THÈSE

Présentée à

L'Ecole Nationale d'Ingénieurs de Tunis

Pour l'obtention du grade de

DOCTEUR DE L'ECOLE NATIONALE D'INGENIEURS DE TUNIS

SPÉCIALITÉ: GÉNIE ÉLECTRIQUE

Pour l'obtention du grade de

DOCTEUR DE L'UNIVERSITE DE CERGY-PONTOISE

Ecole doctorale : Sciences et Ingénierie

Diplôme National - Arrêté du 7 août 2006

SPÉCIALITÉ: GÉNIE ÉLECTRIQUE ET ÉLECTRONIQUE

Présentée par :

Mohamed DAGBAGI

FPGA-Based Embedded Real Time Simulation of Electrical Systems

Publicly defended on 08 October 2015 in front of the jury composed of:

President	:	Prof. Khaled JELASSI, ENIT, Tunisia
Reviewer	:	Prof. Ahmed MASMOUDI, ENIS, Tunisia
Reviewer	:	Prof. Serge PIERFEDERICI, ENSEM Nancy, France
Examiner	:	Prof. François AUGER, Université de Nantes, France
Advisor	:	Prof. Eric MONMASSON, Université de Cergy-Pontoise, France
Advisor	:	Prof. Ilhem SLAMA-BELKHODJA, ENIT, Tunisia
Invited	:	Dr. Lahoucine IDKHAJINE, Université de Cergy-Pontoise, France

This thesis has been performed in the Laboratory of Electrical Systems of ENIT (LR LSE- LR 11 ES 15) and in the SATIE Laboratory of UCP (SATIE - UCP / UMR 8029)

To my parents,
To my family,
To all those who are dear to me.

Abstract

The aim of this thesis work is to develop an IP-Library of FPGA-based embedded real-time simulator IPs (Intellectual Properties) that simulate different elements of an electrical system. These IPs have been designed to be used not only for Hardware-In-the-Loop (HIL) testing of digital controllers but also for low cost embedded control applications, where the simulator IP and the controller are both implemented and run altogether in the same FPGA device. This emerging class of real-time simulators is expected to be more and more included in the next generation of digital controllers. Indeed, such embedded real-time simulator IPs can be advantageously embedded within digital controllers to ensure functions like observation, estimation, diagnostic or health-monitoring. Conversely to the HIL case, the main challenge when designing such simulator IPs is to cope with their complexity having in mind that, in the case of embedded systems, the available hardware resources are limited due to the cost. Furthermore, this challenge is strengthened by the need of very short simulation time-steps which is typically the case when simulating power converters.

To develop these IPs, dedicated design guidelines have been proposed to be followed to manage the complexity of these simulator IPs (model solver, numerical solver, time-step, data conditioning) with regards to the timing and the area/cost constraints (computation time limit, limited hardware resources …).

The simulators IPs to be developed have been organized into two main categories: those dedicated to electromagnetic elements of an electrical system and those dedicated to their switching elements.

The first category gathers elements where electric, magnetic phenomena are modelized in addition to mechanical phenomena (for moving systems) and potentially thermal phenomena. Three cases are dealt with: the embedded real-time simulator of a three-phase DC-excited synchronous machine, the one of a three-phase induction machine and the one of a a three-stage avionics alternator. Also, the advantages of using delta transformation to improve the stability of the numerical solver when short simulation time-step and fixed-point (with limited data precision) are used, have been studied.

The second category concerns switching elements such as power converters where switching events are considered. Here again, several converter topologies have been studied: a half-wave rectifier, a buck DC-DC converter, a bidirectional buck DC-DC converter, a H-bridge DC-DC converter, a single-phase H-bridge DC-AC converter, a three-phase voltage source inverter, a three-phase diode rectifier and a three-phase PWM rectifier. For all these IPs, the Associated Discrete Circuit (ADC) modeling approach is adopted.

The embedded real-time simulator IP of the three-phase PWM rectifier has been applied in the context of an embedded application. The latter consists of a fault-tolerant control of a grid-connected voltage source rectifier. Thus, this simulator IP is associated with the one of a three-phase RL-filter and are both implemented within the rectifier controller to estimate the grid currents. These currents are injected in the controller in the case of a current sensor fault. The ability of this estimator to guarantee the service continuity in the case of faults is validated through HIL tests and experiments.

Keywords

- Field Programmable Gate Array
- Embedded Real-time simulation
- Electrical systems
- Embedded digital controllers
- Hardware In the Loop
- Fault-tolerant control
- Power converters
- AC machines
- Associated Discrete Circuit
- Delta transformation
- Fixed-point data representation

Résumé

L'objectif de ce travail de thèse est de développer une bibliothèque de modules IPs (Intellectual Properties) de simulateurs temps réel embarqués qui simulent différents éléments d'un système électrique. Ces modules ont été conçus pour être utilisés non seulement pour une validation HIL (Hardware-In-the-Loop) des contrôleurs numériques mais aussi pour des applications de contrôle embarquées, où le module IP de simulateur et le contrôleur sont tous les deux implantés et exécutés dans la même cible FPGA. Cette nouvelle classe de simulateurs temps réel devrait être de plus en plus incluse dans la prochaine génération de contrôleurs numériques. En effet, ces modules IPs de simulateurs temps réel embarqués peuvent être avantageusement intégrés dans les contrôleurs numériques pour assurer des fonctions comme l'observation, l'estimation, le diagnostic où la surveillance de la santé. Inversement aux cas de HIL, le principal défi lors de la conception de tels simulateurs est de faire face à leur complexité ayant à l'esprit que, dans le cas des systèmes embarqués, les ressources matérielles disponibles sont limitées en raison du coût. En outre, ce problème est renforcé par la nécessité des pas de simulation très petits. Ceci est généralement le cas lors de la simulation des convertisseurs de puissance.

Pour développer ces modules IPs, des lignes directrices dédiées de conception ont été proposées pour être suivies pour gérer la complexité de ces simulateurs (solveur de modèle, solveur numérique, pas de simulation, conditionnement de données) tout en tenant compte des contraintes temporelles et matérielles/coût (temps de calcul limité, ressources matérielles limitées ...).

Les modules IPs de simulateurs à développer ont été organisés en deux catégories principales: ceux qui sont consacrées aux éléments électromagnétiques d'un système électrique, et ceux dédiés à ses éléments commutés.

La première catégorie regroupe les éléments électromagnétiques où les phénomènes électriques, magnétiques sont modélisés en plus de phénomènes mécaniques (pour les parties mécaniques) et des phénomènes potentiellement thermiques. Trois cas sont traités: le simulateur temps réel embarqué d'une machine synchrone triphasée, celui d'une machine asynchrone triphasée et celui d'un alternateur avionique à trois étages. En plus de cela, les avantages de l'utilisation de la transformation delta pour améliorer la stabilité du solveur numérique lorsque un petit pas de calcul et le codage virgule fixe (avec une précision de données limitée) sont utilisés, ont été étudiés.

La deuxième catégorie concerne des éléments commutés tels que les convertisseurs de puissance où les événements de commutation sont considérés. Là encore, plusieurs topologies de convertisseurs ont été étudiées: un redresseur simple alternance, un hacheur série, un hacheur réversible en courant, un hacheur quatre quadrant, un onduleur monophasé, un onduleur triphasé, un redresseur à diodes triphasé et un redresseur MLI triphasé. Pour tous ces modules IPs de simulateurs, l'approche de modélisation ADC (Associated Discrete Circuit) est adoptée.

Le module IP de simulateur temps réel embarqué du redresseur MLI a été appliqué dans un contexte d'une application embarquée. Cette dernière consiste en une commande tolérante aux défauts d'un convertisseur de tension coté réseau. Ainsi, ce module IP est associé à celui d'un simulateur temps réel d'un filtre RL triphasé et les deux sont embarqués dans le dispositif de commande du redresseur pour estimer les courants de lignes. Ces courants sont injectés dans le dispositif de commande dans le cas d'un défaut de capteur de courant. La capacité de cet estimateur de garantir la continuité de service en cas de défauts est validée par des tests HIL et expérimentalement.

Mots clefs

- Réseaux de portes programmables – Field Programmable Gate Array
- Simulation temps réel embarquée
- Systèmes électriques
- Commande numérique embarquée
- Procédure Hardware In the Loop
- Commande tolérante aux défauts
- Convertisseurs de puissance
- Machines alternatives
- Associated Discrete Circuit
- Transformée delta
- Codage virgule fixe

ملخص

الغرض من هذا العمل هو تطوير مكتبة ملكيات فكرية لأجهزة محاكاة مضمنة في الوقت الحقيقي مستندة على FPGA والتي تحاكي مختلف عناصر النظام الكهربائي. هذه الملكيات الفكرية تم تصميمها ليس فقط لاستخدامها في اختبار وحدات التحكم الرقمية بواسطة الأجهزة في الحلقة HIL و لكن أيضا في تطبيقات تحكم مضمنة ومنخفضة التكلفة، حيث جهاز المحاكاة و وحدة التحكم كلاهما يوجدان ويشتغلان داخل نفس الجذاذة الرقمية FPGA. هذه الفئة الناشئة من أجهزة المحاكاة في الوقت الحقيقي من المتوقع أن تكون أكثر فأكثر مدرجة في الجيل القادم من وحدات التحكم الرقمية. في الواقع، مثل أجهزة المحاكاة المضمنة في الوقت الحقيقي هذه يمكن أن تكون جزأ لا يتجزأ مفيد في وحدات التحكم الرقمية لضمان وظائف مثل الملاحظة والتقدير والتشخيص أو رصد الصحة. على عكس حالة الإختبار بواسطة HIL، فإن التحدي الرئيسي عند تصميم مثل أجهزة المحاكاة هذه هو التعامل مع تعقيداتها مع الأخذ في الاعتبار أنه في حالة الأنظمة المضمنة، موارد الأجهزة المتاحة محدودة بسبب التكلفة. علاوة على ذلك، تم تعزيز هذا التحدي من خلال الحاجة لخطوات محاكاة وقت قصيرة جدا، وهذا هو الحال عادة عند محاكاة محولات الطاقة.

لتطوير هذه الملكيات الفكرية، تم اقتراح إرشادات تصميم مخصص, الواجب اتباعها لإدارة تعقيد هذه المحاكاة (حَلَّال النموذج ، الحَلَّال العددي، خطوة الوقت ، تكييف البيانات) بنظر إلي القيود المتعلقة بالتوقيت و المنطقة / التكلفة (الحد الزمني لحساب، موارد الأجهزة محدودة ...).

المحاكاة التي سيتم تطويرها نظمت إلى فئتين رئيسيتين: تلك المخصصة للعناصر الكهرومغناطيسية للنظام الكهربائي وتلك المخصصة لعناصر التحويل الخاصة به.

الفئة الأولى تجمع عناصر حيث وضعت نماذج للظواهر الكهربائية والمغناطيسية بالإضافة إلى الظواهر الميكانيكية (أنظمة التحرك) والظواهر يحتمل الحرارية. تم تناول ثلاث حالات: جهاز محاكاة مضمن في الوقت الحقيقي لآلة تزامنية ثلاثية الاطوار، واحد لآلة غير تزامنية ثلاثية الاطوار و واحد لمولد إلكترونيات الطيران ثلاث مراحل. أيضا، فإنه تمت دراسة مزايا استخدام التحول دلتا لتحسين استقرار الحَلَّال العددي عندما تستخدم المحاكاة قصيرة خطوة الوقت ونقطة ثابتة لتمثيل البيانات (مع دقة بيانات محدودة).

الفئة الثانية تهتم بعناصر التحويل مثل محولات الطاقة أين أحداث التحويل مأخوذة بعين الأعتبار. هنا مرة أخرى، تمت دراسة عدة مخططات تحويل: مقوم نصف الموجة، محول باك، محول باك ثنائي الاتجاه، محول مستمر متردد أربعة رباعي، محول جهد ثلاثي الاطوار، مقوم بديودات ثلاثي الاطوار, مقوم تظمين عرض النبظة ثلاثي الاطوار. لجميع أجهزة المحاكاة هذه اعتمدت تقنية نهج النماذج ADC.

جهاز محاكاة المظمن في الوقت الحقيقي لمقوم تظمين عرض النبظة ثلاثي الاطوار تم إستعماله في سياق تطبيق المضمنة. يتكون هذا الأخير من عنصر تحكم في مقوم مصدر الجهد متصل بالشبكة قادر على استيعاب الأخطاء. وهكذا، فإن جهاز المحاكاة هذا تم ربطه بجهاز محاكاة أخر لفلتر RL ثلاثي الاطوار وكلاهما تم دمجهما ضمن وحدة التحكم في المقوم لتقدير تيارات الشبكة. يتم حقن هذه التيارات في وحدة التحكم في حالة وجود خطأ في حساس التيار. قدرة هذا المقدر لضمان استمرارية الخدمة في حالة الاعطال تم التحقق من صحتها من خلال اختبارات HIL والتجارب.

كلمات مفاتيح

- شرائح مصفوفة البوابات المنطقية القابلة للبرمجة
- محاكاة مضمنة في الوقت الحقيقي
- الأنظمة الكهربائية
- التحكم الرقمي المضمن
- إجراء الأجهزة في حلقة
- التحكم المتسامح مع الأخطأ
- محولات الطاقة
- ألات تيار متردد
- الدائرة الغير مستمرة المرتبطة
- تحول دلتا
- ترميز النقطة الثابتة

Acknowledgments

The work presented in this thesis has been carried out in the context of a joint PhD dissertation through a joint supervision between two research laboratories: The "Laboratoire des Systèmes Electriques (LSE) " of "Ecole Nationale d'Ingénieurs de Tunis (ENIT)" and the "Laboratoire des Systèmes et Applications des Technologies de l'Information et de l'Energie (SATIE), antenne de l'Université de Cergy-Pontoise (UCP) ".

These few acknowledgments reflect my gratitude for all those who have contributed directly or indirectly to the success of this work.

In the first place, I would like to express my profound gratitude to Professor. Eric MONMASSON, my advisor, and Head of the "Laboratoire SATIE, antenne de l'UCP", for supervising my thesis work, his valuable guidance and providing an excellent research environment during my stay in France.

I which to express my gratitude to Professor Ms. Ilhem SLAMA-BELKHODJA, my advisor and Head of "Laboratoire LSE, à l'ENIT ", for supervising my thesis work, for her confidence and her encouragements.

I would like to strongly thank Mr. Lahoucine IDKHAJINE, Associate Professor in UCP and member of "Laboratoire SATIE", for his patience and generosity devoted valuable time and provided great help to the development of this work.

Also, I thank Professors Khaled JELASSI, Ahmed MASMOUDI, Serge PIERFEDERICI and François AUGER for reviewing and examining this thesis work.

I also take this opportunity to express my deep gratitude to Marie-Hélène MOREAU, Aude BREBANT and Don Abasse BOUKARI for making my stay in France easy by taking care of all the administrative aspects and the small daily worries.

I which also to thank all my family members, especially my loving mother Rebeh and my dear father Hassen, for their encouragements and support during the past years.

Many thanks also to all my laboratory friends in "Laboratoire LSE" and "Laboratoire SATIE" for their friendships and for the scientific discussions along with the common friendly banters that we always enjoy.

Finaly, I would like to thank my past and recent institutes "Institut Supérieur d'Informatique - (ISI)" and "ENIT" for making this possible.

DAGBAGI Mohamed

Contents

List of Figures

Chapter 1:

Chapter 2:

Chapter 3:

Chapter 4:

Chapter 5:

Appendix A:

Appendix E:

Appendix F:

List of Tables

Chapter 1:

Chapter 3:

Chapter 4:

Chapter 5:

Appendix B:

Appendix D:

Appendix E:

General Introduction

During these last years, real-time digital simulation has been an advanced research topic in many engineering fields such as power electronics and AC drive applications. Compared to a standard offline simulation, real-time simulation consists in implementing a real-time simulator that is running in natural time and is reproducing the dynamic behavior of a system. Most of the developed simulators are applied in the context of Hardware-In-the-Loop (HIL) testing of digital controllers in order to validate them in every operating conditions, which would not be possible using only experimental tests [SHA13] [LUC11] [SHAH13] [SUT13][KAM13][HAS14].

The main issue of interest is how to develop real-time simulators based on model solvers and numerical solvers, able to accurately reproduce the system dynamics and transients. High performance digital simulation platforms are now commercialized, covering a wide range of system complexities and operating at very short simulation time-steps. They integrate powerful and scalable multi-core processor boards combined with FPGA (Field Programmable Gate Array) boards. In fact, these FPGA platforms are deployed to address the demand of very fast system dynamics, thus very short simulation time-steps. By this way, many authors have proposed to simulate the electrical system under test on these devices. Most of the time, specific hardware implementation are designed, taking advantage of the massive parallelism offered by such components in terms of logic cells, memory banks and DSP units. This design approach is used, for example, in [OUL13] and in [CHE12] to simulate several kinds of electrical AC machines and in [JIA14] to simulate a non-linear hysteretic power transformer. Besides, some authors have also proposed multi-core architectures, also based on powerful FPGA [MAJ11].

Other important research efforts have been done to enhance the accuracy of these FPGA-based HIL platforms while keeping the simulation time-step as low as possible. These efforts concern for example the floating-point data representation where powerful FPGA-based real-time simulators based on floating-point representation are now proposed. Along this line, OuldBachir et al. have proposed in [OUL13] efficient high-performance self-alignment floating-point calculation engines to implement power converter models. Also, to support this floating-point trend, Arria 10 and Stratix Altera FPGA series integrate hardwired variable precision DSP units that can be configured to operate either in fixed-point mode or in floating-point mode [ALT].

However, when implementing a real-time simulator in the context of HIL testing, the cost is not always taken into account. Indeed, in addition to the optimization of the development time, the main objective of a HIL test is to achieve a high level of accuracy. The consequence is then the use of powerful but costly digital platforms. This is typically the case of FPGA-based real-time simulators where the hardware resources to be used depend on the complexity of the model solver. Thus the price to pay when simulating complex systems is the use of expensive FPGA families (eg. Virtex [XIL] or Stratix [ALT]). The use of such expensive devices is not acceptable in some embedded applications where the real-time simulators are not only developed for HIL testing but also embedded within the controller. These simulators are called here: embedded real-time simulators.

In this thesis, it is intended by an embedded real-time simulator, an Intellectual Property (IP) module that simulates the system to be controlled or a part of it. Thus, this IP and the controller are both implemented and run altogether in the same FPGA device. This emerging class of real-time simulators is expected to be more and more included in the next generation of digital controllers. Indeed, such embedded real-time simulators can be advantageously integrated inside the control closed-loop (eg. observers, estimators) in order to reduce the number of sensors or parallelized with the main controller and used for added-value diagnostic and health-monitoring purposes. As an example of application (induction heating

appliances), in [JIM15], an embedded real-time simulator IP of a series-resonant half-bridge inverter has been included within its FPGA-based controller and used to estimate in real-time the efficiency of the inverter and its safety conditions. These estimations are used to improve the control performances and reliability for a wide range of operating conditions and loads.

However, for this kind of real-time simulator, the performance/cost criteria are of great importance. This is because a compromise has always to be found between the complexity of the simulated system, the expected accuracy (especially when power converters are simulated) and the size (thus the cost) of the FPGA on which it will be implemented. This is then the context of the proposed thesis work, where the implementation in low cost FPGAs of embedded real-time simulator IPs of electrical systems is studied.

1. Thesis objectives and author contributions

The main objective of this thesis is to develop an IP-library of embedded real-time simulator IPs that simulate different elements of an electrical system. These simulator IPs have to be designed to address not only HIL applications but also low cost embedded control applications, keeping in mind the additional constraints they imply. To develop these IPs, the methodology proposed in [NAO07P] and [IDK10] for designing industrial controllers has to be extended in order to manage the complexity of these simulator IPs (model solver, numerical solver, time-step, data conditioning) with regards to the timing and the area/cost constraints (computation time limit, limited hardware resources …).

To start with, the simulators IPs to be developed have been organized into two main categories: those dedicated to electromagnetic elements of an electrical system and those dedicated to their switching elements.

The first category gathers elements where electric, magnetic phenomena are modelized in addition to mechanical phenomena (for moving systems) and potentially thermal phenomena. Three cases are dealt with: the embedded real-time simulator of a 3-phase DC-excited synchronous machine, the one of a 3-phase induction machine and finally the one of a three-stage avionics alternator. Also, the advantages of using delta transformation to improve the stability of the numerical solver when short simulation time-step and fixed-point (with limited data precision) are used, have been studied. These works have led to the following publications: [DAGI11]-[DAGS11] [DAGP13].

The second category concerns switching elements such as power converters where switching events are considered. Here again, several converter topologies have been studied: a half-wave rectifier, a buck DC-DC converter, a bidirectional buck converter, a H-bridge DC-DC converter, a single-phase H-bridge DC-AC converter, a 3-phase voltage source inverter, a 3-phase diode rectifier and a 3-phase PWM rectifier, [DAGS12], [DAGI13], [DAGJ13], [DAGT15], [DAGI15], [DAGP15].

For all these IPs, the Associated Discrete Circuit (ADC) modeling approach is adopted. The embedded real-time simulator IP of the 3-phase PWM rectifier has been applied in the context of an embedded application, [DAGT15]. The latter consists of a fault-tolerant control of a grid-connected voltage source rectifier. Thus, this simulator IP is associated with the one of a 3-phase RL-filter and both are implemented within the rectifier controller to estimate the grid currents. These currents are injected in the controller in the case of a current sensor fault. The ability of this estimator to guarantee the service continuity in the case of faults is validated through HIL tests and experiments. This simulator IP is also used in the context of HIL testing in order to validate several PWM rectifier control algorithms ([DAGI15], [HEM15]).

The development process of all the designed FPGA-based embedded real-time simulator IPs is achieved with the help of dedicated design guidelines which are organized into four major steps: the preliminary system specification, the algorithm development, the FPGA implementation and the experimentations, [NAO07P], [IDK10]. The author's contributions here are:

- During the algorithm development, the model selection step has been added: this step provides the modeling approaches of both electromagnetic and switching elements.
- The constraints associated with each step are strengthened by those related to the embedded real-time simulation.
- The time spent by the designer during the HIL step will be reduced thanks to the simulator IPs already available in the IP-Library.

2. Thesis outline

This thesis report consists of five main chapters, described as follows:

Chapter 1 and Chapter 2 are dedicated to the state of the art of FPGA-based embedded real-time simulation of electrical systems. The first one gives a general discussion about the real-time digital simulation. The second one focuses on the problematics of embedded simulators and their FPGA implementation.

