order of tens of kHz.

With an accurate enough simulated model, real hardware such as protective devices, controllers, electrical machines, etc., can be tested in a closed-loop with the simulated system instead of the real one, provided the means to realize the connections (DACs / ADCs / amplifiers). This testing configuration is termed **hardware-in-the-loop** (HIL) and presents several advantages [1]:

- Hardware testing earlier in the design process, in the case the real system is not yet available;
- Significant reduction of costs involved with the testing, especially if the real system demands expensive resources to operate;
- Modifications to the real system usually require physical changes, which may be difficult, dangerous and resource consuming. Mod-

ifications in the simulated system are relatively fast and trivial, and some can even be done on the fly;

- Critical design flaws in the hardware under test do not translate to risks to the real system, resulting in a safer environment for both equipment and personnel. This extra safety allows, for example, the preview of controller behavior under extreme system conditions, such as failures and above ratings operation.

HIL testing has been used for over two decades in the development of car electronic control units (ECU) [3, 4] and in recent years expanded to many areas such as wind power [5], aircraft control [6], microgrids [7], etc. In light of recent trends and the potential improvements brought to power electronics research activities, it was acquired for INEP[1] (*Instituto de Eletrônica de Potência* – Power Electronics Institute) a state of the art RTS system from the Canada-based company and market leader OPAL-RT. The OP5700 is an x86 and FPGA based computer that runs a real-time Linux OS. It features multiple I/O modules for HIL testing and a software package with RTS specialized algorithms that run on top of a Simulink Simscape Power Systems basis.

One of the main objectives of this master's thesis is the evaluation of OP5700's speed, accuracy and limitations in order to create reference material for future INEP works, or other interested parties. Seeking the presentation of a fairer and richer analysis, a study on general purpose electric circuit simulation theory and on OPAL-RT's implemented methods was initially conducted; part of the theory is presented herein and aims to provide a concise introductory view for nonspecialists in the area – the other objective of this work.

Chapter 1 presents some graph theory definitions, which are the key to the translation of circuit topologies (the geometry) to a form computers can easily work with, i.e., matrices. In addition to geometry information, the physical voltage-current (V-I) characteristics of the circuit elements also must be converted to a mathematical model for a complete analysis. Most circuit simulation softwares

[1] A research laboratory installed at UFSC's Technological Center.

today use a variation of the State Variable Analysis and/or the Nodal Analysis, which are outlined in Chapters 2 and 3, respectively. Chapter 4 initially shows some important characteristics of RTS's and then presents a description of OPAL-RT's system and its implemented strategies to achieve real-time operation with small enough time-steps. Finally, in Chapter 5, the results of several tests performed to evaluate OPAL-RT's simulation system and specialized methods, including speed, accuracy and HIL operation are reported and commented.

Chapter 1

Network topology

A complete electrical network description requires two sets of information: the list of elements, and how they are interconnected. The physical properties of an element are a key part to determine the voltages and currents of the circuit. However, there are a set of rules that are independent of the elements contained in the network, being defined only by the geometrical configuration: Kirchhoff's Current Law (KCL) and Voltage Law (KVL). Those laws and other properties related to the interconnections are studied by network topology, which is a topic in a branch of mathematics called **graph theory**. This chapter presents some graph concepts relevant to the construction of computer-aided electric circuit simulation methods. The present and the two following chapters of this master's thesis are thoroughly based on references [8, 9, 10, 11, 12, 13, 14, 15], which share much of the information presented and will not be repeatedly cited to maintain readability.

1.1 TERMINOLOGY

Two electrical circuits can be seen in Figure 1.1. They are composed of different elements, but their geometry is equivalent. Each element in them may be represented by a line segment called a **branch**. The point at which different branches meet, i.e., the connection point, is called a **node**. In this work, nodes are enumerated in the form (N), while -N- represents branch numbers.

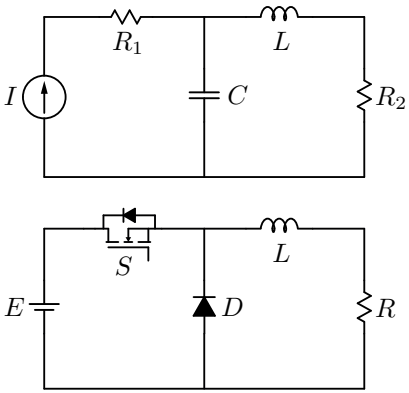


Figure 1.1 – Different circuits with the same geometry.

A **graph** is the structure formed by the interconnection of nodes and branches. Since their topology is the same, a single **oriented** graph representation of both presented circuits is constructed and shown in Figure 1.2. This graph is oriented because each branch has a direction set by an arrow, which is used to define a reference for the analysis of voltages and currents in the elements: the branch current direction agrees with the arrow, which points to the negative voltage sign. Any group of nodes and branches of a graph, with at least one node included, is called a **subgraph**.

A **path** is a set of branches that connect two **terminal** nodes and satisfy three conditions: (a) consecutive branches b_i and b_{i+1} have one common node; (b) two and only two branches of the sequence are incident at each non-terminal node; (c) only one branch is incident at terminal nodes. For example, in Figure 1.2 the branches -1-, -2- and -3- form a path from (4) to (3). Branch -4- is also a path

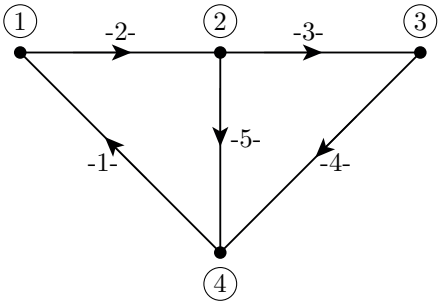


Figure 1.2 – Example oriented graph representation.

between those nodes. However, the sequence -1-, -2-, -5-, -4- is not a path because it breaks rule (c).

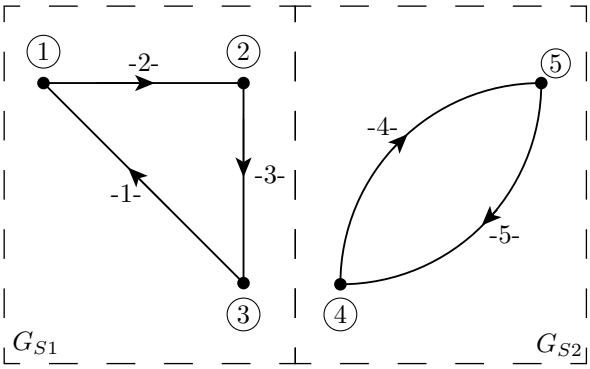


Figure 1.3 – Unconnected graph.

A graph is **connected** if there is at least one path between *any* two nodes. Figure 1.3 shows an unconnected graph composed of the connected subgraphs G_{S1} and G_{S2}. A **loop** is a closed path formed when terminal nodes coincide, i.e., there is only one terminal node. Branches -1-, -2-, -3- and -4- in Figure 1.2 constitute a loop, as well as the branches -1-, -2- and -3- in Figure 1.3.

A **tree** is a connected subgraph of a connected graph that contains every node, and has no loops. Some of the possible trees of the graph in Figure 1.4 are the ones formed by the branch sets (-1-, -2-,-5-) and (-3-, -4-, -5-). The branches of a graph included in a tree

are called **twigs**, while the remaining branches (low opacity in the example figure) are called **links** and form the **cotree**.

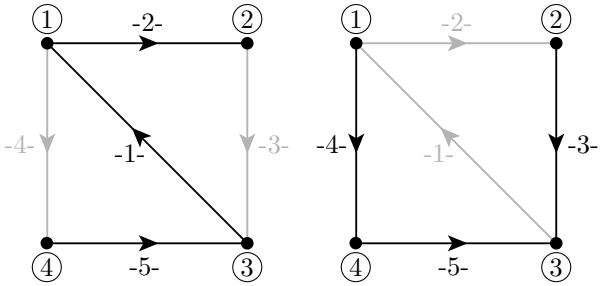


Figure 1.4 – Graph trees.

It can be shown that *for any connected graph with* $n+1$ *nodes, the number of twigs is* n. A very useful consequence is that *if* n *branches of a graph* G *with* $n+1$ *nodes are selected and no loop is formed, the resulting branch set is a tree of* G.

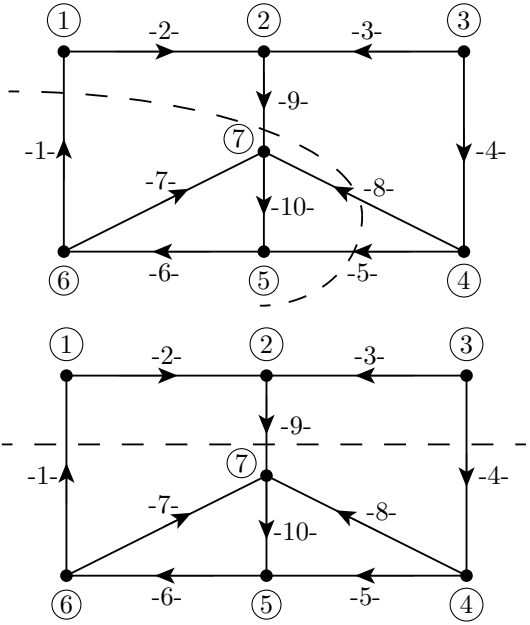


Figure 1.5 – Proper cuts.

Another important definition is the graph **cut**. By removing specific branches, a graph is disconnected and split into two or more subgraphs. If the branches selected split the graph into exactly two connected subgraphs with at least one node, and the restoration of any of those branches reconnects the graph, a **proper** cut is defined (Figure 1.5). Incorrect cuts can be seen in Figure 1.6: the first cut results in three subgraphs, while the second cut includes branch -7-, which is not connected to node ① (the graph stays unconnected upon its restoration). Branches of a proper cut form a **cutset**.

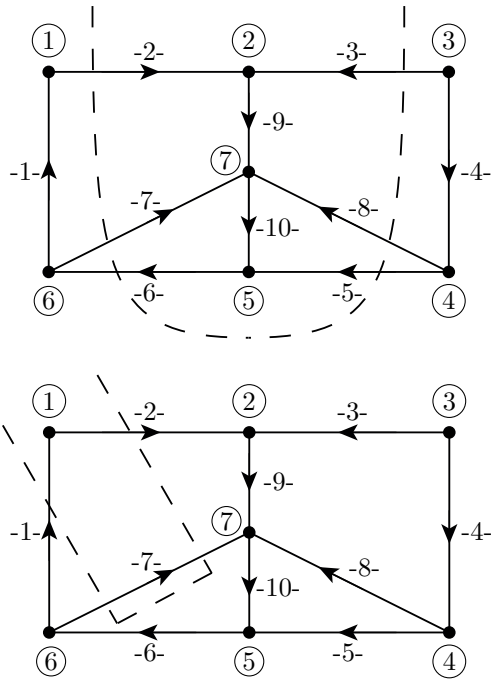


Figure 1.6 – Incorrect cuts.

A cut may be oriented by choosing a direction from one subgraph to the other. In Figure 1.7 the cut is oriented from the inner subgraph to the outer subgraph, as the arrow crossing the cut indicates. Furthermore, cutset branches have a direction relative to the cut. For instance, branch -10- has the same direction as the cut, while -7-, -8- and -9- have opposite directions.

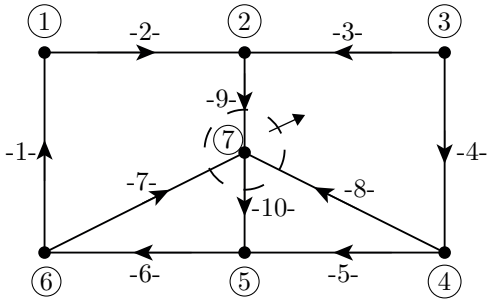


Figure 1.7 – Oriented cut.

1.2 THE INCIDENCE MATRIX

The KCL states that, in an instant of time t, at a node (N) of an electrical circuit, the algebraic sum of the currents flowing into (N) is equal to the sum of the currents flowing out of (N). That is:

$$\sum_{k=1}^{b_N} i_k(t) = 0 \tag{1.1}$$

where b_N is the number of branches connected to the node (N).

A more general equation, which comprises every node of a circuit is presented next. If a set of coefficients a_{jk}, where j corresponds to one of the $n+1$ nodes of the circuit and k to one of the b_N branches, are determined using the following convention:

- if branch k is incident at node j and its direction is leaving the node, $a_{jk} = 1$;
- if branch k is not incident at node j, $a_{jk} = 0$;
- if branch k is incident at node j and it is directed towards the node, $a_{jk} = -1$;

KCL equations may be then written as

$$\boxed{\sum_{k=1}^{b_N} a_{jk} i_k = 0 \qquad j = 0,1,\ldots,n+1} \tag{1.2}$$

Considering the graph of Figure 1.2, the expanded form of (1.2) is:

$$\begin{aligned} a_{11}i_1 + a_{12}i_2 + a_{13}i_3 + a_{14}i_4 + a_{15}i_5 = 0 \\ a_{21}i_1 + a_{22}i_2 + a_{23}i_3 + a_{24}i_4 + a_{25}i_5 = 0 \\ a_{31}i_1 + a_{32}i_2 + a_{33}i_3 + a_{34}i_4 + a_{35}i_5 = 0 \\ a_{41}i_1 + a_{42}i_2 + a_{43}i_3 + a_{44}i_4 + a_{45}i_5 = 0 \end{aligned} \tag{1.3}$$

Substituting the coefficients in accordance to the convention:

$$\begin{aligned} (-1)i_1 + (+1)i_2 + \ (0)i_3 + \ (0)i_4 + \ (0)i_5 = 0 \\ (0)i_1 + (-1)i_2 + (+1)i_3 + \ (0)i_4 + (+1)i_5 = 0 \\ (0)i_1 + \ (0)i_2 + (-1)i_3 + (+1)i_4 + \ (0)i_5 = 0 \\ (+1)i_1 + \ (0)i_2 + \ (0)i_3 + (-1)i_4 + (-1)i_5 = 0 \end{aligned} \tag{1.4}$$

By further simplifying, it becomes evident those are the KCL equations written in the usual form:

$$\begin{aligned} -i_1 + i_2 = 0 \\ i_2 - i_3 - i_5 = 0 \\ i_3 - i_4 = 0 \\ -i_1 + i_4 + i_5 = 0 \end{aligned} \tag{1.5}$$

The a_{ij} coefficients may be grouped inside a $(n+1) \times b$ matrix called the **complete node-to-branch incidence matrix** ($\boldsymbol{A_a}$), which describes how the branches are connected to the nodes of the circuit, i.e., its geometry. The "complete" word means all the nodes are contained in it. If a vector $\boldsymbol{i}$ consists of every branch current, Equation 1.2 may be simply written as:

$$\boldsymbol{A_a i} = \boldsymbol{0} \tag{1.6}$$

The $\boldsymbol{A_a}$ matrix of the previous example is:

$$\boldsymbol{A_a} = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 1 \\ 0 & 0 & -1 & 1 & 0 \\ 1 & 0 & 0 & -1 & -1 \end{bmatrix} \tag{1.7}$$

A branch, by definition, connects only two nodes of the circuit. Therefore, each column of the $\boldsymbol{A_a}$ matrix will contain exactly one $+1$ element, one -1 element and $n-1$ zero elements. Since the sum of every row, column by column, results in a row of zeros, at least one of the rows of $\boldsymbol{A_a}$ is not linearly independent. In other words, $\boldsymbol{A_a}$ contains redundant information. By removing one row of $\boldsymbol{A_a}$, the $n \times b$ **reduced node-to-branch incidence matrix**[1] ($\boldsymbol{A}$) is obtained. It can be shown [9] that this matrix is non-singular by proving its rank[2] is n, the number of rows. Consequently, the equation

$$\boldsymbol{Ai} = \boldsymbol{0} \tag{1.8}$$

describes the maximum set of independent KCL equations.

1.3 LOOP MATRIX

The KVL states that *in an instant of time, the algebraic sum of the branch voltages in each loop of a circuit equals zero*, i.e.,

$$\sum_{k=1}^{b_l} v_k(t) = 0 \qquad l = 1, 2, \ldots, n_l \tag{1.9}$$

where b_l is the number of branches of loop l, and n_l is the total number of loops.

The sign of each branch voltage in Equation 1.9 is relative to a chosen loop orientation. In Figure 1.8 branches -1-, -4- and -5- are part of loop l_1 and their directions oppose l_1's direction, while branches -1-, -2- and -3- are in loop l_2 with matching directions. Considering the following convention:

- if branch k is in loop j and their directions match, $b_{jk} = +1$
- if branch k is not in loop j, $b_{jk} = 0$
- if branch k is in loop j and their directions oppose, $b_{jk} = -1$

[1] Referred to, in this work, simply as the **incidence matrix**.
[2] The order of the largest non-singular square submatrix.

KVL equations may be written as:

$$\boxed{\sum_{k=1}^{b} b_{jk} v_k = 0 \qquad l = 1, 2, \ldots, n_l} \tag{1.10}$$

The coefficients b_{jk} constitute the $n_l \times b$ **complete loop matrix** $\boldsymbol{B_a}$. Equation 1.10 can be written in matrix form as

$$\boldsymbol{B_a v} = \boldsymbol{0} \qquad \boldsymbol{v} = [v_1 \; v_2 \; \ldots \; v_b]^T \tag{1.11}$$

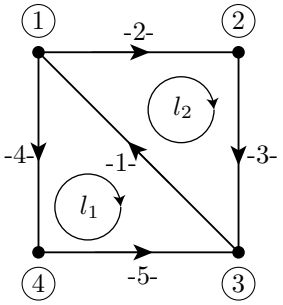


Figure 1.8 – Oriented loops.

Although only two loops (l_1 and l_2) are shown in the graph of Figure 1.8, branches -2-, -3-, -4- and -5- form another loop, thus n_l = 3. This extra loop, however, doesn't add new information because its KVL equation is a linear combination of the KVL equations of l_1 and l_2. Even for a relatively small network n_l can be a very large number, which means it is necessary to ignore redundant equations for an efficient calculation.

It is possible to find only independent KVL equations with the aid of a tree. A **fundamental loop** is the one formed by the combination of one or more twigs and a single link. In a graph with $n+1$ nodes and b branches, there are n twigs and $b-n$ links; therefore $b-n$ is the number of fundamental loops. The direction of each fundamental loop agrees with that of the originating link.

The **fundamental loop matrix** $\boldsymbol{B}$ is a submatrix of $\boldsymbol{B_a}$ which contains only fundamental loops. For the case of Figure 1.8, if

the tree formed by (-2-, -3-, -4-) is chosen,

$$\boldsymbol{B} = \begin{array}{c} \begin{array}{ccc|cc} \text{-2-} & \text{-3-} & \text{-4-} & \text{-1-} & \text{-5-} \end{array} \\ \left[\begin{array}{ccc|cc} 1 & 1 & 0 & 1 & 0 \\ -1 & -1 & 1 & 0 & 1 \end{array}\right] \end{array} \tag{1.12}$$

The order in $\boldsymbol{B}$ is $\left[\boldsymbol{B_T} \mid \boldsymbol{B_L}\right]$ (twigs first; links last), which is an useful way of ordering branch columns of topological matrices, as it will be seen later. It should be clear from the manner $\boldsymbol{B}$ is constructed (a single link per row) that, by using this order, it may be always be partitioned as $\left[\boldsymbol{B_T} \mid \boldsymbol{U}\right]$, where $\boldsymbol{U}$ is the identity (unit) matrix. The presence of $\boldsymbol{U}$ proves the rows of $\boldsymbol{B}$ are linearly independent. Furthermore, it can be shown that any loop of $\boldsymbol{B_a}$ added to $\boldsymbol{B}$ is a linear combination of the latter's rows [8]. $\boldsymbol{Bv} = \boldsymbol{0}$ hence corresponds to the maximum set of independent KVL equations.

1.4 CUTSET MATRIX

A more general form of the KCL may be written as: *at an instant of time, the algebraic sum of all currents through a cutset, from one subgraph to the other, is zero.* This form is more general because each equation can include branches incident at different nodes, while the usual KCL equations include only the branches incident at a single node.

Given a graph with n_c possible oriented cutsets, the $n_c \times b$ **complete cutset matrix** $\boldsymbol{D_a}$ is defined as:

$$\boldsymbol{D_a} = [d_{jk}] \tag{1.13}$$

where

- if branch k is in cutset j and their directions match, $d_{jk} = +1$;
- if branch k is not in cutset j, $d_{jk} = 0$;
- if branch k is in cutset j and their directions oppose, $d_{jk} = -1$.

Like the complete loop matrix, the complete cutset matrix comprises linearly dependent rows. By choosing a tree, a **fundamental cutset matrix** $\boldsymbol{D}$ can be constructed. Each twig together with a

combination of links generates a fundamental cutset. The orientation of each cutset agrees with the originating twig. Since there are n twigs in a connected graph with $n+1$ nodes, there are n fundamental cutsets. As an example, Figure 1.9 shows the fundamental cutsets for the arbitrarily chosen tree (-2-, -3-, -4-). The fundamental cutset matrix in this case is

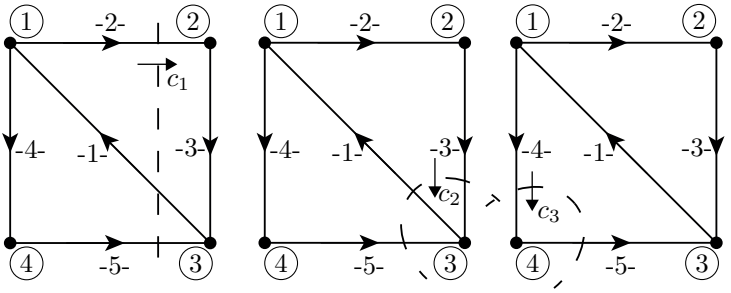


Figure 1.9 – Fundamental cutsets.

$$\boldsymbol{D} = \begin{array}{c} \begin{matrix} \text{-2-} & \text{-3-} & \text{-4-} & \text{-1-} & \text{-5-} \end{matrix} \\ \left[\begin{array}{ccc:cc} 1 & 0 & 0 & -1 & 1 \\ 0 & 1 & 0 & -1 & 1 \\ 0 & 0 & 1 & 0 & -1 \end{array}\right] \end{array} \tag{1.14}$$

It should be clear from the way $\boldsymbol{D}$ is constructed (a single twig per row) that, by using the branch column ordering of (1.14), it may be always be partitioned as $\left[\boldsymbol{U} \mid \boldsymbol{D_L}\right]$. It is obvious from the presence of $\boldsymbol{U}$ that the rows of $\boldsymbol{D}$ are linearly independent. Furthermore, it can be shown that any extra cutset added to $\boldsymbol{D}$ is a linear combination of its rows [8]. Therefore, $\boldsymbol{Di} = \boldsymbol{0}$ corresponds to the maximum set of independent KCL equations.

1.5 BRANCH VARIABLES CORRELATIONS

This section introduces some very useful and simple equations that arise from the constraints imposed by Kirchoff's laws. Consider

the following partitions given a tree:

$$\boldsymbol{B} = \left[\boldsymbol{B_T} \;\vdots\; \boldsymbol{U_\beta}\right] \qquad \boldsymbol{D} = \left[\boldsymbol{U_\delta} \;\vdots\; \boldsymbol{D_L}\right] \qquad \boldsymbol{v} = \begin{bmatrix}\boldsymbol{v_T}\\ \hdashline \boldsymbol{v_L}\end{bmatrix} \qquad \boldsymbol{i} = \begin{bmatrix}\boldsymbol{i_T}\\ \hdashline \boldsymbol{i_L}\end{bmatrix}$$

where the identity matrices subscripts $\beta = b - n$ and $\delta = n$ indicate their size. If the branch columns of $\boldsymbol{B}$ and $\boldsymbol{D}$ are arranged in the same order, it is possible to establish the following relationship between link and twig voltages:

$$\boldsymbol{Bv} = \boldsymbol{0} = \left[\boldsymbol{B_T} \;\vdots\; \boldsymbol{U_\beta}\right] \begin{bmatrix}\boldsymbol{v_T}\\ \hdashline \boldsymbol{v_L}\end{bmatrix} = \boldsymbol{B_T v_T} + \boldsymbol{v_L} \tag{1.15}$$

$$\boxed{\boldsymbol{v_L} = -\boldsymbol{B_T v_T}} \tag{1.16}$$

A similar expression can be found for twig and link currents:

$$\boldsymbol{Di} = \boldsymbol{0} = \left[\boldsymbol{U_\delta} \;\vdots\; \boldsymbol{D_L}\right] \begin{bmatrix}\boldsymbol{i_T}\\ \hdashline \boldsymbol{i_L}\end{bmatrix} = \boldsymbol{i_T} + \boldsymbol{D_L i_L} \tag{1.17}$$

$$\boxed{\boldsymbol{i_T} = -\boldsymbol{D_L i_L}} \tag{1.18}$$

Furthermore, the following relationships between topological matrices are all valid (proof can be found on [8]):

$$\begin{aligned} \boldsymbol{B_a A_a}^t &= \boldsymbol{0} & \boldsymbol{A_a B_a}^t &= \boldsymbol{0} \\ \boldsymbol{B_a A}^t &= \boldsymbol{0} & \boldsymbol{A B_a}^t &= \boldsymbol{0} \\ \boldsymbol{B A}^t &= \boldsymbol{0} & \boldsymbol{A B}^t &= \boldsymbol{0} \\ \boldsymbol{B_a D_a}^t &= \boldsymbol{0} & \boldsymbol{D_a B_a}^t &= \boldsymbol{0} \\ \boldsymbol{B D}^t &= \boldsymbol{0} & \boldsymbol{D B}^t &= \boldsymbol{0} \\ & & \boldsymbol{D_a B}^t &= \boldsymbol{0} \end{aligned} \tag{1.19}$$

Voltage loop and cutset matrices may be easily converted one to the other:

$$\boldsymbol{DB^t} = \left[\boldsymbol{U_\delta} \;\vdots\; \boldsymbol{D_L}\right] \left[\boldsymbol{B_T} \;\vdots\; \boldsymbol{U_\beta}\right]^t = \boldsymbol{B_T}^t + \boldsymbol{D_L} = \boldsymbol{0} \tag{1.20}$$

$$\boxed{\boldsymbol{B_T} = -\boldsymbol{D_L}^t \quad \text{or} \quad \boldsymbol{D_L} = -\boldsymbol{B_T}^t} \tag{1.21}$$

The currents in all branches are expressible in terms of link currents with the help of Equation 1.18 and Equation 1.21:

$$\boldsymbol{i} = \begin{bmatrix} \boldsymbol{i_T} \\ \hline \boldsymbol{i_L} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{D_L} \boldsymbol{i_L} \\ \hline \boldsymbol{i_L} \end{bmatrix} = \begin{bmatrix} -\boldsymbol{D_L} \\ \hline \boldsymbol{U_\beta} \end{bmatrix} \boldsymbol{i_L} = \begin{bmatrix} \boldsymbol{B_T}^t \\ \hline \boldsymbol{U_\beta} \end{bmatrix} \boldsymbol{i_L} \tag{1.22}$$

$$\boxed{\boldsymbol{i} = \boldsymbol{B}^t \boldsymbol{i_L}} \tag{1.23}$$

In the same manner, branch voltages can be expressed in terms of twig voltages by using Equations 1.16 and 1.21:

$$\boldsymbol{v} = \begin{bmatrix} \boldsymbol{v_T} \\ \hline \boldsymbol{v_L} \end{bmatrix} = \begin{bmatrix} \boldsymbol{v_T} \\ \hline -\boldsymbol{B_T} \boldsymbol{v_T} \end{bmatrix} = \begin{bmatrix} \boldsymbol{U_\delta} \\ \hline -\boldsymbol{B_T} \end{bmatrix} \boldsymbol{v_T} = \begin{bmatrix} \boldsymbol{U_\delta} \\ \hline \boldsymbol{D_L}^t \end{bmatrix} \boldsymbol{v_T} \tag{1.24}$$

$$\boxed{\boldsymbol{v} = \boldsymbol{D}^t \boldsymbol{v_T}} \tag{1.25}$$

1.6 COMPUTER GENERATION OF THE TOPOLOGICAL MATRICES

Finding a tree of a given graph by visual inspection and afterwards the $\boldsymbol{B}$ and $\boldsymbol{D}$ matrices is often a trivial job, but developing a computer program that does it automatically and in an efficient way can be a complex task. The formulation of the $\boldsymbol{A_a}$ matrix though is usually direct and based on the user input, which must contain the following triplet for each branch: a branch identifier, the node where the branch starts, and the node where it ends. This can be exemplified in the form of a PSIM *netlist* (also called a connection table). The following line was added to a model netlist after the placement of a resistor (visually) in a circuit:

```
R Rout 1 2 10 0
```

It translates to: *a resistor element named* Rout *connects node* ① *to node* ②*, is directed from* ① *to* ② *, has a* 10 Ω *resistance and its current measurement is deactivated.* Therefore, a branch column of the $\boldsymbol{A_a}$ matrix identified by Rout, has a -1 element in node ①'s row, a $+1$ element in node ②'s row, and zeros filling the remaining rows. The incidence matrix $\boldsymbol{A}$ is obtained when a reference (also called mass or datum) node is selected and its corresponding row is removed from $\boldsymbol{A_a}$.

