Power Converter Emulation on Reconfigurable Hardware Using Device-Level Behavioral Modeling

by

Bo Shi

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Energy Systems

Department of Electrical and Computer Engineering
University of Alberta

Abstract

Precise models of power electronic converters significantly improve the fidelity of hardware-in-the-loop (HIL) simulators, thereby accelerating and reducing costs of design cycles in industrial applications. This thesis proposes detailed device-level hardware models of the IGBT and the power diode, for emulating power electronic converters on the field programmable gate array (FPGA). The hardware emulation utilizes detailed nonlinear behavioral models of these devices, and features a paralleled and fully pipelined implementation using an accurate floating-point data representation. Test cases for simple device-level power electronic circuits and a half-bridge converter are emulated on FPGA. A modular multi-level converter circuit using the proposed models is also emulated. The captured oscilloscope results demonstrate high accuracy of the emulator in comparison to the off-line simulation of the original test systems.

Acknowledgements

First and foremost, I would like to express my earnest gratitude to my supervisor, *Dr. Venkata Dinavahi*, for his full support, continual inspiration and insightful guidance during these years at the University of Alberta. I have been deeply influenced and motivated by his persistent, conscientious, enthusiastic attitude to research. Without his unlimited patience, and constant confidence in me, this work could never have been completed.

It is an honor for me to extend my gratitude to my M.Sc. committee members *Dr. John Salmon* and *Dr. Douglas Barlage* for reviewing my thesis and providing valuable suggestions. Furthermore, I would like to show my cordial thanks to all my colleagues and friends in the RTX-Lab for their care and help.

Finally, I want to extend my sincerest appreciation to my parents and my girlfriend. Without their tremendous support, endless love, constant encouragement, deeply tolerant understanding, I would never have finished this work.

Table of Contents

List of Tables

List of Figures

List of Acronyms

AC	Alternating Current
A/D	Analog to Digital Converter
ABM	Analog Behavioral Modeling
BJT	Bipolar Junction Transistor
CLB	Configurable Logic Block
CMT	Clock Management Tiles
CPU	Central Processing Unit
DAC	Digital to Analog Converter
DC	Direct Current
DCI	Digitally-controlled Impedance
DRC	Design Rule Check
DSP	Digital Signal Processing
EMT	Electro-Magnetic Transient
FDM	Finite Difference Method
FEM	Finite Element Method
FF	Flip Flop
FIFO	First In First Out
FPGA	Field-Programmable Gate Array
GPU	Graphic Processing Unit
HIL	Hardware-in-the-Loop
HLS	High Level Synthesis
HP	High-performance
HR	High-range
HUT	Hardware under Test
HVDC	High-Voltage Direct Current

IC	Integrated Circuit
IDE	Integrated Design Environment
IGBT	Insulated-Gate Bipolar Transistor
I/O	Input/Output
IOB	Input/Output Block
IP	Intellectual Property
LAE	Linear Algebraic Equation
IGBT	Insulated-Gate Bipolar Transistor
LTE	Local Truncation Error
JTAG	Joint Test Action Group
LUT	Look-Up Table
MNA	Modified Nodal Approach
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor
MMC	Modular Multi-Level Converter
NAE	Nonlinear Algebraic Equation
NR	Newton-Raphson
ODE	Ordinary Differential Equation
PCIe	Peripheral Component Interconnect Express
PLL	Phase Locked Loop
PWLD	Piecewise Linear Diode
PWM	Pulse Width Modulation
REF	Row Echelon Form
RREF	Reduced Row Echelon Form
RTL	Register Transfer Level
(S)RAM	(Static) Random Access Memory
SDP	Simple Dual-Port
TDP	True Dual-Port
(V)HDL	(Very-High-Speed Integrated Circuit) Hardware Description Language
VLA	Vivado® Logic Analyzer
VLSI	Very Large Scale Integration
VSC	Voltage Source Converter

VTCM	Variable Time-Step Control Module
VTOM	Variable Time-Step Output Module

1
Introduction

Detailed physics-based *analytical* device-level models for IGBTs are available in the literature, the most prevalent of which are the Hefner [1–4] and Kraus models [5–7]. These models describe the nonlinear physical phenomena associated with each region of the device structure. Highly exact *numerical* models [8, 9] based on the finite element or finite difference methods have also been proposed in the literature. These models describe the phenomena in the semiconductor material such as carrier generation, recombination, drift, and diffusion in terms of nonlinear pde's in space-time. *Hybrid* models [8] which combine the analytical and numerical concepts for improved computational efficiency also exist. Nevertheless, all of the aforementioned IGBT models are seldom used for time-domain circuit simulation of power electronic converters. For circuit simulation employing the above models would contribute to very large computational times even with fewer devices, at a moderate switching frequency, and for a small simulation interval. System-level studies that involve controller optimization or gate circuit design typically require much faster models to enable repeated simulations during the design cycle; however, they also demand higher model accuracy. The purpose of this thesis is to propose FPGA-based hardware emulation of power electronic devices based on *behavioral* modeling.

1.1 Review on Modeling of Power Converters

Behavioral models or macro-models reveal the necessary device dynamics in circuit simulation while omitting excessive device-level details, thus gaining computational advantages. Therefore, such models are better in accuracy and modeling details than simpler system-level models (ideal models); however, macro-models are feasible for real-time emulation. Another advantage of the behavioral models is that they are data-sheet driven, i. e., there is no need for a long list of inaccessible device-correlated parameters as in the analytical or numerical models; behavioral models can be characterized by parameters derived directly from the device manufacturer's datasheet.

1.1.1 Power Diode Models

There were a number of power diode models proposed in the past few years. Tan *et al.* [10] presented a comprehensive review and summary of recent power diode models for circuit simulations and these models were classified as analytical models, numerical models, hybrid models and empirical model. Currently, the most commonly used diode models are several analytical models based on the lumped-charge and dynamic charge concepts. Liang [11] used a pn-diode micro-model which represented forward and reverse recovery phenomena to develop a SPICE based circuit built on semiconductor physical equations such as the charge control equation. Physics-based analytical power diode models using lumped-charge concept addressed by Lauritzen and Ma *et al.* [12–14] were developed from the simple reverse recovery diode to the fully forward and reverse recovery with emitter recombination effects contained power diode. MAST language based behavioral power diode model in Saber® including both static and dynamic features such as switching behavior, junction capacitance, forward and reverse recovery was presented by Courtay [19]. The extracting physical parameters processes based on these models above from data sheet information were provided in their papers correspondingly. A precise power diode model including transient behavior, temperature dependence, emitter recombination, mobile charge carriers in the depletion layer, carrier multiplication, and self-heating was reported in [15] based on dynamic charge concept which was to recognize the dynamic characteristics of the charge distribution in the base region. Similar approaches based on this concept can also be found in [16–18]. However, the parameter extraction procedures of these dynamic charge models were not provided. Recently, a compact power diode model which was using Fourier-series-based solution for ambipolar diffusion equation (ADE) were proposed in [20] with a relatively slower simulation but higher accuracy compared to the dynamic charge and lumped-charge models. Wang *et al.* presented implementation and comparison of proposed power diode models including lumped-charge model, dynamic charge model, and the Fourier-based model for system simulation in MATLAB® and Simulink [21].

1.1.2 IGBT Models

A chronological listing of literature related to IGBT device-level behavioral modeling is given in the references. The origin of the device-level behavioral models can be traced back to the macro-model proposed by Tzou *et al.* [22] which included a voltage-controlled current source I_{CON}, junction capacitances, and various resistances, R_{ON}, R_o, R_{GE}; the model parameters were derived from the device datasheet. Kim *et al.* proposed a data-sheet driven polynomial approximation for the parameter extraction of the static and dynamic characteristics, and their implementation in Spice [23]. Subcircuit representation of the IGBT was proposed by Shen *et al.* in [24] but it was still heavily physics-based, although simplified assumptions were made to improve model flexibility. In [25], Hsu *et al.* formalized the IGBT as a Hammerstein configuration which included a nonlinear static block and a piecewise linear dynamic block; parameter extraction was based on either measurement or simulation of physics-based models. A table-lookup based IGBT model with an anti-parallel freewheeling diode for EMTP simulation was proposed in [26] to calculate converter power dissipation but it was not suitable for IGBT circuit behavior predictions. The analog behavioral modeling (ABM) facility in Spice was utilized in [27, 28] to build an IGBT equivalent circuit containing nonlinear controlled voltage and current sources that implement the device's physical equations. In [29], the AMB facilities were deeper used to build a completely behavioral IGBT macro-model based on physical equations for both internal BJT and MOSFET; the parameters extraction was simplified as only common data-sheets characteristics required, and the curve fitting algorithm was replaced by an LUT based specification. All of the above models are high order (5 or greater). Tichenor *et al.* in [32] proposed a considerably simpler (first-order) model, with only one junction capacitance; however, custom experiments and curve-fitting were used for parameter determination.

An improved Saber® behavioral IGBT model was proposed in Zhang *et al.* [30] to accurately capture the device behavior, and a characterization tool was developed to derive the model parameters from the datasheet. The characteristics of Saber® *IGBT1* model and the choice of the parameters were also introduced in Mei *et al.* [31]. Lauritzen *et al.* [33] proposed the lumped-charge model based on the device physics. Although simpler than the Hefner model, it is still complex with 17 model parameters using exact physical equations; an experimental setup was used for parameter extraction. Consoli *et al.* [34] used neural networks for part of the IGBT model to preserve accuracy while improving computational speed and the proposed PSpice® behavioral PowerMesh$^{\text{TM}}$ IGBT model also included temperature effects both on steady-state and dynamic features. Kang *et al.* [35] provided a parameter extraction algorithm relied on datasheets and MATLAB® optimization toolbox for a hybrid model where the static part was based on empirical measurements, but the dynamic part was based on Hefner's physics-based model. Stier *et al.* [36] proposed a high-order network model for the IGBT, containing both static and dynamic characteristics as well as the conductivity modulation effects which were negligible at hard switching conditions, in Simplorer with data-sheet parameterization. Oh and Nokali presented an IGBT behavioral model consisted of a DC part based on an empirical formula based on a Darlington and a dynamic part based on Hammerstein configuration [37]. As an improvement, Asparuhova *et al.* [38,39] used MATLAB® for parameter extraction and optimization of a Hammerstein-based behavioral IGBT macro-model which consisted of a nonlinear static block and a linear dynamic block for OrCad PSPICE simulator. Baraia *et al.* in [40] used a combination of curve-fitting and experimentation to characterize a lower-order IGBT model and implemented with the MAST language in Saber®. Castellazzi proposed a model for packaged semiconductor power module containing both the IGBT and the anti-parallel diode [41]. In this model, the device physics equations are coupled with heating, and EMI, associated with packaging, layout, and interconnections. Nejadpak *et al.* in [42] used system identification methods based on the step response of a linear system to obtain a state-space IGBT model whose voltages and currents were measured at different switching frequencies and a wide range of power ratings. A behavioral IGBT model proposed by Mijlad *et al.* in [43] based on electric elements of SIMSCAPE library was used to implement in MATLAB®/SIMULINK software.

1.1.3 FPGA based Power Converter Switch Models

FPGA based modelings of power electronic converters were also proposed in the literature. Since precise power electronic converters simulation requires time steps which are too small to be implemented in real-time using FPGA, the dual-time step approach in [48,49] was introduced based on the ideal switch R_{on}/R_{off} model proposed by Hui *et al.* [50]. In [44], a look-up table and state machine approaches were developed for detailed modeling of the turn-on and turn-off characteristics of the power converter. Li *et al.* deduced a mathematical IGBT model based on single cycle delay method for the on/off states and LUT method for the logarithmic and exponential terms in the switching characteristics [51]. Myaing *et al.* in [45] used an LUT based device-level behavior switch model of the power converter (IGBT) including the turn-on and turnoff nonlinear switching characteristics, power losses, and the tailing current behavior to realize FPGA-based real-time emulation of power electronic systems. Kiffe *et al* presented an approach for an automated generation of an FPGA-based oversampling model of power electronic circuits using switch models based on the switching states as a short circuit represented by an inductor or an open circuit represented by a capacitor or by a variable resistor [54]. In FPGA-based real-time power systems simulator, built from modified nodal approach (MNA) in [53], associated discrete circuit ADC-based model which introduced a constant conductance matrix calculated off-line with the independence of the switch state was proposed [52] to reduce the algorithm complexity. The MNA method was basically modeling power electronic switches as a small inductor or capacitor during on/off respectively [46,53]. Herrera *et al.* [46] used an improved Thévenin equivalent circuit based on [47] with modification of the voltage source during the on-state for forward voltage and on resistance voltage drop approximation to represent the switching devices.

In order to perform an FPGA-based simulation of power electronics, Zhang and Sun in [55] used Jacobi iterative methods to solve the discretized circuit equations containing state space converter model with the average current control instead of directly solving them by inverting the matrix. Huang *et al.* proposed an inverter system nonlinear model using linear equivalent function including the IGBT and diode voltage drops and transient process based on FPGA for HIL real-time simulation [56]. Myaing *et al.* provided a comparison of IGBT models including system-level behavioral switch models such as ideal, average, switching function models (classified by Jin [57]) and device-level models, such as LUT-based and behavioral IGBTs, for real-time simulation of electric drives on FPGAs [58]. At system-level, compared to the average models which only considered the low-frequency component but ignored the high-frequency components, ideal and switching function models are mostly used in FPGA power converter simulations [58]. As more accurate and complex models were demanded in current power system simulation, Wang *et.al* in [59] performed hardware emulations of detailed device-level IGBT using the physics-based Hefner's model based on Oziemkiewicz's work [60] and the power diode using the Lauritzen's model on FPGA.

This thesis does not compromise on any macro-model details. The behavioral models including all of their device-level effects are faithfully emulated by using parallel methods to perform their emulation on the FPGA for observing, monitoring and analyzing the device transient behaviors. Both behavioral power diode and IGBT models are Saber® based models [61] which described in MAST languages. These models have high accuracy but they are also suffused with complexity and nonlinearity. In order to implement on hardware, the hardware models should be built by firstly linearizing and discretizing the original Saber® models utilizing finite element and difference methods (FEM and FDM), then modifying in MATLAB®, and finally representing using VHDL in Vivado®. In this work, to achieve the lowest latencies, resource consumption, high fidelity and accuracy requirements, the hardware designs for the IGBT and power diode models are modular, fully paralleled and deeply pipelined using IEEE 64-bit double precision floating-point number representation.

1.2 Motivation and Challenges

Due to the inherent parallelism, hardware simulation on FPGA has been increasingly adopted in power electronics and some other power engineering system designs. It could save time and cost of device development and could replace the verification process using actual devices. For example, engineers can predict the behaviors of any design or test circuits and systems in both transient and steady-state based on corresponding hardware emulation instead of installing and connecting actual electric components and putting into real operation. Thus, HIL emulation is significant. With more advanced features adding to the present FPGAs, the efficiency and performance of HIL simulators were definitely improved.

In power system modeling applications, more accurate models of power system components have been paid attention as the growing demand for the transient analysis. As mentioned above, FPGA-based power converter models were built in both system-level and device-level. In [62], FPGA-based real-time EMTP simulator was firstly presented for large transmission networks and then in order to achieve high accuracy and efficiency of hardware simulation, a real-time fully iterative nonlinear electromagnetic transient solver was introduced to assist the emulating of power system network on FPGA [63]. A digital hardware emulation of accurate and complicated universal machine model and the universal line model were proposed in [64]. Chen *et al.* also developed a multi-FPGA hardware design based on the functional decomposition methodology for detailed real-time EMT simulation of large-scale systems in [65] and proposed a digital hardware building block concept to emulate power system components for the real-time simulation of large-scale power grids on FPGAs in [66]. Several other detailed accurate models of electrical power system components could be found in the literature. As proposed in [67], a digital hardware distance protective relay was emulated in real-time with low-latency on FPGA In [68], a real-time digital multi-function protection system is also emulated on reconfigurable hardware. Liu *et al.* realized the real-time hardware electromagnetic transient simulation of nonlinear power transformer models on FPGA [69–71]. Tavana and Jandaghi *et al.* presented digital hardware emulations of electrical machines with different approaches for HIL simulation on FPGA [72–75]. A real-time hardware emulation of MMC using the data-sheet based device-level transient electro-thermal model on FPGA is presented by Shen *et al.* [76]. Besides these, there are many publications related to real-time FPGA-based as well as non-FPGA based such as PC-Cluster based (using CPU and C language) HIL simulation for power and energy system [77, 78]. Those studies have included various aspects such as AC motor drives [79], time-varying harmonics [80], robust control [81], power system transient stability [82], power transformers [83], generators (wind turbine) [86], wind energy system [84], HVDC system [85], transmission line [88], induction machines [89–91], power converters [92], and some HIL interfacing issues [87].

As the modern FPGA technological progress, based on the foregoing hardware implementation methods and design strategies of power systems, components, and electronics, high accurate emulation of a detailed device-level power converter model becomes feasible and imperative. In current large-scale systems such as smart grid, VCS-HVDC, and renewable energy, power electric converters play an important role. While most of the switch models in converters such as MMC for hardware emulation are system-level simple or ideal switches which make hard to observe the exact practical behaviors. According to the structure of 3 phase MMC, all the submodules have same topologies and they are repetitively placed within one leg. Also, each leg is basically identical with the only phase difference. In that case, the superiority of hardware emulation become more obvious since each leg and even each sub-modules could be implemented in parallel.

This work, which implements digital hardware emulation of device-level behavioral models for the IGBT and the power diode, has not been attempted before. Furthermore, detailed precise device-level power electronic converter models applied to MMC for hardware simulation has not been implemented so far either. These are mainly due to the challenges involved such as discretization and linearization of massive coupled nonlinear elements with several implicit integrations as well as numerical solutions of relatively large and complex system equations and matrices obtained from nodal or mesh analysis of equivalent circuits accompanied by Newton-Raphson iterations. The following provides a detailed list.

- As the modern technology tends to rise the power converter switching frequency so as to reduce the harmonic distortion and usage of filters, the real-time simulation becomes more difficult in high switching frequency range. Basically, real-time simulators usually take time steps within 5 to 10 μs range. In order to accurate emulating the power electronics such as the IGBT on FPGAs, simulation time steps need to be at least 20 times less than the switching frequency which limits the switching frequency within the range of 5kHz to 10kHz [58]. For accurate device-level behavioral model, the time step needs to be small enough usually hundreds or even tens of nanosecond scale so as to observe the transient. Thus, it burdens the hardware emulation and the conflict of small time step required for precise model simulation but large step size needed for reasonable hardware implementation is the main challenge of this work.

- The low speed of generating the results is an issue. When the model is simulated and implemented using the discretized and linearized system equations, linear solvers and N-R iterations are required to avoid the poor numerical convergence and to gather correct results. That causes the hardware emulation more time consuming. Meanwhile, due to the nonlinearity, complexity and coupling features of the models, the formed large matrix for a circuit system are difficult to be divided into or replaced by several small matrices and solved in parallel and individually using the block node diagonal sparse matrix algorithm mentioned in [93]. Some other existing matrix solvers are very time-consuming and the computational time is in exponential growth with respect to the matrix dimension. Thus, decoupling the large matrix and efficient solution of the system equations should be explored otherwise the simulation will be very slow. In addition, for the generality and versatility purpose of the hardware models, all the equations, and the internal variables are solved without using LUT. However, this causes the model more complicated and introduces more latency which makes the entire hardware emulation hard to reach real-time.

- Since highly coupled nonlinear circuit elements appear in the detailed device-level behavioral switch models (IGBT and anti-parallel diode), it becomes difficult to deal the circuit with nodal analysis. Besides, various methods of solving differential equations (ODE) for those formulated nonlinear and highly coupled functions such as Euler methods, trapezoidal rule, and RungeKutta should be considered and tested to correctly discretized and linearize the system since small errors during calculation process may lead to simulation failure. In the hardware implementation, one of the methods should be utilized with optimal timing and resources consumption while maintaining relatively high accuracy.

- Numerous units are contained in the hardware model of the power converter. Therefore, their connections and organizations are not easy. The signal updates within each unit, the sequence and parallel structures of the model need to be considered and controlled carefully using finite state machine based on certain clock cycles.
- Commonly, most off-line power system or power electronic simulation software such as SaberRD® and PSpice® support dynamic time step simulation and are able to keep the resulting waveform shapes proper and normal in the time-domain [94]. While, by contrast, if the dynamic time-step algorithm introduced into HIL emulation without any adjustment or modification, it might affect the output waveform shapes which is also a challenge for the model implementation in hardware. Moreover, even using the fixed time strategy in hardware emulation, the N-R iteration would cause the waveform slightly out of shape due to the uncertain and loose iterative times. For example, if the results are generated by only processing the N-R iteration once during the previous Δt but they need three times iteration in the next Δt, the output waveforms would be longer for the second Δt compared to the first one in the time-domain. This phenomenon more likely appears during transients.
- Finally, since the small time step is applied to the simulation and the calculated results are very sensitive to the parameters as well as the internal variables of the model, a little inaccurate data representation might lead the output error. Thus, more precise floating point format should be used. However, this will cause the slower implementation and the more resource consumption resulting a new challenge.

1.3 Objectives

In order to realize HIL emulation of power electronic circuit using device-level behavioral IGBT and power diode models on FPGAs, the following main research objectives need to be achieved during this work:

1 Owing to using Saber® based behavioral models of the IGBT and power diode, the MAST models should be firstly studied. Courtay [19] provided a priceless and meaningful template to demonstrate the relationship between the Saber® MAST descriptive model of the power diode and the traditional system linear and nonlinear equations. That could be a guide for understanding the IGBT model. Most of the com-

ponents inside the behavioral models are attributed to the device nonlinearities and complexities. Then, once the models are formulated in a normal mathematical way, they need to be linearized and discretized as there are several nonlinear and coupled portions (integral and differential equations). Different methods should be applied and tested so as to obtain the optimized forms before realizing parallel computations and hardware design.

2 In hardware design, firstly, floating point mathematical arithmetic units in Vivado® IP cores such as the comparators, adders, subtracters, multipliers, dividers, reciprocal operators, square root operators, exponential operators, logarithmic operators and ROMs need to be carefully designed and configured to optimal conditions for accuracy, latency and resource consumption. Afterward, the internal and external structures of all hardware units of the behavioral IGBT and diode models should be well organized and designed for maximum parallelism so as to be properly mapped, placed and routed on FPGA. The proposed FPGA-based emulation of a large power converter circuit requires many IGBT and diode devices. Therefore, the behavioral models should be modularized as a single IGBT and anti-parallel diode pair switch module which can be easily and conveniently used for any power electronic HIL design with different topologies so as to increase the module flexibility and universality. Meanwhile, in the case studies, besides the switches, the 5-level MMC circuit also includes other power electronic elements such as independent/dependent supply sources, passive linear/nonlinear R, L, C elements, and pulse-width modulation, transmission line model and controller of a multi-level converter. Those also need to be designed in hardware to complete power electronic circuit emulator.

3 Depending on various demands, different paralleled and pipelined linear solvers are designed in hardware for solving different dimensional matrices or system equations. Basically, Cramer's fast solvers are developed for the small system with matrix size less than 4 dimensions; Gauss elimination with partial pivoting linear solver is designed for larger matrices; and the Gauss-Jordan elimination linear solver is presented for dealing with the matrix that all elements are known and constant.

4 Different control schemes are applied to this work. Comparing to VTCM stated in [59], a part of this work will use a more reliable LTE-based dynamic time step scheme referred to SaberRD® in hardware emulation. Also, output control model for the hardware simulation must be introduced in either fixed or variable time step schemes so as to ensure the resulting waveforms in proper shape.

5 Computational burden of the fully detailed behavioral models of IGBT and power diode is too high to achieve agreeable or desirable execution speed due to the complexity. Thus, the IGBT and anti-parallel diode pair module should be simplified reasonably and logically so that it could be used in large MMC circuit. Meanwhile,

the TLM decoupling technique could be introduced to the simulation of large power system circuits such as MMC which usually focuses on the system-level behaviors and characteristics in steady-state or DC mode. Then, fine-tuning the parameters such as surge impedance of TLM link to obtain accurate results is intricate but it is an important step which needs to be carefully done to overcome several applicable restrictions, constraints, and drawbacks of TLM mentioned in [47,95].

6 Besides coding and programming in MATLAB® and Vivado®, identical power electronic circuits for the case studies are required to be built in off-line simulation software such as SaberRD® which contains various comprehensive precise device-level models and has powerful interfacing capability [94] so as to compare and verify the hardware simulation results.

1.4 Thesis Outline

This thesis contains 5 chapters. The other chapters are organized as follows:

- Chapter 2 presents an overview of FPGA technologies and Xilinx® Vivado® deign tool for hardware simulation. Some advantages of FPGA are stated in this chapter. Meanwhile, its overall architecture, essential components are introduced and the fundamental resources of Virtex® 7 family FPGA devices are compared. Hardware design tools and design flow are illustrated in detail. This chapter also provides several programming techniques for hardware design.

- Chapter 3 elaborates the hardware emulation of the detailed device-level behavioral power diode and IGBT models on the FPGA. Model formulation, discretization, and linearization of both devices are presented in detail. Their sub-units hardware module structures and overall hardware module architecture are described and illustrated. Meanwhile, several approaches for hardware emulation of power electric circuits such as Newton-Raphson iteration method, different time step simulation strategies and the TLM decoupling technique are narrated in this chapter. Different linear solvers are designed in hardware for solving different dimensional matrices or system equations. The model simplification is given. The entire IGBT and anti-parallel diode hardware model are designed as a single unit for being conveniently used in any power converter circuit. This chapter also provides the modeling of other power system models such as passive electrical elements and TLM stub models.

- Chapter 4 presents case studies of simple device circuits, a chopper circuit, a half-bridge, and a multi-level converter circuit. Their hardware emulation results of the steady-state and transient characteristics including voltages, currents and power dissipations are captured, compared and validated with corresponding SaberRD® simulations.

- Chapter 5 briefly summarizes the thesis. The contributions of this work and some suggestions for the future work are indicated.

2

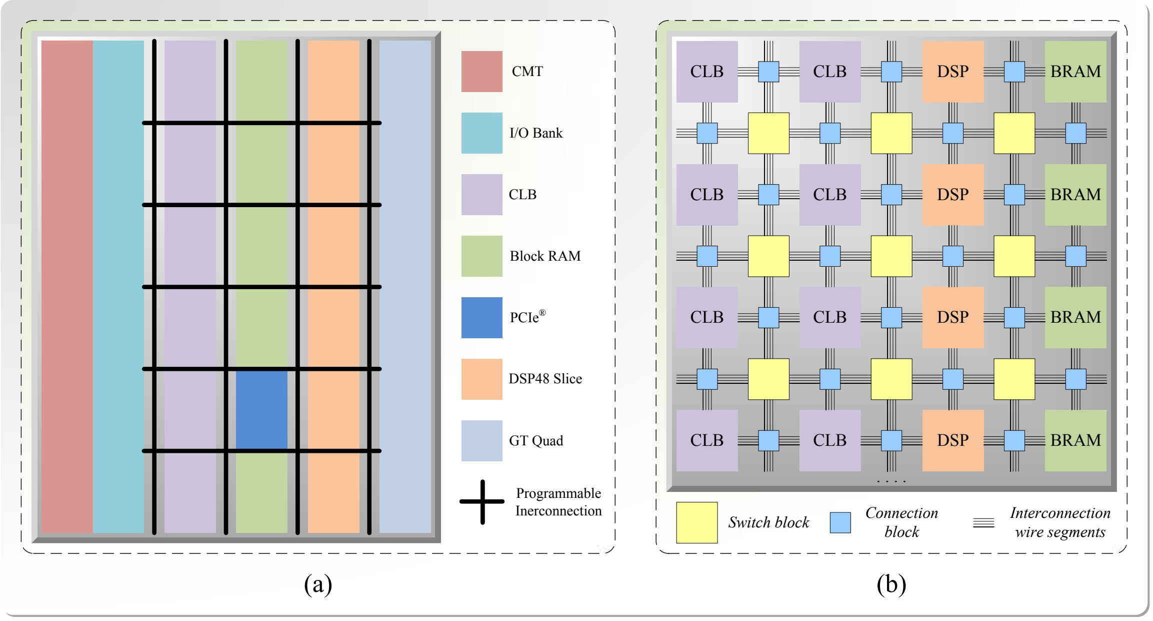
FPGA Overview

FPGA is a semiconductor device same as other IC unit. Compared with GPU, FPGAs is smaller in physical volume and dimension as well as power dissipation. Although CPUs and GPUs have higher computational capabilities or arithmetic speed than FPGAs, their operation time is not constant and stable from each run based on the same programs since they are also determined by their operating system or other hardware and software configurations such as slow data transfer paths and limited local memories. Moreover, a crucial fact is that FPGAs interface functionality which could not be provided by GPU or CPU. Hardware structure of FPGA can be defined, configured and customized by designers; while a CPU or GPU has fixed hardware structure. Similar to processors and processor peripherals, FPGAs can be reprogrammed in-system unlimited times on demand. Meanwhile, FPGAs are more efficient for digital signal processing (DSP) applications due to the implementation of custom and fully massive parallel architecture and algorithms which show the benefits of performance, cost, and power over traditional processors.

Compared to other expensive and time-consuming ASIC designs which have to achieve physical layout and be sent to semiconductor fabricators, the FPGA is reprogrammable and its design only needs to be reconfigured directly on-chip by the designers themselves without physical layout. It also has growth trend in the resource capacity, performance as well as efficiency. Recently, its performance is nearly normalized with ASICs. The latest Virtex® UltraScale$^{\text{TM}}$16nm devices have around 5 million logic cells with the clock frequency range from 100 to 500 MHz. Due to these facts, modeling large and complex design in hardware becomes possible on FPGA devices. Thus, it has such advantages on many aspects as high performance due to the real-time analysis of high-rate data streams; high reliability due to the precise hardware platforms for any feasible tasks; one-time device expenses due to the reconfigurability and durability as well as high developability or exploitability due to rapid and flexible prototyping with fast time-to-market feature. So far, due to rapid growth over the past decades, FPGA has widely seeped into various applications including bioinformatics, radio or video broadcast, medical imaging, networking, computer, storage, telecom, wired or wireless communications, automotive, scientific instruments, industrial automation, aerospace, and defense.

This chapter will give a brief introduction to the FPGA technology. Its architecture and resource elements will be described The FPGA design flow will be interpreted as well. Lastly, some design techniques will be provided.

2.1 FPGA Architecture



(a)

(b)

Figure 2.1: (a) General architecture alignment diagram of Virtex®-7 series FPGAs [96]; (b) FPGA basic fabric floor plan with interconnections [97].

FPGA is an integrated circuit containing logic blocks, memory blocks, the hierarchy of reconfigurable routing channels interconnecting the blocks, and input/output (I/O) banks for data transfer driving and receiving. It can implement both combinational and sequential logic functions with different levels of complexities. The Virtex®-7 family is designed for highest system performance and capacity among the Xilinx® 7 series FPGAs [97]. Fig. 2.1(a) depicts a general architecture alignment diagram of Xilinx® Virtex®-7 series FPGAs as the emulation platform which is applied to the entire power electronic circuit hardware design. The basic architecture of FPGA is a typical two-dimensional symmetrical array. Xilinx® 7 series FPGAs arrange their different resources including CLBs, IOBs, Block RAMs (BRAMs), DSP slices and GT quad(a group of serial transceivers) in columns (see Fig. 2.1(a)) based on the unique columnar approach provided by the ASMBL™ architecture [97]. The periphery of the array was surrounded by I/O banks. Inside the array, there are many logic blocks (usually CLBs), typically arranged in a grid, with wires on all sides. Transistors based programmable switches controlled by fuses, anti-fuses or memory cells and interconnection wire segments around each CLBs compose the FPGA routing channels. As depicted in Fig. 2.1(b), switch blocks and connection blocks located at the routing intersections will decide which logic blocks should be placed in route. For CLBs, connection blocks (or boxes) laid between each two CLBs link the I/O pins of specific defined CLBs to wire segments; while switch boxes placed in the center of four CLBs connect the corresponding connection boxes in both horizontal and vertical directions of routing channels. Some elements on the device and their architectures or functional structures will be specifically interpreted in this section.

2.1.1 Clocking Resources

Clocking resources in Xilinx® 7 series FPGAs manage both complex and simple clocking requirements. The Clock Management Tiles (CMT) provide clock frequency synthesis as well as clock skew and jitter filtering functions [96]. As shown in Fig. 2.1(a), CMTs are arranged in columns beside the I/O bank.

2.1.2 Configurable Logic Block (CLB)

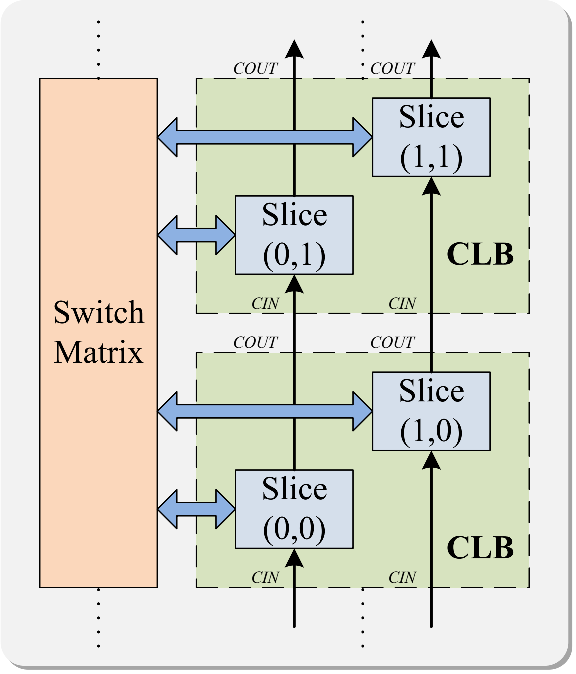


Figure 2.2: Arrangement of slices within CLB and the connection with switch matrix in 7 series FPGAs [97].

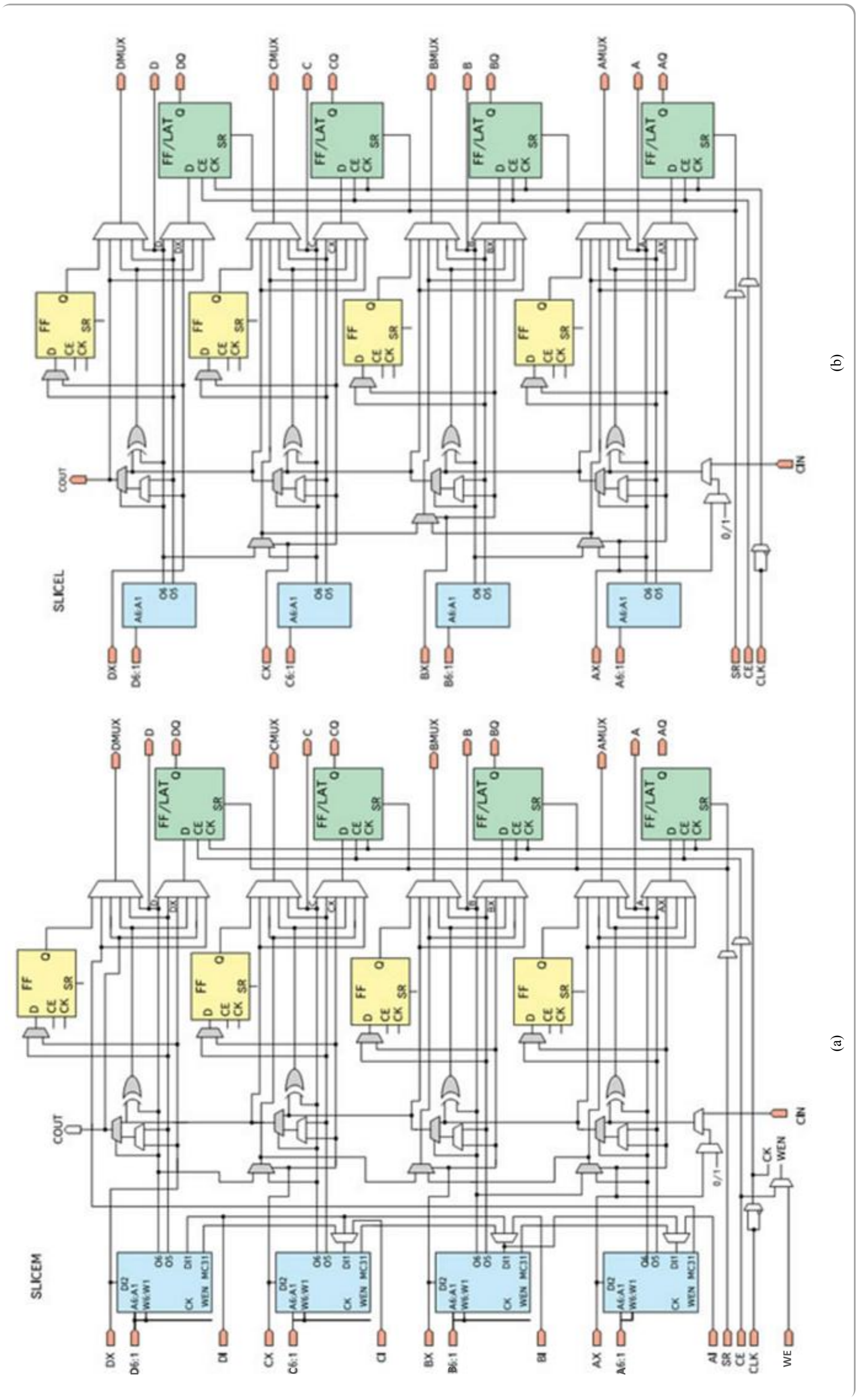


Figure 2.3: Architecture diagrams of 7 series FPGAs slices (a) SLICEM; (b) SLICEL [97].

In 7 series devices, CLBs align in columnar architecture. It can provide advanced, high-performance logic for implementing sequential and combinatorial circuits. As shown in Fig. 2.2, each CLB containing a pair of slices is connected to a switch matrix for access to the general routing matrix and the physical direction of the carry propagation in slices is also indicated. Each slice includes four logic function generators (6-input LUTs), eight distributed storage elements (FFs), several wide function multiplexers and dedicated high-speed arithmetic carry logic. Fig. 2.3 shows the basic structure diagrams for two types of slices - SLICEL and SLICEM. Each CLB can have two SLICEL or a SLICEL and a SLICEM. The device in the lab, VC7VX485T has 75,900 slices (43,200 SLICEL and 32,700 SLICEM); 303,600 6-input LUTs; 8,175 distributed RAMs; 4,088 shift registers and 607,200 Flip-Flops [97].

2.1.3 Input/Output Block (IOB)

According to [100], configurable IOB in 7 series FPGAs can support many standard interfaces and it has such features as programmable control of output strength and slew rate, on-chip termination using DCI, and internal reference voltage generation. The 7 series devices offer both high-performance (HP) and high-range (HR) I/O banks with each I/O bank composed of 50 IOBs. Fig. 2.4 shows the different types of IOBs and their connections to the internal logic and the device pad. Based on the 7 $Series\ FPGAs\ SelectIO\ Resources$, XC7VX485T in the lab has only HP I/O banks (Max. 700 IOBs or 14 I/O banks available [101]) which are shown in Fig. 2.4(a). The SelectIO^{TM} input, output, and 3-state drivers are located inside the IOB and the IOBs lie around the periphery of FPGA chips [100].

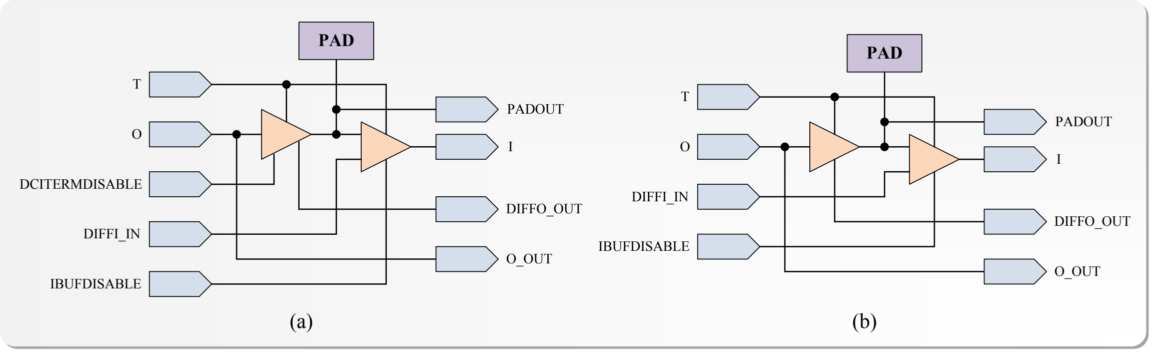


Figure 2.4: Functional structure of the IOBs (a) regular HP; (b) regular HR [100].

2.1.4 Block RAM (BRAM)

In addition to distributed RAM and SelectIO^{TM}, 7 series FPGAs provide vast BRAMs. BRAMs are used for data storage, high-performance state machines, FIFO buffers, shift registers, LUTs or ROMs. In 7 series FPGAs, they are placed in columns. One BRAM can store up to 36 Kbits of data and can be configured as either two independent 18 Kb

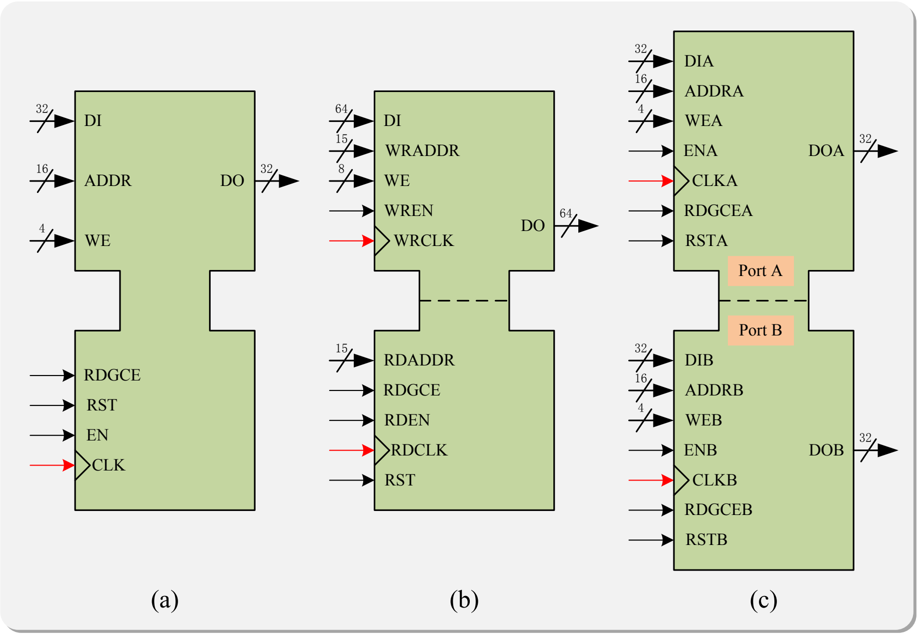


Figure 2.5: Functional structure of BRAMs in 7 series FPGAs (a) Single Port RAM; (b) Simple Dual Port RAM; (c) True Dual Port RAM [98].

RAMs, or a 36 Kb RAM. It operates both Write and Read synchronously. Based on [98], XC7VX485T in the lab has 1030 36 Kb BRAMs formed 15 columns with each column containing the amount of 70 blocks. The Xilinx CORE Generator^{TM} block memory modules can implement single-port or dual-port RAM modules, ROM modules, synchronous FIFOs, and data width converters [98]. BRAM can be configured in different RAM modes (single-port, simple dual-port (SDP) and true dual-port (TDP)) as general purpose shown in Fig. 2.5.

2.1.5 DSP48E1 Slice

The 7 series FPGAs contain many low-power and full-custom DSP slices which can implement such functions as multiplication, accumulation, addition, barrel shift, wide-bus multiplexing, magnitude comparator, bitwise logic functions, pattern detection, and wide counter parallelly. It can enhance computational speed, flexibility, and efficiency for many applications [99]. Fig. 2.6 shows the basic functionality of the DSP48E1 slice.

2.1.6 Virtex®-7 Board Resource Overview

As the Table 2.1 shown, the maximum number of I/O pins is about 1930 for all different boards; the Xilinx® Virtex®-7 XC7VX690T and XC7VX980T have 3600 DSPs which are the highest two among the Table 2.1 and the most available LUT is found on the board Virtex®-7 XC7V2000.

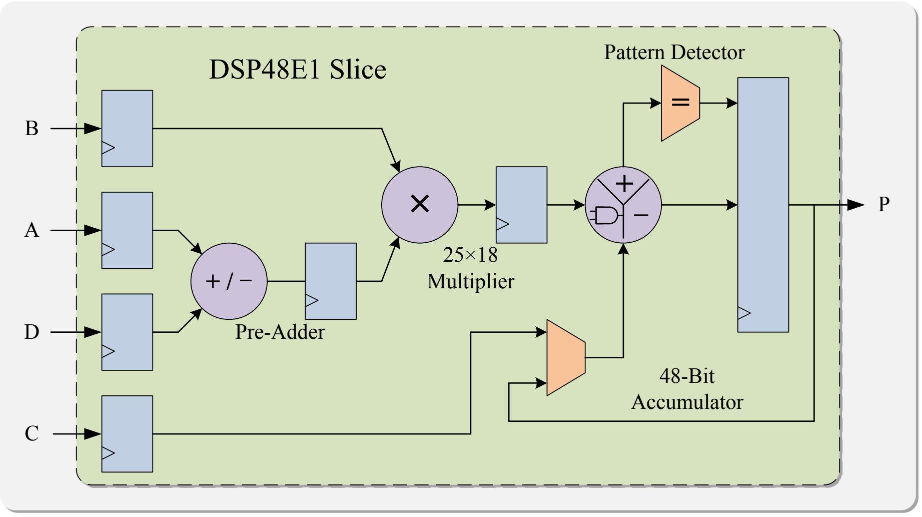


Figure 2.6: Basic functional structure of DSP48E1 slice [99].