Then, in Chapter 1, author starts by presenting the advantages of a digital real-time simulation compared with an offline simulation. This is followed by an overview of the latest application trends. Next to this, the modeling of the elements of electrical systems is presented including the formulation methods (model solvers) and their corresponding constraints. When it comes to the digital realization, the most commonly used numerical solvers, the choice of the simulation time-step and the data representation are all investigated. Finally, the digital implementation is discussed. The evolution of the commercially available digital real-time simulation platforms is presented. Their advantages, limits and the contribution of FPGAs to boost their performances are all given.

In Chapter 2 author starts by discussing the applications where embedded real-time simulators can be encountered. Thus, the benefits of using FPGAs as embedded digital systems are presented. Then, the embedded real-time simulation constraints: those linked to the real-time simulation, those linked to low cost embedded systems and those linked to the FPGA implementation are presented and classified into five categories. Finally, design guidelines to be followed to design embedded real-time simulator IPs and to manage the constraints brought by this type of IPs are proposed.

Chapter 3 discusses the implementation in low cost FPGA of embedded real-time simulator IPs of electromagnetic elements. Three cases have been studied: the embedded real-time simulator of a 3-phase DC-excited synchronous machine, the one of a 3-phase induction machine and finally the one of a three-stage avionics alternator. The proposed design guidelines have been applied to the chosen case studies. Furthermore, the benefits of making a delta transformation are discussed. Indeed, compared to a shift operator, this transformation aims to improve the stability of the numerical solver when short simulation time-step and fixed-point (with limited data precision) are used.

Chapter 4 discusses the implementation in low cost FPGA of switching IP modules for embedded control applications. It has been decided to focus, in this thesis report, on three power converter topologies: a single-phase H-bridge DC-AC converter, a 3-phase voltage source inverter and a 3-phase diode rectifier. For all these IPs, the ADC based modeling

approach has been adopted. Here again, the proposed design guidelines have been also followed.

Chapter 5 presents a complete application where both electromagnetic and switching IPs are deployed and applied in the context of fault-tolerant control of 3-phase grid-connected voltage source rectifier. These IPs are the ADC-based embedded real-time simulator of a 3-phase PWM rectifier and the one of a 3-phase RL-filter. They are used as grid current estimator and the rectifier controller switches to these estimates when a fault appears on the current sensors. HIL and experimental validations are achieved.

3. Nomenclature

3.1. Synbols

Clk	: Clock signal
Reset	: Reset signal
en	: Enable signal
Start	: Start signal
End	: End signal
f, h	: Continuous-time state space matrix, System output matrix
C	: Control signal
T	: Switching period
V_{dc}	: DC-link voltage
I_{dc}	: DC-link current
R, L, C	: Resistance, Inductance, Capacitance
V, I	: Voltages and currents vectors
v_{La}, v_{Lb}, v_{Lc}	: Line voltages
i_{La}, i_{Lb}, i_{Lc}	: Line currents
G	: Conductance
J	: Dependent current source
T_s	: Sampling period
Δt	: Time-step
T_c	: Computation time
Φ	: Flux
v_{sd}, v_{sq}	: d-q stator voltages
$v_{s\alpha}$, $v_{s\beta}$	: α-β stator voltages
i_{sd}, i_{sq}	: d-q stator currents
$i_{s\alpha}$, $i_{s\beta}$	: α-β stator currents
$\phi_{r\alpha}$, $\phi_{r\beta}$	: α-β rotor fluxes
θ_e	: Angular position
ω_e	: Angular speed
E	: Speed voltage vector
i_{exc}, v_{exc}	: Excitation current, excitation voltage of the exciter machine
i_f, v_f	: Excitation current, excitation voltage of the main generator
i_{ap}, i_{bp}, i_{cp}	: 3-phase currents of the main generator
v_{ap}, v_{bp}, v_{cp}	: 3-phase voltages of the main generator
i_{dp}, i_{qp}	: d-q currents of the main generator
v_{dp}, v_{qp}	: d-q voltages of the main generator
i_D, i_Q	: d-q currents of the main generator damper windings
ω_{ep}, ω_{ee}	: Main generator electrical speed, exciter machine electrical speed
θ_{ep}, θ_{ee}	: Main generator electrical position, exciter machine electrical position

s : Laplace-operator
z : Shift-operator
γ : Delta-operator
Re : Real part
Im : Imaginary part
P_s : Most critical pole of the continuous-time model
P_{ci} : Poles of the continuous-time model
P_{ti} : Poles of the shift-operator model
P_{di} : Poles of the delta-operator model
c_i : Parameters of the continuous-time model
t_i : Parameters of the shift-operator model
d_i : Parameters of the delta-operator model
i_L : Load current
V_{emf} : Back EMF voltage
V_{RMS} : RMS voltage
S_D : Diode state
S_T : Transistor state
H : Conductance matrix
v_{Lab}, v_{Lac}, v_{Lbc} : Line to line voltages
Cap : Dummy capacitor

3.2. Indexes

s, r : Stator and rotor index
α, β : Stationary reference frame indexes
d, q : Rotating reference frame indexes
a, b, c : 3-phase reference frame index
$*$, $\wedge$: Reference quantity, Estimated quantity
B : Base quantity for normalization
k : Sampling index
in, out : Input and output indexes
sw : Switch index
dc : DC-link index
ON, OFF : ON and OFF states indexes
L : Line or load indexes
g : Grid index
e : Electromagnetic index
D : Diode index
T : Transistor index

3.3. Abbreviations

SOC : System On Chip
ADC : Associated Discrete Circuit
A^3 : Algorithm Architecture Adequation
LUT : Look-Up Table
DSP : Digital Signal Processor
FPGA : Field Programmable Gate Array
HIL : Hardware-In-the-Loop
IP : Intellectual Property

VSI : Voltage Source Inverter
DR : Diode Rectifier
DC : Direct current
AC : Alternate current
MNA : Modified Nodal Analysis
DSMPC : Direct Sliding Mode Power Control
RAM : Random Access Memory
PWM : Pulse Width Modulation
MSPS : Mega Samples Per Seconde
ARM : Advanced Reduced instruction set computer Machines
I/O : Input / Output
CHIL : Controller Hardware-In-the-Loop
PHUL : Power Hardware-In-the-Loop
ADC : Analog to Digital Converter
DAC : Digital to Analog Converter
MMC : Modular Multi-level Converter
PV : Photovoltaic
IGBT : Insulated-Gate Bipolar Transistor
NPD : Natural Phase-Domain
FEM : Forward Euler Method
BEM : Backward Euler Method
ITM : Implicit Trapezoidal Method
VHDL : Very high speed integrated circuit Hardware Description Language
ACG : Automatic Code Generator
XSG : Xilinx System Generator
CPU : Central Processing Unit
FSM : Finite State Machine
VIO : Virtual Input Output
ILA : Integrated Logic Analyzer
ICON : Integrated Controller
EM : Exciter Machine
MG : Main Generator
kVA : Kilo Volt Ampere
VSR : Voltage Source Rectifier
XADC : Xilinx Analog to Digital Converter

Chapter 1

State of the art real-time simulation of electrical systems

1. Introduction

As discussed in the general introduction, it is intended by an FPGA-based embedded real-time simulator, an IP module that simulates the system to be controlled or a part of it. Then, such simulators are not only limited to the context of HIL testing but can also be embedded with the controller in the same FPGA device to ensure additional functions like observation, estimation, diagnostic or health-monitoring. From this research topic, one can extract several problematics: those linked to the real-time digital simulation of electrical systems, those linked to embedded simulators and those linked to the FPGA implementation. The state of the art is then organized into two chapters. This first one gives a general discussion about the real-time digital simulation and the second one focuses on the problematics of embedded simulators and their FPGA implementation.

Actually, to face today's industry demands in terms of performances, these electrical systems have been the focus of intensive researches starting from the power generation and storage until its consumption. In this context, a wide range of increasingly complex machine drives, power electronic converters and their controllers are now available. Moreover, this trend makes their design very challenging since the time-to-market and the development cost must also be considered. All these reasons make the real-time digital simulation of these electrical systems mandatory in modern design cycles.

Compared to a standard offline simulation, a real-time simulation consists in simulating the behavior of a system in a natural time by implementing its dynamical model. This is of course possible because of the ever increasing computation power of recent digital platforms which include powerful processing units (with multi-core structures) in addition to hardware platforms such as FPGAs. In contrast, including such simulation introduces additional design constraints during the development cycle. We can classify these constraints at three levels: the modeling of the system to be controlled, its digital realization and finally its digital implementation.

Before emphasizing these constraints, we start by making a comparison between a real-time simulation and its offline counterpart. This is followed by an overview of the latest application trends. Then the fourth section deals with the modeling of electrical systems where the formulation methods (model solvers) and their corresponding constraints are highlighted. Since these constraints depend on the nature of the system, a classification into two categories of elements has been made: electromagnetic elements (eg. transformers, AC/DC machines …) and switching elements such as power converters. In the fifth section the digital realization of these models is discussed. The most commonly used numerical solvers, the choice of the simulation time-step and the data representation are all investigated. Finally, the last section provides a study of the digital implementation starting by discussing the evolution of the commercially available digital real-time simulation platforms. Their advantages, limits and the contribution of FPGAs to boost their performances are all given.

2. Offline vs real-time simulation

The purpose of this comparison is to prop up the advantages of achieving a real-time digital simulation. To do so, it is worth noticing that both offline and real-time simulations are essential during the design cycle. The first one is generally achieved at the beginning of the process and aims to develop and functionally validate the model, in addition to its controller, with the help of simulation tools like Matlab/Simulink, PSpice, Saber, PSIM …etc. The real-time simulation is achieved at the end of the design cycle and aims to make fast and safe tests of the system, which is totally or partially replaced by its virtual real-time model.

2.1. Offline simulation

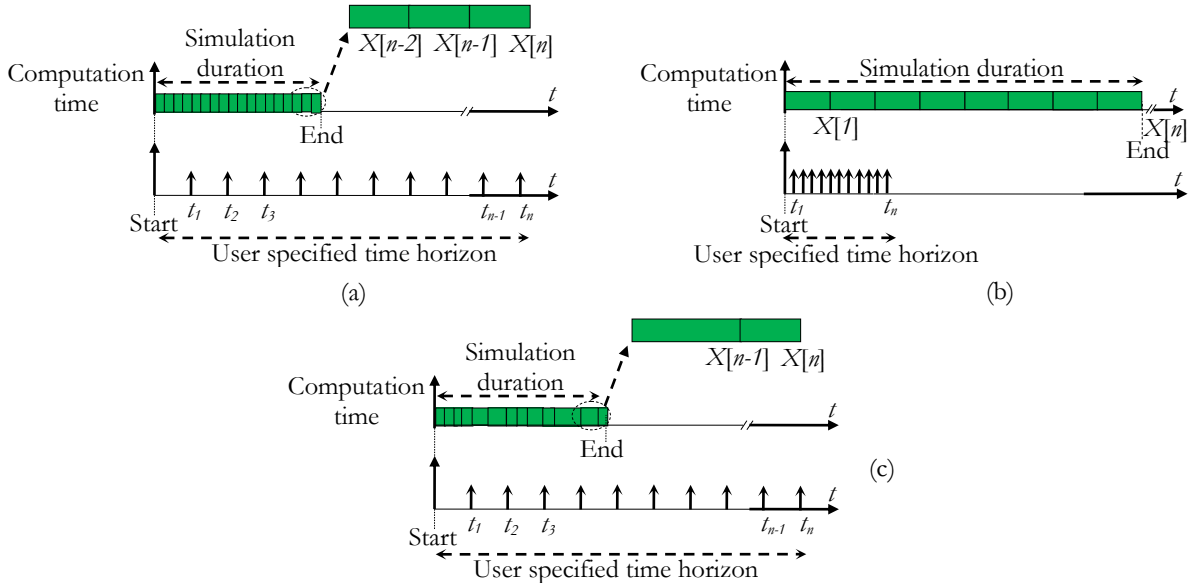


Figure 1. 1: Offline simulation – (a) Fixed-step simulation - duration shorter than the time horizon; (b) Fixed step simulation - duration longer than the time horizon; (c) Variable-step simulation

The digital offline simulation consists in solving at each time-step the mathematical equations representing the system model. In this case, the amount of time needed to solve these equations at each step doesn't matter. The most important is how to achieve the overall simulation (for a specified time horizon) as rapidly and accurately as possible. Thus, the total time needed to make a simulation can be shorter or longer than the specified time horizon, as shown in Figure 1.1-(a,b). This is mainly due to the complexity of the model, the complexity of the used numerical solver and also the processing performances of the used digital platform.

The computation can be achieved with variable-step numerical solver or fixed-step numerical solver. With the first one, the time-step is dynamically adjusted depending on the model dynamics, Figure 1.1-(c). This is typically true for electrical systems where several dynamics can be encountered in the same model: electrical dynamics, mechanical dynamics, power converters switching dynamics and also in some niche applications thermal dynamics. In this case, at each time-step, the solver computes the state value and determines the local error. With regards to the user specified local error tolerance, the time-step is then as necessary increased or decreased. The advantages in this case are to ensure a high level of accuracy and to accelerate the simulation. This is the reason why variable-step based models are generally considered as reference models (eg. those of the Matlab SimPowerSystems toolbox), used as benchmark to validate the obtained real-time results (eg. [ORI06], [RAJ05], [ABD09], [ZHA05], [MAT09], [MAT10], [MAT13]). In contrast, additional efforts have to be made by the designer to determine analytically the acceptable error tolerances.

In the case of fixed-step solvers, the time-step remains constant during the overall simulation. Even if the simulation duration may be enlarged, such solvers remain suitable for applications where the models are synchronized and also applications where automatic code generation is targeted (eg. Matlab/Simulink code generator),[MEN10].

2.2. Real-time simulation

The real-time digital simulation consists in solving the mathematical equations representing the discrete-time system model in a natural time. This is done at each time-step which is commonly accepted to be fixed (a variable-step is hardly usable in real-time because of the time management complexity). Unlike in offline simulation, the instant when results are produced at each time-step is very important. The fundamental challenge is then how to choose the appropriate time-step to simulate accurately the system dynamics and still maintaining the real-time operating condition (to avoid time violation or overruns), see Figure 1.2. Thus, the computation time at each step must be shorter than the chosen time-step.

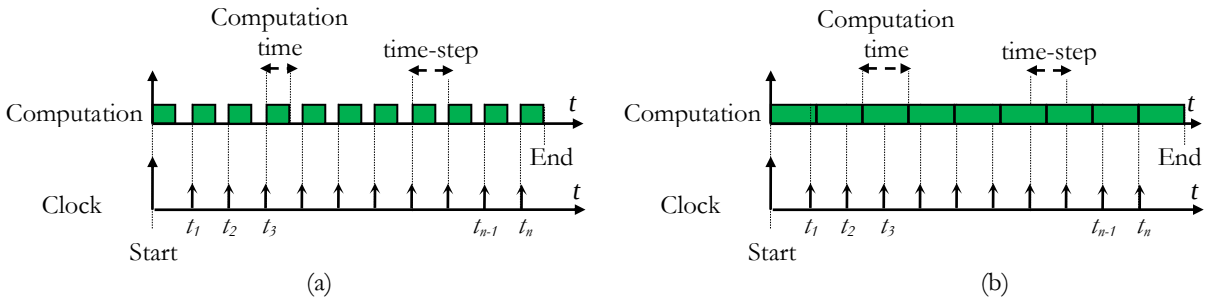


Figure 1. 2: Real-time simulation cases. (a) Real-time operation; (b) Overruns

3. Real-time digital simulation of electrical systems - Applications trends

In addition to rapid prototyping applications, the real-time simulation is systematically known to be applied during the design process to make HIL tests, [BEL-10]. This is an intermediate validation step between a fully computer-based development (offline simulations) and a fully experimental test. The system is then totally or partially replaced by a real-time simulator implemented in a digital simulation platform. Three main parts can then be defined (Figure 1.3):

- A hardware part that includes the controller and possibly a part of the actual system to be controlled (if it is not totally real-time simulated).
- The digital simulation platform where the real-time simulator is implemented. Note that this same platform can be used to validate the controller.
- The interface between the two first parts.

From these parts, we can classify the HIL simulation into two main categories: the Controller HIL (CHIL) and the Power HIL (PHIL).

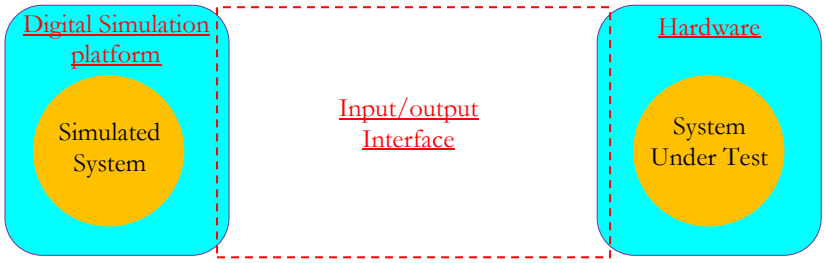


Figure 1. 3: General structure of a HIL simulation

3.1. Controller HIL simulation

The structure of a typical CHIL simulation is presented in Figure 1.4. The hardware part contains only the controller and the power system is fully modeled and simulated by the digital simulation platform. Low power/level signals are exchanged in this case. To take into account the effects of the acquisition chain, Analog to Digital and Digital to Analog Converters (ADC, DAC) associated with conditioning circuits can be used. When power converters are used in the simulated system, the interfacing signals are the control signals. It should also be noted that a CHIL can be achieved in the same hardware device where both the controller and the real-time system model are implemented and the interfacing is then inherent to the device (fully digital interfacing).

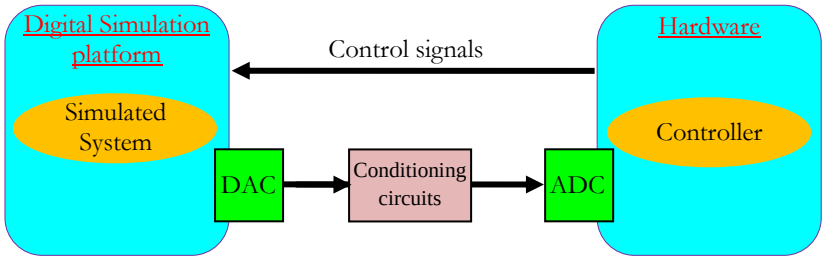


Figure 1. 4: Structure of a CHIL simulation

A CHIL allows a first realistic validation and provides first experimental operating guarantees of the overall electrical system. User can then validate the developed controller under a wide range of realistic conditions including safe operating and fault operating conditions. This has the credit of minimizing the development time/cost and the damage risks. In this context, a wide range of applications are provided in the literature. As examples, in [SEU12], control algorithms for a DC-DC buck converter and DC-AC converter have been HIL validated by connecting them to the RTDS digital simulation platform. In [CAM13], HIL simulation platform based on LABVIEW from Nation Instruments has been used for the verification and validation of digital controllers for power converters.

3.2. Power HIL simulation

As for a PHIL simulation, a part of the power system is real-time simulated and the other part remains actual. The power exchange between these parts is ensured by power amplifiers. The basic setup of a PHIL simulation is illustrated in Figure 1.5.

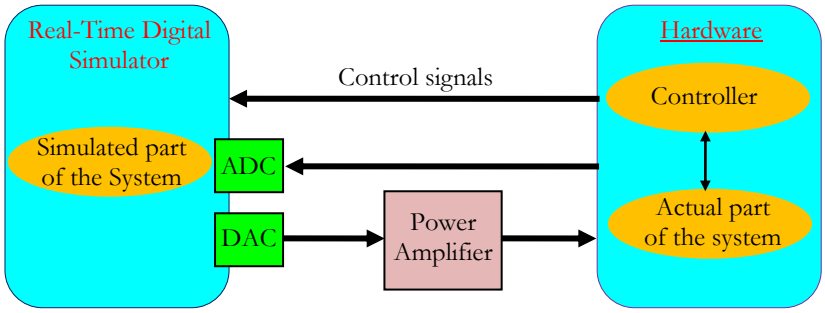


Figure 1. 5: Structure of a PHIL simulation

A very interesting application example is provided in [L2EP] where the aim is to control and manage the electrical power in a distributed energy platform. The latter contains a set of heterogeneous sources (eg. wind turbine, batteries, gaz turbine, PV panels, super-capacitors for storage ...), heterogeneous power converters (DC/DC, AC/DC, CD/AC, MMC ...) and

loads (eg. motor drives, electrical/hybrid vehicles ...) all interconnected through a power network (AC and DC). A part of these elements (eg. wind turbine, PV panels, electrical/hybrid vehicles, power converters, power network …) are simulated using RT-LAB real-time simulation platforms and power amplifiers are used to interface them with the actual parts (see Figure 1.6).

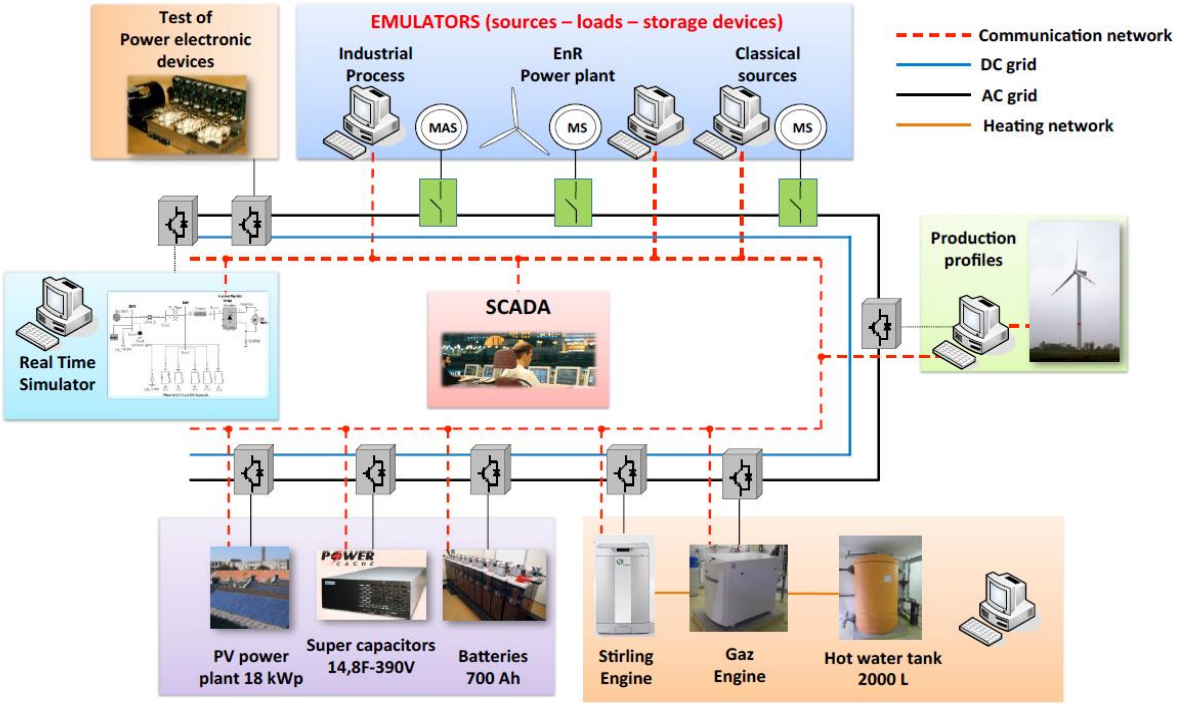


Figure 1. 6: Distributed Energy Lab - Source: [L2EP]

4. Modeling and representation of electrical systems

The starting point that influences the real-time simulation quality of an electrical system is the developed model solvers. The latter must be formulated so as to represent accurately the static and dynamic behavior of the actual system. This accuracy is also linked to the hypotheses that consider or neglect physical phenomena that occur in the system (eg. saliencies, saturations, core losses, power converter commutations, skin effects …). In contrast, designer has to be aware that the more accurate is the model, the higher is its complexity and thus the more constraining is its real-time simulation.