1.6.1 Finding a tree

The presented method for the selection of a tree is based on the following theorem: *a set of n branches of a graph with n+1 nodes form a tree if and only if the corresponding n columns of the matrix* $\boldsymbol{A}$ *are linearly independent* [9].

Equation 1.26 shows a hypothetical incidence matrix already reduced to a row echelon form; Equation 1.27 is obtained after reordering branch columns. It is easy to see, considering the aforementioned theorem, branches -1-, -2-, -3-, -5- and -7- are twigs. The computer program therefore must write the matrix $\boldsymbol{A}$ in a row echelon form (Gaussian elimination); the resulting columns with the number 1 as the pivot are linearly independent and thus the corresponding branches are selected as the twigs.

$$\boldsymbol{A} = \begin{array}{c} \begin{array}{ccccccc} \text{-1-} & \text{-2-} & \text{-3-} & \text{-4-} & \text{-5-} & \text{-6-} & \text{-7-} \end{array} \\ \begin{bmatrix} \mathbf{1} & 0 & 0 & -1 & 1 & -1 & 0 \\ 0 & \mathbf{1} & -1 & 0 & 0 & 1 & 0 \\ 0 & 0 & \mathbf{1} & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \end{bmatrix} \end{array} \tag{1.26}$$

$$\boldsymbol{A} = \begin{array}{c} \begin{array}{ccccc:cc} \text{-1-} & \text{-2-} & \text{-3-} & \text{-5-} & \text{-7-} & \text{-4-} & \text{-6-} \end{array} \\ \left[\begin{array}{ccccc:cc} \mathbf{1} & 0 & 0 & 1 & 0 & -1 & -1 \\ 0 & \mathbf{1} & -1 & 0 & 0 & 0 & 1 \\ 0 & 0 & \mathbf{1} & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \end{array}\right] \end{array} \tag{1.27}$$

1.6.2 Generating the loop and cutset matrices

Once a tree has been determined, $\boldsymbol{B}$ and $\boldsymbol{D}$ can be calculated. Starting with the relation $\boldsymbol{AB^t = 0}$:

$$\boldsymbol{AB^t = 0} = \left[\boldsymbol{A_T} \;\vdots\; \boldsymbol{A_L}\right] \begin{bmatrix} \boldsymbol{B_T}^t \\ \hdashline \boldsymbol{U_\beta} \end{bmatrix} = \boldsymbol{A_T B_T}^t + \boldsymbol{A_L} \tag{1.28}$$

Thus,

$$\boldsymbol{B_T}^t = -\boldsymbol{A_T}^{-1}\boldsymbol{A_L} \tag{1.29}$$

Substituting in Equation 1.21:

$$\boldsymbol{D_L} = \boldsymbol{A_T}^{-1}\boldsymbol{A_L} \tag{1.30}$$

Hence, $\boldsymbol{D}$ can be equated to

$$\boldsymbol{D} = \left[\boldsymbol{U_\delta} \;\vdots\; \boldsymbol{D_L}\right] = \boldsymbol{A_T}^{-1}\left[\boldsymbol{A_T} \;\vdots\; \boldsymbol{A_L}\right] = \boldsymbol{A_T}^{-1}\boldsymbol{A} \tag{1.31}$$

Finally,

$$\boldsymbol{B} = \left[\boldsymbol{B_T} \;\vdots\; \boldsymbol{U_\beta}\right] = \left[-\boldsymbol{D_L}^t \;\vdots\; \boldsymbol{U_\beta}\right] \tag{1.32}$$

The cutset matrix thus may be found from the incidence matrix and afterwards the cutset links submatrix can be used to calculate the voltage loop matrix. However, in order to calculate (1.31) directly, the inversion of $\boldsymbol{A_T}$ must be performed. This can be very inefficient because matrix inversions are computationally intensive for large sizes. An alternative route is shown next, employing elementary matrices. An elementary matrix is any matrix obtained after a single elementary row operation is performed on the identity matrix. The following theorems on elementary matrices are valid [8]:

Every elementary matrix has an inverse, which is also elementary.

To perform an elementary row operation on a matrix $\boldsymbol{Q}$, calculate the product $\boldsymbol{\epsilon Q}$, where $\boldsymbol{\epsilon}$ is the elementary matrix obtained by performing the intended row operation on the identity matrix.

For example, consider the following generic $\boldsymbol{Q}$ matrix:

$$\boldsymbol{Q} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \tag{1.33}$$

If the intended elementary operation is to add 2 times the third row of $\boldsymbol{Q}$ to the first, the operation can be done on the $\boldsymbol{U}$ matrix, resulting in $\boldsymbol{\epsilon}$:

$$\boldsymbol{\epsilon} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.34}$$

and then the $\boldsymbol{\epsilon}$ matrix is multiplied by $\boldsymbol{Q}$

$$\boldsymbol{\epsilon Q} = \begin{bmatrix} 1 & 4 & 3 & 6 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \tag{1.35}$$

Furthermore $\boldsymbol{Q}$ is a nonsingular square matrix, there is always a sequence of m elementary row operations that reduce $\boldsymbol{Q}$ to the identity, i.e.:

$$\boldsymbol{\epsilon_m \epsilon_{m-1} \dots \epsilon_3 \epsilon_2 \epsilon_1 Q = U} \tag{1.36}$$

$$\boldsymbol{Q^{-1} = \epsilon_m \epsilon_{m-1} \dots \epsilon_3 \epsilon_2 \epsilon_1} \tag{1.37}$$

Finally, since ϵ_m matrices are square and nonsingular:

$$\boldsymbol{Q = \epsilon_m^{-1} \epsilon_{m-1}^{-1} \dots \epsilon_3^{-1} \epsilon_2^{-1} \epsilon_1^{-1}} \tag{1.38}$$

According to Equation 1.38, any nonsingular matrix can be expressed as the product of elementary matrices. Therefore, as $\boldsymbol{A_T}$ is nonsingular, the matrix inversion in Equation 1.31 can be written as:

$$\boldsymbol{{A_T}^{-1} = \epsilon_m \epsilon_{m-1} \dots \epsilon_3 \epsilon_2 \epsilon_1} \tag{1.39}$$

and considering

$$\boldsymbol{D} = \boldsymbol{A_T}^{-1}\boldsymbol{A} = \boldsymbol{\epsilon_m \epsilon_{m-1} \ldots \epsilon_3 \epsilon_2 \epsilon_1 A} \tag{1.40}$$

it becomes clear the cutset matrix may be obtained by performing a sequence of elementary row operations on the incidence matrix. The last step is to determine those operations. As it was previously shown, cutset and incidence matrices may be partitioned as:

$$\boldsymbol{D} = \left[\boldsymbol{U_\delta} \;\vdots\; \boldsymbol{D_L}\right] \qquad \boldsymbol{A} = \left[\boldsymbol{A_T} \;\vdots\; \boldsymbol{A_L}\right] \tag{1.41}$$

The elementary operations required are those that convert $\boldsymbol{A_T}$ to $\boldsymbol{U_\delta}$ or – since the size of $\boldsymbol{U_\delta}$ and $\boldsymbol{A_T}$ is n – simply to the identity:

$$\boldsymbol{\epsilon_m \epsilon_{m-1} \ldots \epsilon_3 \epsilon_2 \epsilon_1 A_T} = \boldsymbol{U} \tag{1.42}$$

1.7 CHAPTER REVIEW

- Kirchoff's Current and Voltage laws are applied directly to the geometry of circuits and are thus valid independently of the characteristics of the circuit elements;
- This geometry can be represented in graph form and studied accordingly;
- Independent KVL equations can be obtained from the fundamental loop matrix $\boldsymbol{B}$ and KCL equations from the incidence $\boldsymbol{A}$ and fundamental cutset $\boldsymbol{D}$ matrices;
- A computer can automatically generate the topological matrices with this sequence of operations:
  1) Read the user input, convert it to $\boldsymbol{A_a}$ and eliminate the row corresponding to the reference node;
  2) Obtain the tree by reducing $\boldsymbol{A}$ to a row echelon form;
  3) Perform a sequence of elementary operations on $\boldsymbol{A}$ to convert $\boldsymbol{A_T}$ to $\boldsymbol{U}$ – the resulting matrix is $\boldsymbol{D}$;
  4) Finally, use the relation $\boldsymbol{B_T} = \boldsymbol{D_L}^t$ to obtain $\boldsymbol{B}$.

Chapter 2

State Variable Analysis

Prior to the appearance of the numerical integration substitution in nodal methods, the State Variable Analysis (SVA) – also called State-Space Analysis – was the most common technique. Since the SVA is especially powerful for modeling control systems, it was implemented as the core method in MATLAB Simulink, one of the most proeminent simulation softwares. This results in a seamless integration between power and control model parts when the Simscape Power Systems (SPS) toolbox is used for power systems or power electronics simulations [16].

In the SVA the entire network is initially represented by a set of first order differential equations in state-space form, which are then solved by numerical integration. Since the equations are defined before the discretization process[1], the use of variable step length integration is facilitated as well as step synchronization with the switching instants, eliminating possible spikes in the waveforms due to numerical oscillations [15]. On the other hand, the automatic formulation of the system matrices and equations is very complex and must be done for every topological change, rendering the process slow for a high number of switching components.

[1] If desirable, the state-space equations can be discretized.

The computational load may be reduced by separating the system into constant and frequently switching parts and using voltage and current sources to interface them. Another solution is the pre-calculation of the SVA matrices of every possible topological state. This method however has a significant drawback: the required memory grows exponentially with the number of switches [17]; this is the case, for example, of the PLECS simulation software.

This chapter initially presents an overview of the SVA and some information about the automatic formulation of state equations. Later, due to the fact that OPAL-RT uses SPS as the base of their algorithms, some characteristics of modeling process used by SPS are discussed.

2.1 STATE VARIABLES

Given an electrical network that allows its representation by a system of linearly independent first-order differential equations, a state-space form can be obtained with the proper rearrangement and collection of the equations. State variables describe the complete energy storage state of the network and, thus, can be used to obtain desired outputs. The SVA equations are

$$\begin{aligned} \dot{\boldsymbol{x}} &= \hat{\boldsymbol{A}}\boldsymbol{x} + \hat{\boldsymbol{B}}\boldsymbol{u} \\ \boldsymbol{y} &= \hat{\boldsymbol{C}}\boldsymbol{x} + \hat{\boldsymbol{D}}\boldsymbol{u} \end{aligned} \tag{2.1}$$

where
$\boldsymbol{u}$ is a $m \times 1$ vector containing the m inputs (independent sources)
$\boldsymbol{y}$ is a $p \times 1$ vector containing p outputs (selected voltages/currents)
$\boldsymbol{x}$ is a $n \times 1$ vector containing n independent state variables
$\dot{\boldsymbol{x}}$ contains the derivatives of the state variables
$\hat{\boldsymbol{A}}$, $\hat{\boldsymbol{B}}$, $\hat{\boldsymbol{C}}$ and $\hat{\boldsymbol{D}}$ are constant matrices ($n \times n$, $n \times m$, $p \times n$, and $p \times m$, respectively)

The computer algorithm must build the SVA matrices and solve the equations using a numerical integration method. A description of the usual solution methods is not in the scope of this master's thesis and can be found easily on the literature. Some examples are

the iterative Adams-Moulton predictor-corrector, Runge-Kutta, and matrix exponentials determination methods [8, 14].

A solution to the system exists if the number of state variables is equal to the number of **independent** energy-storage elements. Thus, the easy approach of finding the KCL/KVL equations and selecting every capacitor voltage and inductor current as state variables can be problematic. A loop formed only of capacitors and/or voltage sources results in a state variable being a linear combination of the others. Similarly, a dependent inductor current exists given a cutset formed of inductors and current sources. The simulation program therefore must be able to identify and select only independent elements.

The dependency among branch currents is not always obvious because there may be intervening resistors or capacitors instead of inductors connected directly to a node, which results in the necessity of complex algorithms for the identification.The closed-loop capacitors dependency though may be solved in an easier way with the transform method, described next.

2.1.1 Transform method

The transform method is based on the **node transformation**, a very useful relation among branch voltages and node voltages of which a quick proof will be given next.

Proposition: *The branch voltages* $\boldsymbol{v}$ *of a connected network with* $n+1$ *nodes are related to the node-to-reference voltages* $\boldsymbol{v_r}$ *by*

$$\boldsymbol{v} = \boldsymbol{A}^T \boldsymbol{v_r} \tag{2.2}$$

where

$$\boldsymbol{v_r} = \begin{bmatrix} v_{1r} \\ v_{2r} \\ \vdots \\ v_{nr} \end{bmatrix} \tag{2.3}$$

Proof: Any of the nodes of a network may be chosen as a reference node r. This node's corresponding row is absent from the

reduced incidence matrix $\boldsymbol{A}$. There are only three different ways to connect a branch k of the network:

1) From a node i to the reference node r;

2) From the reference node r to a node i;

3) From a node i to a node j $(i \neq j \neq r)$.

In case 1, there is only one non-zero element in the column of $\boldsymbol{A}$ corresponding to branch k: $a_{ik} = 1$. The k-th scalar equation of (2.2) is this case

$$v_k = v_{ir} \tag{2.4}$$

which can be verified as the correct correlation. Case 2 denotes simply the opposite direction, thus $a_{ik} = -1$ and

$$v_k = -v_{ir} \tag{2.5}$$

For the third case, there will be two non-zero elements in column k: $a_{ik} = 1$ and $a_{jk} = -1$. The k-th scalar equation of (2.2) is:

$$v_k = v_{ir} - v_{jr} \tag{2.6}$$

which is also correct. Therefore, Equation 2.2 is valid for every possibility. ■

The change of state variables from capacitor voltages to charge at nodes is an example of the transform method. It is basically a linear transformation that reduces the total number of state variables. A simple loop of three capacitors is shown in Figure 2.1.

Capacitor voltages are chosen initially as state variables. The equation relating them to the capacitor charges is:

$$\begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} C_1 & 0 & 0 \\ 0 & C_2 & 0 \\ 0 & 0 & C_3 \end{bmatrix} \begin{bmatrix} v_{C_1} \\ v_{C_2} \\ v_{C_3} \end{bmatrix} \tag{2.7}$$

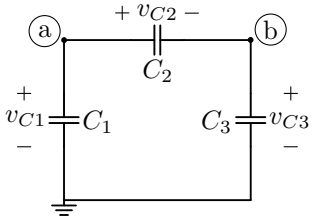


Figure 2.1 – Composite branch representation.

The incidence matrix of the circuit is:

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \tag{2.8}$$

Applying the node transformation discussed previously,

$$\begin{bmatrix} v_{C_1} \\ v_{C_2} \\ v_{C_3} \end{bmatrix} = \boldsymbol{A}^T \begin{bmatrix} v_a \\ v_b \end{bmatrix} \tag{2.9}$$

is obtained. Furthermore, the relation between node and capacitor charges may be written as:

$$\begin{bmatrix} q_a \\ q_b \end{bmatrix} = \boldsymbol{A} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{2.10}$$

The substitution of (2.7) and (2.9) into (2.10) yields

$$\begin{bmatrix} q_a \\ q_b \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} C_1 & 0 & 0 \\ 0 & C_2 & 0 \\ 0 & 0 & C_3 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_a \\ v_b \end{bmatrix} \tag{2.11}$$

Finally:

$$\begin{bmatrix} q_a \\ q_b \end{bmatrix} = \begin{bmatrix} C_1 + C_2 & -C_2 \\ -C_2 & C_2 + C_3 \end{bmatrix} \begin{bmatrix} v_a \\ v_b \end{bmatrix} \tag{2.12}$$

It is evident that the number of state variables was reduced from 3 to 2, in this case. There is an important restriction though for the application of this method: all capacitor subnetworks must be connected to the reference node. When the method is applied to the example circuit shown in Figure 2.2, it generates the singular $[C]$ matrix of Equation 2.13, which means one independent equation is missing. To circumvent this problem, an auxiliary fictitious capacitor (C_{aux}) may be added to connect the subnetwork to the reference node, generating the new $[C]$ matrix of Equation 2.14. However, since this new capacitor must be very small to avoid changing the dynamics of the system in a significant way, a small time-step is required for the integration routine to converge [14].

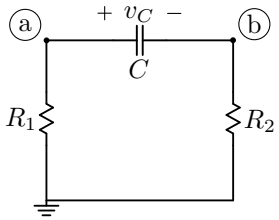


Figure 2.2 – Circuit unsolvable with the transform method.

$$\begin{bmatrix} q_a \\ q_b \end{bmatrix} = \begin{bmatrix} C & -C \\ -C & C \end{bmatrix} \begin{bmatrix} v_a \\ v_b \end{bmatrix} \tag{2.13}$$

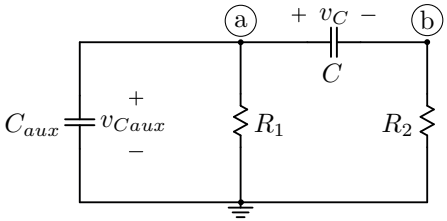


Figure 2.3 – Auxiliary capacitor added to produce a non-singular $[C]$.

$$\begin{bmatrix} q_a \\ q_b \end{bmatrix} = \begin{bmatrix} C + C_{aux} & -C \\ -C & C \end{bmatrix} \begin{bmatrix} v_a \\ v_b \end{bmatrix} \tag{2.14}$$

A more general formulation procedure, which inherently avoids the selection of dependent elements as state variables, although computationally intensive, will be discussed next.

2.2 AUTOMATIC SVA FORMULATION USING TOPOLOGICAL MATRICES

Given an RLC network where no loops consist of only independent voltage sources and no cutsets consist exclusively of independent current sources (any practical circuit), a **proper tree** can be always constructed. This special type of tree contains:

- Every independent voltage source;
- No independent current sources;
- As many capacitors as possible;
- As few inductors as possible.

By using the tree generation method discussed in Section 1.6.1 on an incidence matrix in which the chosen branch order satisfies the conditions of a proper tree, the necessary independent variables of the network can be found. Considering that order of preference for the elements, and using the subscript t to identify twig elements and l for link elements, the vectors of voltages and currents may be written as

$$\boldsymbol{v} \triangleq \begin{bmatrix} \boldsymbol{v}_{Et} \\ \boldsymbol{v}_{Ct} \\ \boldsymbol{v}_{Rt} \\ \boldsymbol{v}_{Lt} \\ \boldsymbol{v}_{Jl} \\ \boldsymbol{v}_{Ll} \\ \boldsymbol{v}_{Rl} \\ \boldsymbol{v}_{Cl} \end{bmatrix} \quad , \quad \boldsymbol{i} \triangleq \begin{bmatrix} \boldsymbol{i}_{Et} \\ \boldsymbol{i}_{Ct} \\ \boldsymbol{i}_{Rt} \\ \boldsymbol{i}_{Lt} \\ \boldsymbol{i}_{Jl} \\ \boldsymbol{i}_{Ll} \\ \boldsymbol{i}_{Rl} \\ \boldsymbol{i}_{Cl} \end{bmatrix} \tag{2.15}$$

where E identifies independent voltage sources and J, independent current sources. The next step is generating the loop matrix from

the proper tree. The general KVL matrix equation of the network has the form [8]:

$$\boldsymbol{B}\boldsymbol{v} = \left[\begin{array}{cccc:cccc} E_t & C_t & R_t & L_t & J_l & L_l & R_l & C_l \\ -\boldsymbol{F}_{11}{}^T & -\boldsymbol{F}_{21}{}^T & -\boldsymbol{F}_{31}{}^T & -\boldsymbol{F}_{41}{}^T & \boldsymbol{U}_{Jl} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ -\boldsymbol{F}_{12}{}^T & -\boldsymbol{F}_{22}{}^T & -\boldsymbol{F}_{32}{}^T & -\boldsymbol{F}_{42}{}^T & \boldsymbol{0} & \boldsymbol{U}_{Ll} & \boldsymbol{0} & \boldsymbol{0} \\ -\boldsymbol{F}_{13}{}^T & -\boldsymbol{F}_{23}{}^T & -\boldsymbol{F}_{33}{}^T & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{U}_{Rl} & \boldsymbol{0} \\ -\boldsymbol{F}_{14}{}^T & -\boldsymbol{F}_{24}{}^T & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{U}_{Cl} \end{array}\right] \boldsymbol{v} = \boldsymbol{0} \tag{2.16}$$

As established before, a proper tree must contain as many capacitors as possible. They have to be left out only if they would otherwise form a loop with other capacitors and voltage sources that are already part of the tree (violating the tree definition). Therefore, the fundamental loop corresponding to a link capacitor C_l cannot contain resistors or inductors, resulting in $-\boldsymbol{F}_{34}{}^T = -\boldsymbol{F}_{44}{}^T = \boldsymbol{0}$. Similarly, a resistor must be in the cotree only if it would otherwise form a loop with capacitors, sources and resistors already chosen as twigs; the fundamental loop constructed with a link resistor R_l therefore contains no inductors, accounting for $-\boldsymbol{F}_{43}{}^T = \boldsymbol{0}$.

Using the relationship $\boldsymbol{B_t} = -\boldsymbol{D_l}^T$ (Equation 1.21), the KCL equations are found:

$$\boldsymbol{D}\boldsymbol{i} = \left[\begin{array}{cccc:cccc} E_t & C_t & R_t & L_t & J_l & L_l & R_l & C_l \\ \boldsymbol{U}_{Et} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{F}_{11} & \boldsymbol{F}_{12} & \boldsymbol{F}_{13} & \boldsymbol{F}_{14} \\ \boldsymbol{0} & \boldsymbol{U}_{Ct} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{F}_{21} & \boldsymbol{F}_{22} & \boldsymbol{F}_{23} & \boldsymbol{F}_{24} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{U}_{Rt} & \boldsymbol{0} & \boldsymbol{F}_{31} & \boldsymbol{F}_{32} & \boldsymbol{F}_{33} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{U}_{Lt} & \boldsymbol{F}_{41} & \boldsymbol{F}_{42} & \boldsymbol{0} & \boldsymbol{0} \end{array}\right] \boldsymbol{i} = \boldsymbol{0} \tag{2.17}$$

By algebraically manipulating Equations 2.16 and 2.17 and using the elements' characteristic equations (e.g. $\boldsymbol{v}_{Rt} = \boldsymbol{R_t}\boldsymbol{i}_{Rt}$) with $\boldsymbol{v}_{Ct}$ and $\boldsymbol{i}_{Ll}$ chosen as state variables, it is possible to obtain the explicit state space form [8]. The procedure is algebraically intensive and yields the following equation:

$$\hat{\boldsymbol{M}}^{(0)} \frac{d}{dt}\begin{bmatrix}\boldsymbol{v}_{Ct} \\ \boldsymbol{i}_{Ll}\end{bmatrix} = \hat{\boldsymbol{A}}^{(0)}\begin{bmatrix}\boldsymbol{v}_{Et} \\ \boldsymbol{i}_{Jl}\end{bmatrix} + \hat{\boldsymbol{B}}^{(0)}\begin{bmatrix}\boldsymbol{v}_{Et} \\ \boldsymbol{i}_{Ll}\end{bmatrix} + \hat{\boldsymbol{B}}_{\boldsymbol{1}}{}^{(0)} \frac{d}{dt}\begin{bmatrix}\boldsymbol{v}_{Ct} \\ \boldsymbol{i}_{Ll}\end{bmatrix} \tag{2.18}$$

where

$$\hat{\boldsymbol{M}}^{(0)} = \begin{bmatrix} \boldsymbol{C_t} + \boldsymbol{F}_{24}\boldsymbol{C_l}\boldsymbol{F}_{24}^T & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{L_l} - \boldsymbol{F}_{42}{}^T\boldsymbol{L}_{tl} - \boldsymbol{L}_{lt}\boldsymbol{F}_{42} + \boldsymbol{F}_{42}{}^T\boldsymbol{L}_{tt}\boldsymbol{F}_{42} \end{bmatrix} \tag{2.19}$$

$$\hat{\boldsymbol{A}}^{(0)} = \begin{bmatrix} -\boldsymbol{F}_{23}\boldsymbol{R}^{-1}\boldsymbol{F}_{23}{}^T & -\boldsymbol{F}_{22} + \boldsymbol{F}_{23}\boldsymbol{R}^{-1}\boldsymbol{F}_{33}{}^T\boldsymbol{R_t}\boldsymbol{F}_{32} \\ \boldsymbol{F}_{22}{}^T - \boldsymbol{F}_{32}{}^T\boldsymbol{G}^{-1}\boldsymbol{F}_{33}\boldsymbol{G_l}\boldsymbol{F}_{23}{}^T & -\boldsymbol{F}_{32}{}^T\boldsymbol{G}^{-1}\boldsymbol{F}_{32} \end{bmatrix} \tag{2.20}$$

$$\hat{\boldsymbol{B}}^{(0)} = \begin{bmatrix} -\boldsymbol{F}_{23}\boldsymbol{R}^{-1}\boldsymbol{F}_{13} & -\boldsymbol{F}_{21} + \boldsymbol{F}_{23}\boldsymbol{R}^{-1}\boldsymbol{F}_{33}{}^T\boldsymbol{R_t}\boldsymbol{F}_{31} \\ \boldsymbol{F}_{12}{}^T - \boldsymbol{F}_{32}{}^T\boldsymbol{G}^{-1}\boldsymbol{F}_{33}\boldsymbol{G_l}\boldsymbol{F}_{13}{}^T & -\boldsymbol{F}_{32}{}^T\boldsymbol{G}^{-1}\boldsymbol{F}_{31} \end{bmatrix} \tag{2.21}$$

$$\hat{\boldsymbol{B}_{\boldsymbol{1}}}^{(0)} = \begin{bmatrix} -\boldsymbol{F}_{24}\boldsymbol{C_l}\boldsymbol{F}_{14}^T & \boldsymbol{0} \\ \boldsymbol{0} & -\boldsymbol{F}_{42}{}^T\boldsymbol{L}_{tt}\boldsymbol{F}_{41} + \boldsymbol{L}_{lt}\boldsymbol{F}_{41} \end{bmatrix} \tag{2.22}$$

and

$$\boldsymbol{G} = \boldsymbol{G_t} + \boldsymbol{F}_{33}\boldsymbol{G_l}\boldsymbol{F}_{33}{}^T \tag{2.23}$$

$$\boldsymbol{R} = \boldsymbol{R_l} + \boldsymbol{F}_{33}{}^T\boldsymbol{R_t}\boldsymbol{F}_{33} \tag{2.24}$$

The multiplication of both sides of Equation 2.18 by $[\hat{\boldsymbol{M}}^{(0)}]^{-1}$ yields

$$\dot{\tilde{\boldsymbol{x}}} = \hat{\boldsymbol{A}}\tilde{\boldsymbol{x}} + \hat{\boldsymbol{B}}\boldsymbol{u} + \hat{\boldsymbol{B}}_{\boldsymbol{1}}\dot{\boldsymbol{u}} \tag{2.25}$$

with $\tilde{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{v}_{Ct} & \boldsymbol{i}_{Ll} \end{bmatrix}^T$, $\boldsymbol{u} = \begin{bmatrix} \boldsymbol{v}_{Et} & \boldsymbol{i}_{Jl} \end{bmatrix}^T$, and

$$\begin{aligned} \hat{\boldsymbol{A}} &= [\hat{\boldsymbol{M}}^{(0)}]^{-1}\hat{\boldsymbol{A}}^{(0)} \\ \hat{\boldsymbol{B}} &= [\hat{\boldsymbol{M}}^{(0)}]^{-1}\hat{\boldsymbol{B}}^{(0)} \\ \hat{\boldsymbol{B}}_{\boldsymbol{1}} &= [\hat{\boldsymbol{M}}^{(0)}]^{-1}\hat{\boldsymbol{B}_{\boldsymbol{1}}}^{(0)} \end{aligned} \tag{2.26}$$

Finally, by performing the following change of variables

$$\boldsymbol{x} = \tilde{\boldsymbol{x}} - \hat{\boldsymbol{B}}_{\boldsymbol{1}}\boldsymbol{u} \tag{2.27}$$

the usual form (Equation 2.28) is obtained. The formulation of the $\hat{\boldsymbol{C}}$ and $\hat{\boldsymbol{D}}$ matrices is trivial since they represent a combination of state-variables and inputs to return the desired outputs.

$$\dot{\boldsymbol{x}} = \hat{\boldsymbol{A}}\boldsymbol{x} + (\hat{\boldsymbol{B}} + \hat{\boldsymbol{A}}\hat{\boldsymbol{B}}_{\boldsymbol{1}})\boldsymbol{u} \tag{2.28}$$

This formulation process is therefore powerful but involves many matrix operations, and can be very demanding for large switching networks since different topological states have different sets of state matrices which must be recalculated after a switching event. In the next section it will be shown the modeling strategy adopted in SPS and how the algorithms adapt the formulation process to deal with switchings.

2.3 SIMSCAPE POWER SYSTEMS MODELING

The algorithms in an SPS simulation separate the linear elements from the nonlinear ones into two different parts. A state-space representation of the linear components is calculated, and nonlinear components are modeled in the linear part as current sources. The values of the current sources are determined by an external model, which uses the nonlinear component voltages as input. This can be viewed as a feedback loop between the outputs and inputs of the state-space model [16]. Figure 2.4 depicts the methodology.