Table 2.1: Comparison of the Xilinx® different platforms

FPGA Virtex®-7 family boards	I/O	$BRAM$	$DSP48$	FF	LUT
XC7VX485T	1157-1930	1030	2800	607200	303600
XC7VX550T	1185-1927	1180	2880	692800	346400
XC7V2000T	1761-1925	1292	2160	2443200	1221600
XC7VX1140T	1926-1930	1880	3360	1424000	712000
XC7VX690T	1157-1930	1470	3600	866400	433200
XC7VX980T	1926-1930	1500	3600	1224000	612000

2.2 FPGA Design Tools and Design Flow

Xilinx® Vivado® Design Suite introduces system-level integration flow centered on intellectual property (IP) design into the traditional register transfer level (RTL)-to-bitstream FPGA design flow resulting a more advanced and efficient design for large scale hardware emulation [102]. The design flow with different stages, basically containing design entry, synthesis, implementation, programming and debugging, is shown in Fig. 2.7. Using Xilinx® Vivado® Design Suite, the entire FPGA hardware design can be managed from full RTL creation to bitstream generation in its Integrated Design Environment (IDE) which provides not only an interface to assemble, implement, and validate the design and the IP but also such features including logic simulation, I/O and clock planning, power and timing analysis, design rule checks, visualization of design logic, modification of implementation results, programming and debugging. It enables designers to manage such different types of sources as RTL, IP, XDC or SDC constraints, test benches for logic simulation, HLS and other design documentations in the project. Meanwhile, designers can

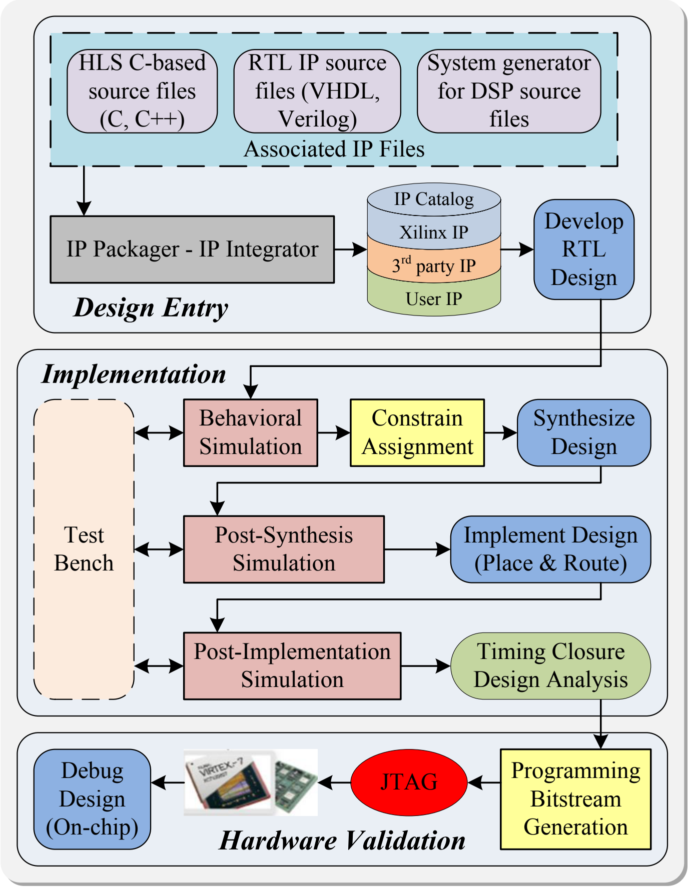

HLS C-based source files (C, C++)
RTL IP source files (VHDL, Verilog)
System generator for DSP source files
Associated IP Files
IP Packager - IP Integrator
IP Catalog
Xilinx IP
3rd party IP
User IP
Develop RTL Design
Design Entry
Implementation
Test Bench
Behavioral Simulation
Constrain Assignment
Synthesize Design
Post-Synthesis Simulation
Implement Design (Place & Route)
Post-Implementation Simulation
Timing Closure Design Analysis
Debug Design (On-chip)
JTAG
Programming Bitstream Generation
Hardware Validation

Figure 2.7: Vivado® design flow in general [102].

also do analysis and constraints assignment as well as any interaction with the design at different stages such as after RTL elaboration, synthesis, or implementation [103].

2.2.1 Design Entry

Vivado® system-level design entry consists of creating projects, adding source files, elaborating the RTL design, and inserting and configuring debug information at RTL stage.

2.2.1.1 IP Customization

IP Catalog in Xilinx® Vivado® IDE contains such IP sources as Xilinx® IP, third party IP and user IP. IP sources could be configured into the project either in RTL or netlist format to realize their various functions (such as math functions and memories or storages elements) via instantiating them in the HDL code or system-level design [102, 103]. Fig. 2.8 shows the customized IP cores used for 64-bits floating point calculation throughout the power converter hardware design.

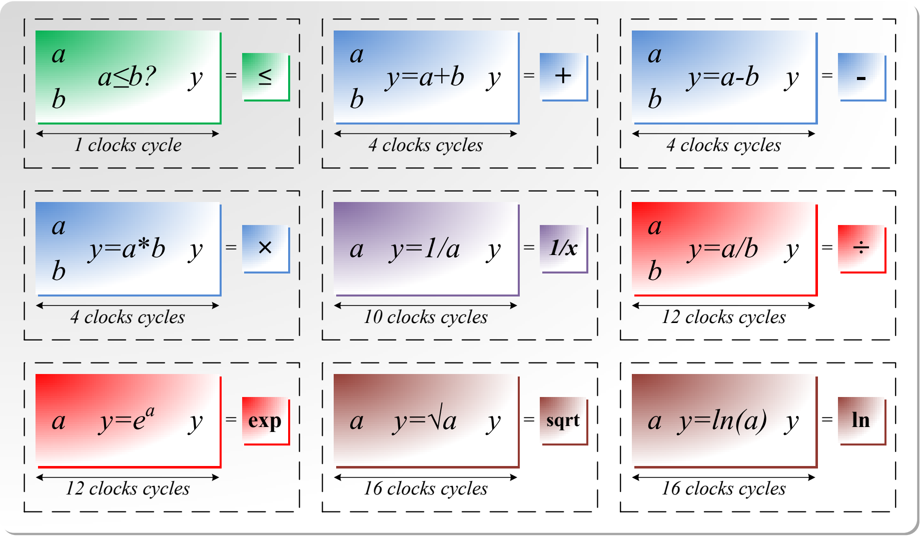


Figure 2.8: Vivado® customized 64-bits floating point IP cores [102].

2.2.1.2 HLS Description

High-Level-Synthesis (HLS) design using C-based languages to realize various logic functions. It has already doped in modern FPGA for UltraFast™ design. In Vivado® HLS, the C-based specifications are transformed to RTL module which can be packaged and implemented as an IP for future design use. The generated RTL can be simulated using C-based test benches. HLS bridges hardware and software domains so it can improve productivity for hardware designers [102, 103].

2.2.1.3 RTL Development

Vivado® supports HDL-based descriptions to represent the behavior or structure of a digital integrated circuit system. All source files can be imported, created, managed and modified in the Vivado® IDE. The RTL source files can be elaborated for RTL compilation validation, structure and syntax checking as well as logic definitions verification; RTL elaboration also performs the design rule checks (DRCs) for RTL lint-style [102]. Meanwhile, the logical design hierarchy can be provided as RTL netlist with an expandable logic tree, RTL hierarchy with graphical representation and RTL schematic, for exploration and design analysis [104]. After this step, the RTL design is preliminary optimized to an FPGA technology at an early stage before implementation [104].

2.2.2 Logic Synthesis

Synthesis in Vivado® IDE is the process of transforming an RTL-specified design such as an HDL-based behavioral or structural specification into a gate-level representation. Within timing constraints, it generates a netlist which is equivalent to RTL functionality [102]. After analysis of the RTL sources files, Vivado® synthesis derives optimized logic constructions for memory usage and performance. During this step, the estimated routing and component delays can be provided and device-specific primitives can be technologically mapped [106].

2.2.3 Design Implementation

Vivado® implementation is a process of place and route the generated netlist onto the FPGA device resources within the logical, physical, and timing constraints [102]. During this step, interconnect structure of the placed and routed design can be optimized [107]. After implementation, results and reports are available for design analysis.

2.2.4 Logic Simulation

Simulation in Vivado® IDE is a process of emulating hardware design and it helps verify the functionality and behavior of the design by injecting stimulus and checking the corresponding simulation results [102].

2.2.4.1 Behavioral Simulation

Behavioral simulation is a functional simulation at the RTL level to verify the syntax, structure, logic and functionality of the RTL design [102]. The design can be simulated by any translation made by later stages such as synthesis or implementation. For this logic simulation, timing information is not provided and timing constraint is not required consequently [105].

2.2.4.2 Post-Synthesis and Post-Implementation Simulation

Both of the simulations are based on structural netlists. According to [102], post-synthesis simulation is performed to verify that the synthesized design meets the behavior and function requirements. Meanwhile, it provides an opportunity for the designers to observe critical paths before implementation. Similarly, post-implementation simulation is a closer emulation to on-chip debug, performed to ensure that the implemented design meets the behavior, function and timing requirements [102]. These two netlist simulations can help designers to identify such potential issues as mismatches created by synthesis attributes or constraints, dual port RAM collision, missing or improper timing constraints, asynchronous path timing errors and functional problems caused by optimizations during synthesis and implementation [105]. A thing to note here is that post-synthesis and post-implementation functional simulations are supported for both VHDL and Verilog, but the timing simulations are only available for Verilog [105].

2.2.5 Programming, Hardware Verification, and Debugging

After successfully completed the implementation, the design is ready to run on hardware by programming the FPGA device and debugging in-system. The generated Bitstreams, as binary files representing the design, contain configuration data that can be loaded into the internal memory of FPGA board. The FPGA programming process is realized by connecting the target hardware and host PC using JTAG with a specific download cable supported by Xilinx® [102]. The Vivado logic analyzer in IDE can be used for performing interactive hardware verification and debugging of the design for analysis of the routing or device resources [103]. The digital output signals can also be sent to an oscilloscope via digital-to-analog converter (DAC) board. When the design meets the requirements and expectations on the hardware, the debugging process is done and the FPGA design is complete.

2.3 Programming Techniques

Nowadays, FPGA devices become more advanced and can implement more complex designs. Then, the strategies and methodologies would be important. The stages in design entry (C, C++, and HDL coding) have dominant impact on design performance, density, and power. Once the design does not meet requirements, the early stages including HDL design and synthesis should be reviewed rather than the later stages [103]. With the hierarchical design flow, the entire system should be partitioned into smaller and more manageable modules to be processed independently so as to increase the design efficiency [103]. Both floating-point and fixed-point number representations are needed for the power converter hardware design. Xilinx® supports different precision formats. Because the power converter simulation needs small time step size Δt about 200 ns or 2×10^{-7} s for the transient analysis, the single precision floating point representation is not enough for computational accuracy as it can only accurate up to six decimal places of a base-10 decimal number. Thus, if the small Δt is involved in the operation, for example, added by 1, the 2×10^{-7} will be automatically ignored. Therefore, instead of single precision format, the $IEEE\ 745\ Standard$ defined double-precision binary floating-point format should be employed in the design. Basically, the format has 64 bits, with a 52-bit fraction, 11-bit exponent, and 1-bit sign. Since the significand having an implicit integer with a constant 1, only the 52 bits of the fractional significand part is retained in the memory format and then the total precision is 53 bits (accuracy up to 13 dismal places) which are sufficient for the requirement of EMTP algorithm and the design. Although higher precision computation is more resource and time consuming, it can improve the results accuracy and reduce Newton-Raphson iterations turning out to a certain degree time saving and energy saving within the design. The fixed-point representation is only needed for the final output to DAC board and plotting on an oscilloscope.

2.4 Summary

Based on all the advantages mentioned before, FPGAs have become a widely used platform for various applications. Developers put great effort to break the limits to achieve higher clock frequency and hardware resources capacity. After system-level design flow and IP design implemented, the complex physics-based behavioral-level power electronic circuit can be emulated on the FPGA. To sum up, this chapter provided a brief introduction to the FPGA technology, an overview of Xilinx® 7 series FPGA architecture and resource elements and an interpretation of the FPGA Xilinx® Vivado® design flow. Moreover, some techniques are described for the thesis in terms of hardware design.

3

Hardware Emulation of Power Electronic Circuits Using Behavioral Switch Model

Precise models of power electronic converters can significantly improve the fidelity of hardware-in-the-loop simulators. In this chapter, digital hardware emulations for device-level behavioral power diode and IGBT models on FPGA are developed. Both IGBT and diode are detailed nonlinear models and feature a fully paralleled implementation throughout the entire design. The double-precision floating point format is introduced for an accurate computational process. VHDL as a basic design entry is used. Firstly, with some passive electronic components placed inside, the discrete-time linearized power diode and IGBT hardware modules are presented in detail including their design techniques and several algorithms. Then the circuit design and hardware emulation of an entire switch as a sub-module in part of a multi-level converter circuit are proposed later in this chapter.

3.1 Passive Electronic Components

Passive elements are the electronic components whose characteristics can be represented without external source or power supply. Resistors, capacitors, and inductors are three main passive electronic components.

3.1.1 Module Formulation

Passive electronic components can be linearized using Taylor expansion or Taylor series then discretized by Euler methods or Trapezoidal rule. After that, they can be represented by an equivalent resistance paralleled with a history current source in each simulation time-step.

3.1.1.1 Resistance Component - R

The discrete-time model of a resistance R shown in Fig. 3.1 is the same as its continuous-time model. The relationship between voltage and current is given as:

$$v_{km}(t) = Ri_{km}(t). \tag{3.1}$$

(3.1) can be represented as the following using conductance G instead of resistance R:

$$i_{km}(t) = Gv_{km}(t) = \frac{1}{R}v_{km}(t). \tag{3.2}$$

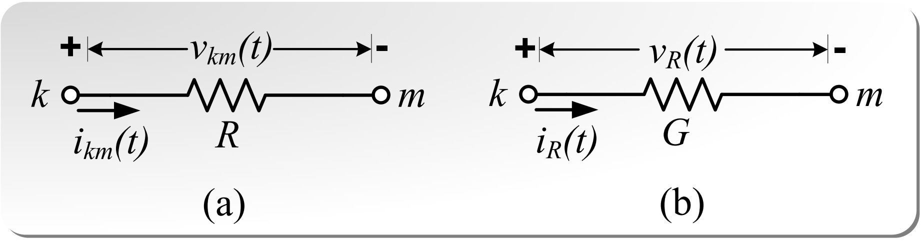


Figure 3.1: (a) Resistance R component and (b) Equivalent conductance model.

3.1.1.2 Capacitance Component - C

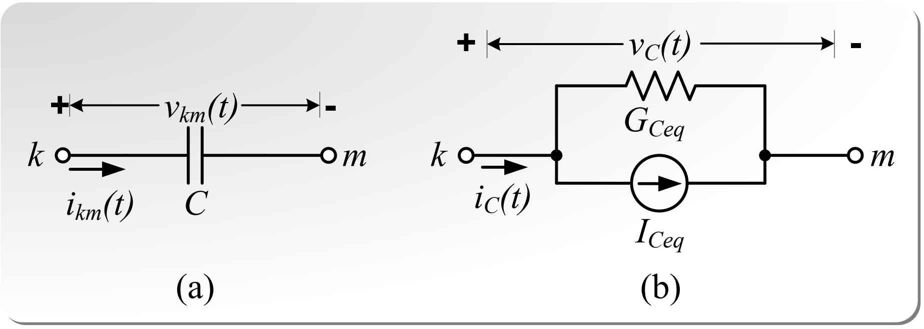


Figure 3.2: (a) Capacitance C component and (b) Norton equivalent model.

The differential equation for a capacitance C shown in Fig. 3.2(a) is given as:

$$i_{km}(t) = C\frac{dv_{km}(t)}{dt}. \tag{3.3}$$

Integrating this equation from time $(t - \Delta t)$ to t gives:

$$\int_{t-\Delta t}^{t} i_{km}(t)dt = C[v_{km}(t) - v_{km}(t - \Delta t)]. \tag{3.4}$$

Applying **Backward Euler** (implicit Euler method) to the integration parts and reorganizing gives:

$$\begin{aligned} i_{km}(t) &= \frac{C}{\Delta t}[v_{km}(t) - v_{km}(t-\Delta t)] \\ &= \frac{C}{\Delta t}v_{km}(t) - \frac{C}{\Delta t}v_{km}(t-\Delta t) \\ &= \frac{C}{\Delta t}v_{km}(t) + I_{Ceq}. \end{aligned} \tag{3.5}$$

(3.5) can be represented as an equivalent conductance G_{Ceq} in parallel with an equivalent current source I_{Ceq} as shown in Fig. 3.2(b), where the equivalent conductance G_{Ceq} is defined as

$$G_{Ceq} = \frac{C}{\Delta t}, \tag{3.6}$$

and the equivalent current source is a history term using the voltage solution from previous time-step $(t - \Delta t)$:

$$I_{Ceq} = -\frac{C}{\Delta t}v_{km}(t-\Delta t). \tag{3.7}$$

Applying **Forward Euler** (explicit Euler method) to the integration parts and reorganizing gives:

$$\begin{aligned} i_{km}(t) &= \frac{C}{\Delta t}[v_{km}(t+\Delta t) - v_{km}(t)] \\ &= \frac{C}{\Delta t}v_{km}(t+\Delta t) - \frac{C}{\Delta t}v_{km}(t) \\ &= \frac{C}{\Delta t}v_{km}(t+\Delta t) + I_{Ceq}. \end{aligned} \tag{3.8}$$

From (3.8), the equivalent conductance G_{Ceq} can be defined as:

$$G_{Ceq} = \frac{C}{\Delta t}, \tag{3.9}$$

and the equivalent current source is using the current value of v represented as:

$$I_{Ceq} = -\frac{C}{\Delta t}v_{km}(t). \tag{3.10}$$

Applying the **trapezoidal rule** to the integration part and reorganizing gives:

$$\begin{aligned} i_{km}(t) &= -i_{km}(t-\Delta t) + \frac{2C}{\Delta t}[v_{km}(t) - v_{km}(t-\Delta t)] \\ &= \frac{2C}{\Delta t}v_{km}(t) + [-i_{km}(t-\Delta t) - \frac{2C}{\Delta t}v_{km}(t-\Delta t)] \\ &= \frac{2C}{\Delta t}v_{km}(t) + I_{Ceq}. \end{aligned} \tag{3.11}$$

From (3.11), the equivalent conductance G_{Ceq} can be defined as:

$$G_{Ceq} = \frac{2C}{\Delta t}, \tag{3.12}$$

and the equivalent current source is a history term using both of the voltage and current solutions from previous time-step $(t - \Delta t)$ represented as:

$$I_{Ceq} = -\frac{2C}{\Delta t}v_{km}(t-\Delta t) - i_{km}(t-\Delta t). \tag{3.13}$$

3.1.1.3 Inductance Component - L

The differential equation for an inductance L shown in Fig. 3.3(a) is given as

$$v_{km}(t) = L\frac{di_{km}(t)}{dt}. \tag{3.14}$$

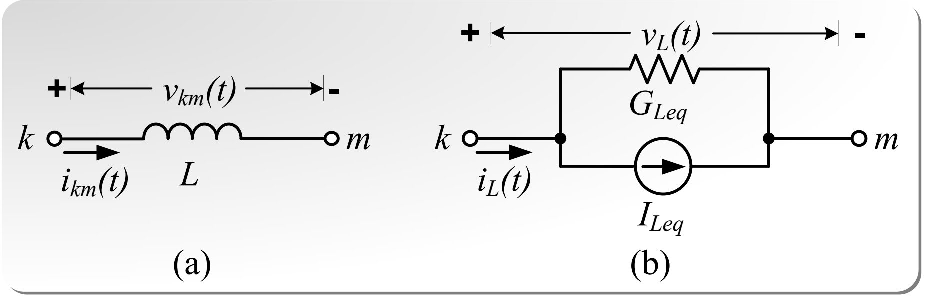


Figure 3.3: (a) Inductance L component and (b) Norton equivalent model.

Integrating this equation from time $(t - \Delta t)$ to t gives

$$\int_{t-\Delta t}^{t} v_{km}(t)dt = L[i_{km}(t) - i_{km}(t-\Delta t)]. \tag{3.15}$$

Applying **Backward Euler** (implicit Euler method) to the integration parts and reorganizing gives:

$$\begin{aligned} i_{km}(t) &= \frac{\Delta t}{L}v_{km}(t) + i_{km}(t-\Delta t) \\ &= \frac{\Delta t}{L}v_{km}(t) + I_{Leq}. \end{aligned} \tag{3.16}$$

(3.16) can be represented as an equivalent conductance G_{Leq} in parallel with an equivalent current source I_{Leq} as shown in Fig. 3.3(b), where the equivalent conductance G_{Leq} is defined as

$$G_{Leq} = \frac{\Delta t}{L}, \tag{3.17}$$

and the equivalent current source is a history term using the current value from previous time-step $(t - \Delta t)$ defined as:

$$I_{Leq} = i_{km}(t - \Delta t). \tag{3.18}$$

Applying **Forward Euler** (explicit Euler method) to the integration parts and reorganizing gives:

$$\begin{aligned} i_{km}(t) &= \frac{\Delta t}{L}v_{km}(t-\Delta t) + i_{km}(t-\Delta t) \\ &= \frac{\Delta t}{L}v_{km}(t-\Delta t) + I_{Leq}. \end{aligned} \tag{3.19}$$

From (3.19), the equivalent conductance G_{Leq} can be defined as:

$$G_{Leq} = \frac{\Delta t}{L}, \tag{3.20}$$

and the equivalent current source is a history term using the current value from previous time-step $(t-\Delta t)$ represented as:

$$I_{Leq} = i_{km}(t-\Delta t). \tag{3.21}$$

Applying the **trapezoidal rule** to integration part and reorganizing gives:

$$\begin{aligned} i_{km}(t) &= i_{km}(t-\Delta t) + \frac{\Delta t}{2L}[v_{km}(t) + v_{km}(t-\Delta t)] \\ &= \frac{\Delta t}{2L}v_{km}(t) + [i_{km}(t-\Delta t) + \frac{\Delta t}{2L}v_{km}(t-\Delta t)] \\ &= \frac{\Delta t}{2L}v_{km}(t) + I_{Leq}. \end{aligned} \tag{3.22}$$

From (3.22), the equivalent conductance G_{Leq} can be defined as:

$$G_{Leq} = \frac{\Delta t}{2L}, \tag{3.23}$$

and the equivalent current source is a history term using both of the voltage and current solutions from previous time-step $(t-\Delta t)$ represented as:

$$I_{Leq} = \frac{\Delta t}{2L}v_{km}(t-\Delta t) + i_{km}(t-\Delta t). \tag{3.24}$$

The fully comparison of the linearization methods are listed in Table 3.1 below, where h is the time-step as Δt.

Table 3.1: Comparison of the linearization methods

General Equation	$f(y,t) = \frac{dy(t)}{dt}$
Backward Euler	$y_n = y_{n-1} + hf(y_n, t_n)$
Forward Euler	$y_n = y_{n-1} + hf(y_{n-1}, t_{n-1})$
Trapezoidal Rule	$y_n = y_{n-1} + \frac{h}{2}[f(y_n, t_n) + f(y_{n-1}, t_{n-1})]$

For efficiency and accuracy purposes, the converted Backward Euler method has been implemented. Forward Euler method using the future value from the next time-step will introduce more errors. The trapezoidal rule is accurate but needs more computation steps. Since in power electronic simulation such as the diode and IGBT, the time-step needs to be small enough to obtain the transient characteristics. Then due to the small time-step, Backward Euler method is sufficient that could keep the accuracy and uses fewer computation steps compared to the Trapezoidal rule. Therefore, the hardware implementation for the passive electronic components would be based on the Backward Euler method.

3.1.1.4 Resistance and Capacitance Components - $R-C$

Similarly, the Backward Euler can applied to $R-C$ components which are connected in series by treating them as a whole element. Applying **Backward Euler** to the integration

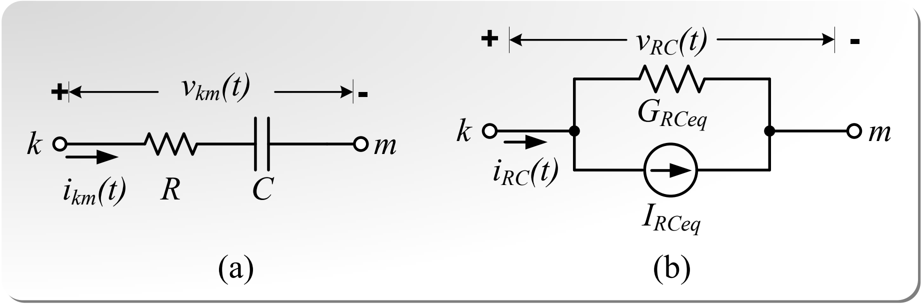


Figure 3.4: (a) $R - C$ component and (b) Norton equivalent model.

parts (C) as stated before and reorganizing gives:

$$\begin{aligned} v_{km}(t) &= v_C(t) + v_R(t) \\ &= \frac{\Delta t}{C} i_{km}(t) + v_C(t - \Delta t) + i_{km}(t)R \\ &= \frac{\Delta t}{C} i_{km}(t) + [v_{km}(t - \Delta t) - i_{km}(t - \Delta t)R] + i_{km}(t)R, \end{aligned} \tag{3.25}$$

where v_C and v_R present the voltages of the capacitor and resistor respectively. Then, the current can be obtained as:

$$\begin{aligned} i_{km}(t) &= \frac{C}{\Delta t + RC}[v_{km}(t) - v_{km}(t - \Delta t)] + \frac{RC}{\Delta t + RC} i_{km}(t - \Delta t) \\ &= G_{RCeq} v_{km}(t) + I_{RCeq}. \end{aligned} \tag{3.26}$$

(3.26) can be represented as an equivalent conductance G_{RCeq} in parallel with an equivalent current source I_{RCeq} as shown in Fig. 3.4(b), where the equivalent conductance G_{RCeq} is defined as

$$G_{RCeq} = \frac{C}{\Delta t + RC}, \tag{3.27}$$

and the equivalent current source is a history term using both of the voltage and the current solutions from previous time-step ($t - \Delta t$):

$$I_{RCeq} = \frac{RC}{\Delta t + RC} i_{km}(t - \Delta t) - \frac{C}{\Delta t + RC} v_{km}(t - \Delta t). \tag{3.28}$$

3.1.1.5 Resistance and Inductance Components - $R - L$

In the same way, the Backward Euler can also applied to $R - L$ components which are connected in series by treating them as a whole element. Applying **Backward Euler** to the integration parts – L as stated before and reorganizing gives:

$$\begin{aligned} v_{km}(t) &= v_L(t) + v_R(t) \\ &= \frac{L}{\Delta t}[i_{km}(t) - i_{km}(t - \Delta t)] + i_{km}(t)R \\ &= i_{km}(t)(\frac{L}{\Delta t} + R) - i_{km}(t - \Delta t)\frac{L}{\Delta t}, \end{aligned} \tag{3.29}$$

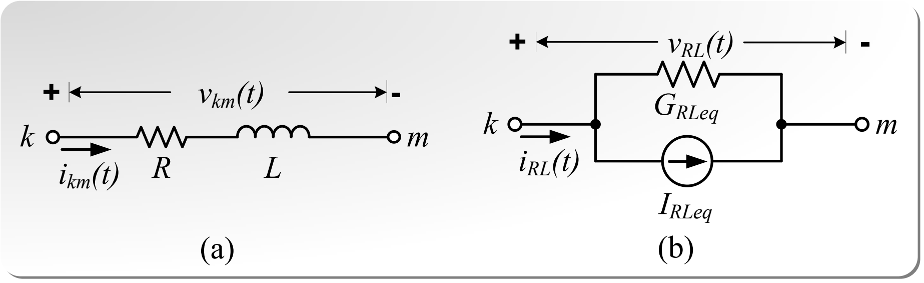


Figure 3.5: (a) $R - L$ component and (b) Norton equivalent model.

where v_L and v_R present the voltages of the inductor and resistor respectively. Then, the current can be obtained as:

$$\begin{aligned} i_{km}(t) &= \frac{\Delta t}{L + R\Delta t} v_{km}(t) + \frac{L}{R\Delta t + L} i_{km}(t - \Delta t) \\ &= G_{RLeq} v_{km}(t) + I_{RLeq}. \end{aligned} \tag{3.30}$$

(3.30) can be represented as an equivalent conductance G_{RLeq} in parallel with an equivalent current source I_{RLeq} as shown in Fig. 3.5(b), where the equivalent conductance G_{RLeq} is defined as

$$G_{RLeq} = \frac{\Delta t}{R\Delta t + L}, \tag{3.31}$$

and the equivalent current source is a history term using only the current solution from previous time-step $(t - \Delta t)$:

$$I_{RLeq} = \frac{L}{R\Delta t + L} i_{km}(t - \Delta t). \tag{3.32}$$

In summary, using **Backward Euler**, the generic equation for updating RLC components equivalent currents basically obtained from the history terms can be formulated as:

$$I_{eq} = P_1 i_{km}(t - \Delta t) + P_2 v_{km}(t - \Delta t), \tag{3.33}$$

where P_1, P_2 are coefficients with respect to different type of components which are listed in Table 3.2. For resistance R, P_1, P_2 are zeros since it has no equivalent current source.

Table 3.2: Coefficients for updating RLC components equivalent history currents (3.33)

RLC elements	G_{eq}	P_1	P_2
R	$\frac{1}{R}$	0	0
C	$\frac{C}{\Delta t}$	0	$-\frac{C}{\Delta t}$
L	$\frac{\Delta t}{L}$	1	0
$R - C$	$\frac{C}{\Delta t + RC}$	$\frac{RC}{\Delta t + RC}$	$-\frac{C}{\Delta t + RC}$
$R - L$	$\frac{\Delta t}{R\Delta t + L}$	$\frac{L}{R\Delta t + L}$	0

Thus, in general, the branch current for all RCL components can be represented as following:

$$i_{km}(t) = G_{eq} v_{km}(t) + I_{eq}. \tag{3.34}$$

3.1.2 Hardware Modules of Passive Electronic Components

From the previous figures, the voltage $v_{km}(t)$ is the voltage difference between the nodes k and m. In the hardware implementation, a subtraction procedure is needed with 4-clock cycles latency as shown below in Fig. 3.6. The resistance component R only has the parameter G as a constant signal stored in RAM which does not need arithmetical operation. The hardware designs of C and L on FPGA are shown in Fig. 3.7. Fig. 3.7(a) is the hardware architecture of capacitance component and Fig. 3.7(b) is the hardware architecture of inductance component. The pre-calculated conductance G_{Ceq} and G_{Leq} as well as the constant signals P_1 and P_2 are stored in RAMs. When the corresponding modules are triggered realized by high start signal 1 at rising clock edge, the values would be transferred to the IP cores for floating point computation. As the Fig. 3.7 shown, $i(t)$ and $i_{eq}(t - \Delta t)$ could be partially executed in parallel with a critical path latency of 8-clock cycles.

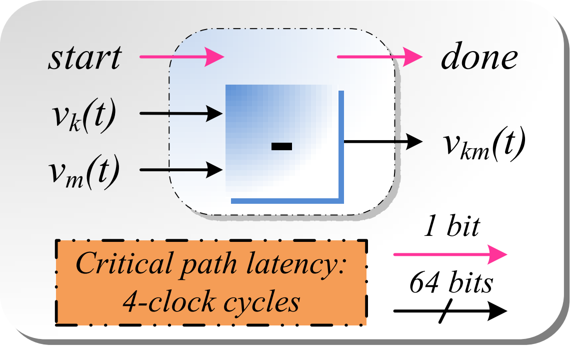


Figure 3.6: Hardware implementation Voltage difference between two nodes.

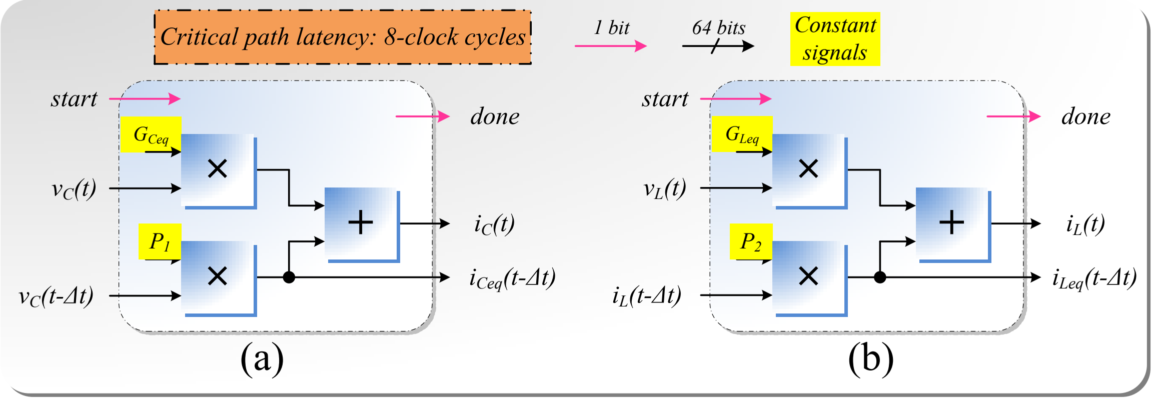


Figure 3.7: (a) Capacitance component hardware implementation and (b) Inductance component hardware implementation.

3.2 Power Diode Module

The device-level behavioral power diode model and its hardware design emulation on FPGA are provided in this section. The junction capacitance, current reverse recovery, and voltage forward recovery characteristics are the most important behavior of power diode, and they are essential items for switching power loss estimation and EMT analysis.

3.2.1 Model Description

A Saber®-based behavioral power diode model is used for the hardware design due to its accuracy, calculation speed and robustness features. It is a device level model which use parameters extracted from the available datasheet and applied to the related model equations. In order to achieve hardware design, those underlying model equations should be converted to VHDL expressions. Meanwhile, such techniques as discretization and linearization of the model are proposed. This power diode model contains the followings four sub-models based on their characteristics: static model, reverse recovery, forward recovery and junction capacitance (see Fig. 3.8(b)).

3.2.1.1 Simple Static Model

The basis for all diode dynamic models is diode static sub-model as shown in Fig. 3.8(b) which contains a contact resistance R_{on} and an ideal diode.

$$I_d = I_s \cdot \left[e^{\left(\frac{V_j}{V_b}\right)} - 1 \right], \tag{3.35}$$

$$V_b = \frac{V_{on} - R_{on}}{ln\left(1 + \frac{1}{I_s}\right)}, \tag{3.36}$$

where I_d is the current of the static diode model, V_b is the junction barrier potential, V_j is the junction voltage across the ideal diode, I_s is the leakage current and V_{on} is the knee voltage of the diode model when 1 amp current goes through it [12].

3.2.1.2 Reverse Recovery

This part combines the steady state and reverse recovery phenomena to form an equivalent model which is also converted based on the micro-model proposed in [12], [13]. Reverse recovery widely occurs in the power converters circuit. Its over-voltages and high power dissipation can affect the system dynamically during transient [12,19]. The following equations represent the behavior of this feature mathematically:

$$i_d(t) = I_{rrm} \cdot \left[e^{-\frac{R_L \cdot (t - t_s)}{L}} \right], \tag{3.37}$$

where t_s is the time then the current reaches $-I_{rrm}$, and the parameters such as I_{rrm}(peak value of reverse current), t_{rr}(reverse recovery time) are usually provided by most diode

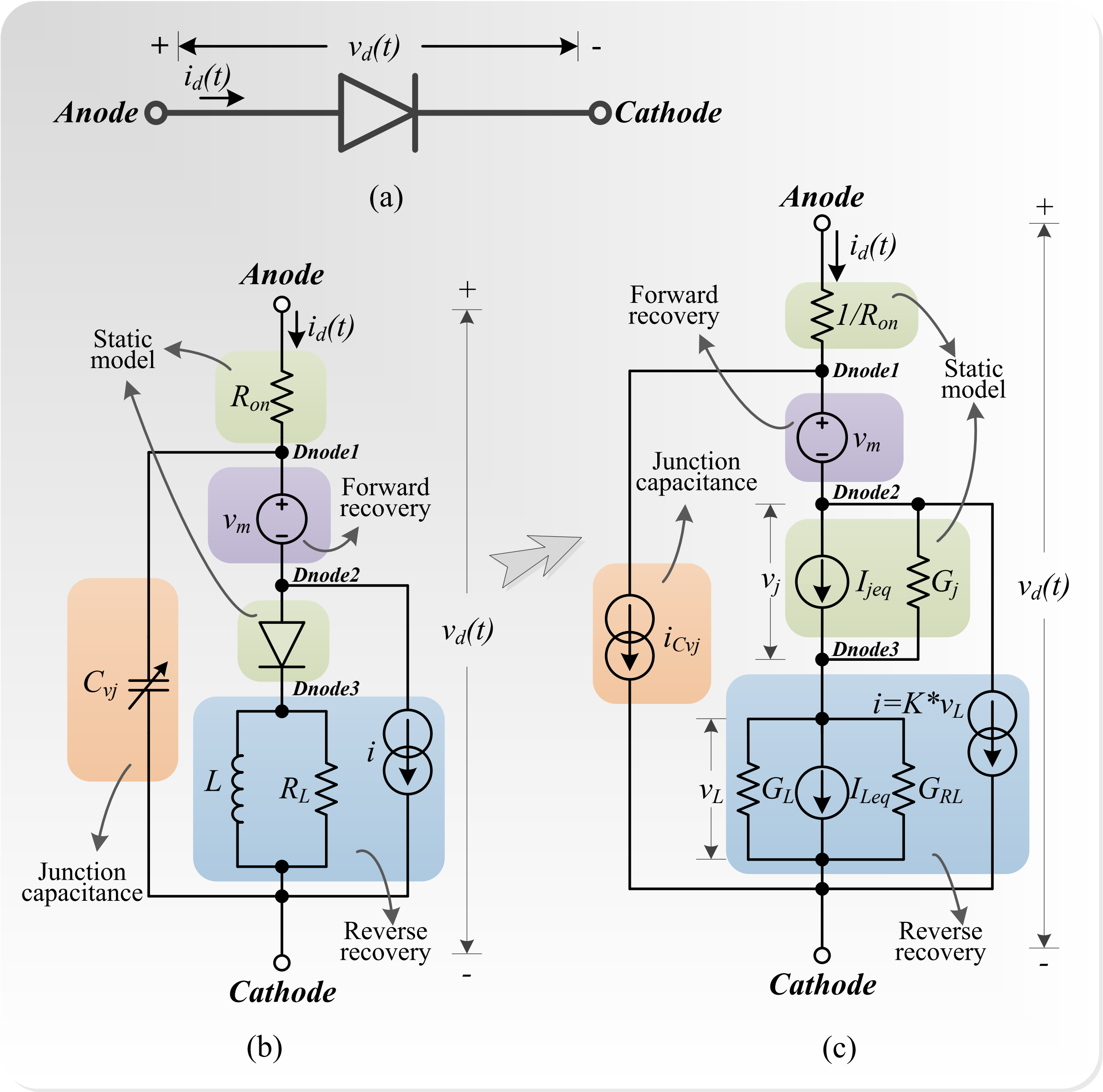


Figure 3.8: (a) Device Schematic of Power Diode. (b) Nonlinear Behavioral Power Diode Model. (c) Linearized Discrete-time Equivalent Model of Power Diode.

data sheets. As shown in Fig. 3.8, the reverse recovery effect is modeled by a first-order circuit composed of a resistor R_L in parallel with an inductor L and a voltage controlled current source i with the coefficient K. Then the model only depends on two intrinsic parameters related to the value of L/R_L and K which can be solved from the followings:

$$R_L \quad = \quad \frac{ln10 \cdot L}{t_{rr} - I_{rrm}(\frac{dt}{dI_r})}, \tag{3.38}$$

$$K = f(K) = \frac{I_{rrm}}{L}(\frac{dt}{dI_r}) \cdot \left\{1 - exp\left[\frac{-I_{Fo} - I_{rrm}}{L\frac{dI_r}{dt}(K + \frac{1}{R_L})}\right]\right\}^{-1}, \tag{3.39}$$

where I_{Fo} is on-state current before turn off process and dI_r/dt indicates current slope at turn-off. As seen in (3.39), the value of K is related to itself which form a equation with respect to $f(K)$. Since the corresponding function $f(K)$ is guaranteed to meet the demands as the condition in (3.39) for any acceptable parameters in reverse recovery condition, K can be solved by using the iterative method described in [19].

$$-1 < \frac{d[f(K)]}{dK} < 1, \tag{3.40}$$

Referring to Fig. 3.8(b), without the forward recovery part v_m and the junction capacitance, the model can be given equivalently by the Kirchhoff's current law:

$$\begin{aligned} i_d &= i + i_{R_L} + i_L \\ &= K \cdot v_L + \frac{v_L}{R_L} + i_L, \end{aligned} \tag{3.41}$$

where i_{R_L} is the current of the resistor R_L, i_L is the current of the inductor L and v_L is the voltage across the inductor L or R_L). Then, the (3.41) can be expanded equivalently as followings:

$$\begin{aligned} i_d &= R_L \cdot K \cdot \left(i_{R_L} + \frac{i_{R_L}}{R_L \cdot K} + \frac{i_L}{R_L \cdot K} \right) \\ &= R_L \cdot K \cdot \left[(i_{R_L} + i_L) \cdot \left(1 + \frac{1}{R_L \cdot K} \right) - i_L \right] \\ &= \frac{KL \left(1 + \frac{1}{R_L K}\right)^2 (i_{R_L} + i_L) - KL \left(1 + \frac{1}{R_L K}\right) i_L}{\frac{L}{R_L} \left(1 + \frac{1}{R_L K}\right)}, \end{aligned} \tag{3.42}$$

The steady state and reverse recovery behaviors of Lautirzen's diode model [12] are described as following:

$$i_d(t) = \frac{q_e(t) - q_m(t)}{T_m}, \tag{3.43}$$

$$i_d(t) = \frac{dq_m(t)}{dt} + \frac{q_m(t)}{\tau}, \tag{3.44}$$

$$q_e(t) = I_s^\star \cdot \tau [e^{\frac{v_j(t)}{2V_t^\star}} - 1], \tag{3.45}$$

$$v_d(t) = v_j(t) + R_{on} \cdot i_d(t), \tag{3.46}$$

where $i_d(t)$ is the diode current; $q_e(t)$ is the junction charge variable; $q_m(t)$ is the charge in the lightly doped region; T_m is the diffusion transit time; τ is the carrier lifetime; $I_s^\star$ is the diffusion leakage current constant; $v_j(t)$ is the junction voltage; the constant $V_t^\star$ is the junction barrier voltage and the constant R_{on} is the internal series resistance. The differential term in (3.44) will be eliminated in during steady state mode. Therefore, combine equation (3.43) and (3.44), the q_e term in steady state condition can be described as:

$$q_e = (\tau + T_m) \cdot i_d, \tag{3.47}$$

Then substitution the 3.47 into 3.45 gives:

$$i_d = \frac{I_s^\star \cdot \tau}{(\tau + T_m)} \cdot \tau[e^{\frac{v_j}{2V_t^\star}} - 1], \tag{3.48}$$

By comparing equations from (3.41) to (3.45), the relationships linking τ, T_m, q_m and q_e can be obtained as shown below:

$$\begin{aligned}
\tau &= KL\left(1 + \frac{1}{R_L K}\right), & (3.49)\\
T_m &= \frac{L}{R_L}\left(1 + \frac{1}{R_L K}\right), & (3.50)\\
q_m(t) &= KL\left(1 + \frac{1}{R_L K}\right) i_L(t), & (3.51)\\
q_e(t) &= KL\left(1 + \frac{1}{R_L K}\right)^2 \left(i_{R_L}(t) + i_L(t)\right), & (3.52)
\end{aligned}$$

Also, the equivalent expressions for $I_s^\star$ and $V_t^\star$ in circuit model corresponding to the behavioral model I_s and V_b can be obtained.

$$\begin{aligned}
I_s^\star &= I_s \cdot \left(1 + \frac{1}{R_L K}\right), & (3.53)\\
V_t^\star &= \frac{V_b}{2}, & (3.54)
\end{aligned}$$

3.2.1.3 Forward Recovery

The forward recovery phenomenon is also a diode major dynamic characteristic. The voltage overshoot occurs during the transient from diode non-conducting to conducting which is because the conductivity is low in the lightly-doped region at the initial and then it rapidly increases due to the increased concentration of injected carriers [13, 19]. In order to characterize the forward recovery voltage, a dependent voltage source $v_m(t)$ shown in Fig. 3.8(b) is modeled as:

$$v_m(t) = \begin{cases} 0 & \frac{dq_m}{dt}(t) < 0 \\ \frac{i_d^2(t)}{\frac{q_e^2(t)}{\beta\tau} + \frac{I_d(t)}{R_0}} - \frac{i_d(t)}{\frac{\tau}{\beta}\left(1+\frac{T_m}{\tau}\right)^2 i_d(t) + \frac{1}{R_0}} & \frac{dq_m}{dt}(t) \geq 0 \end{cases}, \tag{3.55}$$

where $i_d(t)$ represents the total diode current, R_0 is the initial resistance in the lightly-doped region and β is a parameter depending on various geometrical and physical parameters of the diode. The method and the procedure of extracting R_0 and β for this sub-model are described in the equation section of a MAST template [13, 19].