An electrical system can be described as a set of interconnected elements that have different characteristics and dynamics starting from fast to slow dynamics (power converter dynamics, electrical dynamics, mechanical dynamics and thermal dynamics).These elements have been classified in this work into two categories: electromagnetic elements and switching elements, see Figure 1.7.

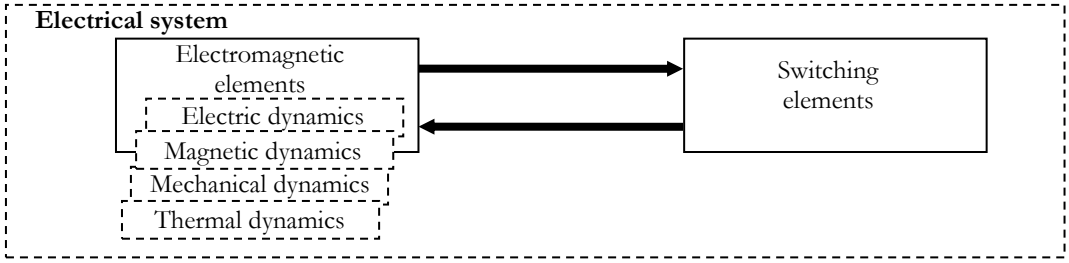


Figure 1. 7: Components of an electrical system

The first category gathers electromagnetic elements where electric, magnetic phenomena are operated in addition to mechanical phenomena (for moving systems) and potentially thermal phenomena. Depending on the application, these elements may contain power sources which can be provided by batteries, generators or a power grid, transformers or auto-transformers, filters, electromechanical load (DC or AC machines) and mechanical load.

The second category concerns switching elements such as power electronic converters. These elements are composed of power switches such as diodes, IGBTs and thyristors. Depending in the nature and level of the modulated power, several types of power electronic converters with different topologies can be encountered: DC-DC converters, DC-AC converters, AC-DC converters; multi-level, multi-phase, matrix converters...

4.1. Model solvers

Different types of model representations can be found in the literature. In the context of real-time simulation, one can underline the state-space and nodal solvers.

a- State-space solver

A state-space solver consists in representing the system differential equations as:

$$\begin{cases} \dot{x} = f(x, u, t) \\ y = h(x, u, t) \end{cases} \tag{1.1}$$

Where x is the state-space vector, u is the input vector, y is the output vector, f is the state function and h is the output function. For linear time-invariant systems, these equations are written in a matrix form:

$$\begin{cases} \dot{x} = A.x + B.u \\ y = C.x + D.u \end{cases} \tag{1.2}$$

Where A is the state matrix, B is the input matrix, C and D are the output matrices (D is null for many physical systems). The size of these matrices depends on the order of the system (number of state variables).

The state-space solvers are generally adapted to electromagnetic elements since differential equations can be easily extracted, [OUL10][MYA11]. This is not the case for switching elements. Indeed, for a given topology and a given number of switches, several state-space equations have to be extracted and their number depends on the number of combinations, formed by the states of the power switches, [MAJ11]. The issue here is the complexity, thus the necessary computational time, the computation resources and also the synchronization between all these equations when moving from a switches combination to another. To overcome this issue, it is possible to unify these state equations by linearizing them around a specific operating point or developing a state-space average model [REZ13]. This will however influence the model accuracy.

b- Nodal solver

This solver is mostly used to simulate electrical circuits (EMTP, SPICE…). When a system is modeled as an equivalent electrical circuit, it is possible to make a nodal analysis (based on Kirchhoff's law and Ohm's law) to extract the relations between voltages and currents. These relations can be extracted as a matrix equation by applying the so-called Modified Nodal Analysis (MNA), [PEJ94]. This consists in extracting this equation as a vector *x* of unknown variables (voltages/currents) equal to the inverse of a conductance matrix *H* multiplied with a vector *b* of known variables (voltage/current sources). The size (or the order) of this equation depends on the number of the specified circuit nodes. Relation 1.3 presents the general form of this matrix equation. Its extraction methodology is discussed in Appendix-A.

$$x = \begin{bmatrix} x_{11} \\ x_{21} \\ \vdots \\ x_{n+m,1} \end{bmatrix} = H^{-1}.b = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1,n+m} \\ h_{21} & h_{22} & \cdots & h_{2,n+m} \\ \vdots & \vdots & \vdots & \vdots \\ h_{n+m,1} & h_{n+m,2} & \cdots & h_{n+m,n+m} \end{bmatrix}^{-1} . \begin{bmatrix} b_{11} \\ b_{21} \\ \vdots \\ b_{n+m,1} \end{bmatrix} \quad (1.3)$$

Such nodal approach is also widely used in electromagnetic transient applications, (eg. [DOM69]) and is adapted to large scale and complex electrical systems. Furthermore, this approach is appropriate for switching systems where an accurate individual modeling of switches is required and the extraction of the matrix equation is easily and quickly achieved.

4.2. Modeling of switching elements

Modeling of switching elements for real-time simulation is a very challenging task. This is, here again, due to the need of models that satisfy a compromise between two contradictory requirements; higher accuracy and lower complexity. In this context, two modeling approaches can be defined: the modeling at the system scale and the modeling at the switch scale.

a- Modeling at the system scale

This approach consists in considering the whole converter as a transfer function linking its inputs and outputs. Although it provides low complexity and simple models, this approach suffers from accuracy limitation since it does not reproduce realistically the switches nonlinearities. In the following, the most widely used system scale modeling methods are presented.

- Switching function modeling

It consists in representing a converter by a set of switching functions, [SAL94][BYO01][MAR06]. The relationships between voltage-current outputs and inputs are then given by:

$$\begin{cases} [V_{out}] = [f_{sw1}].[V_{in}] \\ [I_{out}] = [f_{sw2}].[I_{in}] \end{cases} \quad (1.4)$$

Where, V_{out} and V_{in} are the vectors of output voltages and input voltages, respectively. I_{out} and I_{in} are the vectors of output currents and input currents, respectively. f_{sw1} and f_{sw2} are the switching functions which depend on the control signals C_i.

As an example, the modeling of a 3-phase inverter (Figure 1.8) gives the following equations:

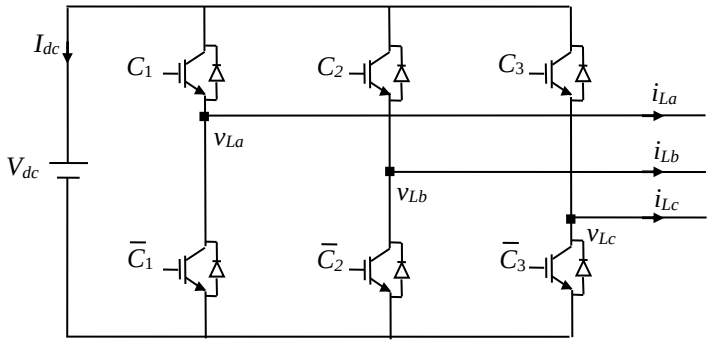


Figure 1. 8: Circuit diagram of a 3-phase inverter

$$\begin{cases} [V_{out}] = \begin{bmatrix} v_{La} \\ v_{Lb} \\ v_{Lc} \end{bmatrix} = \frac{1}{3}\begin{bmatrix} 2.C_1 - C_2 - C_3 \\ -C_1 + 2.C_2 - C_3 \\ -C_1 - C_2 + 2.C_3 \end{bmatrix}.[V_{dc}] = [f_{sw1}].[V_{in}] \\ [I_{out}] = [I_{dc}] = [C_1 \quad C_2 \quad C_3].\begin{bmatrix} i_{La} \\ i_{Lb} \\ i_{Lc} \end{bmatrix} = [f_{sw2}].[I_{in}] \end{cases} \tag{1.5}$$

This modeling method is highly dependent on the converter topology. It is then not suitable when complex topologies (eg. Multi-level converter, Matrix converters) are considered. It is also worth noticing that this method is not able to represent anti-parallel natural switching diodes and therefore, for the proposed example, the rectifier mode is not ensured [SYB06]. To cope with this problem, an efficient solution consists in describing the behavior of the converter using a state machine. This solution has been applied in [RAK11] to design a switching function model of three-level neutral-clamped that support rectifier mode.

- Averaged modeling

An averaged modeling approach considers only the average value of the system variables in addition to the control signals duty cycle [JIN97], [KIF11]. Generally, the averaging is made over one switching period and the oscillations (due to commutations) are not considered leading to a relatively simple model.

Staying with the same example as before, to obtain the average model of a 3-phase inverter, the moving average equation defined by relation 1.6 is applied to the equations 1.5.

$$\langle variable \rangle = \frac{1}{T}\int_{kT}^{(k+1)T} variable(t).dt \tag{1.6}$$

The obtained equations (over one switching period) are then:

$$\begin{cases} \begin{bmatrix} \langle v_{La} \rangle \\ \langle v_{Lb} \rangle \\ \langle v_{Lc} \rangle \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 2\langle C_1 . V_{dc} \rangle - \langle C_2 . V_{dc} \rangle - \langle C_3 . V_{dc} \rangle \\ -\langle C_1 . V_{dc} \rangle + 2\langle C_2 . V_{dc} \rangle - \langle C_3 . V_{dc} \rangle \\ -\langle C_1 . V_{dc} \rangle - \langle C_2 . V_{dc} \rangle + 2\langle C_3 . V_{dc} \rangle \end{bmatrix} \\ [\langle I_{dc} \rangle] = [\langle C_1 . i_{La} \rangle + \langle C_2 . i_{Lb} \rangle + \langle C_3 . i_{Lc} \rangle] \end{cases} \quad (1.7)$$

Note that the following simplifications can be applied when the averaging is made over one switching period.

$$\begin{cases} \langle C_i . V_{dc} \rangle = \langle C_i \rangle . \langle V_{dc} \rangle \\ \langle C_i . i_i \rangle = \langle C_i \rangle . \langle i_i \rangle \end{cases} \quad (1.8)$$

The main drawback of such approach is that the commutation events are not covered during a switching period which limits the bandwidth and the accuracy of the model. An alternative to overtake this issue consists in averaging the model over one simulation time-step instead one switching period [LIA05]. This is possible of course since the simulation time-step is typically much lower than the switching period.

b- Modeling at the switch scale

This approach consists in modeling each switch individually. Unlike system scale modeling, it allows representing the turn-on and turn-off nonlinear characteristics, the switching power losses [MYA11], and even the thermal characteristics [KRU99] of the switches. In the following, the most widely used switch scale modeling methods are presented.

- Physical modeling

It consists in modeling each switch by its physical model. This latter is generally based on semiconductor physics describing the charge carrier distribution in the device. As an example, the most popular physical models used for IGBTs modeling are the Hefner [HEF88] and Kraus [KRA93] which have been implemented in offline simulators such as SABER and SPICE. This method is well-known for its notable accuracy. However, its use remains limited in real-time simulation because it leads to complex models which are difficult to be implemented in real-time.

- Behavioral modeling

It consists in representing each switch by its voltage and current characteristics obtained either from fitting curves [BLA96] or experimentations [MYA11]. The converter model is then represented by a state machine which interacts at high rate with memories where these characteristics are stored. As an example, in [MYA11], a device-level behavioral model of an IGBT-based multilevel voltage source converter has been implemented. In this example, the voltage and current characteristics of the IGBT module (CM50DU-24F from Powerex) are obtained from an experimental setup (Figure 1.9, extracted from [MYA11]).

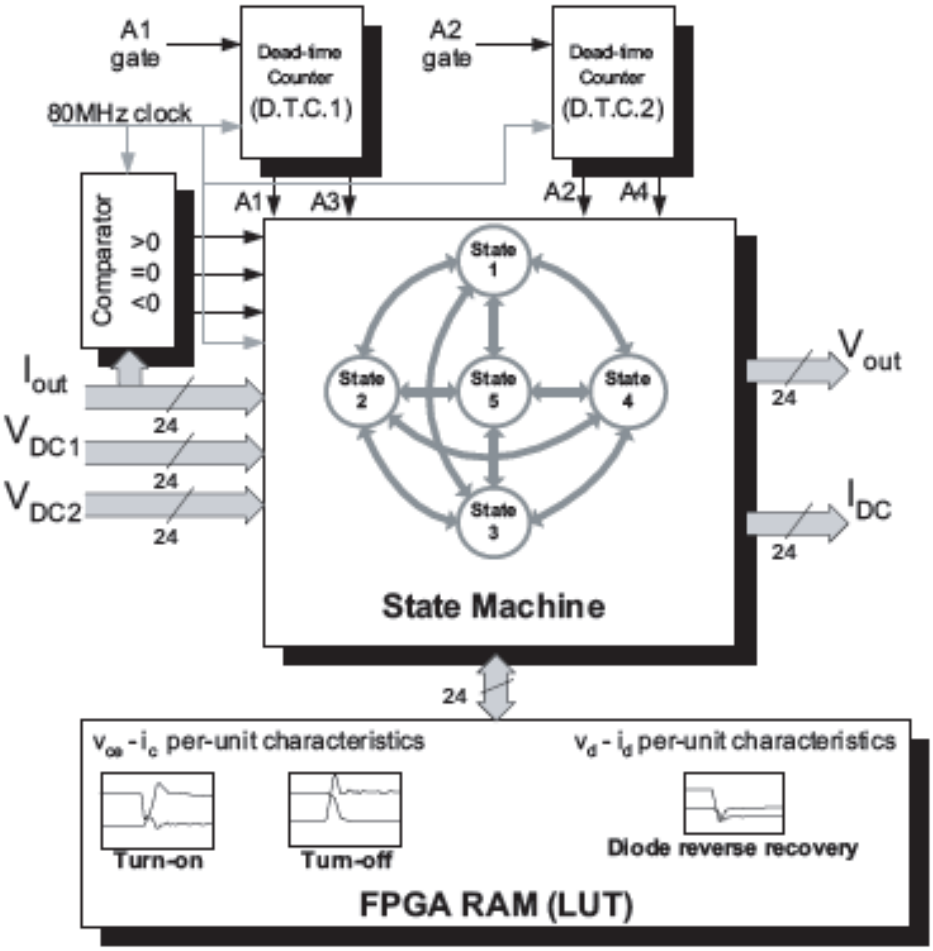


Figure 1. 9: Functional block diagram of one 3-level voltage source converter leg, extracted from [MAY11]

Compared to physical modeling, a behavioral modeling provides models with almost similar accuracy and low complexity. However, this method seems to be unsuitable for complex converter topologies where modeling all possible operating modes by a state machine is difficult.

- Two-valued switch modeling

This method consists in modeling each switch by a variable impedance, controlled by a switching logic having as inputs the switch voltage and current, and the control signal. One can distinguish two modeling approaches here: the small/large resistor model and the Associated Discrete Circuit (ADC) model. In the following, each of these models is presented.

- Small/large resistor model

It consists in modeling a switch by a small resistance R_{ON} when it is closed and a large resistance R_{OFF} when it is open [WAT03] (see also Figure 1.10). Careful choices are necessary for the values of these resistances in order to minimize numerical instabilities and to avoid therefore problems of convergence.

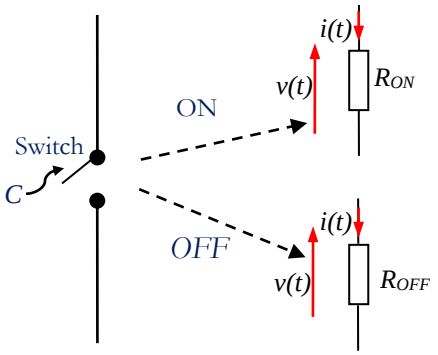


Figure 1. 10: Small/large resistor representation

Such a model is applied in many offline simulators such as Spice and PSCAD. It offers several advantages such as its ability to be used for short circuit and open circuit fault simulations. However, in real-time, this method require many time consuming mathematical operations (in particular matrix inversion) since the converter equations are strongly dependent on the state of switches and need to be modified at every simulation time-step [WAT03]. To overtake this issue a solution consists in pre-computing and storing each of equations that correspond to each of topologies, which in contrast requires a significant amount of memory to store all equations.

- ADC-based model

It consists in representing a switch by an inductance (L_{sw}) when it is ON and by a capacitance (C_{sw}) when it is OFF [PEJ94], [HUI94], [MAT11IPST], [MAT10]. In both cases (ON/OFF), a switch can then be represented by a conductance G in parallel with a dependent current source $J[k]$. The principle is shown in Figure 1.11.

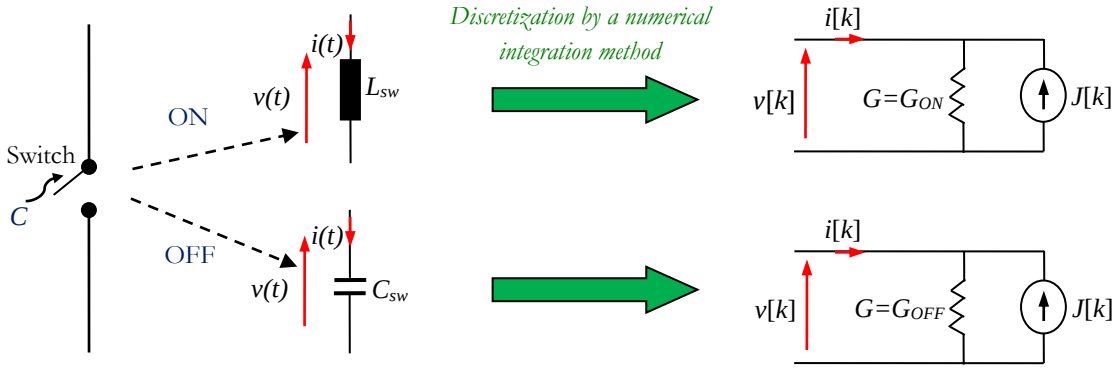


Figure 1. 11: Principal of ADC-based model

Unlike small/large resistor model, it is possible, by properly selecting the values of C_{sw} and L_{sw}, to obtain equations completely independent of the state of switches and thus a constant conductance matrix that can be inverted one-time and offline. This provides an efficient solution to the processing limitations mentioned above and allows the reduction of the model complexity without downgrading the accuracy. This model has been widely used in the literature and it is the main switching element model adopted in this work. It should however be noted that this approach introduces overshoots and oscillations due to LC circuit that can be resolved by adding damping elements. This point will be deeply discussed in chapter 4.

4.3. Modeling of electromagnetic elements

The discussion will be limited here to the case of AC machines (especially the 3-phase machines noting that the presented modeling approaches can be adapted to polyphase machines). In the literature, the modeling approach, that is taken as a reference, is based on the Natural Phase-Domain (NPD), here 3-phase domain. To optimize the complexity, Bi-phase domain models can be deployed. These models are derived after frame transformations.

a- Natural phase-domain model

A NPD model consists in modeling an AC machine by a set of resistances in series with a set of time varying self and mutual inductances, representing its variables in their physical form and solving its equations in the natural phase coordinates [DEH05][LAG06]. In the case of a 3-phase machine, the relationship between the machine voltages and currents is given by:

$$[v_{abc}(t)] = [R].[i_{abc}(t)] + \frac{d[\Phi_{abc}(t)]}{dt} \tag{1.9}$$

$$[\Phi_{abc}(t)] = [L(\theta, t)].[i_{abc}(t)] \tag{1.10}$$

Where $v_{abc}(t)$, $i_{abc}(t)$, and $\Phi_{abc}(t)$ are the vectors of machine voltages, currents, and flux linkages respectively. θ is the rotor position. R is a diagonal matrix of the machine winding resistances and $L(\theta,t)$ is a symmetrical matrix of the machine winding self and mutual inductances.

The main advantage of this model is that the stator circuit is directly interfaced to the external power network which is also represented in physical 3-phase coordinates. This avoids the need to additional interfaces (eg. coordinate transformations) and avoids numerical instabilities that can arise due to the delay introduced by this interface [DUF07PESC]. In addition, such way of modeling can easily provide more accurate representation of machine internal phenomena, such as internal machine faults [MUT01], electrical and mechanical harmonics and saturation effects [MAR97], [QUE12].

However, since the self and mutual inductances across stator and rotor windings are time-variant, the inductance matrix is not constant. Thus, the overall machine admittance matrix needs to be upgraded and inverted at every simulation time-step as the inductances change. This is the main drawback of this model since it leads to increase the model complexity.

b- Bi-phase model

A Bi-phase AC machine model consists in representing the differential equations either in fixed frame (α,β) associated to the Clarke/Concordia transformations or in rotating frame (d,q) associated to the Park transformations [DEH05][MAT11]. The continuous-time differential equations representing its electrical subsystem in fixed or rotating frame can be then expressed as:

$$\left[v_{dq\ or\ \alpha\beta}(t)\right] = [R].\left[i_{dq\ or\ \alpha\beta}(t)\right] + [L].\frac{d\left[i_{dq\ or\ \alpha\beta}(t)\right]}{dt} + [E(t)] \tag{1.11}$$

Where $v_{dq\ or\ \alpha\beta}$ is the vector of machine voltages, $i_{dq\ or\ \alpha\beta}$ is the vector of machine currents, R is the matrix of machine resistances, L is a constant matrix of machine inductances, and E is the speed voltage vector.