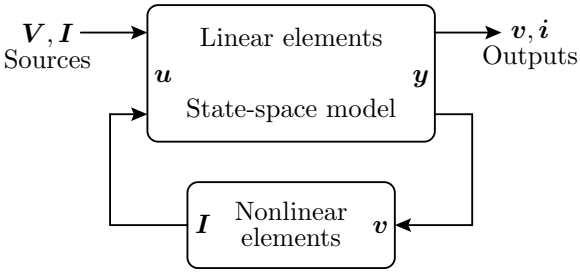


Figure 2.4 – Simscape Power Systems SVA modelling strategy.

Depending on the solving method chosen in the configurations – continuous or discrete – and the component parameters,

different switch models, and consequently formulation strategies, are used. In the continuous method, which is appropriate for using a variable time-step, if the switching component has an inductance value the calculated current of the nonlinear part sets the value of the linear current source, following the feedback method of Figure 2.4. This is exemplified in the diode representation of Figure 2.5.

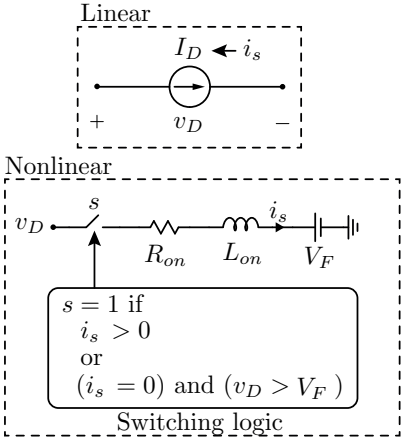


Figure 2.5 – SPS diode model interpretation.

If $L_{on} = 0$, once a switching event is detected the topology is changed and SVA matrices are recalculated[2]. Since the formulation method described in the previous section is very demanding, it cannot be used every time a switch changes its state. An implemented adaptation, which was first shown in [18], reduces significantly the computational effort and is detailed next.

Initially, the proper tree method is used to generate the state-space matrices (Equation 2.29) of the whole linear system.

$$\begin{aligned} \dot{\boldsymbol{x}} &= \hat{\boldsymbol{A}}_{\boldsymbol{0}}\boldsymbol{x} + \hat{\boldsymbol{B}}_{\boldsymbol{0}}\boldsymbol{u} \\ \boldsymbol{y} &= \hat{\boldsymbol{C}}_{\boldsymbol{0}}\boldsymbol{x} + \hat{\boldsymbol{D}}_{\boldsymbol{0}}\boldsymbol{u} \end{aligned} \tag{2.29}$$

A new state-space representation is then created, as shown

[2] This is always valid if a discrete solving method in chosen because L_{on} is then automatically set to zero.

in Figure 2.6. The pictured variables are described in (2.30) and (2.31), and $\boldsymbol{G}_{sw}$ contains the switch conductances.

$$\boldsymbol{u}_{sr} = \begin{bmatrix} \mathbf{0} \\ \hline \boldsymbol{u}_{in} \end{bmatrix} \quad \boldsymbol{u}_{fb} = \begin{bmatrix} \boldsymbol{i}_{sw} \\ \hline \mathbf{0} \end{bmatrix} \quad \boldsymbol{u} = \begin{bmatrix} \boldsymbol{i}_{sw} \\ \hline \boldsymbol{u}_{in} \end{bmatrix} \quad \boldsymbol{y} = \begin{bmatrix} \boldsymbol{v}_{sw} \\ \hline \boldsymbol{y}_{meas} \end{bmatrix} \tag{2.30}$$

$$\boldsymbol{K}_{sw} = \begin{bmatrix} \boldsymbol{G}_{sw} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{2.31}$$

where $\boldsymbol{u}_{sr}$ is the vector of sources and $\boldsymbol{u}_{fb}$ is the feedback vector (switch currents).

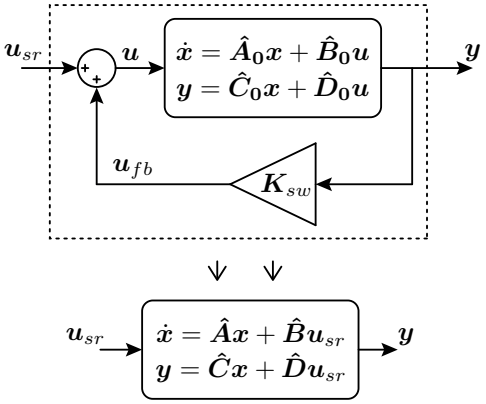


Figure 2.6 – SPS state equations recalculation strategy.

It can be seen from the block diagram that

$$\boldsymbol{u} = \boldsymbol{u}_{sr} + \boldsymbol{u}_{fb} \tag{2.32}$$

$$\boldsymbol{u} = \boldsymbol{u}_{sr} + \boldsymbol{K}_{sw}\boldsymbol{y} \tag{2.33}$$

Substituting (2.33) in the second line of (2.29):

$$\boldsymbol{y} = \hat{\boldsymbol{C}}_{\mathbf{0}}\boldsymbol{x} + \hat{\boldsymbol{D}}_{\mathbf{0}}\boldsymbol{u}_{sr} + \hat{\boldsymbol{D}}_{\mathbf{0}}\boldsymbol{K}_{sw}\boldsymbol{y} \tag{2.34}$$

$$\boldsymbol{y} - \hat{\boldsymbol{D}}_{\mathbf{0}}\boldsymbol{K}_{sw}\boldsymbol{y} = \hat{\boldsymbol{C}}_{\mathbf{0}}\boldsymbol{x} + \hat{\boldsymbol{D}}_{\mathbf{0}}\boldsymbol{u}_{sr} \tag{2.35}$$

Lastly, by defining

$$\boldsymbol{D}_{\boldsymbol{i}} \triangleq \left(\boldsymbol{U} - \hat{\boldsymbol{D}}_{\mathbf{0}}\boldsymbol{K}_{sw}\right)^{-1} \tag{2.36}$$

the new output equation can be obtained:

$$\boldsymbol{y} = \hat{\boldsymbol{C}}\boldsymbol{x} + \hat{\boldsymbol{D}}\boldsymbol{u}_{sr} \tag{2.37}$$

where

$$\hat{\boldsymbol{C}} = \boldsymbol{D_i}\hat{\boldsymbol{C}}_{\boldsymbol{0}} \tag{2.38}$$

$$\hat{\boldsymbol{D}} = \boldsymbol{D_i}\hat{\boldsymbol{D}}_{\boldsymbol{0}} \tag{2.39}$$

The next step is the determination of the state equation. Substituting (2.33) in the first line of (2.29):

$$\dot{\boldsymbol{x}} = \hat{\boldsymbol{A}}_{\boldsymbol{0}}\boldsymbol{x} + \hat{\boldsymbol{B}}_{\boldsymbol{0}}\big(\boldsymbol{u}_{sr} + \boldsymbol{K}_{sw}\boldsymbol{y}\big) \tag{2.40}$$

Then, substituting $\boldsymbol{y}$ with (2.37):

$$\dot{\boldsymbol{x}} = \hat{\boldsymbol{A}}_{\boldsymbol{0}}\boldsymbol{x} + \hat{\boldsymbol{B}}_{\boldsymbol{0}}\boldsymbol{u}_{sr} + \hat{\boldsymbol{B}}_{\boldsymbol{0}}\boldsymbol{K}_{sw}\big(\hat{\boldsymbol{C}}\boldsymbol{x} + \hat{\boldsymbol{D}}\boldsymbol{u}_{sr}\big) \tag{2.41}$$

Finally, the following definition

$$\boldsymbol{B_i} \triangleq \hat{\boldsymbol{B}}_{\boldsymbol{0}}\boldsymbol{K}_{sw} \tag{2.42}$$

along with the reorganization of (2.41) yields

$$\dot{\boldsymbol{x}} = \big(\hat{\boldsymbol{A}}_{\boldsymbol{0}} + \boldsymbol{B_i}\hat{\boldsymbol{C}}\big)\boldsymbol{x} + \big(\hat{\boldsymbol{B}}_{\boldsymbol{0}} + \boldsymbol{B_i}\hat{\boldsymbol{D}}\big)\boldsymbol{u}_{sr} \tag{2.43}$$

and the new state equation is thus found

$$\dot{\boldsymbol{x}} = \hat{\boldsymbol{A}}\boldsymbol{x} + \hat{\boldsymbol{B}}\boldsymbol{u}_{sr} \tag{2.44}$$

where

$$\hat{\boldsymbol{A}} = \hat{\boldsymbol{A}}_{\boldsymbol{0}} + \boldsymbol{B_i}\hat{\boldsymbol{C}} \tag{2.45}$$

$$\hat{\boldsymbol{B}} = \hat{\boldsymbol{B}}_{\boldsymbol{0}} + \boldsymbol{B_i}\hat{\boldsymbol{D}} \tag{2.46}$$

The extra steps needed for the reformulation of SVA equations due to switching are therefore simple:

1) Update $\boldsymbol{K}_{sw}$

2) Calculate $\boldsymbol{D_i}$ and $\boldsymbol{B_i}$

3) Calculate $\hat{\boldsymbol{C}}$ and $\hat{\boldsymbol{D}}$

4) Calculate $\hat{\boldsymbol{A}}$ and $\hat{\boldsymbol{B}}$

Since nonlinear components and the conductance modeled switches are fed back as current sources at some point, they cannot be placed in series with each other or current sources. Snubber RC circuits can be used to overcome this problem, but the values must be carefully chosen to avoid significant parasitic high-frequency oscillations with the switch RL model, otherwise a reduction in the time-step will ensue, degrading the speed of the simulation.

2.4 CHAPTER REVIEW

- A set of N first-order differential equations can be used to describe an LTI system. The solution of the system of equations depend on the existence of N independent initial conditions, which are provided by energy storage elements in the case of electric circuits. The set of equations can be manipulated into a state-space representation.

- The state-space form contains a vector of derivatives. Since the numerical solution involves a discretization process, a change of time-step value can be done without overhead.

- Selecting every capacitor voltage and inductor current as state variables can be problematic for there may be dependent relations. Algorithms can be created for the identification and elimination of dependent elements. A more direct although computationally intensive approach has been presented. Furthermore the formulations are only valid for analysis methods based on linear RLC network representations.

- In SPS a linear state-space representation is formulated with the proper tree method. Nonlinear V-I characteristics that include an inductance are calculated separately and introduced in the state-space equations as current source inputs. Switches can

be modeled as varying resistance values if the SVA equations are recalculated at each switching event, and in this case a modification to the original equations is done in order to reduce the needed computation.

Chapter 3

Nodal Analysis

The nodal formulation is a derivation from the KCL. Originally, it resulted in integro-differential equations that were solved by the Laplace transform, and therefore was significantly behind SVA based methods, especially for calculating large-scale networks. Later, the development of associated discrete models (also called companion models) turned the nodal analysis into the most efficient method for the simulation of large power systems [8]. The discretized element models can be constructed through the numerical substitution of integrals obtained from the elements' V-I characteristics.

The numerical solution normally depends on the history (last known values) of the variables and, since a digital system is able to store only "snapshots" of that history at definite time-steps, instability due to truncation errors or poorly damped responses may exist depending on the integration method used [19]. For example, the Electromagnetic Transients Program (EMTP) [20] – probably the most used power systems simulation software of the past decades and responsible for the popularization of nodal methods – uses the trapezoidal integration, which is A-stable[1], but can generate numeric oscillations.

[1] Stable if the analytical solution also is.

In this chapter, the nodal formulation for resistive networks, which is appropriate for the use of associated discrete models, is shown first. Next, the fundamental characteristics of the EMTP method are discussed. Finally, the Modified Nodal Analysis (MNA) [21], an evolution of the classic nodal analysis that solves some of its limitations is presented.

3.1 BASIC NODAL METHOD FORMULATION

The nodal equations formulation process for linear resistive networks will be shown next, based on [8]. A network branch $-B-$ may be represented by a composite form, as depicted in Figure 3.1, where the element k_B may be either a linear resistor or a voltage controlled current source.

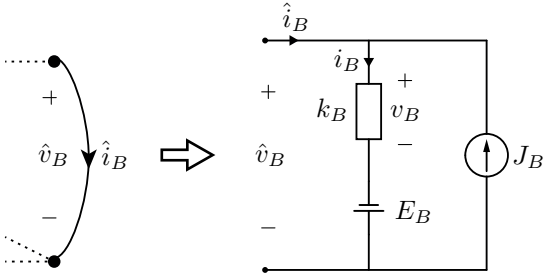


Figure 3.1 – Composite branch representation.

Consider a network which consists of b composite branches numbered from 1 to b and $n+1$ nodes from 0 to n, with node 0 used as reference. The vectors of node-to-reference voltages and currents are defined as

$$\hat{\boldsymbol{v}} \triangleq \begin{bmatrix} \hat{v}_1 \\ \hat{v}_2 \\ \vdots \\ \hat{v}_b \end{bmatrix}, \quad \boldsymbol{v} \triangleq \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_b \end{bmatrix}, \quad \boldsymbol{E} \triangleq \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_b \end{bmatrix} \tag{3.1}$$

$$\hat{\boldsymbol{i}} \triangleq \begin{bmatrix} \hat{i}_1 \\ \hat{i}_2 \\ \vdots \\ \hat{i}_b \end{bmatrix}, \quad \boldsymbol{i} \triangleq \begin{bmatrix} i_1 \\ i_2 \\ \vdots \\ i_b \end{bmatrix}, \quad \boldsymbol{J} \triangleq \begin{bmatrix} J_1 \\ J_2 \\ \vdots \\ J_b \end{bmatrix} \tag{3.2}$$

It follows that

$$\hat{\boldsymbol{v}} = \boldsymbol{v} - \boldsymbol{E} \tag{3.3}$$

$$\hat{\boldsymbol{i}} = \boldsymbol{i} - \boldsymbol{J} \tag{3.4}$$

The independent KCL equations of this network are, according to Equation 1.8,

$$\boldsymbol{A}\hat{\boldsymbol{i}} = \boldsymbol{0} \tag{3.5}$$

Substituting (3.4) into (3.5) yields

$$\boldsymbol{A}\boldsymbol{i} = \boldsymbol{A}\boldsymbol{J} \tag{3.6}$$

Depending on the two-terminal element k present in a branch, its V-I characteristics may be defined either by Equation 3.7 (if a resistor) or by Equation 3.8 (if it is a current source controlled by a v_j voltage).

$$i_k = \frac{v_k}{R_k} \tag{3.7}$$

$$i_k = g_{kj} v_j \tag{3.8}$$

The characteristics of all branches of the network can therefore be expressed by

$$\begin{bmatrix} i_1 \\ i_2 \\ \vdots \\ i_b \end{bmatrix} = \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1b} \\ y_{21} & y_{22} & \cdots & y_{2b} \\ \vdots & \vdots & \ddots & \vdots \\ y_{21} & y_{22} & \cdots & y_{bb} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_b \end{bmatrix} \tag{3.9}$$

or, in a compact form,

$$\boldsymbol{i} = \boldsymbol{Y}_b \boldsymbol{v} \tag{3.10}$$

where

$$y_{k\alpha} = \begin{cases} 0, & \alpha \neq k \\ \dfrac{1}{R_k}, & \alpha = k \end{cases} \qquad \text{or} \qquad y_{k\alpha} = \begin{cases} 0, & \alpha \neq j \\ g_{kj}, & \alpha = j \end{cases} \tag{3.11}$$

Substituting (3.3) and (3.6) into (3.10) yields

$$\boldsymbol{A}\boldsymbol{Y_b}\hat{\boldsymbol{v}} = \boldsymbol{A}(\boldsymbol{J} - \boldsymbol{Y_b}\boldsymbol{E}) \tag{3.12}$$

$\boldsymbol{Y}_b$ is called the **branch-admittance matrix**. By applying the node transformation presented in Section 2.1.1 to (3.12), a node-to-reference format can be obtained:

$$\hat{\boldsymbol{v}} = \boldsymbol{A}^T\boldsymbol{v_r} \tag{3.13}$$

$$(\boldsymbol{A}\boldsymbol{Y_b}\boldsymbol{A}^T)\boldsymbol{v_r} = \boldsymbol{A}(\boldsymbol{J} - \boldsymbol{Y_b}\boldsymbol{E}) \tag{3.14}$$

Defining the **node-admittance matrix $\boldsymbol{Y_r}$**

$$\boldsymbol{Y_r} \triangleq \boldsymbol{A}\boldsymbol{Y_b}\boldsymbol{A}^T \tag{3.15}$$

and the **equivalent nodal current source vector $\boldsymbol{J_r}$**

$$\boldsymbol{J_r} \triangleq \boldsymbol{A}(\boldsymbol{J} - \boldsymbol{Y_b}\boldsymbol{E}) \tag{3.16}$$

the **nodal equation** – as it is commonly called – is obtained:

$$\boxed{\boldsymbol{Y_r}\boldsymbol{v_r} = \boldsymbol{J_r}} \tag{3.17}$$

The network solution is completely and uniquely described by (3.17). Every node voltage may be calculated with the aid of ${Y_r}^{-1}$, however Gaussian elimination or LU factorization are usually performed to avoid the matrix inversion. A description of the solution algorithms can be found in [8] and many other works. In the next section, a method which derives suitable elements for the population of the nodal equation matrices is shown.

3.2 THE ELECTROMAGNETIC TRANSIENTS PROGRAM

The original EMTP method, published in 1969 by Hermann W. Dommel [20], was developed with focus on the simulation of transients in power systems. The involved element modeling combines the method of characteristics and the trapezoidal rule. Each branch element V-I characteristic equation is individually discretized and then the resulting difference equations combined into a single system-wide nodal equation. The discretization of the basic power system elements is presented next.

3.2.1 Discretization of system elements

In Dommel's method the continuous integrals of the differential equations which describe the voltage/current relationship of the elements are substituted by a numerical form based on the trapezoidal rule. This form of approximation is illustrated on Figure 3.2.

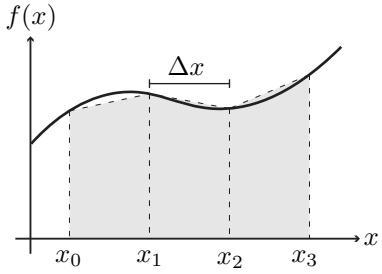


Figure 3.2 – The trapezoidal integration approximation.

$$A_T = \Delta x \left[\frac{f(x_{i-1}) + f(x_i)}{2} \right] \tag{3.18}$$

The integral of a function $f(x)$ limited from $x = a$ to $x = b$ can be approximated by the sum of the area of n trapezoids in the following manner [22]:

$$\int_a^b f(x)dx \approx \frac{\Delta x}{2}[f(x_0) + 2f(x_1) + ... + 2f(x_{n-1}) + f(x_n)] \tag{3.19}$$

where $\Delta x = (b-a)/n$ and $x_k = a + k\Delta x$.

Discretization of a resistance:

The characteristic equation of a resistive element (Figure 3.3) is algebraic, thus the calculation is simple and the discretized result is direct.

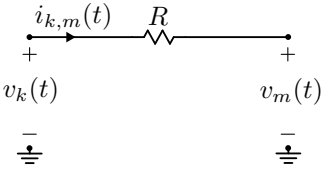


Figure 3.3 – Resistance discretization.

$$\boxed{i_{k,m}(t) = \frac{v_k(t) - v_m(t)}{R}} \tag{3.20}$$

Very small resistor values mean very large $1/R$ elements present in the nodal-admittance matrix. This can result in poor solution conditioning, considering the finite numerical precision. Some computer programs have special treatments for admittance values that cross certain thresholds to prevent such problems.

Discretization of an inductance:

The voltage across an inductor at the time t is given by:

$$v_L(t) = L\frac{di_L(t)}{dt} \tag{3.21}$$

Therefore, for the case represented in Figure 3.4a:

$$v_k(t) - v_m(t) = L\frac{di_{k,m}(t)}{dt} \tag{3.22}$$

Solving for the current $i_{k,m}(t)$:

$$i_{k,m}(t) = i_{k,m}(t - \Delta t) + \frac{1}{L}\int_{t-\Delta t}^{t} \Big[v_k(t) - v_m(t)\Big]dt \tag{3.23}$$

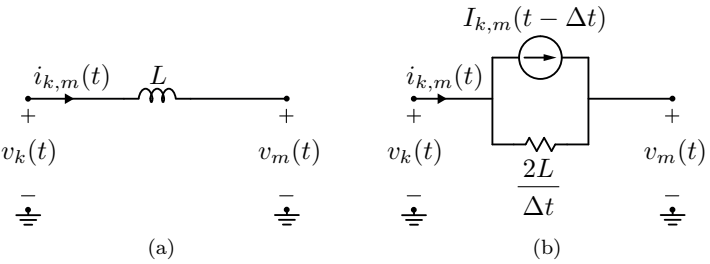


Figure 3.4 – Inductance discretization.

The continuous integral is then substituted by the the trapezoidal rule discretized form:

$$\begin{aligned} i_{k,m}(t) &= i_{k,m}(t-\Delta t) \\ &+ \frac{\Delta t}{2L}\Big[v_k(t) - v_m(t) + v_k(t-\Delta t) - v_m(t-\Delta t)\Big] \end{aligned} \tag{3.24}$$

Finally, reorganizing the elements according to the time argument:

$$\boxed{i_{k,m}(t) = \frac{\Delta t}{2L}\Big[v_k(t) - v_m(t)\Big] + I_{k,m}(t-\Delta t)} \tag{3.25}$$

where

$$I_{k,m}(t-\Delta t) = i_{k,m}(t-\Delta t) + \frac{\Delta t}{2L}\Big[v_k(t-\Delta t) - v_m(t-\Delta t)\Big] \tag{3.26}$$

is the portion that contains the history of the element, i.e., the values at the previous time step. The equivalent circuit representation of the final difference equation is illustrated in Figure 3.4b.

Discretization of a capacitance:

The differential equation for the capacitance of Figure 3.5a is:

$$i_{k,m}(t) = C\frac{d[v_k(t) - v_m(t)]}{dt} \tag{3.27}$$

or

$$v_k(t) - v_m(t) = \frac{1}{C}\int_{t-\Delta t}^{t} i_{k,m}(t)dt + v_k(t-\Delta t) - v_m(t-\Delta t) \tag{3.28}$$

After the integration by the trapezoidal rule:

$$v_k(t) - v_m(t) = \frac{\Delta t}{2C}[i_{k,m}(t) + i_{k,m}(t-\Delta t)]dt + v_k(t-\Delta t) - v_m(t-\Delta t) \tag{3.29}$$

$$\boxed{i_{k,m}(t) = \frac{2C}{\Delta t}[v_k(t) - v_m(t)] + I_{k,m}(t-\Delta t)} \tag{3.30}$$

where

$$I_{k,m}(t-\Delta t) = -i_{k,m}(t-\Delta t) - \frac{2C}{\Delta t}[v_k(t-\Delta t) - v_m(t-\Delta t)] \tag{3.31}$$

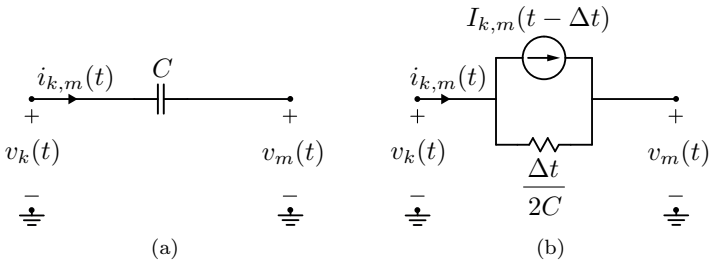


Figure 3.5 – Capacitance discretization.

The equivalent circuit is illustrated in Figure 3.5b

Notice that the value of L and C discretized models depend on the time-step value. A change in the step length therefore demands the recalculation of the admittance matrix, adding significant overhead to the method. For that reason, nodal analysis based simulation softwares usually work with fixed time-step values.

3.2.2 Distributed parameters line (DPL)

The parameters of a frequency independent lossless uniform transmission line – inductance (L') and capacitance (C') per unit length – are distributed evenly along the line. An arbitrary transmission line section is represented in Figure 3.6.

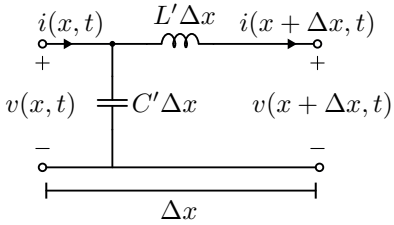


Figure 3.6 – Distributed parameters line section.

Applying KCL and KVL yields:

$$v(x+\Delta x,t) = v(x,t) - L'\Delta x \frac{\delta i(x+\Delta x,t)}{\delta t} \tag{3.32}$$

$$i(x,t) = i(x+\Delta x,t) + C'\Delta x \frac{\delta v(x,t)}{\delta t} \tag{3.33}$$

Reorganizing:

$$\frac{v(x+\Delta x,t) - v(x,t)}{\Delta x} = -L' \frac{\delta i(x+\Delta x,t)}{\delta t} \tag{3.34}$$

$$\frac{i(x+\Delta x,t) - i(x,t)}{\Delta x} = -C' \frac{\delta v(x,t)}{\delta t} \tag{3.35}$$

Taking the limit $\Delta x \to 0$, i.e., for an infinitesimal line section of δx length:

$$\frac{\delta v(x,t)}{\delta x} = -L' \frac{\delta i(x,t)}{\delta t} \tag{3.36}$$

$$\frac{\delta i(x,t)}{\delta x} = -C' \frac{\delta v(x,t)}{\delta t} \tag{3.37}$$

The general solutions are [15]:

$$i(x,t) = f_1(x-at) + f_2(x+at) \tag{3.38}$$

$$v(x,t) = Z f_1(x-at) - Z f_2(x+at) \tag{3.39}$$

where $Z = \sqrt{L'/C'}$ is the characteristic impedance and $a = 1/\sqrt{L'/C'}$ is the velocity of propagation of a wave along the line.

The $f_1(x-at)$ and $f_2(x+at)$ components are physically interpreted as waves traveling in opposite directions. After the multiplication of (3.38) by Z and the addition of the result to (3.39), $f_2(x+at)$ is eliminated, resulting in:

$$v(x,t)+Zi(x,t)=2Zf_1(x-at) \tag{3.40}$$

If an observer is traveling with the wave along the transmission line, $(x-at)$ is constant, and therefore the left hand side of (3.40) is also constant. Considering d the distance between terminals m and k, the time taken for a wave to travel between the terminals is

$$\tau=\frac{d}{a}=d\sqrt{L'C'} \tag{3.41}$$

If the wave departures from terminal m at the time $t-\tau$ and arrives at k at the time t,

$$v_m(t-\tau)+Zi_{m,k}(t-\tau)=v_k(t)+Zi_{m,k}(t)=v_k(t)+Z(-i_{k,m})(t) \tag{3.42}$$

Hence:

$$i_{k,m}(t)=\frac{1}{Z}v_k(t)\underbrace{-\frac{1}{Z}v_m(t-\tau)-i_{m,k}(t-\tau)}_{I_k(t-\tau)} \tag{3.43}$$

By analogy:

$$i_{m,k}(t)=\frac{1}{Z}v_m(t)\underbrace{-\frac{1}{Z}v_k(t-\tau)-i_{k,m}(t-\tau)}_{I_m(t-\tau)} \tag{3.44}$$

Figure 3.7 illustrates the equivalent circuit of the lossless line, in which the terminals are not directly connected. A ring buffer stores a number of I_k/I_m values that depends on τ.

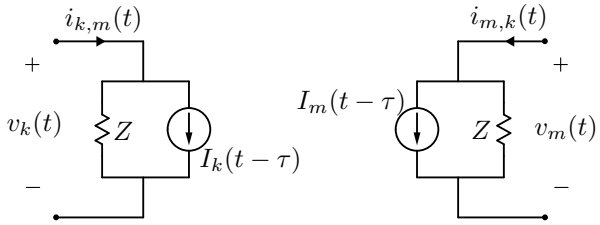


Figure 3.7 – Distributed parameters line dual Norton model.

3.2.3 Oscillations due to the trapezoidal integration

Oscillations are triggered when there are cutsets with only current sources and inductors or loops with voltage sources and capacitors. Consider a voltage source E that is connected in parallel with a capacitor C when an ideal switch is closed. The current through the capacitor is:

$$i_C(t) = C\frac{dv_C}{dt} \tag{3.45}$$

After the trapezoidal integration:

$$v_C(t+\Delta t) = v_C(t) + \frac{\Delta t}{2C}\Big[i_C(t) + i_C(t+\Delta t)\Big] \tag{3.46}$$

Rearranging:

$$i_C(t+\Delta t) = -i_C(t) + \frac{2C}{\Delta t}\Big[v_C(t+\Delta t) - v_C(t)\Big] \tag{3.47}$$

If $v_C(t) = 0$, $i_C(t) = 0$ and the switch is closed at $t+\Delta t$,

$$i_C(t+\Delta t) = -(0) + 2C\Big[(E) - (0)\Big] = \frac{2CE}{\Delta t} \tag{3.48}$$

In the next and subsequent steps, $v_C = E$

$$i_C(t+2\Delta t) = -\left(\frac{2CE}{\Delta t}\right) + 2C\Big[(E) - (E)\Big] = -\frac{2CE}{\Delta t} \tag{3.49}$$

A current oscillation defined by $i_C(t+\Delta t) = -i_C(t)$ will remain unless damped. The EMTP method considers switches as

ideal, and thus compensation methods such as interpolation or the use of different integration methods after switching events are required to mitigate the problem. In modern simulation programs, snubbers and switch resistances are used to damp the oscillations [23].