3.2.1.4 Junction Capacitance

As shown in Fig. 3.8(b), the last feature included in the model is the junction capacitance which is based on the width of the space charge regions and the applied cross voltage [13, 19]. This dynamic effect often appears as oscillations at turn-off [19] and can be formulated as:

$$C_{vj} = \begin{cases} \frac{C_0}{\left(1-\frac{v_j}{V_b}\right)^m} & v_j < 0 \\ C_0\left(1+\frac{m\cdot v_j}{V_b}\right) & v_j \geq 0 \end{cases}, \tag{3.56}$$

where m is the gradient factor typically ranging from 0.3 to 0.5 and C_0 is junction capacitance for zero bias. From (3.56), the capacitor C_{vj} is nonlinear and its value is actually related to v_j between $Dnode2$ and $Dnode3$ rather than the voltage between $Dnode1$ and $Cathode$ in Fig. 3.8. The current through the junction capacitance can be treated as a voltage controlled current source defined by the following:

$$i_{Cvj} = \frac{d}{dt}\int C_{vj}dv_j, \tag{3.57}$$

Then, combining equations (3.56) and (3.57), i_{Cvj} can be expressed as:

$$i_{Cvj} = \begin{cases} \frac{d}{dt}\left[\frac{C_0\cdot V_b}{m-1}\left(\left(1-\frac{v_j}{V_b}\right)^{1-m}-1\right)\right] & v_j < 0 \\ \frac{d}{dt}\left[C_0\cdot v_j\cdot\left(1+\frac{m\cdot v_j}{2V_b}\right)\right] & v_j \geq 0 \end{cases}, \tag{3.58}$$

To sum up, as seen in Fig. 3.8, the complete power diode model is all the above characteristic sub-models including static model, voltage forward recovery, current reverse recovery and junction capacitance combined together.

3.2.2 Model Discretization and Linearization

From (3.35) to (3.58), the power diode model is demonstrated as a nonlinear system. Thus, the model need to be discretized and linearized so that a discrete-time equivalent circuit can be obtained for simulation. Such methods as Forward Euler, Backward Euler, Trapezoidal and Runge-Kutta can be used for solving ordinary differential equations. In this thesis, the Backward Euler Method is introduced to discretize and linearize the nonlinear elements as stated before. The process for the linear passive elements in this model such as the inductor L and R_L can be easily achieved found in the previous section. The inductor L can be expressed as its equivalent circuit (a current source in parallel with a conductance as shown in Fig. 3.8):

$$G_L = \frac{L}{\Delta t}, \tag{3.59}$$

$$I_{Leq} = i_L(t-\Delta t), \tag{3.60}$$

$$i_L(t) = I_{Leq} + G_L\cdot v_L(t). \tag{3.61}$$

The differential term, junction capacitance current differential equation (3.58), can be discretized as:

$$i_{Cvj} = \begin{cases} \frac{C_0\cdot V_b}{\Delta t\cdot(m-1)}\cdot\left[\left(1-\frac{v_j(t)}{V_b}\right)^{(1-m)}-\left(1-\frac{v_j(t-\Delta t)}{V_b}\right)^{(1-m)}\right] & v_j < 0 \\ \frac{C_0}{\Delta t}\cdot\left[v_j(t)+\frac{m\cdot v_j^2(t)}{2V_b}-v_j^2(t-\Delta t)-\frac{m\cdot v_j^2(t-\Delta t)}{2V_b}\right] & v_j \geq 0 \end{cases}, \tag{3.62}$$

As shown in Fig. 3.8(c), the junction capacitance can be discretized as follows:

$$G_{Cvj} = \begin{cases} \frac{C_0}{\Delta t \cdot \left(1 - \frac{v_j}{V_b}\right)^m} & v_j < 0 \\ \frac{C_0}{\Delta t} \cdot \left(1 + \frac{m \cdot v_j}{V_b}\right) & v_j \geq 0 \end{cases}, \tag{3.63}$$

Then, the corresponding equivalent current can be found by:

$$I_{Cvjeq} = i_{Cvj} - G_{Cvj} \cdot v_j(t), \tag{3.64}$$

Similarly, the steady state or DC ideal diode inside the model based on (3.36) with across voltage of v_j are linearized to obtain the corresponding conductance and current source pair G_j and I_{jeq} in Fig. 3.8(c) respectively:

$$i_j = I_s \cdot [e^{\frac{v_j(t)}{V_b}} - 1], \tag{3.65}$$

$$G_j = \frac{\partial i_j}{\partial v_j} = \frac{I_s}{V_b} e^{\frac{v_j(t)}{V_b}}, \tag{3.66}$$

$$I_{jeq} = i_j - G_j \cdot v_j(t), \tag{3.67}$$

While, this intrinsic or internal diode can be treated as a piecewise linear diode described later.

The forward recovery phenomenon (3.55) is represented as a controlled voltage source, so there will be a super-node in nodal analysis such as $v_{Dnode1} = v_{Dnode2} + v_m$. Finally, the complete power diode model is discretized and linearized shown in Fig. 3.8(c). It can be expressed as follows based on Kirchhoff's current and voltage law:

$$\mathbf{G}^{Diode} \cdot \mathbf{v}^{Diode} = \mathbf{I}_{eq}^{Diode}, \tag{3.68}$$

where the 4×4 conductance matrix is given as:

$$\mathbf{G}^{Diode} = \begin{bmatrix} \frac{1}{R_{on}} & -\frac{1}{R_{on}} & 0 & 0 \\ -\frac{1}{R_{on}} & \frac{1}{R_{on}} + G_j + G_{Cvj} & K - G_j - G_{Cvj} & -K \\ 0 & -G_j & G_j + G_L + G_{RL} & -G_L - G_{RL} \\ 0 & -G_{Cvj} & G_{Cvj} - G_L - G_{RL} - K & G_L + G_{RL} + K \end{bmatrix}, \tag{3.69}$$

the node voltage vector is given as

$$\mathbf{v}^{Diode} = \begin{bmatrix} v_{Anode}, & v_{Dnode2}, & v_{Dnode3}, & v_{Cathode} \end{bmatrix}^T, \tag{3.70}$$

and the equivalent current source vector is given as

$$\mathbf{I}_{eq}^{Diode} = \begin{bmatrix} \frac{v_m}{R_{on}}, & -I_{jeq} - I_{Cvjeq} - \frac{v_m}{R_{on}}, & I_{jeq} - I_{Leq}, & I_{Leq} + I_{Cvjeq} \end{bmatrix}^T. \tag{3.71}$$

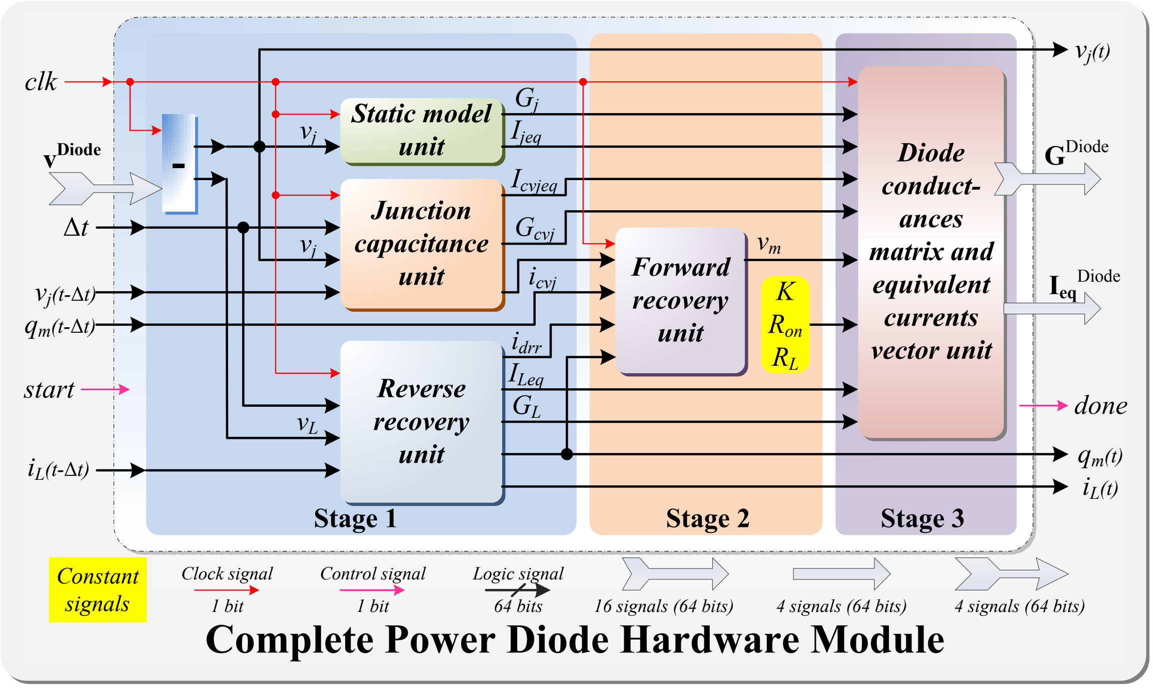


Figure 3.9: Architecture of complete power diode hardware module.

3.2.3 Hardware Emulation on FPGA

Fig. 3.9 shows the architecture of the complete power diode hardware module which is composed of six subunits basically implementing in 3 stages. At stage 1, after finishing the computation of corresponding voltages, three sub-units which are the Static Model Unit,

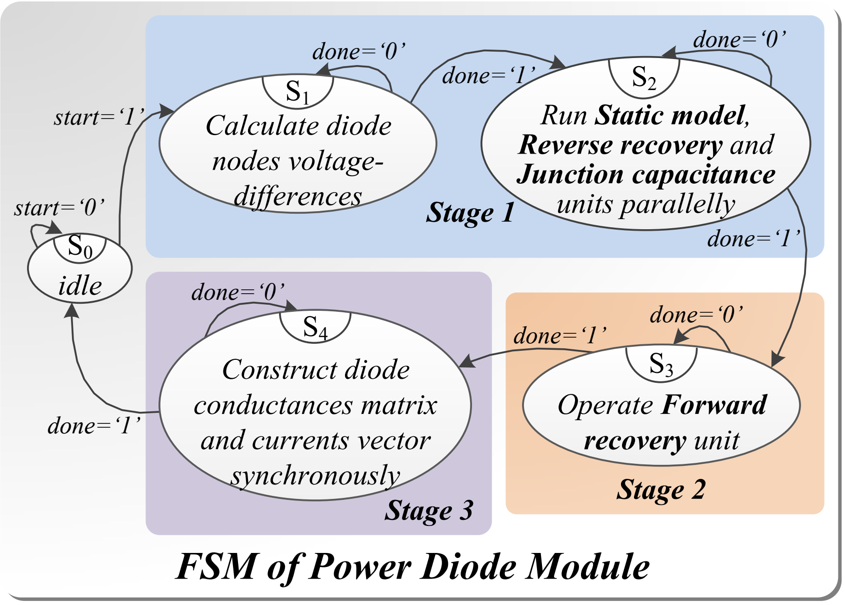

start='0'
S_0
idle
start='1'
done='0'
S_1
Calculate diode nodes voltage-differences
done='1'
done='0'
S_2
Run Static model, Reverse recovery and Junction capacitance units parallelly
Stage 1
done='1'
done='0'
S_4
Construct diode conductances matrix and currents vector synchronously
Stage 3
done='1'
done='1'
done='0'
S_3
Operate Forward recovery unit
Stage 2
FSM of Power Diode Module

Figure 3.10: Finite state machine of the complete power diode module.

the Reverse Recovery Unit, and the Junction Capacitance Unit would implement in parallel. Forward Recovery Unit at stage 2 will be triggered after the previous three units accomplishing their calculation as its input signals are based on their output signals. Lastly, at Stage 3, the conductance matrix and the equivalent current source vector are constructed based on the calculated results from previous sub-units hardware models. This hardware structure, not only among the 6 units but within a single sub-unit, is highly paralleled and pipelined. For example, within the Reverse Recovery Unit, the calculations of G_L, I_{Leq} and i_L for the passive component L pipeline with q_m and i_{drr} calculation and their respective implementation are paralleled. The detailed module of the inductance component L can be found in previous section 3.1.1.3 and its hardware structure is the same as shown in Fig. 3.7(b). Moreover, even those computations operators such as adders, multipliers and so on from IP catalog conform to such policy of deep pipeline and parallelism. Since such terms as K, R_L, and τ are constant, they are pre-calculated before used in the hardware design as constant signals. Therefore, computational complexity and latency can be adequately reduced. The finite state machine in Fig. 3.10 is to control the operation sequence and process of this module. At stage 1, after the required voltage differences calculated in S_1, the 3 aforementioned sub-units will be triggered simultaneously and implement in parallel in S_2. Then, the rest forward recovery unit executes in S_3 at stage 2 before the formation of $\mathbf{G}^{Diode}$ and $\mathbf{I}_{eq}^{Diode}$ within the next state S_4 at stage 3.

3.3 IGBT Module

IGBT has become the most widely used semiconductor in various power electronic applications since it has such advantages as large input impedance, convenient control, high operating frequency and so on. The non-physical based Saber® *IGBT1* model is an accurate IGBT behavioral model featuring both static and dynamic characteristics, non-linear elements such as inter-electrode capacitance and hard turn-off tail current. It can accurately represent the device behavior for steady-state and transient analysis.

3.3.1 Model Formulation

The behavioral Saber® *IGBT1* model and its equivalent circuit are shown in Fig. 3.11. A piecewise linear diode represented as a PWLD is placed between collector and Inode1 shaded in green. The blue shade indicates that the inside elements are passive components mentioned previously. While, the purple and orange shades cover the elements which are nonlinear components such as nonlinear voltage controlled current sources i_{mos} and i_{tail} as well as the nonlinear capacitors C_{ce} and C_{cg}. R_g is the resistance to the gate pole. Both the collector to emitter v_{ce} and the gate to emitter v_{ge} voltages determine the current source i_{mos} value. The turn-off tail current i_{tail} is controlled by the internal R_{tail} and C_{tail} which are parallel connected with each other and sited between current source i_{mos} and emitter.

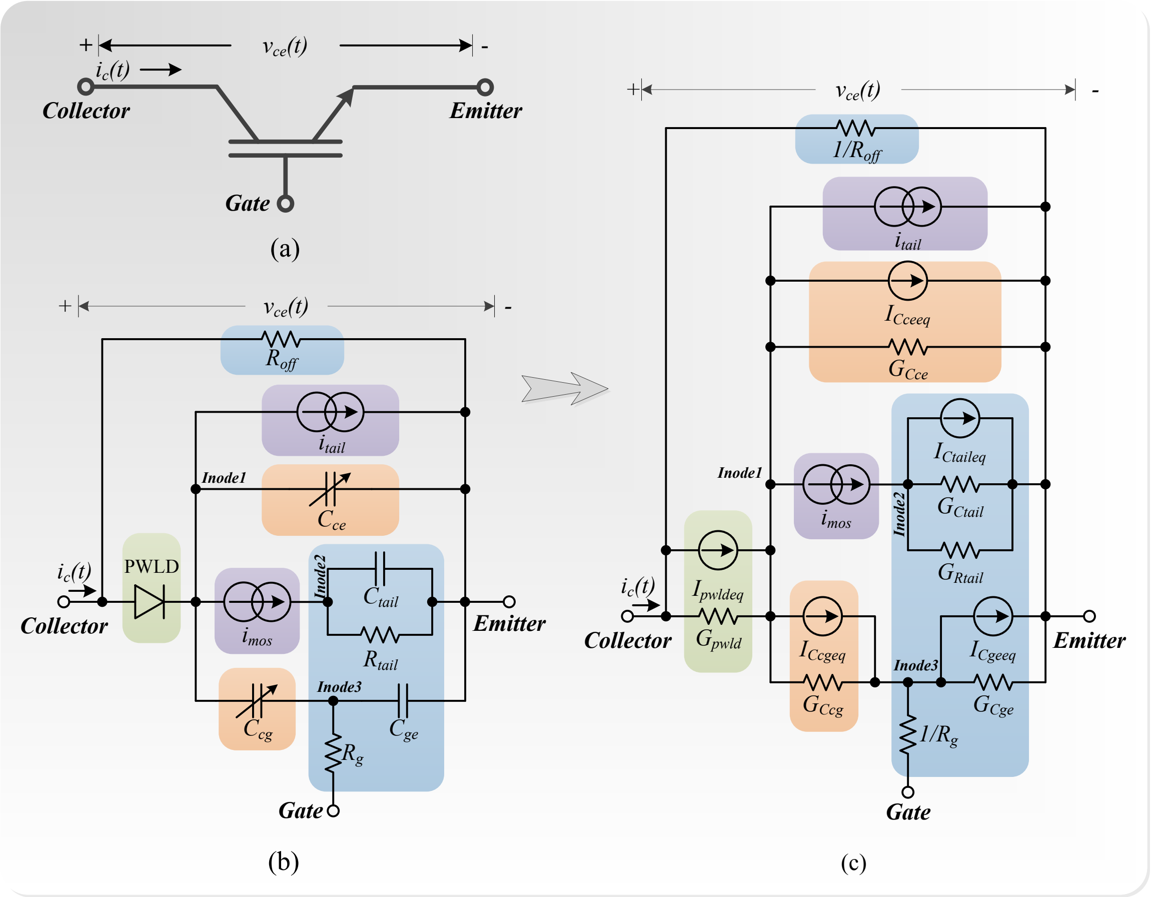


Figure 3.11: (a) Device Schematic of Power Diode. (b) Nonlinear Behavioral Saber® *IGBT1* Model. (c) Linearized Discrete-time Equivalent Model of Saber® *IGBT1*.

It occurs when the IGBT is turned off with a gate command. The inter-electrode capacitors (C_{ce},C_{cg},C_{ge}), which are accurately modeled to generate switching behaviors and losses, meet the following relationship corresponding to the curves provided by data sheets [31]:

$$C_{rss} = C_{cg}, \tag{3.72}$$

$$C_{oss} = C_{cg} + C_{ce}, \tag{3.73}$$

$$C_{iss} = C_{cg} + C_{ge}, \tag{3.74}$$

where C_{rss} indicates the reverse transfer capacitance, C_{oss} presents the output capacitance, and C_{iss} is the input capacitance.

The basic operation can be summarized as when the collector-emitter voltage value v_{ce} is less than the threshold voltage V_{on}, the piecewise linear diode keep off and collector current i_c is zero in DC mode; while, when the value of v_{ce} is between V_{on} and V_{sat}, the device behavior will be represented as i_{mos} in the quasi-linear region; then, when v_{ce} is greater than the saturation voltage V_{sat}, i_c will mainly depend on the gate-emitter voltage v_{ge} and v_{ce} as well [30]. Using the IGBT tools in SaberRD®, the corresponding static parameters of $IGBT1$ can be acquired based on the characteristics of the device in data sheets. The dynamic parameters such as the non-linear capacitances can be obtained from the inter-electrode capacitance curves and gate-charge plot using the IGBT tool as well. Such the tail terms as C_{tail}, R_{tail} and i_{tail} only have influence on device behaviors during turn-off. The relative parameters can be extracted by experimentally measuring the IGBT current in a chopper circuit as mentioned in Saberbook.

The PWLD in the behavioral *IGBT1* model can be basically treated as binary conductance model where g_{on} is diode on-state conductance and g_{off} is diode off-state conductance. Based on the SaberRD® software R_{on} / R_{off} ideal diode model, the characteristic of the piecewise linear diode can be approximately modeling as followings:

$$i_{pwld} \approx \begin{cases} i_{max} & (v_{pn} > V_{limit}) \\ g_{on} \cdot (v_{pn} - V_{onpwld}) & (v_{pn} > V_{onpwld}) \\ g_{off} \cdot (v_{pn}) & (others) \end{cases}, \tag{3.75}$$

where v_{pn} is the voltage across the PWLD, V_{onpwld} is the PWLD forward threshold voltage which could be regarded as V_{on} for the collector to emitter threshold voltage when the IGBT conducts. V_{limit} is the diode limiting rated voltage and i_{max} is the diode maximum current. It could also be simplified as:

$$G_{pwld} = \begin{cases} g_{on} & (v_{pn} > V_{onpwld}) \\ g_{off} & (v_{pn} \leq V_{onpwld}) \end{cases}, \tag{3.76}$$

$$i_{pwld} \approx G_{pwld} \cdot (v_{pn} - V_{onpwld}), \tag{3.77}$$

$$I_{pwldeq} = -G_{pwld} \cdot V_{onpwld}, \tag{3.78}$$

3.3.2 Model Discretization and Linearization

Fig. 3.11(c) illustrates the discrete-time linearized Saber® *IGBT1* model. Using the similar procedure given in previous diode section, this IGBT model can be discretized and linearized based on the inside components. Since the voltage controlled current sources (e.g. i_{mos} and i_{tail}) depend on different nodes voltage difference which is similar to the diode reverse recovery current i unit, they could not be drawn out as several parts and shown in the figure. However, rather than that, they could still be treated as a combination of conductances (e.g. G_{mosvd}) and equivalent current sources (e.g. I_{moseq}) in discretization and linearization process. Some circuit elements such as C_{tail} and C_{ge} which are the passive electronic components can be discretized as follow:

$$G_{Ctail} = \frac{C_{tail}}{\Delta t}, \tag{3.79}$$

$$i_{Ctail} = G_{Ctail} \cdot [v_{Ctail}(t) - v_{Ctail}(t - \Delta t)], \tag{3.80}$$

$$I_{Ctaileq} = -G_{Ctail} \cdot v_{Ctail}(t - \Delta t), \tag{3.81}$$

$$G_{Cge} = \frac{C_{ge}}{\Delta t}, \tag{3.82}$$

$$i_{Cge} = G_{Cge} \cdot [v_{Cge}(t) - v_{Cge}(t - \Delta t)], \tag{3.83}$$

$$I_{Cgeeq} = -G_{Cge} \cdot v_{Cge}(t - \Delta t), \tag{3.84}$$

For the nonlinear capacitance, the C_{cg} capacitor unit can be discretized as follows:

$$G_{Ccg} = \begin{cases} \frac{(ccgo \cdot (1 + \frac{v_{Ccg}}{vcgo})^{-m})}{\Delta t} & v_{Ccg} > 0 \\ \frac{ccgo}{\Delta t} & v_{Ccg} \leq 0 \end{cases}, \tag{3.85}$$

$$i_{Ccgeq} = \frac{q_{Ccg}(t) - q_{Ccg}(t - \Delta t)}{\Delta t} - G_{Ccg} \cdot v_{Ccg}(t), \tag{3.86}$$

where m is the Miller capacitance exponent coefficient and usually set to be 0.5 as default (typically ranges from 0.3 to 0.8). This coefficient can affect the current rise and fall times as well as the switching losses. Similarly, C_{ce} unit can be discretized as:

$$G_{Cce} = \begin{cases} \frac{(cceo \cdot (1 + \frac{v_{Cce}}{vceo})^{-0.5}}{\Delta t} & v_{Cce} > 0 \\ \frac{cceo}{\Delta t} & v_{Cce} \leq 0 \end{cases}, \tag{3.87}$$

$$I_{Cceeq} = \frac{q_{Cce}(t) - q_{Cce}(t - \Delta t)}{\Delta t} - G_{Cce} \cdot v_{Cce}(t), \tag{3.88}$$

As can be seen, although the nonlinear capacitors are more complex, they could be dealt in the same way. The voltage controlled current sources (e.g. i_{mos} and i_{tail}) are more complicated. They are discretized and linearized to multiple conductance parts and corresponding the equivalent sources need to be calculated based on all the relative conductance. The i_{mos} current unit can brunch off the conductance $G_{mosvcge}$ which can be derived by $\frac{\partial i_{mos}}{\partial v_{Cge}}$ formulated as :

$$G_{mosvcge} = \begin{cases} 0 & (v_{Cge} < V_t) || (vd \leq 0) \\ \frac{\partial a2}{\partial v_{Cge}} \cdot v_d^{(z+1)} - \frac{\partial b2}{\partial v_{Cge}} \cdot v_d^{(z+2)} & v_d < (y \cdot (v_{Cge} - V_t))^{\frac{1}{x}} \\ \frac{2 \cdot (v_{Cge} - V_t)}{a1 + b1 \cdot (v_{Cge} - V_t)} - \frac{b1 \cdot (v_{Cge} - V_t)^2}{(a1 + b1 \cdot (v_{Cge} - V_t))^2} & (others) \end{cases}, \tag{3.89}$$

and G_{mosvd} which can be derived by $\frac{\partial i_{mos}}{\partial v_d}$ formulated as:

$$G_{mosvd} = \begin{cases} 0 & (v_{Cge} < V_t) || (vd \leq 0) \\ a2(z+1)v_d^z - b2(z+2)v_d^{(z+1)} & v_d < (y \cdot (v_{Cge} - V_t))^{\frac{1}{x}} \\ 0 & (others) \end{cases}, \tag{3.90}$$

and its equivalent current I_{moseq} can be obtained as:

$$I_{moseq} = i_{mos} - G_{mosvd} \cdot v_d - G_{mosvcge} \cdot v_{Cge}, \tag{3.91}$$

where $a1, b1, a2, b2, z$ are internal parameters obtained from Saber®, V_t is the channel threshold voltage on the i_c to v_{ge} transfer characteristic and v_d is the voltage difference between $Inode1$ and $Inode2$ across the i_{mos}. Similarly, the equivalent conductances from i_{tail} unit can also be found as:

$$G_{tailvtail} = \begin{cases} 0 & \frac{v_{tail}}{R_{tail}} \leq i_{mos} \\ \frac{itail2}{R_{tail} \cdot itail1} & \frac{v_{tail}}{R_{tail}} > i_{mos} \end{cases}, \tag{3.92}$$

$$G_{tailvd} = \begin{cases} 0 & \frac{v_{tail}}{R_{tail}} \leq i_{mos} \\ -\frac{itail2}{G_{mosvd} \cdot itail1} & \frac{v_{tail}}{R_{tail}} > i_{mos} \end{cases}, \tag{3.93}$$

$$G_{tailvcge} = \begin{cases} 0 & \frac{v_{tail}}{R_{tail}} \leq i_{mos} \\ -\frac{itail2}{G_{mosvcge} \cdot itail1} & \frac{v_{tail}}{R_{tail}} > i_{mos} \end{cases}, \tag{3.94}$$

and the equivalent current sources as:

$$I_{taileq} = i_{tail} - G_{tailvd} \cdot v_d - G_{tailvcge} \cdot v_{Cge} - G_{tailvtail} \cdot v_{tail}, \tag{3.95}$$

From (3.89) to (3.95), the partitioned conductances from the i_{mos} and i_{tail} are all nonlinear. They are cross correlated to other branch voltages rather than the voltage between v_{Inode1} and v_{Inode2} for i_{mos} or the voltage between v_{Inode1} and $v_{Emitter}$ for i_{tail} only. For example, $G_{mosvcge}$ is the i_{mos} taking partial derivative with respect to v_{Cge} which is the voltage difference between v_{Inode3} and $v_{Emitter}$. Moreover, the conductances from i_{tail} are even related to the conductance from i_{mos}. Thus, these features make the model more complicated. Finally all the sub-units are combined to construct a 5×5 conductance matrix $\mathbf{G}^{IGBT}$ and current source vector $\mathbf{I}_{eq}^{IGBT}$. With IGBT node voltage vector $\mathbf{v}^{IGBT}$ together, a resulting linear system of equations of this model can be shown below:

$$\mathbf{G}^{IGBT} \cdot \mathbf{v}^{IGBT} = \mathbf{I}_{eq}^{IGBT}, \tag{3.96}$$

where $\mathbf{G}^{IGBT}$, $\mathbf{v}^{IGBT}$ and $\mathbf{I}_{eq}^{IGBT}$ are expressed in the equations (3.97), (3.98) and (3.99) respectively. Note that, V_s is the gate voltage source which is equal to the voltage difference between v_{Gate} and $v_{Emitter}$.

$$\mathbf{G}^{IGBT} = \begin{bmatrix} G_{pwld} + \frac{1}{R_{off}} & -G_{pwld} & 0 & 0 & -\frac{1}{R_{off}} \\ -G_{pwld} & \begin{array}{c} G_{Cce} + G_{Ccg} + G_{pwld} \\ +G_{mosvd} + G_{tailvd} \end{array} & \begin{array}{c} G_{tailvtail} - G_{mosvd} \\ -G_{tailvd} \end{array} & \begin{array}{c} G_{mosvcge} + G_{tailvcge} \\ -G_{Ccg} \end{array} & \begin{array}{c} -G_{mosvcge} - G_{tailvcge} \\ -G_{tailvtail} - G_{Cce} \end{array} \\ 0 & -G_{mosvd} & \begin{array}{c} G_{mosvd} + G_{tail} \\ +G_{Ctail} \end{array} & -G_{mosvcge} & \begin{array}{c} G_{mosvcge} - G_{tail} \\ -G_{Ctail} \end{array} \\ 0 & -G_{Ccg} & 0 & G_{Cge} + G_{Ccg} + \frac{1}{R_g+R_{gs}} & -G_{Cge} - \frac{1}{R_g+R_{gs}} \\ -\frac{1}{R_{off}} & -G_{Cce} - G_{tailvd} & \begin{array}{c} G_{tailvd} - G_{tailvtail} \\ -G_{tail} - G_{Ctail} \end{array} & \begin{array}{c} -G_{Cge} - G_{tailvcge} \\ -\frac{1}{R_g+R_{gs}} \end{array} & \begin{array}{c} G_{Cge} + G_{Cce} + \frac{1}{R_{off}} \\ +G_{tailvtail} + G_{tailvcge} \\ +G_{Ctail} + G_{tail} + \frac{1}{R_g+R_{gs}} \end{array} \end{bmatrix}, \quad (3.97)$$

$$\mathbf{v}^{IGBT} = \left[v_{Collector} \ \ v_{Inode1} \ \ v_{Inode2} \ \ v_{Inode3} \ \ v_{Emitter} \right]^T \quad (3.98)$$

$$\mathbf{I}_{eq}^{IGBT} = \begin{bmatrix} -I_{pwldeq} & \begin{array}{c} I_{pwldeq} - I_{moseq} - I_{taileq} \\ -I_{Cceeq} - I_{Ccgeq} \end{array} & I_{moseq} - I_{Ctaileq} & \begin{array}{c} I_{Ccgeq} - I_{Cgeeq} \\ +\frac{V_s}{R_g+R_{gs}} \end{array} & \begin{array}{c} I_{taileq} + I_{Ctaileq} \\ +I_{Cgeeq} + I_{Cceeq} \\ -\frac{V_s}{R_g+R_{gs}} \end{array} \end{bmatrix}^T, \quad (3.99)$$

3.3.3 Hardware Emulation on FPGA

In order to implement this IGBT model in hardware, all the aforementioned sub-units or components need to be designed as their corresponding hardware modules using VHDL language coding in Vivado®.

3.3.3.1 Hardware Designs of the PWLD Units

From Fig. 3.11, the junction diode unit between the node $Collector$ and node $Inode1$ is represented as a piece-wise linear diode model. In (3.76), g_{on} and g_{off} can be replaced by G_{pwld} for convenience. Its hardware structure is shown in Fig. 3.12. From the Fig. 3.12,

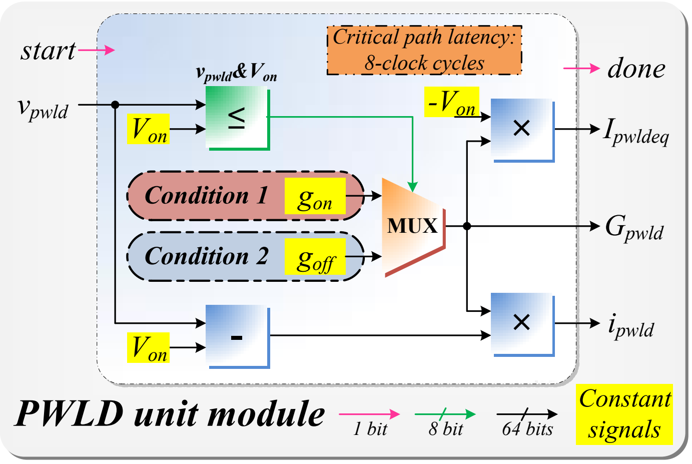


Figure 3.12: Hardware structure of PWLD unit with its latency.

G_{pwld} is chosen from g_{on} and g_{off} which are the inputs of a multiplexer controlled by a comparison IP core signal. The unit will be done after the output i_{pwld} signal is been generated. 8-clock cycles are the latency based on the sequence of a subtraction and a multiplication processes, while the I_{pwldeq} and i_{pwld} computation are still in parallel. Its finite state machine could be designed as in Fig. 3.13 to control the operation process of this hardware module. After the comparison step finished in S_1, G_{pwld} will be assigned in S_2 and transfer to S_3 for equivalent current calculation.

3.3.3.2 Hardware Designs of the Capacitance Units

The behavioral *IGBT1* model contains 4 independent capacitance units - C_{ce}, C_{cg}, C_{ge} and C_{tail}. Both C_{ge} and C_{tail} are basic passive capacitance components, as stated in section 3.1, they have the same structure and the latency, as shown in Fig. 3.2. Also, their hardware modules can run in parallel. While, different from the basic passive capacitance component which has a simple combinational circuit structure, C_{ce} and C_{cg} have more complex features based on (3.87) — (3.86) to generate not only the currents, dynamic conductances, equivalent currents but also the charges. Thus, more complicated hardware structures

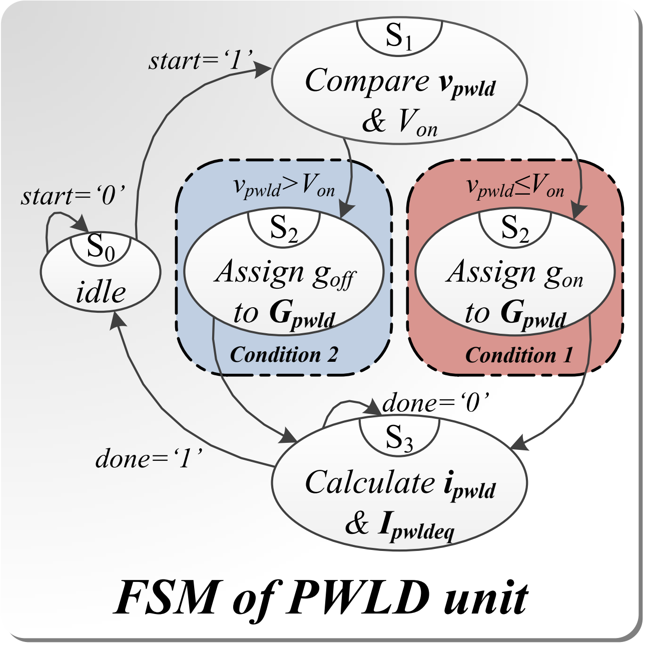


Figure 3.13: Hardware structure of PWLD unit with its latency.

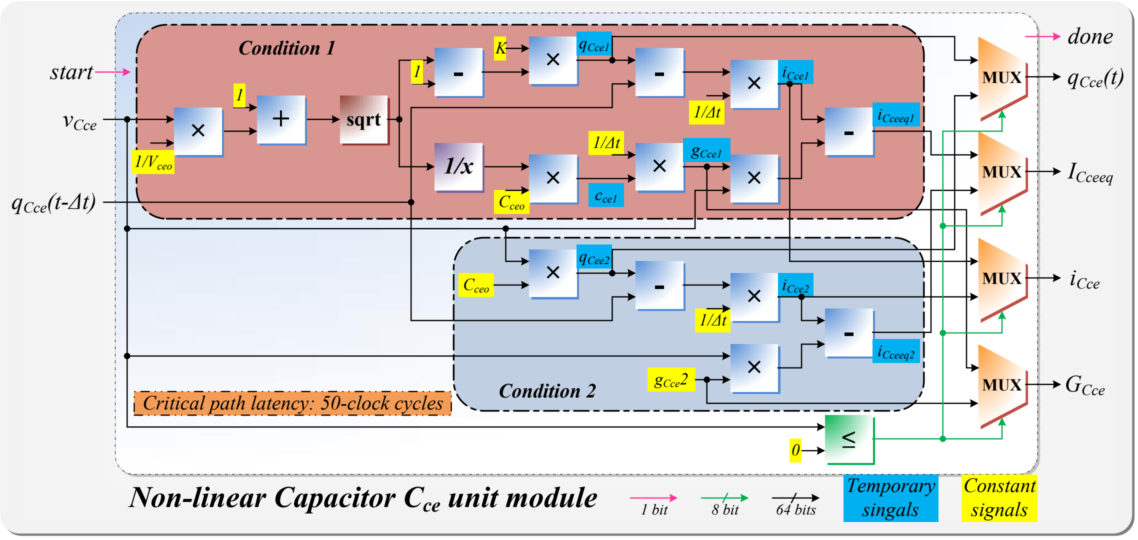


Figure 3.14: Hardware structure of C_{ce} unit with its latency.

would be formed. As an example, the hardware architecture of C_{ce} is shown in Fig. 3.14. Then, its finite state machine could be designed as shown in Fig. 3.15 to control the implementation procedure. C_{cg} unit has almost the same structure comparing to C_{ce}, and since they are independent, parallel computation can be performed. Similar to the linear passive capacitance, the calculations of these nonlinear capacitor units require the history terms from the last time-step as well. In Fig. 3.14, the voltage V_{Cce} and the history value of $q_{Cce}(t-\Delta t)$ from the last time-step are sent to the capacitor unit C_{ce} hardware module so as to generate $q_{Cce}(t)$ as one of the module output signals which can be used as the input for the next time-step calculations after N-R iterations. Basically, the outputs of this unit chosen from the results obtained in two different conditions using corresponding four multiplexers controlled by a comparison functional IP core signal. The *done* signal would be output after all the output signals generated by this hardware module. The critical path of the unit has about 50-clock cycles latency based on the sequence of subtraction, multiplication, reciprocal computation and square-root calculation processes. While some computations in condition 1 and 2 are still in parallel. Fig. 3.15 provides the finite state machine for this unit. It contains 4 states including the idle state S_0. After partially parallel calculation under both conditions in S_1 which generates two group of results, the comparison step in S_2 can choose which set of results should be assigned to outputs for S_3.

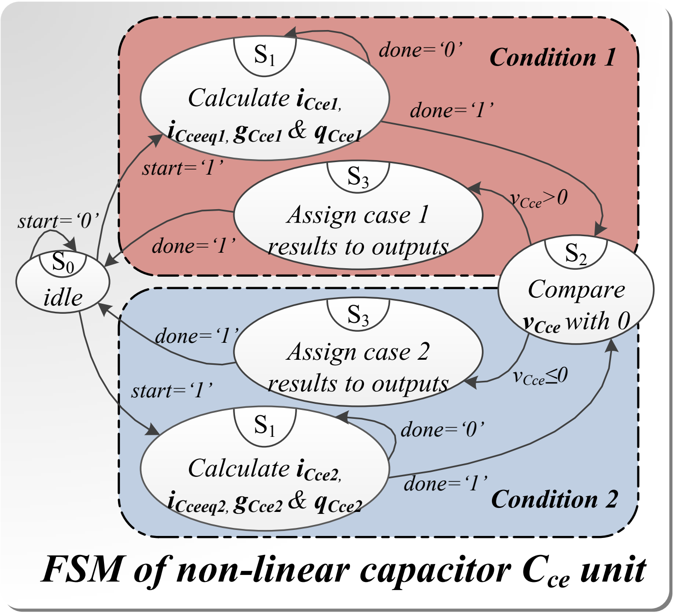

S1
Calculate iCce1, iCceeq1, gCce1 & qCce1
done='0'
Condition 1
done='1'
start='1'
S3
Assign case 1 results to outputs
vCce>0
start='0'
done='1'
S0
idle
S2
Compare vCce with 0
done='1'
S3
Assign case 2 results to outputs
vCce≤0
start='1'
S1
Calculate iCce2, iCceeq2, gCce2 & qCce2
done='0'
done='1'
Condition 2
FSM of non-linear capacitor Cce unit

Figure 3.15: Hardware structure of C_{ce} unit with its latency.

3.3.3.3 Hardware Designs of the Current Units

Fig. 3.16 illustrates voltage controlled current source i_{mos} unit hardware structure and demonstrates the parallel computation within this hardware module. According to (3.89) — (3.91), the hardware design of i_{mos} is more complicated compared to the previous capacitance units. It also contains more complex operators for logarithm and exponential operations which can be realized using IP core. As shown in Fig. 3.16, 3 comparators send their comparison results or signals to multiplexers, and the 4 multiplexers take the responsibility of selecting correct results among the 3 nested conditions. In condition 1, the results are all zeros. In the other two conditions, the temporary results for i_{mos}, G_{mosvd}, $G_{mosvcge}$

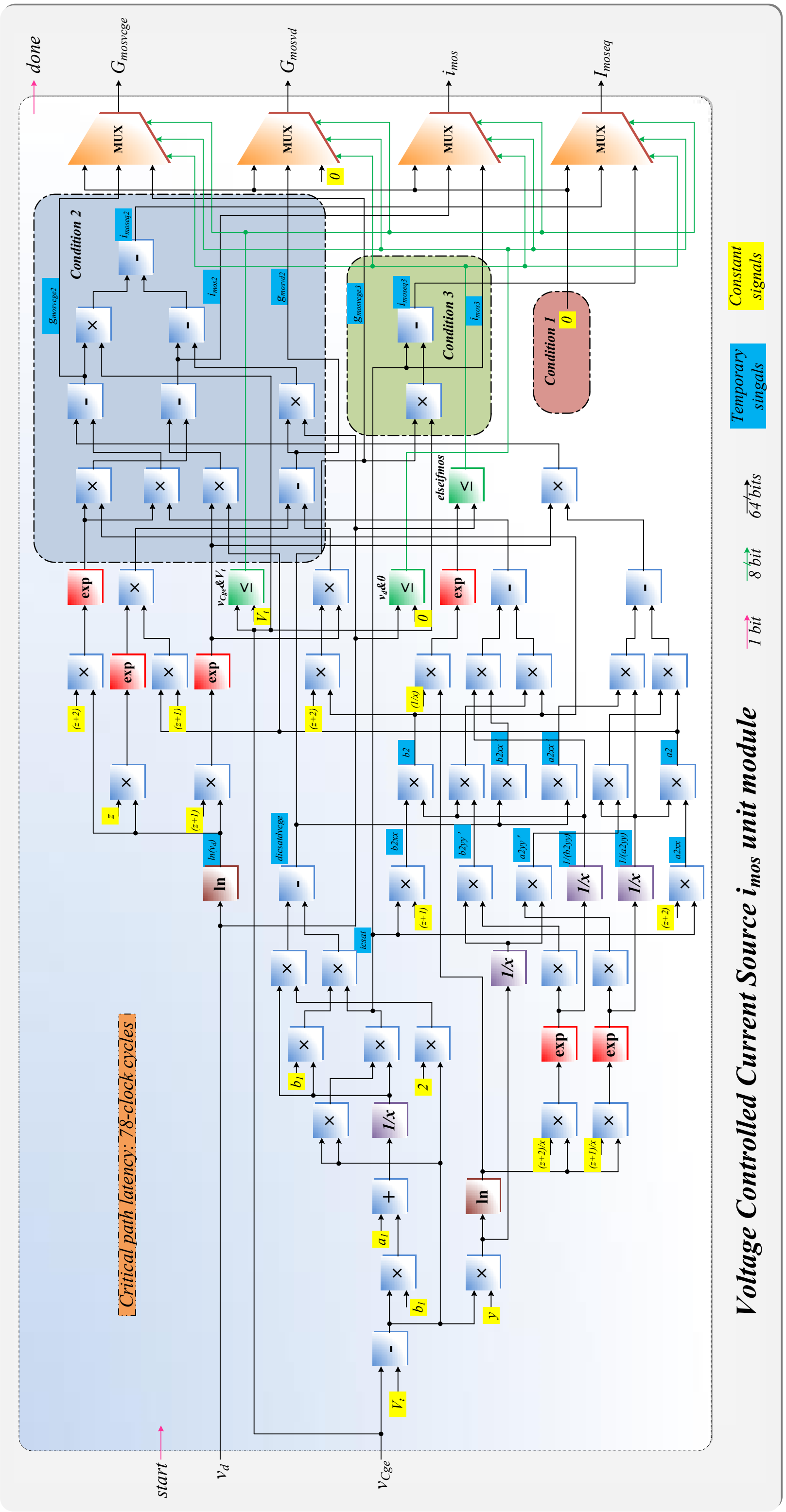


Figure 3.16: Hardware structure of the voltage controlled i_{mos} unit.