Compared to a natural phase-domain model, the bi-phase model allows reducing the number of equations and their complexity since the inductance matrix becomes constant. However, the simulation of this model requires adding interfaces (frame transformations) for connecting it to the rest of the power network, Figure 1.12. This is in fact the main drawback of this model since these interfaces are the subject of interfacing errors which may deteriorate the accuracy and may cause convergence problem especially when larger time-steps are used [GOL84], [WAN10]. This is why, when using this model, the used simulation time-step has to be adequately smaller than 50µs ([DEH05]) in order to keep the interfacing error within an acceptable range and avoid convergence problems, [DEH05], [MAT11].

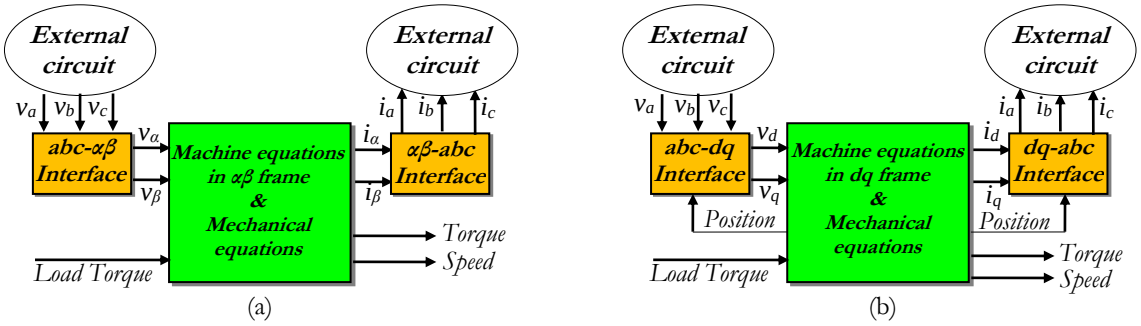


Figure 1. 12: Bi-phase model (a) Model in fixed frame (α,β) (b) Model in rotating frame (d,q)

5. Digital realization

Besides its mathematical modeling, the real-time simulation of an electrical system is also conditioned by the digital realization of the chosen models. This consists in choosing the appropriate numerical solver, the appropriate simulation time-step and then the numerical representation of the processed data. Here again, the choice has to consider several criteria in terms of complexity (number of operations to implement), accuracy (with regards to the original continuous-time model), numerical stability and respect of real-time operation (computation time less than the time-step).

5.1. Numerical solvers

The numerical solvers use a set of integration algorithms that approximate (based on Taylor series) the continuous-time differential equations of the model. The approximation order has an important impact and introduces truncation errors that should be put in mind, [RAK14]. When limiting the discussion to first order approximation, the integration methods can be divided into two categories: the explicit and the implicit methods.

a- Explicit methods

The explicit methods are based on the principle that each state variable of a system is computed at the current time-step only from its previous values. A relevant example of such methods is the Forward Euler Method (FEM). The latter consists in approximating the derivative term (Laplace operator) by the following:

$$s \to \frac{z-1}{\Delta t} \tag{1.12}$$

Where, Δt is the simulation time-step and z is the shift operator.

The FEM is known for its simplicity since it requires few arithmetic operations. However, the main drawback is related to the numerical stability. Indeed, the latter is influenced by the simulation time-step that may not satisfy the following stability condition.

$$\sqrt{(1+\Delta t.Re(p_s))^2+(\Delta t.Im(p_s))^2} < 1 \tag{1.13}$$

Where, $Re(p_s)$ is the real part of the most critical pole of the continuous-time model and $Im(p_s)$ is its imaginary part. To ensure the stability, the time-step must then satisfy the following condition:

$$\Delta t < \frac{2}{|p_s|} \tag{1.14}$$

As illustrated in Figure 1.13, with a FEM, a stable continuous-time model may become unstable after its discretization when the time-step is not properly chosen.

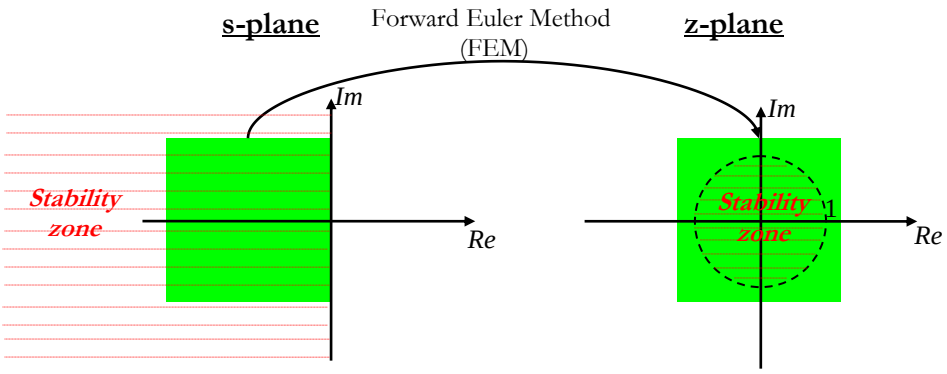


Figure 1. 13: Stability of the Forward Euler Method

b- Implicit methods

The implicit methods are based on the principle that each state variable is processed by a combination of its current value and the previous values. Here, the Implicit Trapezoidal Method (ITM) and the Backward Euler Method (BEM) are mostly deployed.

The ITM consists in approximating the derivative operator by the following:

$$s \to \frac{2}{\Delta t}.\frac{z-1}{z+1} \tag{1.15}$$

Despite the additional complexity, this method is widely used for the real-time simulation thanks to its accuracy and stability features, [DIN01][JIA14], [CHA04], [MAY11]. In fact, this method is called A-stable, [MAR89], which means that it is unconditionally stable because the condition below is always true (of course when the continuous-time pole is stable, $Re(p_s)<0$) :

$$\frac{\sqrt{(2+\Delta t.Re(p_s))^2+(\Delta t.Im(p_s))^2}}{\sqrt{(2-\Delta t.Re(p_s))^2+(\Delta t.Im(p_s))^2}}<1 \qquad (1.16)$$

As Figure 1.14 illustrates, a stable continuous-time model remains always stable after its discretization with an ITM.

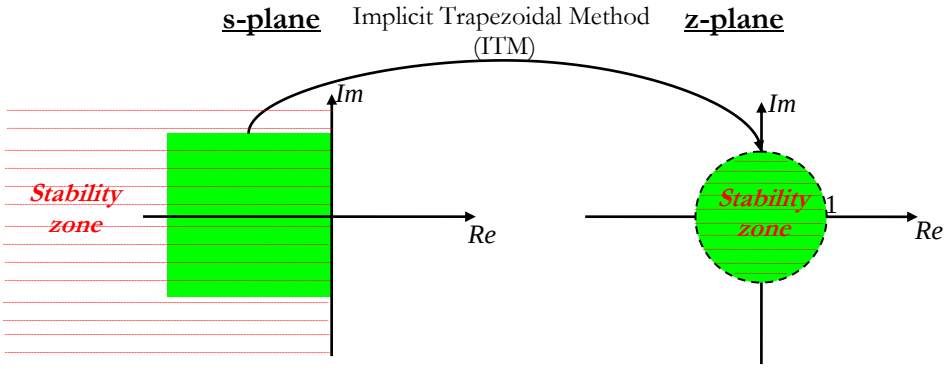


Figure 1. 14: Stability of the Implicit Trapezoidal Method

However, the ITM is not always suitable because of numerical oscillations. This is typically true when the system model contains combinations of L-C circuits such as the ADC-based model of a power converter [PEJ94]. The example of a DC-DC converter is emphasized in Figure 1.15. The ON state (for sw1) and OFF state (for sw2) give the following equivalent circuit:

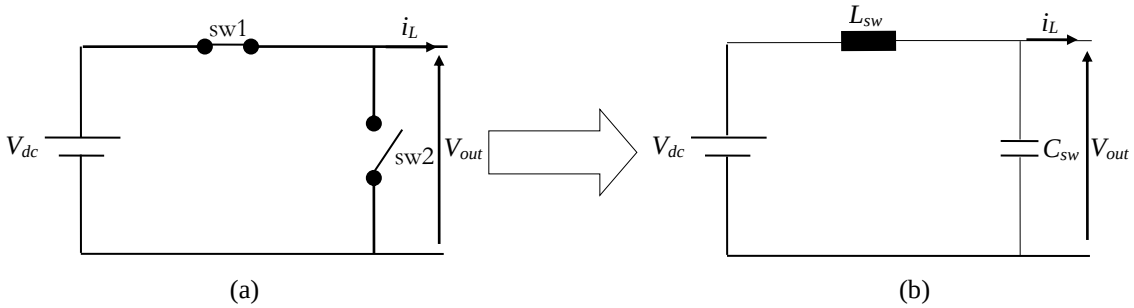


Figure 1. 15: (a) DC-DC converter circuit after switching (b) its equivalent ADC-based circuit

As Figure 1.16 illustrates, when the ITM is used, the discrete-time voltage response V_{out} has an oscillating behavior since the corresponding poles are placed on the unit circle of the z-plane.

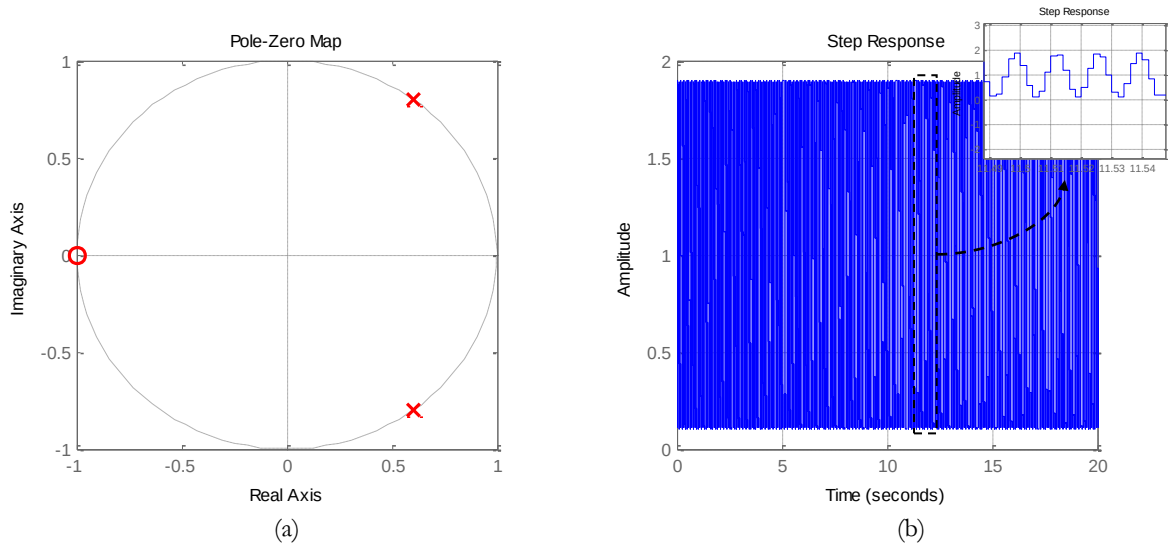


Figure 1. 16: Results when the ITM is used (a) Location of the poles and zeros in the z-plane (b) Step response

To avoid such oscillations, the use of the BEM is suitable since it has a numerical damping property, [MAR89], see also Figure 1.17.

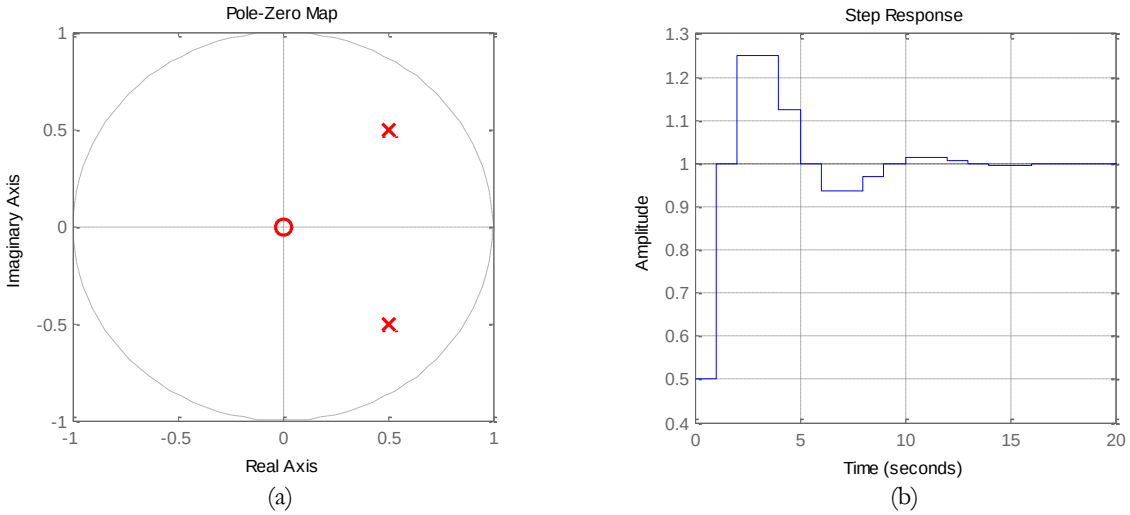


Figure 1. 17: Results when the BEM is used (a) Location of the poles and zeros in the z-plane (b) Step response

The Backward Euler method consists in approximating the derivative term by the following:

$$s = \frac{z-1}{\Delta t.z} \tag{1.17}$$

In addition to this damping property, this method is of course less complex than the ITM one and has the same stability feature since it is also unconditionally stable (the following condition is always true as far as $Re(p_s)$ is negative):

$$\sqrt{(1-\Delta t.Re(p_s))^2 + (\Delta t.Im(p_s))^2} > 1 \tag{1.18}$$

As Figure 1.18 illustrates, a stable continuous-time model remains always stable after its discretization with a BEM. However, to achieve a level of accuracy equivalent to ITM, a BEM based model needs a much smaller time-step, [MAR89].

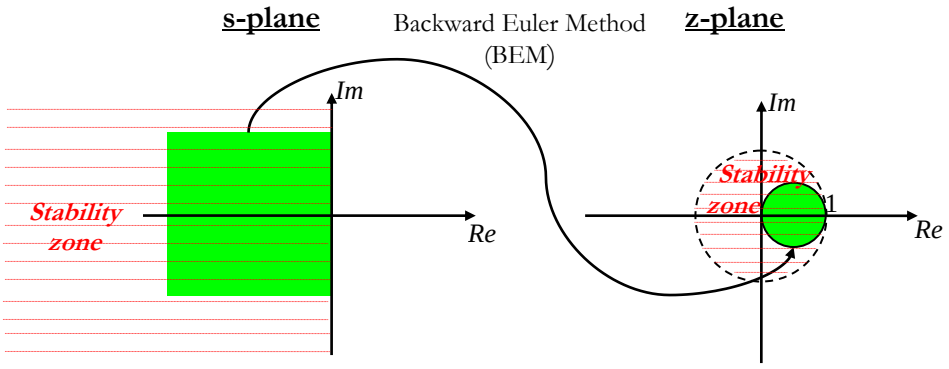


Figure 1. 18: Stability of the Backward Euler Method

5.2. Simulation time-step selection

To make a real-time simulation, the fundamental constraint is the choice of the appropriate simulation time-step. The latter has to be long enough to allow the processing of all model equations but at the same time short enough to accurately represent the system. In the following, we are addressing some of the most important selection criteria and requirements to take into account.

a- Time-step vs system dynamics

When an electromagnetic element is of concern, it is commonly accepted to choose a time-step from 5% to 10% of the smallest time constant. As for switching elements, the choice is more related to the topology of the converter and to how this converter is modeled. Basically, the simulation time-step should be at least 10 times smaller than the switching period, [BEL10], but as will be seen in the following, this value has to be smaller.

b- Time-step vs interfacing errors

Another requirement is to reduce the errors introduced by the inter-simulation time-step events. This issue has been discussed in detail in [NOD11] and [MAT10]. Indeed, when it is interfaced with a controller (eg. CHIL application), the power converter real-time simulator can respond to its control signals only at each simulation time-step (Figure 1.19). As result, switching events that occur between two steps cannot be detected and therefore errors are introduced to the simulation results [DIN00]. These additional errors can be avoided when the time-step is highly reduced with regard to the switching period. According to [MAT10], the time-step should be at least 100 times the switching period.

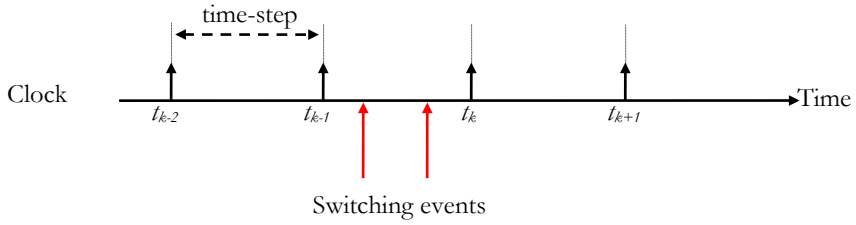


Figure 1. 19: Inter simulation time-step issue

c- Time-step vs numerical stability

The numerical stability has to be also taken into account. In this direction, whatever the used numerical solver (even if it is unconditionally stable), when the time-step decreases, the poles of the discrete-time model move closer to the stability limit (|z|=1). Due to their location, poles are therefore very sensitive to variations caused for example by limited precision data quantification, [LI93]. These variations may then lead to instability especially when fixed-point format is used. In the chapter 3, this problem will be deeply discussed and an efficient solution based on delta-operator transformation will be presented. The main advantage of such solution remains in the possibility to use a very small time-step without affecting the stability. Indeed, with the delta-operator approach, when the simulation time-step decreases, the stability zone increases and becomes closer to the stability zone of the s-domain.

d- Time-step vs real-time operation

The simulation time-step must be short enough but at the same time long enough to allow the processing of the model equations. To guarantee the real-time operation and avoid overruns, the processing time has then to be less than the chosen time-step.

5.3. Numerical Data Representation

Regarding the numerical data representation, the floating-point or fixed-point representations can be adopted. The first one is used when higher data precision is needed but the price to pay is the complexity of floating-point operations and thus the computation time. In [JIA14], a floating-point FPGA-based real-time nonlinear hysteretic transformer model is developed and implemented (due to its complexity) on a high computational resources Virtex-7 FPGA device. For this application, floating-point data representation requires 3 to 4 computation resources more and is 3 to 4 times slower than its fixed-point counterpart.

To reduce the computation time and optimize the computation resources, a fixed-point format is more suitable. In [SAN12], HIL simulations of a boost power converter have been made with different data representations. It has been shown that using fixed-point representation, HIL testing can be speeded. Even if the main drawback of a fixed-point format is its accuracy, a proper selection of the bit-width and location of the binary point can provides sufficient computation accuracy, [MAT09][MAT11]. In this work, the fixed-point, rather than the floating-point data representation, is adopted.

6. Digital implementation

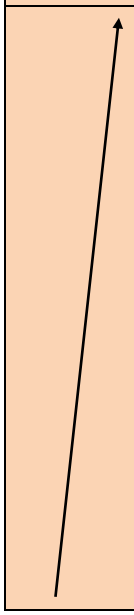
Now as the importance of selecting the right model, the right numerical solver, the right simulation time-step and the right data format has been underlined, the final step that influences the performances of a real-time simulation is the digital implementation.

By this way, the real-time digital simulation technologies have been the focus of intensive progress to meet several demands. From the application point of view, the main challenge is then how to find the appropriate platform that is able to support the complexity of the simulated system and at the same time ensure the required timing performances (simulation time-step with regards to frequency of the highest system dynamic to simulate).

6.1. Evolution of real-time digital simulation technologies

The first real-time digital simulators have been proposed during the 80's and were based on yesteryear's microprocessors and DSPs. Their performances have considerably grown to conciliate the always increasing industrial demands in terms of flexibility, cost optimization, scalability in addition to timing and computation power.

Table 1. 1: Examples of commercialized real-time simulation platforms and their application

System scale (complexity)	Example of applications	Time-step range	Examples of Real-time simulation plateforms		
			OPAL-RT		RTDS Technologies
			Real-time simulator	Used processor	
	Very large power grids (country scale): buses, generators, distribution network, loads … *RMS values computation*	~ 1ms	ePHASORSim (up to 30000 nodes)	Combinations of OP5600 chassis: - Intel ATX mother board - up to 12 cores - 3.3 GHz - QNX or Red Hat Linux OS	Combinations of customizable processing units organized in racks (up to 144 nodes per rack): - PB5 processor board per rack - 2 Freescale PowerPC MC7448 RISC processors per board - 1.7 GHz
	Large power grid, HVDC systems, FACTS, micro-grids, wind-farms … *Fundamental frequency + electromagnetic transients*	Down to 10µs	HYPERSIM (up to 10000 nodes)		
			eMEGASim (up to 1200 nodes)		
	AC/DC motor drives, power converters, batteries …		eDRIVESim		

Then, to boost these first digital simulators, computer-based simulators have appeared so as to cover, on the one hand the simulation of large-scale systems (using supercomputer-based simulators, eg, Hydro Quebec's HYPERSIM or RTDS) and, on the other hand to cover the low cost demands for the simulation of lower-scale systems (using PC-based systems). Note that with these low cost PC-based platforms, large-scale cases can also be covered with high timing performances by using multiple PC-based simulators to ensure parallel treatments.

Today the quasi-exponential evolution of the processors performances has induced a wide range of commercialized digital simulators. To have a large overview of their application to electrical systems, Table 1.1 shows an example of classification into three main categories depending on the level of system complexity. The proposed examples integrate powerful multi-core processors and the number of these processors is adaptable with regards to user specifications (system scale, complexity, number of I/Os, required timing performances, cost ...).

From a design point of view, these simulators are provided with a set of user-friendly design tools that allow setting up the simulation, controlling and modifying the system parameters during a simulation, data acquisition and results analysis. These design tools are also in some cases integrated as toolboxes to traditional simulation tools like Matlab/Simulink. Examples are: RSCAD Software Suite from RTDS, [RTD], and RT-LAB from OPAL-RT, [OPA].