3.3 MODIFIED NODAL ANALYSIS

Two key disadvantages of the classic nodal/EMTP method are: 1) every branch element must have an admittance representation, which disallows, for example, ungrounded ideal voltage sources and current-controlled elements, and 2) the absence of branch currents as outputs of the nodal equation introduces extra inconvenience and numerical errors for their acquisition. The Modified Nodal Analysis (MNA) removes those limitations in a simple and straightforward way while maintaining the advantages [24]. This refined method is the basis of many programs, including SPICE [25], Multisim [26] and EMTP-RV (in this case a further refined method called modified-augmented nodal analysis), the modern version of the EMTP [24].

In the MNA, the vector of node voltages $\boldsymbol{v_r}$ is expanded to include voltage source currents, controlling currents, and any branch current desired as an output. The equivalent nodal current source vector $\boldsymbol{J_r}$ is also expanded, and can include ideal voltage sources. The resulting expanded matrix equation is:

$$\underbrace{\begin{bmatrix} \boldsymbol{Y_r} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{bmatrix}}_{\boldsymbol{Y}} \underbrace{\begin{bmatrix} \boldsymbol{v_r} \\ \boldsymbol{I} \end{bmatrix}}_{\boldsymbol{V}} = \underbrace{\begin{bmatrix} \boldsymbol{J_r} \\ \boldsymbol{F} \end{bmatrix}}_{\boldsymbol{J}} \tag{3.50}$$

It is convenient to analyze contributions to the formation of (3.50) by each element separately. Given a "general node", depicted in Figure 3.8, a set of predefined "stamps" can be used to construct the MNA equation. The stamps were derived on [21] with the Backward Euler integration method (Equation 3.51) and are presented in Tables 3.1 through 3.5.

$$\begin{aligned} i_C(t) &= \frac{C}{\Delta t}\left[v_C(t) - v_{cp}\right] \\ v_L(t) &= \frac{L}{\Delta t}\left[i_L(t) - i_{lp}\right] \end{aligned} \tag{3.51}$$

where Δt is the time-step and v_{cp} / i_{lp} are the previous time-step values. Controlled voltage/current sources and other elements such as operational amplifiers, transformers, gyrators, etc., have their own stamps. A comprehensive list can be found on [13].

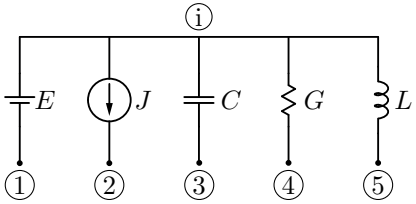


Figure 3.8 – General node.

	v_i	v_1	$\boldsymbol{B}$	$\boldsymbol{J}$
i			$+1$	
1			-1	
$\boldsymbol{C}$	$+1$	-1		E

Current is output

Table 3.1 – Voltage source element E stamp.

				$\boldsymbol{J}$
i				$-J$
2				$+J$

Current is not output

			$\boldsymbol{B}/\boldsymbol{D}$	$\boldsymbol{J}$
i			$+1$	
2			-1	
$\boldsymbol{D}$			$+1$	$+J$

Current is output

Table 3.2 – Current source element J stamp.

	v_{i}	v_1		$\boldsymbol{J}$
i	$+\frac{C}{\Delta t}$	$-\frac{C}{\Delta t}$		$+\frac{C}{\Delta t}v_{cp}$
3	$-\frac{C}{\Delta t}$	$+\frac{C}{\Delta t}$		$-\frac{C}{\Delta t}v_{cp}$

Current is not output

	v_{i}	v_1	$\boldsymbol{B}/\boldsymbol{D}$	$\boldsymbol{J}$
i			$+1$	
3			-1	
$\boldsymbol{C}/\boldsymbol{D}$	$+\frac{C}{\Delta t}$	$-\frac{C}{\Delta t}$	-1	$+\frac{C}{\Delta t}v_{cp}$

Current is output

Table 3.3 – Capacitance element C stamp.

	v_{i}	v_1		
i	$+G$	$-G$		
4	$-G$	$+G$		

Current is not output

	v_{i}	v_1	$\boldsymbol{B}/\boldsymbol{D}$	
i			$+1$	
4			-1	
$\boldsymbol{C}/\boldsymbol{D}$	$+G$	$-G$	-1	

Current is output

Table 3.4 – Conductance element G stamp.

	v_{i}	v_1	$\boldsymbol{B}/\boldsymbol{D}$	$\boldsymbol{J}$
i			$+1$	
5			-1	
$\boldsymbol{C}/\boldsymbol{D}$	$+1$	-1	$-\frac{L}{\Delta t}$	$-\frac{L}{\Delta t}i_{lp}$

Current is output

Table 3.5 – Inductance element L stamp.

3.3.1 MNA formulation example

The circuit of Figure 3.9 is used to give an example of the MNA equation formulation.

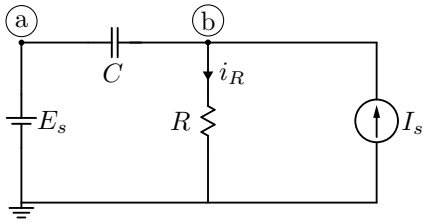


Figure 3.9 – MNA formulation example.

The current i_R through resistor R is selected as an output in this example, therefore the $\boldsymbol{V}$ vector is defined as:

$$\boldsymbol{V} = \begin{bmatrix} v_a \\ v_b \\ i_{Es} \\ i_R \end{bmatrix} \tag{3.52}$$

The MNA equation

$$\begin{bmatrix} \frac{C}{\Delta t} & -\frac{C}{\Delta t} & 1 & 0 \\ -\frac{C}{\Delta t} & \frac{C}{\Delta t} & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & \frac{1}{R} & 0 & -1 \end{bmatrix} \begin{bmatrix} v_a \\ v_b \\ i_{Es} \\ i_R \end{bmatrix} = \begin{bmatrix} \frac{C}{\Delta t} v_{cp} \\ -\frac{C}{\Delta t} v_{cp} + I_s \\ E_s \\ 0 \end{bmatrix} \tag{3.53}$$

was derived from the combination (sum) of the stamps of each element in the example, which are detailed next.

STAMPS

Element: E $\quad \boldsymbol{Y}: \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \boldsymbol{J}: \begin{bmatrix} 0 \\ 0 \\ E_s \\ 0 \end{bmatrix}$

Element: C $\quad \boldsymbol{Y}: \begin{bmatrix} \frac{C}{\Delta t} & -\frac{C}{\Delta t} & 0 & 0 \\ -\frac{C}{\Delta t} & \frac{C}{\Delta t} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \boldsymbol{J}: \begin{bmatrix} \frac{C}{\Delta t} v_{cp} \\ -\frac{C}{\Delta t} v_{cp} \\ 0 \\ 0 \end{bmatrix}$

Element: R $\quad \boldsymbol{Y}: \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{R} & 0 & -1 \end{bmatrix} \quad \boldsymbol{J}: \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

Element: I_s $\quad \boldsymbol{Y}: \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \boldsymbol{J}: \begin{bmatrix} 0 \\ I_s \\ 0 \\ 0 \end{bmatrix}$

3.4 CHAPTER REVIEW

- Contrary to the SVA method, in nodal analysis methods the discretization is performed before the system equation is formulated.
- The use of variable step length adds overhead because the admittance matrix needs to be reformulated with new element discretizations.

- The solution of nodal equations in each time-step is usually faster than the SVA[2] because only relatively simple calculation algorithms such as LU factorization are needed.
- Trapezoidal integration can create numeric oscillations, which must be damped artificially. This is usually done with snubber circuits or parasitic resistances.
- The classic nodal method has severe limitations, which were overcome with the development of the MNA. There are many modern commercial programs based on this method.

[2] Except for discretized SVA equations.

Chapter 4

Digital real-time simulation

According to the IEEE Technical Committee on Real-Time Systems (TCRTS) [27], a real-time system is defined as "a computing system whose correct behavior depends not only on the value of the computation but also on the time at which outputs are produced". In a real-time circuit simulation, the imposed timing constraints lead to simulated voltages and currents with a dynamic equivalent to their real system counterparts.

The elements of a simulation such as power converters, generators, protective devices, controllers, etc., may be entirely described within the mathematical model. However, with the assistance of proper DACs and ADCs, real electric hardware can be added as a component of the RTS, characterizing the simulation mode termed hardware-in-the-loop. The main test configurations of this mode are summarized next [1]:

Real controller validation[1]. This scheme consists of a simulated plant and a real controller added to the simulation loop (Figure 4.1). It is usual for a new controller to go through several cycles of testing and redesign while connected directly to a real power stage or prototype. The simulation of the power stage reduces cost, time

[1] Sometimes referred to directly as HIL, or as "Controller HIL".

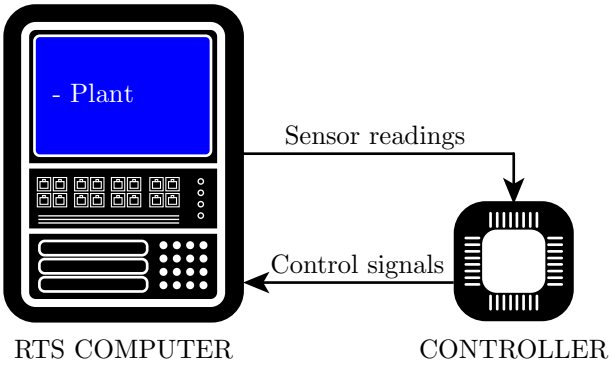


Figure 4.1 – Real controller validation.

for assembly, needed human resources, wasted energy and physical space. At the same time it protects the real equipment from controller flaws and, consequently, provides a much higher level of safety to the involved staff.

Power hardware-in-the-loop (PHIL). A HIL test normally involves the exchange of low power signals between the computer and the connected hardware. With the aid of a power amplifier which feeds voltage and current sensor information back to the computer, the devices under test can be expanded to electric motors, generators, converters, etc. An example PHIL test is illustrated in Figure 4.2.

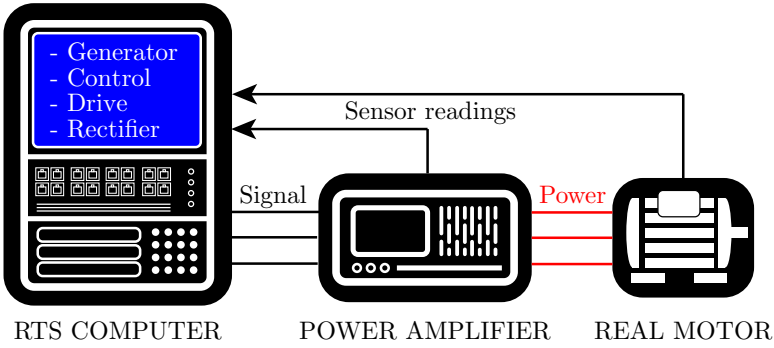


Figure 4.2 – Power hardware-in-the-loop example.

Rapid control prototyping. In this structure, a real plant is connected to a computer that simulates the controller in real-time (Figure 4.3). A simulated controller is more flexible and easier to implement and debug, since every state can be read and the parameters can be altered on the fly. Furthermore, the required controller hardware may not always be readily available for testing. The dSPACE Prototyping System [28] is one example of such configuration.

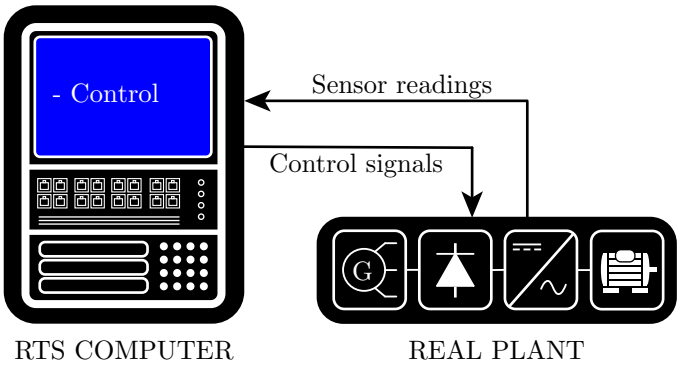


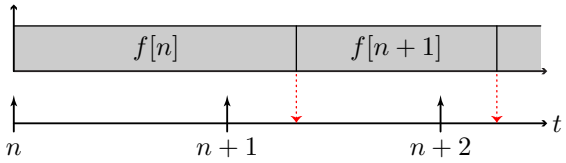
Figure 4.3 – Rapid control prototyping.

A simulation which has no timing constraints and aims at the fastest possible completion is usually referred to as an **offline** type simulation [1, 29, 30]. This flexibility in task completion time allows for high calculation precision and very detailed models, and is still the appropriate approach for the accelerating long term simulations or when very short time-steps are needed. RTS algorithms/models, however, need to abdicate from some of that accuracy and demand extra considerations to maintain synchronism. This chapter details some of those different aspects and presents the methods implemented by OPAL-RT in their system, studied in this master's thesis.

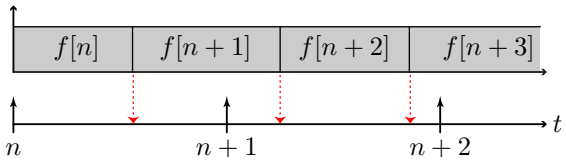
4.1 TIME-STEP CONSTRAINTS

Figure 4.4 compares offline simulations to RTS's from a time perspective. "Real world" time is represented by the t axis, with a $T_s = t_{n+1} - t_n$ value defining a real system output sampling period, or the fixed time-step parameter defined in a simulation software.

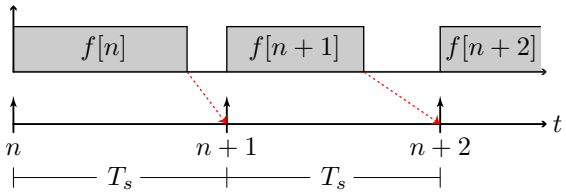
There are three main tasks (represented by $f[n]$) a program must perform at each simulation time-step: 1) update the outputs with the values calculated in the previous time-step, 2) read new inputs and 3) calculate new outputs. In an offline type simulation, a new time-step starts immediately after those tasks are finished, as pointed by the red dashed lines. Figure 4.4a shows the case where the time needed to perform the tasks exceeds T_s, resulting in a slower than real-time simulation, while in Figure 4.4b the simulated circuit runs faster than real-time.



(a) Slower than real-time offline simulation



(b) Faster than real-time offline simulation



(c) Real-time simulation

Figure 4.4 – RTS vs offline simulation.

In an RTS the computation of the outputs ends faster than T_s, but unlike an offline simulation, the program doesn't immediately begin $f[n+1]$ and instead remains idle until t_{n+1} is reached. Figure 4.4c portrays this case. During an RTS, if the computing time exceeds T_s at a time-step, the synchronism with the real system is lost and this event is known as an **overrun**. If the number of overruns in a

simulation is significant and/or increasing, it cannot be trusted. RTS softwares usually have a configurable acceptable number of overruns before terminating the simulation.

Although variable time-step techniques exist to better deal with impulsive events and non-linear systems, they are not currently used in RTS algorithms. With a variable-time step, the computational load is severely decreased during lower frequency periods, thus accelerating the completion of the simulation without losing significant precision; this acceleration, however, is not intended in an RTS.

The latencies involved in the x86 processors memory access and serialized calculation characteristic, results in minimum time-steps of around 10 μs with the simplest of the models [31], which limits the highest frequencies to 5 kHz according to accurate power electronics simulation guidelines [32] – too low for many applications today. Although an FGPA has a lower clock speed than a modern x86 CPU, it is able to reach smaller time-steps with heavy parallelization. This can be achieved with proper modeling techniques [29, 33], and results in minimum time-steps of around 100 ns.

4.2 EVENTS BETWEEN TIME-STEPS

The use of a discrete-time-step solver and A/D sampling in RTS's give rise to some problems. For example, a simple thyristor circuit is shown on Figure 4.5. The switch is triggered at the peak of the input voltage, however there is no guarantee that this event will coincide with the start of a time-step. Figure 4.6 shows the current waveforms of both an ideal circuit and one simulated with a large time-step[2]. A difference can be seen in the interval between input voltage peaks and subsequent time-steps, causing the simulated circuit to read different values of input voltage during each of the switching events depicted, therefore calculating significantly different current outputs and producing unreal subharmonics. This deviation from the real periodicity is known as **jitter** [1].

[2] Although discrete, the simulated circuit waveform is presented ideally shaped for better visibility and comprehension.

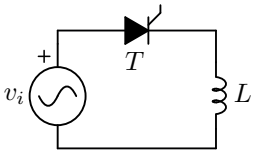


Figure 4.5 – Jitter example - thyristor circuit.

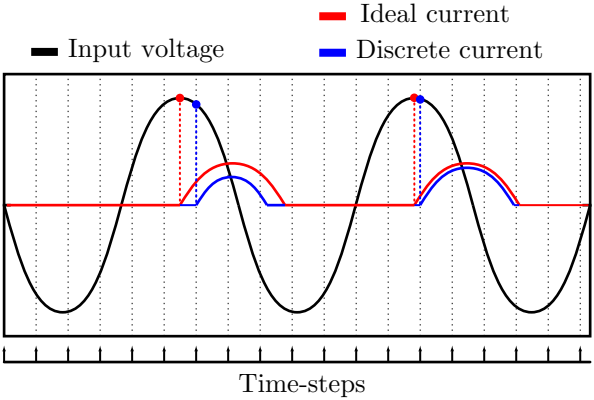


Figure 4.6 – Jitter example - waveforms.

4.3 PARALLEL PROCESSING

A distribution of the computational load between processors and/or processor cores is at times crucial for reaching a small enough time-step, especially when the simulated system is a large network. In a true parallel processing method, at each time-step n the outputs which are part of a subsystem are calculated without the need of $x[n]$ variables from another subsystem, i.e., the subsystems are decoupled.

The Bergeron model for distributed parameters lines presented in Chapter 3, for example, can be used to enable parallel operation with the EMTP method. Equations 3.43 and 3.44 are repeated for convenience:

$$i_{k,m}(t) = \frac{1}{Z} v_k(t) - \frac{1}{Z} v_m(t-\tau) - i_{m,k}(t-\tau) \tag{4.1}$$

$$i_{m,k}(t) = \frac{1}{Z} v_m(t) - \frac{1}{Z} v_k(t-\tau) - i_{k,m}(t-\tau) \tag{4.2}$$

The calculation of the branch current at one side of the transmission line thus depends on the present voltage on the same side and **past** information (delayed by τ) about the voltage and current at the other side. Since $\tau = d\sqrt{L'C'}$, the DPL parameters can be chosen so τ represents one time-step delay. Figure 4.7 illustrates the splitting of a coupled network model into two decoupled subsystems A and B by means of a DPL model.

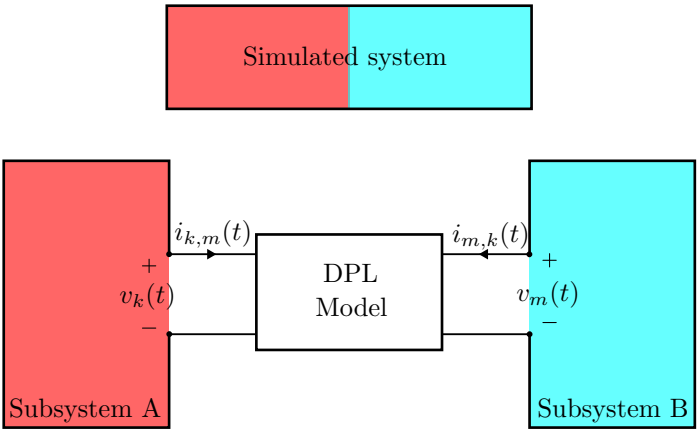


Figure 4.7 – Creation of decoupled subsystems by the addition of a DPL interface.

This introduced delay generates errors, however, and may even cause instabilities, therefore it is important to choose splitting points with approximately constant variables, such as inductor currents or capacitor voltages [15]. Obviously if a transmission line exists in the model, there are no introduced errors; this allows the RTS of large power system networks to be fast and accurate. Since in that case there is no instantaneous term that links variables in one terminal to the other, the resulting admittance matrix has the structure shown in (4.3), where $\boldsymbol{Y}_{ssN}$ are the admittance matrices of each subsystem.

$$\boldsymbol{Y} = \begin{bmatrix} \boldsymbol{Y}_{ss1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Y}_{ss2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{Y}_{ssN} \end{bmatrix} \tag{4.3}$$

4.4 OPAL-RT OP5700 RTS SYSTEM

As it was previously discussed, the real-time simulation of complex switching electrical networks demands powerful processing units and specialized algorithms. If the purpose of the RTS is the realization of tests in a hardware-in-the-loop configuration, fast and precise DACs and ADCs are also needed. Finally, if many different kinds of circuits must be simulated, the software used to create the models must be comprehensive, fast, and provide smooth integration with the hardware.

The RTS system OPAL-RT OP5700 studied in this master's thesis aims to deliver the aforementioned attributes in a single package. In this section, an overview of the main hardware and software aspects is given first, and thereafter the key algorithms implemented by OPAL-RT to allow real-time operation, the **Pejovic** and **State space-nodal** methods, as well as the **RT-Events** package, are presented with more detail.

4.4.1 Hardware specifications

The OP5700 is ultimately a x86 + FPGA system with input/output and communication capabilities. The real-time operation is enabled by an OPAL-RT modified Redhat Linux version. Figure 4.8 shows the front panel of the system. Its hardware consists of:

- A X10DRL-I Supermicro motherboard;
- 32 GB of RAM;
- Two Intel Xeon E5 2.3 GHz processors with 16 cores each;
- A Xilinx VC707 board which includes a Virtex 7 485T FPGA;

- An Oregano syn1588-PCIe clock synchronization board;
- A Softing CAN-AC2 board for Controller Area Network communication;
- 16 small form-factor pluggable (SFP) transceiver ports;
- 4 OPAL-RT developed D/A I/O boards, one for each function.

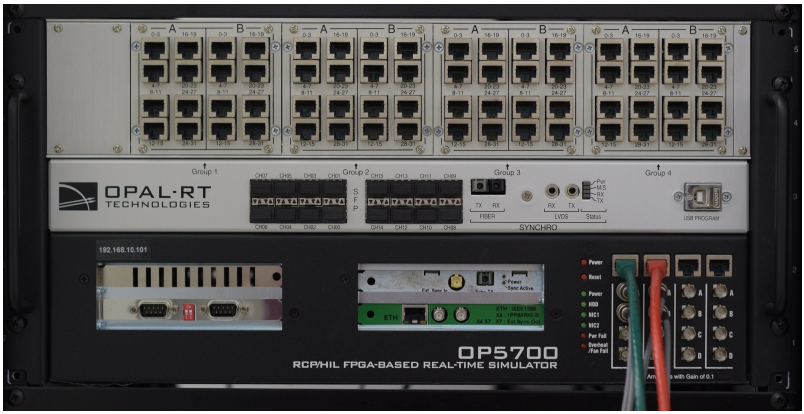


Figure 4.8 – OPAL-RT OP5700 RTS system - front view.

Optical/LVDS synchronization ports allow the interconnection of several OPAL-RT simulation systems for parallel operation. The digital and analog I/Os the system are controlled by the FPGA, which receives signals from the interface boards. The main characteristics of the available I/Os are summarized on Table 4.1.

The software packages responsible for the integration of all the functionalities of the RTS system are:

- *RT-LAB* - A Java/Eclipse IDE based UI used to manage and set parameters of OPAL-RT RTS systems, instruct them to compile and load circuit models (which must be created with Simulink), and change values of the simulation and observe outputs on-the-fly.
- *eMEGASIM* - A Simulink toolbox which includes specialized solvers for multi-CPU processing (ARTEMiS package) and the RT-Events blockset;

	Inputs		Outputs	
	Digital	**Analog**	**Digital**	**Analog**
Number	32	16	32	16
Voltage	4 to 50 V	±24 V	5 to 30 V	±16 V
Current	3,6 mA	-	50 mA max	15 mA max
Rate	10 MSPS	500 KSPS	500 kHz	1 MSPS

Table 4.1 – Digital and analog I/Os

- *eFPGASIM* - A Simulink toolbox used to configure and load models to the FPGA. Contains the eHS package, which allows an user to run simulations on the FPGA without knowing FPGA programming;

In the following subsections, the added Simulink packages are further detailed along with their underlying algorithms.

4.4.2 Advanced Real-Time Electromagnetic Simulator

ARTEMiS is a Simulink package that consists of power systems models and solvers optimized for RTS. It enables the simulation of a large electrical network with many switches by means of subdividing it into smaller, decoupled groups. The mathematical method implemented for this end is called the State Space-Nodal (SSN) solver. The SSN is a method for interfacing state space equations and nodal equations of an arbitrary topology, characterizing a hybrid approach [17].

In Chapter 2 it was shown that the complexity and computational power required for the automatic formulation of the state space equations grow rapidly with the increase of the number of switches. The exponential characteristic of that growth makes the SVA method inefficient for the real-time simulation of systems with many switches. However, since every possible combination of switch

states, and the resulting topologies, are known before runtime, a precalculation of the system of equations is possible. The major limitation, in this case, is the amount of system memory required, which also grows exponentially.