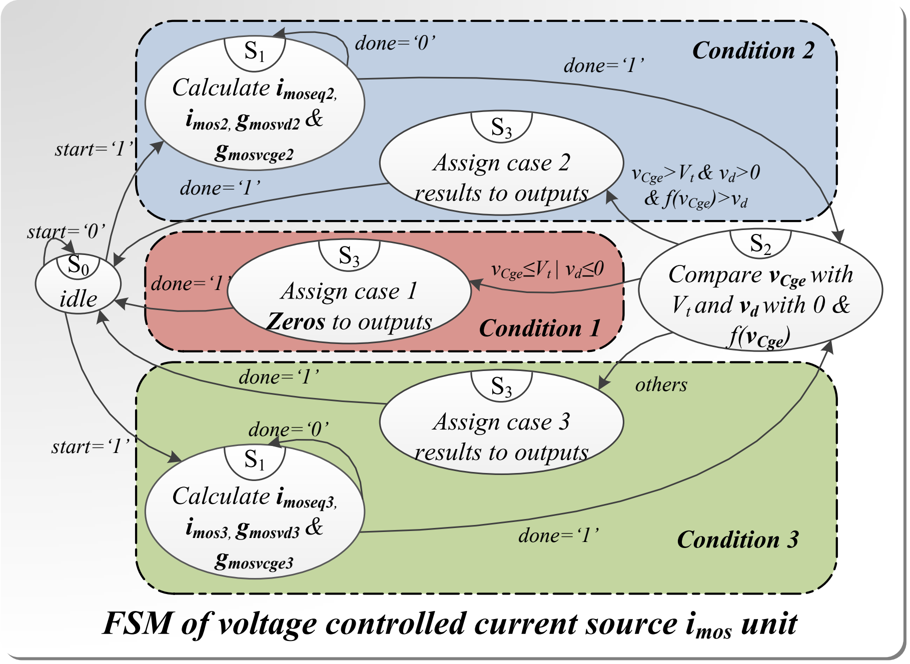

S1
done='0'
Condition 2
done='1'
Calculate i_moseq2, i_mos2, g_mosvd2 & g_mosvcge2
S3
Assign case 2 results to outputs
start='1'
done='1'
v_Cge>V_t & v_d>0 & f(v_Cge)>v_d
start='0'
S0
idle
done='1'
S3
Assign case 1 Zeros to outputs
v_Cge≤V_t | v_d≤0
Condition 1
S2
Compare v_Cge with V_t and v_d with 0 & f(v_Cge)
done='1'
others
S3
Assign case 3 results to outputs
start='1'
done='0'
S1
Calculate i_moseq3, i_mos3, g_mosvd3 & g_mosvcge3
done='1'
Condition 3
FSM of voltage controlled current source i_mos unit

Figure 3.17: Finite state machine of the i_{mos} Unit.

need to be calculated before being selected as the final outputs. The value of I_{moseq} can be determined based on the results of i_{mos}, G_{mosvd}, $G_{mosvcge}$ under different conditions. This i_{mos} hardware unit has the critical path with about 78-clock cycles latency. The finite state machine of this unit is shown in Fig. 3.17. Partially parallel calculation occurs under condition 2 and 3 in S_1 which generates two group of results. The output values are all zeros in condition 1. After the comparison step in S_2, the required set of results is chosen to be sent to outputs in S_3.

The hardware structure of the other internal voltage controlled current source i_{tail} unit is a relatively smaller compared to the previous i_{mos} unit. It receives the outputs signals from i_{mos} unit and calculates i_{tail}, i_{taileq} and its four conductances $G_{tailvtail}$ ($\frac{\partial i_{tail}}{\partial v_{tail}}$), G_{tailvd} ($\frac{\partial i_{tail}}{\partial v_d}$) and $G_{tailvcge}$ ($\frac{\partial i_{tail}}{\partial v_{Cge}}$) according to (3.92) — (3.95). Its hardware structure is shown in Fig. 3.18 which contains only one comparator but 5 multiplexers taking the responsibility of selecting correct results between two conditions. Due to the all zeros in condition 1, partially parallel operation only occurs in condition 2 for generating the corresponding results. Based on the signal from the IP core comparator, the group of 5 multiplexers chooses which set results from the two condition should be assigned to the final outputs. The finite state machine are shown in Fig. 3.19 to control the operation of this i_{tail} hardware unit. Although the values in condition 1 are all zeros, the assignment process still need to wait until the computational operation for condition 2 done in S_1 as well as the comparison step in S_2 finished. In general, these two units are to generate their corresponding currents, dynamic conductances, and equivalent current sources in hardware emulation.

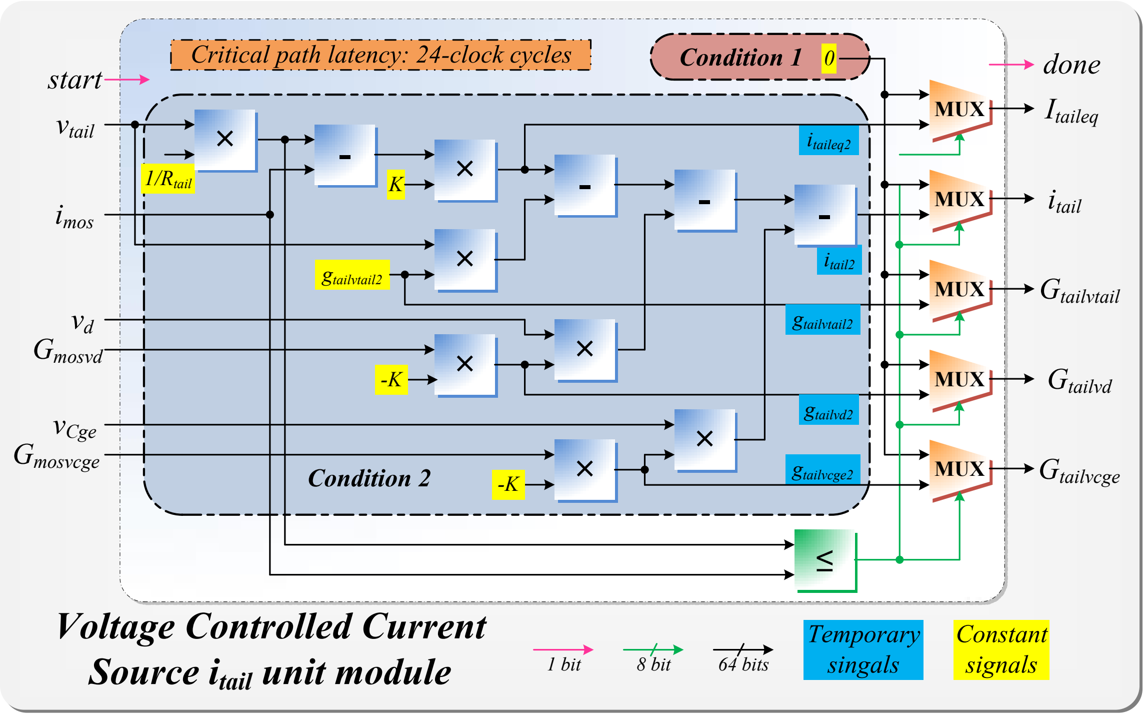


Figure 3.18: Hardware structure of the i_{tail} Unit with latency.

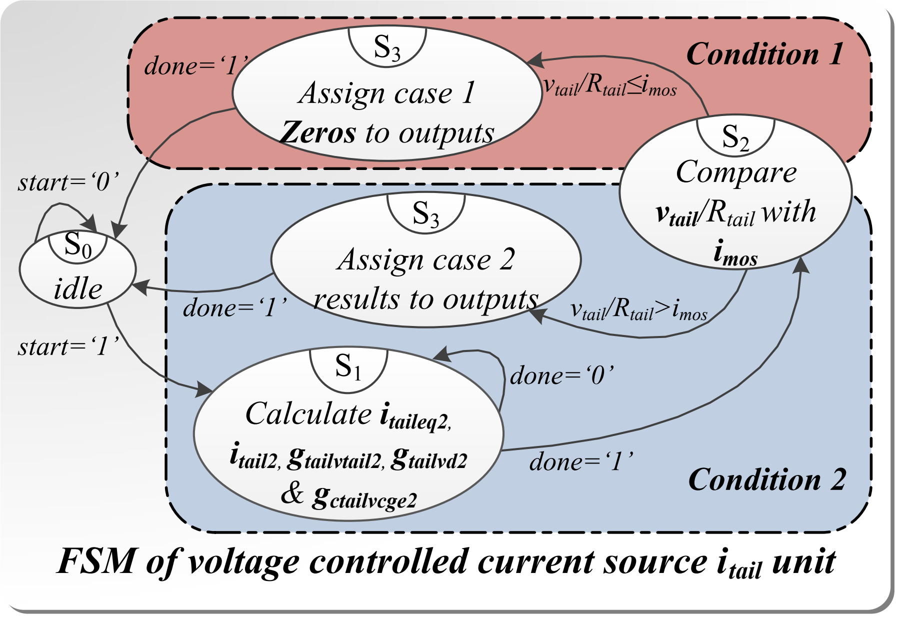


Figure 3.19: Finite state machine of the i_{tail} Unit.

In hardware emulation, this accurate IGBT device model can provide the corresponding accurate results. After each hardware modules of aforementioned sub-units were established and tested, the equivalent circuit in Fig. 3.11(b) could be successfully converted to the IGBT hardware model shown in Fig. 3.20. As illustrated in Fig. 3.20, the IGBT hard-

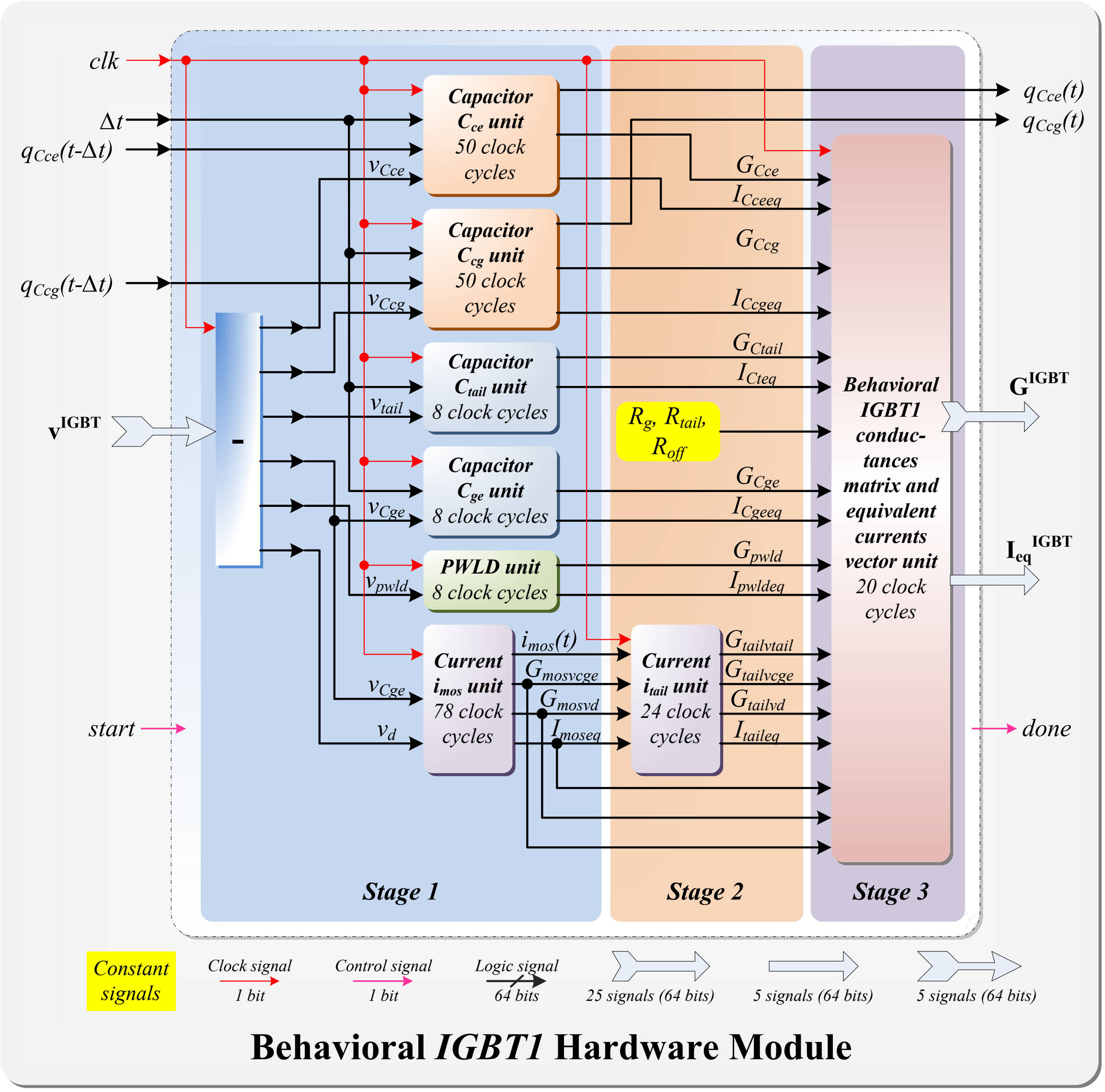

clk
Δt
$q_{Cce}(t\text{-}\Delta t)$
$q_{Ccg}(t\text{-}\Delta t)$
$\mathbf{v}^{\mathbf{IGBT}}$
-
start
Capacitor C_{ce} unit 50 clock cycles
Capacitor C_{cg} unit 50 clock cycles
Capacitor C_{tail} unit 8 clock cycles
Capacitor C_{ge} unit 8 clock cycles
PWLD unit 8 clock cycles
Current i_{mos} unit 78 clock cycles
Current i_{tail} unit 24 clock cycles
v_{Cce}
v_{Ccg}
v_{tail}
v_{Cge}
v_{pwld}
v_{Cge}
v_d
$i_{mos}(t)$
$G_{mosvcge}$
G_{mosvd}
I_{moseq}
G_{Cce}
I_{Cceeq}
G_{Ccg}
I_{Ccgeq}
G_{Ctail}
I_{Cteq}
R_g, R_{tail}, R_{off}
G_{Cge}
I_{Cgeeq}
G_{pwld}
I_{pwldeq}
$G_{tailvtail}$
$G_{tailvcge}$
G_{tailvd}
I_{taileq}
Behavioral IGBT1 conduc-tances matrix and equivalent currents vector unit 20 clock cycles
$q_{Cce}(t)$
$q_{Ccg}(t)$
$\mathbf{G}^{\mathbf{IGBT}}$
$\mathbf{I}_{\mathbf{eq}}^{\mathbf{IGBT}}$
done
Stage 1
Stage 2
Stage 3
Constant signals
Clock signal 1 bit
Control signal 1 bit
Logic signal 64 bits
25 signals (64 bits)
5 signals (64 bits)
5 signals (64 bits)
Behavioral IGBT1 Hardware Module

Figure 3.20: Architecture of the Saber® *IGBT1* hardware module with all sub-units horizontally scaled with respect to latency.

ware module executes in 3 stages. After the required voltages are calculated, the independent 5 sub-units can run in parallel such as the capacitor units (C_{ce}, C_{cg}, C_{tail}, C_{ge}) , PWLD unit and the i_{mos} unit at stage 1. These hardware units can start implementation individually and concurrently. While the correlative units are implemented in a sequential way. After the i_{mos} unit output its correct results, the i_{tail} unit receives those results and executes at stage 2. The unit at stage 3 needs the output signals such as all the calculated dynamic conductances and equivalent currents from the other units in previous stages so as to form the conductance matrix $\mathbf{G}^{IGBT}$ and equivalent current vector $\mathbf{I}_{eq}^{IGBT}$. It has similar functions to those used in the power diode hardware module. Some constant terms such as R_{tail} and several internal parameters are pre-calculated and directly used in the hardware design as constant signals. Meanwhile, the internal calculations using the floating-point IP core operators of the sub-units have deeply pipelined structure. Therefore, computational complexity and latency can be adequately reduced. The circuit node voltages are the input signals to the IGBT hardware module including the collector voltage $v_{Collector}$, internal nodes voltages v_{Inode1} and v_{Inode2}, gate voltage v_{Inode3} and emitter voltage $v_{Emitter}$. The 2 history signals $q_{Cce}(t-\Delta t)$, $q_{Ccg}(t-\Delta t)$ are also the inputs to the hardware module. After N-R iterations were done within the current time interval for circuit simulation, their 2 output counterparts $q_{Cce}(t)$, $q_{Ccg}(t)$ would be correctly calculated which indicates that the necessary history values are successive updated after one time-step Δt.

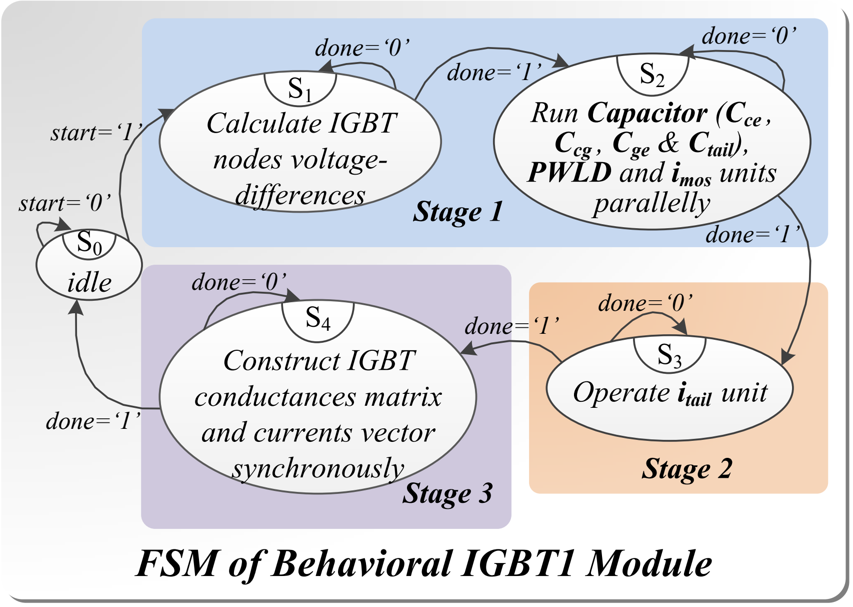

done='0'
S1
Calculate IGBT nodes voltage-differences
Stage 1
done='1'
done='0'
S2
Run Capacitor (Cce, Ccg, Cge & Ctail), PWLD and imos units parallelly
done='1'
start='1'
start='0'
S0
idle
done='0'
S4
Construct IGBT conductances matrix and currents vector synchronously
Stage 3
done='1'
done='0'
S3
Operate itail unit
Stage 2
done='1'
FSM of Behavioral IGBT1 Module

Figure 3.21: Finite state machine of the ® *IGBT1* hardware module.

Fig. 3.21 provides the finite state machine to control the operation sequence of the IGBT hardware module. Similar to the power diode, there are 5 states including the idle state S_0. The IGBT node voltage differences are calculated in S_1 and once the $done$ signal is sent to S_2, all capacitors, PWLD and i_{mos} units will start execution and perform parallel implementation. Then, since the i_{tail} unit is interconnected to i_{mos} subunit, it needs to be triggered at the right time when the i_{mos} subunit is completely done. Lastly, S_4 performs the IGBT conductances matrix and current vector construction and informs $done$ signal to the idle state when the all operation finished. To sum up, the total latency of the 1 time fully calculation within Δt is about 126 clock cycles based on its critical path. This IGBT hardware module fully implements the hardware parallelism feature and it is highly configurable based on actual devices so that it can be widely used in any circuit topologies hardware design.

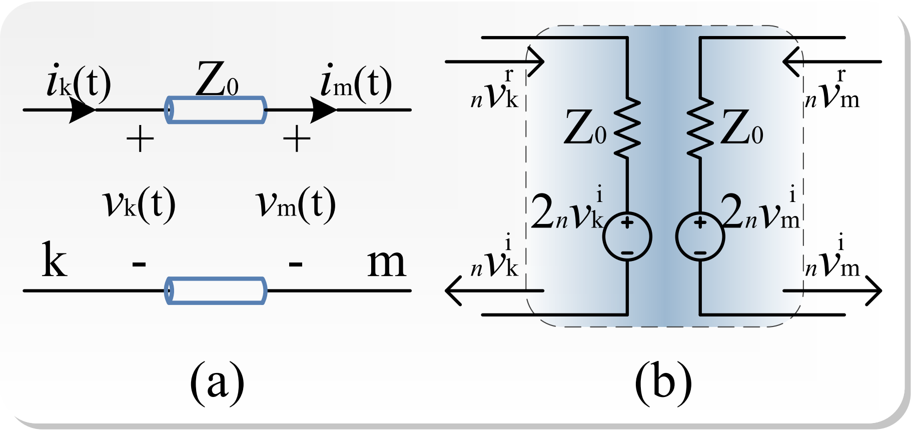


Figure 3.22: (a) Lossless TLM link. (b) TLM link Thévenin equivalent circuit model.

3.4 Transmission Line Model Decoupling Method

Transmission line modeling is increasingly gaining popularity in large-system simulation due to its capability to separate complex nonlinear elements from either linear elements or other nonlinearities whilst maintain high computing precision [113–115]. Although the TLM decoupling technique has several applicable restrictions and drawbacks [47, 95], in system level power converter circuit simulations such as MMC, the TLM method could be applied since more focus is on DC mode rather than the switching dead time effects or the device level transient detail. There are mainly two types of lossless TLM models seen in literature: stub and link. Both of them are widely applied in system simulation, but their usages are different from each other.

3.4.1 Discrete-Time Model of TLM Link

TLM link, a two-port model of a transmission line, has typically been used to decouple a large circuit into several small sub-circuits, reduce the complexity of simulation as well as significantly shorten the computing time so as to acquire the results as fast as possible. In Fig. 3.22(a) $v_k(t)$, $i_k(t)$, $v_m(t)$ and $i_m(t)$ are time-domain voltage and current at terminal k and m, respectively. Z_0 represents line surge impedance, also known as characteristic impedance ($\sqrt{\frac{L}{C}}$). According to transmission line theory, terminal voltages can be divided into two components: the incident pulses which are denoted by superscript i and reflected pulses by superscript r shown in Fig. 3.22(b). For digital simulation, the model can be discretized as:

$$
{}_{n+1}v_k^i = {}_n v_m^r, \tag{3.100}
$$

$$
{}_{n+1}v_m^i = {}_n v_k^r, \tag{3.101}
$$

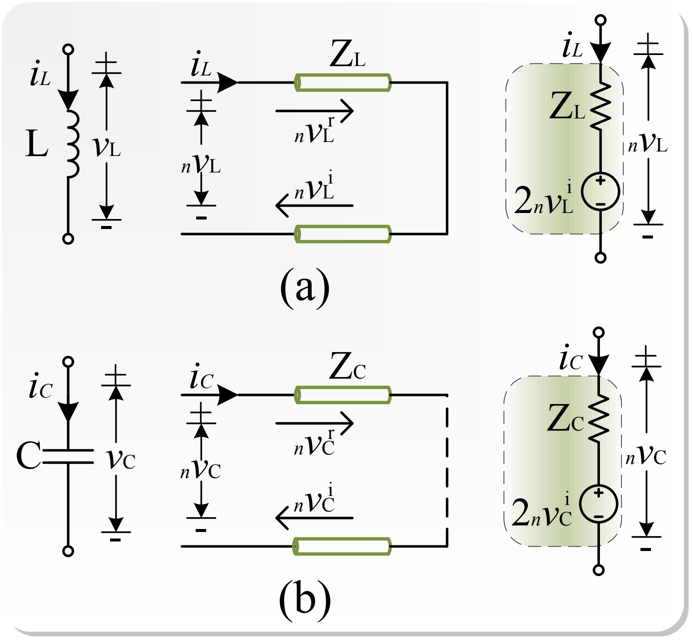


Figure 3.23: (a) Inductor TLM stub model and Thévenin equivalent circuit. (b) Capacitor TLM stub model and Thévenin equivalent circuit.

By rearranging the above equations (3.100) — (3.101) for all the equivalent circuit shown in Fig. 3.22(b), the following can be obtained:

$$
\begin{aligned}
{}_nv_k^r &= {}_nv_k -_n v_k^i, && (3.102)\\
{}_nv_m^r &= {}_nv_m -_n v_m^i, && (3.103)\\
{}_nv_k &= {}_ni_k \cdot Z_0 + 2_nv_k^i, && (3.104)\\
{}_nv_m &= -{}_ni_m \cdot Z_0 + 2_nv_m^i, && (3.105)
\end{aligned}
$$

3.4.2 Discrete-Time Model of TLM Stub

TLM stub is a commonly seen modeling method which could be used to replace linear and more often nonlinear elements in a circuit to solve complex lumped networks [111]. Different from its link counterpart that is modeled into two separate parts, TLM stub could be treated as reactive elements, no matter linear or nonlinear [116]. As stated above, as the surge impedance Z_0 defined as ($\sqrt{L/C}$) depends on the values of L and C, this lossless line can be predominantly inductive or capacitive which means it is possible to replace an inductor or a capacitor by an equivalent TLM model. The former is modeled by a short-circuited line with surge impedance $Z_L = 2L/T$ while the latter is represented by an open-circuited line whose surge impedance is $Z_C = T/2C$ where T is the pulses round-trip time taking from port to the end and back again and is subsequently set as the time-step. Fig. 3.23 shows the TLM models of reactive elements and their Thévenin equivalent circuits.

It can be seen that an associated stray capacitance and an associated parasitic inductance are introduced to the TLM models of inductor and capacitor, respectively, with values $C_s = T^2/(4L)$ and $L_s = T^2/(4C)$. This clearly shows reducing time-step could significantly attenuate associated elements impact on simulation.

For either L or C model, the terminal voltage at any time-step is the sum of incident pulse and reflected pulse.

$$_n v_{L,C} =_n v^i_{L,C} +_n v^r_{L,C}, \tag{3.106}$$

From the Thévenin equivalent circuits, the voltage across the L or C stub model can be obtained by:

$$_n v_{L,C} =_n i_{L,C} \cdot Z_{L,C} + 2_n v^i_{L,C}, \tag{3.107}$$

The reflection coefficient is 1 for open-circuit end, which means the incident pulse reflects without inversion after reaching the end. Then, the reflected pulse will become the incident pulse for the next iteration.

$$_{n+1} v^i_C = \ _n v^r_C, \tag{3.108}$$

While, with regard to short-circuit end stub, the reflection coefficient is -1, so the incident pulse is inverted when it touches down the end and returns to the port so as to update the incident pulse for the next time-step.

$$_{n+1} v^i_L = - \ _n v^r_L, \tag{3.109}$$

3.5 Matrix Solver

As the linear equation systems introduced by placing the behavioral models in the circuit, the node voltages need to be solved. This section will illustrate several linear solvers.

3.5.1 Cramer's Rule Fast Linear Solver

In linear algebra, Cramer's rule is an explicit formula for the solution of a system of linear equations with as many equations as unknowns, valid whenever the system has a unique solution. It expresses the solution in terms of the determinants of the (square) coefficient matrix and of matrices obtained from it by replacing one column by the vector of right-hand sides of the equations. Generally, a system with n linear equations for n unknowns can be represented in matrix multiplication form as:

$$\mathbf{A} \cdot \mathbf{X} = \mathbf{B}, \tag{3.110}$$

where the $n \times n$ matrix $\mathbf{A}$ has a nonzero determinant, the vector $\mathbf{X} = (x_1, \ldots, x_n)^{\mathrm{T}}$ is the column vector of the variables, and $\mathbf{B}$ is a $n \times 1$ vector $\mathbf{B} = (b_1, \ldots, b_n)^{\mathrm{T}}$. Then the system has a unique solution which can be found by:

$$x_i = \frac{\det(\mathbf{A}_i)}{\det(\mathbf{A})} \qquad (i = 1, 2, \cdots, n), \tag{3.111}$$

where $\mathbf{A}_i$ is the matrix formed by replacing the i^{th} column of $\mathbf{A}$ by the column vector $\mathbf{B}$. When the matrix is 2-dimensional or 3-dimensional, it is simple to solve the equations using Cramer's rule. Firstly, consider the system has 2 linear equations for 2 unknowns as follows:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 = b_1 \\ a_{21}x_1 + a_{22}x_2 = b_2 \end{cases}, \tag{3.112}$$

Then, x_1 and x_2 can be found as

$$\begin{cases} x_1 = \dfrac{\begin{vmatrix} b_1 & a_{12} \\ b_2 & a_{22} \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}} = \frac{b_1a_{22}-a_{12}b_2}{a_{11}a_{22}-a_{12}a_{21}} \\ x_2 = \dfrac{\begin{vmatrix} a_{11} & b_1 \\ a_{21} & b_2 \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}} = \frac{a_{11}b_2-b_1a_{21}}{a_{11}a_{22}-a_{12}a_{21}} \end{cases}, \tag{3.113}$$

According to (3.113), the denominators are the same for both x_1 and x_2, and the numerators have the same format with two multiply and one subtract operations. In hardware emulation, as shown in Fig. 3.24, the equations can be solved in parallel by applying the multiply operation firstly, then the subtraction operation and lastly the division operation. The total latency would be 22 clocks cycles.

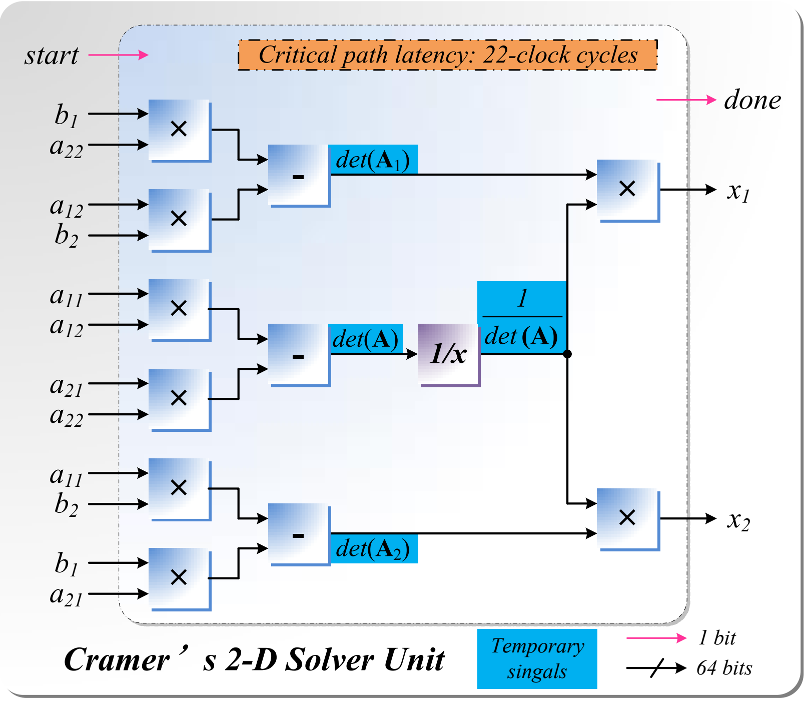


Figure 3.24: Hardware structure of 2-dimensional Parallel Cramer's Fast Linear Solver.

The 3 by 3 matrices (3 linear equations with 3 unknowns) will have similar format using Cramer's rule:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3 \end{cases}, \tag{3.114}$$

Then, the solutions of x_1, x_2 and x_3 can be found as:

$$\begin{cases} x_1 = \dfrac{\begin{vmatrix} b_1 & a_{12} & a_{13} \\ b_2 & a_{22} & a_{23} \\ b_3 & a_{32} & a_{33} \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}} = \frac{b_1a_{22}a_{33}+a_{12}a_{23}b_3+a_{13}b_2a_{32}-b_1a_{23}a_{32}-a_{12}b_2a_{33}-a_{13}a_{22}b_3}{a_{11}a_{22}a_{33}+a_{12}a_{23}a_{31}+a_{13}a_{21}a_{32}-a_{11}a_{23}a_{32}-a_{12}a_{21}a_{33}-a_{13}a_{22}a_{31}} \\ x_2 = \dfrac{\begin{vmatrix} a_{11} & b_1 & a_{13} \\ a_{21} & b_2 & a_{23} \\ a_{31} & b_3 & a_{33} \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}} = \frac{a_{11}b_2a_{33}+b_1a_{23}a_{31}+a_{13}a_{21}b_3-a_{11}a_{23}b_3-b_1a_{21}a_{33}-a_{13}b_2a_{31}}{a_{11}a_{22}a_{33}+a_{12}a_{23}a_{31}+a_{13}a_{21}a_{32}-a_{11}a_{23}a_{32}-a_{12}a_{21}a_{33}-a_{13}a_{22}a_{31}} \\ x_3 = \dfrac{\begin{vmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ a_{31} & a_{32} & b_3 \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}} = \frac{a_{11}a_{22}b_3+a_{12}b_2a_{31}+b_1a_{21}a_{32}-a_{11}b_2a_{32}-a_{12}a_{21}b_3-b_1a_{22}a_{31}}{a_{11}a_{22}a_{33}+a_{12}a_{23}a_{31}+a_{13}a_{21}a_{32}-a_{11}a_{23}a_{32}-a_{12}a_{21}a_{33}-a_{13}a_{22}a_{31}} \end{cases}. \tag{3.115}$$

According to (3.115), the denominators are the same for x_1, x_2, and x_3, and the numerators have the same format as the denominator. In hardware emulation, as shown in Fig. 3.25, the equations can be solved in parallel by applying the detailed sub-unit operation and then the division operation. The total latency would be 34 clocks cycles.

3.5.2 Parallel Gauss Elimination with Partial Pivoting Linear Solver

Cramer's rule can solve the linear equations as finding numbers of $n + 1$ determinants for the system has n linear equations for n unknowns. For example, a 2-dimensional matrix requires two terms polynomial (each term contains 2 elements) in both numerator and denominator for Cramer's rule; a 3-dimensional matrix requires six terms (factorial of 3) polynomial (each term contains 3 elements multiplied together); and by parity of reasoning, a 4-dimensional matrix requires polynomials of 24 terms (factorial of 4) where each term contains 4 elements multiplied together.Thus, it needs lots of computations and calculations when n becomes large. Meanwhile, in hardware emulation, it will consume

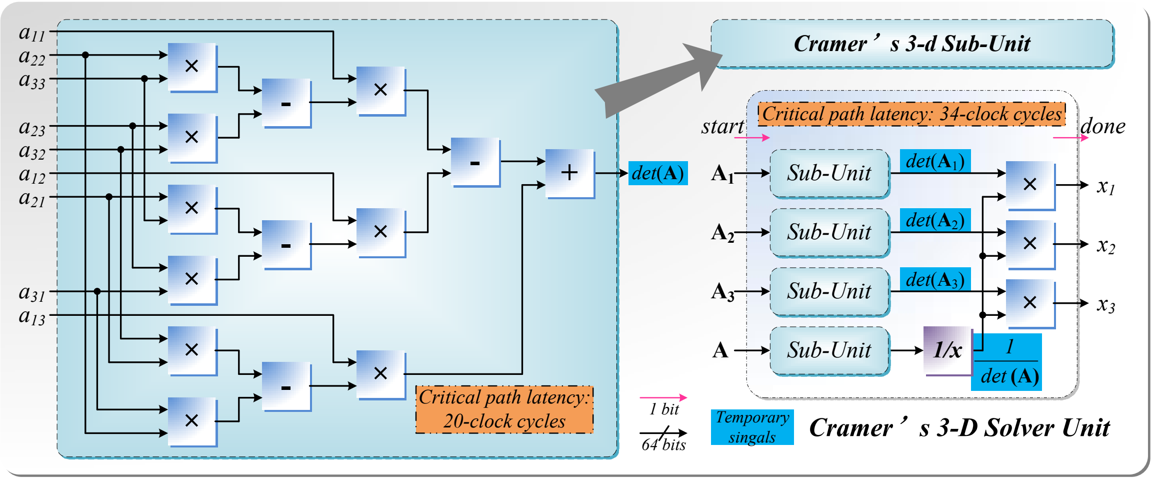


Figure 3.25: Structure of 3-dimensional parallel Cramer's solver.

many resources. Therefore, when the order of a linear system of equations is large or equal to 4, the Gauss Elimination should be introduced.

In power converter simulation, the linear system of equations needs to be solved to obtain the node voltages. This section is provided a parallel Gauss elimination with partial pivoting linear solver module to deal with the problems. There are three basic steps of conventional Gaussian elimination as pivoting operation, forward elimination, and backward substitution. In this thesis, the solver module using reduced row echelon form has only two main steps: pivoting and elimination. In hardware emulation, in order to reduce the latency and improve the performance, elimination process is deeply paralleled at the cost of using more hardware resources. The pivoting step is to ensure the computer calculated results accuracy by simply reassign the appropriate row vectors after the comparison. A similar hardware implementation can be found in [63]. For example, a 4-dimensional linear system of equations is formulated as

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 = a_{15} \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 = a_{25} \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 = a_{35} \\ a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 = a_{45} \end{cases}, \tag{3.116}$$

where a_{ij} $(i = 1, \cdots 4; j = 1, \cdots 5)$ are the elements usually known as constants, and x_i $(i = 1, \cdots 4)$ are the variables needed to be solved. After pairwise comparison of a_{11}, a_{21}, a_{31} and a_{41} using six IP core 64-bit floating point comparators in parallel, partial pivoting should be performed before the elimination process. The comparators outputs are combined as a vector to control the pivoting operation. This control vector can be represented as:

$$\mathbf{c} = [c_{12}, c_{13}, c_{14}, c_{23}, c_{24}, c_{34}], \tag{3.117}$$

where c_{12} indicates the result based on comparison between the first elements in row 1 and row 2 and so on for the rest c. c could be either 1 or 0. For example, c_{12} is 1 if $a_{11} \geq a_{21}$ and 0 if $a_{11} < a_{21}$. Then the pivoting operates in the way like if $\mathbf{c} = [0, 1, 1, 1, 1, 0]$, swapping the second row with the first row. If a_{11} has the largest absolute value comparing to a_{21}, a_{31} and a_{41}, without swapping rows after the pivoting process, the argumented matrix shown below

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \end{bmatrix}, \tag{3.118}$$

will be performed the elimination and transform to

$$\begin{bmatrix} a_{22} - \frac{a_{21}a_{12}}{a_{11}} & a_{23} - \frac{a_{21}a_{13}}{a_{11}} & a_{24} - \frac{a_{21}a_{14}}{a_{11}} & a_{25} - \frac{a_{21}a_{15}}{a_{11}} & 0 \\ a_{32} - \frac{a_{31}a_{12}}{a_{11}} & a_{33} - \frac{a_{31}a_{13}}{a_{11}} & a_{34} - \frac{a_{31}a_{14}}{a_{11}} & a_{35} - \frac{a_{31}a_{15}}{a_{11}} & 0 \\ a_{42} - \frac{a_{41}a_{12}}{a_{11}} & a_{43} - \frac{a_{41}a_{13}}{a_{11}} & a_{44} - \frac{a_{41}a_{14}}{a_{11}} & a_{45} - \frac{a_{41}a_{15}}{a_{11}} & 0 \\ a_{12} & a_{13} & a_{14} & a_{15} & a_{11} \end{bmatrix}, \tag{3.119}$$

From the above, all the elements will be shifted left and up for once during each cycle and the right end elements are supplemented by 0. Then elements within the new formed argumented matrix (3.119) are temporarily stored in registers waiting for the next cycle of pivoting and elimination. After each cycle, the number of comparable rows for pivoting operation will be reduced by 1. For example, after the first cycle of pivoting and elimination, the comparison only needs to perform on the first elements of the first three rows. A 4-dimensional matrix needs 4 times operation to generate the solutions and that costs about 100 clock cycle as the module entire latency based on the double precision floating point calculation.

This solver module can be applied to any dimensional linear system of equations. The entire latency of this solver is in proportion to the matrix dimensions. While the latency within each cycle keeps consistent (around 25 clock cycles) regardless of matrix dimensions. Consequently, the larger the matrix dimension, the higher the efficiency of solving the equations and the more the advantages of this module.

3.5.3 Parallel Gauss-Jordan Elimination Linear Solver

Gaussian elimination and Gauss-Jordan elimination are both used to solve systems of linear equations, as well as finding inverses of non-singular matrices based on elementary row operations. Gaussian elimination helps to put an augmented matrix in row echelon form (REF), while Gauss-Jordan elimination puts a matrix in reduced row echelon form (RREF). As to the form, the main difference is that Gaussian elimination brings the augmented matrix into the lower triangular form on the left side, but Gauss-Jordan reduces the augmented matrix to an identity matrix. Thus, assuming that the system is consistent, the solution set can be obtained by backward substitution in the case of Gaussian elimination, whereas, it can be gained directly from the final matrix in case of Gauss-Jordan elimination. Although Gaussian elimination is occasionally computationally more efficient for computers, Gauss-Jordan elimination is usually more convenient to explicitly solve for each variable represented in the small matrix system. When the elements inside the matrix are constant and the matrix is nearly a sparse matrix, such as in the case as follows, it is easy and efficient to implement Gauss-Jordan Elimination. To solve the linear system, the procedure is to do the same elementary row operations without swapping the positions of rows for the right-hand side of the equation array (such as **U** in the following example).

$$\begin{bmatrix} a & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b & 0 \\ 0 & a & 0 & 0 & 0 & 0 & 0 & 0 & b & 0 \\ 0 & 0 & a & 0 & 0 & 0 & 0 & 0 & b & 0 \\ 0 & 0 & 0 & a & 0 & 0 & 0 & 0 & b & 0 \\ 0 & 0 & 0 & 0 & a & 0 & 0 & 0 & 0 & b \\ 0 & 0 & 0 & 0 & 0 & a & 0 & 0 & 0 & b \\ 0 & 0 & 0 & 0 & 0 & 0 & a & 0 & 0 & b \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & a & 0 & b \\ b & b & b & b & 0 & 0 & 0 & 0 & c & d \\ 0 & 0 & 0 & 0 & b & b & b & b & d & c \end{bmatrix} \times [\mathbf{I}] = [\mathbf{U}], \tag{3.120}$$

where by taking MMC with TLM case studies in the next Chapter 4:

$$a = 2Z_S + Z_C, \tag{3.121}$$

$$b = -Z_S, \tag{3.122}$$

$$c = 4Z_S + Z_{Lu} + Z_L + R, \tag{3.123}$$

$$d = -R - Z_L. \tag{3.124}$$

According to (3.120), the augmented matrix can be found as follows:

$$\left[\begin{array}{cccccccccc|c} a & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b & 0 & U_1 \\ 0 & a & 0 & 0 & 0 & 0 & 0 & 0 & b & 0 & U_2 \\ 0 & 0 & a & 0 & 0 & 0 & 0 & 0 & b & 0 & U_3 \\ 0 & 0 & 0 & a & 0 & 0 & 0 & 0 & b & 0 & U_4 \\ 0 & 0 & 0 & 0 & a & 0 & 0 & 0 & 0 & b & U_5 \\ 0 & 0 & 0 & 0 & 0 & a & 0 & 0 & 0 & b & U_6 \\ 0 & 0 & 0 & 0 & 0 & 0 & a & 0 & 0 & b & U_7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & a & 0 & b & U_8 \\ b & b & b & b & 0 & 0 & 0 & 0 & c & d & U_9 \\ 0 & 0 & 0 & 0 & b & b & b & b & d & c & U_{10} \end{array}\right]. \tag{3.125}$$

The purpose is to perform row operations to the above system until a reduced row echelon form obtained. Then the augmented matrix becomes an identity matrix on the left, and the solution set can be obtained directly from the final matrix on the right shown below:

$$\left[\begin{array}{cccccccccc|c} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & U_1^{'} \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & U_2^{'} \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & U_3^{'} \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & U_4^{'} \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & U_5^{'} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & U_6^{'} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & U_7^{'} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & U_8^{'} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & U_9^{'} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & U_{10}^{'} \end{array}\right] \tag{3.126}$$

Therefore, the solution set of I will be the corresponding values from set $U^{'}$. In hardware design, some row operations can be implemented in parallel and the procedure is only dealing with U.

3.6 Newton-Raphson Iterations

Newton-Raphson (N-R) iterations usually called Newton's method in numerical analysis is adopted in power converter simulation for finding successively better approximations. An n-dimensional nonlinear power converter circuit can be represented as:

$$\mathbf{i} = f(\mathbf{v}), \tag{3.127}$$

where $\mathbf{v} = (v_1, v_2, \ldots, v_n)^T$, $\mathbf{i} = (i_1, i_2, \ldots, i_n)^T$ are node voltages and current vectors respectively, and f indicates the general nonlinear operator that the values of $\mathbf{i}$ are the function with respect to the values of $\mathbf{v}$. Basically, in the thesis, the node voltage vector at $(k+1)^{th}$ iteration are calculated using N-R method as:

$$\mathbf{v}_{k+1} = \mathbf{G}^{-1}(\mathbf{v}_k) \cdot \mathbf{I}_{eq}(\mathbf{v}_k), \tag{3.128}$$

where $\mathbf{I}_{eq}(\mathbf{v}_k)$ is equivalent current sources vector determined by:

$$\mathbf{I}_{eq}(\mathbf{v}_k) = \mathbf{i}(\mathbf{v}_k) - \mathbf{G}(\mathbf{v}_k) \cdot \mathbf{v}_k, \tag{3.129}$$

and $\mathbf{G}(\mathbf{v}_k)$ is the general linearized conductance matrix found from the partial derivative formulated as:

$$\mathbf{G}(\mathbf{v}_k) = \begin{pmatrix} \frac{\partial i_1}{\partial v_1}|_{\mathbf{v}_k} & \frac{\partial i_1}{\partial v_2}|_{\mathbf{v}_k} & \cdots & \frac{\partial i_1}{\partial v_n}|_{\mathbf{v}_k} \\ \vdots & & & \vdots \\ \frac{\partial i_n}{\partial v_1}|_{\mathbf{v}_k} & \frac{\partial i_n}{\partial v_2}|_{\mathbf{v}_k} & \cdots & \frac{\partial i_n}{\partial v_n}|_{\mathbf{v}_k} \end{pmatrix}. \tag{3.130}$$

Thus, the approximations of $\mathbf{v}$ can be treated as relative accurate results when each value within the voltage vector set meets the iteration convergence criteria given by:

$$|\frac{v_{i(k+1)} - v_{i(k)}}{v_{i(k)}}| \leq \epsilon \qquad (i = 1, 2, \cdots, n), \tag{3.131}$$

where ϵ is the error tolerance chosen to be 10^{-3} in the power converter simulation.

3.7 Different Time-Step Schemes

Several time-step schemes can be adopted for the power converter simulation such as fixed time-step solution and variable time-step solution.