When the timing performances of these digital simulators are of concern, it is clear that the processing capabilities of the integrated multi-core processors allow fast treatments, thus the possibility to make simulations with short time-steps. However, today's experience feedbacks given in the literature underline that, depending on the focused application these simulators operate at time-steps down to around 5µs, [BEL07]. Figure 1.20, extracted from, [BEL07], gives a very good overview of typical time-step and number of processors required for each type of application.

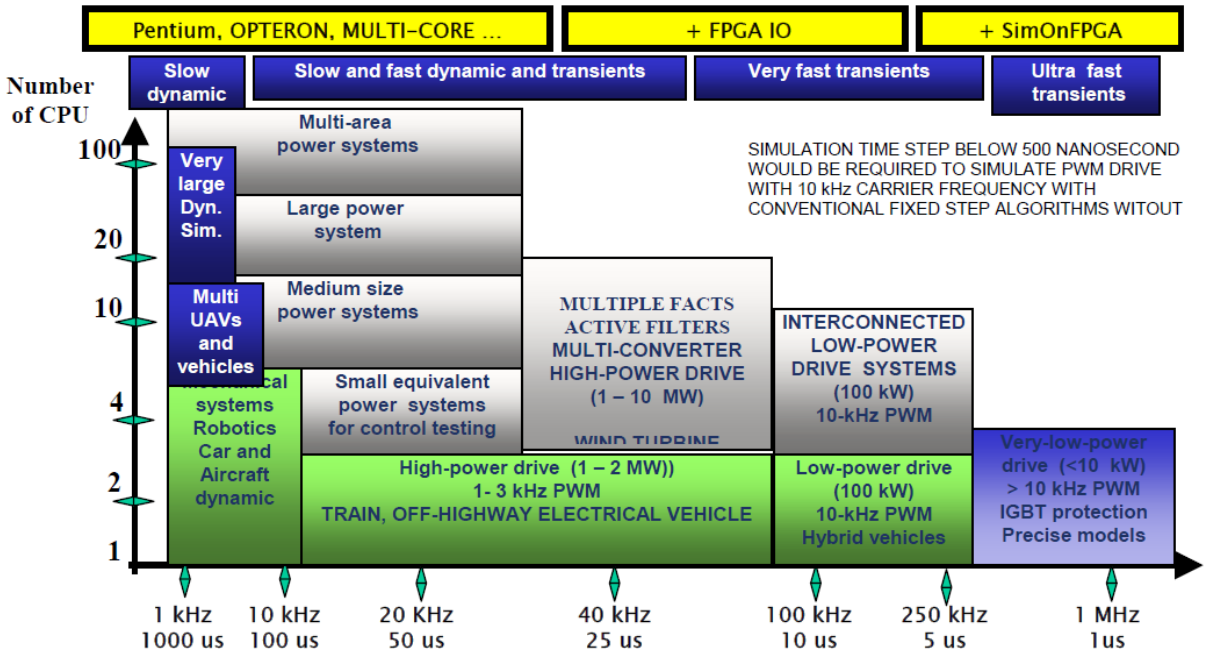


Figure 1. 20: Simulation time-step vs number of processors, extracted from [BEL07]

6.2. Contribution of FPGA in real-time digital simulation

To boost the timing performances and accurately simulate ultra-fast transients, most of today's real-time simulation platforms include additional FPGA-based boards. Thus, the FPGAs and multi-core processors are combined to meet the requirements from slow dynamic systems to ultra-fast dynamic systems. The following points are the assets of FPGAs which justify their positive contribution:

- Thanks to the parallel computational capability of FPGAs, the real-time simulation of electrical systems with ultra-fast transients such as high-frequency IGBT-based power converters is now possible [MYA11],[MAT10]. Indeed, by preserving all the potential parallelism inherent to the algorithm, the computation time is reduced allowing then the use of very small time-step in the order of 1 microsecond or less. Furthermore, such short time-step is required to detect fast switching events which improves the synchronization with the controller and then reduces the inter-simulation time-step events.
- Thanks to the hardware resources available in FPGAs (logic blocks, distributed memories, DSP blocks, high speed I/Os, hardwired processors, ADCs …, [XIL], [ALT], [IDK10]). Indeed, the combination of these resources gives the possibility to implement complex models. For example, the real-time simulation within the same FPGA device of complex power converters with very large number of switches like

multi-level modular converter is feasible [MAT11IPST]. Combination of multiple FPGAs can also be deployed to simulate large-scale systems (eg. [CHE13]).

FPGAs have also the advantage of allowing variables to have flexible word-size. This led to FPGA internal resource saving. The desired precision can then be achieved at no extra resource cost.

- Recent FPGAs give the possibility to implement software treatment since they can integrate CPU-based processor cores. This makes them efficient full System on Chip (SoC) solutions [BEN10], [MON12]. The calculation load can be then partitioned between the processor core and the logic programmable blocks to exploit the flexibility of the processor and the parallelism offered by FPGAs. Figure 1.21 illustrates the main advantages of using mixed software/hardware FPGA device. Recent FPGAs integrate also analog cores such as Analog-to-Digital Converter (ADC) and Digital-to-Analog Converter (DAC) which make them suitable for real-time simulation, especially for HIL applications since they facilitate their interface with the actual controllers under test.

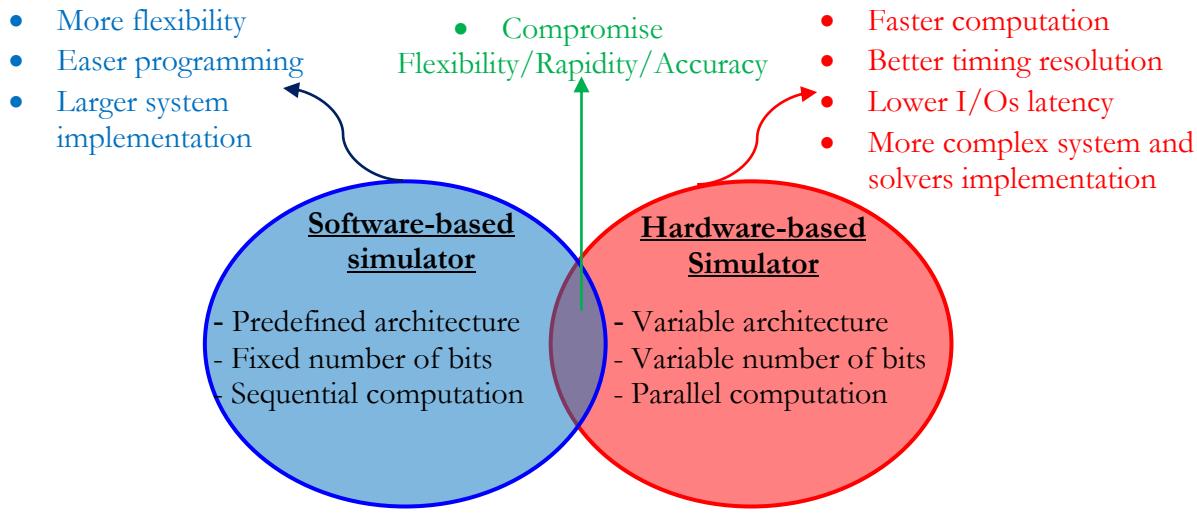


Figure 1. 21: Software-based and hardware-based simulations

- Typically, FPGAs are programmed using a Hardware Description Language (HDL), such as VHDL or Verilog. The use of such language may be seen as a drawback since designers are not systematically familiar with it, like the C-programming language. To address this issue, new high-level graphical tools for generating code are now available. The transition from the system model to its code is performed with an Automatic Code Generator (ACG) that comes today with most commercial simulation tools. As examples, the National Instruments LabVIEW FPGA Module can be used without the need for low-level HDL language [CAL13]. Also, when using FPGA target from Xilinx, designer can use the Xilinx System Generator (XSG) blockset tool [DUF06] under Simulink for generating automatically the code of the system model to be simulated. If this code is intended to be implemented in Altera FPGAs, the Simulink Altera DSP builder toolbox can be used.

7. Conclusion

In this chapter, a state of the art real-time simulation of electrical systems has been done. At first, the advantages of a real-time simulation over an offline one have been discussed. Then an overview of the latest application trends has been presented. The case of HIL

applications has been focused on. Finally, a deeper discussion about the modeling of electrical systems, its digital realization and its digital implementation has been made in sections 4.5 and 6.

From this discussion, it is clear that for HIL purposes the main challenge is how to improve the simulation accuracy and the main constraints are related to the model accuracy and the simulation time-step. To meet the expected performances, recent real-time simulators integrate high performance CPU-based boards combined with expensive FPGA boards. In such HIL applications, the cost is not critical. However, when dealing with embedded real-time simulation (which is the main topic of this thesis work), the cost becomes very challenging and the use of low cost devices (FPGAs here) is essential. Then, for a FPGA-based embedded real-time simulator, a compromise has always to be found between the expected accuracy of the simulator to reproduce dynamics and transients and the size of the FPGA on which it will be implemented (size obviously linked to the cost). An efficient methodology to design FPGA-based embedded real-time simulators for electrical systems is therefore mandatory.

In the next chapter the FPGA-based embedded real-time simulation of electrical systems will be discussed and an efficient design methodology will be proposed.

Chapter 2

FPGA-based embedded real-time simulation of electrical systems

1. Introduction

We have seen from the previous chapter that the main issue of interest is how to develop real-time simulators able to accurately reproduce the electrical system dynamics and transients. To satisfy this, high performance digital simulation platforms are now commercialized covering a wide range of system complexity and operating at very short simulation time-steps. At the same time, this high level of performance induces the use of powerful digital technologies (multi-core processor boards associated with high performance FPGA boards), which means that the cost of these platforms is not always taken into account.

However, the use of such expensive platforms is not acceptable in the case of embedded applications where real-time simulator IPs are not only developed for HIL testing but also embedded within digital controllers. Consequently, the constraints linked to the real-time simulation (discussed in the former chapter) are strengthened by those of embedded systems such as cost, power consumption, reliability, size and flexibility.

In this chapter, author starts by discussing the applications where embedded real-time simulators can be encountered and deals with the benefits of using FPGAs as embedded digital systems. Then, the induced constraints that must be considered during the development process are emphasized and organized into five dependent groups. Finally, design guidelines to be followed in order to manage these constraints are proposed.

2. FPGA-based embedded real-time simulation

2.1. Embedded real-time simulation

As seen in the former chapter, when talking about real-time simulation of electrical systems, one can systematically put in mind the HIL testing. This is not always the case since an emerging class of real-time simulators is expected to be more and more included to the next generation of embedded digital controllers. Then, the development of real-time simulators in this case must satisfy the real-time simulations constraints highlighted in the previous chapter and at the same time the constraints linked to the embedded systems.

An embedded real-time simulator can be seen as an IP module that simulates the system to be controlled or a part of it. Thus, this IP and the controller are both implemented and run altogether in the same device. The simulator IP is then considered as a part of an embedded control system. Depending on the way this simulator will be inserted into the control loop and how it will be connected to the controller and I/O interface, it can play several roles.

First of all, this simulator IP must be usable for HIL testing as illustrated in Figure 2.1. Then this IP can be included inside the control closed-loop or parallelized with the controller.

When it is included inside the control closed-loop as shown in Figures 2.2 and 2.3, the simulator plays a crucial role since it influences the control performance. It can be seen as an observer (estimator) that estimates non measured quantities (currents, voltages, position, speed...) in the context of sensors reduction (sensorless control, see Figure 2.2), [YIN13], [PEL12] or in the context of improving the system reliability (fault-tolerant control) by ensuring a digital system redundancy. The simulator is then activated and inserted in the control loop in the case of sensor fault, see Figure 2.3), [DEB12]. Another application of embedded real-time simulators can be the model predictive control, [COR09], [MAR12].

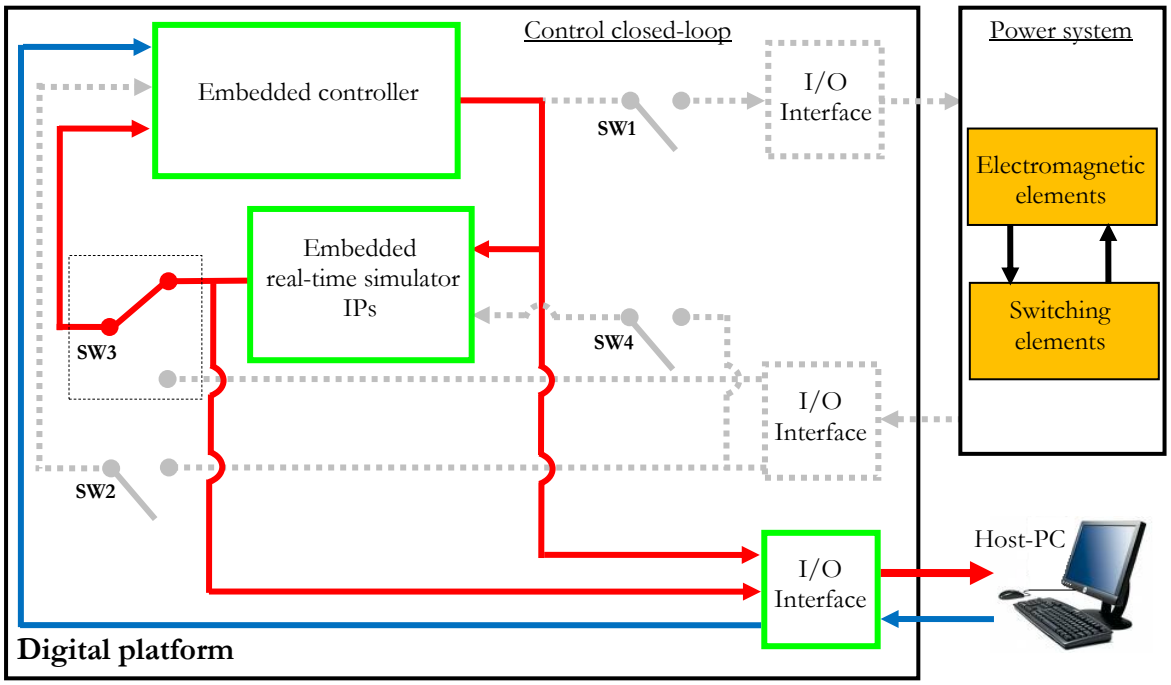


Figure 2.1: Embedded real-time simulator IPs applied in the context of HIL validation

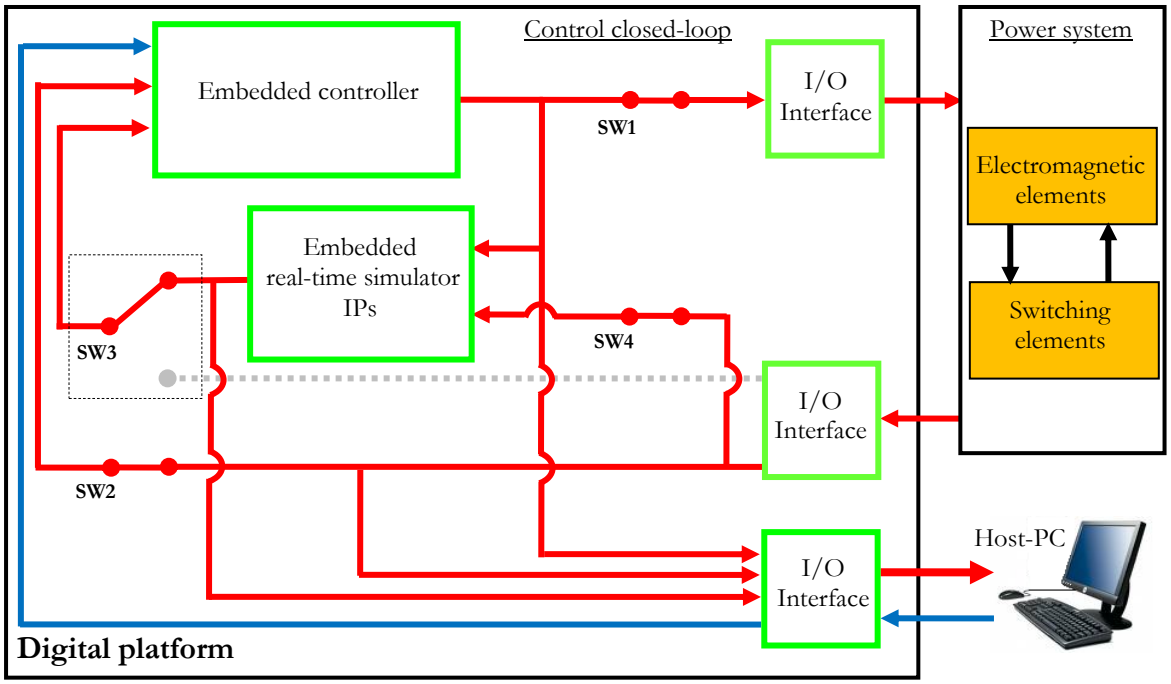


Figure 2.2: Embedded real-time simulator IPs included inside the control closed-loop and applied in the context of sensorless control

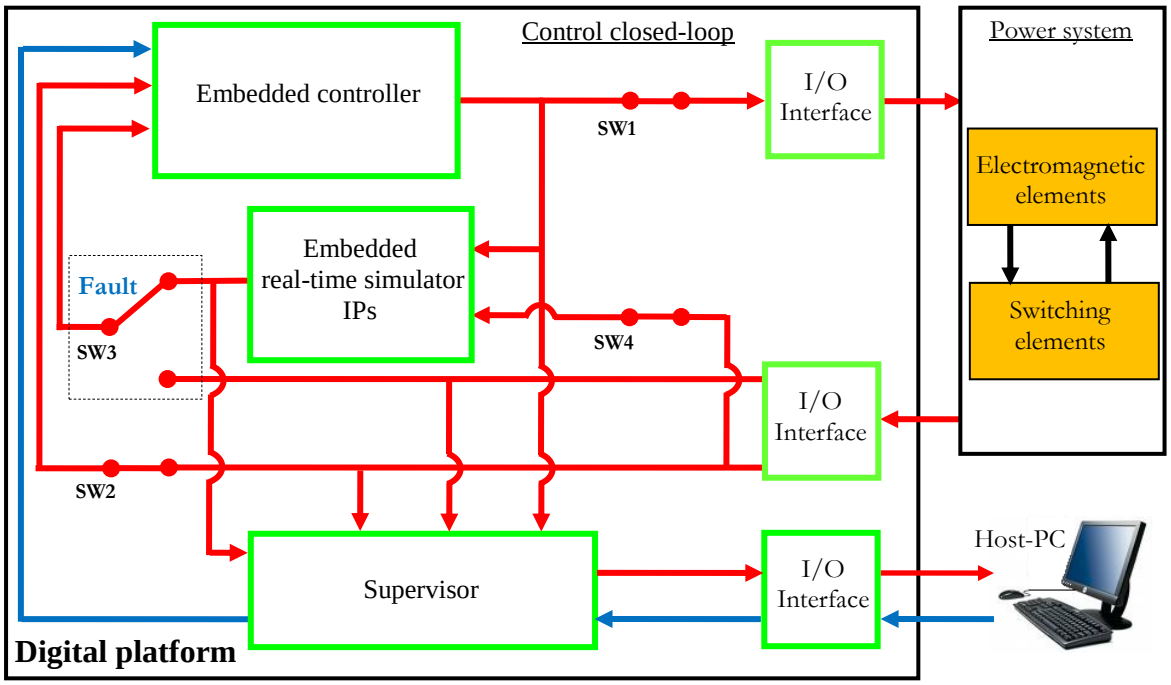


Figure 2.3: Embedded real-time simulator IPs included inside the control closed-loop and applied in the context of fault-tolerant control

The role of the embedded real-time simulator when it is parallelized with the controller is also of significant interest. In this case, the simulator can play a major role in the context of real-time fault detection and diagnosis [QUN11], [KAR09], [RAJ13], [DEL11]. It can also be useful in the context of a health monitoring for preventing unexpected failures and then avoid unpredictable system downtime [BER12], [ZOE14], [NUS14], [BAB14]. Another application example could be the estimation in real-time of the conduction and commutation losses of a power electronic converter. Figure 2.4 presents the principle synoptic of an embedded real-time simulator parallelized with the controller.

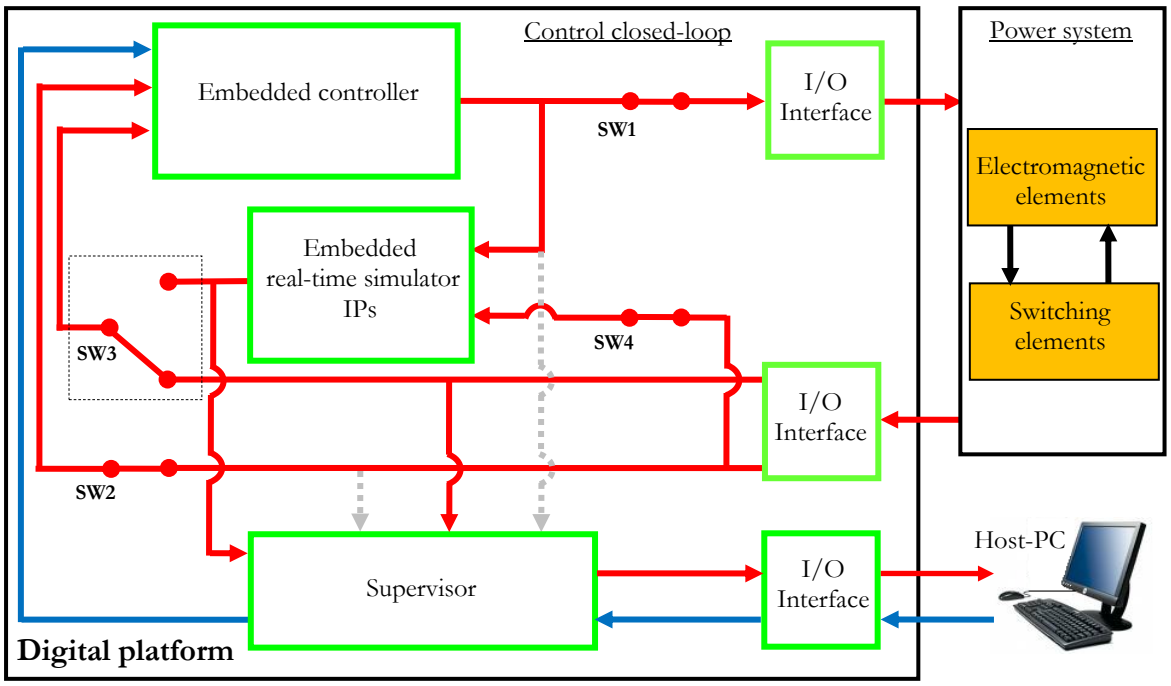


Figure 2.4: Embedded real-time simulator IPs parallelized with the controller and applied in the context of diagnostic and health-monitoring

2.2. Use of FPGAs

As far as the implementation is concerned, FPGA System on Chip (SoC) devices are certainly the most appropriate technological solution to implement both controller and the embedded real-time simulator. Compared to traditional solutions such as microprocessors and DSPs, such devices combine the software flexibility with the help of embedded processors (soft-processors and/or hard-processors, eg. Xilinx Zynq device includes a dual-core ARM processor, [XIL]) and the hardware processing performances with the help of FPGA configurable logic blocks. This is of course in addition to memory blocks (distributed or not), hardwired DSP blocks, clock management blocks, analog peripherals (ADC, DAC) and high speed IO blocks, [XIL]. This has the credit of offering the designer the possibility to efficiently deploy all these features depending on the need. As an example, Figure 2.5 gives an overview of the internal structure of the Xilinx Zynq-7000 FPGA SoC (device that is used in the proposed thesis work).