The SSN method discretizes SVA equations from smaller subsets of the circuit called "SSN groups", which are user defined. A limit of 15 switches per group is imposed to allow the pre-calculation of the SVA matrices for each topological state. The results of each group are then injected in a nodal admittance matrix that is smaller than one representing the whole system would be. A description of the SSN formulation based on [17] is now given. Starting with a generic state-space system represented by Equation 4.4:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \hat{\boldsymbol{A}}\boldsymbol{x} + \hat{\boldsymbol{B}}\boldsymbol{u} \\ \boldsymbol{y} &= \hat{\boldsymbol{C}}\boldsymbol{x} + \hat{\boldsymbol{D}}\boldsymbol{u} \end{aligned} \tag{4.4}$$

Consider Δt the duration of the time-step; time point t corresponds to the last calculated value and $t+\Delta t$ to the next value (undergoing calculation). Applying the trapezoidal rule to discretize the first equation of the system:

$$\boldsymbol{x}_{t+\Delta t} = \boldsymbol{x}_t + \int_t^{t+\Delta t} [\hat{\boldsymbol{A}}\boldsymbol{x} + \hat{\boldsymbol{B}}\boldsymbol{u}] d\tau \tag{4.5}$$

$$\boldsymbol{x}_{t+\Delta t} = \boldsymbol{x}_t + \frac{\Delta t}{2}[\hat{\boldsymbol{A}}\boldsymbol{x}_t + \hat{\boldsymbol{A}}\boldsymbol{x}_{t+\Delta t} + \hat{\boldsymbol{B}}\boldsymbol{u}_t + \hat{\boldsymbol{B}}\boldsymbol{u}_{t+\Delta t}] \tag{4.6}$$

$$\boldsymbol{x}_{t+\Delta t} = \tilde{\boldsymbol{A}}\boldsymbol{x}_t + \tilde{\boldsymbol{B}}\big[\boldsymbol{u}_t + \boldsymbol{u}_{t+\Delta t}\big] \tag{4.7}$$

where

$$\tilde{\boldsymbol{A}} = \left[\boldsymbol{U} + \hat{\boldsymbol{A}}\frac{\Delta t}{2}\right]\left[\boldsymbol{U} - \hat{\boldsymbol{A}}\frac{\Delta t}{2}\right]^{-1} \tag{4.8}$$

$$\tilde{\boldsymbol{B}} = \left[\hat{\boldsymbol{B}}\Delta t\right]\left[2\boldsymbol{U} - \hat{\boldsymbol{A}}\Delta t\right]^{-1} \tag{4.9}$$

Each switch state defines a different topology, therefore the subscript k is used to indicate the matrices which correspond to the k-eth topology.

$$\boldsymbol{x}_{t+\Delta t} = \tilde{\boldsymbol{A}}_{\boldsymbol{k}}\boldsymbol{x}_t + \tilde{\boldsymbol{B}}_{\boldsymbol{k}}\boldsymbol{u}_t + \tilde{B}_k\boldsymbol{u}_{t+\Delta t} \tag{4.10}$$

The $\boldsymbol{u}_{t+\Delta t}$ sources are then subdivided into internal (known) sources, represented by the subscript i, and external (unknown) nodal injections, represented by the subscript e. Hence:

$$\boldsymbol{x}_{t+\Delta t} = \tilde{\boldsymbol{A}}_{\boldsymbol{k}}\boldsymbol{x}_t + \tilde{\boldsymbol{B}}_{\boldsymbol{k}}\boldsymbol{u}_t + \begin{bmatrix}\tilde{\boldsymbol{B}}_{k_i} & \tilde{\boldsymbol{B}}_{k_e}\end{bmatrix}\begin{bmatrix}\boldsymbol{u}_{i_{t+\Delta t}} \\ \boldsymbol{u}_{e_{t+\Delta t}}\end{bmatrix} \tag{4.11}$$

$$\begin{bmatrix}\boldsymbol{y}_{i_{t+\Delta t}} \\ \boldsymbol{y}_{e_{t+\Delta t}}\end{bmatrix} = \begin{bmatrix}\hat{\boldsymbol{C}}_{k_i} \\ \hat{\boldsymbol{C}}_{k_e}\end{bmatrix}\boldsymbol{x}_{t+\Delta t} + \begin{bmatrix}\hat{\boldsymbol{D}}_{k_{ii}} & \hat{\boldsymbol{D}}_{k_{ie}} \\ \hat{\boldsymbol{D}}_{k_{ei}} & \hat{\boldsymbol{D}}_{k_{ee}}\end{bmatrix}\begin{bmatrix}\boldsymbol{u}_{i_{t+\Delta t}} \\ \boldsymbol{u}_{e_{t+\Delta t}}\end{bmatrix} \tag{4.12}$$

The substitution of (4.11) into the second row of (4.12) yields:

$$\begin{aligned}\boldsymbol{y}_{e_{t+\Delta t}} = \hat{\boldsymbol{C}}_{k_e}(\tilde{\boldsymbol{A}}_{\boldsymbol{k}}\boldsymbol{x}_t + \tilde{\boldsymbol{B}}_{\boldsymbol{k}}\boldsymbol{u}_t + \tilde{\boldsymbol{B}}_{k_i}\boldsymbol{u}_{i_{t+\Delta t}} + \tilde{\boldsymbol{B}}_{k_e}\boldsymbol{u}_{e_{t+\Delta t}}) + \\ \hat{\boldsymbol{D}}_{k_{ei}}\boldsymbol{u}_{i_{t+\Delta t}} + \hat{\boldsymbol{D}}_{k_{ee}}\boldsymbol{u}_{e_{t+\Delta t}}\end{aligned} \tag{4.13}$$

Rearranging:

$$\begin{aligned}\boldsymbol{y}_{e_{t+\Delta t}} = \underbrace{\hat{\boldsymbol{C}}_{k_e}(\tilde{\boldsymbol{A}}_{\boldsymbol{k}}\boldsymbol{x}_t + \tilde{\boldsymbol{B}}_{\boldsymbol{k}}\boldsymbol{u}_t + \tilde{\boldsymbol{B}}_{k_i}\boldsymbol{u}_{i_{t+\Delta t}}) + \hat{\boldsymbol{D}}_{k_{ei}}\boldsymbol{u}_{i_{t+\Delta t}}}_{hist} + \\ \underbrace{(\hat{\boldsymbol{C}}_{k_e}\tilde{\boldsymbol{B}}_{k_e} + \hat{\boldsymbol{D}}_{k_{ee}})}_{\boldsymbol{W}_{k_e}}\boldsymbol{u}_{e_{t+\Delta t}}\end{aligned} \tag{4.14}$$

The "hist" term indicated in Equation 4.14 denotes the known variables. In a compact form:

$$\boldsymbol{y}_{e_{t+\Delta t}} = \boldsymbol{y}_{k_{hist}} + \boldsymbol{W}_{k_e}\boldsymbol{u}_{e_{t+\Delta t}} \tag{4.15}$$

The $\boldsymbol{y}_e$ term is a generic response term, and it may represent node voltages or current injections. In the case of current injections, $\boldsymbol{u}_e$ contains nodal voltages, $\boldsymbol{W}_{k_e}$ is an admittance matrix and in $\boldsymbol{y}_{k_{hist}}$ are known currents. This is called a V-type SSN group in [17]. Similarly, if $\boldsymbol{y}_e$ represents node voltages, $\boldsymbol{u}_e$ has current injections, $\boldsymbol{W}_{k_e}$ is an impedance matrix and $\boldsymbol{y}_{k_{hist}}$ has known node voltages. This is an I-type SSN group. Equation 4.15 can be written as follows, considering both group types:

$$\begin{bmatrix}\boldsymbol{v}^I_{e_{t+\Delta t}} \\ \boldsymbol{i}^V_{e_{t+\Delta t}}\end{bmatrix} = \begin{bmatrix}\boldsymbol{v}_{k_{hist}} \\ \boldsymbol{i}_{k_{hist}}\end{bmatrix} + \begin{bmatrix}\boldsymbol{W}_{k_eII} & \boldsymbol{W}_{k_eIV} \\ \boldsymbol{W}_{k_eVI} & \boldsymbol{W}_{k_eVV}\end{bmatrix}\begin{bmatrix}\boldsymbol{i}^I_{e_{t+\Delta t}} \\ \boldsymbol{v}^V_{e_{t+\Delta t}}\end{bmatrix} \tag{4.16}$$

To transform Equation 4.16 into a nodal representation $I = YV$, the first line must be changed so all current vectors group on the left side. The resulting equation after the manipulation is:

$$\begin{bmatrix} \boldsymbol{i}^I_{e_{t+\Delta t}} \\ \boldsymbol{i}^V_{e_{t+\Delta t}} \end{bmatrix} = \boldsymbol{\Gamma}_{k_e} \begin{bmatrix} \boldsymbol{v}_{k_{hist}} \\ \boldsymbol{i}_{k_{hist}} \end{bmatrix} + \boldsymbol{Y}_{k_e} \begin{bmatrix} \boldsymbol{v}^I_{e_{t+\Delta t}} \\ \boldsymbol{v}^V_{e_{t+\Delta t}} \end{bmatrix} \tag{4.17}$$

with

$$\boldsymbol{\Gamma}_{k_e} = \begin{bmatrix} -\boldsymbol{W}_{k_e II}{}^{-1} & \boldsymbol{U} \end{bmatrix} \tag{4.18}$$

and

$$\boldsymbol{Y}_{k_e} = \begin{bmatrix} \boldsymbol{W}_{k_e II}{}^{-1} & \boldsymbol{W}_{k_e II}{}^{-1} \boldsymbol{W}_{k_e IV} \\ \boldsymbol{W}_{k_e VI} & \boldsymbol{W}_{k_e VV} \end{bmatrix} \tag{4.19}$$

Finally, the $\boldsymbol{Y}_{k_e}$ admittance matrix derived is inserted into the global nodal admittance matrix, with its position determined by the group nodes. The negative of the first part in the right hand side of (4.17) contributes to the vector of known currents.

4.4.3 Electric hardware solver (eHS)

The eHS package enables the user to run circuit simulations on the FPGA without the need of direct programming with languages such as VHDL. Instead, algorithms automatically convert a model constructed with SimPowerSystems[3] blocks to a set of matrices which are then loaded into the FPGA and calculated using a nodal method referenced by OPAL-RT as the *Pejovic method*, published on [34]. Using the original paper as basis, the key points of this method are discussed next.

Some circuit simulators, such as SPICE, model switches by a large resistor value when the switch is open (R_{off}) and a small resistor value when it is closed (R_{on}), with a smooth transition between R_{on} and R_{off}. However, a recalculation of the admittance matrix is necessary with the updated resistance values, as it is in ideal switching based methods, which append or remove rows and columns depending on the switching state.

[3] Some PSIM, Multisim and PLECS versions are also supported.

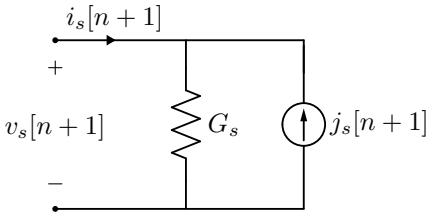


Figure 4.9 – Discretized switch model.

Consider the discretized model of Figure 4.9, where $n+1$ represents the current time-step. The use of KCL results in:

$$i_s[n+1] = G_s v_s[n+1] - j_s[n+1] \tag{4.20}$$

The strategy of the Pejovic method is using the discretized model with a constant G_s, and working with the j_s current source value to define the switch state. This approach results in a constant admittance matrix, which must be inverted a single time for the entire simulation duration, or even before runtime – which boosts the simulation speed and facilitates FPGA implementation. The j_s current sources from switch models enter the vector of independent sources of the nodal equation.

If the simulation time-step T_s is small enough for the approximations $v_s[n+1] \approx v_s[n]$ and $i_s[n+1] \approx i_s[n]$ to be valid, the ideal switch characteristic can be obtained if the current source defined by (4.21) is chosen.

$$j_s[n+1] = \begin{cases} -i_s[n] & \text{if the switch is closed at } n+1 \\ G_s v_s[n] & \text{if the switch is open at } n+1 \end{cases} \tag{4.21}$$

Errors (non-zero closed switch voltage and open switch current) introduced by this approximation are defined by Equations 4.22 and 4.23. It may be shown that they converge if the rest of the network is a stable LTI system with no time-varying sources.

$$v_s[n+1]|_{s=1} = G_s^{-1}(i_s[n+1] - i_s[n]) \tag{4.22}$$

$$i_s[n+1]|_{s=0} = G_s(v_s[n+1] - v_s[n]) \tag{4.23}$$

It is evident from the equation of $j_s[n+1]$ that this switch model depends on values at the previous time-step, which suggests an interpretation of the model as a discretization of an energy storage element equation. An LTI capacitor C is described by the equation $Cdv_C(t)/dt = i_C(t)$. Integrating:

$$v_C(t) = v_C(0) + \frac{1}{C}\int_0^t i_C(t)dt \tag{4.24}$$

In discrete form:

$$v_C[n+1] = v_C[n] + \frac{1}{C}\int_{t[n]}^{t[n+1]} i_C[t]dt \tag{4.25}$$

If the Backward Euler method is used to approximate the integral, the resulting equation is:

$$i_C[n+1] = \frac{C}{T}v_C[n+1] - \frac{C}{T}v_C[n] \tag{4.26}$$

If C/T is a conductance G_C:

$$\begin{aligned} i_C[n+1] &= G_C v_C[n+1] - G_C v_C[n] \\ &= G_C v_C[n+1] - j_C[n+1] \end{aligned} \tag{4.27}$$

After applying the same procedure to a LTI inductor described by $Ldi_L(t)/dt = v_L(t)$, the final discrete equation obtained is

$$\begin{aligned} i_L[n+1] &= G_L v_L[n+1] + i_L[n] \\ &= G_L v_L[n+1] - j_L[n+1] \end{aligned} \tag{4.28}$$

where $G_L = T/L$.

The comparison of (4.27) and (4.28) with (4.20) and (4.21) confirms that the Pejovic switch model is analog to a discrete inductance model when closed and to a discrete capacitance model when open. The G_s parameter is constant, therefore:

$$G_s = G_C = G_L = \frac{C}{T} = \frac{T}{L} \tag{4.29}$$

Equation 4.21 was obtained with Backward Euler integration, but the LC interpretation implies that different numerical integration methods can be applied to the energy storage element models with the restriction that $G_L = G_C = G_s$ to obtain valid switch models. As an example, the following equations are the result of a trapezoidal rule integration:

$$G_s = \frac{2C_s}{T} = \frac{T}{2L_s} \tag{4.30}$$

$$j_s[n+1] = \begin{cases} -i_s[n] - G_s v_s[n] & \text{if closed switch at } n+1 \\ i_s[n] + G_s v_s[n] & \text{if open switch at } n+1 \end{cases} \tag{4.31}$$

Finally, the paper investigates how to properly choose switch conductance and time-step values. It is concluded that on-switch instantaneous voltage errors reduce with a higher G_s and off-switch instantaneous current errors are reduced with a lower G_s. Shorter time-steps reduce the cumulative, i.e., steady-state errors introduced. This will be experimentally verified in Chapter 5.

4.4.4 Real-time events blockset

As discussed in Section 4.2, events usually occur in-between time steps and, thus, create errors and oscillations. The use of variable time-step algorithms is one way to address this problem, but it is not applicable in RTS's. The Real-time Events (RTE) blockset developed by OPAL-RT aims to deliver much more precise calculations whilst maintaining a fixed time-step value.

The *RTE PWM* block operation is shown in Figure 4.10 and exemplifies one of the implemented strategies. The input values of the block are scalars that define the PWM carrier frequency and the duty-cycle. Considering a Δt CPU time-step value, the maximum possible signal frequency is $1/2\Delta t$. The RTE PWM block, however, outputs a data type that consists of a pair of vectors with switching instants and the corresponding states. This data can be converted by the FPGA to a signal with the frequency determined by the total number of events (eight in the example) and limited by the FPGA's

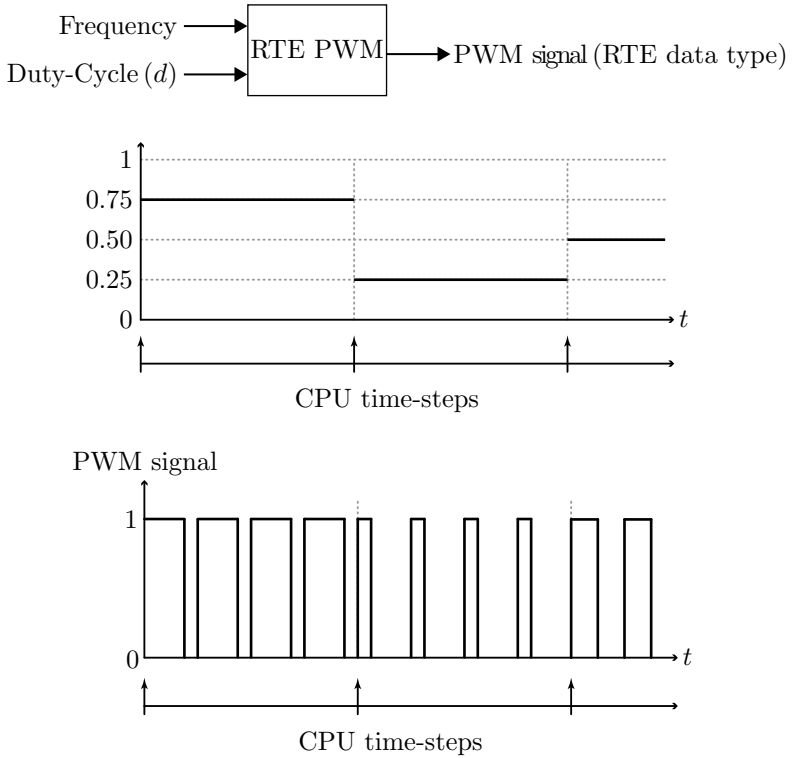


Figure 4.10 – RTE data type.

smaller time-step. The update of the duty-cycle and frequency input values can only be done at the start of CPU time-steps nonetheless.

As another example, the jitter problem previously mentioned can be mitigated. An I/O card is able to acquire information at a much higher rate than the simulation[4], and therefore is able to detect switching events faster. At the start of a new time-step, instead of simply informing the CPU the last sampled value, the I/O card transmits the detected events and their time of occurrence, allowing RTE algorithms to add a compensation value, which is usually calculated by averaging the pulses.

[4] See Table 4.1

4.5 CHAPTER REVIEW

- The RTS of electric circuits, especially in a HIL configuration, brings several advantages to a design process like safety, flexibility and speed;
- There are certain restrictions in an RTS that are not found in an offline simulation, including the practical obligation to use a fixed time-step method, which calculates impulsive signals with lower precision and can create some problems such as jitter;
- A method for the paralellization of the calculations is often mandatory for a precise RTS of large systems;
- The SSN is a hybrid method. It is able to solve – in parallel – circuit subdivisions represented by discretized state-space equations in order to reduce the size and complexity of the system nodal equation;
- Switches are modeled in the Pejovic method as discrete capacitances when off and inductances when on. This results in a constant admittance matrix at the expense of numerical errors during transitions;
- A CPU simulation can only detect or create switching events at the start of a new time-step. The RTE blockset applies compensations to the generated signals due to this delay.

Chapter 5

Experimental results

In order to evaluate the accuracy and speed of the OP5700 real-time simulator and the implemented OPAL-RT algorithms discussed in the previous chapter, multiple simulations were prepared. Although the possibilities are practically endless, the studied computer-aided circuit analysis theory helped with the determination of concise and meaningful tests.

This chapter starts with an overview of a designed signal conditioning board used in the HIL simulations. Next, an assessment of the ARTEMiS SSN solver performance gains is done for single and multi-core CPU usage configurations. After that, an application example of the RTE package is given. A practical analysis of the Pejovic method limitations is then presented. Finally, some hardware-in-the-loop simulations are shown to confirm the functionality.

Since each circuit has its particularities, the feasibility and merits of its RTS/HIL oriented model must be carefully assessed by the engineer. At the end of this chapter, it is expected that the reader acquired a notion of the minimum time-steps that are achievable with each solver; of what type of simulations benefit the most from OPAL-RT's methods; of the magnitude of the introduced errors and the circuit characteristics that cannot be accurately modeled

because of those errors; and of the practicality and accuracy of HIL simulations.

5.1 SIGNAL CONDITIONING BOARD

The controller hardware used in HIL simulations is a Texas Instruments F28069 LaunchPad DSC, which demanded the design of an interface board[1], shown in Figure 5.1, to allow the exchange of signals with OP5700's I/Os. Although the routing arrangement was chosen to match the F28377S DSC, there is almost full pin compatibility with other LaunchPad DSC kits, which are the most widespread digital control platforms employed at INEP.

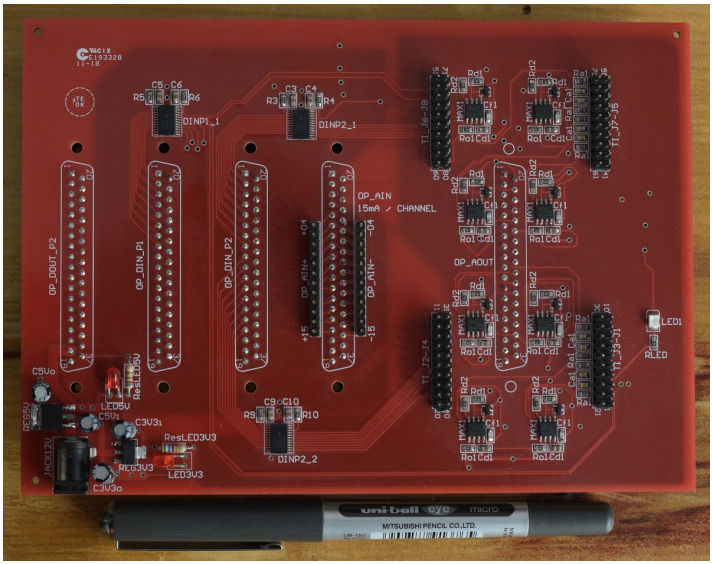


Figure 5.1 – Signal conditioning board.

The PCB is powered by a 12 V / 1 A power supply through a standard 5.5 mm power jack. 5 V and 3.3 V power planes are supplied by two linear regulators (12–5 V and 5–3.3 V). The planes feed 8-bit dual supply bus transceivers, which are the interface between OP5700's digital I/Os and the DSC's digital I/Os, as illustrated in Figure 5.2. A DIR (direction) bit defines which are the transceiver's input and output sides. OP5700's analog outputs range from −16 to +16 V, while the DSC's ADC inputs support 0 to 3.3 V signals. The

[1] Complete schematics and the PCB layout are available in Appendix A.

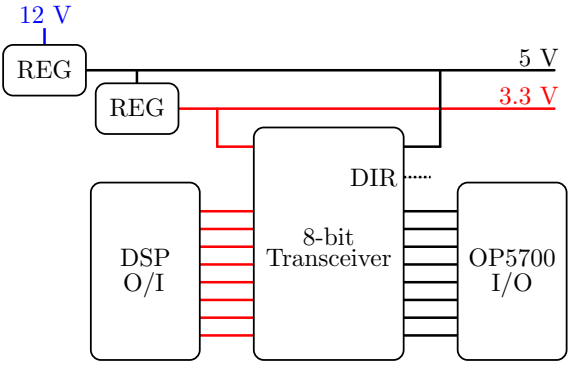


Figure 5.2 – Conditioning of the digital channels.

circuit represented in Figure 5.3 provides the appropriate gain and an extra layer of protection to the ADCs by means of an asymmetric rail-to-rail op-amp output configuration. One bit of precision from the internal DACs of OP5700's analog outputs is forfeit, corresponding to the negative voltage range. There is no overall precision loss however since the DSC ADCs are 12-bit and OP5700's output DACs are 16-bit.

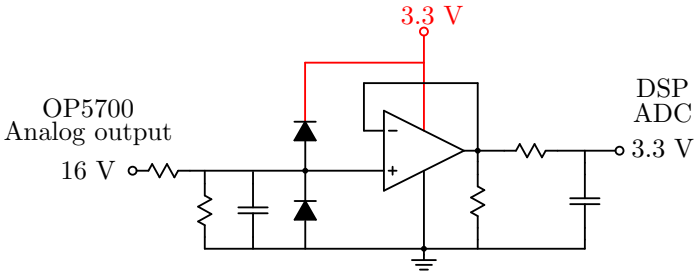


Figure 5.3 – Conditioning of the analog channels.

5.2 SSN PERFORMANCE

Nodal methods are theoretically faster than SVA based methods for calculating large switching networks. The simulation depicted in Figure 5.4 was therefore performed to compare the SSN solver with the standard Simscape Power Systems (SPS) discrete SVA solver in

terms of calculation speed. Each SSN group included two rectifiers and, thus, twelve diodes, adding to a total of 120 simulated switches. The SPS solver ignores the partitioning and calculates the network as a whole. It is interesting to add that without the group separation, the discretization and preallocation of the state equation matrices would require the absurd amount of 2.5×10^{36} megabytes of memory, as reported by the SSN algorithm.

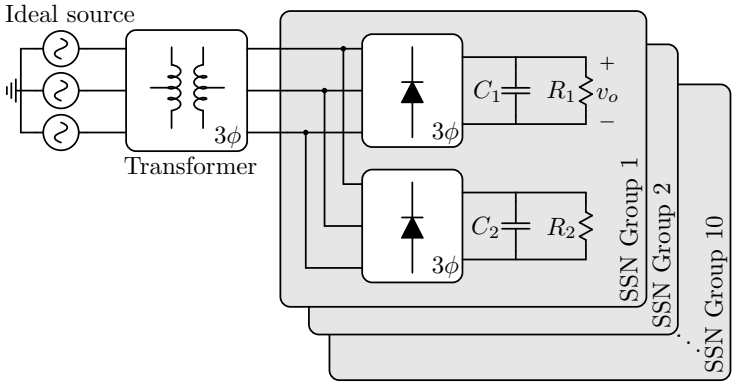


Figure 5.4 – SSN speed test: parallel three-phase rectifiers.

Backward Euler integration with 20 μs time-step was used, and the total simulated time was set as 10 s. In the single CPU core test, the SPS algorithm took 142 seconds to complete the simulation

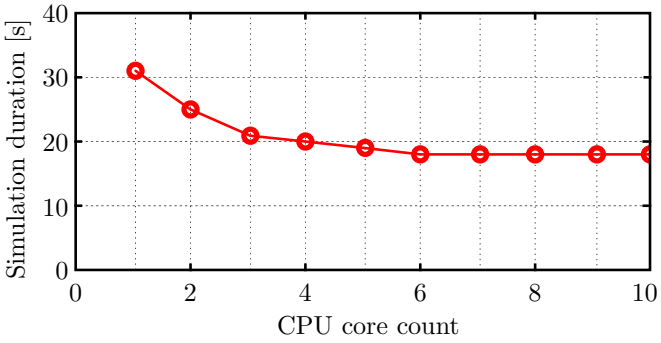


Figure 5.5 – SSN speed test: core count vs. simulation duration.

against 31 seconds for the SSN; a significant 78% time reduction. Moreover, considering the subdivision in 10 groups, an acceleration was expected with the addition of up to 9 cores. As can be seen in Figure 5.5 there was a saturation after 6 cores, however. This can be explained either by imperfections in the algorithm or, most probably, a point was reached where the nodal equation is accountable for most of the calculation time. Regardless, with 6 cores the SSN solver reduced the duration of the simulation to 18 seconds, approximately an eighth of the SPS time.

5.3 SSN ACCURACY

The next step taken was confirming the simulation speed gains didn't signify major precision loss. For that end, the simulation of the parallel rectifiers was performed again, but with a small 2 μs time-step and using the SPS solver. The intent was the creation of accurate reference values, which are plotted in Figure 5.6.

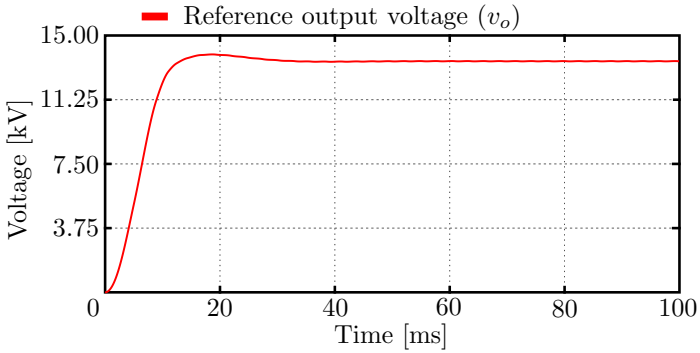


Figure 5.6 – Rectifier 2 μs time-step reference simulation.

Afterwards, the voltage calculation results of SPS and SSN simulations with a ten times larger time-step were subtracted from the reference values and the graphs of Figures 5.7 and 5.8 plotted. It can be observed the numerical errors introduced by increasing the time-step have similar magnitudes for both the SSN and the SPS solvers.

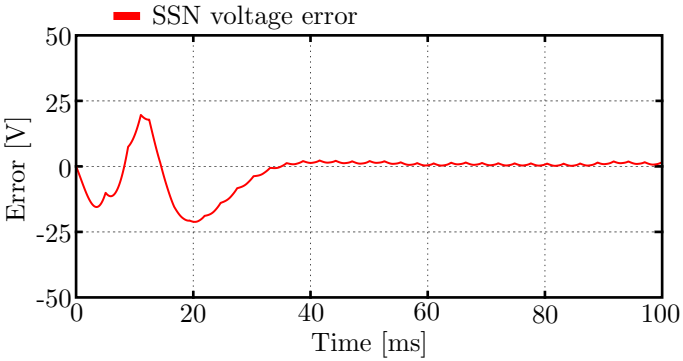


Figure 5.7 – Rectifier 20 μs SSN solver error.

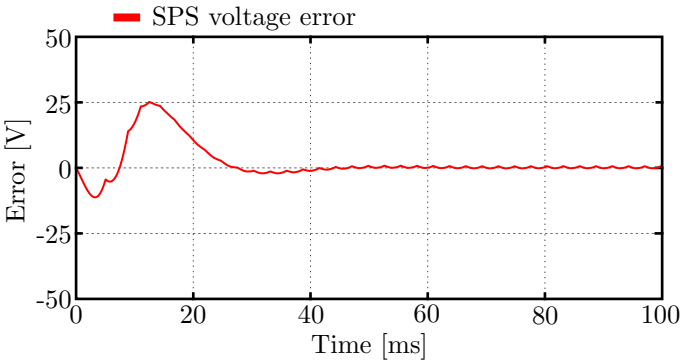


Figure 5.8 – Rectifier 20 μs SPS solver error.

Another test following the same methodology was conducted with a neutral-point clamped (NPC) inverter to observe and compare the accuracy of the solvers considering forced switching elements. A representation of the simulated circuit can be seen in Figure 5.12, where $E = 200$ V and the in-phase disposition (IPD) [35] carrier-based PWM modulating signal is a 30 Hz sine wave. A relatively small 1.5 kHz carrier frequency was used so precise enough gate signals could be generated considering the 20 μs time-step. Although a load current with high harmonic distortion resulted, it is not relevant for the intended comparison. Figure 5.9 shows the reference simulation load current. The comparison of calculation errors presented in Figures 5.10 and 5.11 confirm the satisfactory SSN precision.

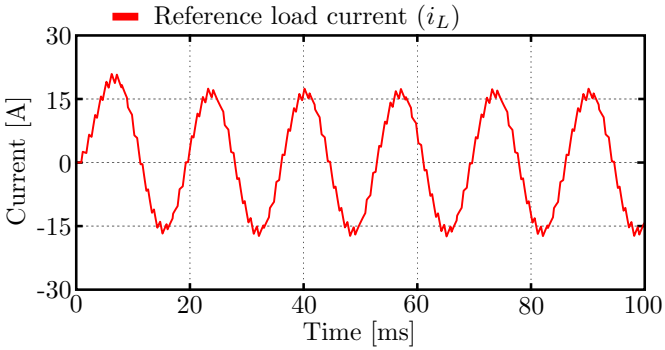


Figure 5.9 – NPC 2 μs time-step SPS reference simulation.

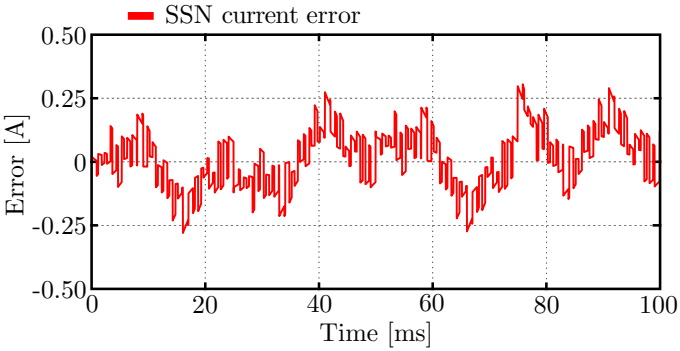


Figure 5.10 – NPC 20 μs SSN solver error.