3.7.1 Fixed Time-Step Solution

Hardware emulation of power system normally applies fixed time-step strategy which is relatively simple to implement and compatible to the oscilloscope. At system level simulation, it is usually enough to set a relatively large value for fixed time-step. When simulating the power converter such as diode and IGBT, since a relatively large fixed time-step would make the results hard to converge and omit transient characteristics which usually range in the nanosecond, a small time-step should be used so as to observe and analysis the transient behavior or evaluate switching power loss. Consequently, small time-step can get the accurate results, but it would make the hardware simulations very time-consuming and difficult to reach the real-time level for power electronic devices as the latest and fastest FPGA only have hundreds MHz level clock frequency. Therefore, completing the massive computation of the power converter for a single time-step within a small step size at relatively low FPGA clock frequency is a challenge for the real-time simulation. Anyhow, the small fixed time-step can be applied to the TLM decoupled power converter rather than the variable time-step due to the fact that the parameters of TLM have to be modified every time with respect to the changes of time-step which introduces more calculation steps and control procedures.

3.7.2 Variable Time-Step Solution

The variable time-step algorithm is used in such software simulators as SaberRD® and PSpice®. It can speed up the simulation time while ensuring the accurate simulation results [109]. As mentioned in [108], predictor and corrector schemes were provided for the numerical solution used as estimates of the single step truncation error. In hardware emulation, accuracy and speed are also important. In paper [59], the VTCM was stated which was adjusting the step size Δt based on the times of Newton-Raphson iterations similar in Spice® [110]. Instead of counting the N-R times, this section introduces the analogous or approximate local truncation error (LTE) variable time-step control method so as to reduce the simulation and execution time. The time-step Δt could be large during steady-state operation; while it should be small during transient so that a high-accurate analysis could be achieved. This dynamic algorithm calculates an internal time-step based on the local truncation error. The local truncation error time-step algorithm is setting the new step size which could be found by multiplying or dividing the previous step size by 2. Within the

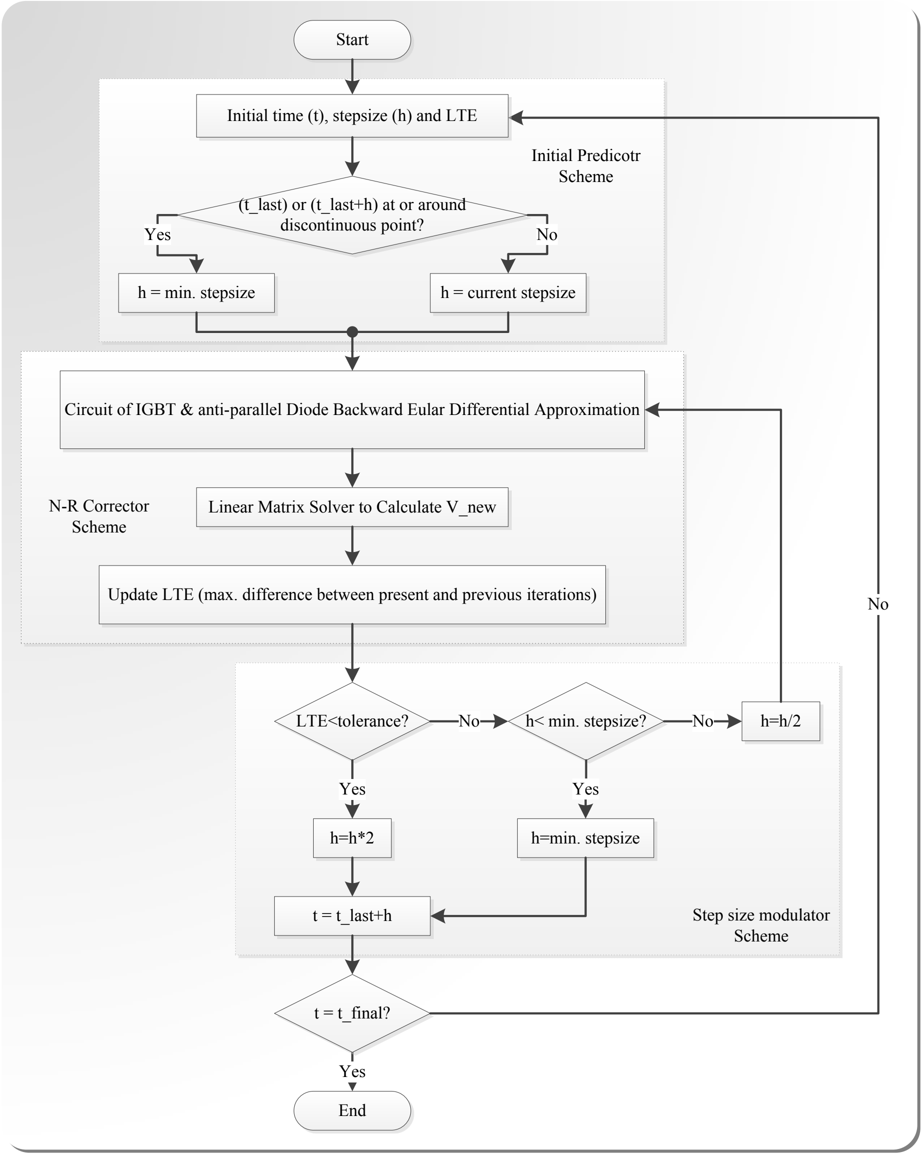

Start
Initial time (t), stepsize (h) and LTE
Initial Predicotr
Scheme
(t_last) or (t_last+h) at or around
discontinuous point?
Yes
No
h = min. stepsize
h = current stepsize
Circuit of IGBT & anti-parallel Diode Backward Eular Differential Approximation
N-R Corrector
Scheme
Linear Matrix Solver to Calculate V_new
Update LTE (max. difference between present and previous iterations)
No
LTE<tolerance?
No
h< min. stepsize?
No
h=h/2
Yes
Yes
h=h*2
h=min. stepsize
t = t_last+h
Step size modulator
Scheme
t = t_final?
Yes
End

Figure 3.26: LTE controlled variable time-step algorithm .

current time interval, it adjusts the time-step Δt based on the local truncation errors between two the N-R iterations of the current time interval and the calculated results from last time interval. If the calculated $LTE \leq tolerance$, the time-step increased by 2 for the next time interval, and when $LTE > tolerance$, the time-step is decreased by 2, the time point is set back for recalculation in the current time interval (time reversal). In this case, the largest and smallest values of time-step are necessarily set for limitation. If the calculated time-step exceeds the limitation, the step size will be set to the corresponding limited values and use for the next interval. Moreover, the time-step, during transient state or when a transient occurs such as a pulse or a fault, will be set to smallest or minimum value all the time so as to ensure the calculation accuracy. The detailed LTE control algorithm for variable time-step are shown in Fig. 3.26 as a flow chart. As using the variable time-step, the usage of hardware resources would be reduced and the energy is saved. For example, from t_1 to t_2, there are 10-time intervals for the fixed time-step; while the variable time-step scheme might only have 3 intervals. That means the whole module could only run 3 times based on the variable time-step rather than 10 times under fixed Δt.

3.7.3 Output Time Control Solution

Using the fixed time-step algorithm in hardware emulation, if without N-R, the output values can be directly sent out to form the resulting waveforms as the time-step and intervals are always the same. However, when the variable time-step is introduced, the time interval of the output needs to be considered in hardware emulation. The required output signals calculated from the converged node voltages should be sent out at a fixed sample rate to guarantee that the waveforms are in a correct shape which is also mentioned in [59]. Even though the fixed time-step is used but there are N-R iterations included, the output intervals still need to be adjusted since different iteration times. Thus, the output control should be performed so that the uneven calculation data from the different time-step sizes can be output at equal space of time. For fixed time solution, in order to ensure that the required output signals can be sent out at a fixed sample rate, the iteration times should be set as fixed to each time intervals or choose the most iterations as the output intervals. For variable time solution, as mentioned in [59], the output signals can be controlled and spaced evenly based on an integer value obtained by multiplying a coefficient $\frac{1}{\Delta t_{min}}$ and the current time-step Δt indicating the number of duplicated output results stored in FIFOs for the current time interval.

3.8 Power Converter Hardware Emulation

In industrial design, an entire switch of a power converter is alway an IGBT and an anti-parallel diode combined pack model. The IGBT MOSFET which is an unidirectional switch can only carry positive current (n channel MOSFET, from drain to source). As most loads are inductive (for example, feedback rectifiers and induction motors), if they are connected to the converter, there must be some current going through the opposite direction of the IGBT. Since inductive current will not be ceased or interrupted but will generate high voltage peaks when the switch is turned off, the voltage over the inductive load needs to be reversed. Then, the anti-parallel diode should be used in a power circuit to reverse this voltage and give the inductive load current a path to flow so as to protect the converter operating normally. Based on the device level converter, the anti-parallel diode is usually a fast reverse recovery diode or a freewheeling diode. Therefore, the behavioral reverse recovery diode module is connected across the IGBT module anti-parallelly by linking the cathode of the diode to the collector of IGBT and the anode to the emitter.

3.8.1 Diode Model Simplification

As the complexity of the complete power diode represented previously, a simplified diode based on that complete model should be introduced. Firstly, from (3.65) to (3.67), the v_j diode part which is containing I_s and exponential term in static model could be reduced by replacing a piecewise linear diode which would be given written as:

$$G_j = \begin{cases} g_{on} & (v_j > V_{onpwld}) \\ g_{off} & (v_j \leq V_{onpwld}) \end{cases}, \tag{3.132}$$

$$i_j = i_{pwld} \approx G_j \cdot (v_j - V_{onpwld}), \tag{3.133}$$

$$I_{jeq} = -G_j \cdot V_{onpwld}. \tag{3.134}$$

Then, the effect of the R_{on} resistor could be merged into the PWLD and the exponential term and could be eliminated. Thus, the entire model in Fig. 3.8 would be simplified as shown in Fig. 3.27(a). Moreover, when power diode is connected anti-parallel with IGBT, it is usually a fast reverse recovery diode or a freewheeling diode, the voltage forward recovery and the junction capacitance units could be neglect. Therefore, the number of nodes in the model is reduced from 5 to 3 as shown in Fig. 3.27(b) that makes less computation on hardware emulation of the IGBT and anti-parallel diode cell. While, based on

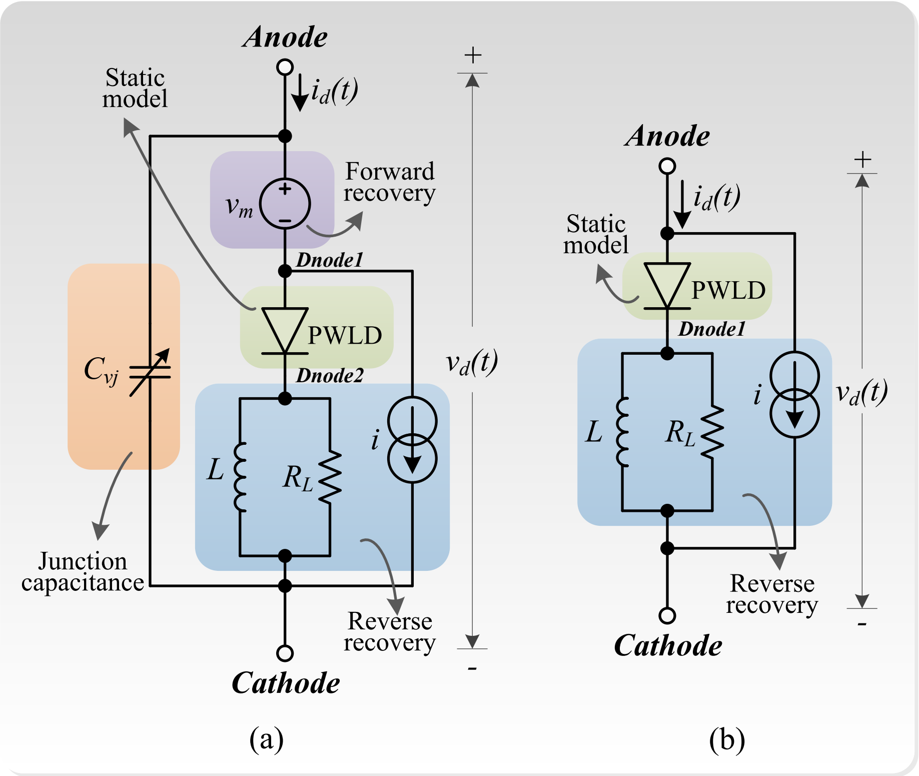


Figure 3.27: (a) Simplified Nonlinear Behavioral Power Diode Complete Model. (b) Simplified Reverse Recovery Power Diode Model.

the Fig. 3.11, the parallel resistance r_{off} could be omitted and the rest of behavioral *IGBT1* model can keep the same structure since its all features and characteristics are necessary and dominate in the entire switch model.

3.8.2 Decomposition of Simplified Behavioral IGBT1 and Anti-parallel Diode Pair with Introduced Transmission-Line Module - SPTLM

Without the TLM decoupling, the entire simplified IGBT, and anti-parallel diode pair (with only reverse recovery) can form a six-node discrete-time linear equation system with conductance matrix shown in (3.135), node voltage vector shown in (3.136) and equivalent current vector shown in (3.137).

$$\mathbf{G}^{pair} = \begin{bmatrix} \begin{array}{c} K+G_{RL} \\ +G_L+G_{pwld} \end{array} & G_{pwld} & 0 & 0 & \begin{array}{c} K \quad G_{RL} \\ G_L \end{array} & 0 \\ G_{pwld} & \begin{array}{c} G_{Cce}+G_{Ccg} \\ +G_{pwld}+G_{mosvd} \\ +G_{tailvd} \end{array} & \begin{array}{c} G_{tailvtail} \quad G_{mosvd} \\ G_{tailvd} \end{array} & \begin{array}{c} G_{mosvcge}+G_{tailvcge} \\ G_{Ccg} \end{array} & 0 & \begin{array}{c} G_{mosvcge} \quad G_{tailvcge} \\ G_{tailvtail} \quad G_{Cce} \end{array} \\ 0 & G_{mosvd} & \begin{array}{c} G_{mosvd}+G_{tail} \\ +G_{Ctail} \end{array} & G_{mosvcge} & 0 & \begin{array}{c} G_{mosvcge} \quad G_{tail} \\ G_{Ctail} \end{array} \\ 0 & G_{Ccg} & 0 & \begin{array}{c} G_{Cge}+G_{Ccg} \\ +\frac{1}{R_g+R_{gs}} \end{array} & 0 & G_{Cge} \quad \frac{1}{R_g+R_{gs}} \\ G_{RL} \quad G_L & 0 & 0 & 0 & \begin{array}{c} G_j+G_{RL} \\ +G_L \end{array} & G_j \\ K & G_{Cce} \quad G_{tailvd} & \begin{array}{c} G_{tailvd} \quad G_{tailvtail} \\ G_{tail} \quad G_{Ctail} \end{array} & \begin{array}{c} G_{Cge} \quad G_{tailvcge} \\ \frac{1}{R_g+R_{gs}} \end{array} & K \quad G_j & \begin{array}{c} G_j+G_{Cge}+G_{Cce} \\ +G_{tailvtail}+G_{tailvcge} \\ +G_{Ctail}+G_{tail} \\ +\frac{1}{R_g+R_{gs}} \end{array} \end{bmatrix}, \tag{3.135}$$

$$\mathbf{v}^{pair} = \begin{bmatrix} v_{Collector}, & v_{Inode1}, & v_{Inode2}, & v_{Inode3}, & v_{Dnode1}, & v_{Emitter} \end{bmatrix}^T, \tag{3.136}$$

$$\mathbf{I}_{eq}^{pair} = \begin{bmatrix} I_{Leq} & I_{pwldeq} & \begin{array}{c} I_{pwldeq} \quad I_{moseq} \\ I_{Cceeq} \quad I_{Ccgeq} \\ I_{taileq} \end{array} & I_{moseq} & I_{Ctaileq} & \begin{array}{c} I_{Ccgeq} \quad I_{Cgeeq} \\ +\frac{V_s}{R_g+R_{gs}} \end{array} & I_{jeq} & I_{Leq} & \begin{array}{c} I_{taileq}+I_{Ctaileq} \\ +I_{Cgeeq}+I_{Cceeq} \\ \frac{V_s}{R_g+R_{gs}} \quad I_{jeq} \end{array} \end{bmatrix}^T, \tag{3.137}$$

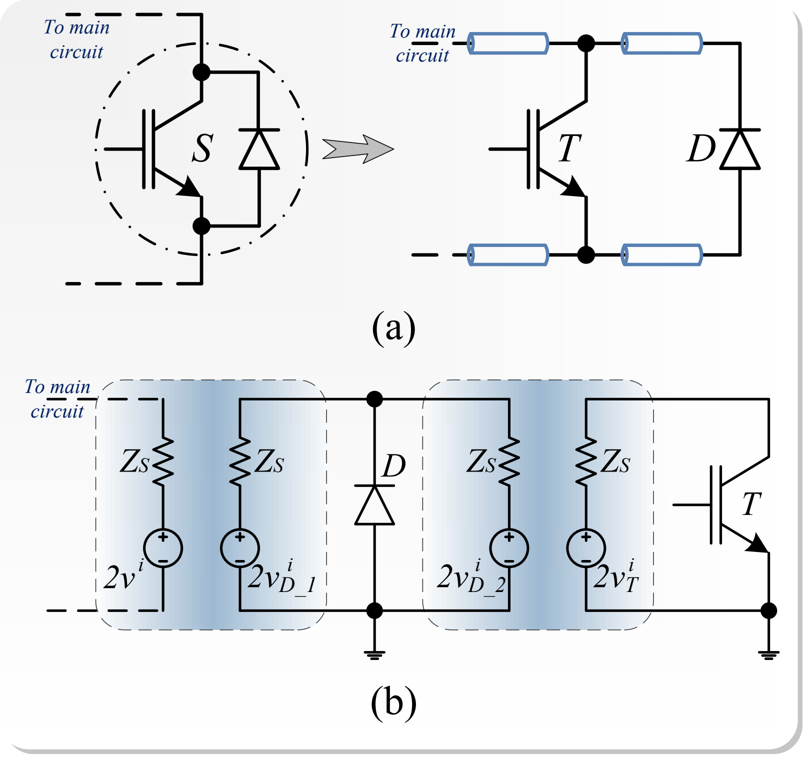


Figure 3.28: (a) SPTLM formation. (b) SPTLM link discretized equivalent circuit.

Using aforementioned TLM decoupling method, in system level simulation for MMC, IGBT and diode could be isolated from each other by inserting a TLM links between them as shown in Fig. 3.28 so as to perform the parallel computation and reduce the system matrix size. Due to the decomposition, each diode and IGBT combined with transmission line model forms a circuit individually and can perform the calculation separately and in parallel. Meanwhile, each of the decoupled circuits has one node connected to ground (anode of diode and emitter of IGBT). Thus, both of their matrices sizes would be deducted by 1 as the constant 0 voltage of related nodes. Consequently, the IGBT has 4 unknown internal nodes which need to be calculated based on its decoupled circuit while the diode has only 2 nodes. The simplified diode model with only reverse recovery (footnoted by sr) can be written in an updated form where the 2×2 conductance matrix is given by:

$$\mathbf{G}_{sr}^{Diode} = \begin{bmatrix} K + G_{RL} + G_L & -K - G_{RL} - G_L \\ -G_{RL} - G_L & G_j + G_{RL} + G_L \end{bmatrix}, \tag{3.138}$$

the node voltage vector is given by

$$\mathbf{v}_{sr}^{Diode} = \begin{bmatrix} v_{Cathode}, & v_{Dnode1} \end{bmatrix}^T, \tag{3.139}$$

and the equivalent current source vector is given by

$$\mathbf{I}_{eqsr}^{Diode} = \begin{bmatrix} I_{Leq}, & I_{jeq} - I_{Leq} \end{bmatrix}^T. \tag{3.140}$$

Similarly, the one node omitted IGBT model (footnoted by s) can be given as followings where the 4×4 conductance matrix can be written as:

$$\mathbf{G}_s^{IGBT} = \begin{bmatrix} G_{pwld} + \frac{1}{R_{off}} & -G_{pwld} & 0 & 0 \\ -G_{pwld} & \begin{matrix} G_{tailvd} + G_{mosvd} \\ +G_{Cce} + G_{Ccg} \\ +G_{pwld} \end{matrix} & \begin{matrix} G_{tailvtail} - G_{mosvd} \\ -G_{tailvd} \end{matrix} & \begin{matrix} G_{mosvcge} + G_{tailvcge} \\ -G_{Ccg} \end{matrix} \\ 0 & -G_{mosvd} & \begin{matrix} \frac{1}{R_{tail}} + G_{C_{tail}} \\ +G_{mosvd} \end{matrix} & -G_{mosvcge} \\ 0 & -G_{Ccg} & 0 & \begin{matrix} G_{Ccg} + G_{Cge} \\ +\frac{1}{R_g + R_{gs}} \end{matrix} \end{bmatrix}, \tag{3.141}$$

the node voltage vector is written as:

$$\mathbf{v}_s^{IGBT} = \begin{bmatrix} v_{Collector}, & v_{Inode1}, & v_{Inode2}, & v_{Inode3} \end{bmatrix}^T, \tag{3.142}$$

and the equivalent current source vector is written as:

$$\mathbf{I}_{eqs}^{IGBT} = \begin{bmatrix} -I_{pwldeq} & \begin{matrix} I_{pwldeq} - I_{moseq} - I_{taileq} \\ -I_{Cceeq} - I_{Ccgeq} \end{matrix} & I_{moseq} - I_{Ctaileq} & \begin{matrix} I_{Ccgeq} - I_{Cgeeq} \\ +\frac{V_s}{R_g + R_{gs}} \end{matrix} \end{bmatrix}^T, \tag{3.143}$$

With detailed hardware modules for the power diode, the IGBT, and the linear circuit components, the emulation of a complete power converter can be realized. As shown in Fig. 3.29, it is composed of 3 terms for a simulation time-step Δt. The past interval calculated the values at the $(t - \Delta t)$ and it is responsible for sending the results to the current time or present interval as its history terms. Then, the present term is responsible for performing the two stages functions. Stage 1 processes the N-R iterations, which includes the calculations of the conductance matrix, equivalent current source vector for the decoupled converter circuit and parallel computing the circuit nodes voltages using linear solvers. Stage 2 executes the transmission line module to obtain the updated solution and send them to the next time interval. The future term receives the values from the present term and calculates results as the same procedure as previous for the time at $(t + \Delta t)$. The finite state machine is shown in Fig. 3.30. The diode part and IGBT part is triggered simultaneously and run in parallel. At stage 1, within the sub-units, some passive components remains the same results during N-R iterations unlike those of the nonlinear ones, but they are deliberately kept in the loop due to a more straightforward parallelism of the entire hardware structure. The history terms are updated based on successive a time-step and N-R history values are based on the N-R iterations within the current interval.

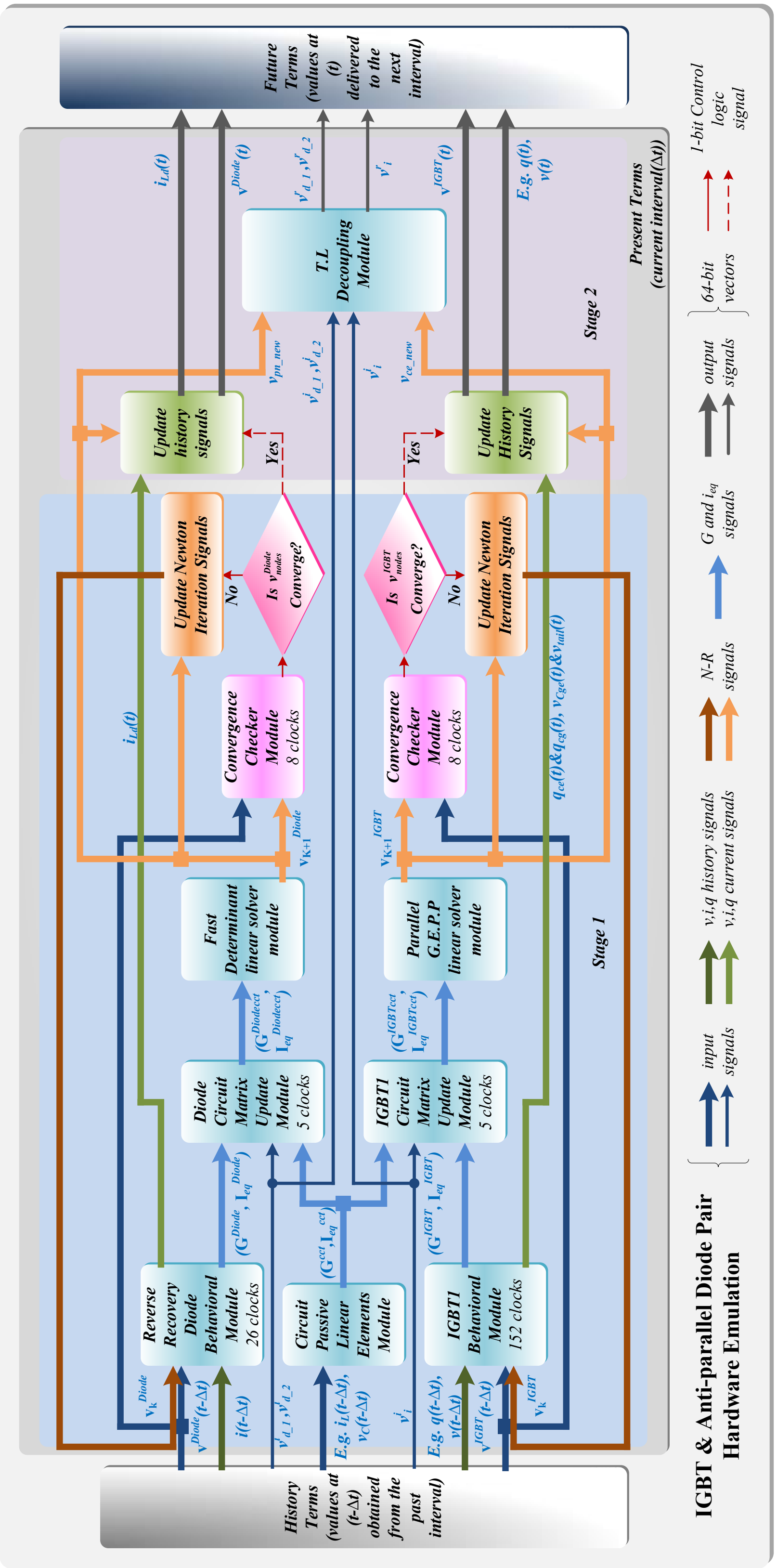

Future Terms (values at (t) delivered to the next interval)
$i_{Ld}(t)$
$v^{Diode}(t)$
$v^{r}_{d\_1}, v^{r}_{d\_2}$
v^{r}_{i}
$v^{IGBT}(t)$
E.g. $q(t)$, $v(t)$
T.L Decoupling Module
Present Terms (current interval(Δt))
Stage 2
v_{pn_new}
$v^{i}_{d\_1}, v^{i}_{d\_2}$
v^{i}_{i}
v_{ce_new}
Update history signals
Update History Signals
Yes
Yes
Update Newton Iteration Signals
Update Newton Iteration Signals
No
No
Is v^{Diode}_{nodes} Converge?
Is v^{IGBT}_{nodes} Converge?
$i_{Ld}(t)$
Convergence Checker Module 8 clocks
Convergence Checker Module 8 clocks
$q_{ce}(t)\&q_{cg}(t)$, $v_{Cge}(t)\&v_{tail}(t)$
v^{Diode}_{K+1}
v^{IGBT}_{K+1}
Fast Determinant linear solver module
Parallel G.E.P.P linear solver module
Stage 1
$(G^{Diodecct}, I^{Diodecct}_{eq})$
$(G^{IGBTcct}, I^{IGBTcct}_{eq})$
Diode Circuit Matrix Update Module 5 clocks
IGBT1 Circuit Matrix Update Module 5 clocks
$(G^{Diode}, I^{Diode}_{eq})$
(G^{cct}, I^{cct}_{eq})
$(G^{IGBT}, I^{IGBT}_{eq})$
Reverse Recovery Diode Behavioral Module 26 clocks
Circuit Passive Linear Elements Module
IGBT1 Behavioral Module 152 clocks
v^{Diode}_{k}
$v^{Diode}(t-\Delta t)$
$i(t-\Delta t)$
$v^{i}_{d\_1}, v^{i}_{d\_2}$
E.g. $i_L(t-\Delta t)$, $v_C(t-\Delta t)$
v^{i}_{i}
E.g. $q(t-\Delta t)$, $v(t-\Delta t)$
$v^{IGBT}(t-\Delta t)$
v^{IGBT}_{k}
History Terms (values at (t-Δt) obtained from the past interval)
IGBT & Anti-parallel Diode Pair Hardware Emulation
input signals
v,i,q history signals
v,i,q current signals
N-R signals
G and i_{eq} signals
output signals
64-bit vectors
1-bit Control logic signal

Figure 3.29: Architecture of the power converter SPTLM hardware emulation.

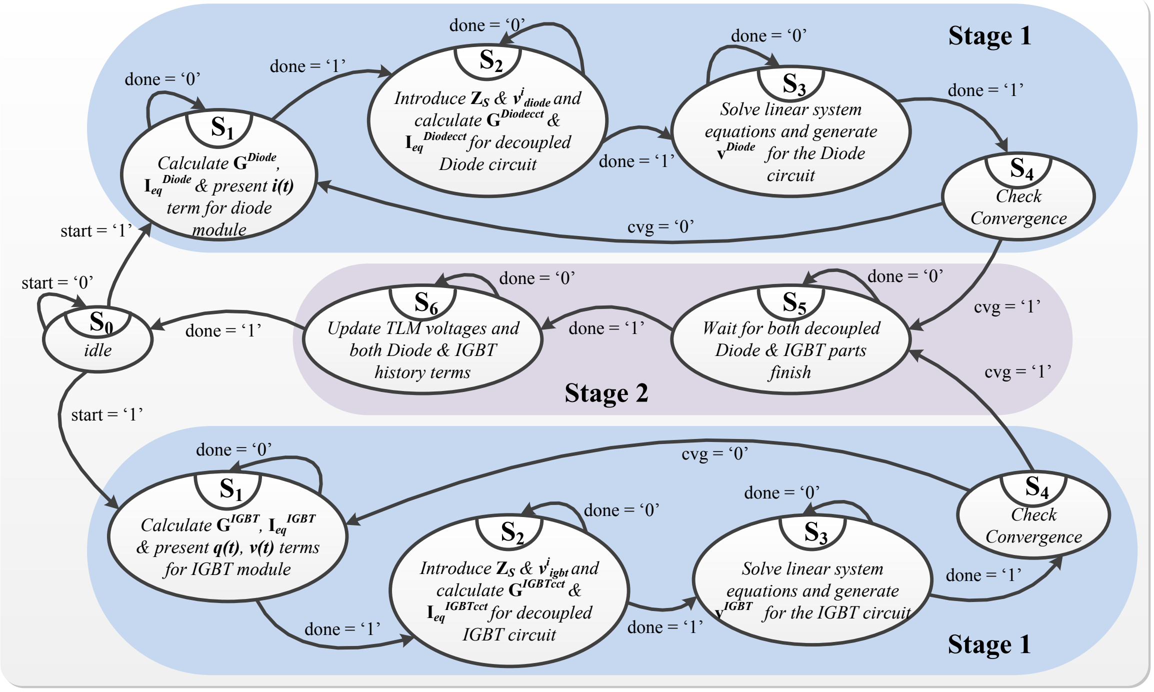

Stage 1
done = ‘0’
done = ‘1’
S₂
Introduce Z_S & v^i_diode and calculate G^Diodecct & I_eq^Diodecct for decoupled Diode circuit
done = ‘0’
S₃
Solve linear system equations and generate v^Diode for the Diode circuit
done = ‘1’
done = ‘0’
S₁
Calculate G^Diode, I_eq^Diode & present i(t) term for diode module
done = ‘1’
S₄
Check Convergence
cvg = ‘0’
start = ‘1’
start = ‘0’
S₀
idle
done = ‘1’
S₆
Update TLM voltages and both Diode & IGBT history terms
done = ‘0’
done = ‘1’
S₅
Wait for both decoupled Diode & IGBT parts finish
done = ‘0’
cvg = ‘1’
cvg = ‘1’
Stage 2
start = ‘1’
done = ‘0’
S₁
Calculate G^IGBT, I_eq^IGBT & present q(t), v(t) terms for IGBT module
cvg = ‘0’
S₄
Check Convergence
done = ‘0’
S₂
Introduce Z_S & v^i_igbt and calculate G^IGBTcct & I_eq^IGBTcct for decoupled IGBT circuit
done = ‘0’
S₃
Solve linear system equations and generate v^IGBT for the IGBT circuit
done = ‘1’
done = ‘1’
done = ‘1’
Stage 1

Figure 3.30: Finite state machine of the SPTLM.

3.8.3 Circuit Modeling and Decoupling for Parallel Computing

Using the proposed TLM method, large and complex circuits such as MMC with a considerable number of nodes can be divided into several separate sub-circuits which are linked to one another by an incident and reflected pulses. Therefore, the original large admittance matrix for the whole MMC circuit is replaced by multiple smaller matrices in terms of rank. Since the switches of MMC applied behavioral IGBT and anti-parallel diode pair models and there are quite a lot of nonlinearities, computation time will be extraordinarily long if a sequential method or conventional strategy is adopted (eg. directly solving a huge system of equations). Thus, detaching the switches from the MMC by appropriately inserting TLM in the circuit is necessary so that parallel computation can be achieved and emulation speed can be consequently accelerated.

Fig. 3.31(a) shows the modular multilevel converter basic topology. A single fundamental MMC cell in Fig. 3.31(b) usually noted as a submodule, consists of one DC capacitor and two switches. After introducing TLM links and stub models to each switch seen in Fig. 3.31(c), the circuit can be drawn as its hybrid Thévenin equivalent circuit shown in Fig. 3.31(d). As can be seen, the linear main part on the left side of Fig. 3.31(d) has only linear elements a resistor and a voltage source represented by RLC TLM stubs. Meanwhile, the IGBT and diode who cause extra resources in circuit computation are detached as well so that small separate sub-circuits are constituted. Consequently, by parity of reasoning, for MMC, the main circuit, and all sub-circuits could be computed in parallel and sub-circuits exchange information with the main circuit at the end of iteration within a time-step or the beginning of next time-step.

As an example of 3-phase 5-level MMC, since each leg or phase is identical except the phase difference, only one phase needs to be analyzed. In this condition, using Thévenin equivalent of TLM stub and link models to replace elements such as the switches (sub-modules), the main linear resistive circuit is established. Since there are 10 loops and 20 nodes in the main circuit, the rank of the matrix is smaller by using mesh analysis rather than nodal analysis. Then, the impedance $\mathbf{Z}$ matrix can be built as

$$\mathbf{Z}=\begin{pmatrix}
2\boldsymbol{Z_S}+\boldsymbol{Z_{C1}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\boldsymbol{Z_S} & 0\\
0 & 2\boldsymbol{Z_S}+\boldsymbol{Z_{C2}} & 0 & 0 & 0 & 0 & 0 & 0 & -\boldsymbol{Z_S} & 0\\
0 & 0 & 2\boldsymbol{Z_S}+\boldsymbol{Z_{C3}} & 0 & 0 & 0 & 0 & 0 & -\boldsymbol{Z_S} & 0\\
0 & 0 & 0 & 2\boldsymbol{Z_S}+\boldsymbol{Z_{C4}} & 0 & 0 & 0 & 0 & -\boldsymbol{Z_S} & 0\\
0 & 0 & 0 & 0 & 2\boldsymbol{Z_S}+\boldsymbol{Z_{C5}} & 0 & 0 & 0 & 0 & -\boldsymbol{Z_S}\\
0 & 0 & 0 & 0 & 0 & 2\boldsymbol{Z_S}+\boldsymbol{Z_{C6}} & 0 & 0 & 0 & -\boldsymbol{Z_S}\\
0 & 0 & 0 & 0 & 0 & 0 & 2\boldsymbol{Z_S}+\boldsymbol{Z_{C7}} & 0 & 0 & -\boldsymbol{Z_S}\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 2\boldsymbol{Z_S}+\boldsymbol{Z_{C8}} & 0 & -\boldsymbol{Z_S}\\
-\boldsymbol{Z_S} & -\boldsymbol{Z_S} & -\boldsymbol{Z_S} & -\boldsymbol{Z_S} & 0 & 0 & 0 & 0 & 4\boldsymbol{Z_S}+\boldsymbol{Z_L}+\boldsymbol{Z_{Lu}}+\boldsymbol{R} & -\boldsymbol{R}-\boldsymbol{Z_L}\\
0 & 0 & 0 & 0 & -\boldsymbol{Z_S} & -\boldsymbol{Z_S} & -\boldsymbol{Z_S} & -\boldsymbol{Z_S} & -\boldsymbol{R}-\boldsymbol{Z_L} & 4\boldsymbol{Z_S}+\boldsymbol{Z_L}+\boldsymbol{Z_{Lu}}+\boldsymbol{R}
\end{pmatrix} \tag{3.144}$$

where $Z_{Ci}(i=1,2,\cdots)$ is the surge impedance of DC capacitor, Z_L, Z_{Lu} and Z_{Ld} are surge impedance of L, Lu and Ld respectively. Hence, the equation for the main circuit can be

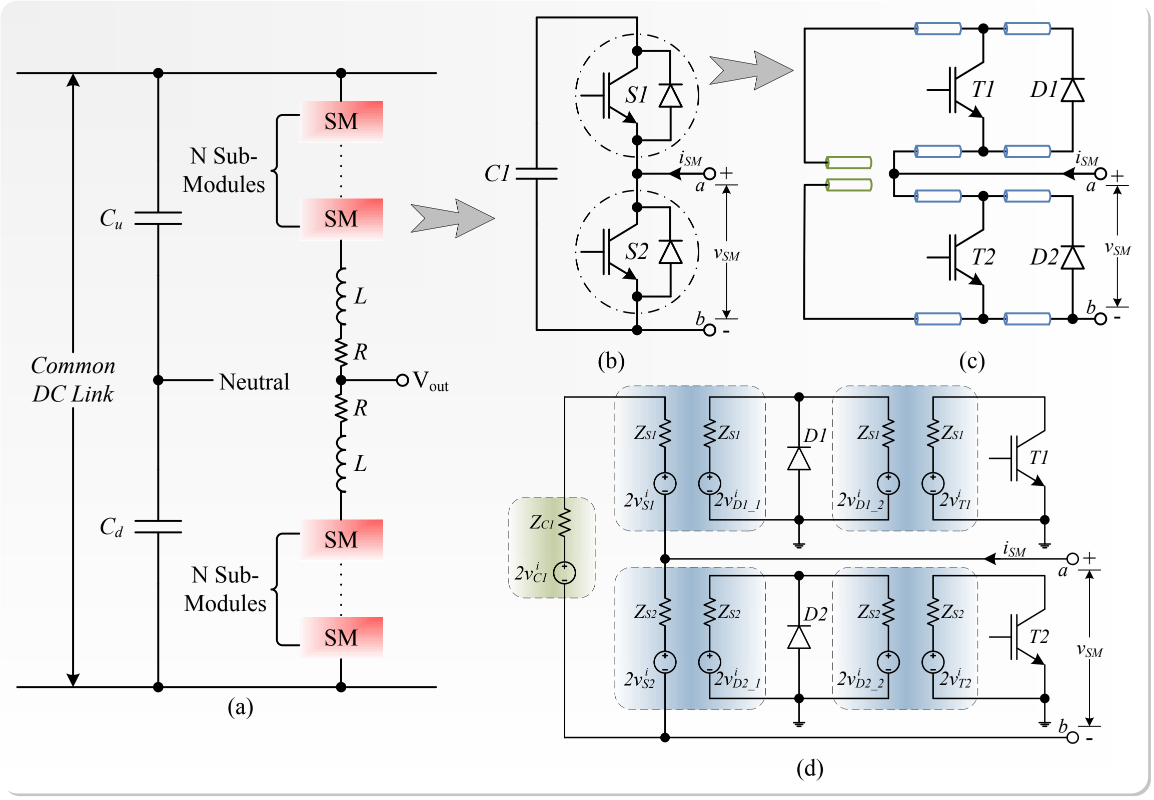


Figure 3.31: (a) Configuration of the MMC topology and its sub-module. (b) Fundamental MMC cell. (c) TLM links and stub inserted MMC sub-module. (d) Sub-module with hybrid TLM Thévenin equivalent circuit.

written as:

$$ {}_{n}\mathbf{I} = \mathbf{Z}^{-1} \cdot_{n} \mathbf{U}, \tag{3.145} $$

where the loop current vector to be solved takes the form of ${}_{n}\mathbf{I} = [{}_{n}I_1 \;\; {}_{n}I_2 \;\; {}_{n}I_3 \cdots_{n} I_j]^T$ and the voltage vector, whose elements can be represented as a function of incident pulse vector, reflects voltage source distribution in the Thévenin circuit model-based circuit forms as ${}_{n}\mathbf{U} = [f_1({}_{n}v^i) \quad f_2({}_{n}v^i) \quad f_3({}_{n}v^i) \cdots f_j({}_{n}v^i)]^T$ where j denoted the number of elements in the main circuit. The incident pulse vector and the reflected pulse vector are given as ${}_{n}\mathbf{v}^{i,r} = [{}_{n}v_1^{i,r} \quad {}_{n}v_2^{i,r} \quad {}_{n}v_3^{i,r} \cdots_{n} v_k^{i,r}]^T$ where k is the number of elements in all Thévenin equivalent sub-circuits. Then, the loop current in the circuit can be obtained from above equation and the voltage across each of sub-module Thévenin circuits can be calculated by:

$$ {}_{n}v_k = (\mathbf{C}_{n}\mathbf{I})_k \cdot Z_k + 2_{n}v_k^i, k = 1, 2 \cdots, \tag{3.146} $$

where $\mathbf{C}$ is a matrix filled with constants used to calculate branch current, Z represents either Z_{ci}, Z_L, Z_{Lu} or Z_{Ld}. Noting that the sum of incident and reflected pulses constitutes the actual voltage. For TLM stubs, according to aforementioned analysis, the reflected pulses travel back through lines to the original ports and subsequently become incident pulses for the next time-step. While for TLM link, they become incident pulses for the other terminal in separate sub-circuits.

The sub-circuits are structurally identical. By applying KCL analysis, the nodal voltages of each unit can be expressed as:

$$ {}_{n}\mathbf{U} = \mathbf{G}^{-1} \cdot_{n} \mathbf{I}, \tag{3.147} $$

where ${}_{n}\mathbf{U}$ represents IGBT and diode pair nodal voltage vector $[{}_{n}U_1 \;\; {}_{n}U_2 \cdots_{n} U_l]^T$ (l is the level amount of MMC). $\mathbf{G}$ and ${}_{n}\mathbf{I}$ are the nodal admittance matrix and nodal source vector of the network at n^{th} step. Then, the voltage across TLM link models' right-hand side part of the Norton equivalent circuit ${}_{n}v$ is obviously equal to ${}_{n}U_1$. Thus, the reflected pulse can be calculated, which in turn, becomes the incident pulse at the corresponding port in the main circuit for the next time-step.

3.9 Summary

This chapter proposed several algorithms for parallel computation of different parts of a power converter such as Backward Euler, N-R iteration, and transmission line decoupling. Meanwhile, digital hardware emulation of the detailed power diode, Saber® based device-level behavioral $IGBT1$ models and their combination with TLM are also elaborated in this chapter. These models are fully paralleled on the FPGA. Their hardware emulations are based on a unified numerical framework, and can be extended in a straightforward fashion to model complete power electronics circuits. The Newton-Raphson iteration and Backward Euler linearization are very advantageous in power converter hardware emulations, which may involve all kinds of different topologies and these methods are quite useful due to their strong flexibility and convenience in the linearized model formulation. The variable time-step solution makes the hardware emulations of the power converter faster and consumes fewer resources. One phase of the three-phase-symmetrical five-level MMC is studied as an example of the implementation of SPTLM which is placed inside a switch or sub-module in MMC. Using the transmission line decomposition, the MMC is effectively separated into seventeen parts. Each part has a nonlinear power switch network consisting of a behavioral $IGBT1$ and a diode linked by transmission lines. The solution to this system is the simultaneous calculation of all seventeen parts by independent matrices, for the main circuit that is linear, data acquired from mesh current equations are exactly the final results, whereas iterations are needed for nonlinear sub-circuits. The SPTLM is applicable to any type of power converter circuit regardless of how many switches are contained so as to decouple the system and perform the parallel computation.

4
Case Studies and Experimental Results

The case studies firstly focus on the basic device-level simulation of a single power diode and an IGBT to validate their hardware models. Secondly, the proposed IGBT and simplified diode hardware models are emulated using chopper circuit for the testing purpose. Then, both of the IGBT and power diode with reverse recovery models are applied to a half-bridge circuit to check their own as well as their combined functionality as a pair. Lastly, a five-level MMC is simulated to verify the proposed circuit splitting methodologies. The hardware emulation oscilloscope results of each device-level test cases are captured; and they are compared with the corresponding off-line SaberRD® software simulations. The SaberRD® software runs under 64-bit Windows® 7 Enterprise SP1 operating system and the hardware configuration of the host PC is Intel® Core$^{\text{TM}}$i5 3.33GHz CPU with 4.00GB RAM memory. The oscilloscope waveforms are captured from Tektronix DPO 7054 Digital Phosphor Oscilloscope which has 4 input channels (where channel 1 and 2 are for 16-bits fixed-point inputs and channel 3 and 4 are for 14-bits fixed-point inputs).