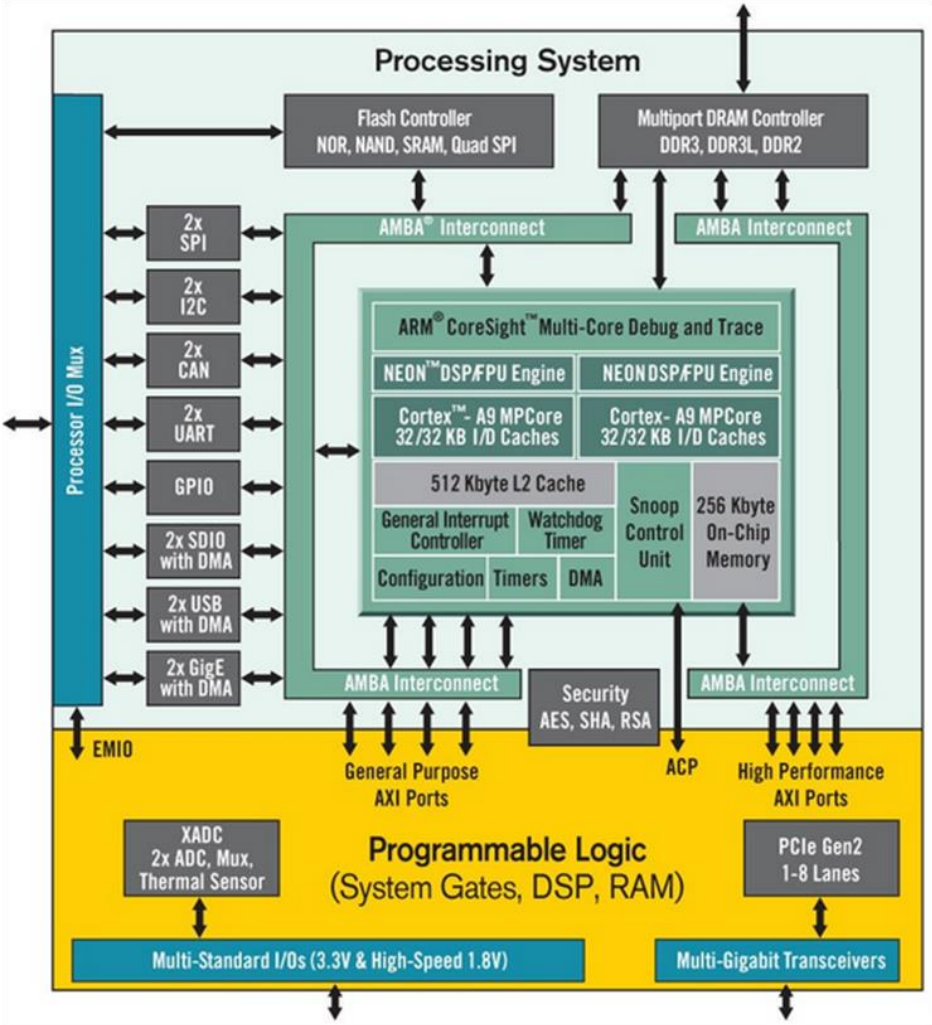


Figure 2.5: Example of FPGA SoC internal structure: Zynq-7000 device (source: [XIL])

In the focused applications, the most relevant way to preserve real-time operation and at the same time satisfy the real-time simulation accuracy (covering very fast dynamics) is to implement the simulator fully in hardware using the FPGA fabric of the SoC (FPGA configurable logic blocks in addition to the DSP blocks). This is due to the assets brought by this fabric in terms of time/area performances (detailed at the end of the previous chapter). Note that, in the case of slow system dynamics, it is possible to implement the simulator fully

in software (using a combination of embedded hardcore processors or softcore processors) or in mixed hardware/software by making the necessary partitioning. The system controller can also be implemented with these approaches depending on the required control bandwidth: fully in hardware, fully in software or in mixed hardware/software. By this way, an interesting partitioning methodology is deeply studied in [BAH11].

3. FPGA-based embedded real-time simulation constraints

To design an FPGA-based embedded real-time simulator that reproduces the dynamic behavior of the actual system and at the same time respects the resources limitations of the used FPGA, designer must consider a set of constraints during the development process. They can be organized into five dependent groups as shown in Figure 2.6.

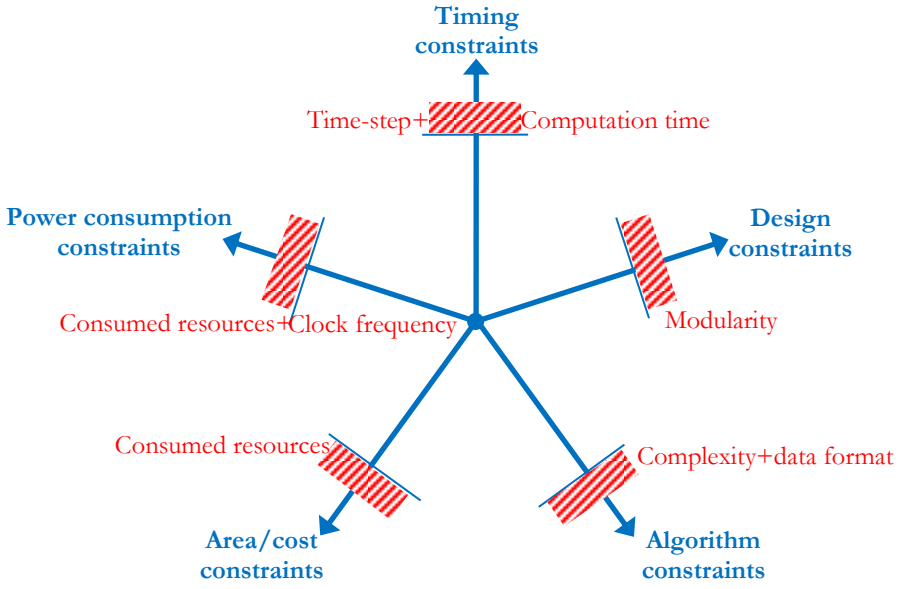


Figure 2.6: FPGA-based embedded real-time simulation constraints

3.1. Timing constraints

To make an embedded real-time simulation, the fundamental constraint is the choice of the appropriate simulation time-step. The latter has to be long enough to allow the processing of all model equations (to satisfy the real-time operation) but at the same time short enough to accurately represent the system. The selection of this time-step is deeply discussed in the previous chapter and in the case of embedded simulators, its choice is strengthened by the computation time of the implemented models. This computation time depends clearly on the model complexity, on the used FPGA resources and on the frequency of used system clock.

3.2. Design constraints (modularity)

To optimize the development process and then the time-to-market, one of the key-issues is how to make the design manageable and well-structured. To this aim, it is essential to ensure the modularity and reusability of the developed simulator. One can then easily understand the importance of making an IP-Library that gathers the developed FPGA-based real-time simulators in multi-level hierarchical decomposition.

Along this line, author proposes in Figure 2.7 an overview of the adopted library, dedicated to electrical systems and which can be seen as an extension of the digital library of control IPs, [NAO07P].

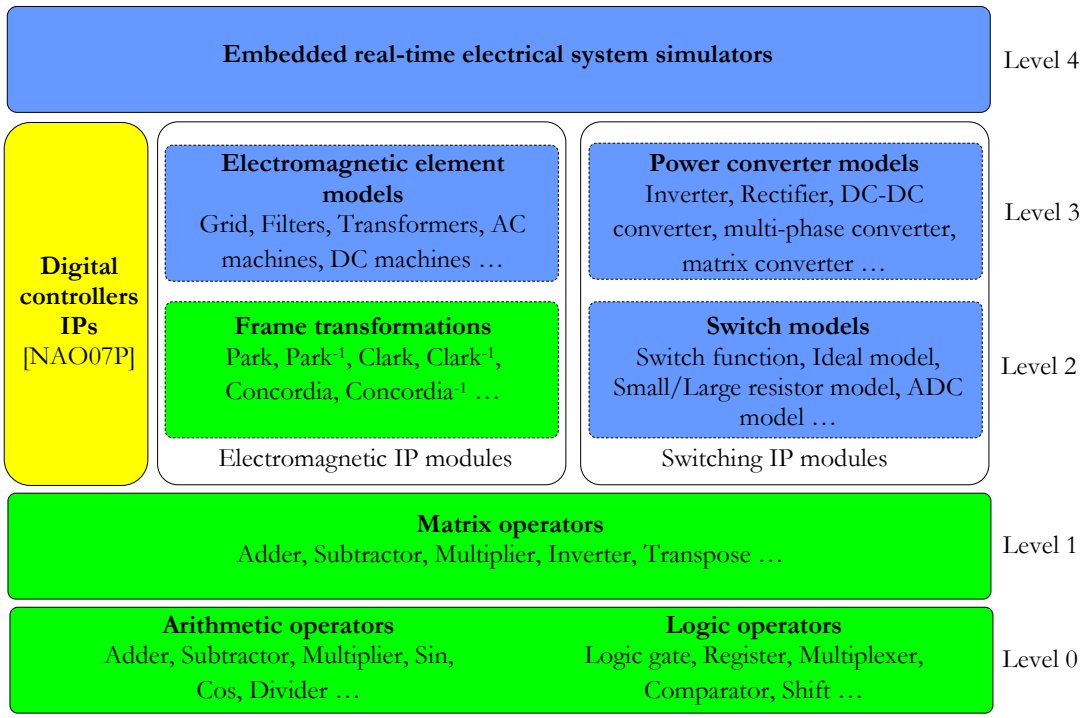


Figure 2.7: IP-Library for FPGA-based embedded real-time simulators

One can observe in this library a classification of real-time simulators into two categories; electromagnetic IP modules and switching IP modules. The first category gathers electromagnetic elements such as grid, filters, transformers, AC machines and DC machines. The second category concerns switching elements such as power converters where switching events are considered. Note that the transformations module such as Park and Park^{-1} is shared here between the controller and the simulated electrical system modules.

3.3. Algorithm constraints

The algorithm complexity is also one of the main constraints. As discussed in the previous chapter, the level of this complexity is linked to the level of the details included in the model and the way this model has been discretized and how its quantities have been represented.

Regarding the electromagnetic modules, the complexity depends on the order of the extracted differential equations with regards to the hypotheses that can be applied (eg. working frame, saturations, nonlinearities, saliencies…). As for the switching modules, the chosen model takes into account the topology of the converter in addition to the modeling level (system scale or switch scale modeling of switches).

From digital realization side, the complexity depends on the used numerical solver. Indeed, the number of processed mathematical operations is then linked to the adopted discretization method. This latter must of course be chosen by considering the precision and the numerical stability of the model.

Besides the complexity, the data conditioning is also another significant concern. Indeed, a well-chosen data format implies a good precision, a good signal-to-noise ratio and prevents from numerical instabilities. To this aim, a systematic choice could be the use of floating-

point format [BAC13] [JIA14]. Although the latter is well-known for its precision, it presents limitations especially when using FPGA devices. Indeed, the main challenge is the complexity of floating-point operators which impacts the FPGA hardware resources and the total computation time (thus the real-time simulation time-step). That is to say, in some cases, an optimal choice of fixed-point data format remains an appropriate alternative [SAN12]. In [GON14], it has been demonstrated that is possible to implement power converter models using the fixed-point format without loss of precision, while the consumed resources are minimized and the clock frequency is maximized. This is also confirmed in [SCH14] where a fixed-point nonlinear permanent magnet synchronous machine model has been compared to its floating-point counterpart to check the precision and small difference has been obtained.

3.4. Area/Cost constraints

The preservation of the potential parallelism inherent to the algorithm is strongly linked to the available hardware resources which are limited by the cost of the used FPGA target. The development of the FPGA architecture must then consider these resources limitation. Another important concern is the computation time that must be less than the real-time simulation time-step (to ensure real-time operation). In addition to the latency of the algorithm, this computation time depends on the used clock frequency. This latter is closely linked to the quality of the designed architecture and precisely to the propagation delays between the arithmetic operators. Designer must reduce these delays to increase the maximum achievable clock frequency and consequently reduce the computation time.

To face this time/area issue and always ensure a low cost FPGA implementation, designer has to optimize the use of internal resources. At first, the use of hardwired resources such as DSP units, memory blocks and the integrated processors (when software implementation is required) must be privileged since they have already been optimized by the device manufacturer. The pipelining of the architecture is also important to minimize the impact of the propagation delays. Additionally, one of the interesting techniques to overcome the resources limitation is to develop a factorized FPGA architecture by adopting the A3 methodology (Algorithm Architecture Adequation) [GRA99]. For a given algorithm complexity, the objective is the reduction of hardware resources by factorizing heavy operators (such as multipliers). In contrast, the treatment is locally serialized which may enlarge the computation time. A compromise must then be established in order to meet the available resources and the level of factorization with regards to the computation time limit.

3.5. Power consumption constraints

These constraints concern typically applications such as portable and power-conscious embedded systems where low power consumption is a key-issue. Here again the higher are the clock frequency and the used hardware resources, the higher is the consumed power. This point is not maintained in the following discussion since in the case of electrical systems applications, the power consumption of the digital control unit doesn’t matter compared to the power consumed by the power system itself.

4. Design guidelines for developing FPGA-based embedded real time simulators

The proposed design guidelines consist of a set of steps and rules to be followed in order to get a better compromise between the expected performances of the real-time simulator and the

limits of the internal computational resources of the low-cost controller in which the simulator will be embedded.

As it can be seen in Figure 2.8, the particularity of the presented design guidelines consists in providing generic and well-structured practical rules that are organized into four major steps: the preliminary system specification, the algorithm development, the FPGA implementation and the experimentations.

These guidelines can be regarded as a natural extension of those proposed in [NAO07] for designing industrial controllers.

In addition, a significant distinction between the development of the algorithm and the development of the FPGA architecture is made. This distinction has the advantage of making the algorithm totally independent of the way of its implementation. Once the algorithm is developed, it can be implemented fully in software, fully in hardware or in mixed software/hardware.

In the following a brief description of this methodology is presented. Its details will be presented in the rest of PhD report (chapters 3.4.5).

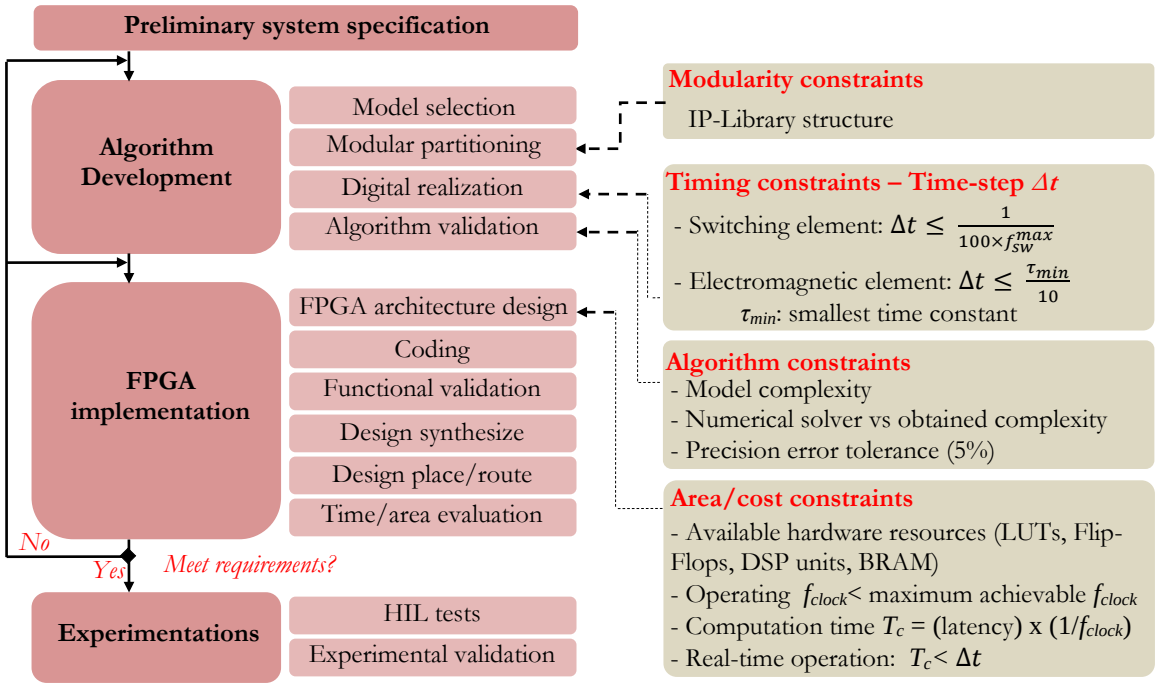


Figure 2.8: Design guidelines for FPGA-based embedded real-time simulators

4.1. Preliminary system specification

When developing an FPGA-based embedded real-time simulator, designer has to firstly begin by making a preliminary system specification regarding the whole application where the embedded real-time simulator IP will be used. This involves defining which actual system will be controlled, which control strategy will be adopted, which FPGA device will be used and which interface boards will be adopted.

Depending on the expected function of the embedded real-time simulator, designer has to define also the components of the power system that need to be simulated in real-time. The simulation of the whole power system is not always necessary (see chapter 5). The IP

modules to be used and which have already been developed are then collected from the IP-Library. Their complexity and time/area performances will be considered during the next steps.

Defining these components will also lead to deduce the necessary connections and interfaces between the controller, its embedded real-time simulator and the actual power system.

4.2. Algorithm development

The algorithm development step is aimed to validate the overall controller including its embedded real-time simulator with regards to the timing, design and algorithm constraints. Since the study will be focused only on this simulator, author supposes that the controller module has already been developed and extracted from the IP-Library.

During this step, the first task consists in choosing the convenient model solver to be implemented for each element of the electrical system to be simulated. Then, a modular partitioning is adopted. It consists in dividing each model into independent and reusable modules. These modules are organized hierarchically with respect to the structure of the proposed IP-library (see Figure 2.7). Then, a digital version of the simulator algorithm is derived. During this phase, the simulation time-step is firstly specified based on the timing constraints imposed by the system (switching frequency of the power converter, time constant of the electromagnetic elements). This is followed by the selection of the appropriate numerical solver, the numerical data representation format and the base values for the normalization, having in mind the implied algorithm constraints.

Finally, an offline simulation is performed using Matlab/Simulink tools. This allows the verification of the correct functionality of the simulator algorithm when it is included in the considered control system.

4.3. FPGA Implementation

Once the algorithm development is validated, the designer can start the FPGA implementation phase. During this phase, the design and validation of the corresponding FPGA architecture is achieved. Note that for the previously mentioned reasons, a fully hardware implementation is adopted for the simulator.

To make the design process less constraining, an interesting solution consists in generating automatically the FPGA-based architecture from Matlab/Simulink, using automatic code generator [RIC02]. However, this solution is dependent on the performance of this code generator and can lead to un-optimized architecture that may violate the constraints. This is why, in the proposed design guidelines, designer has to design, optimize and code himself the FPGA-based architecture with the help of a set of well-known sub steps but always keeping in mind the area/cost constraints. The starting point here is the evaluation of the available hardware resources to be allocated to the simulator IP of concern. Based on that, the corresponding architecture is developed, coded and included in the whole design.

The hardware architecture of each of the developed modules (according to the adopted partitioning) is composed of two main parts: a data-path and a control unit. The data-path contains the used arithmetic or logic operators and data buses between them. The control unit which is a simple Finite State Machine (FSM) is used for generating a suitable timing schedule to control the data-path. It is activated via a start pulse signal and generates an End pulse signal when the computation is completed. The different operational modules are

controlled by the same sequencer named a global control unit. Figure 2.9 illustrates the proposed structure of an FPGA-based architecture of an electrical system model.