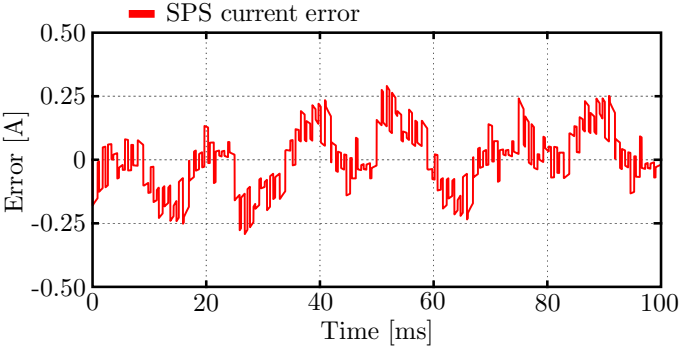


Figure 5.11 – NPC 20 μs SPS solver error.

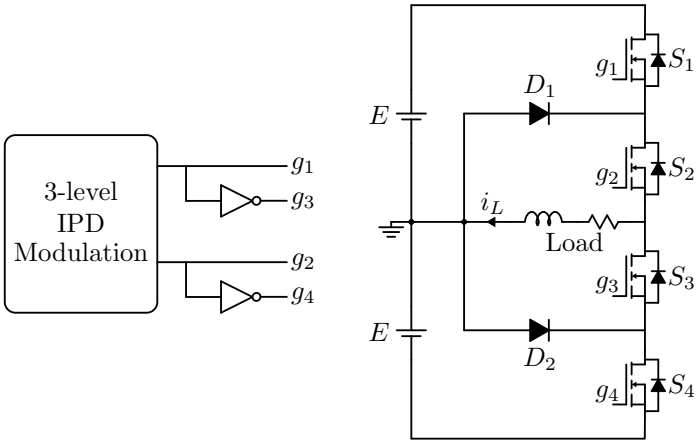


Figure 5.12 – SSN accuracy test: single phase NPC inverter.

5.4 REAL TIME EVENTS

Figure 5.13 presents the simulation prepared to verify the RTE toolbox effectiveness. A DSC is used to generate complementary 10 kHz SPWM gate signals to trigger the switches of a half-bridge

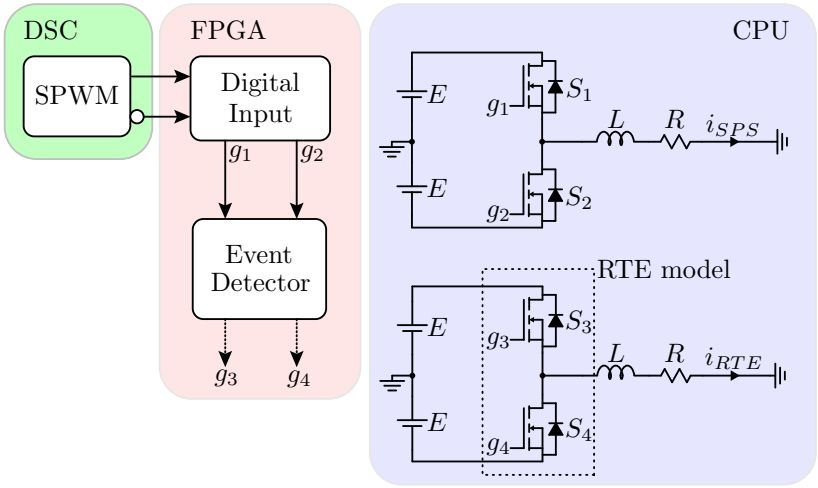


Figure 5.13 – RTE test: implemented model.

inverter. The chosen time-step of the simulation is 10 μs, which is too large to sample the g_1 and g_2 10 kHz signals appropriately; severe errors in the RL load current i_{SPS}, generated by the standard SPS bridge model, are thus expected.

The OP5700 FPGA card, however, is able to sample the gate signals accurately given its 10 MSPS rate. An event detector block inside the FPGA converts the g_1 and g_2 pulses to data signals g_3 and g_4, which contain every state transition as well as time information. The RTE bridge model interprets g_3 and g_4 and calculates a compensated output voltage, which its turn generates the much more accurate i_{RTE} current. The compensated output voltage of the RTE model does not correspond to the one produced with an adequate time-step, for there are not enough points to represent the needed duty-cycle variations. Instead, sinusoidal components which result from the averaging of the RTE-type pulses will be present, as it can be seen on Figure 5.14.

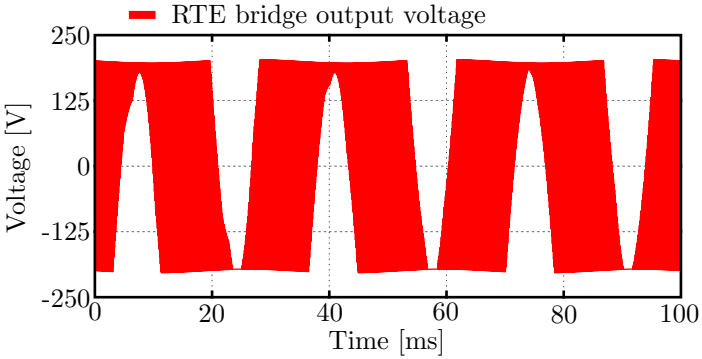


Figure 5.14 – Compensated output voltage of the RTE bridge model.

The load currents for the SPS and the RTE bridge models can be observed in Figure 5.15. With the SPS bridge there are significant errors, including jitter, while the RTE bridge produces a smooth sinusoidal current. A zoom-in in the first peak is shown in Figure 5.16 for extra detail and easier visualization of the magnitude of the errors.

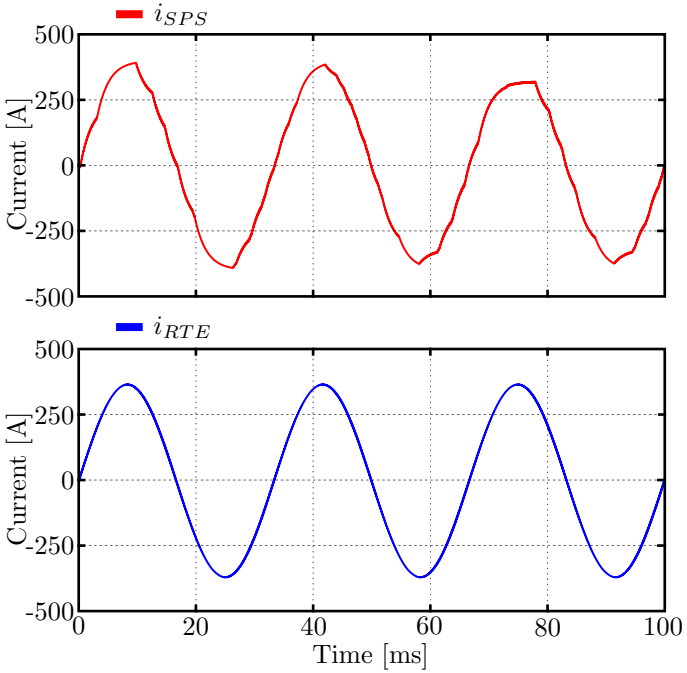


Figure 5.15 – SPS model and RTE model load currents.

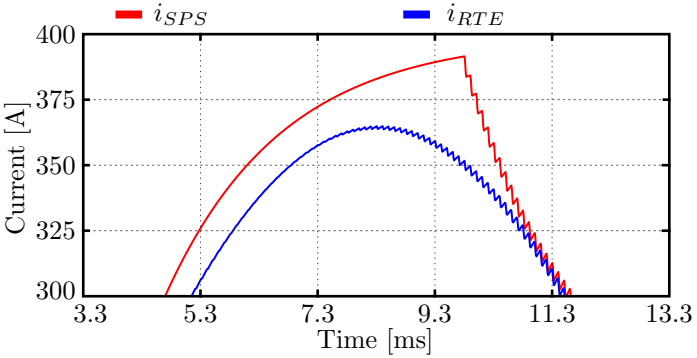


Figure 5.16 – Zoomed-in first peak of the load currents.

5.5 EHS – COMPUTATIONAL PERFORMANCE

The eHS package limits the number of switching components in a model to 144, at the time of this writing. In order to test the performance of the FPGA RTS with a large number of switches, a Multilevel Modular Converter (MMC) model as shown on the schematic of Figure 5.17 was prepared, which contains 120 switches in total. Open-loop gate signals were generated in the CPU by phase-shift modulation [35]. The parameters of the simulation are summarized on Table 5.1.

Modulating frequency	60 Hz
Carrier frequency	1 kHz
Input voltage	10 kV
Load current amplitude	300 A
FPGA time-step	1.25 μs
CPU time-step	10 μs

Table 5.1 – MMC model parameters

The chosen FPGA time-step was 1.25 μs, the minimum achievable for this model according to eHS automatic calculations. With that value, the converter was simulated in real-time effectively and without overruns. In order to measure the performance gain by simulating on the FPGA, an equivalent SPS model was run on an AMD Ryzen 7 1700X 3.4 GHz CPU and configured to a 1.25 μs time-step. A ratio of 14 seconds per simulation second was needed by the CPU, while the FPGA in acceleration mode (offline type simulation) was able to calculate each simulation second in half a second.

Figure 5.18 shows the upper (CH1) and lower (CH2) arm voltages of leg A (0.001 gain), as well as the difference between them, which corresponds to the phase A voltage. As expected, the lack of a strategy to balance capacitor voltages resulted in varying voltages for each level. The unbalanced capacitor voltages in submodules SM1A to SM4A can be seen in the CH1 to CH4 waveforms of Figure 5.19 (2 kV offset applied).

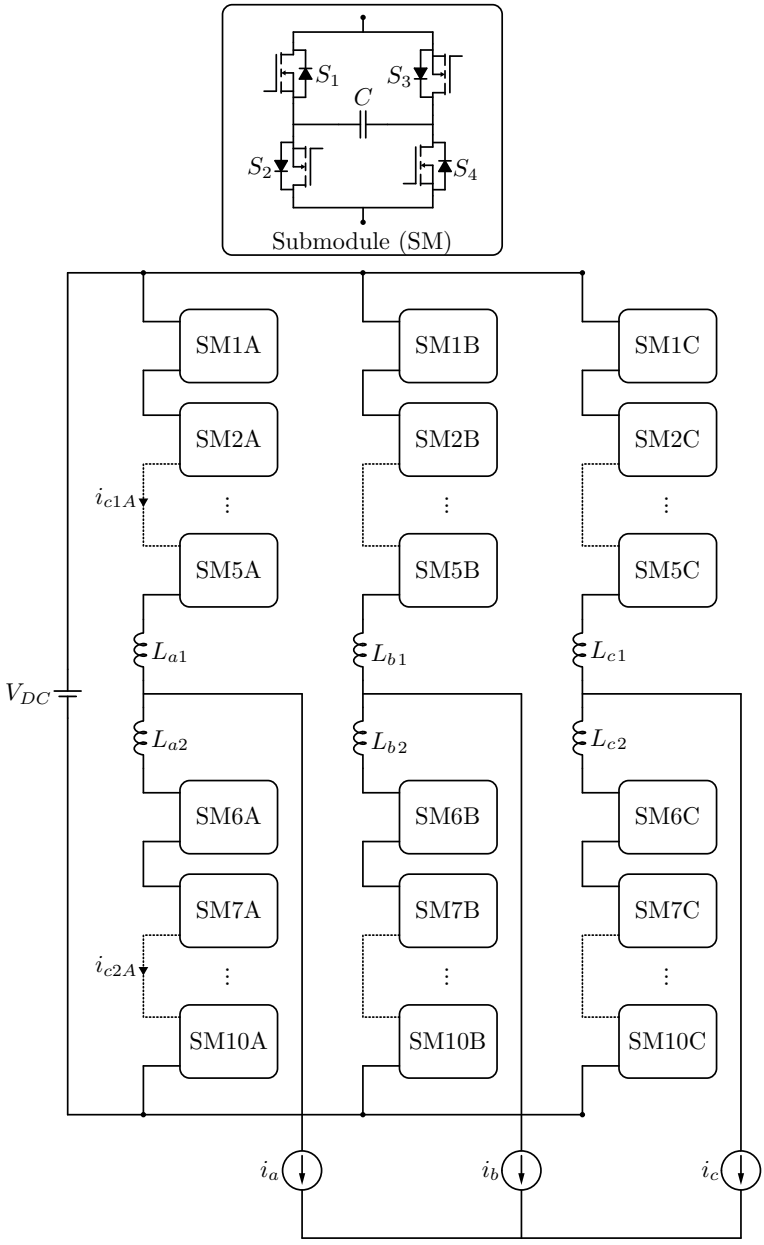


Figure 5.17 – MMC model simulated by the FPGA.

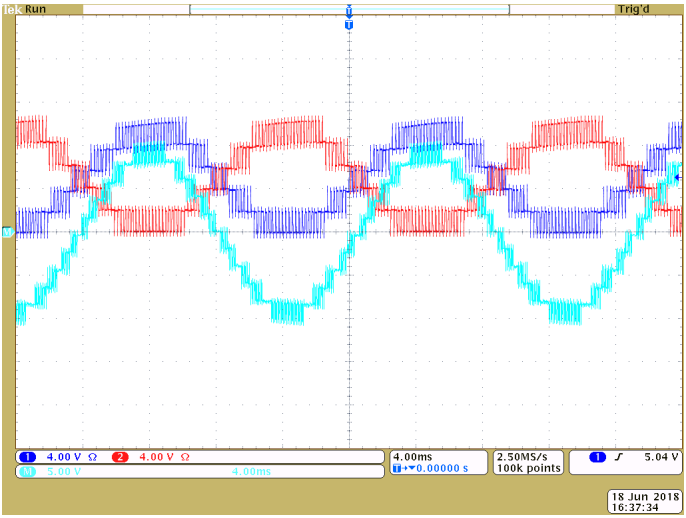


Figure 5.18 – MMC: Leg A top (blue) and down (red) arm voltages and their difference (phase A voltage, turquoise).

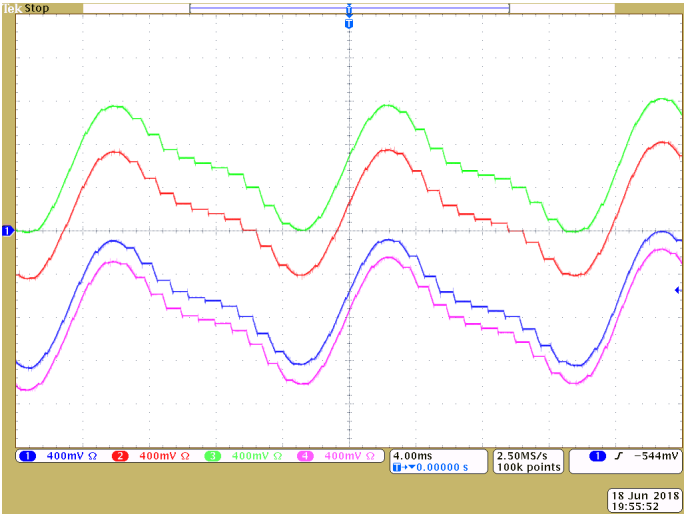


Figure 5.19 – MMC: capacitor voltages in submodules 1A (blue), 2A (red), 3A (green) and 4A (pink).

5.6 EHS – ACCURACY OF THE PEJOVIC METHOD

The Pejovic method, described in Section 4.4.3, is used on the FPGA with the eHS package. In order to allow a preview of the simulation in the user computer instead of the RTS system, OPAL-RT made available a Simulink block called "eHS Offline Simulation" which substitutes the default SPS algorithms with the Pejovic method. Comparison tests between the output of simulations using the offline block and the output of the RTS system were done in order to verify the validity of the block.

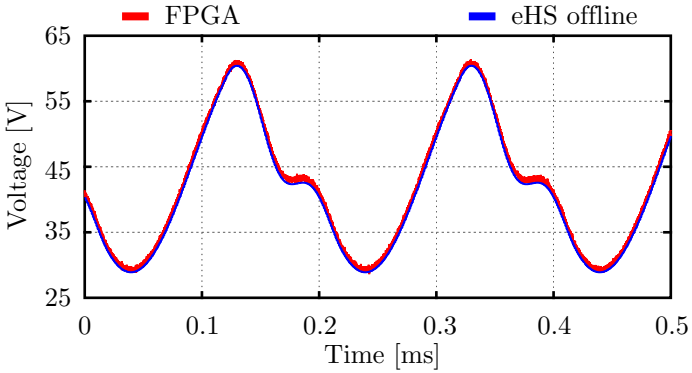


Figure 5.20 – eHS offline block validation: 1 μs time-step.

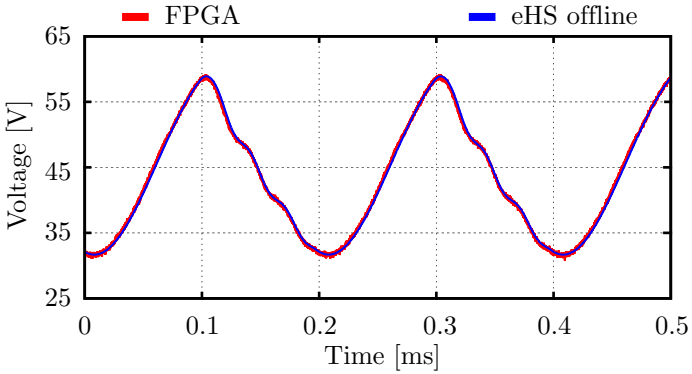


Figure 5.21 – eHS offline block validation: 250 ns time-step.

The simulations consisted of random circuits and were done for several values of time-step. Figures 5.20 and 5.21 show example outputs (voltages) of 1 μs and 250 ns time-step tests, and it can be seen that – except for errors introduced by D/A conversions and oscilloscope readings – the responses are equivalent. This correspondence was observed in all the tests. The Offline Simulation block was then used to evaluate the effect of the Pejovic method for the accuracy of the results.

An open-loop buck converter simulation which used the standard SPS solver and with a very small time-step (10 ns, for a 40 kHz switching frequency) was performed to be used as reference. In Figure 5.22 the switch voltage waveform for $G_s = 0.1$ is plotted over the reference waveform. Voltage peaks can be observed along with a steady-state error. The effect of different G_s values on the switch current and voltage errors can be seen in Figures 5.23 and 5.24. As specified in [34], increasing G_s results in a better switch voltage representation whilst the current error grows.

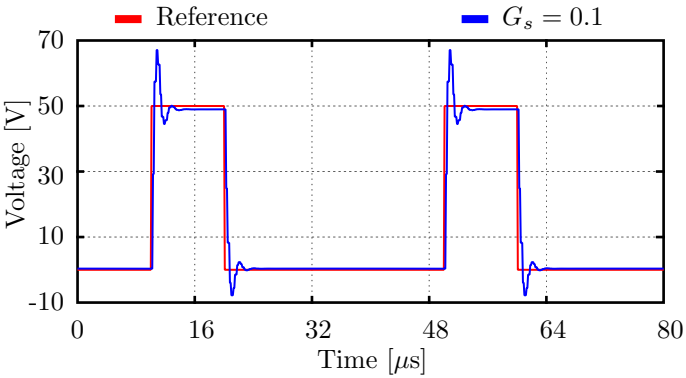


Figure 5.22 – Switch voltage waveforms comparison: reference 10ns time-step simulation vs eHS with 250ns time-step and $G_s = 0.1$.

As it can be seen in Figure 5.22, the error spikes were very significant – a 34% overvoltage at switch turn-off instants was present. The settling time of the voltage was around 2.5 μs, which equals to 10 time-steps. This observed error magnitude suggested further

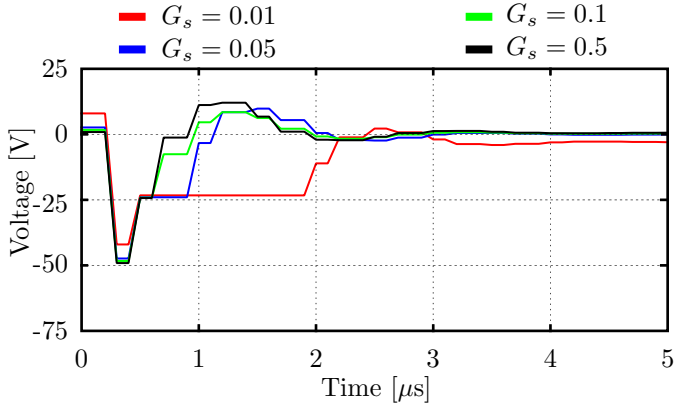


Figure 5.23 – G_s parameter comparison: voltage error.

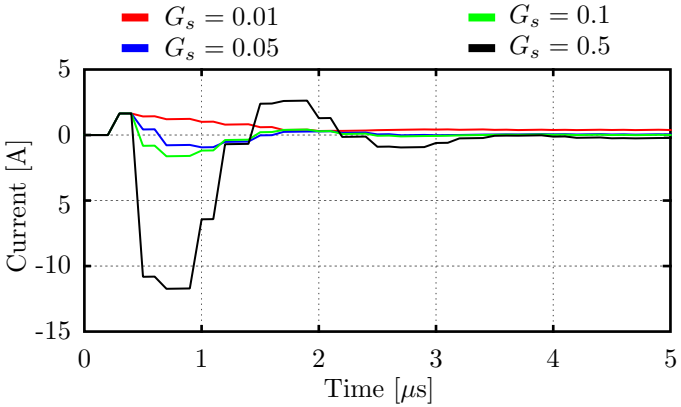


Figure 5.24 – G_s parameter comparison: current error.

inspection was needed, especially regarding the propagation to filtered outputs. Another battery of tests was thus prepared to visualize the sensitivity of the calculation results to the alteration of several parameters.

A buck converter simulation with the parameters shown in Table 5.2 was first conducted. The output voltage of a buck converter in continuous conduction mode is defined by $v_o = dv_i$; initial parameter values were hence chosen so the correct converter

gain was attained. Subsequently, new values of load resistance, switch conductance and switching frequency were defined and the effect on the output voltage measured. Continuous conduction operation was guaranteed for every parameter change.

Switching frequency (f_s)	20 kHz
Input voltage	50 V
Duty-cycle (d)	0.1
Output voltage (v_o)	5 V
Load resistance (R_o)	2 Ω
FPGA time-step	200 ns
Switch conductance (G_s)	0.1 S

Table 5.2 – Pejovic method test: initial buck converter parameters

Tables 5.3a and 5.3b (where column e contains the percent error in relation to the correct v_o value) reveal that the output voltage is heavily dependent on the load and switch conductance values, following a proportional pattern. A good choice of G_s for the buck converter thus have to take into consideration the expected load current.

R_o (Ω)	v_o (V)	e (%)
0.5	4.5	10
1	4.7	6
2	5	–
3	5.1	2
4	5.2	4

(a) $v_o \times R_o$

G_s (S)	v_o(V)	e (%)
0.05	4.7	5
0.075	4.9	2
0.1	5	–
0.15	5.2	4
0.2	5.3	5

(b) $v_o \times G_s$

Table 5.3 – Output voltage vs. resistive parameters

In Table 5.4 it can be seen that increasing the frequency also has a negative effect, but the results do not indicate it is related to the Pejovic method. At 40 kHz frequency, for example, there are 125

samples in a period, which translate to 0.008 duty-cycle or 0.4 V v_o steps. At 100 kHz, there are only 50 points – errors of up to 1 V (20 %) are therefore expected.

f_s (kHz)	v_o (V)	e (%)
10	5	–
20	5	–
30	5.1	2
40	5.2	4
50	5.2	4
75	5.4	8
100	5.8	16

Table 5.4 – Output voltage vs. switching frequency

Since voltage and current errors of the Pejovic switch model have a settling time, it was presumed that decreasing the duty-cycle too much can generate switching events while the error is still very large, resulting in immensely incorrect v_o values. In order to confirm that, Table 5.5 was generated with a 20 kHz switching frequency.

d	v_o (V)	e (%)
0.01	1.7	240
0.02	1.8	80
0.03	2.0	33
0.04	2.4	20
0.05	2.9	16
0.06	3.2	6.7
0.07	3.7	5.7
0.08	4.1	2.5
0.09	4.6	2.2

Table 5.5 – Output voltage vs. duty cycle ($f_s = 20$kHz)

The numerical errors produced by the Pejovic method were

thus shown to be large in certain conditions and very sensitive to alteration of circuit parameters. However, this is not enough to invalidate its purpose. Although gate signals generated by the controller when connected to the RTS model probably will not be equal to those generated for a real converter, it is reasonable to assume the correct controller operation can still be tested; the engineer just needs to interpret results with the method's characteristics in consideration. In the next section, hardware-in-the-loop tests are shown to corroborate the affirmation.

5.7 HARDWARE-IN-THE-LOOP

Three HIL simulations are discussed in this section to exemplify OP5700's capabilities. The first two system models were calculated by the FPGA and the last, by the CPU. The control strategy and the DSC code did not need to undergo any special adaptations for the RTS environment.

5.7.1 Buck converter

A simple HIL test was performed initially and consisted of a Buck converter with the parameters resumed on Table 5.6. Figure 5.25 shows a schematic of the implemented simulation. The DSC reads v_o from an analog output of the RT simulator, applies a basic proportional-integral control and synthesizes a PWM signal, which is fed to an OP5700 digital input.

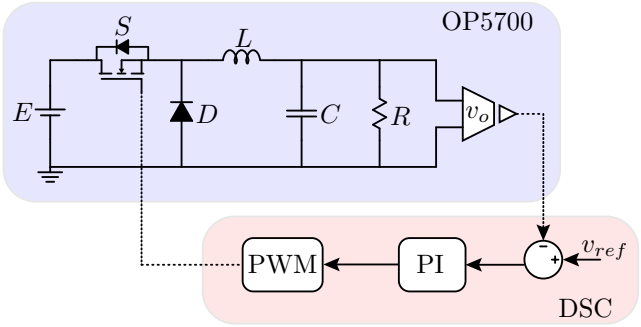


Figure 5.25 – Buck converter hardware-in-the-loop test.

Switching frequency	40 kHz
Input voltage	100 V
Load resistor	10 Ω
FPGA time-step	250 ns
CPU time-step[2]	20 μs

Table 5.6 – Buck converter HIL test parameters

With the real time simulation running, the gain of the PI controller (K_c) was altered on the fly by means of a variable in Texas Instruments' Code Composer Studio (CCS). Output voltage reference steps (25 to 75 V and back) were automatically applied each second, and the effects of the K_c variation on the control response were immediately seen in the oscilloscope. With a $K_c = 10^{-5}$ gain, the reference was reached in approximately 500 ms, as can be seen in Figure 5.26. Figure 5.27 shows that $K_c = 10^{-3}$ yielded a more than 100 times faster response. With $K_c = 5 \cdot 10^{-3}$ an oscillatory v_o started being observed (Figure 5.28).

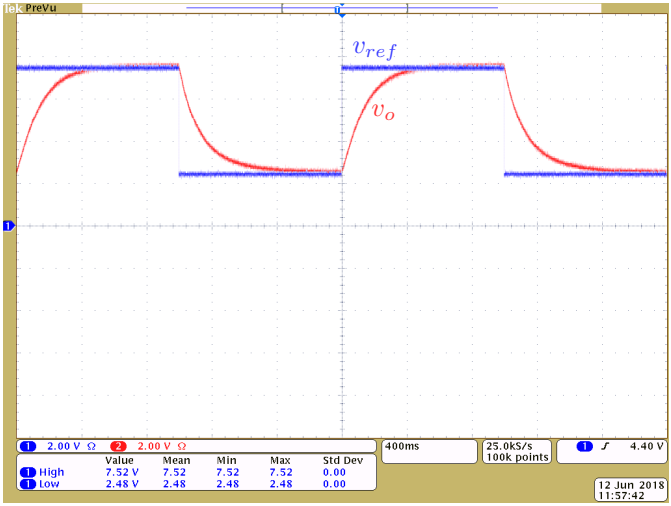


Figure 5.26 – Real-time controller parameter variation ($K_c = 10^{-5}$).

[2] The eHS block, which configures the FPGA, is a CPU-level Simulink block.

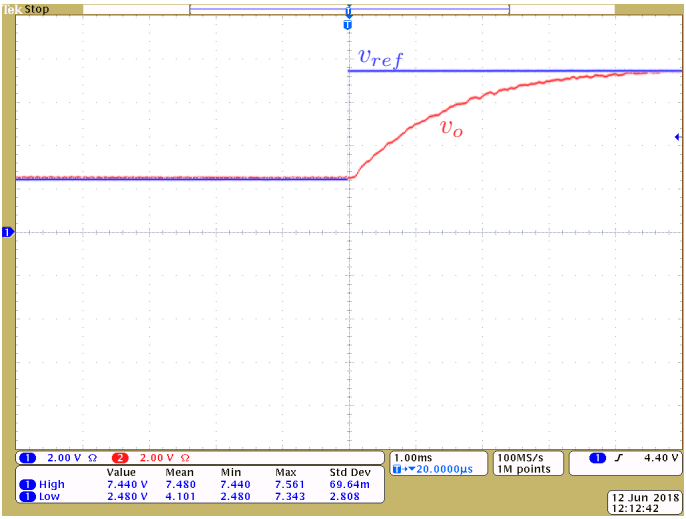


Figure 5.27 – Real-time controller parameter variation ($K_c = 10^{-3}$).