4.1 Simple IGBT and Power Diode Test Circuits

The proposed power diode and IGBT behavioral hardware models are firstly tested in the simple circuits shown in Fig. 4.1.

4.1.1 Single Diode

As shown in Fig. 4.1(a), with a resistance load and square wave voltage source controlled by PWM, the diode turn-on, and turn-off transient characteristics as well as the DC mode behaviors are obtained. Table 4.1 lists the test circuit and diode parameters obtained from Saber®.

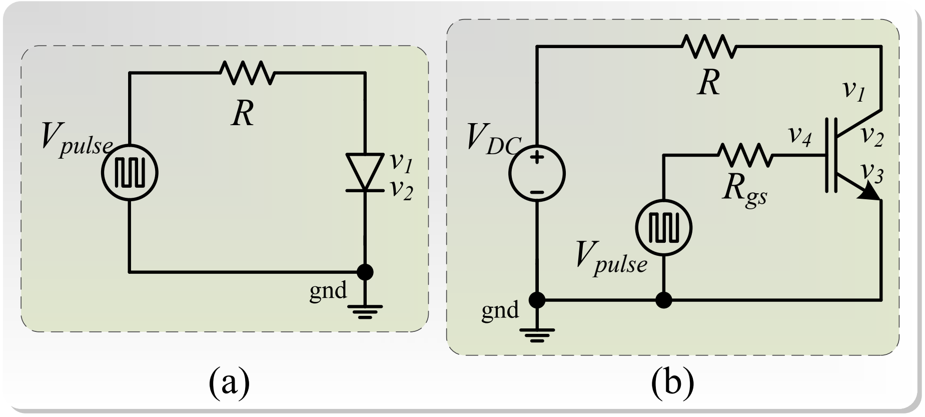


Figure 4.1: (a) Test circuit for diode. (b) Test circuit for IGBT.

Table 4.1: Test Circuit and Diode Parameters

Test Circuit Parameters
$V_{pulse} = \pm 1V$, $(f = 2.5kHz, width = 20\mu s, period = 40\mu s)$, $R = 5\Omega$

Diode Parameters
$r_{on} = 10m\Omega$, $r_{off} = 100k\Omega$, $V_{on} = 0.7V$, $I_{Fo} = 10A$, $\frac{\mathrm{d}I_r}{\mathrm{d}t} = 50 \times 10^6$, $I_{rrm} = 10A$, $t_{rr} = 2\mu s$, $\frac{\mathrm{d}I_f}{\mathrm{d}t} = 100 \times 10^6$, $V_{fp} = 50V$, $t_{fp} = 200ns$, $V_1 = 2V$, $V_2 = 20V$, $C_1 = 1nF$, $C_2 = 0.2nF$, $T_m = 1.400\mu s$, $\beta = 9.500 \times 10^{-5}$, $\tau = 1.770\mu s$, $R_0 = 28.219\Omega$, $K = 9.883 \times 10^4$, $L = 10 \times 10^{-12}H$, $R_L = 1.279 \times 10^{-5}\Omega$

4.1.1.1 Circuit Description

Basically, the diode used in this circuit is previously shown in Fig. 3.27(a) which includes both forward and reverse recovery phenomena. The circuit node voltages v_1 and v_2 are the corresponding diode internal node voltages v_{Dnode1} and v_{Dnode2}. Referring to Fig. 3.8, with the cathode grounded and the voltage source V_m placed between anode and Dnode1, the order of the circuit can be reduced and the linear system of equations can be formed as following (4.1) based on nodal analysis:

$$\begin{bmatrix} \frac{1}{R} + G_j + G_{Cvj} & K - G_{Cvj} - G_j \\ G_j & -G_j - G_{RL} - G_L \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \frac{V_s - V_m}{R} - I_{jeq} - I_{Cvjeq} \\ I_{Leq} - I_{jeq} \end{bmatrix}, \tag{4.1}$$

where V_s is the voltage source V_{pulse}. Then, the fully diode voltage between anode and cathode can be given by:

$$v_d = v_{Anode} - v_{Cathode} = v_1 + V_m. \tag{4.2}$$

The diode current is calculated by:

$$i_d = \frac{(V_s - v_{Anode})}{R} = \frac{V_s - (V_m + v_1)}{R}. \tag{4.3}$$

Then, the diode instantaneous power dissipation can be obtained by:

$$P_{diode} = v_d \times i_d. \tag{4.4}$$

4.1.1.2 Resource Usage

The Xilinx® Virtex®-7 VC707 XC7VX485T FPGA board is used to emulate this test circuit and the board resource usage is listed in Table 4.2.

Table 4.2: Diode Test Circuit Hardware Resources Utilization

Resources	Overall Diode Circuit
LUT	15485 (5.10%)
$LUTRAM$	272 (0.21%)
FF	11440 (1.88%)
$BRAM$	4 (0.39%)
$DSP48$	256 (9.14%)
I/O	91 (13.86%)
$BUFG$	1 (3.12%)

4.1.1.3 Results and Comparison

The clock frequency is set to 100 MHz. The simulation time-step is chosen to be 200 ns and the computational latency of the circuit hardware emulation for a full iteration within a time-step from initial inputs to final outputs is 92 clock cycles. With proper output control, the speed of hardware emulation is about 10 times slower than the real-time. For example, to simulate the test circuit in Fig. 4.2 for 2 ms (5-period duration), it would actually take 20 ms on board. However, the executive time of SaberRD® simulation is about 100 ms using its built-in variable time control. This indicates simulating the diode hardware model on FPGA board is 5 times faster than executing it in the off-line SaberRD® software. The steady-state results of diode voltage v_d in Fig. 4.2(a), current i_d in Fig. 4.2(b) and instantaneous power dissipation P_{diode} in Fig. 4.2(c) are captured in oscilloscope and compared with SaberRD®. The transient results of the device-level model during diode ON in Fig. 4.3(a) and OFF in Fig. 4.3(b) are depicted such as the voltage forward recovery phenomenon and current reverse recovery phenomenon where t_{fr} indicates the forward recovery time and t_{rr} refers to the reverse recovery time. All the waveforms from hardware emulation in the oscilloscope (left side) and their corresponding off-line simulation in SaberRD® (right side) are fully and comprehensively compared with each other. Table 4.3 provides the comparison of transient times including forward and reverse recovery. Meanwhile, the comparison of average power dissipation of diode during on/off for forward and reverse recovery (P_{fr}/P_{rr}) as well as conduction (P_{cond}) are also listed.

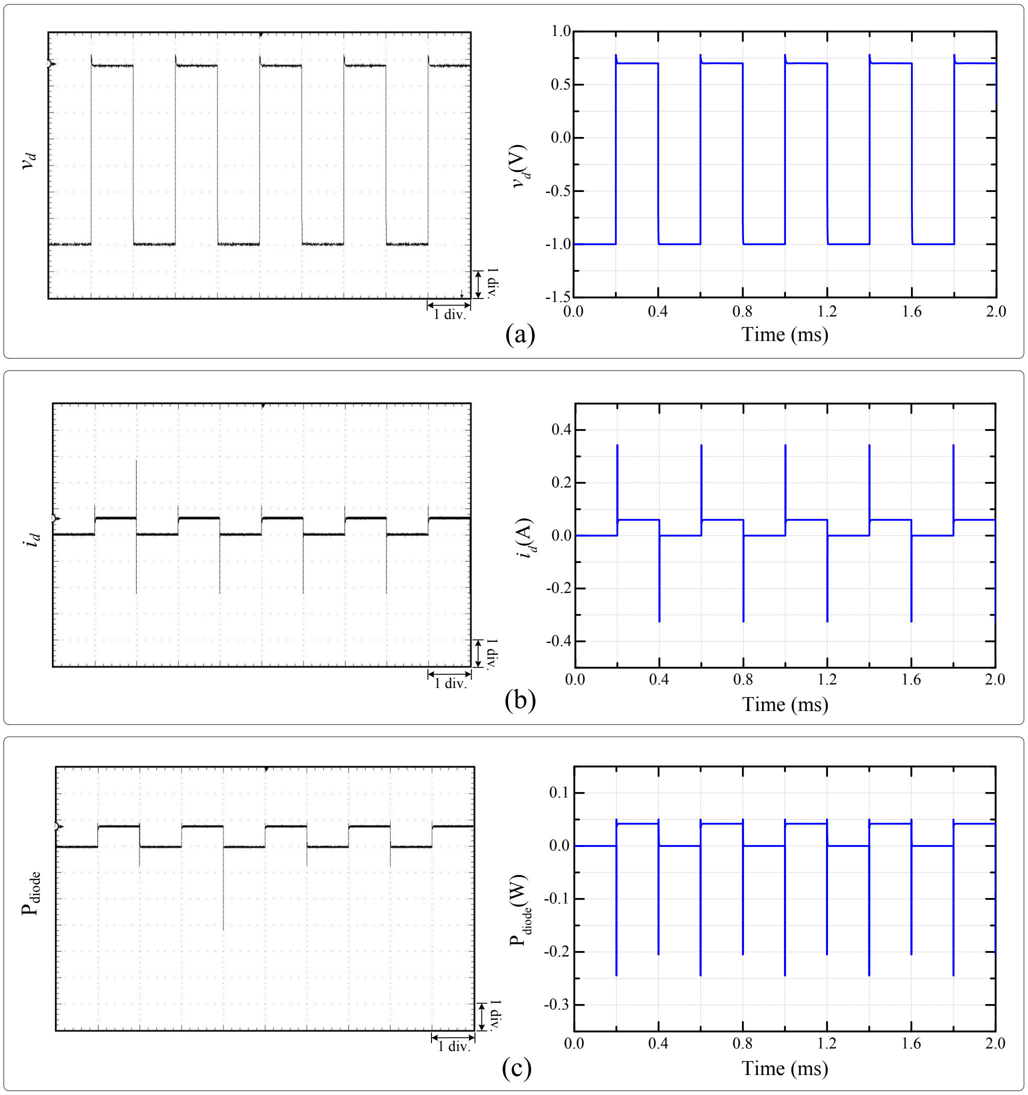


Figure 4.2: Steady-state results form diode hardware emulation in oscilloscope (left side) and off-line simulation in SaberRD® (right side) of (a) diode voltage, (b) diode current, and (c) diode instantaneous power dissipation. Scale: y-axis: (a) 0.25 V/div., (b) 0.1 A/div., (c) 0.05 W/div.; x-axis: (a)-(c) 2 ms.

Some errors are expected and acceptable mainly due to the low output precision (16-bit fixed-point number) and different numerical solution approaches compared to SaberRD®. Based on the comparison results, under the switching frequency of 2.5kHz, all the oscilloscope waveforms basically agree with the simulation results from SaberRD®. Some small amplitude differences are caused by the comparatively lower precession and sample rates from the oscilloscope. Thus, the validation process is completed and the device-level hardware model can provide an accurate diode behavior under both steady-state and transient conditions.

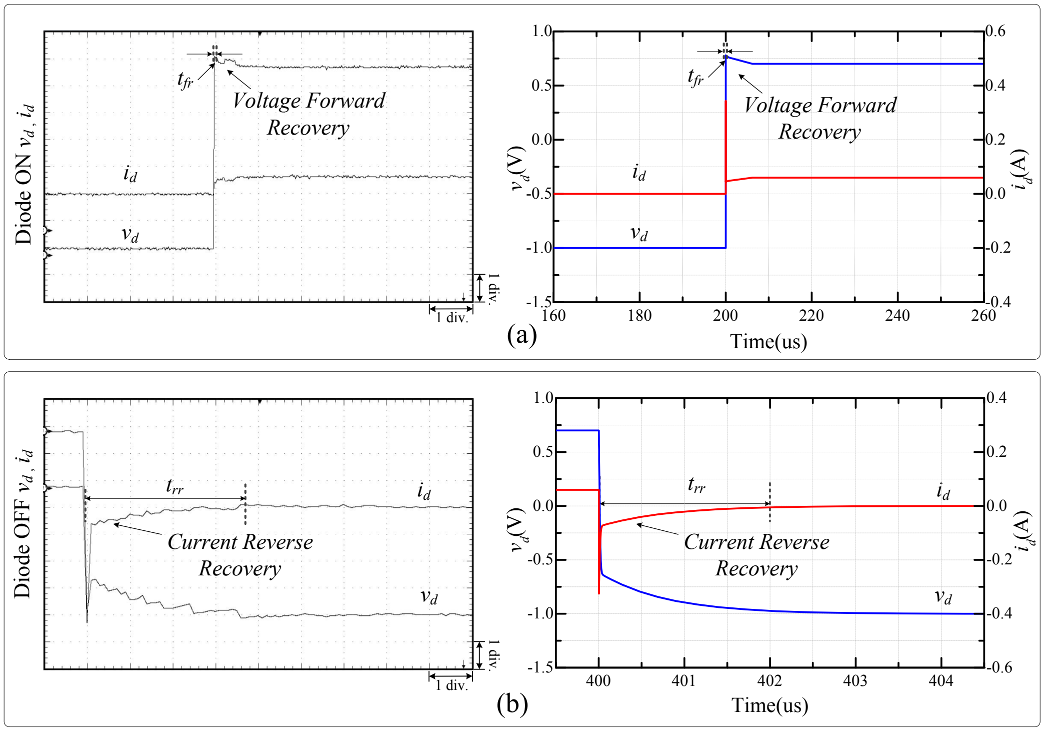


Figure 4.3: Simple diode device-level hardware emulation transient-state results in oscilloscope (left side) and off-line simulation results in SaberRD® (right side) of (a) diode on voltage and current, (b) diode off voltages and currents. Scale: y-axis: (a)-(b) 0.25 V/div., 0.1 A/div.,; x-axis: (a) 100 μs, (b) 50 μs.

Table 4.3: Comparison of Transient Time and Power Dissipation of Diode under a Volatge Pulse at Frequency of 2.5kHz

	SaberRD®	**FPGA**	**Error**
	Switching times		
t_{rr}	1950ns	2000ns	2.5%
t_{fr}	210ns	200ns	5.0%
	Dissipated Power		
P_{fr}	9.8mW	10.4mW	5.8%
P_{rr}	23.4mW	24.6mW	4.9%
P_{cond}	42mW	41.9mW	0.2%

4.1.2 Single IGBT

As shown in Fig. 4.1(b), with a resistance load, a square wave PWM voltage source at the gate and a DC voltage source in the test circuit, the IGBT steady-state behaviors as well as the turn-on and turn-off transient characteristics can be observed. In this test circuit, the Siemens BSM300GA160D is chosen to be the IGBT device whose parameters could be acquired from SaberRD®. Table 4.4 lists the parameters of this test case for the IGBT behavioral model and the other elements connected in the circuit.

Table 4.4: Test Circuit and IGBT Parameters

Test Circuit Parameters
$V_{pulse} = 5 \pm 10V$, $(f = 2.5kHz, width = 20\mu s, period = 40\mu s)$, $R = 30\Omega$, $R_{gs} = 35\Omega$, $V_{DC} = 100V$

IGBT Parameters
$r_{off} = 10^9\Omega$, $g_{off} = 10^{-12}S$, $g_{on} = 10^6S$, $R_g = 5\Omega$, $vce1 = 4.8V$, $vge1 = 9V$, $ic1 = 225A$, $vce2 = 1.8V$, $vge2 = 7V$, $ic2 = 20A$, $vce3 = 4V$, $vge3 = 17V$, $ic3 = 400A$, $V_t = 6.3V$, $V_{on} = 0.8V$, $vce4 = 10V$, $vce5 = 4V$, $vce6 = 800V$, $vge4 = 10V$, $vge5 = 20V$, $itail1 = 100A$, $itail2 = 5A$, $crss1 = 30nF$, $crss2 = 1.6nF$, $coss1 = 42nF$, $coss2 = 5nF$, $q1 = 400nC$, $q2 = 2000nC$, $q3 = 3500nC$, $\tau = 10\mu s$, $m = 0.5$, $R_{tail} = 1\mu\Omega$, $C_{tail} = 10F$, $a1 = 0.0217$, $a3 = 91.705$, $b1 = 0.00395$, $b3 = 3.221$, $x = 0.973$, $y = 1.428$, $z = 0.369$, $icsat3 = 1.789kA$, $cceo = 12nF$, $ccgo = 110nF$, $cgeo = 40nF$, $vceo = 0.873V$, $vcgo = 0.0189V$ obtained from SaberRD®

4.1.2.1 Circuit Description

In Fig. 4.1(b), the circuit node voltages v_1, v_2, v_3 and v_4 are all the corresponding IGBT node voltages $v_{Collector}$, v_{Inode1}, v_{Inode2} and v_{Inode3} illustrated in Fig. 3.11. According to Fig. 3.11 and the equations from (3.97) to (3.99), with the emitter grounded, one node can be reduced from the circuit linear system. Then, the matrix equation can be given as the following (4.5) based on nodal analysis:

$$\begin{bmatrix} \mathbf{G}^{IGBT}(1,1) + \frac{1}{R} & \mathbf{G}^{IGBT}(1,2) & \mathbf{G}^{IGBT}(1,3) & \mathbf{G}^{IGBT}(1,4) \\ \mathbf{G}^{IGBT}(2,1) & \mathbf{G}^{IGBT}(2,2) & \mathbf{G}^{IGBT}(2,3) & \mathbf{G}^{IGBT}(2,4) \\ \mathbf{G}^{IGBT}(3,1) & \mathbf{G}^{IGBT}(3,2) & \mathbf{G}^{IGBT}(3,3) & \mathbf{G}^{IGBT}(3,4) \\ \mathbf{G}^{IGBT}(4,1) & \mathbf{G}^{IGBT}(4,2) & \mathbf{G}^{IGBT}(4,3) & \mathbf{G}^{IGBT}(4,4) \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{eq}^{IGBT}(1) + \frac{V_{DC}}{R} \\ \mathbf{I}_{eq}^{IGBT}(2) \\ \mathbf{I}_{eq}^{IGBT}(3) \\ \mathbf{I}_{eq}^{IGBT}(4) \end{bmatrix}. \tag{4.5}$$

where V_s in previous Ch. 3 (3.97) is the voltage source V_{pulse}. Then, the IGBT collector to emitter voltage is given by:

$$v_{ce} = v_{Collector} - v_{Emitter} = v_1. \tag{4.6}$$

As the emitter is grounded, the collector current can be calculated by:

$$i_c = \frac{(V_{DC} - v_{Collector})}{R} = \frac{V_{DC} - v_1}{R}. \tag{4.7}$$

Also, the IGBT instantaneous power dissipation, which is mainly form the switching loss and conduction loss, can be obtained by:

$$P_{IGBT} = v_{ce} \times i_c. \tag{4.8}$$

4.1.2.2 Resource Usage

The resource utilization of emulating this test circuit on the Xilinx® Virtex®-7 VC707 XC7VX485T FPGA board is listed in Table 4.5.

Table 4.5: IGBT Test Circuit Hardware Resources Utilization

Resources	Overall IGBT Circuit
LUT	38226 (12.59%)
$LUTRAM$	959 (0.74%)
FF	26929 (4.44%)
$BRAM$	10 (0.97%)
$DSP48$	823 (29.41%)
I/O	97 (13.86%)
$BUFG$	1 (3.12%)

4.1.2.3 Results and Comparison

The clock frequency is set to 100 MHz. The simulation time-step is chosen to be 200 ns and the computational latency of the circuit hardware emulation for a full iteration within a time-step from initial inputs to final outputs is 226 clock cycles. With proper output control, the speed of hardware emulation is about 25 times slower than the real-time. For example, to simulate the test circuit for 2 ms (5-period duration) as in Fig. 4.4, it would actually take 50 ms on board. However, the executive time of SaberRD® simulation is about 100 ms using its built-in variable time control which indicates simulating this hardware model on FPGA board is 2 times faster than executing it in the off-line SaberRD® software. As shown in Fig. 4.4(a) — (d), single IGBT device-level hardware emulation results in oscilloscope (left side) and off-line simulation results in SaberRD® (right side) are fully and comprehensively compared with each others such as the steady-state and transient-state collector-emitter voltage v_{ce} and collector current i_c as well as the IGBT instantaneous power dissipation P_{IGBT} in Fig. 4.4(a) — (d). Table 4.6 provides the comparison of the transient times t_r (rise time from 10% to 90% of collector current) and t_f (fall time from 90% to 10% of collector current) as well as the average power dissipation for IGBT during switching on/off (P_{on}/P_{off}) and conduction (P_{cond}). Some errors are expected and acceptable as they are mainly due to the low output precision (16-bit fixed-point number) and different numerical solution approaches compared to SaberRD®. Based on the comparison results,

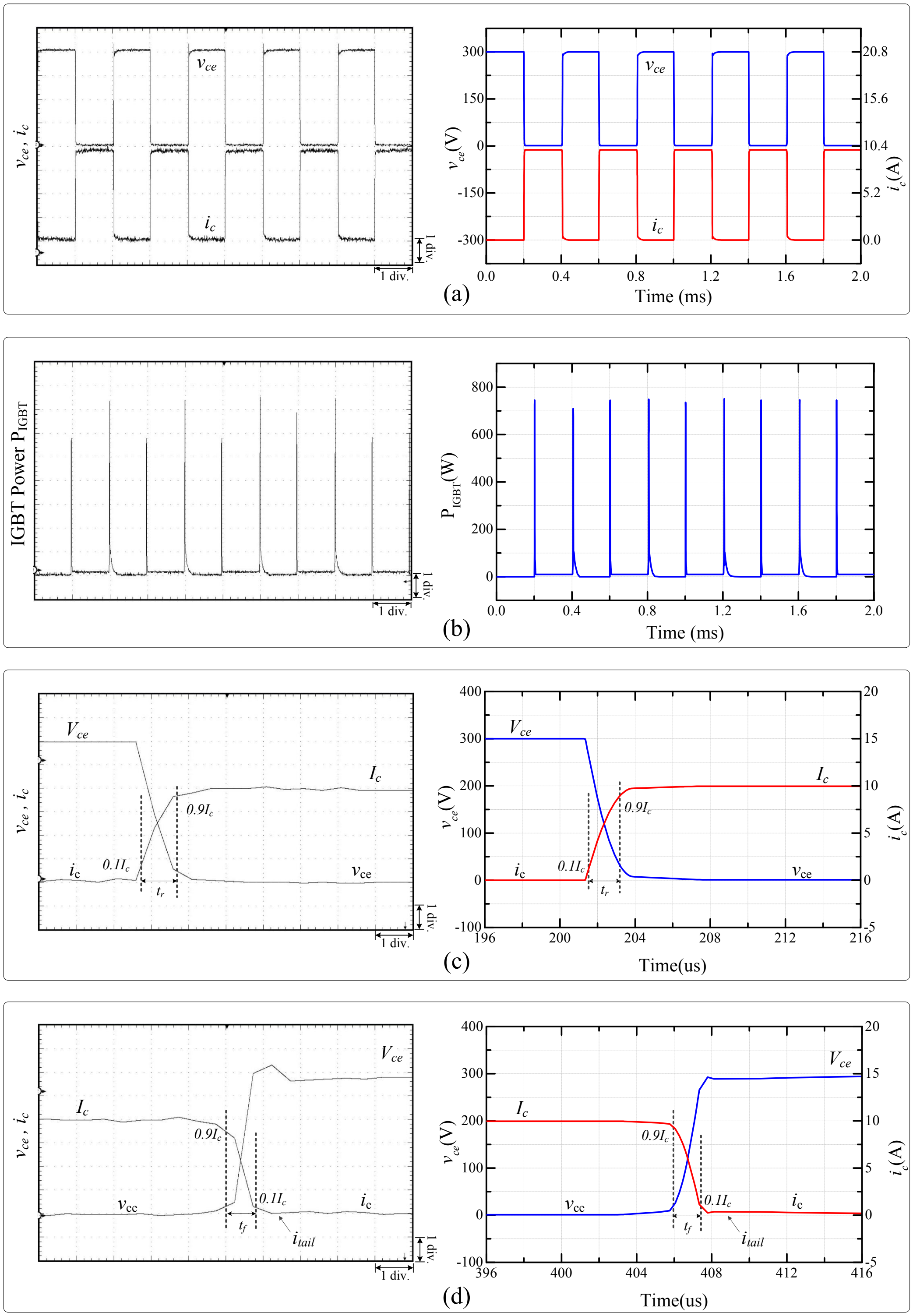


Figure 4.4: Single IGBT device-level hardware emulation results in oscilloscope (left side) and off-line simulation results in SaberRD® (right side) of (a) steady-state collector-emitter voltage and collector current, (b) IGBT instantaneous power dissipation, (c) and (d) transient-state voltages and currents. Scale: (a) y-axis: 75 V/div., 2.6 A/div.; (b) y-axis: 100 W/div.; (a)-(b) x-axis: 5 ms, (c)-(d) y-axis: 50 V/div., 2.5 A/div.; (c)-(d) x-axis: 50 μs.

under the switching frequency of 2.5kHz, all the oscilloscope waveforms basically agree with the simulation results from SaberRD®. Some small amplitude differences are caused by the comparatively lower precession and sample rates from the oscilloscope. Thus, the validation process is completed and the device-level hardware model can provide an accurate IGBT behavior under both steady-state and transient conditions. Moreover, the hardware model is also working at a relatively high frequency of 25kHz.

Table 4.6: Comparison of Switching Time and Power Dissipation of IGBT under a Switching Frequency of 2.5kHz

	SaberRD®	**FPGA**	**Error**
	Switching times		
t_r	2.0 μs	1.9 μs	5.3%
t_f	1.6 μs	1.5 μs	6.7%
	Dissipated Power		
P_{on}	191.5 W	203.6W	5.9%
P_{off}	79.5W	75.8W	4.9%
P_{cond}	10.0W	9.95W	0.5%

4.2 Chopper Circuit

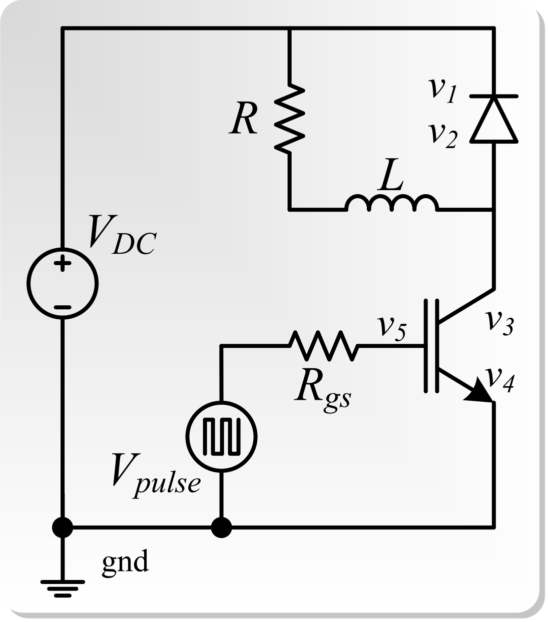


Figure 4.5: Chopper test circuit.

As shown in Fig. 4.5, a standard chopper circuit is built for testing the IGBT and diode model. The IGBT used in this chopper circuit is the same as the one in the single IGBT test circuit and the diode is the Saber® based diode model with only reverse recovery which is proposed in the previous chapter as the simplified diode model seen in Fig. 3.27(b). Their parameters can refer to Table 4.1 and Table 4.4. Table 4.7 provides the other basic parameters in this test circuit such as the DC voltage source, the series-connected resistor and inductor.

Table 4.7: Chopper Test Circuit and Device Parameters

Test Circuit Parameters
$V_{pulse} = 5 \pm 10V$, $(f = 2.5kHz, width = 20\mu s, period = 40\mu s)$, $V_{DC} = 50V$, $R = 1\Omega$, $L = 100\mu H$, $R_{gs} = 35\Omega$

4.2.1 Circuit Description

The IGBT gate voltages are square waves controlled by PWM with a duty cycle of 0.5. v_1 and v_2 are the diode node voltages. While, v_3, v_4 and v_5 are the internal IGBT node voltages. The other passive linear components such as the inductor L, resistors R and R_{gs}, can be discretized as described in previous chapter. For example, the RL part can be discretized as given in Fig. 3.5 with the corresponding equations expressed in (3.29) — (3.32). Finally, combining with the foregone nodes such as the ground connected emitter of the IGBT and the DC voltage source connected cathode of the diode as well as referring to (3.138) — (3.143), the entire chopper circuit can be discretized and linearized as the linear matrix system of equations shown in (4.9).

$$\begin{bmatrix} \mathbf{G}_{sr}^{Diode}(2,2) & -G_j & 0 & 0 & 0 \\ K - G_j & \begin{array}{c} G_j + G_{RLeq} \\ +G_{pwld} \end{array} & -G_{pwld} & 0 & 0 \\ 0 & \mathbf{G}_s^{IGBT}(2,1) & \mathbf{G}_s^{IGBT}(2,2) & \mathbf{G}_s^{IGBT}(2,3) & \mathbf{G}_s^{IGBT}(2,4) \\ 0 & \mathbf{G}_s^{IGBT}(3,1) & \mathbf{G}_s^{IGBT}(3,2) & \mathbf{G}_s^{IGBT}(3,3) & \mathbf{G}_s^{IGBT}(3,4) \\ 0 & \mathbf{G}_s^{IGBT}(4,1) & \mathbf{G}_s^{IGBT}(4,2) & \mathbf{G}_s^{IGBT}(4,3) & \mathbf{G}_s^{IGBT}(4,4) \end{bmatrix} \cdot \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{bmatrix} = \begin{bmatrix} \begin{array}{c} \mathbf{I}_{eqsr}^{Diode}(2) - \\ V_{DC}\mathbf{G}_{sr}^{Diode}(2,1) \end{array} \\ \begin{array}{c} \mathbf{I}_{eqs}^{IGBT}(1) + \\ V_{DC}(K + G_{RLeq}) \\ +I_{RLeq} - I_{jeq} \end{array} \\ \mathbf{I}_{eqs}^{IGBT}(2) \\ \mathbf{I}_{eqs}^{IGBT}(3) \\ \mathbf{I}_{eqs}^{IGBT}(4) \end{bmatrix} . \tag{4.9}$$

Then, the voltage between anode and cathode of the reverse recovery diode can be given by:

$$v_{rd} = v_{Anode} - v_{Cathode} = v_2 - V_{DC}, \tag{4.10}$$

and the diode current can be calculated by:

$$i_{rd} = (v_1 - V_{DC}) * \mathbf{G}_{sr}^{Diode}(1,1) + I_{Leq}. \tag{4.11}$$

Meanwhile, the voltage between collector and emitter of the IGBT can be given by:

$$v_{ce} = v_{Collector} - v_{Emitter} = v_2, \tag{4.12}$$

and the collector current can be calculated by:

$$i_c = (v_2 - v_3)G_{pwld} - I_{pwldeq}. \tag{4.13}$$

4.2.1.1 Resource Usage

The resource utilization of emulating this test circuit on the Xilinx® Virtex®-7 VC707 XC7VX485T FPGA board is listed in Table 4.8.

Table 4.8: Chopper Test Circuit Hardware Resources Utilization

Resources	**Overall Chopper Circuit**
LUT	49387 (16.27%)
$LUTRAM$	1085 (0.84%)
FF	34987 (5.77%)
$BRAM$	12 (1.16%)
$DSP48$	1003 (35.84%)
I/O	162 (23.15%)
$BUFG$	1 (3.12%)

4.2.1.2 Results and Comparison

The FPGA board clock frequency is set to 100 MHz. The simulation time-step is chosen to be 200 ns and the computational latency of the circuit hardware emulation for a full iteration (from initial inputs to final outputs) within a single time-step is 256 clock cycles. The speed of hardware emulation of this chopper circuit is about 25 times slower than the real-time after using proper output control. The total latency is close to the IGBT single test circuit since all of the components are executed in parallel and the dominated part is still the IGBT. The order of the linear system of equations is increased by 1 so implementing the matrix solver unit would be more time consuming. However, the executive time of SaberRD® simulation is about 100 ms using its built-in variable time control which indicates simulating this hardware model on FPGA board is 2 times faster than executing it in the off-line SaberRD® software. Then, the amount of speedup is the same as the IGBT test circuit. As seen in Fig. 4.7(d), the transient behavior reverse recovery is completely an accurately captured based on this chopper circuit. In Fig. 4.6 and Fig. 4.7, the IGBT and diode models hardware emulation waveforms in oscilloscope (left side) based on the chopper circuit are fully and comprehensively compared with the off-line simulation results in SaberRD® (right side) such as the IGBT collector-emitter voltage v_{ce}, collector current i_c,

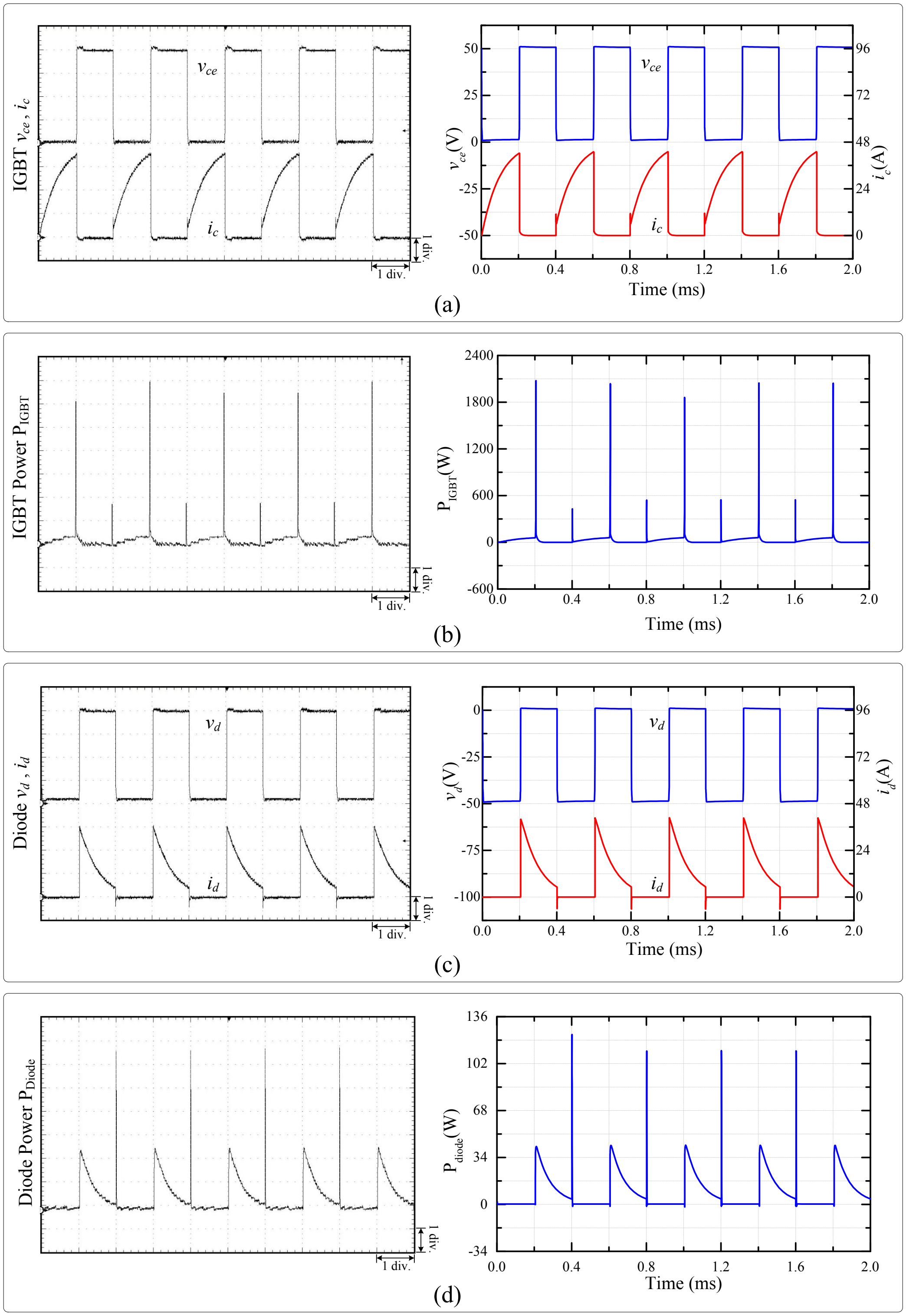


Figure 4.6: Chopper circuit hardware emulation steady-state results in oscilloscope (left side) and off-line simulation results in SaberRD® (right side) of (a) IGBT collector-emitter voltage and collector current, (b) IGBT instantaneous power dissipation, (c) Diode voltage and current, and (d) Diode instantaneous power dissipation. Scale: (a) and (c) y-axis: 12.5 V/div., 12 A/div.; (b) y-axis: 300 W/div.; (d) y-axis: 17 W/div.; (a)-(d) x-axis: 5 ms.

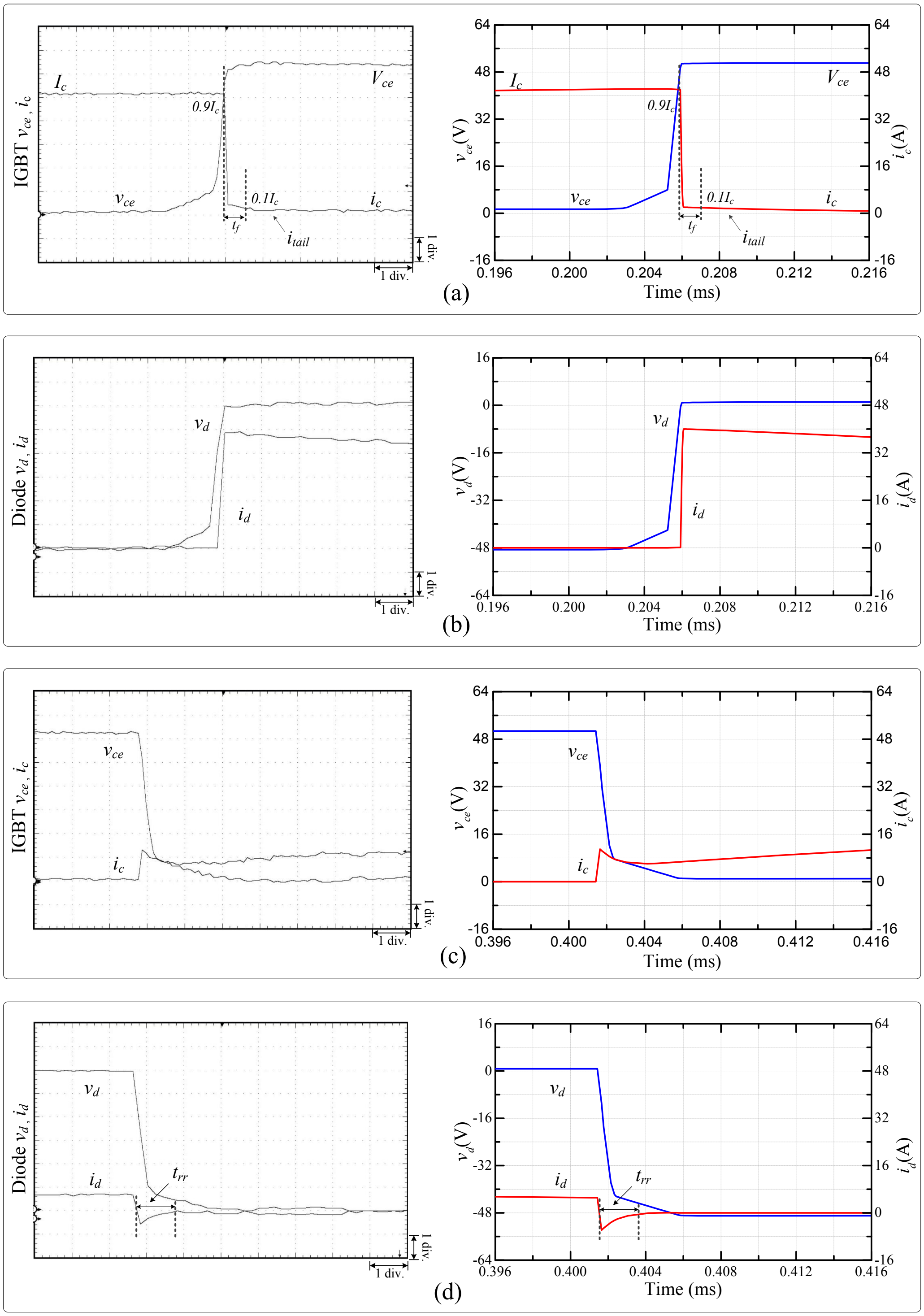


Figure 4.7: Chopper circuit hardware emulation transient results in oscilloscope (left side) and off-line simulation results in SaberRD® (right side) of (a) and (c) IGBT collector-emitter voltage and collector current, (b) and (d) Diode voltage and current. Scale: (a) - (d) y-axis: 8 V/div., 8 A/div.; x-axis: 50 μs.

diode voltage v_d, diode current i_d and their instantaneous power dissipation P_{IGBT} and P_{Diode}. Table 4.9 provides the comparison of transient times. Generally, based on the comparison results from the table and the figures, under the switching frequency of 2.5kHz, all the oscilloscope waveforms show great consistency with the simulation results from SaberRD®. Some errors are expected and acceptable as they are mainly due to the low output precision (16-bit fixed-point number) and different numerical solution approaches compared to SaberRD®. Some small amplitude differences from the figures are caused by the comparatively lower precession and sample rates from the oscilloscope. Thus, the validation process is completed and the device-level hardware model can provide accurate IGBT and reverse recovery diode behaviors under both steady-state and transient conditions. Then, the purposed simplified reverse recovery diode model can be applied for future power converter studies and hardware emulations such as an anti-parallel diode connected to IGBT in the later half-bridge circuit.

Table 4.9: Comparison of Switching Time of IGBT and diode under a Switching Frequency of 2.5kHz

	SaberRD®	**FPGA**	**Error**
t_r (IGBT)	1.6 μs	1.5 μs	6.7%
t_{rr} (Diode)	2050 ns	2000 ns	2.5%

4.3 Half-Bridge Circuit

Fig. 4.8 shows another the circuit for simulating the transient process as well as the steady-state of the IGBT and simplified diode hardware models. These two models are formed as a pair where the IGBT is connected with diode anti-parallelly so that their combined functionality could be tested. In this circuit, there are two switches, $S1$ and $S2$ (each containing an IGBT and a diode), 4 linear components and 4 voltage sources. The parameters of these components are given in Table 4.10 and the rest parameters for both reverse recovery diodes and IGBTs devices are the same as the one in the single diode (reverse recovery part) and IGBT test circuit obtained from SaberRD® referred to Table 4.1 and Table 4.4. In the diode model, the forward recovery phenomenon and the junction capacitance are omitted since the flywheel diode which is a commonly used as the anti-parallel diode to the IGBT has fast recovery feature and the reverse recovery is more significant. Additionally from the energy aspect, comparing to reverse recovery, forward recovery is faster and usually takes less time which indicates that the forward recovery energy consumption is much lower and can be neglected. Thus, the simplified diode model purposed in the previous chapter shown in Fig. 3.27(b) is used in this test circuit.

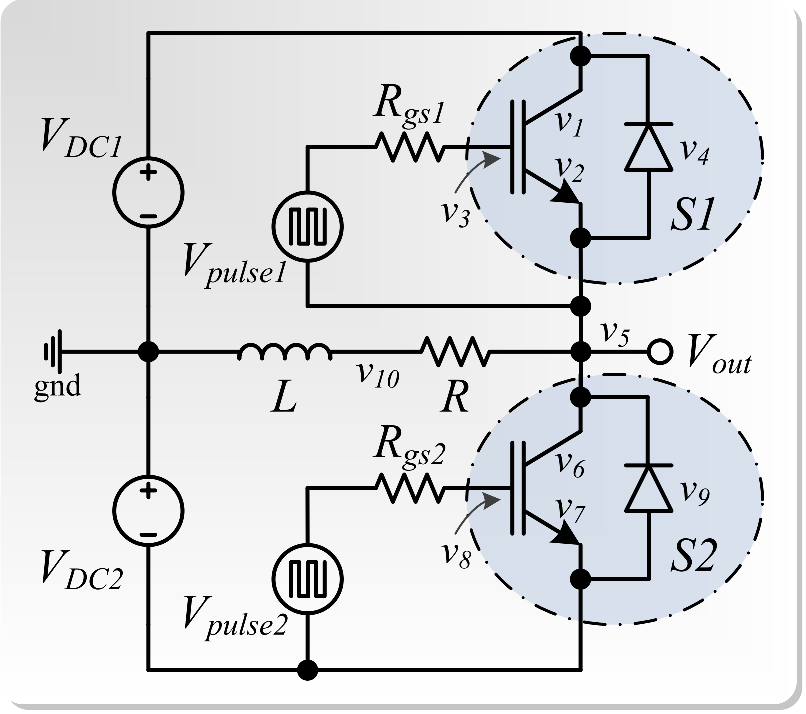


Figure 4.8: Half-bridge converter test circuit.