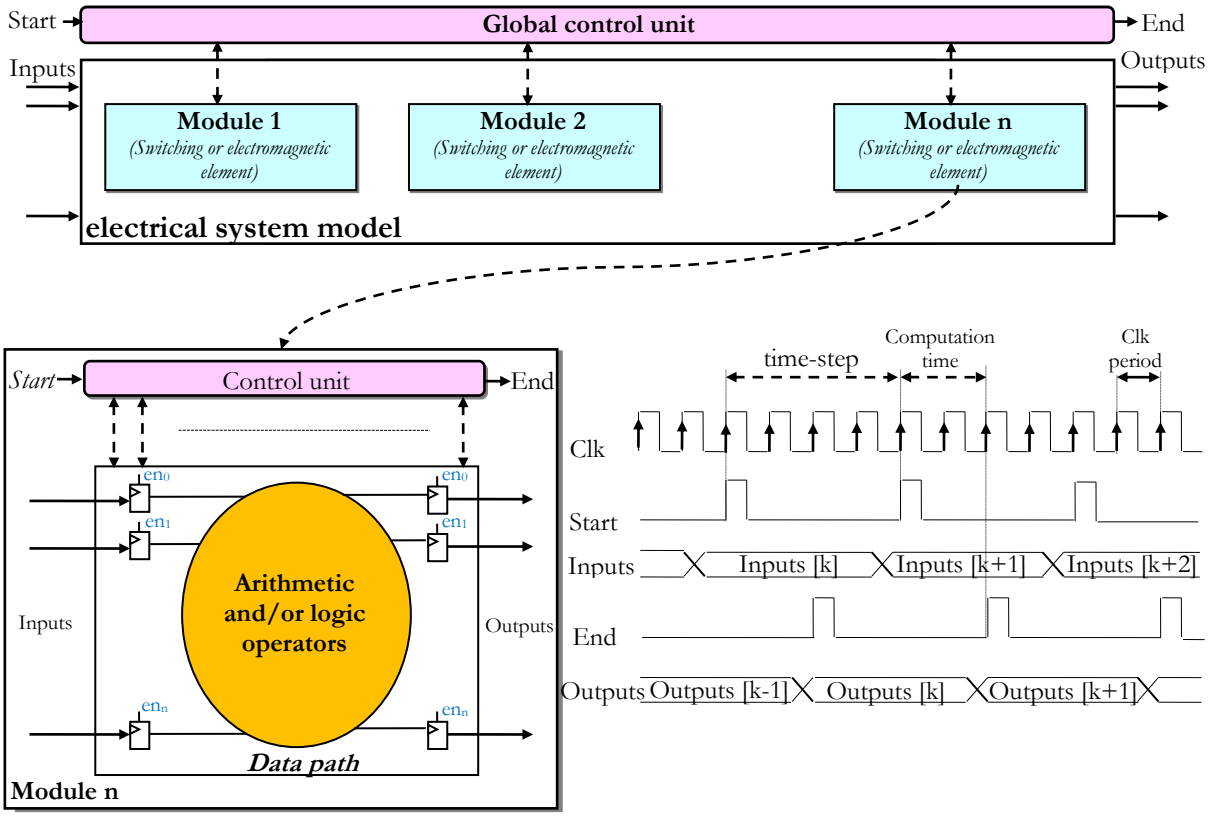


Figure 2.9: Proposed structure of an FPGA-based architecture of an electrical system model

Here, the design and optimization of the hardware FPGA-based architectures are made with regards to the timing and area/cost constraints. To satisfy these constraints, two main techniques can be adopted. This first one is the A^3 methodology. It consists in factorizing the architecture to find optimized hardware architecture which satisfies at the same time the expected timing performance and the limits of the internal computational resources of the used low-cost FPGA device. The second one is the pipeline technique. It is used to reduce propagation delays to increase the maximum achievable clock frequency and consequently reduce the computation time.

Once the FPGA architecture is designed and coded with VHDL or Verilog, the next step is the functional validation using dedicated tools such as the well-known ModelSim. During this step, the obtained simulation results are compared to those obtained by the Matlab/Simulink during the algorithm development.

Once the FPGA architecture is functionally validated, the design synthesis and the time/area performances (latency, maximum clock frequency, consumed hardware resources) are evaluated and balanced with their corresponding constraints (computation time limit, available hardware resources).

If the obtained time/area performances satisfy the corresponding constraints, the experimentation phase is then initiated. In the opposite case designer has to optimize either the algorithm or the architecture.

4.4. Experimentations

The last step aims to make experimental validations starting with HIL tests where the association and interaction between the controller and its embedded real-time simulator are tested, and ending by a full experimental validation using the experimental platform.

5. Conclusion

In this chapter, a state of the art FPGA-based embedded real-time simulation of electrical systems has been done. At first, the applications where embedded real-time simulators can be encountered have been discussed. Then, the benefits of using FPGAs as embedded digital systems have been presented. The most important constraints linked to the FPGA-based embedded real-time simulators have been then emphasized. Also, design guidelines to be followed in order to manage these constraints during the development process have been proposed.

In the next chapter the implementation in low-cost FPGA of embedded real-time simulator IPs of electromagnetic elements of an electrical system will be discussed and the proposed guidelines will be adopted.

Chapter 3

Implementation in low cost FPGA of embedded real-time simulator IPs of electromagnetic elements

1. Introduction

As we have seen in the previous chapter, making an IP-library devoted to embedded real-time simulator IPs is of great importance to ensure their modularity and reusability. In the proposed IP-library, these modules are classified into two main categories; those dedicated to electromagnetic systems (including moving systems) and those dedicated to switching systems.

In this chapter the first category is focused on. Three cases have been studied: the embedded real-time simulator of a 3-phase DC-excited synchronous machine, the one of a 3-phase induction machine and finally the one of a three-stage avionics alternator. As a recall, application examples of these IPs have been discussed in the previous chapter.

The main challenge when developing an FPGA-based embedded real-time simulator is how to accurately simulate the system dynamics (electrical dynamics, mechanical dynamics and possibly thermal dynamics), generally described by differential equations, but having always in mind the timing, the algorithm and the area/cost constraints. This is the reason why the proposed design guidelines have been applied to the chosen case studies.

Furthermore, during the digital realization of the synchronous machine simulator IP, the benefits of making a delta transformation is discussed. Indeed, compared to a shift operator, this transformation aims to improve the stability of the numerical solver when short simulation time-step and fixed-point (with limited data precision) are used.

2. FPGA-based embedded real-time simulator of a 3-phase DC-excited synchronous machine

In this section, the development of an FPGA-based embedded real-time simulator IP of a synchronous machine is dealt with. This development has been presented and structured according to the proposed design guidelines.

2.1. Preliminary system specification

The chosen 3-phase DC-excited synchronous machine is a wound-rotor, Y-connected and salient machine. Its parameters are listed below:

Table 3. 1: Synchronous Machine Parameters

0.8 KVA, 230V, 50 Hz, 3 Phases, Y connection	
Stator resistance R_s= 10.5 Ω	Rotor resistance R_r= 62.5 Ω
d axis stator inductance L_{sd} = 0.245 H	Mutual inductance M_{sr}= 0.85 H
q axis stator inductance L_{sq}= 0.229 H	Nominal stator current I_{sn}= 1.52 A
Rotor inertia J=0.0006 Kg.m²	Viscous friction coefficient f=0.05 Kg.m²/s
Number of pole pairs p =2	Excitation current I_{rd}=1.5 A

The machine simulator IP has to be reusable for a wide range of powers and for different structures. To make a complete validation, this simulator is associated to the IP of a 3-phase 2-level inverter (studied in chapter 4) and the one of a mechanical load. Also an ON-OFF

hysteresis stator current controller is used, [NAO07P]. All these IPs are implemented in the same Xilinx XC7Z020 Zynq device [XIL]. Also, it should be noted that in the following, the developed synchronous machine simulator IP has been applied in the context of HIL validation (see Figure 2.1, chapter 2).

2.2. Algorithm development

a- Model selection

To start with, the d-q based model with external Park transformations has been chosen. As investigated in the first chapter, this model allows reducing the number of equations and their complexity but at the same time allows reaching an acceptable level of accuracy. These characteristics are very important in the context of this work since they can contribute to find the best compromise between the timing and area/cost constraints.

The proposed model assumes that the saturations, the core losses and the skin effects are neglected. Moreover, it is assumed that rotor excitation current is kept constant (see Table 3.1). The state-space continuous-time model is described by the following equations (relations 3.1 and 3.2).The latter are expressed in rotor reference frame (d-q coordinates, with d-axis linked to the inductor).

$$\begin{cases} s.i_{sd} = -\dfrac{R_s}{L_{sd}}.i_{sd} + \dfrac{L_{sq}}{L_{sd}}.\omega_e.i_{sq} + \dfrac{1}{L_{sd}}.v_{sd} \\ s.i_{sq} = -\dfrac{L_{sd}}{L_{sq}}.\omega_e.i_{sd} - \dfrac{R_s}{L_{sq}}.i_{sq} - \dfrac{M_{sr}}{L_{sq}}.I_{rd} + \dfrac{1}{L_{sq}}.v_{sq} \end{cases} \tag{3.1}$$

$$\begin{cases} s.\omega_e = -\dfrac{3}{2}.\dfrac{p^2}{J}\left(\left[L_{sd} - L_{sq}\right].i_{sd}.i_{sq} + M_{sr}.I_{rd}.i_{sq}\right) - \dfrac{f}{J}.\omega_e - \dfrac{p}{J}.T_L \\ s.\theta_e = \omega_e \end{cases} \tag{3.2}$$

The expression of the electromagnetic torque T_e is given in relation 3.3. In our case, the modelled synchronous machine is salient (L_{sd}<L_{sq}).

$$T_e = \frac{3}{2}.p.\left[L_{sd} - L_{sq}\right].i_{sd}.i_{sq} + \frac{3}{2}.p.M_{sr}.I_{rd}.i_{sq} \tag{3.3}$$

b- Modular partitioning

The selected model has been partitioned with respect to the structure of the proposed IP-Library. To this aim, the d-q based synchronous machine model is divided into three independent and reusable modules (Figure 3.1): the Park transformation (*abc-dq*), the machine model (electrical and mechanical models) and the Park^{-1} transformation (*dq-abc*). Figure 3.1 highlights the interconnections between these modules.

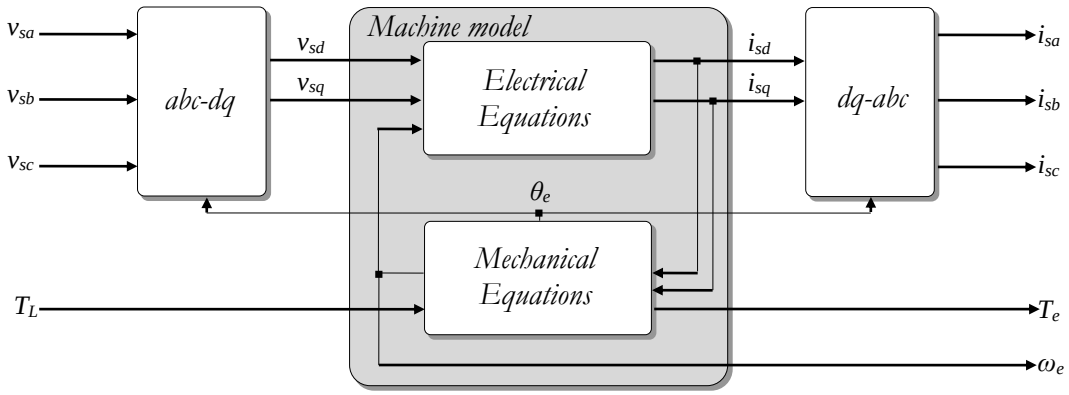


Figure 3.1: d-q based synchronous machine model - modular partitioning

c- Digital realization

After having developed and partitioned the d-q based synchronous machine model, its digital realization has been studied. To start with, let's recall, at first, that this model needs the use of small simulation time-step to reduce interfacing errors between the module of electrical and mechanical equations and the external Park transformations modules (see chapter 1). To keep these errors within an acceptable range and at the same time satisfy the timing constraint imposed by the smallest time constant of the machine, the use of small simulation time-step in order of few microseconds is mandatory.

However, it is worth noticing that, when using such small time-step range, a particular attention has to be given to the stability of the numerical model, especially when this model is carried out with fixed-point format. Indeed, when the time-step decreases, the poles move closer to the stability limit (|z|=1) and due to their location, they become very sensitive to the error introduced by the limited data precision. This error could lead to the instability [LI20][GAN93].

To cope with this issue, an interesting solution is to make a delta transformation. Indeed, with this transformation, the circle of stability region approaches the stability zone of the s-domain (left half complex plane) when the time-step decreases. As a consequence, poles are far from the region of stability limit and their variations are not sensitive to the quantization errors introduced by the fixed-point format [GAN93_2] [KAU97]. Therefore, the numerical stability problems are avoided. To illustrate this, Figure 3.2 shows the difference between the stability regions of shift-operator (z-domain) and delta-operator (γ-domain) with regards to the time-step.

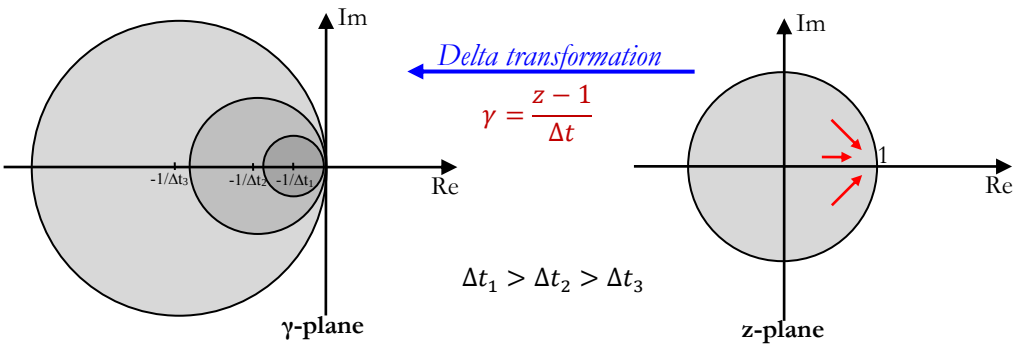


Figure 3.2: Stability regions of z-domain and γ-domain with regards to the time-step Δt

During this design step, it has been decided to examine the advantages of using delta-operator over the shift-operator to design the machine digital realization. Two digital realizations for the continuous-time synchronous machine model have been then developed and compared. The first one is based on the classical shift-operator and the second one is based on the delta-operator.

- Digital realization based on shift-operator

The digital realization based on shift-operator is firstly developed. To this purpose, the implicit trapezoidal approximation has been adopted. The state-space continuous-time model has been then transformed to its discrete-time counterpart according to Relation 3.4.

$$s \to \frac{2}{\Delta t}.\frac{z-1}{z+1} \tag{3.4}$$

The obtained discrete-time normalized model is given in the following,

$$\begin{cases} i_{sd} = z^{-1}\left(t_1.i_{sd} + t_2.i_{sq}.\omega_e + t_3.v_{sd}\right) + \left(t_4.i_{sq}.\omega_e + t_5.v_{sd}\right) \\ i_{sq} = z^{-1}\left(t_6.i_{sq} + t_7.i_{sd}.\omega_e + t_8.\omega_e + t_9.v_{sq}\right) + \left(t_{10}.i_{sd}.\omega_e + t_{11}.\omega_e + t_{12}.v_{sq}\right) \\ \theta_e = z^{-1}.(t_{13}.\omega_e) + t_{14}.\omega_e \\ T_e = t_{15}.i_{sd}.i_{sq} + t_{16}.i_{sq} \\ \omega_e = z^{-1}.(t_{17}.\omega_e + t_{18}.T_e - t_{19}.T_L) + (t_{20}.T_e - t_{21}.T_L) \end{cases} \tag{3.5}$$

Where, $t_{i\ (i=1....21)}$ are the coefficients of the normalized shift-operator based model.

- Digital realization based on delta-operator

The development of the digital realization based on delta-operator is now presented. To this aim the previously developed discrete-time model based on trapezoidal method has been transformed to its discrete-time delta-operator based model.

The transformation of the trapezoidal form into the delta form is derived using the following relation:

$$z \to 1 + \gamma.\Delta t \tag{3.6}$$

Where γ is the new transformation variable associated to the delta-operator [MID90]. The obtained discrete-time normalized model based on the delta operator is then expressed as follows.

$$\begin{cases} i_{sd} = \gamma^{-1}\left(d_1.i_{sd} + d_2.i_{sq}.\omega_e + d_3.v_{sd}\right) + \left(d_4.i_{sq}.\omega_e + d_5.v_{sd}\right) \\ i_{sq} = \gamma^{-1}\left(d_6.i_{sq} + d_7.i_{sd}.\omega_e + d_8.\omega_e + d_9.v_{sq}\right) + \left(d_{10}.i_{sd}.\omega_e + d_{11}.\omega_e + d_{12}.v_{sq}\right) \\ \theta_e = \gamma^{-1}.(d_{13}.\omega_e) + d_{14}.\omega_e \\ T_e = d_{15}.i_{sd}.i_{sq} + d_{16}.i_{sq} \\ \omega_e = \gamma^{-1}.(d_{17}.\omega_e + d_{18}.T_e - d_{19}.T_L) + (d_{20}.T_e - d_{21}.T_L) \end{cases} \tag{3.7}$$

Where, d_i coefficients depend on the time-step, the parameters of the machine and the base quantities used for normalization. Their expressions are given in Appendix-B.

- Comparative study

In order to make a quantitative comparison and highlight the advantages of delta-operator based model, the chosen criterion is the numerical stability. To this aim, the poles of the developed continuous-time, shift-operator and delta-operator discrete-time based models are examined. The influence of the simulation time step and the influence of the fixed-point data precision are evaluated.

To start with, let's denote p_{ci}, p_{ti}, and p_{di} the ith pole of, respectively, the continuous-time, the shift-operator and the delta-operator based synchronous machine models. Their expressions are given in the flowing. More details about these equations and how they are obtained are given in Appendix-C.

$$\begin{cases} p_{c1} = \dfrac{c_5 + c_1}{2} + i\dfrac{\sqrt{-(c_1 + c_5)^2 + 4.c_1.c_5 - 4.c_2.c_4.\omega_e}}{2} \\ p_{c2} = \dfrac{c_5 + c_1}{2} - i\dfrac{\sqrt{-(c_1 + c_5)^2 + 4.c_1.c_5 - 4.c_2.c_4.\omega_e}}{2} \end{cases} \tag{3.8}$$

$$p_{ti} = \frac{p_{ci} + \dfrac{2}{\Delta t}}{\dfrac{2}{\Delta t} - p_{ci}} \tag{3.9}$$

$$p_{di} = \frac{1}{\dfrac{1}{p_{ci}} - \dfrac{\Delta t}{2}} \tag{3.10}$$

Where,

$$c_1 = \frac{-R_s}{L_{sd}}; c_2 = \frac{L_{sq}}{L_{sd}};\ c_4 = \frac{-L_{sd}}{L_{sq}};\ c_5 = \frac{-R_s}{L_{sq}}$$

Based on these expressions, the locations of poles with regards to the time-step are calculated for each model and shown in Table 3.2. As it can be seen in this table, when the time-step decreases, the poles of the delta-operator based model move towards their continuous-time counterparts (see also Figure 3.3) and the poles of shift-operator based model move closer to the stability limit (see also Figure 3.4).

Based on these results, it can be concluded that the problem of stability does not occur for the model based on delta-operator when short time-step is used, since their poles converge to their continuous-time counterparts and are far from the region of stability limit. This is not the case for shift-operator since the stability margin becomes smaller with short time-step and the poles are then very sensitive to the coefficients variations introduced by the fixed-point format. Large variations can then lead to the instability of this model.

To verify this, Figure 3.5 and Figure 3.6 show how the data format of coefficients influences the stability of the shift-operator based model especially when short time-step is used. It can be seen that, for the model based on shift-operator, the pole moves more and more toward the region of the stability limit when the precision (number of bits on the fractional part of the format) of its coefficients decreases. In our case, the model becomes unstable when the precision is less than 15bits (Δt is set to 1µs). In [CHA08], an analytical method is proposed which allows defining the limits of model coefficient word length in order to preserve stability.

Table 3. 2: Poles location depending on the time-step

Poles of the continuous-time model: Pc_1= -38.929 + 297.38i Pc_2= -38.929 - 297.38i		
Δt=100 µs	Δt =10 µs	Δt =1 µs
Poles of the shift-operator model		
Pt_1= 0.9961 + 0.0297i Pt_2= 0.9996 - 0.0029i	Pt_1= 0.9996 + 0.0029i Pt_2= 0.9996 - 0.0029i	Pt_1= 0.9999 + 0.0002i Pt_2= 0.9999 - 0.0002i
Poles of the delta-operator model		
Pd_1= -43.248 + 296.16i Pd_2= -43.248 - 296.16i	Pd_1= -39.363 + 297.26i Pd_2= -39.363 - 297.26i	Pd_1= -38.972 + 297.36i Pd_2= -38.972 – 297.36i

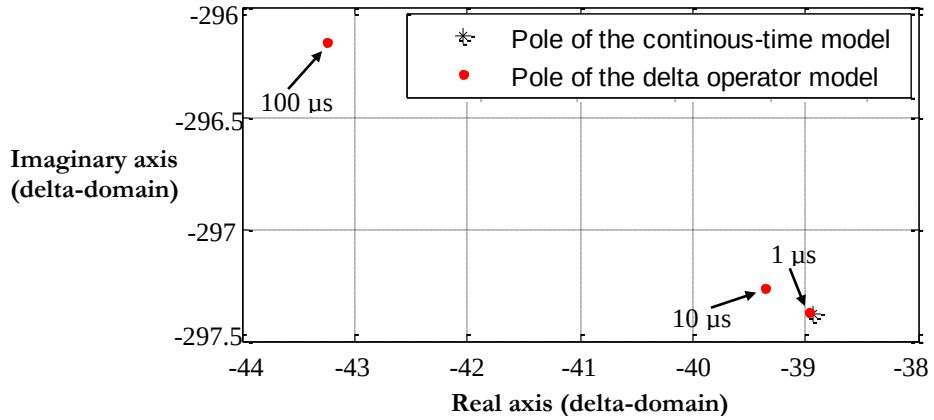


Figure 3.3: Location of delta-operator model poles depending on the time-step

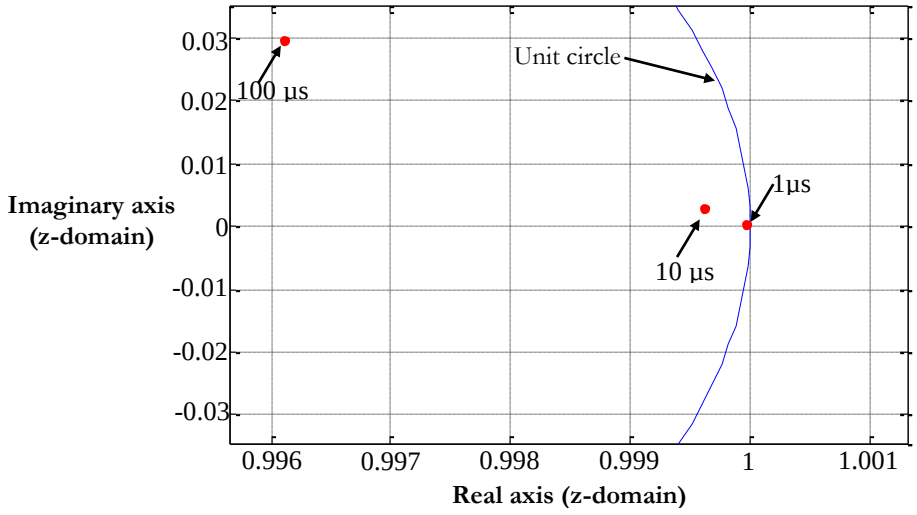


Figure 3.4: Location of shift-operator model poles depending on the time-step

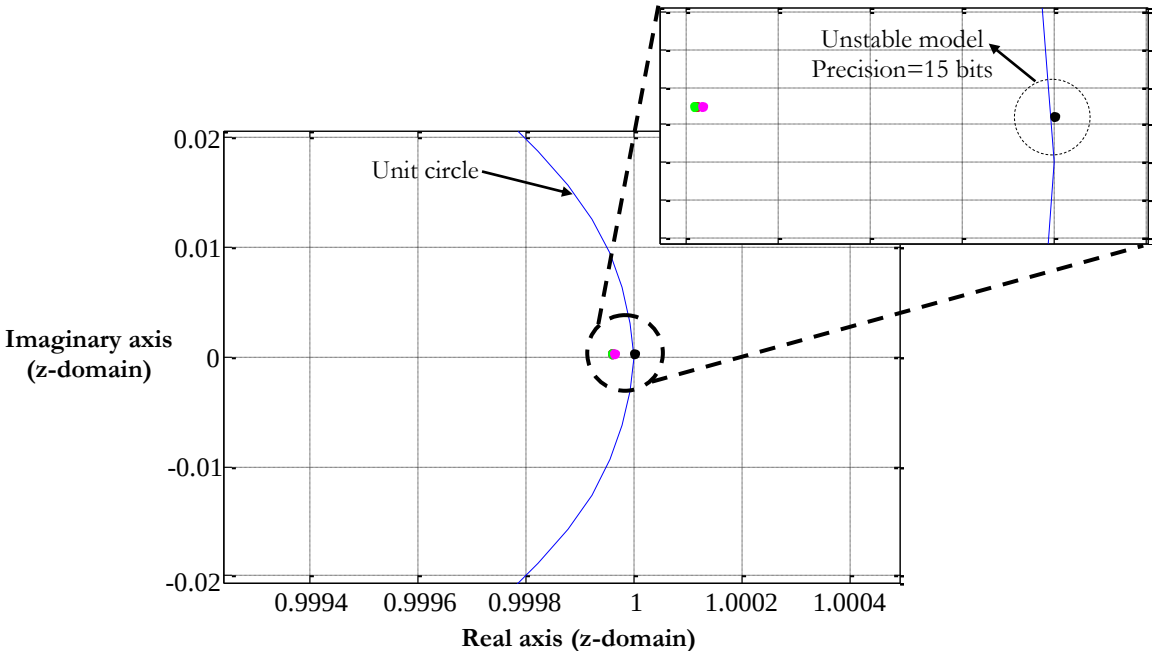


Figure 3.5: Impact of the fixed-point format on the stability of shift-operator model