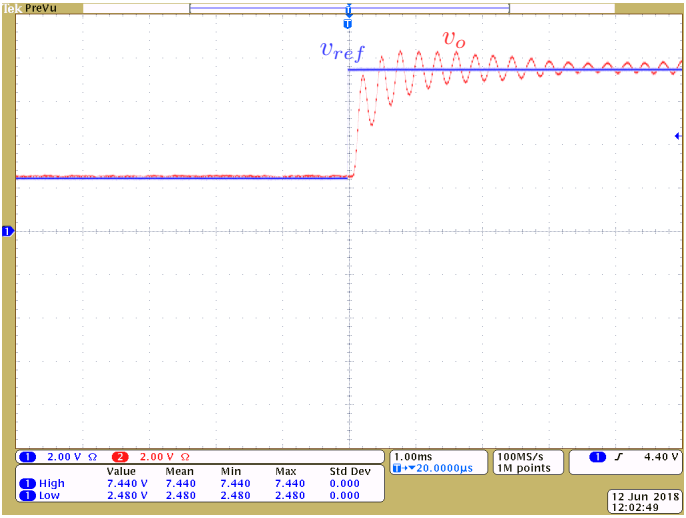


Figure 5.28 – Real-time controller parameter variation ($K_c = 5 \cdot 10^{-3}$).

5.7.2 Voltage source inverter (VSI)

The second simulation consisted of a three-phase VSI with an RL load and the parameters of Table 5.7; a schematic is shown in Figure 5.29. Load currents i_a and i_b are read by the DSC, which feeds back the switch gate signals after performing a simple $\alpha\beta$ current PI compensation [36].

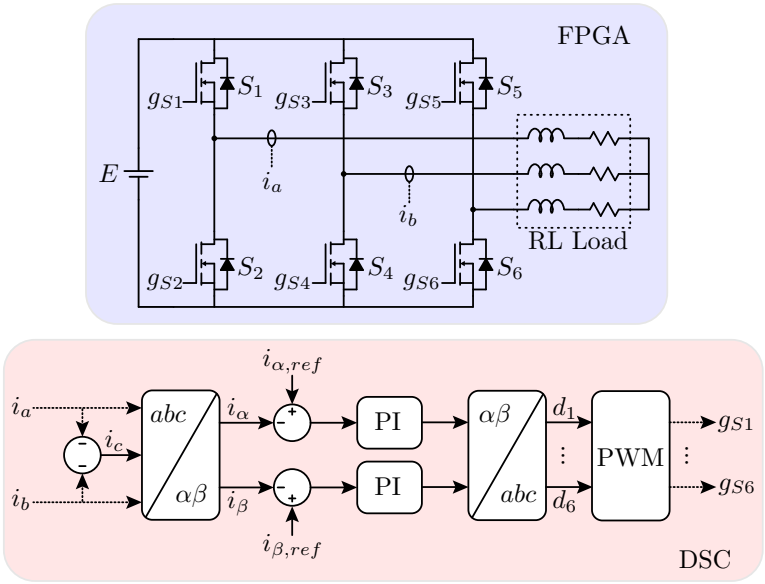


Figure 5.29 – Voltage source inverter hardware-in-the-loop test.

Modulating frequency	60 Hz
Switching frequency	20 kHz
Output current amplitude	10 A
Load resistor	5 Ω
Load inductor	1 mH
FPGA time-step	250 ns
CPU time-step	10 μs

Table 5.7 – Voltage Source Inverter HIL test parameters

Line voltages for a 200 V input DC voltage source can be seen in Figure 5.30 (0.01 sensor gain). The noise observed at the extremities is due to the introduced Pejovic switch voltage errors.

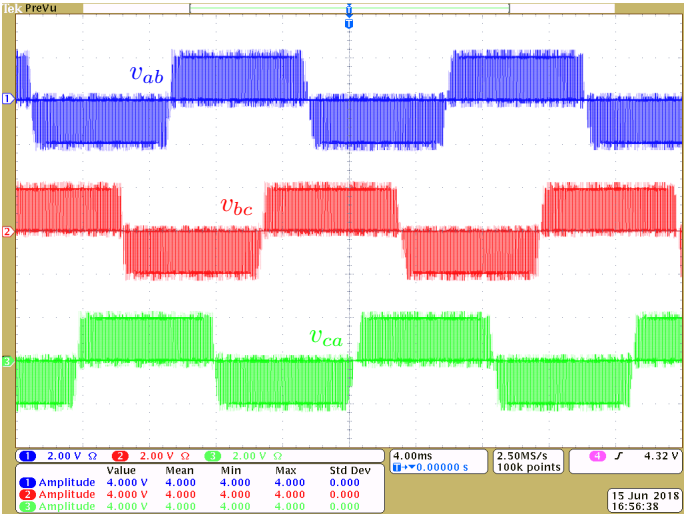


Figure 5.30 – Voltage source inverter - line voltages.

The resulting three-phase load currents are shown in Figure 5.31 (0.1 sensor gain). Figure 5.32 shows in detail the line voltages v_{ab}, v_{bc} and v_{ca} (CH1, CH2 and CH3 respectively) at the peak of phase current i_a (CH4). As expected, v_{ab} is almost complementary to v_{ca}, while v_{bc} approaches zero.

In order to verify the correct operation of the DSC control, a 125 to 275 V input voltage step was applied. After the perturbation, the phase currents settled in less than 1 ms, as can be seen in Figure 5.33.

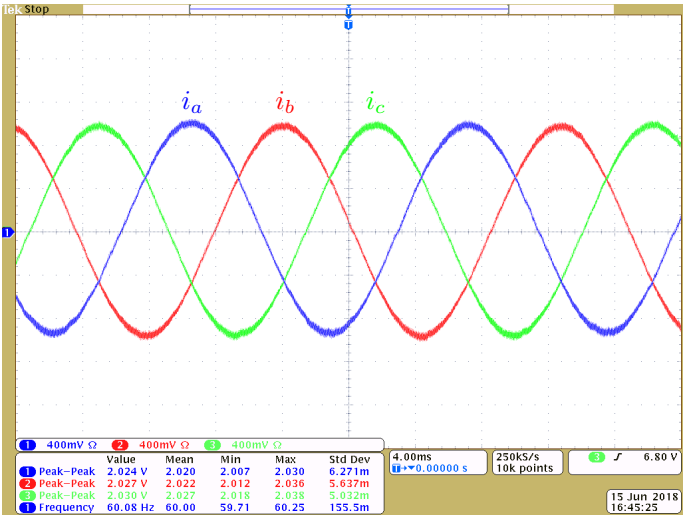


Figure 5.31 – Voltage source inverter - line currents.

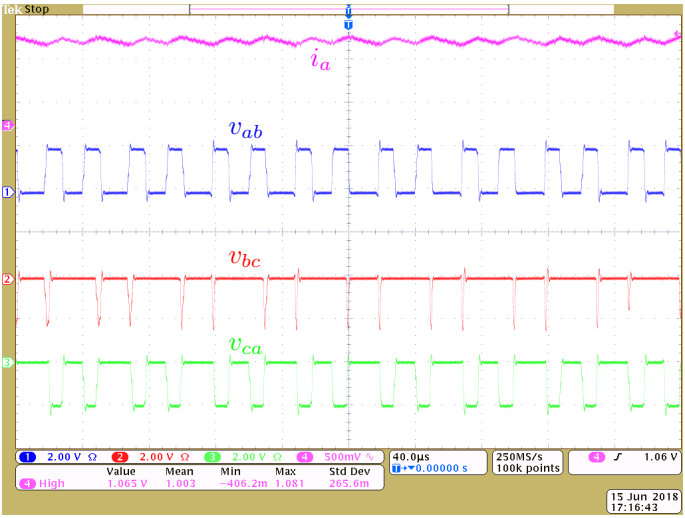


Figure 5.32 – Voltage source inverter - line voltages at the peak of i_a.

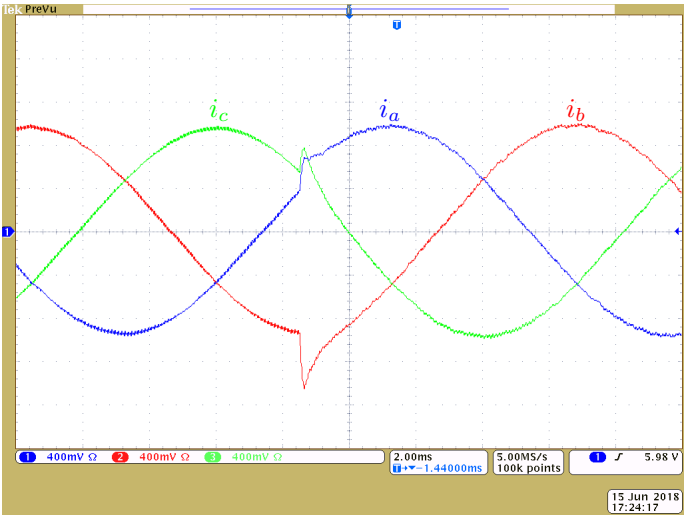


Figure 5.33 – Voltage source inverter - control response.

5.7.3 Motor drive with optimized space vector modulation

The schematic of the last HIL simulation can be seen in Figure 5.34, in which an induction motor is driven by an NPC converter with a synchronous optimum modulation (SOM) technique [37]. The power stage was simulated entirely on two cores of the CPU with a 10 μs time-step and consisted of the NPC converter, an LC filter, a transformer, a 10 km wideband line model[3] and the motor.

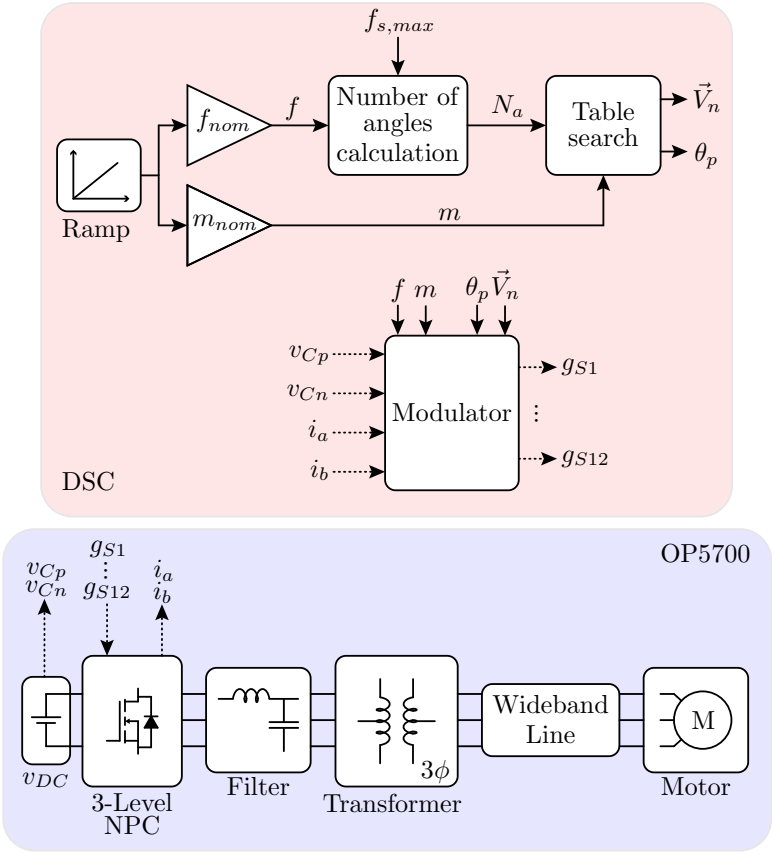


Figure 5.34 – Motor drive with SOM modulation HIL test.

[3] This model theory can be found on [38]. The necessary parameter matrices were calculated automatically with an EMTP-RV software routine by providing physical construction details of the cable.

In the DSC, a 20 seconds ramp is generated and multiplied by the nominal values of frequency and modulation index (60 Hz and 1, respectively). The maximum switching frequency ($f_{s,max} = 480$ Hz) is then divided by the frequency at the instant (f), and the floor of the result is taken as the number of angles (N_a). Since theoretically an infinite amount of angles would be needed to start the machine, conventional space vector modulation (SVM) is applied by the modulator until $m > 0.4$, which corresponds to 19 angles. With the N_a/m pair, an adequate vector $\vec{V}_N$ and its permanence angle θ_p are found within a pre-generated table. The entries of the table are optimized for the minimization of the THD of the motor currents.

If the difference of measured capacitor voltages v_{Cp} and v_{Cn} exceed 5% (absolute value) of the DC bus voltage, a balancing strategy starts being applied. When the difference reduces to 2%, balancing stops and a number of commutations reduction criteria is adopted. The DC source of the schematic is detailed in Figure 5.35. In order to demonstrate the capacitor voltage balancing strategy in operation, a switch (S_{im}) is used to force an imbalance. Currents i_a and i_b at the NPC's output are measured, therefore the direction of the current at the neutral point can be determined for each switching state. When redundant states are available, it is chosen the one that restores the voltage balance.

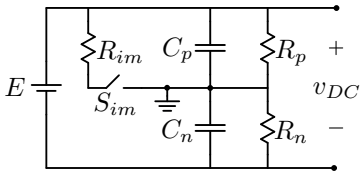


Figure 5.35 – DC bus detail.

A quadratic profile load is connected to the motor for the start-up process. Figure 5.36 shows the rotor speed during start-up. The electrical torque is presented on Figure 5.37 and the phase voltage v_a on Figure 5.38. The transition from SVM to SOM occurs at 9 seconds and no evident effects can be observed in any of the shown physical quantities.

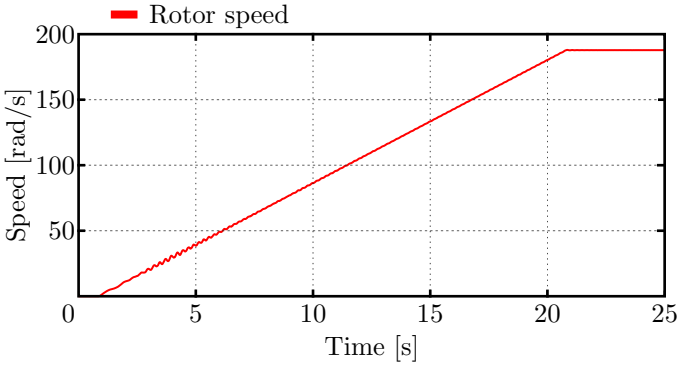


Figure 5.36 – Rotor speed during motor start-up.

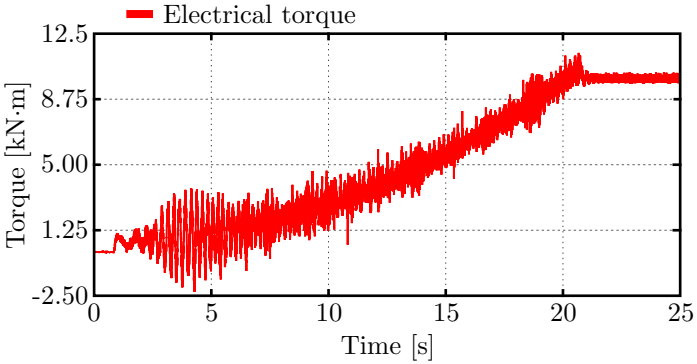


Figure 5.37 – Torque during motor start-up.

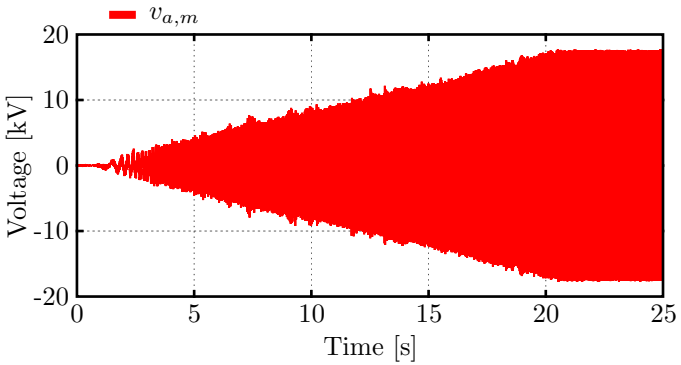


Figure 5.38 – Phase a voltage during motor start-up.

Figure 5.39 shows the v_a phase voltage (CH2) and the v_{ab} line voltage (CH1), with a 10^{-3} sensor gain. There are 8 switching events in a quarter phase wave, as expected for 60 Hz.

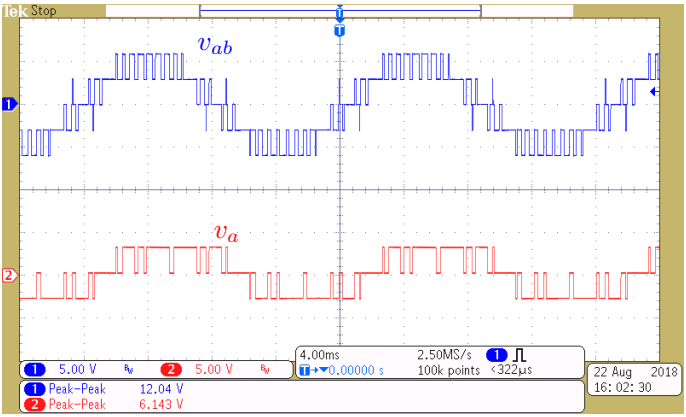


Figure 5.39 – Phase voltage v_a and line voltage v_{ab} at steady-state operation.

The unfiltered current (i_a) can be observed in CH1 of Figure 5.40 with a 0.01 sensor gain. A very low THD motor current ($i_{a,m}$) can be seen in CH2. The sensor gain for the motor current was

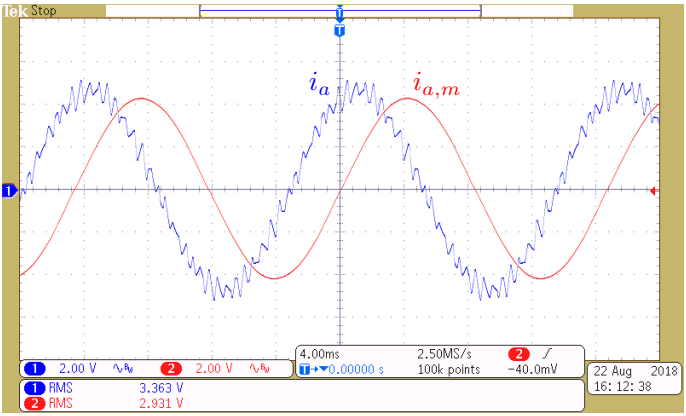


Figure 5.40 – NPC output current (i_a) and motor current ($i_{a,m}$) at steady-state operation.

multiplied by the transformation ratio (3.27) in order to equalize magnitudes.

Initially with the capacitor balancing strategy disabled, the S_{im} switch was closed. It can be seen In Figure 5.41 that v_{Cn} (CH1) rises 1 kV and v_{Cp} decreases; CH3 is the measure of the electrical torque, which starts oscillating due to the low frequency

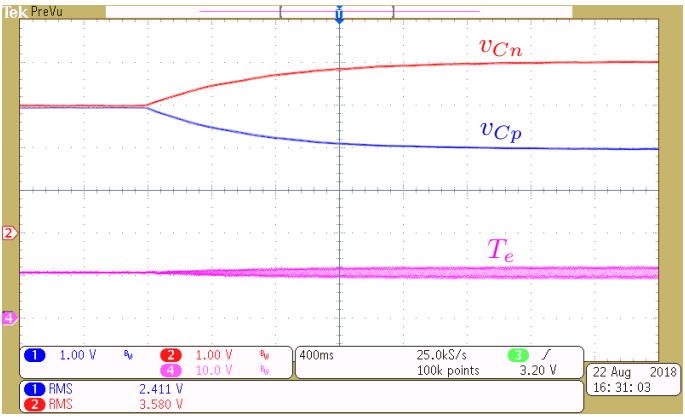


Figure 5.41 – Forced capacitor imbalance.

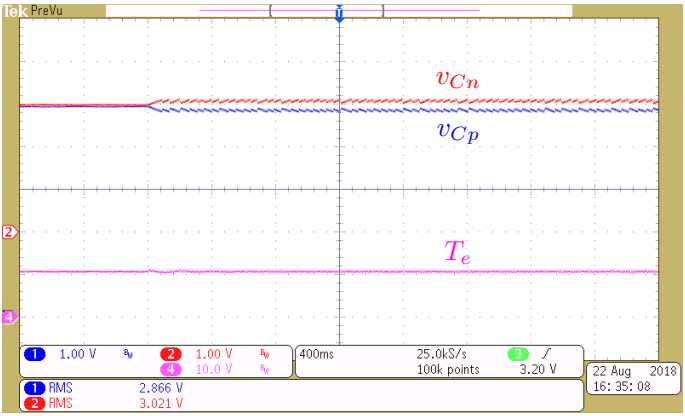


Figure 5.42 – Capacitor balancing activated.

harmonics generated by the unbalanced NPC DC bus. Figure 5.42 shows the capacitor voltages after the balancing strategy was enabled and S_{im} closed. The sawtooth shape is due to the hysteresis characteristic of the employed balancing strategy.

5.7.4 Additional comments

Despite spikes due to numerical errors appearing in the waveforms, the DSC control responded adequately in the shown FPGA tests and in other performed HIL simulations. Although the CPU-based test presented much more precise waveforms, it was only possible because the switching frequency was relatively low. Nevertheless, using the CPU allowed HIL testing a very complex model, which would not be executable with the eHS package.

In several opportunities, errors in the DSC implementation (coding mistakes) were detected and corrected. In a real converter test environment, some of those errors could mean the destruction of switches. Weighing the presented level of accuracy, the needed effort for implementation and the prompt access to responses, the tests were considered very satisfactory for preliminary controller evaluations.

5.8 CHAPTER REVIEW

- A signal conditioning board was designed for the exchange of digital and analog signals between the OP5700 real-time simulator and Texas Instruments LaunchPad DSCs.

- The reduction in calculation time by the substitution of the SPS SVA algorithm by the SSN was significant in the test conducted. The best result possible was with at least 6 CPU cores and less than 13% of the SPS time was required. This is critical for enabling real-time operation in many cases. It was also shown that the speed gain did not cause loss of accuracy.

- In the eHS test with an MMC model, a 28-fold speed increase was observed versus a relatively fast PC CPU. Accuracy tests

however revealed that the Pejovic algorithm produces large error spikes after switch state transitions, which can propagate to the filtered outputs. In the buck converter test there was a significant error sensitivity to the alteration of G_s and the magnitude of the load current. By comparing the output of the eHS Offline block with a reference model, optimal values of G_s can be chosen to obtain more precise results at nominal operation; this is only feasible for a small quantity of switches, however.

- In the RTE package test a deliberately high switching frequency was chosen so SPS' diode bridge model would produce numerical errors and jitter in the output current. The creation of RTE-type gate signals with the aid of the FPGA's fast sampling and the compensation applied to the output voltage produced by RTE's bridge model resulted in a much better represented output current.

- Despite the errors introduced by the Pejovic algorithm, the controller responses in the FPGA HIL tests were as expected. Using the CPU allowed for testing a complex model, although the switching frequency was relatively small. The benefits of hardware-in-the-loop type simulations could be experienced and the adequate operation of the OP5700 real-time simulator confirmed.

Conclusion

This master's thesis is the result of a study conducted on real-time simulation methods and the practical utilization of the RTS platform OPAL-RT OP5700. Fundamental theoretical aspects that support modern general purpose simulation softwares and some details of OPAL-RT's approach on the processing of RTS models were first outlined. Finally, an evaluation of OPAL-RT's methods and the capabilities of the OP5700 system was done.

The modeling process of a general purpose simulation software starts with the user input. This is done nowadays by means of a graphical user interface in which circuit element symbols are placed and interconnected. As seen in the fundamental concepts of graph theory, this input is converted to a mathematical topological representation, the incidence matrix. With the aid of a tree, Kirchoff's Voltage and Current laws can be applied to the circuit by means of the loop and cutset matrices. The generation of the topological matrices are an important step included in circuit analysis methods.

Chapters 2 and 3 presented the dominant circuit analysis methods utilized in computer simulations: the State Variable Analysis and the Nodal Analysis. A method may be more advantageous than the other depending on the intended simulation, as the the following comparisons summarize:

System formulation: Linearly dependent energy-storage elements must be identified for the SVA equations formulation, which is

a non-trivial and resource-demanding process. In contrast, the nodal equation is formed with associated discrete models, and even sets of predetermined stamps can be simply summed to obtain it.

Use of variable time-step: The admittance matrix of the nodal equation contains several entries that depend on the time-step, resulting from the elements' discretization. Any change in the time-step requires therefore a thorough reformulation, adding significant overhead. Conversely, the SVA equations are defined before the discretization, which can be done at the solution stage and must include the time-step value nevertheless.

Numerical solution: The nodal analysis demands the calculation of the algebraic $\boldsymbol{YV} = \boldsymbol{I}$ equation. This can be efficiently done with straightforward methods such as LU factorization or gaussian elimination. Discretized SVA systems can be solved in a similar manner. For continuous SVA systems, however, differential equation solving – with predictor-corrector or determination of matrix exponentials base methods, for example – is needed.

If some prerequisites are met, a circuit model can be simulated in real-time. The most important is the synchronization of simulation time-steps with a real-world clock; this allows for a dynamic response equivalent to a real circuit counterpart. To ensure a faithful and synchronized simulation, every calculation must finish before a new time-step begins. As the complexity of the circuit or the highest frequencies involved grow, so does the importance of fast algorithms and parallel processing. The specialized algorithms in OPAL-RT's package are the Pejovic method (utilized for simulations on the FPGA) and the SSN method (for CPU simulations).

When digital and analog input/output capabilities are available in the computer running the RTS, real equipment can be added as part of the simulation, characterizing the hardware-in-the-loop approach. Among the most useful applications is the testing of controllers; problems in the implementation can be found much earlier in the design process since the controller operation can be checked even before the real plant is available. Also, depending on the plant, using it for tests can be expensive, risky, or even impractical.

For the examination of the OP5700 RTS system and the mathematical methods it utilizes, comparison tests in Simulink between the standard SPS state-space method and OPAL-RT's implementations in terms of speed and accuracy were performed. The SSN solver, which calculates subparts of the circuit in parallel without adding delays, was almost 8 times as fast as the SPS solver for the same level of accuracy. The advantage, however, was substantial because the tested model had many switches and energy-storage elements; the calculation times are similar in the case of small circuits.

Whereas time-steps of the 10 μs order are tipically the minimum possible for overrun-free CPU simulations – even for relatively simple models – the FPGA-oriented eHS package allowed time-step values down to 150 ns. On the other hand, the Pejovic method, utilized for obtaining a constant admittance matrix, has shown poor levels of accuracy in some cases. Although the preliminary tests reported in this master's thesis used a single G_s value for each simulation, OPAL-RT allows G_s to be set individually for each switch. An interesting study thus would be accessing the feasibility of an output error minimization technique for those values, and also if combining proper G_s values with extra resistances and snubber circuits can accelerate the switch error damping and reduce propagation.

While the characteristics of the Pejovic method or the minimum time-step achievable with CPU simulations impede a detailed representation of impulsive voltages and currents, those limitations do not cause significant problems for HIL testing, as it was verified. Furthermore, the immense practicality of HIL simulations was experienced: controller behavior was within the expected, and circuit parameters could be changed on the fly with the implications instantaneously observable.

It is hoped that a new OP5700 user (or even other RTS systems), supported by the information presented in this master's thesis, will be able to better discern what methods and parameters fit best an intended model, and what elements can be left out in order to reduce the simulation time-step as much as possible without overruns or loss of accuracy.

Bibliography

1 BÉLANGER, J.; VENNE, P.; PAQUIN, J. The what, where, and why of real-time simulation. p. 37–49, 01 2010. Cited 4 times in pages 32, 87, 89, and 91.

2 PETERSON, H. A. An electric circuit transient analyser. *GE Review*, v. 42, p. 394, 1939. Cited in page 32.

3 HANSELMANN, H. Hardware-in-the-loop simulation testing and its integration into a CACSD toolset. In: *Proceedings of Joint Conference on Control Applications Intelligent Control and Computer Aided Control System Design*. 1996. p. 152–156. Cited in page 33.

4 ENISZ, K. et al. Reconfigurable real-time hardware-in-the-loop environment for automotive electronic control unit testing and verification. *IEEE Instrumentation Measurement Magazine*, v. 17, n. 4, p. 31–36, Aug 2014. ISSN 1094-6969. Cited in page 33.

5 LUO, K.; SHI, W.; TANG, H. Implementation and value of power hardware in the loop testing bed for wind turbines integrated into grid. *The Journal of Engineering*, v. 2017, n. 13, p. 1635–1639, 2017. Cited in page 33.

6 GILIBERTI, G. et al. Applying hardware in the loop to designing, integrating, verifying and validating the control system of new aircraft engines. In: *EVI-GTI Conference on Gas Turbine Instrumentation (GTI 2017)*. 2017. p. 1–6. Cited in page 33.