Table 4.10: Half-Bridge Test Circuit and Device Parameters

Test Circuit Parameters
$V_{DC1} = V_{DC2} = 100V$, $R = 1\Omega$, $L = 100\mu H$, $R_{gs1} = R_{gs2} = 35\Omega$

4.3.1 Circuit Description

Both of the gate voltages for the two IGBTs are square waves controlled by PWM with a duty cycle of 0.45 which indicates that the dead time of the switches is contained in and accounted for the test circuit and the circuit is practical; otherwise, there is a risk when both switches are turned on at the same time instantaneously. v_1, v_2 and v_3 are the internal IGBT node voltages and v_4 is the internal diode node voltage for $S1$. Similarly, for $S2$, v_6, v_7, v_8 and v_9 are the internal node voltages as well. The other passive linear components such as the inductance load L, load resistance load R, and the two gate resistors R_{gs1}, R_{gs2}, can be discretized as described in previous chapter. Finally, with the reduction of the foregone nodes and referring to (3.135), (3.136), and (3.137), the entire system can be discretized and linearized as a matrix equation shown in (4.14).

$$\begin{bmatrix} \mathbf{G}^{pair1}(2,2) & \mathbf{G}^{pair1}(2,3) & \mathbf{G}^{pair1}(2,4) & \mathbf{G}^{pair1}(2,5) & \mathbf{G}^{pair1}(2,6) & 0 & 0 & 0 & 0 & 0 \\ \mathbf{G}^{pair1}(3,2) & \mathbf{G}^{pair1}(3,3) & \mathbf{G}^{pair1}(3,4) & \mathbf{G}^{pair1}(3,5) & \mathbf{G}^{pair1}(3,6) & 0 & 0 & 0 & 0 & 0 \\ \mathbf{G}^{pair1}(4,2) & \mathbf{G}^{IGBT1}(4,3) & \mathbf{G}^{IGBT1}(4,4) & \mathbf{G}^{pair1}(4,5) & \mathbf{G}^{pair1}(4,6) & 0 & 0 & 0 & 0 & 0 \\ \mathbf{G}^{pair1}(5,2) & \mathbf{G}^{pair2}(5,3) & \mathbf{G}^{pair2}(5,4) & \mathbf{G}^{pair2}(5,5) & \mathbf{G}^{pair1}(5,6) & 0 & 0 & 0 & 0 & 0 \\ \mathbf{G}^{pair1}(6,2) & \mathbf{G}^{pair2}(6,3) & \mathbf{G}^{pair2}(6,4) & \mathbf{G}^{pair2}(6,5) & \mathbf{G}^{pair1}(6,6)+\mathbf{G}^{pair2}(1,1)+\frac{1}{R_load} & \mathbf{G}^{pair2}(1,2) & \mathbf{G}^{pair2}(1,3) & \mathbf{G}^{pair2}(1,4) & \mathbf{G}^{pair2}(1,5) & -\frac{1}{R_load} \\ 0 & 0 & 0 & 0 & \mathbf{G}^{pair2}(2,1) & \mathbf{G}^{pair2}(2,2) & \mathbf{G}^{pair2}(2,3) & \mathbf{G}^{pair2}(2,4) & \mathbf{G}^{pair2}(2,5) & 0 \\ 0 & 0 & 0 & 0 & \mathbf{G}^{pair2}(3,1) & \mathbf{G}^{pair2}(3,2) & \mathbf{G}^{pair2}(3,3) & \mathbf{G}^{pair2}(3,4) & \mathbf{G}^{pair2}(2,5) & 0 \\ 0 & 0 & 0 & 0 & \mathbf{G}^{pair2}(4,1) & \mathbf{G}^{pair2}(4,2) & \mathbf{G}^{pair2}(4,3) & \mathbf{G}^{pair2}(4,4) & \mathbf{G}^{pair2}(4,5) & 0 \\ 0 & 0 & 0 & 0 & \mathbf{G}^{pair2}(5,1) & \mathbf{G}^{pair2}(5,2) & \mathbf{G}^{pair2}(5,3) & \mathbf{G}^{pair2}(5,4) & \mathbf{G}^{pair2}(5,5) & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{R_load} & 0 & 0 & 0 & 0 & \frac{1}{R_load}+G_{Lload} \end{bmatrix}$$

$$\cdot \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ v_8 \\ v_9 \\ v_{10} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{eq}^{pair1}(2) - V_{DC}1 \cdot \mathbf{G}^{pair1}(2,1) \\ \mathbf{I}_{eq}^{pair1}(3) - V_{DC}1 \cdot \mathbf{G}^{pair1}(3,1) \\ \mathbf{I}_{eq}^{pair1}(4) - V_{DC}1 \cdot \mathbf{G}^{pair1}(4,1) \\ \mathbf{I}_{eq}^{pair1}(5) - V_{DC}1 \cdot \mathbf{G}^{pair1}(5,1) \\ \begin{matrix} \mathbf{I}_{eq}^{pair1}(6) + \mathbf{I}_{eq}^{pair2}(1) \\ -V_{DC}1 \cdot \mathbf{G}^{pair1}(6,1) + V_{DC}2 \cdot \mathbf{G}^{pair1}(1,6) \end{matrix} \\ \mathbf{I}_{eq}^{pair2}(2) + V_{DC}2 \cdot \mathbf{G}^{pair1}(2,6) \\ \mathbf{I}_{eq}^{pair2}(3) + V_{DC}2 \cdot \mathbf{G}^{pair1}(3,6) \\ \mathbf{I}_{eq}^{pair2}(4) + V_{DC}2 \cdot \mathbf{G}^{pair1}(4,6) \\ \mathbf{I}_{eq}^{pair2}(5) + V_{DC}2 \cdot \mathbf{G}^{pair1}(5,6) \\ -I_{Lloadeq} \end{bmatrix} \qquad (4.14)$$

After applying the TLM decoupling technique, the test circuit in Fig. 4.8 can be treated as TLM link and stub hybrid equivalent circuit shown in Fig. 4.9. Then, the entire system is divided into 5 parts which can be solved separately so as to achieve parallel computation. In this way, instead of solving the 10×10 dimensional linear system of equations as stated in (4.14), two 2×2 and two 4×4 dimensional linear system of equations formed by diodes and IGBTs with TLM-link circuits from switch S1 and S2 correspondingly referring to the equations from (3.138) to (3.143), as well as a 2×2 linear system of equations shown in (4.15) from the main circuit with TLM-stub module based on KCL and KVL analysis need to be solved. Z_{S1} and Z_{S2} could be the same and noted by Z_S indicating the switch line surge impedance. i_1 and i_2 are the upper and under parts circulating currents of the main circuit. For using TLM method, in order to obtain accurate simulation results, the time-step needs to be small. In this case study, the TLM method is tested using MATLAB® based on the half-bridge circuit as a prelude and a preparation for the later hardware implementation of MMC.

$$\begin{bmatrix} Z_S + Z_L + Z_R & -Z_L - Z_R \\ -Z_L - Z_R & Z_S + Z_L + Z_R \end{bmatrix} \cdot \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} = \begin{bmatrix} V_{DC1} - 2v^i_{S1} - 2v^i_L \\ V_{DC2} - 2v^i_{S2} + 2v^i_L \end{bmatrix}, \tag{4.15}$$

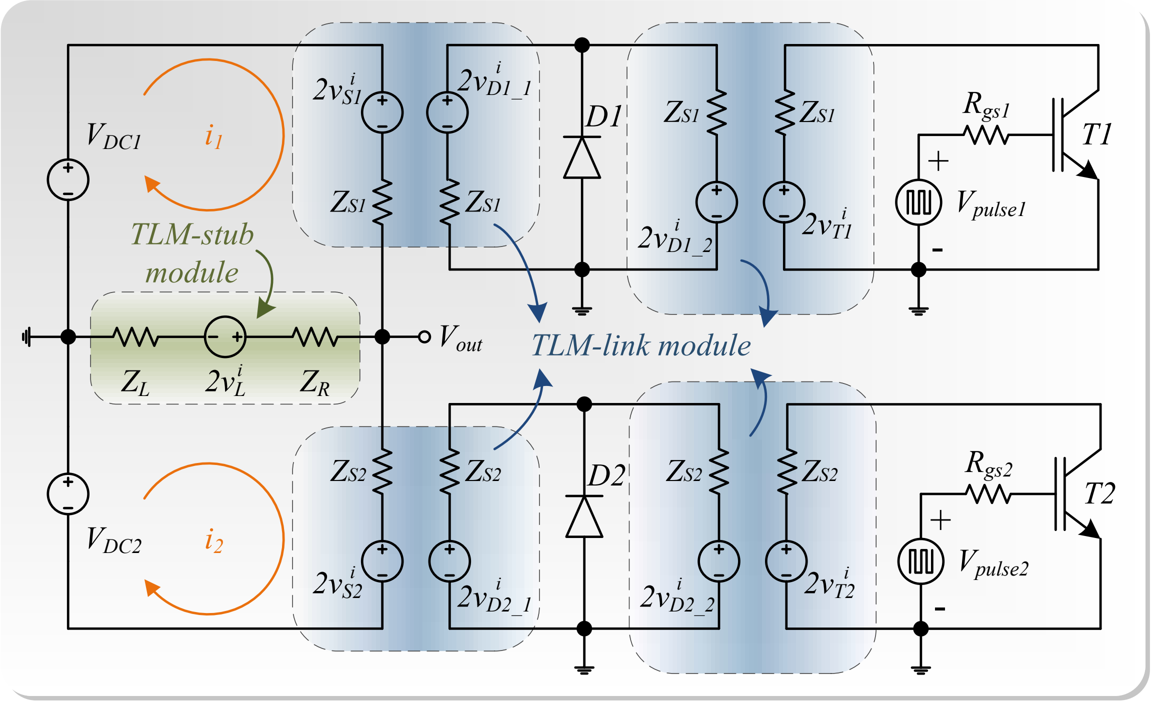


Figure 4.9: Half-bridge converter with TLM hybrid equivalent circuit.

4.3.2 Hardware Resources

The Xilinx® Virtex®-7 VC707 XC7VX485T FPGA board is used to emulate this test circuit shown in Fig. 4.8, and the resource utilization is listed in Table 4.11.

Table 4.11: Test Circuit Hardware Resources Utilizations

Resources	Overall Half-bridge Circuit
LUT	156696 (51.61%)
$LUTRAM$	2879 (2.21%)
FF	98416 (16.21%)
$BRAM$	22 (2.14%)
$DSP48$	1787 (63.83%)
I/O	388 (55.43%)
$BUFG$	1 (3.12%)

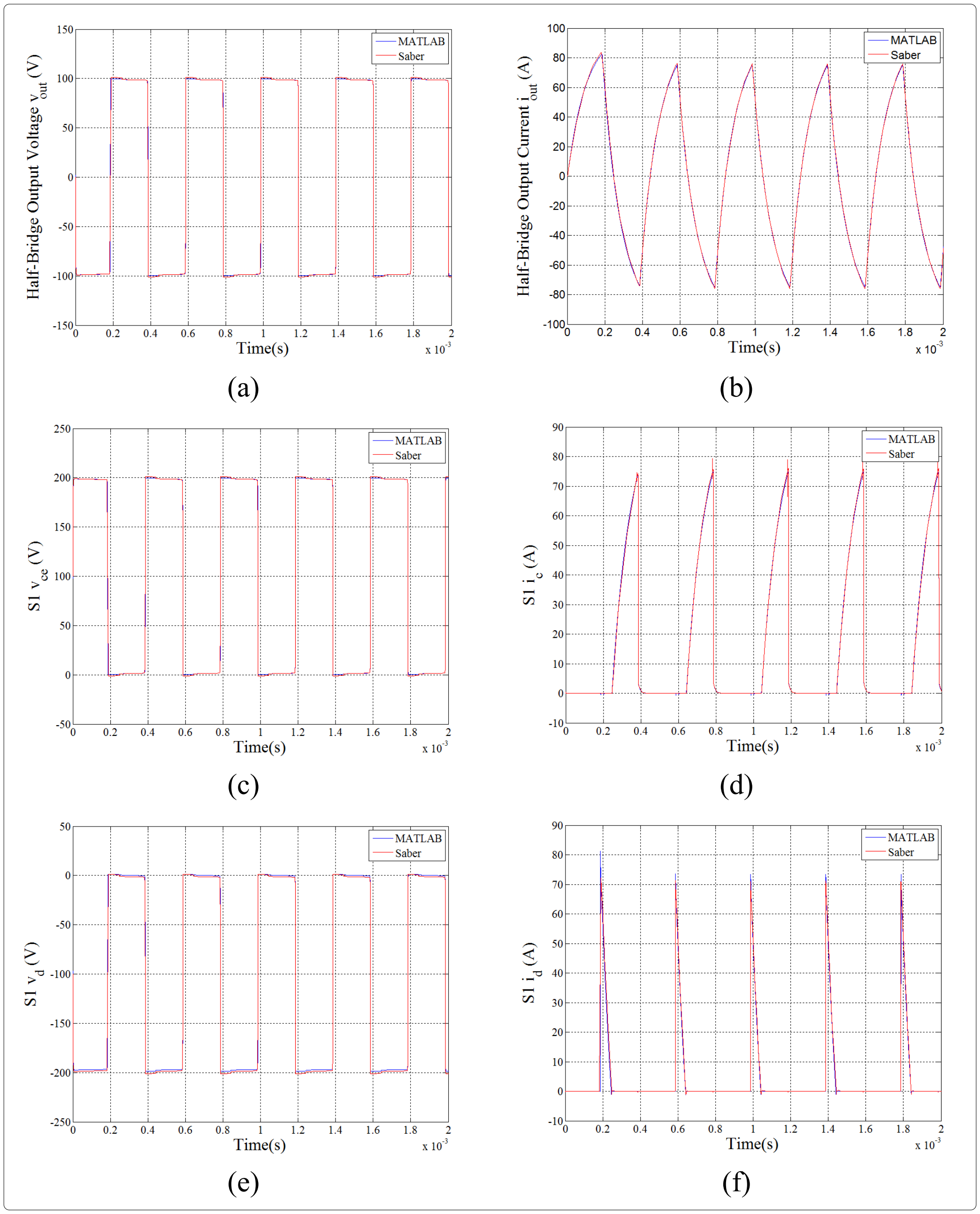
4.3.3 Results and Comparisons

4.3.3.1 TLM Method Verification

The simulation results from MATLAB® based on the Fig. 4.9 are compared to SaberRD®. With TLM method applied to the circuit, the time-step is set to 100ns for processing the MATLAB® simulation. After fine-tuning the line surge impedance for the TLM link model and proper adjusting the delays, the simulation results can be accurately generated. Fig. 4.10(a) — (f) shows the half-bridge output voltage, output current, the voltage and current of IGBT and diode for the switch $S1$ under steady-state condition. The transient waveforms during diode and IGBT on/off are shown in Fig. 4.11. Also, the instantaneous power dissipation of IGBT in $S1$ is given in Fig. 4.12. As can be seen from both steady-state and transient waveforms as well as the instantaneous power dissipation of IGBT, the results are basically matched with each other. Similarly, for switch $S2$, the MATLAB® results are agreed with SaberRD® as well and those comparison results are not necessary to illustrate again. Thus, the TLM decoupling method is verified and can be applied to the later MMC hardware implementation.

4.3.3.2 Hardware Emulation

Hardware emulation is processed using the original models without the TLM since larger simulation step-size can be applied. The simulation time-step is chosen to be 200 ns and the FPGA clock frequency of this design is set to 100 MHz. Switching frequency for both $S1$ and $S2$ are 2.5 kHz. The computational latency of the circuit hardware emulation for a full iteration from initial inputs to final outputs within a single time-step is 496 clock cycles. With proper output control, the hardware emulation results are about 50 times slower



(a)

(b)

(c)

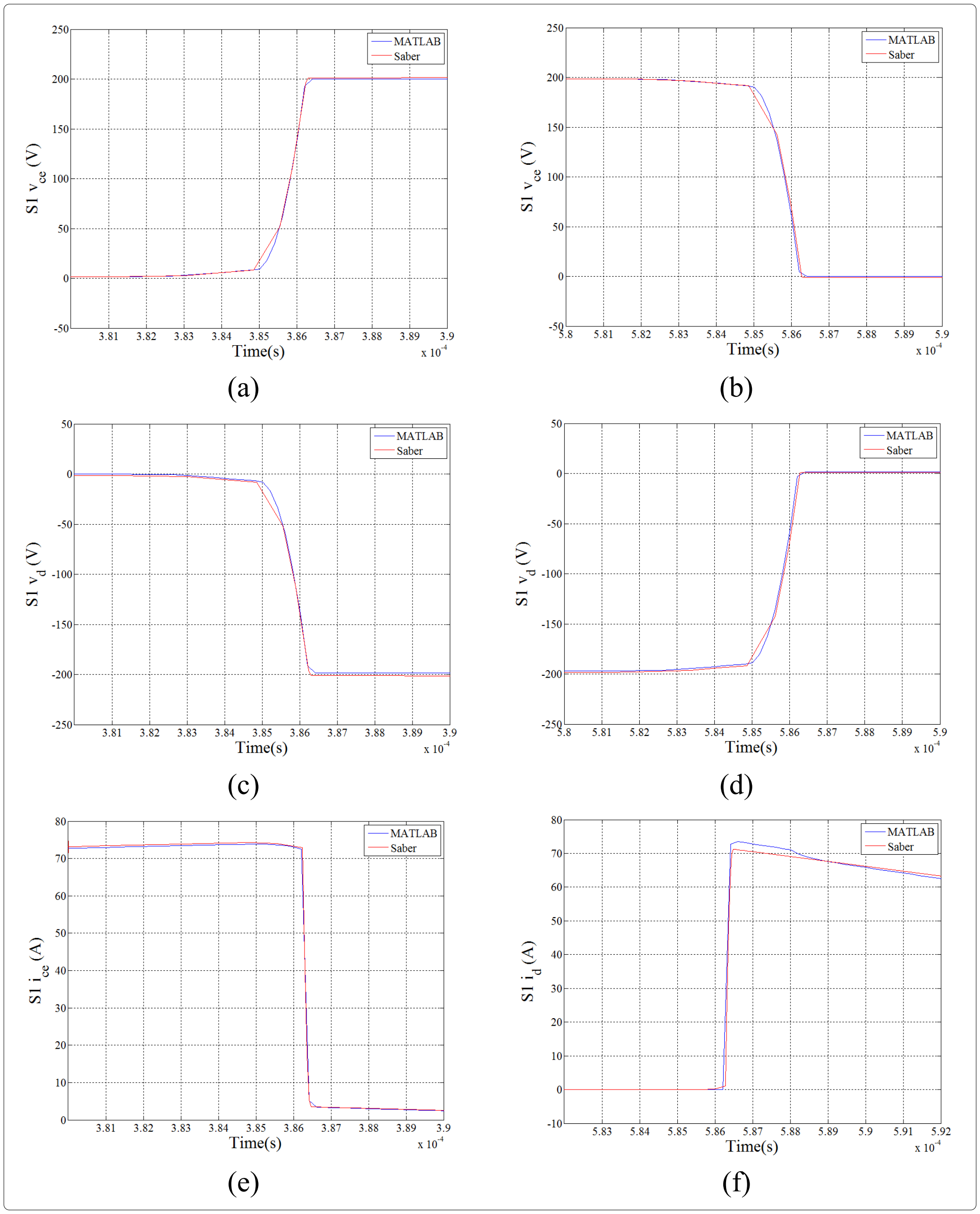
(d)

(e)

(f)

Figure 4.10: Steady-state results comparison (a) Output voltage and current waveforms of half-bridge circuit, (c-f) voltage and current of IGBT and diode for switch ($S1$) from MATLAB® and SaberRD®.

than the real-time. For example, to simulate the test circuit for 2 ms (5-period duration), it would actually take 100 ms on board. However, the executive time of SaberRD® simulation is about 250 ms using its built-in variable time scheme which indicates simulating this hardware model on FPGA board is 2.5 times faster than executing it in the off-line SaberRD® software. The output voltage v_{out} and current i_{out} of the half-bridge converter is shown in Fig. 4.13. The steady-state results of the IGBT collector-emitter voltages v_{ce},



(a)

(b)

(c)

(d)

(e)

(f)

Figure 4.11: Transient results comparison for the devices in $S1$ of half-bridge circuit from MATLAB® and SaberRD®.

collector currents i_c, the diode voltages v_d, and the diode currents i_d in both $S1$ and $S2$ of the half-bridge circuit are shown in Figs. 4.14. Diode power is too small compared to IGBT so the illustration is not necessary here. The instantaneous power dissipation mainly contributed by IGBTs P_{IGBT} for each two switches are shown in Fig. 4.15(a) — (b) for $S1$ and $S2$ respectively. The transient results of device-level IGBTs (v_{ce}, i_c), and diodes (v_d, i_d) for each switches $S1$ and $S2$ during turn-on and turn-off are shown in Fig. 4.16 and

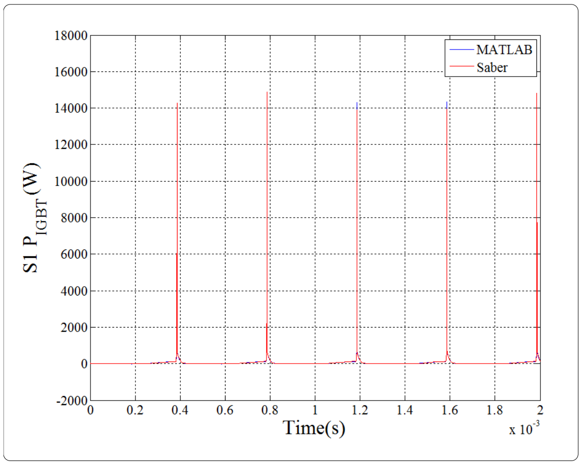


Figure 4.12: Comparison of IGBT instantaneous power dissipation waveforms in half-bridge circuit S1 from MATLAB® and SaberRD®.

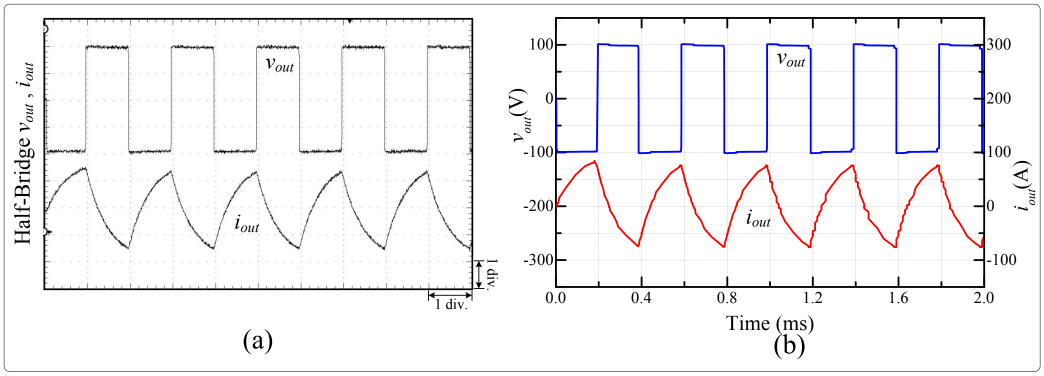


Figure 4.13: Steady-state output voltage and current waveforms of half-bridge circuit from (a) hardware emulation results in oscilloscope and (b) off-line simulation results in SaberRD®. Scale: y-axis: 50 V/div. (v_{out}), 50 A/div. (i_{out}); x-axis: 10 ms.

Fig. 4.17. Meanwhile, the diode reverse recovery phenomena in each switch are indicated in Fig. 4.18. Table 4.12 provides the comparison of average power dissipation for IGBT during switching on/off P_{swit} and conduction P_{cond}. Some errors are expected and acceptable mainly due to the low output precision (16-bit fixed-point number) and different numerical solution approaches compared to SaberRD®.

From all the waveforms in Fig. 4.14 to Fig. 4.18, device-level half-bridge hardware emulation results in oscilloscope (left side) and off-line simulation results in SaberRD® (right side) are fully and comprehensively compared with each others such as the voltages and currents in steady-state and transient-state mode as well as the instantaneous power dissipation. According to the comparison results, with the switching frequency of 2.5kHz, all the oscilloscope waveforms are basically consistent with the simulation results from SaberRD®. Some small amplitude differences are caused by the comparatively lower precession and sample rates from the oscilloscope. Thus, the validation process is completed

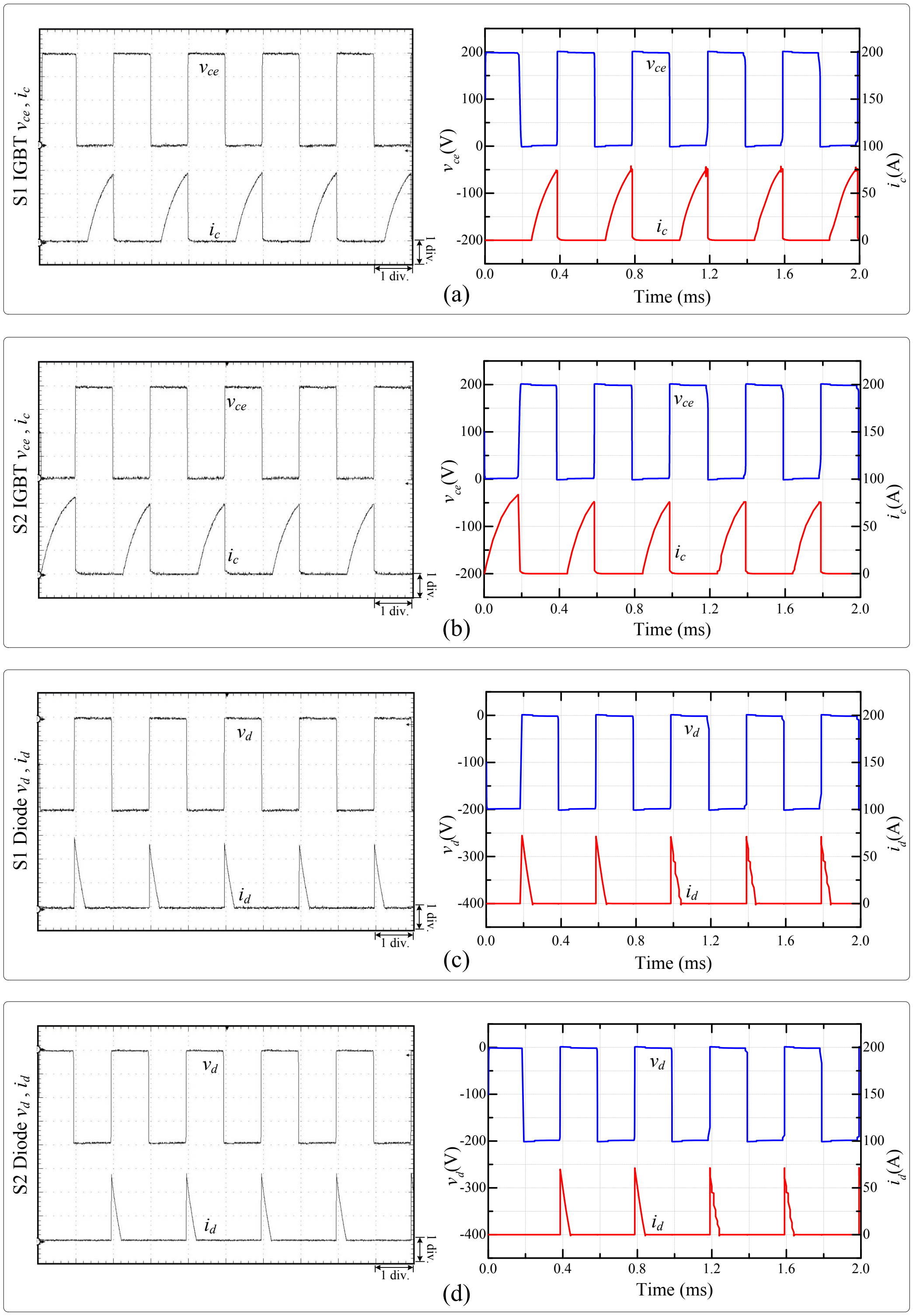


Figure 4.14: Steady-state results for the devices of half-bridge circuit from hardware emulation (oscilloscope) and off-line SaberRD® software simulation. Scale: (a)-(d) y-axis: 50 V/div.(v_{ce}, v_d), 25 A/div.(i_c, i_d); (a)-(d) x-axis: 10 ms.

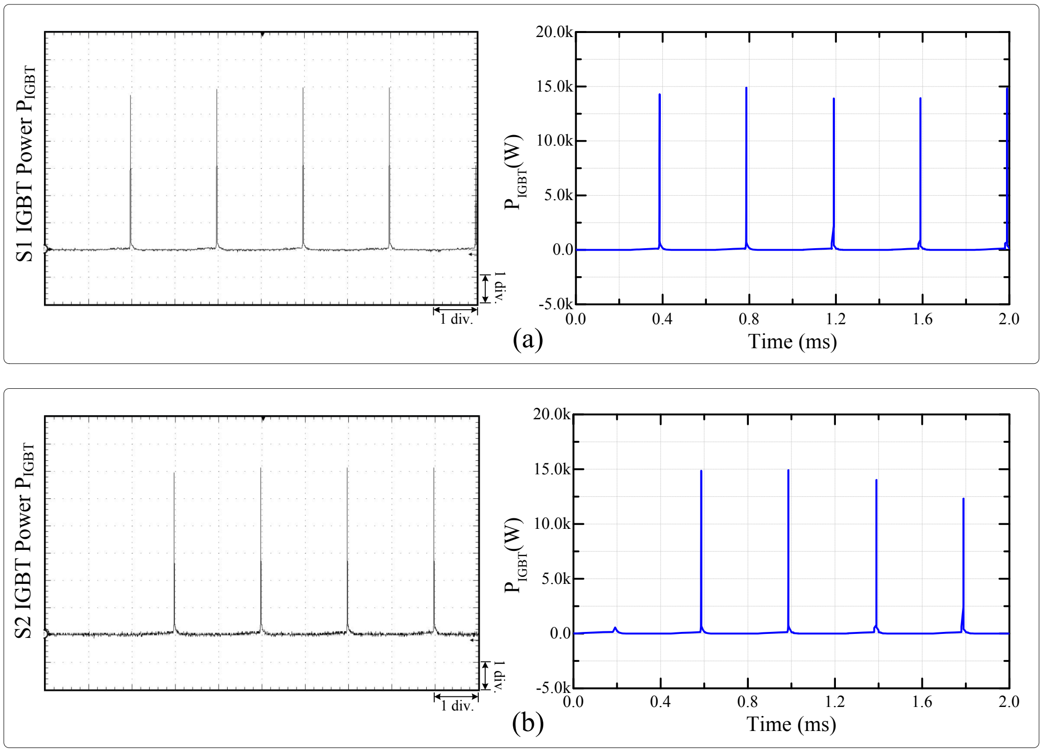


Figure 4.15: Steady-state waveforms of instantaneous power dissipation from (a) S1 IGBT and (b) S2 IGBT in half-bridge circuit hardware emulation results in oscilloscope (left side) and off-line simulation results in SaberRD® (right side). Scale: (a)-(b) y-axis: 2.5k W/div. (P_{IGBT}); x-axis: 10 ms.

and the device-level hardware model can provide accurate behaviors under both steady-state and transient conditions. Furthermore, the half-bridge hardware model also works at a relatively high switching frequency of 25kHz.

Table 4.12: Comparison of IGBT Power Dissipation under a Switching Frequency of 2.5kHz

	SaberRD®	**FPGA**	**Error**
$S1\ P_{swit}$	865.1W	834.5W	3.6%
$S1\ P_{cond}$	62.6W	60.5W	3.8%
$S2\ P_{swit}$	1025.2W	993.8W	3.2%
$S2\ P_{cond}$	70.1W	67.6W	3.8%

4.4 Five-Level MMC

Nowadays, MMC is the most common type of VSC (voltage-source converter) for the current HVDC system. Fig. 4.19 shows the circuit configuration of a 3 phase 5-level MMC which has 4 submodules in both upper and lower bridge of each phase. Every submod-

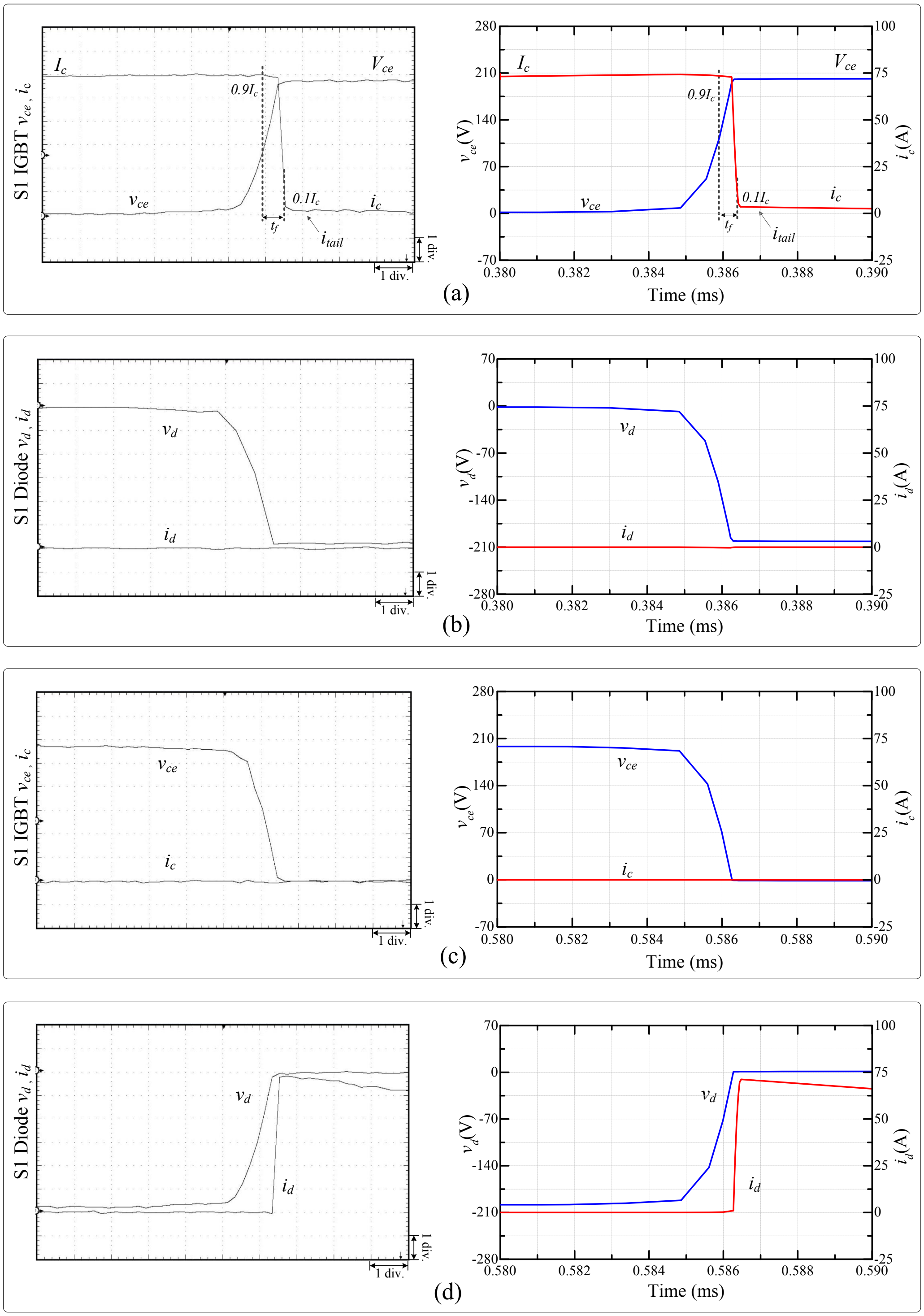


Figure 4.16: Transient results for the devices of half-bridge circuit S1 from hardware emulation (oscilloscope) and off-line SaberRD® software simulation. Scale: (a)-(d) y-axis: 70 V/div. (v_{ce}, v_d), 12.5 A/div. (i_c, (i_d); (a)-(d) x-axis: 0.05 ms.

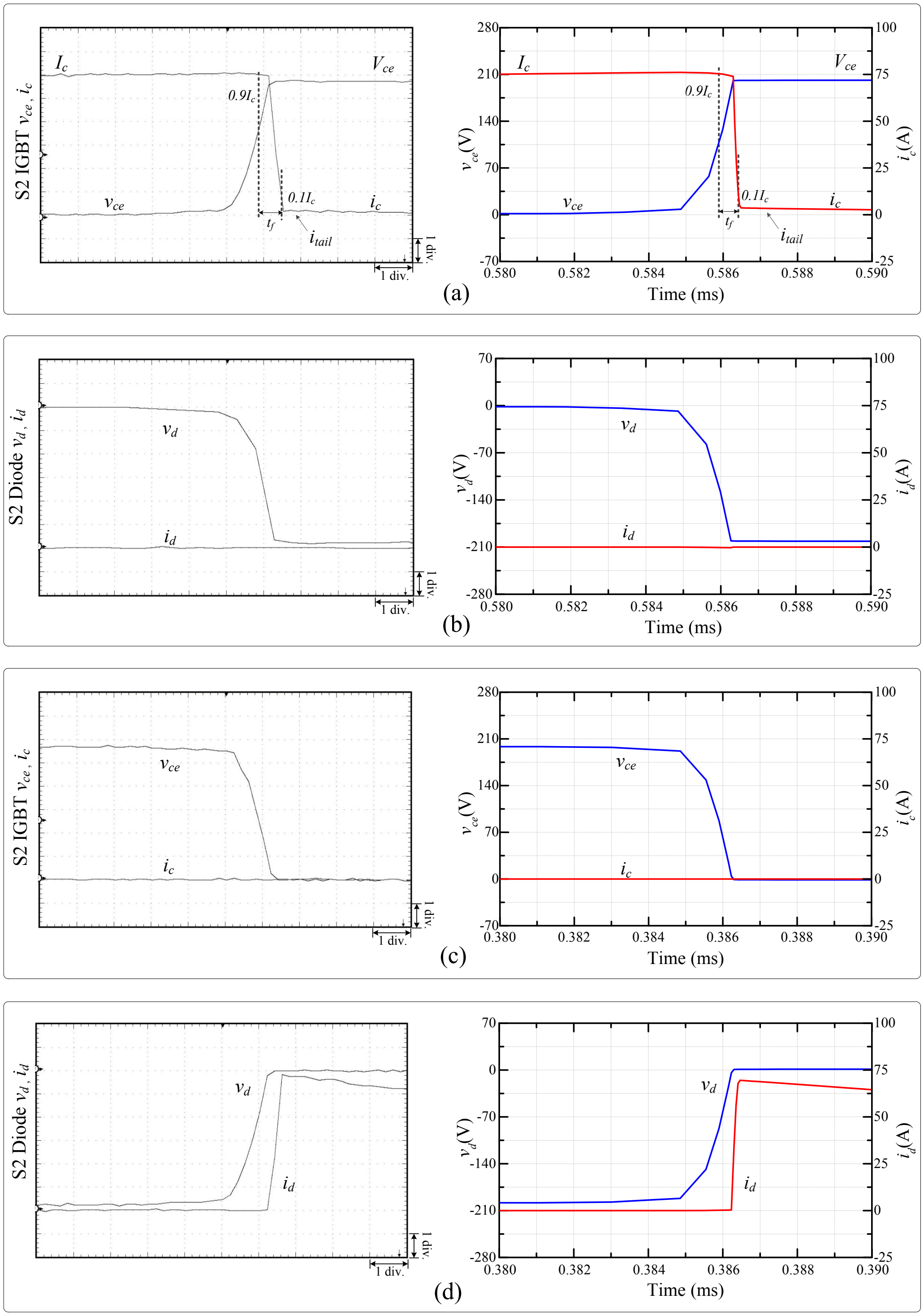


Figure 4.17: Transient results for the devices of half-bridge circuit S2 from hardware emulation (oscilloscope) and off-line SaberRD® software simulation. Scale: (a)-(d) y-axis: 70 V/div. (v_{ce}, v_d), 12.5 A/div. (i_c, (i_d); (a)-(d) x-axis: 0.05 ms.

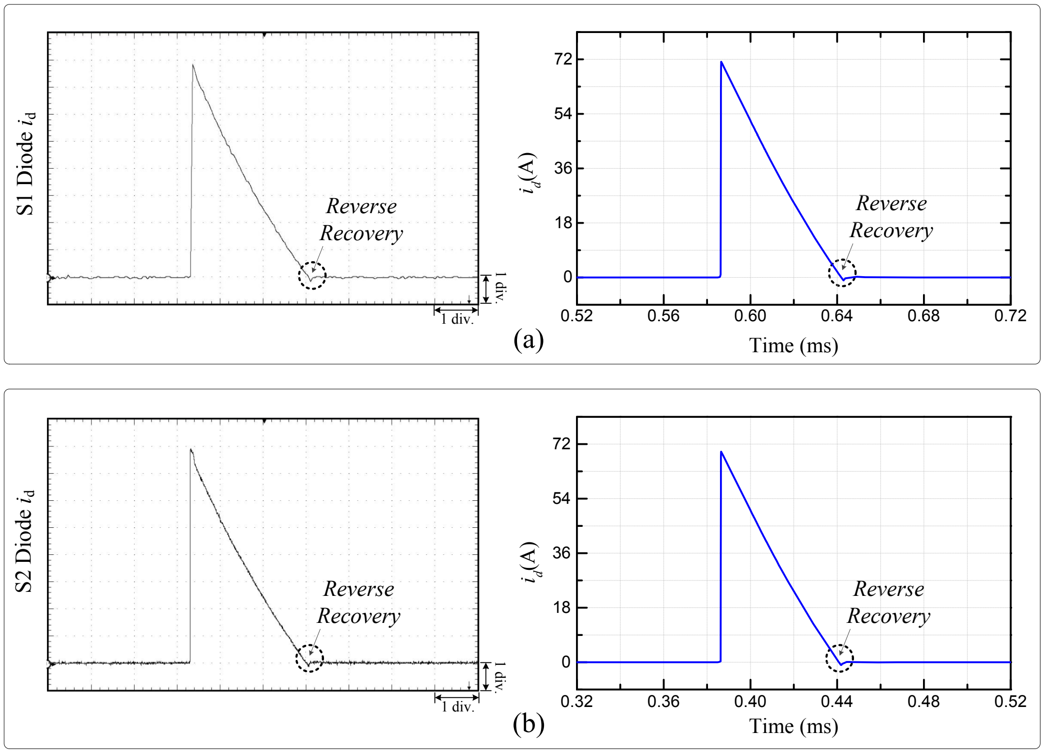


Figure 4.18: Transient results of the diodes current i_d in half-bridge circuit from hardware emulation oscilloscope waveforms and off-line SaberRD® software simulation waveforms. Scale: y-axis: 9 A/div.; x-axis: 1 ms.

ule consists of one DC capacitor and two switches as interpreted in Ch. 3 Fig. 3.31. In the circuit, IGBT and anti-parallel diode contained inside of the switches are the nonlinear elements; the other elements, capacitors, inductors and their parasitic resistors can be deemed as linear elements. Simulating the MMC using off-line software such as SaberRD® and PSpice® usually take much longer when the number of switching elements in each leg increases since they run on CPU which is basically sequential processing. When detailed device-level IGBT and diode models which contain many nonlinearities are adopted in the MMC circuit, the executive time is obviously very long. For a single-phase five-level MMC, there would be 80 nodes in the nonlinear circuit system. After linearization and discretization, an 80×80 dimensional large linear system of equations need to be solved. Thus, it would be very time-consuming. Meanwhile, it is also impractical to build the corresponding large matrix solver for implementation or hardware emulation. In order to reduce the simulation time and avoid the large matrix formation, partitioning them into several relatively independent units using aforementioned TLM decoupling technique is necessary so that they can be handled separately and simultaneously by hardware. In addition, since the three legs with $R - L$ loads among the three-phase MMC circuit are identical with the only phase difference (120° between each two phases), they could be implemented separately and in parallel as a single phase with $R - L$ load circuit. That way,

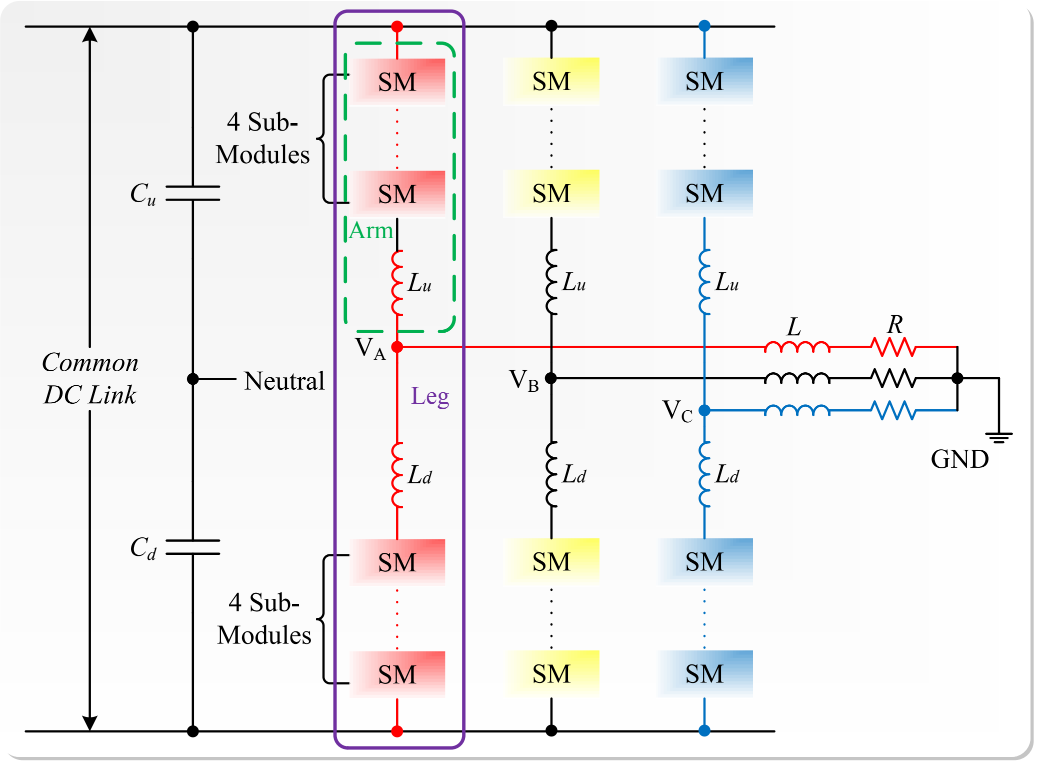


Figure 4.19: Configuration of a three phase 5 level MMC with R-L load.

the hardware parallelism superiority could be played out.

4.4.1 Circuit Description

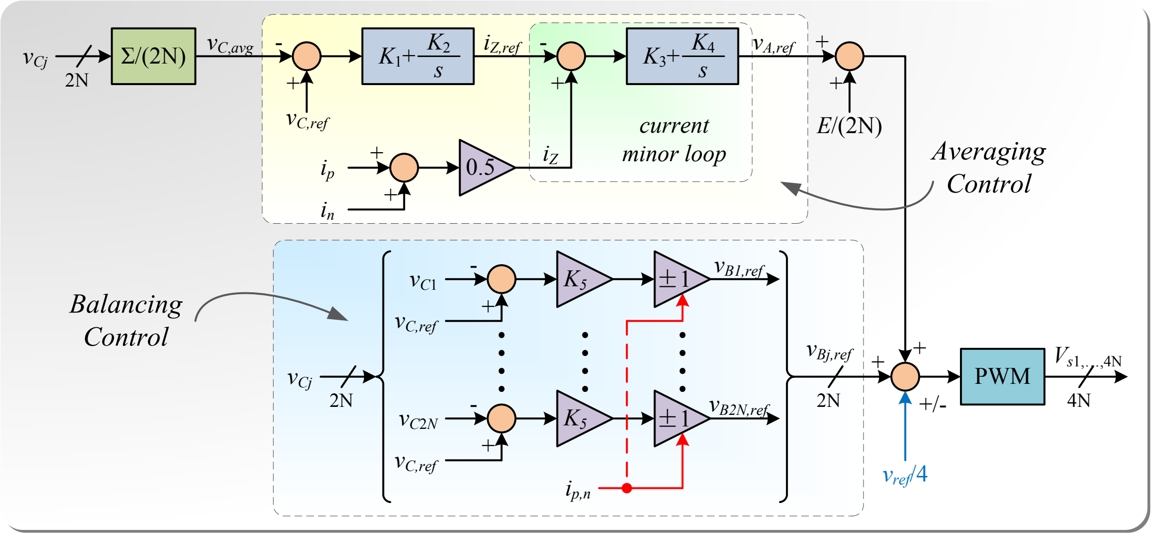


Figure 4.20: MMC DC capacitor voltage control strategy.

Specifically, in this test case, a single-phase (one leg) circuit including 16 IGBTs with

Table 4.13: MMC Test Circuit Parameters

Single Phase MMC Test Circuit Parameters
$E = 900V$, $R = 5\Omega$, $L = 6mH$, $L_u = L_d = 3mH$, $C_j = 1.9mF$, $Z_s = 0.05\Omega$, $f = 50Hz$, $f_{sw} = 2.5kHz$

16 anti-parallel diodes is simulated to illustrate the hardware model functionality and its fidelity. Table 4.13 provides circuit parameters referring to (3.144) and the rest parameters for both reverse recovery diodes and IGBTs devices are the same as those used in the half-bridge circuit referred to Table 4.1 and Table 4.4. E is a supply common DC link voltage set to be 900 volts, then each arm has V_{DC} with 450 volts. The switching frequency f_{sw} is 2.5k Hz and AC output reference frequency is 50 Hz. Proper closed-loop control strategy [117] for the PWM voltage signal output to the switches is also applied in the MMC circuit. The detailed control diagram is illustrated in Fig. 4.20 where N is the number of submodule in each arm, $j = 1, 2, 3, ..., 2N$ means the numbering of each submodule. For a single-phase five-level MMC, N is equal to 4 which means each converter arm has four SMs. i_Z is defined as single-phase dc loop current, i_p and i_n indicate the positive and negative arm currents respectively. The sign of proportional coefficient inside the balancing control is decided by the polarity of inductor currents (i_p, i_n). v_{Cj} refers to DC capacitor voltage ($j = 1, 2, 3, 4$ is for the upper arm and $j = 5, 6, 7, 8$ is for lower arm correspondingly). A hardware control module, containing both averaging and balancing control parts, is generated by this control unit using Vivado® HLS with latency around 50 clock cycles. After the control unit, gate signals are obtained for each IGBT. As a result of introducing TLM decoupling technique, 16 sub-circuits (basically nonlinear power switch networks and each containing an IGBT sub-circuit and a diode sub-circuit) are formed and they can be run in parallel. As the IGBT is more sophisticated than the diode, the latency of IGBT sub-circuit would be the decisive factor for the entire latency of MMC circuit hardware simulation. The model equations and the matrix formation of each sub-unit circuits (IGBT and diode) as well as the main circuit based on mesh current and nodal analysis, are all described in previous Ch. 3. where $Z_{Cj}(j = 1, 2, \cdots)$ is the surge impedance of DC capacitor, Z_L, Z_{Lu} and Z_{Ld} are surge impedance of L, Lu and Ld respectively as shown in Fig. 3.31.

4.4.2 Hardware Implementation

The single-phase MMC model basically contain 8 sub-module (half-bridge topology). Based on the half-bridge converter stated before, the hardware resources utilization of this MMC test circuit is over 8 times more than the half-bridge and then definitely exceeds the limit of the Xilinx® Virtex®-7 XC7VX485T FPGA board especially the DSP48 usage. Since the model which includes lots of device-level IGBTs and diodes is too large and complicated to fit the board, the strategy is changed as choosing another platform in Vivado® and executing the simulation rather than debugging on the FPGA board.

The simulation time-step is chosen to be 50 ns and the FPGA clock frequency is set to 100 MHz. Theoretically, the computational latency of the circuit hardware emulation for a full iteration within a time-step from initial inputs to final outputs is 326 clock cycles and the hardware executive time of simulating the model for 100ms would be around 5s with proper output control which provides nearly 100 times faster than the off-line SaberRD® simulation. Therefore, it can be concluded and predicted as the number of sub-modules connected increases, the higher the advantages of hardware emulation if larger FPGA board could be utilized.

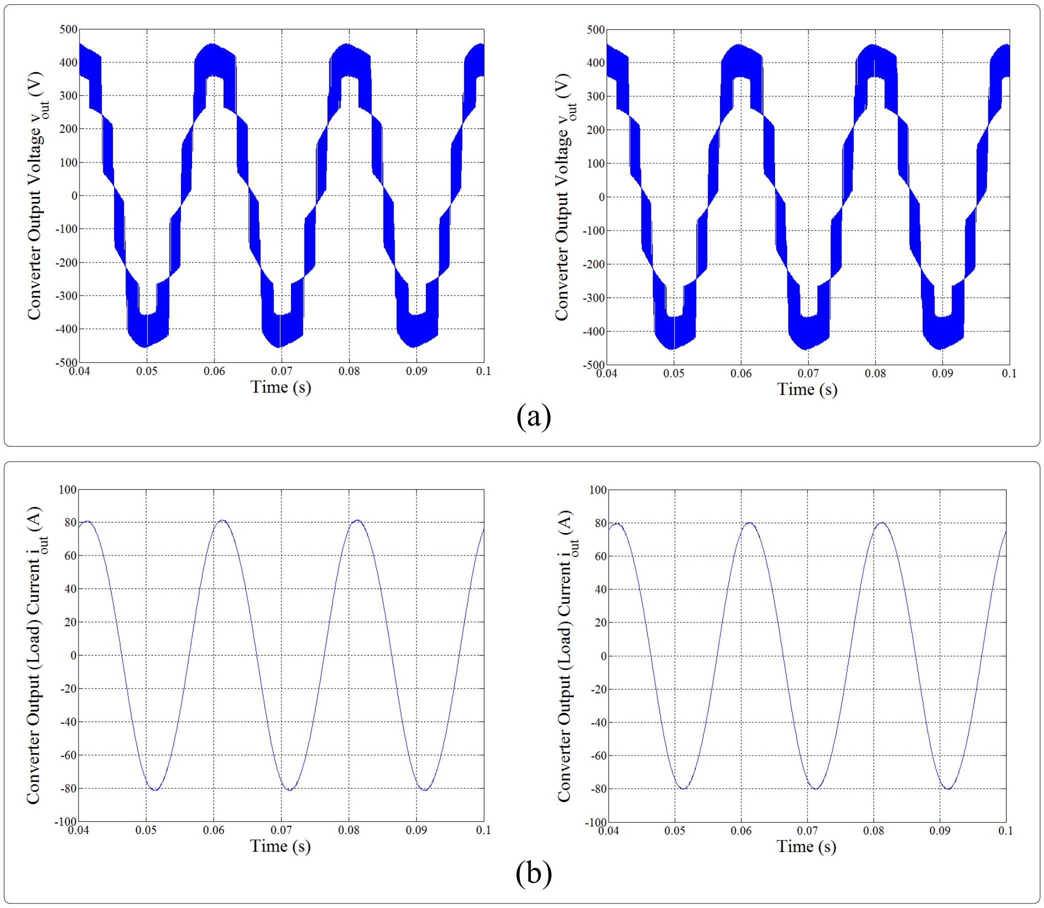


Figure 4.21: Single-phase 5-level MMC (a) output vltage and (b) load current waveforms from hardware simulation (left) and SaberRD® simulation (left).

4.4.3 Results and Comparisons

Fig. 4.21 presents the system-level results of the single-phase five-level MMC output voltage v_{out} and load current i_{out} from hardware Vivado® simulation and the SaberRD® software simulation. Fig. 4.22 shows waveforms of the DC capacitor voltages of all SMs in both positive and negative converter arms, and the converter arm currents i_p and i_n. As can be seen from Fig. 4.21, the results agree with each other where the output voltages are all five level with the peak value around 450 volts as expected and the load currents are nearly 50 Hz sine waves. In Fig. 4.22, the values of SMs DC voltage ripples in upper and lower arms for the 5-level converter fluctuate between 200 to 250 volts with reference of 225 volts. Meanwhile, the waveforms of the upper and lower arm currents (i_p, i_n) from hardware simulation are basically consistent with SaberRD®. Therefore, these can demonstrate the MMC control module and switch modules are correct and working properly. Furthermore, the hardware simulation reaches steady-state a little faster than SaberRD® and SaberRD® takes very long simulation time to generate the results (over several minutes). It indicates that when the system becomes large and complex, off-line simulators such as SaberRD® are difficult to execute the simulation and obtain the results as too many nodes contained. Nevertheless, due to the parallelism process and the proposed TLM decoupling technique introduced, simulation of large power electronic systems become possible in hardware on FPGA. Thus, even though the TLM decoupling strategy has several applicable restrictions, constraints and drawbacks [47,95], in system-level MMC simulation with fine-tuning the TLM parameters, it is proved as an efficient method.

4.5 Summary

This chapter demonstrated several device-level case studies such as the behavioral IGBT and diode simple test circuits, the chopper circuit, the half-bridge converter and a system level one phase of a 3-phase-symmetrical 5-level MMC circuit case study. All the cases were implemented in hardware using the detailed device-level behavioral IGBT and diode hardware models as well as the proposed algorithms such as the N-R iterations, matrix solutions, discretization or linearization methods and decoupling technique. Based on the comparison results with SaberRD®, these models are verified and have high accuracy. Some small differences are mainly caused by the numerical solutions, low precision (16-bits) fixed point outputs to the oscilloscope and the FPGA board noises. Meanwhile, the hardware emulation of the developed models shows higher efficiency and high fidelity according to the faster execution speed than SaberRD®. A single-phase five-level MMC circuit was effectively separated into one main circuit and 16 sub-circuits to realize parallel computation using TLM decoupling technique. The results reflected the superiority of proposed models and methods. Therefore, the hardware models could be arbitrarily used in any circuit topologies. Although it is still slower than real-time operation, if more refined numerical solutions or algorithms are found and new generation FPGAs with larger capacity and higher clock frequency are developed in the future, it is feasible to run the models in real-time.

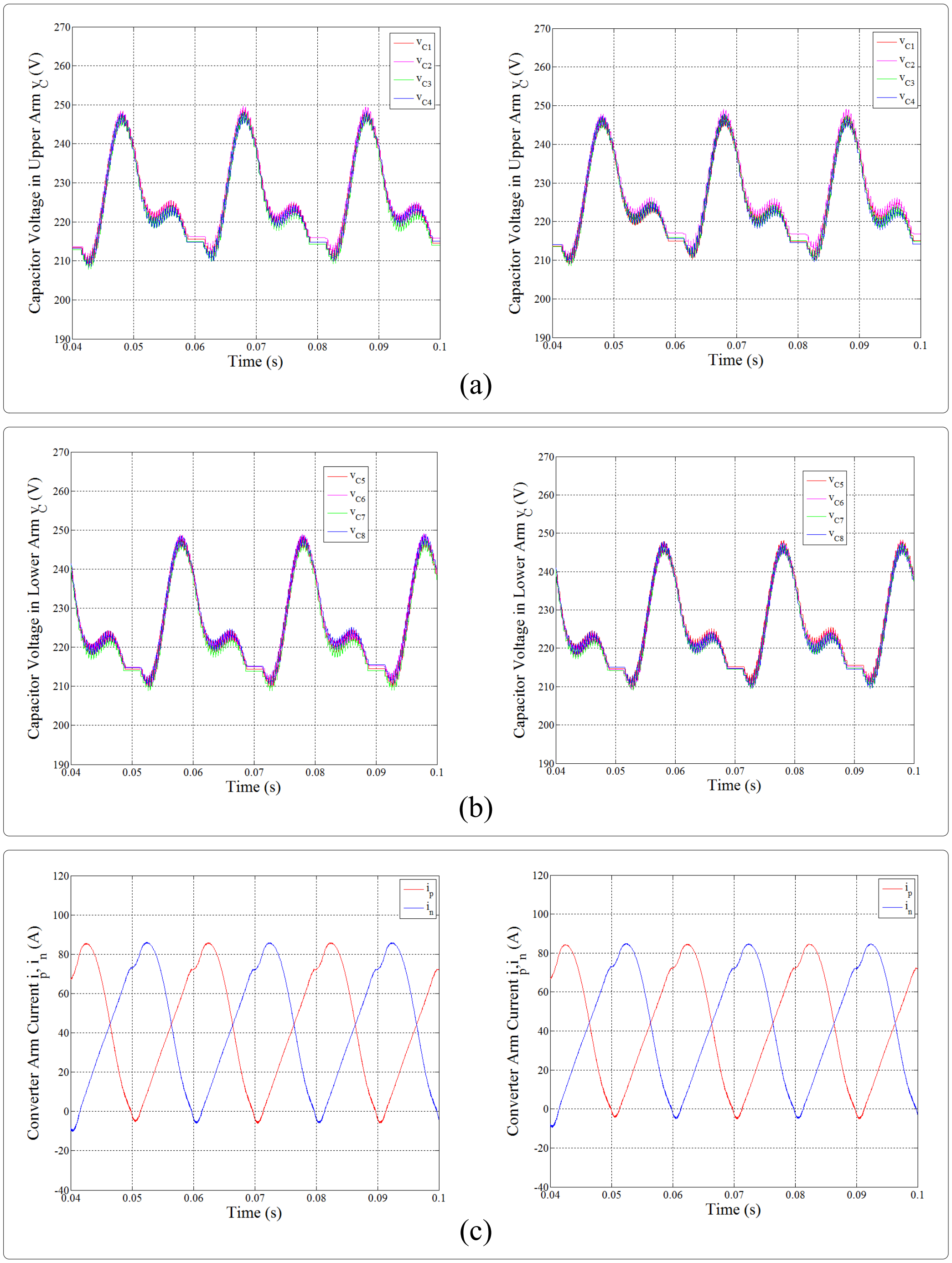


Figure 4.22: Single-phase 5-level MMC (a)-(b) DC voltage ripples of submodules in upper and lower arms, and (c) arm currents waveforms from hardware simulation (left) and SaberRD® simulation (left).

5 Conclusion and Future Work

Power electronic converters are widely used not only in current power systems but also in automotive and aerospace industries for controlling and converting electrical energy or power. Such semiconductor devices as diode and IGBT are the fundamental components of converters. Since it is not always possible to study and analyze those devices after being placed in system operation, hardware emulation is significant. As the HIL simulation and electromagnetic transient studies become more and more prevalent in power systems, FPGAs, due to the inherent high performance and efficiency, can be introduced to emulate the power electronic circuit design for realizing massive parallel process and fast computational speed.

In this thesis, detailed device-level behavioral models of power electronic systems for HIL simulation are developed and several case studies for the proposed models with different simulation strategies or algorithm are implemented. The rest of this chapter provides the contributions of this work and suggestion of future work.

5.1 Contributions of this Work

The main contributions of this work are summarized as follows:

(a) This work proposed hardware modeling and emulation of device-level behavioral IGBT and power diode Saber® based models as well as some model simplifications. Each of the models is designed in VHDL for parallel computation and implemented on the FPGAs. Both of the hardware models can be generally used in any different topologies of power electronic converter circuits which contain the corresponding components so as to reduce the future design work and save the time and cost of devices development before appearing on the market.

(b) Several algorithms and design methods are presented in this work such as the Backward Euler method, Newton-Raphson iteration and TLM decoupling technique so as to deal with and overcome the complicated nonlinearity and high coupling of the models. The TLM decoupling technique could be introduced to the simulation of large power system circuits such as MMC and HVDC which usually focus on the system-level behavior and characteristics in steady-state or DC mode.

(c) Different time-step schemes are implemented in hardware simulation. Besides the conventional fixed time-step scheme, a dynamic time-step scheme based on LTE referred to Saber® is developed.

(d) Depending on the various demands of the system sizes, such parallel linear solvers as Cramer's fast solvers, Gauss elimination with partial pivoting linear solver and Gauss-Jordan linear solver are designed in hardware for high accuracy and low latency purpose.

(e) Identical simulation cases of power electronic circuits are developed in Saber® to verify the proposed hardware emulation. Their results from steady-state and transient are consistent with each other. It also indicates that running the simulation in hardware (on FPGA) is faster than running the simulation in off-line software Saber®.

5.2 Suggestions for Future Work

The following suggestions can be proposed for the future work:

(a) Model simplification could be developed in future work. Since the fully detailed device-level behavioral IGBT and power diode models are very complicated and form relatively high-order linear system equations or matrices based on their internal nodes, albeit the high accuracy, they are time and resources consuming for hardware emulation. Thus, it is necessary and imperative to put more efforts on further simplification of both models while maintaining the same accurate level so as to reduce the latency and resource consumption.

(b) The Saber® based IGBT and power diode behavioral models described in this thesis do not consider the effects of temperature variation to the device electric parameters. In practical operations, the IGBT and power diode behaviors change as the devices temperatures rise and fall. For example, the operating temperature can be really high due to the high current on the switches, the heat dissipation from the sources and electric machines. Therefore, in the future work, the dynamic thermal part should be added in the models or other types of device level dynamic electro-thermal behavioral IGBT and power diode models could be developed and studied for the more accurate and more realistic power electronic circuit hardware emulation.

Bibliography

[1] A. R. Hefner and D. Blackburn, "An analytical model for the steady-state and transient characteristics of the power insulated-gate bipolar transistor," *Solid State Electron.,* vol. 31, no. 10, pp. 1513-1532, May 1988.

[2] A. R. Hefner, "A dynamic electro-thermal model for the IGBT," *IEEE Trans. Industry Applications,* vol.30, no.2, pp.394-405, Mar/Apr. 1994.

[3] A. R. Hefner and D. M. Diebolt, "An experimentally verified IGBT model implemented in the Saber circuit simulator," *IEEE Trans. Power Electron.,* vol.9, no.5, pp.532-542, Sept. 1994.

[4] A. R. Hefner, "Modeling buffer layer IGBTs for circuit simulation," *IEEE Trans. Power Electron.,* vol. 10, no. 2, pp. 111-123, Mar. 1995.

[5] R. Kraus and K. Hoffmann, "An analytical model of IGBTs with low emitter efficiency," *Power Semiconductor Devices and ICs, ISPSD '93., Proc. 5th Int. Symp.,* Monterey, CA, 1993, pp. 3034.

[6] J. Sigg, P. Turkes and R. Kraus, "Parameter extraction methodology and validation for an electro-thermal physics-based NPT IGBT model," *Industry Applications Conf. 3rd-2nd IAS Annu. Meeting, IAS '97., Conf. Rec. 1997 IEEE,* New Orleans, LA, 1997, pp. 1166-1173, vol.2.

[7] R. Kraus, P. Turkes and J.Sigg, "Physics-based models of power semiconductor devices for the circuit simulator SPICE," *Power Electronics Specialists Conf., PESC 98 Rec., 29th Annu. IEEE,* Fukuoka, 1998, pp. 1726-1731, vol.2.

[8] K. Sheng, B. W. Williams, and S. J. Finney, "A review of IGBT models," *IEEE Trans. Power Electron.,* vol. 15, no. 6, pp. 12501266, Nov. 2000.

[9] B.J. Baliga, "Analytical Modeling of IGBTs: Challenges and Solutions," *IEEE Trans. Electron Devices,* vol.60, no.2, pp.535-543, Feb. 2013.

[10] C. M. Tan and K. J. Tseng, "Using power diode models for circuit simulations-a comprehensive review," *IEEE Trans. Power Electron.,* vol. 46, no. 3, pp. 637-645, June 1999.

[11] Y. C. Liang and V. J. Gosbell, "Diode forward and reverse recovery model for power electronic SPICE simulations," *IEEE Trans. Power Electron.,* vol. 5, no. 3, pp. 346-356, July 1990.

[12] P. O. Lauritzen and C. L. Ma, "A simple diode model with reverse recovery," *IEEE Trans. Power Electron.,* vol. 6, pp. 188-191, Apr. 1991.

[13] C. L. Ma and P. O. Lauritzen, "A simple power diode model with forward and reverse recovery," *IEEE Trans. Power Electron.,* vol. 8, no. 4, pp. 342-346, Oct. 1993.

[14] C. L. Ma, P. O. Lauritzen and J. Sigg, "Modeling of power diodes with the lumped-charge modeling technique," *IEEE Trans. Power Electron.,* vol. 12, no. 3, pp. 398-405, May 1997.

[15] R. Kraus, K. Hoffmann and H. J. Mattausch, "A precise model for the transient characteristics of power diodes," *Power Electronics Specialists Conf., PESC '92 Rec., 23rd Annu. IEEE,* Toledo, 1992, pp. 863-869, vol.2.

[16] A. T. Yang, Yu Liu and J. T. Yao, "An efficient nonquasi-static diode model for circuit simulation," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.,* vol. 13, no. 2, pp. 231-239, Feb. 1994.

[17] K. J. Tseng, C. F. Foo, and P. R. Palmer, "Implementing power diode models in SPICE and Saber," *Power Electronics Specialists Conf., PESC '94 Rec., 25th Annu. IEEE,* Taipei, 1994, pp. 59-63, vol.1.

[18] K. J. Tseng and S. Pan, "Modified charge-control equation for more realistic simulation of power diode characteristics," *Proc. Power Conversion Conf.,* Nagaoka, 1997, pp. 439-444, vol.1.

[19] A. Courtay, "MAST Power Diode and Thyristor Models Including Automatic Parameter Extraction," *SABER User Group Metting.,* Brighton, UK, Sept. 1995.

[20] A. T. Bryant, P. R. Palmer, E. Santi and J. L. Hudgins, "A Compact Diode Model for the Simulation of Fast Power Diodes including the Effects of Avalanche and Carrier Lifetime Zoning," *2005 IEEE 36th Power Electronics Specialists Conf.,* Recife, 2005, pp. 2042-2048.

[21] Z. Wang, A. T. Bryant, J. Wu and P. R. Palmer, "Implementation and Comparison of Power Diode Models for System Simulation," *2005 Int. Conf. Power Electronics and Drives Systems,* Kuala Lumpur, 2005, pp. 694-699.

[22] Ying-Yu Tzou and Lun-Jun Hsu, "A practical SPICE macro model for the IGBT," *Industrial Electronics, Control, and Instrumentation, Proc. IECON '93., Int. Conf.,* Maui, HI, 1993, pp. 762-766, vol.2.

[23] H. S. Kim, Y. H. Cho, S. D. Kim and Y. I. Choi, "Parameter extraction for the static and dynamic model of IGBT," *Power Electronics Specialists Conf., PESC '93 Rec., 24th Annu. IEEE,* Seattle, WA, 1993, pp. 71-74.

[24] Z. Shen and T. P. Chow, "Modeling and characterization of the insulated gate bipolar transistor (IGBT) for SPICE simulation," *Power Semiconductor Devices and ICs, ISPSD '93., Proc. 5th Int. Symp.,* Monterey, CA, 1993, pp. 165-170.

[25] J. T. Hsu and K. D. T. Ngo, "Behavioral modeling of the IGBT using the Hammerstein configuration," *IEEE Trans. Power Electron.,* vol.11, no.6, pp.746-754, Nov. 1996.

[26] C. Wong, "EMTP modeling of IGBT dynamic performance for power dissipation estimation," *IEEE Trans. Ind. Appl.,* vol. 33, no. 1, pp. 64-71, Jan/Feb. 1997.

[27] M. Cotorogea, "Implementation of mathematical models of power devices for circuit simulation in PSpice," *Computers in Power Electronics, 6th Workshop,* Cernobbio, 1998, pp. 17-22.

[28] A. Maxim, D. Andreu and J. Boucher, "High accuracy SPICE behavioral macromodeling of insulated gate bipolar transistor (IGBT)," *Applied Power Electronics Conference and Exposition, APEC '98. Conf. Proc. 1998., 13th Annu.,* Anaheim, CA, 1998, pp. 749-755, vol.2.

[29] A. Maxim and G. Maxim, "A novel analog behavioral IGBT spice macromodel," *Power Electronics Specialists Conf., PESC 99. 30th Annu. IEEE,* Charleston, SC, 1999, pp. 364-369, vol.1.

[30] Min Zhang, A. Courtay and Zhilian Yang, "An improved behavioral IGBT model and its characterization tool," *Electron Devices Meeting, Proc. 2000 IEEE Hong Kong,* Hong Kong, 2000, pp. 142-145.

[31] J. Mei, J. Zheng, M. Hu and Y. Rao, "Application of Saber's Simulation Model IGBT1 in Solid-State Switch Design" *2005 Int. Conf. Electrical Machines and Systems,* Nanjing, 2005, pp. 2013-2017, Vol. 3.

[32] J. L. Tichenor, S. D. Sudhoff and J. L. Drewniak, "Behavioral IGBT modeling for predicting high frequency effects in motor drives," *IEEE Trans. Power Electron.,* vol. 15, no. 2, pp. 354-360, Mar. 2000.

[33] P. O. Lauritzen, G. K. Andersen and M. Helsper, "A basic IGBT model with easy parameter extraction," *Power Electronics Specialists Conf., PESC. 2001 IEEE 32nd Annu.,* Vancouver, BC, 2001, pp. 2160-2165, vol. 4.

[34] A. Consoli *et al.*, "A SPICE behavioural model of PowerMESH$^{\text{TM}}$ IGBTs," *Power Electronics Specialists Conf., PESC. 2004 IEEE 35th Annu.,* 2004, pp. 2983-2989, Vol.4.

[35] Wanying Kang, Hyungkeun Ahn and M. A. E. Nokali, "A parameter extraction algorithm for an IGBT behavioral model," *IEEE Trans. Power Electron.,* vol. 19, no. 6, pp. 1365-1371, Nov. 2004.

[36] S. H. Stier and P. Mutschler, "An investigation of a reliable behavioral IGBT model in comparison to PSpice and measurement in hard- and soft switching applications," *Power Electronics and Applications, 2005 European Conf.,* Dresden, 2005, pp. P.1-P.10.

[37] H. S. Oh, M. El Nokali, "A new IGBT behavioral model," *Solid- State Electron.,* vol. 45, pp. 2069-2075, Nov. 2001.

[38] K. Asparuhova and T. Grigorova, "IGBT Behavioral PSPICE Model," *25th Int. Conf. Microelectronics,* Belgrade, 2006, pp. 203-206.

[39] K. Asparuhova and T. Grigorova, "IGBT high accuracy behavioral macromodel," *Microelectronics, MIEL 2008. 26th Int. Conf.,* Nis, 2008, pp. 185-188.

[40] I. Baraia, J. Galarza, J. A. Barrena and J. M. Canales, "An IGBT behavioural model based on curve fitting methods," *2008 IEEE Power Electronics Specialists Conf.,* Rhodes, 2008, pp. 1971-1977.

[41] A. Castellazzi, "Comprehensive Compact Models for the Circuit Simulation of Multichip Power Modules," *IEEE Trans. Power Electron.,* vol. 25, no. 5, pp. 1251-1264, May 2010.

[42] A. Nejadpak and O. A. Mohammed, "Functional ON/OFF behavioral modeling of power IGBT using system identification methods," *2012 27th Annu. IEEE Applied Power Electronics Conf. and Expo. (APEC),* Orlando, FL, 2012, pp. 1826-1832.

[43] N. Mijlad, E. Elwarraki and A. Elbacha, "Implementation of a behavioral IGBT model in SIMULINK," *2015 Int. Conf. Electrical and Information Technologies (ICEIT),* Marrakech, 2015, pp. 129-133.

[44] G. G. Parma and V. Dinavahi, "Real-Time Digital Hardware Simulation of Power Electronics and Drives," *IEEE Trans. Power Del.,* vol. 22, no. 2, pp. 1235-1246, Apr. 2007.

[45] A. Myaing and V. Dinavahi, "FPGA-based real-time emulation of power electronic systems with detailed representation of device characteristics," *IEEE Trans. Ind. Electron.,* vol. 58, no. 1, pp. 358-368, Jan. 2011.

[46] L. Herrera, C. Li, X. Yao, D. Jiao and J. Wang, "FPGA based real time electro-thermal modeling of power electronic converters," *2013 28th Annu. IEEE Applied Power Electronics Conf. and Expo. (APEC),* Long Beach, CA, 2013, pp. 1725-1729.

[47] H. J. Wu and W. S. Feng, "Efficient simulation of switched networks using reduced unification matrix," *IEEE Trans. Power Electron.,* vol. 14, no. 3, pp. 481-494, May 1999.

[48] P. Le-Huy, S. Guerette, L. A. Dessaint and H. Le-Huy, "Real-Time Simulation of Power Electronics in Power Systems using an FPGA," *2006 Canadian Conf. Electrical and Computer Engineering,* Ottawa, Ont., 2006, pp. 873-877.

[49] P. Le-Huy, S. Guerette, L. A. Dessaint and H. Le-Huy, "Dual-Step Real-Time Simulation of Power Electronic Converters Using an FPGA," *2006 IEEE Int. Symp. Industrial Electronics,* Montreal, Que., 2006, pp. 1548-1553.

[50] S. Hui and C. Christopoulos, "A discrete approach to the modeling of power electronic switching networks," *IEEE Trans. Power Electron.,* vol. 5, no. 4, pp. 398-403, Oct. 1990.

[51] Y. Li, T. Chang, F. Bai and Y. Chen, "Switching characteristics simulation of IGBT based on FPGA," *Computational Problem-Solving (ICCP), 2010 Int. Conf.,* vol. Lijiang, 2010, pp. 386-391.

[52] M. Dagbagi, L. Idkhajine, E. Monmasson and I. Slama-Belkhodja, "FPGA implementation of Power Electronic Converter real-time model," *Power Electronics, Electrical Drives, Automation and Motion (SPEEDAM), 2012 Int. Symp.,* Sorrento, 2012, pp. 658-663.

[53] P. Pejovic and D. Maksimovic, "A method for fast time-domain simulation of networks with switches," *IEEE Trans. Power Electron.,* vol. 9, No. 4, pp. 449-456, July 1994.

[54] A. Kiffe, S. Geng and T. Schulte, "Automated generation of a FPGA-based oversampling model of power electronic circuits," *Power Electronics and Motion Control Conf. (EPE/PEMC), 2012 15th Int.,* Novi Sad, 2012, pp. DS3f.5-1-DS3f.5-8.

[55] H. Zhang and J. Sun, "FPGA-based simulation of power electronics using iterative methods," *2014 Int. Power Electronics Conf. (IPEC-Hiroshima 2014 - ECCE ASIA),* Hiroshima, 2014, pp. 2202-2207.

[56] S. Huang, Q. Zhou and J. Gao, "Nonlinear modeling and test of inverter based on FPGA," *Electrical Machines and Systems (ICEMS), 2014 17th Int. Conf.,* Hangzhou, 2014, pp. 958-962.

[57] H. Jin, "Behavior-mode simulation of power electronic circuits," *IEEE Trans. Power Electron.,* vol. 12, no. 3, pp. 443-452, May 1997.

[58] A. Myaing, M. O. Faruque, V. Dinavahi and C. Dufour, "Comparison of insulated gate bipolar transistor models for FPGA-based real-time simulation of electric drives and application guideline," *IET Power Electron.,* vol. 5, no. 3, pp. 293-303, Mar. 2012.

[59] W. Wang, Z. Shen and V. Dinavahi, "Physics-Based Device-Level Power Electronic Circuit Hardware Emulation on FPGA," *IEEE Trans. Ind. Informat.,* vol. 10, no. 4, pp. 2166-2179, Nov. 2014.

[60] G. T. Oziemkiewicz, “Implementation and Development of the NIST IGBT Model in a SPICE-Based Commercial Circuit Simulator”, Master of Science Thesis at University of Florida, 1995.

[61] *“Saber user guide,”* Synopsys, Inc., USA, Sept. 2011.

[62] Y. Chen and V. Dinavahi, “FPGA-Based Real-Time EMTP,” *IEEE Trans. Power Del.,* vol. 24, no. 2, pp. 892-902, Apr. 2009.

[63] Y. Chen and V. Dinavahi, “An iterative real-time nonlinear electromagnetic transient solver on FPGA,” *IEEE Trans. Ind. Electron.,* vol. 58, no. 6, pp. 2547-2555, June 2011.

[64] Y. Chen and V. Dinavahi, “Digital hardware emulation of universal machine and universal line models for real-time electromagnetic transient simulation,” *IEEE Trans. Ind. Electron.,* vol. 59, no. 2, pp. 1300-1309, Feb. 2012.

[65] Y. Chen and V. Dinavahi, “Multi-FPGA digital hardware design for detailed large-scale real-time electromagnetic transient simulation of power systems”, *IET Gener. Transm. Distrib.,* vol. 7, no. 5, pp. 451-463, May 2013.

[66] Y. Chen and V. Dinavahi, “Hardware emulation building blocks for real-time simulation of large-scale power grids,” *IEEE Trans. Ind. Informat.,* vol. 10, no. 1, pp. 373-381, Feb. 2014.

[67] Y. Wang and V. Dinavahi, “Low-latency distance protective relay on FPGA,” *IEEE Trans. Smart Grid,* vol. 5, no. 2, pp. 896-905, Mar. 2014.

[68] Y. Wang and V. Dinavahi, “Real-time digital multi-function protection system on reconfigurable hardware,” *IET em IET Gener. Transm. Distrib.,* vol. 10, no. 10, pp. 2295-2305, July 2016.

[69] J. Liu and V. Dinavahi, “A real-time nonlinear hysteretic power transformer transient model on FPGA,” *IEEE Trans. Ind. Electron.,* vol. 61, no. 7, pp. 3587-3597, July 2014.

[70] J. Liu and V. Dinavahi, “Detailed Magnetic Equivalent Circuit Based Real-Time Nonlinear Power Transformer Model on FPGA for Electromagnetic Transient Studies,” *IEEE Trans. Ind. Electron.,* vol. 63, no. 2, pp. 1191-1202, Feb. 2016.

[71] J. Liu and V. Dinavahi, “Nonlinear Magnetic Equivalent Circuit Based Real-time Sen Transformer Electromagnetic Transient Model on FPGA for HIL Emulation,” *IEEE Trans. Power Del.,* vol. PP, no. 99, pp. 1-1, Jan. 2016.

[72] N. R. Tavana and V. Dinavahi, “A general framework for FPGA-based real-time emulation of electrical machines for HIL applications,” *IEEE Trans. Ind. Electron.,* vol. 62, no. 4, pp. 2041-2053, Apr. 2015.

[73] N. R. Tavana and V. Dinavahi, "Real-Time FPGA-Based Analytical Space Harmonic Model of Permanent Magnet Machines for Hardware-in-the-Loop Simulation," *IEEE Trans. Magn.,* vol. 51, no. 8, pp. 1-9, Aug. 2015.

[74] N. R. Tavana and V. Dinavahi, "Real-Time Nonlinear Magnetic Equivalent Circuit Model of Induction Machine on FPGA for Hardware-in-the-Loop Simulation," *IEEE Trans. Energy Convers.,* vol. 31, no. 2, pp. 520-530, June 2016.

[75] B. Jandaghi and V. Dinavahi, "Hardware-in-the-Loop Emulation of Linear Induction Motor Drive for MagLev Application," *IEEE Trans. Plasma Sci,* vol. 44, no. 4, pp. 679-686, Apr. 2016.

[76] Z. Shen and V. Dinavahi, "Real-Time Device-Level Transient Electrothermal Model for Modular Multilevel Converter on FPGA," *IEEE Trans. Power Electron.,* vol. 31, no. 9, pp. 6155-6168, Sept. 2016.

[77] M. D. Omar Faruque *et al.,* "Real-Time Simulation Technologies for Power Systems Design, Testing, and Analysis," *IEEE J. Power and Energy Technology Syst.,* vol. 2, no. 2, pp. 63-73, June 2015.

[78] X. Guillaud *et al.*, "Applications of Real-Time Simulation Technologies in Power and Energy Systems," *IEEE J. Power and Energy Technology Syst.,* vol. 2, no. 3, pp. 103-115, Sept. 2015.

[79] L. F. Pak, M. O. Faruque, X. Nie, and V. Dinavahi, "A versatile cluster-based real-time digital simulator for power engineering research," *IEEE Trans. Power Syst.,* vol. 21, no. 2, pp. 455-465, May 2006.

[80] L. F. Pak, V. Dinavahi, G. Chang, M. Steurer and P. F. Ribeiro, "Real-Time Digital Time-Varying Harmonic Modeling and Simulation Techniques IEEE Task Force on Harmonics Modeling and Simulation," *IEEE Trans. Power Del.,* vol. 22, no. 2, pp. 1218-1227, Apr. 2007.

[81] X. Nie, Y. Chen, and V. Dinavahi, "Real-Time Transient Simulation Based on a Robust Two-Layer Network Equivalent," *IEEE Trans. Power Syst.,* vol. 22, no. 4, pp. 1771-1781, Nov. 2007.

[82] V. Jalili-Marandi and V. Dinavahi, "Instantaneous Relaxation-Based Real-Time Transient Stability Simulation," *IEEE Trans. Power Syst.,* vol. 24, no. 3, pp. 1327-1336, Aug. 2009.

[83] B. Asghari, M. O. Faruque and V. Dinavahi, "Detailed Real-Time Transient Model of the ?Sen? Transformer," *IEEE Trans. Power Del.,* vol. 23, no. 3, pp. 1513-1521, July 2008.

[84] L. F. Pak and V. Dinavahi, "Real-Time Simulation of a Wind Energy System Based on the Doubly-Fed Induction Generator," *IEEE Trans. Power Syst.,* vol. 24, no. 3, pp. 1301-1309, Aug. 2009.

[85] M. O. Faruque and V. Dinavahi, "Hardware-in-the-Loop Simulation of Power Electronic Systems Using Adaptive Discretization" *IEEE Trans. Ind. Electron.,* vol. 57, no. 4, pp. 1146-1158, Apr. 2010.

[86] V. Jalili-Marandi, L. F. Pak and V. Dinavahi, "Real-Time Simulation of Grid-Connected Wind Farms Using Physical Aggregation," *IEEE Trans. Ind. Electron.,* vol. 57, no. 9, pp. 3010-3021, Sept. 2010.

[87] W. Ren *et al.*, "Interfacing issues in real-time digital simulators," *IEEE Trans. Power Del.,* vol. 26, no. 2, pp. 1221-1230, Apr. 2011.

[88] B. Asghari and V. Dinavahi, "Real-Time Nonlinear Transient Simulation Based on Optimized Transmission Line Modeling," *IEEE Trans. Power Syst.,* vol. 26, no. 2, pp. 699-709, May 2011.

[89] B. Asghari and V. Dinavahi, "Novel Transmission Line Modeling Method for Nonlinear Permeance Network Based Simulation of Induction Machines," *IEEE Trans. Magn.,* vol. 47, no. 8, pp. 2100-2108, Aug. 2011.

[90] B. Asghari and V. Dinavahi, "Experimental Validation of a Geometrical Nonlinear Permeance Network Based Real-Time Induction Machine Model," *IEEE Trans. Ind. Electron.,* vol. 59, no. 11, pp. 4049-4062, Nov. 2012.

[91] S. Wang, V. Dinavahi and J. Xiao, "Multi-rate real-time model-based parameter estimation and state identification for induction motors," *IET Electric Power Applicat.,* vol. 7, no. 1, pp. 77-86, Jan. 2013.

[92] Jimnez *et al.*, "Implementation of an FPGA-Based Online Hardware-in-the-Loop Emulator Using High-Level Synthesis Tools for Resonant Power Converters Applied to Induction Heating Appliances," *IEEE Trans. Ind. Electron.,* vol. 62, no. 4, pp. 2206-2214, Apr. 2015.

[93] Z. Zhou and V. Dinavahi, "Parallel massive-thread electromagnetic transient simulation on GPU," *IEEE Trans. Power Del.,* vol. 29, no. 3, pp. 1045-1053, June 2014.

[94] W. Wang, W. Yang and V. Dinavahi, "Co-simulation interfacing capabilities in device-level power electronic circuit simulation tools: an overview," *Int. J. Power Electron. and Drive Syst. (IJPEDS),* vol. 6, no. 2, pp. 665-682, Dec. 2015.

[95] K. K. Fung and S. Y. R. Hui, "Fast simulation of multistage power electronic systems with widely separated operating frequencies," *IEEE Trans. Power Electron.,* vol. 11, no. 3, pp. 405-412, May 1996.

[96] *"7 series FPGAs clocking resources user guide,"* Xilinx, Inc., USA, June 2015.

[97] *"7 series FPGAs configurable logic block user guide,"* Xilinx, Inc., USA, Nov. 2014.

[98] *"7 series FPGAs Memory Resources user guide,"* Xilinx, Inc., USA, Nov. 2014.

[99] *"7 series DSP48E1 slice user guide,"* Xilinx, Inc., USA, Nov. 2014.

[100] *"7 series FPGAs SelectIO resources user guide,"* Xilinx, Inc., USA, Sep. 2015.

[101] *"7 series FPGAs Packaging and Pinout Product Specification,"* Xilinx, Inc., USA, Mar. 2016.

[102] *"Vivado design suite user guide design flows overview,"* Xilinx, Inc., USA, Apr. 2016.

[103] *"UltraFast Design Methodology Guide for the Vivado Design Suite,"* Xilinx, Inc., USA, Nov. 2015.

[104] *"Vivado Design Suite User Guide System-Level Design Entry,"* Xilinx, Inc., USA, Apr. 2016.

[105] *"Vivado Design Suite User Guide Logic Simulation,"* Xilinx, Inc., USA, Apr. 2016.

[106] *"Vivado Design Suite User Guide Synthesis,"* Xilinx, Inc., USA, Apr. 2016.

[107] *"Vivado Design Suite User Guide Implementation,"* Xilinx, Inc., USA, Apr. 2016.

[108] C. Gear, "Simultaneous Numerical Solution of Differential-Algebraic Equations," *IEEE Trans. Circuit Theory.,* vol. 18, pp. 89-94, 1971.

[109] M. Aredes *et al.,* "Comparative Analysis of Shunt Active Filter Models in the EMTP/ATP and SABER Programs," *International Conference on Power Systems Transients IPST.,* New Orleans, USA, Oct. 2003.

[110] G. Massobrio and P. Antognetti, *Semiconductor Device Modeling with SPICE.* McGraw-Hill, Inc., USA, 1993.

[111] P. B. Johns, M. O'Brien, "Use of the transmission-line modeling (TLM) method to solve nonlinear lumped networks," *Radio and Electron. Eng.,* vol. 50, no. (1/2), pp. 59-70, 1980.

[112] Zhang Bowei, Gu Guochang, Sun Lin, and Wu Yanxia, "Floating-Point FPGA Gaussian Elimination in Reconfigurable Computing System," *Chinese Journal of Electronics,* vol.20, no. 1, pp. 51-54, Jan. 2011

[113] V. Dinavahi, "Transient analysis of systems with multiple nonlinear elements using TLM," *IEEE Trans. Power Syst.,* vol. 19, no. 4, pp. 2102-2103, Nov. 2004.

[114] S. Y. R. Hui and C. Christopoulos, "Modeling non-linear power electronic circuits with the transmission-line modeling technique," *IEEE Trans. Power Electron.,* vol. 10, no. 1, pp. 48-54, Jan. 1995.

[115] S. Y. R. Hui and K. K. Fung, "Fast decoupled simulation of large power electronic systems using new two-port companion link models," *IEEE Trans. Power Electron.,* vol. 12, no. 3, pp. 462-473, May 1997.

[116] C. J. Smartt and C. Christopoulos, "Modelling nonlinear and dispersive propagation problems by using the TLM method," *Microwaves, Antennas and Propagation, IEE Proc.,* vol. 145, no. 3, pp. 193-200, Jan. 1998.

[117] M. Hagiwara and H. Akagi, "Control and Experiment of Pulsewidth-Modulated Modular Multilevel Converters," *IEEE Trans. Power Electron.,* vol. 24, no. 7, pp. 1737-1746, July 2009.