7 WANG, J. et al. Development of a universal platform for hardware in-the-loop testing of microgrids. *IEEE Transactions on Industrial Informatics*, v. 10, n. 4, p. 2154–2165, Nov 2014. ISSN 1551-3203. Cited in page 33.

8 CHUA, L. O.; LIN, P. Y. *Computer-Aided Analysis of Electronic Circuits: Algorithms and Computational Techniques*. Englewood Cliffs, NJ, USA: Prentice Hall Professional Technical Reference, 1975. ISBN 0131654152. Cited 10 times in pages 35, 44, 45, 46, 49, 55, 60, 69, 70, and 72.

9 IOINOVICI, A. *Computer-aided analysis of active circuits*. New York, NY, USA: Marcel Dekker, Inc., 1990. ISBN 0824781260. Cited 3 times in pages 35, 42, and 48.

10 SILVA, A. R. de J. *Programa scvolt: simulação numérica de conversores estáticos, método do voltímetro, empregando a técnica do passo de cálculo variável*. Master's thesis — Universidade Federal de Santa Catarina, 2003. Cited in page 35.

11 SESHU, S.; BALABANIAN, N. *Linear Network Analysis*. J Wiley, 1959. Cited in page 35.

12 RAJAGOPALAN, V. *Computer-aided analysis of power electronic systems*. CRC Press, 1987. Cited in page 35.

13 VLACH, J.; SINGHAL, K. *Computer methods for circuit analysis and design*. Springer Science & Business Media, 1983. Cited 2 times in pages 35 and 81.

14 WATSON, N.; ARRILLAGA, J. *Power Systems Electromagnetic Transients Simulation*. Institution of Engineering and Technology, 2003. (Energy Engineering). Available at: <http://digital-library.theiet.org/content/books/po/pbpo039e>. Cited 3 times in pages 35, 55, and 58.

15 ARRILLAGA, J.; ARNOLD, C. P. *Computer Analysis of Power Systems*. JOHN WILEY & SONS INC, 1991. ISBN 0471927600. Cited 4 times in pages 35, 53, 77, and 93.

16 Hydro-Quebéc, TransÉnergie Technologies. *SimPowerSystems User's Guide*. <http://www.mathworks.com/help/releases/R13sp2/pdf_doc/physmod/powersys/powersys.pdf>. Visited on 03/08/2018. Cited 2 times in pages 53 and 62.

17 DUFOUR, C.; MAHSEREDJIAN, J.; BéLANGER, J. A combined state-space nodal method for the simulation of power system transients. *IEEE Transactions on Power Delivery*, Institute of Electrical and Electronics Engineers (IEEE), v. 26, n. 2, p. 928–935, apr 2011. Available at: <https://doi.org/10.1109%2Ftpwrd.2010.2090364>. Cited 4 times in pages 54, 96, 97, and 98.

18 CHAMPAGNE, R. *Simulation en temps réel à l'aide de la représentation d'état : application à un entraînement électrique basé sur une machine asynchrone*. Dissertation (Ph.D.) — École de technologie supérieure, Montréal, 2001. Cited in page 63.

19 BRANIN JR., F. H. Computer methods of network analysis. In: *Proceedings of the 4th Design Automation Conference*. New York, NY, USA: ACM, 1967. (DAC '67), p. 8.1–8.19. Available at: <http://doi.acm.org/10.1145/800270.810863>. Cited in page 69.

20 DOMMEL, H. Digital computer solution of electromagnetic transients in single-and multiphase networks. *IEEE Transactions on Power Apparatus and Systems*, Institute of Electrical and Electronics Engineers (IEEE), PAS-88, n. 4, p. 388–399, apr 1969. Available at: <https://doi.org/10.1109%2Ftpas.1969.292459>. Cited 2 times in pages 69 and 73.

21 HO, C.; RUEHLI, A.; BRENNAN, P. The modified nodal approach to network analysis. *IEEE Transactions on Circuits and Systems*, v. 22, n. 6, p. 504–509, June 1975. ISSN 0098-4094. Cited 2 times in pages 70 and 80.

22 STEWART, J. *Calculus*. BROOKS COLE PUB CO, 2015. ISBN 1285740629. Cited in page 73.

23 MARTI, J. R.; LIN, J. Suppression of numerical oscillations in the emtp power systems. *IEEE Transactions on Power Systems*, v. 4, n. 2, p. 739–747, May 1989. ISSN 0885-8950. Cited in page 80.

24 MAHSEREDJIAN, J. et al. On a new approach for the simulation of transients in power systems. *Electric power systems research*, Elsevier, v. 77, n. 11, p. 1514–1520, 2007. Cited in page 80.

25 WARWICK, C. In a nutshell: How SPICE works. *IEEE EMC Soc. Newsletter*, n. 22, 2009. Cited in page 80.

26 MULTISIM Help – Nodal Analysis and Matrix Solving. <http://zone.ni.com/reference/en-XX/help/375482B-01/multisim/nodalanalysismatrixsolving/>. Visited on 11/07/2018. Cited in page 80.

27 IEEE Technical Committee on Real-Time Systems - Terminology and notation. <http://sites.ieee.org/tcrts/education/terminology-and-notation/>. Visited on 25/07/2018. Cited in page 87.

28 dSPACE Prototyping Systems. <https://www.dspace.com/en/pub/home/products/systems/functp.cfm>. Visited on 26/09/2018. Cited in page 89.

29 BLANCHETTE, H. F.; Ould-Bachir, T.; DAVID, J. P. A state-space modeling approach for the fpga-based real-time simulation of high switching frequency power converters. *IEEE Transactions on Industrial Electronics*, IEEE, v. 59, n. 12, p. 4555–4567, 2012. Cited 2 times in pages 89 and 91.

30 PARMA, G. G.; DINAVAHI, V. Real-time digital hardware simulation of power electronics and drives. *IEEE Transactions on Power Delivery*, IEEE, v. 22, n. 2, p. 1235–1246, 2007. Cited in page 89.

31 MAJSTOROVIC, D. et al. Ultralow-latency hardware-in-the-loop platform for rapid validation of power electronics designs. *IEEE Transactions on Industrial Electronics*, v. 58, n. 10, p. 4708–4716, Oct 2011. ISSN 0278-0046. Cited in page 91.

32 GOLE, A. M. et al. Guidelines for modeling power electronics in electric power engineering applications. *IEEE Transactions on Power Delivery*, v. 12, n. 1, p. 505–514, Jan 1997. ISSN 0885-8977. Cited in page 91.

33 MATAR, M.; IRAVANI, R. Fpga implementation of the power electronic converter model for real-time simulation of electromagnetic transients. *IEEE Transactions on Power Delivery*, IEEE, v. 25, n. 2, p. 852–860, 2010. Cited in page 91.

34 PEJOVIC, P.; MAKSIMOVIC, D. A method for fast time-domain simulation of networks with switches. *IEEE Transactions on Power Electronics*, v. 9, n. 4, p. 449–456, Jul 1994. ISSN 0885-8993. Cited 2 times in pages 99 and 120.

35 MCGRATH, B. P.; HOLMES, D. G. Multicarrier pwm strategies for multilevel inverters. *IEEE Transactions on Industrial Electronics*, v. 49, n. 4, p. 858–867, Aug 2002. ISSN 0278-0046. Cited 2 times in pages 111 and 116.

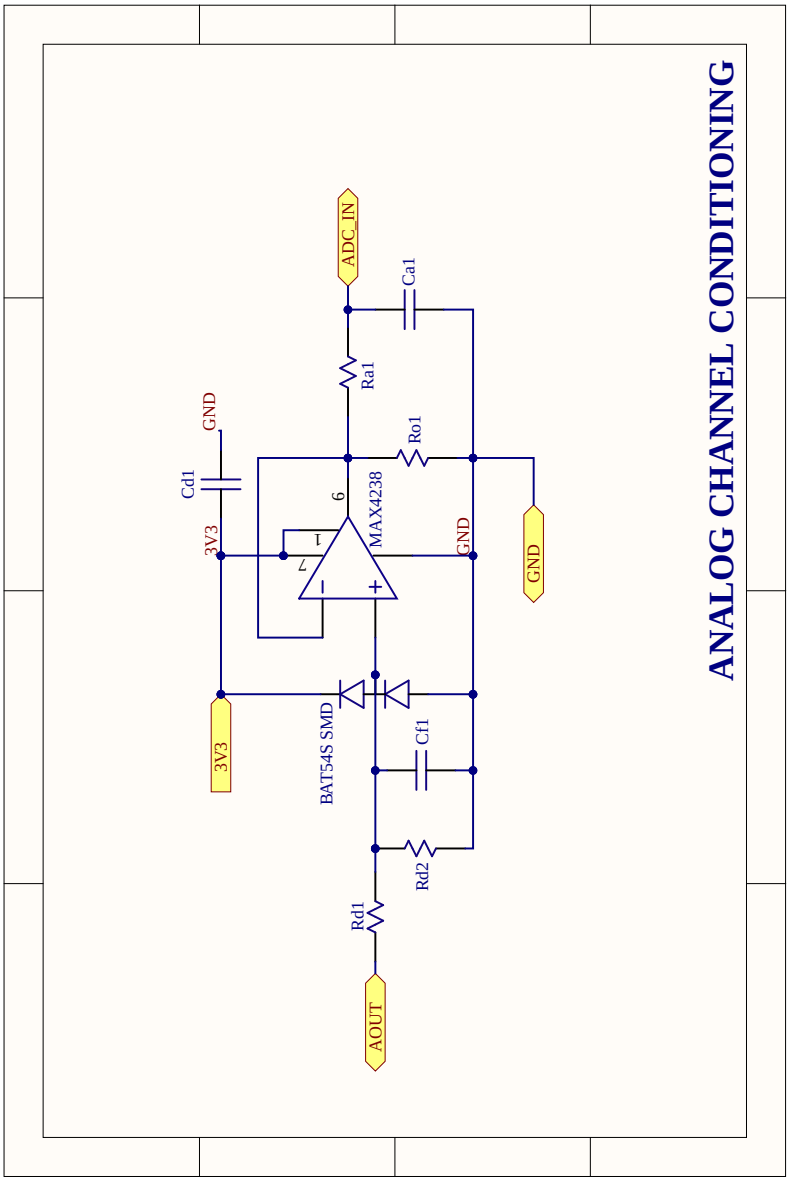
36 TEODORESCU, R.; LISERRE, M.; RODRIGUEZ, P. *Grid converters for photovoltaic and wind power systems*. John Wiley & Sons, 2011. v. 29. Cited in page 127.

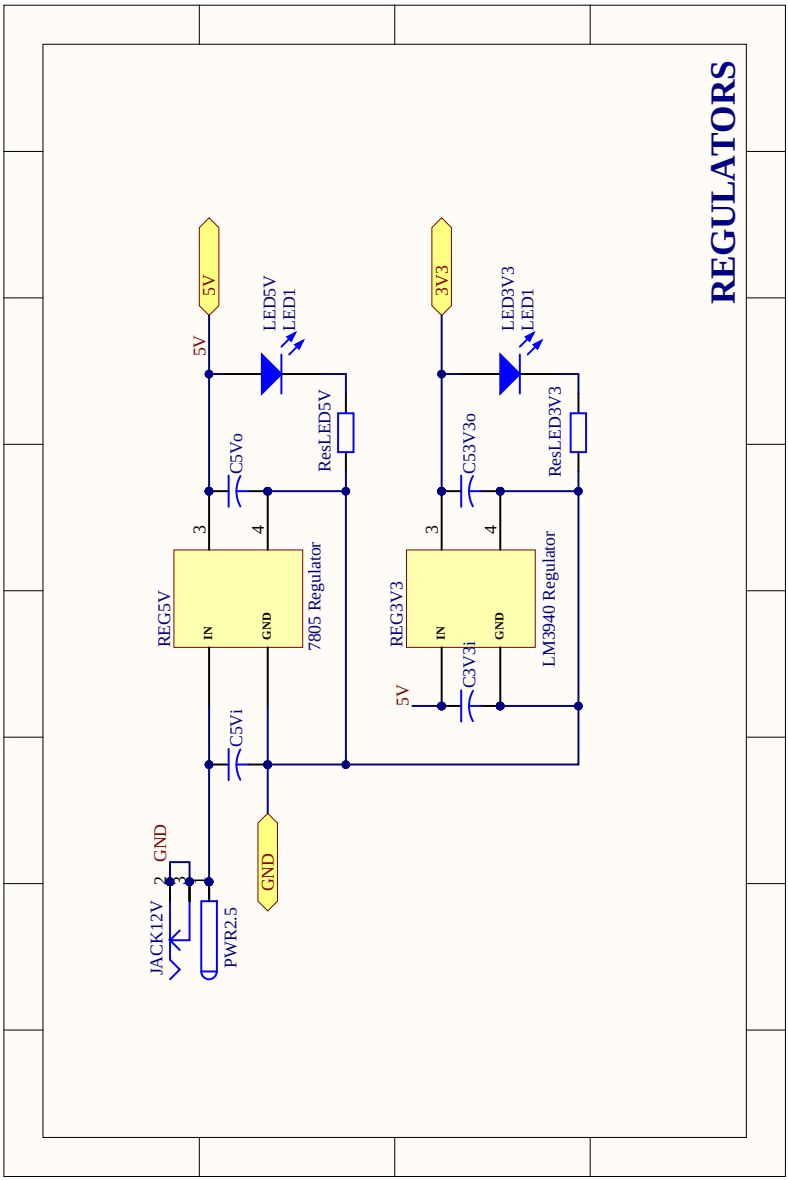
37 LAGO, J. *Técnicas de modulação síncrona otimizada para a melhoria de desempenho de conversores multiníveis no acionamento de máquinas elétricas*. Dissertation (Ph.D.) — Universidade Federal de Santa Catarina, Santa Catarina, Brazil, 2015. Cited in page 131.

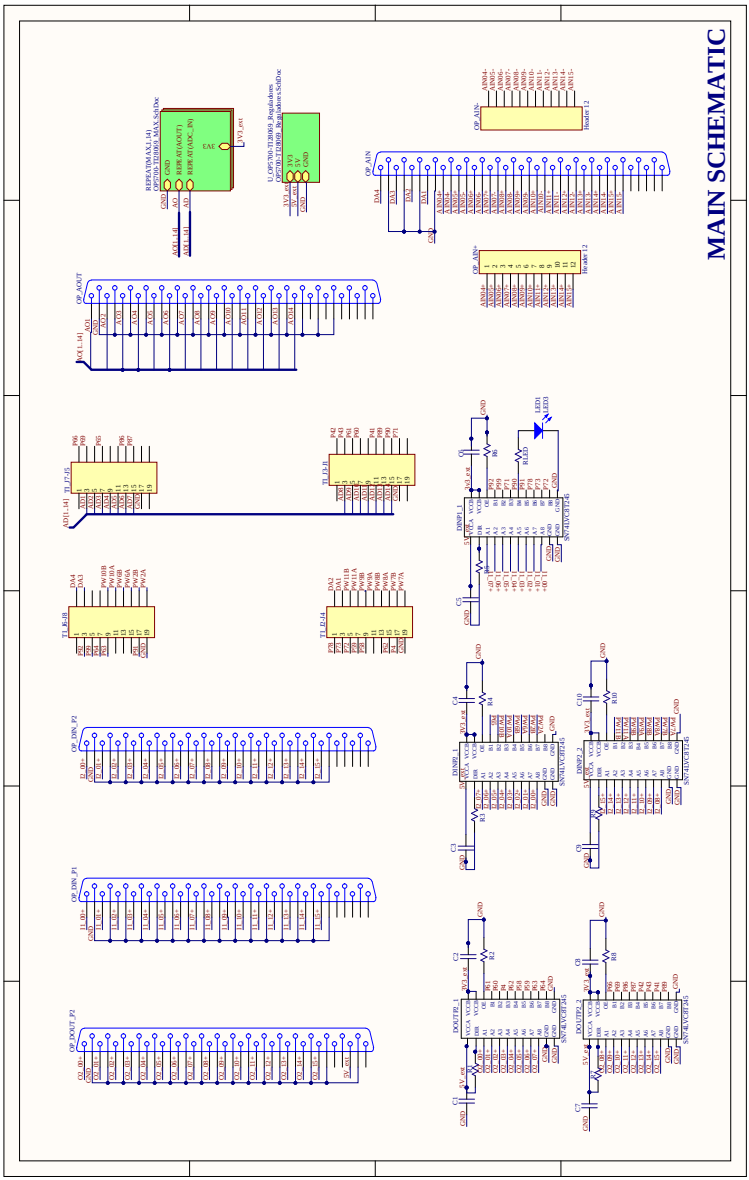
38 RAMOS-LEANOS, O. et al. A wideband line/cable model for real-time simulations of power system transients. *IEEE Transactions on Power Delivery*, v. 27, n. 4, p. 2211–2218, Oct 2012. ISSN 0885-8977. Cited in page 131.

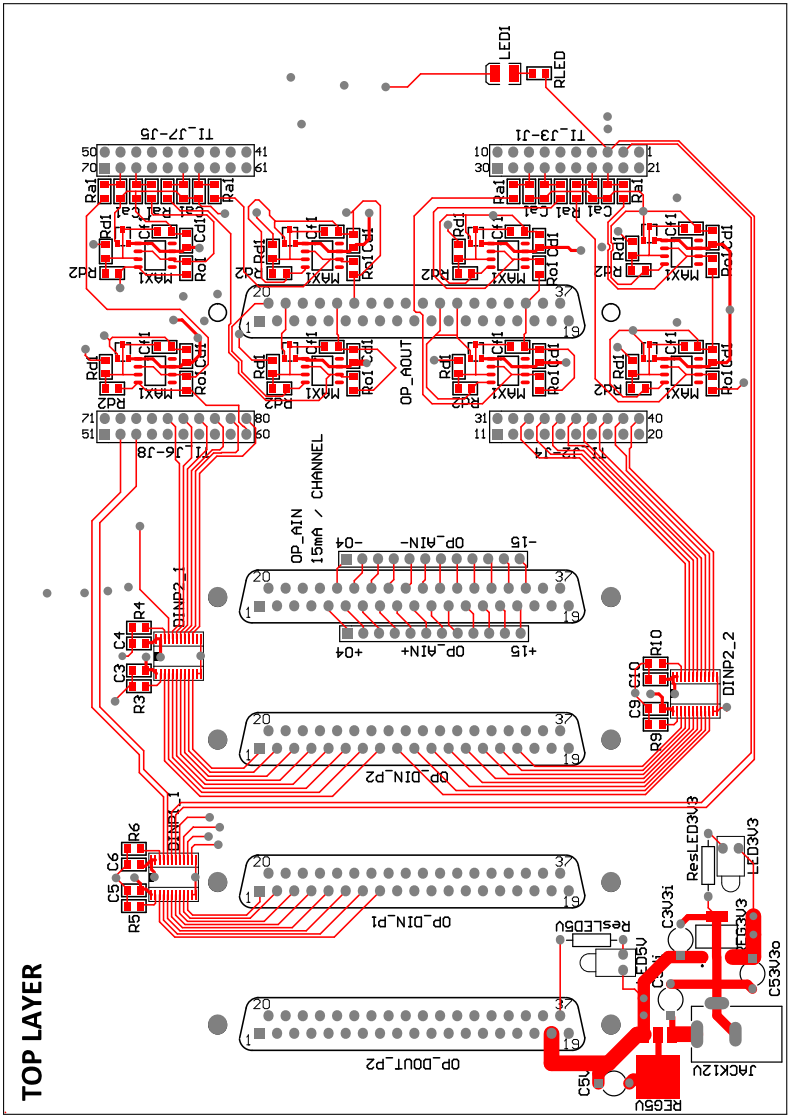
Appendix A

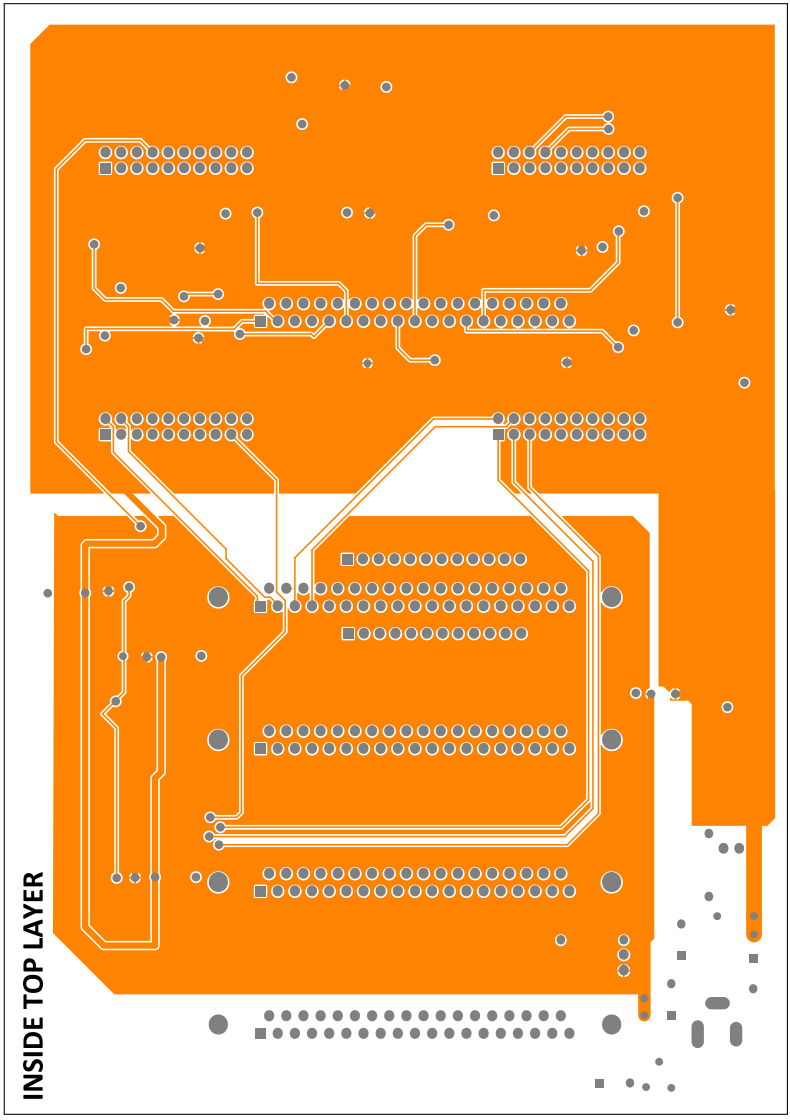
Board schematics and PCB

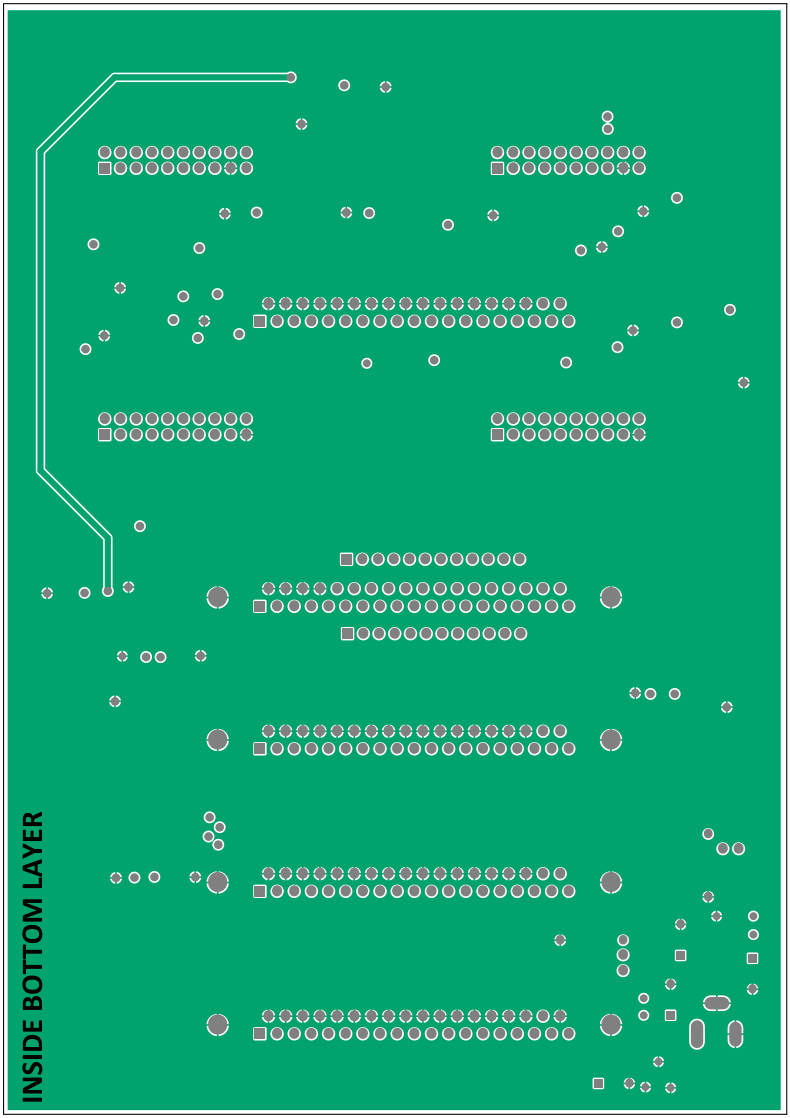

ANALOG CHANNEL CONDITIONING
ADC_IN
Ca1
Ra1
Ro1
GND
Cd1
3V3
6
MAX4238
1
7
GND
GND
3V3
BAT54S SMD
Cf1
Rd2
Rd1
AOUT

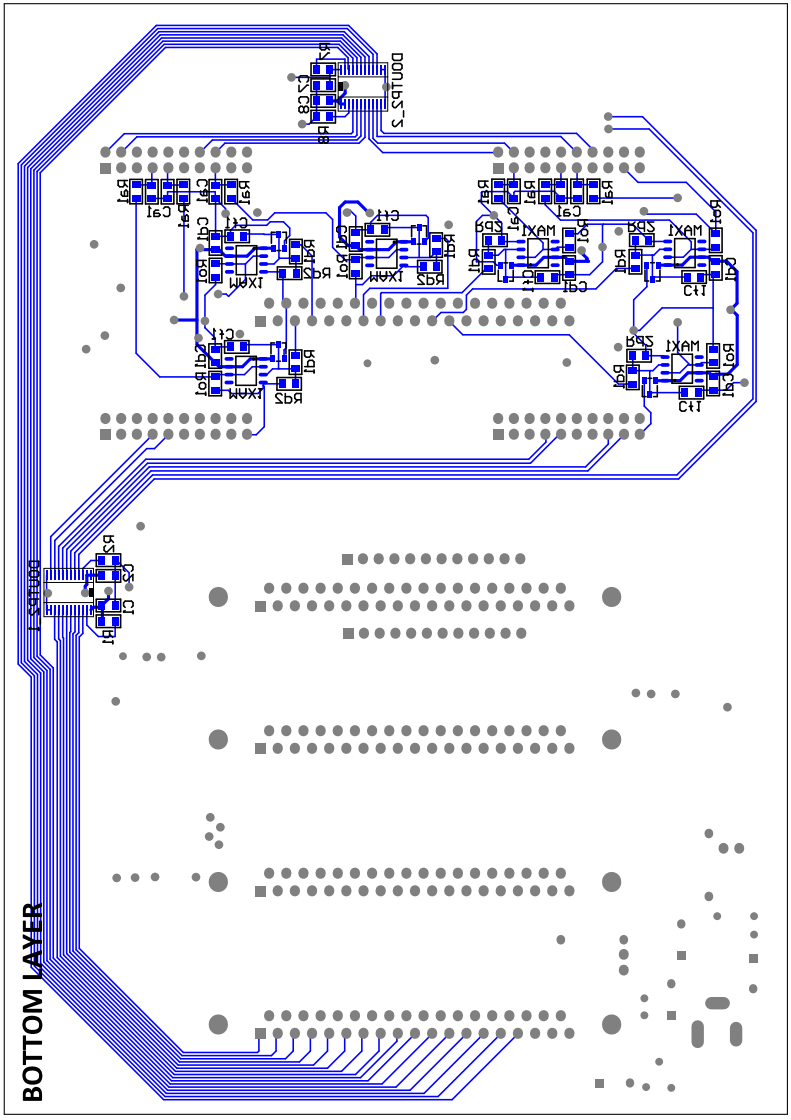

REGULATORS
5V
LED5V
LED1
5V
C5Vo
ResLED5V
3
4
REG5V
IN
GND
7805 Regulator
C5Vi
GND
JACK12V
2
3
GND
PWR2.5
3V3
LED3V3
LED1
C53V3o
ResLED3V3
3
4
REG3V3
IN
GND
LM3940 Regulator
5V
C3V3i


MAIN SCHEMATIC


TOP LAYER
LED1
RLED
TI_J3-J1
TI_J7-J5
TI_J6-J8
TI_J2-J4
OP_AOUT
OP_AIN
15mA / CHANNEL
OP_AIN-
OP_AIN+
OP_DIN_P2
OP_DIN_P1
OP_DOUT_P2
DINP2_1
DINP2_2
DINP1_1
ResLED3V3
LED3V3
ResLED5V
LED5V
C3V3i
C3V3o
C5V
REG5V
JACK12V


INSIDE TOP LAYER


INSIDE BOTTOM LAYER


BOTTOM LAYER