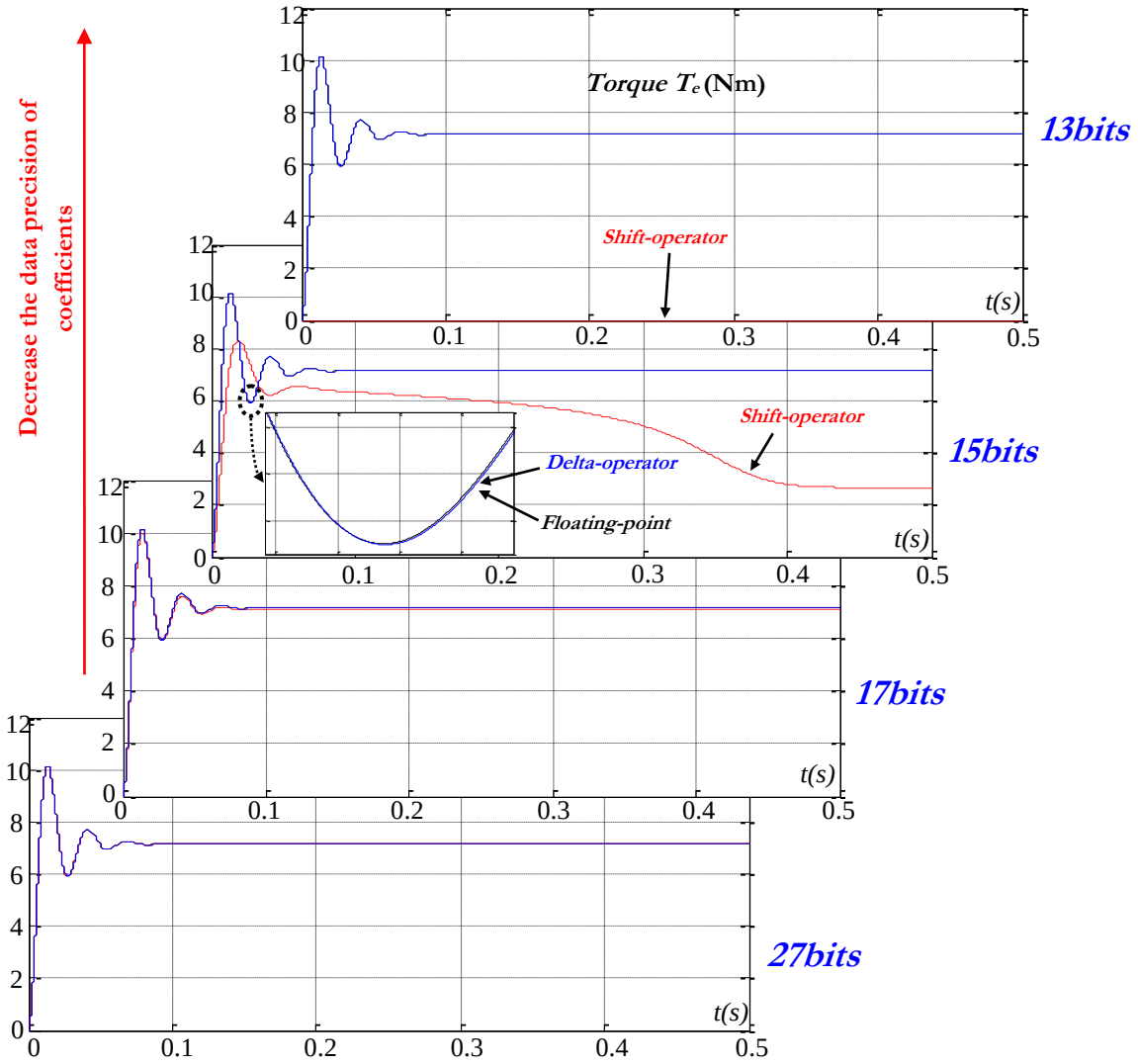


Figure 3.6: Influence of the data fixed-point precision on the torque response

As far as the model based on delta-operator is concerned, it can be observed in Figure 3.7 that the poles are less affected by the variation of coefficients caused by a limited precision since poles are far from the region of stability limit and the stability problem is of any concern even if the data precision of coefficients is equal to 15 bits. This is confirmed also in Figure 3.6).

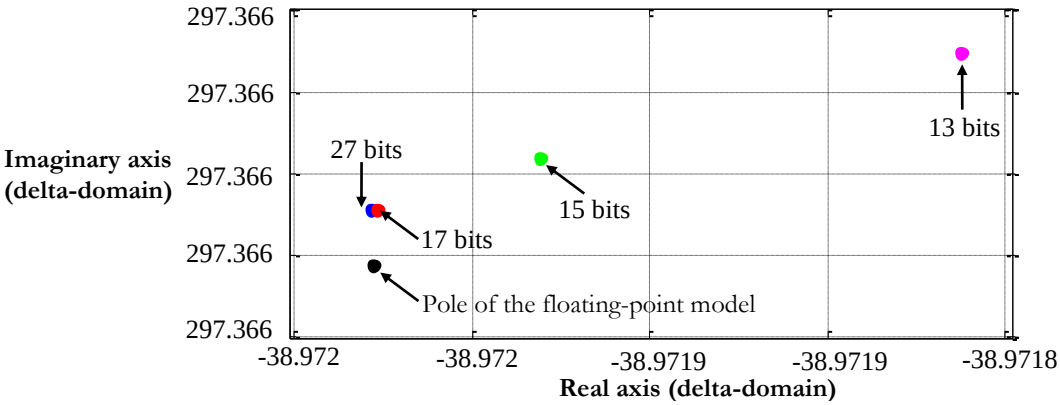


Figure 3.7: Impact of the fixed-point format on the stability of delta-operator model

Up to now, it has been seen that when using small time-step and fixed-point format with limited data precision which are both required in the context of this work, the digital realization based on delta-operator is more accurate and its stability is more preserved compared to the digital realization based on shift-operator.

This is why; the digital realization based on delta-operator is chosen to be implemented. To satisfy timing and algorithm constraints the simulation time-step has been set to 1µs which is much smaller than the smallest time constant of the machine and sufficiently small to reduce interfacing errors between the machine model and the external Park transformations. As for the data word length, it has been set to 28 bits which leads to small quantification error.

d- Algorithm validation

Now as the digital realization based on delta-operator is chosen to be implemented, an offline simulation is to be made so as to make the validation of the choices made during the digital realization. The chosen realization is then implemented under Matlab/simulink tool. Figure 3.8 presents the structure of the implemented delta-operator based realization. The case of the electrical equations (i_{sd} and i_{sq}) is presented.

An open-loop test is firstly made. Figure 3.9 compares the electromagnetic torque T_e and stator current i_{sa} obtained after a fixed-point simulation with those obtained after a continuous-time simulation. These results are obtained during start up and with the same operating conditions (voltages references and load conditions). The compared waveforms are similar and the delta-based digital realization is then functionally validated in open loop.

A closed-loop test is also made. Figure 3.10 presents the offline closed-loop simulation results obtained after having associated the synchronous machine simulator IP with its hysteresis controller. These responses are obtained for the d-q component references of the stator current vector i_{sd}* and i_{sq}* equal respectively to 0A and 2A, a hysteresis controller bandwidth set to 0A, a time-step Δt equal to 1µs and a controller sampling period T_s equal to 100µs. The obtained results demonstrate the good functionality of the developed algorithm. The FPGA implementation step can then be started.

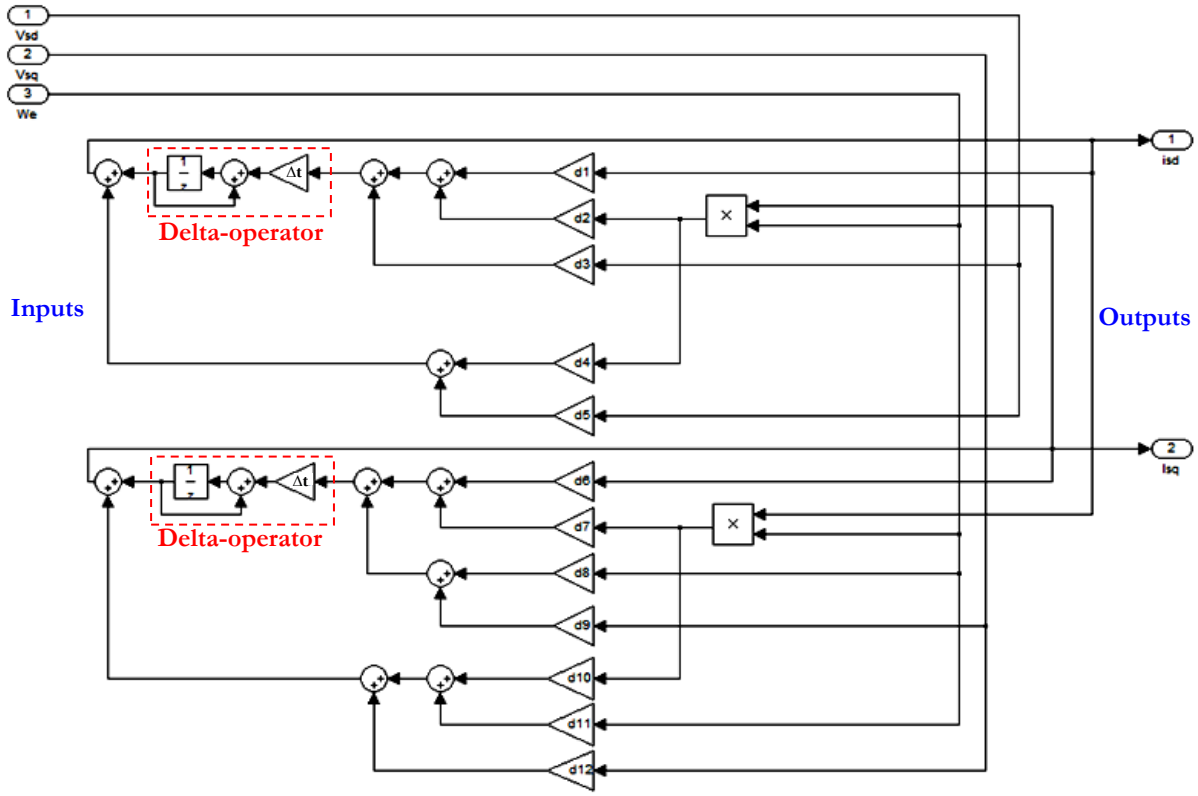


Figure 3.8: Structure of the delta-operator based electrical equations

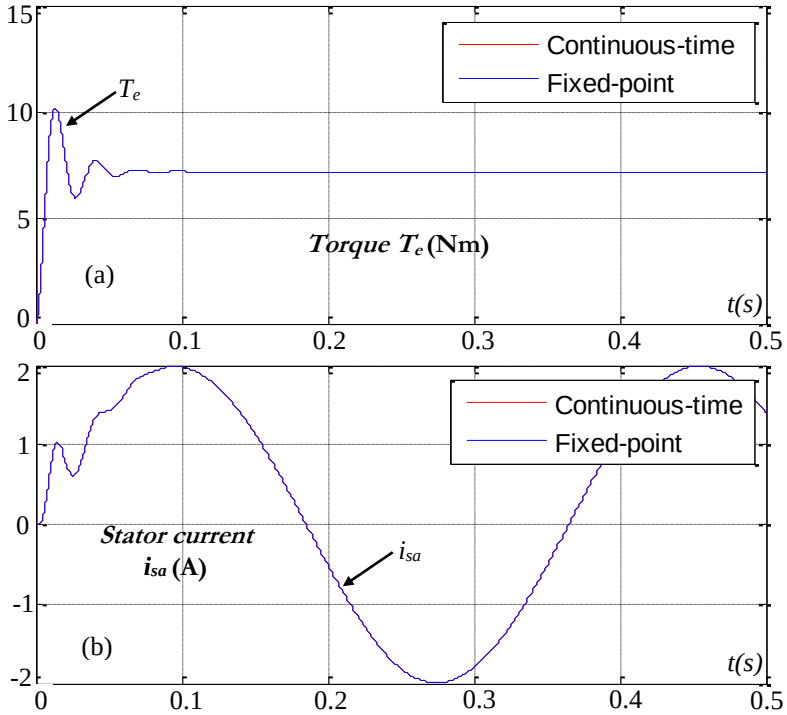


Figure 3.9: Open-loop simulation results: (a) electromagnetic torque T_e and (b) stator current i_{sa}

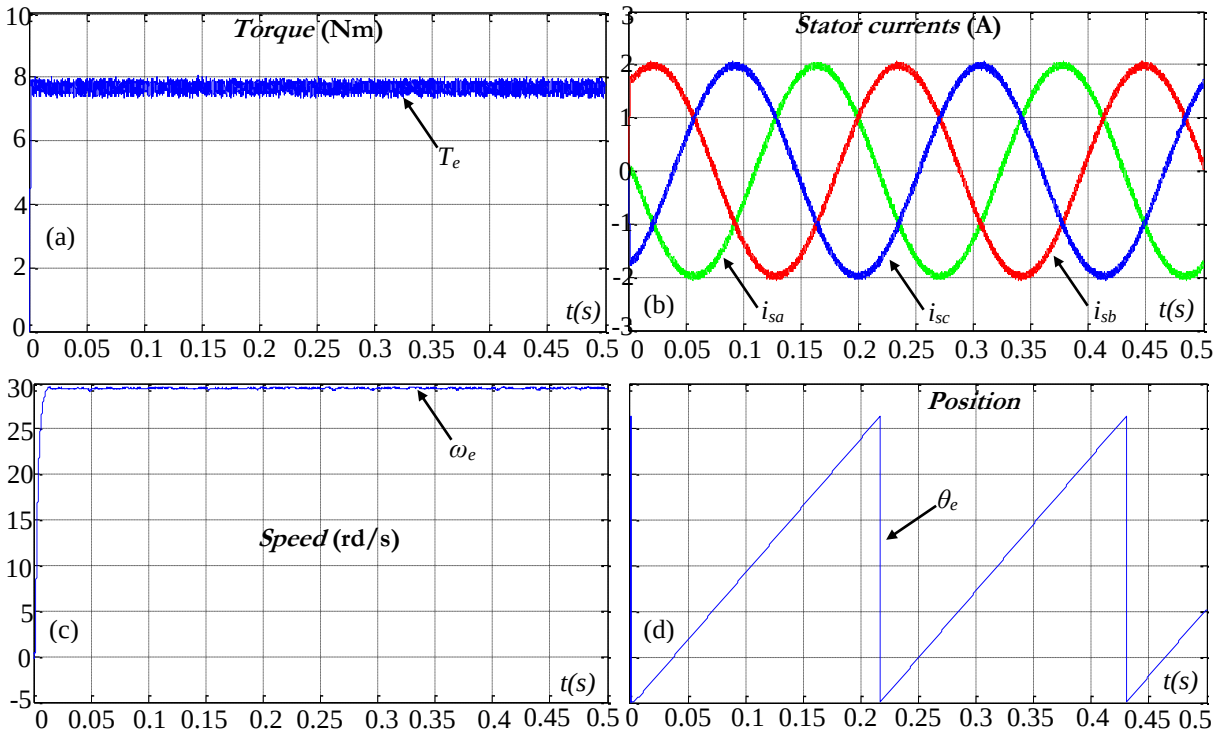


Figure 3.10: Closed-loop simulations results (a) electromagnetic torque T_e (b) stator currents $i_{si\ (i=a,b,c)}$ (c) speed ω_e (d) position θ_e

2.3. FPGA implementation

a- FPGA architecture design

To start with, the FPGA architecture has to be now designed. The hardware architecture of each of the developed modules (according to the adopted modular partitioning) has to be then implemented and both associated and properly synchronized.

The treatment here starts by activating the *Park* transformation module to compute the d-q stator voltages (v_{sd} and v_{sq}). These latter and the rotor speed ω_e (at previous time step) are then used to solve the electrical equations (i_{sd} and i_{sq}). This is followed by solving the mechanical equations (T_e, ω_e and θ_e). Finally, the *Park*$^{-1}$transformation module is activated and the 3-phase currents (i_{sa}, i_{sb} and i_{sc}) are then computed.

In order to optimize the use of FPGA resources, the data-path of each of module architecture is factorized. In order to increase speed performances, each data-path is also pipelined using a set of registers.

Figure 3.11 shows the factorized and pipelined data-path of the electrical equations which requires the most important hardware resources.

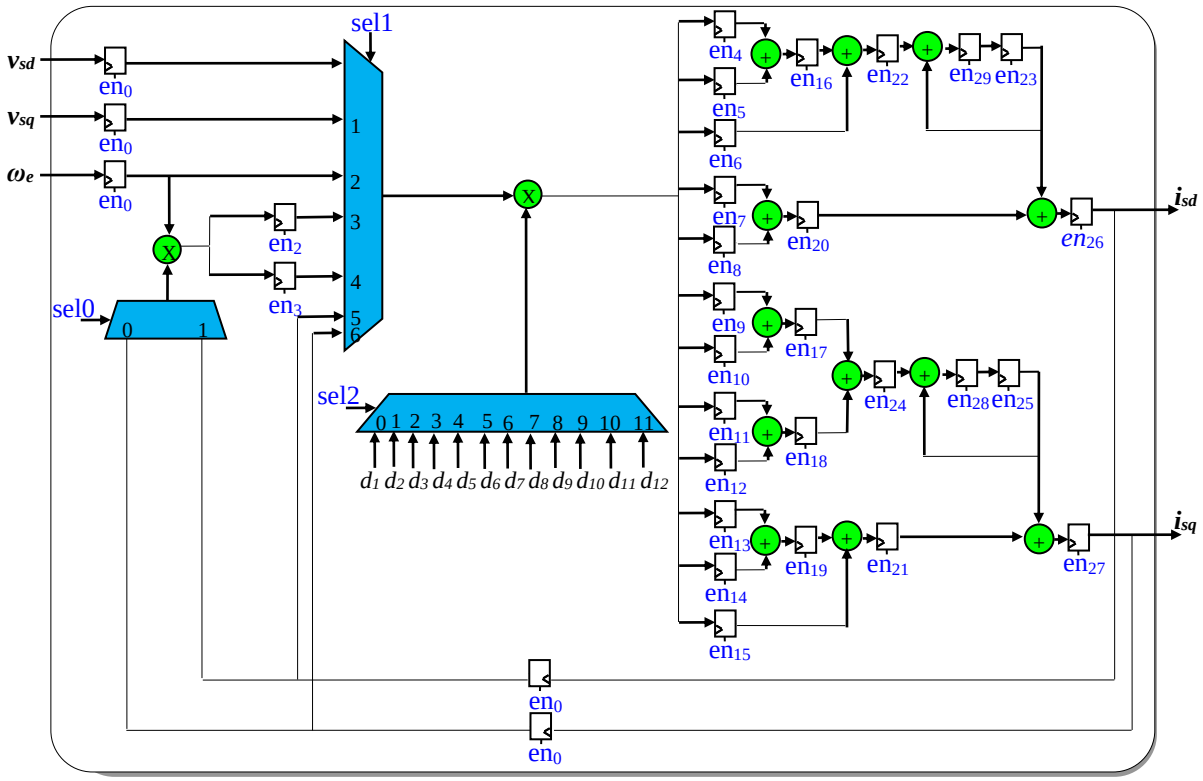


Figure 3.11: Architecture of delta-operator based synchronous machine real-time simulator - electrical module

b- Coding

Before being implemented on FPGA, the proposed architecture is coded in VHDL.

c- Functional validation

The developed VHDL code has been functionally simulated using ModelSim. The test has been done by applying Testbench stimulis to the inputs of the developed architecture. These stimulis have been extracted from their corresponding Matlab/Simulink counterparts.

In order to confirm the good functionality of the architecture, the same simulation test has been made in Matlab/Simulink environment with the same input stimulis. The obtained simulation results are then compared to those obtained with ModelSim.

Figure 3.12 shows almost no difference between Matlab/Simulink simulation results and ModelSim results. Therefore, the good functionality of the delta-operator based embedded synchronous machine real-time simulator is validated.

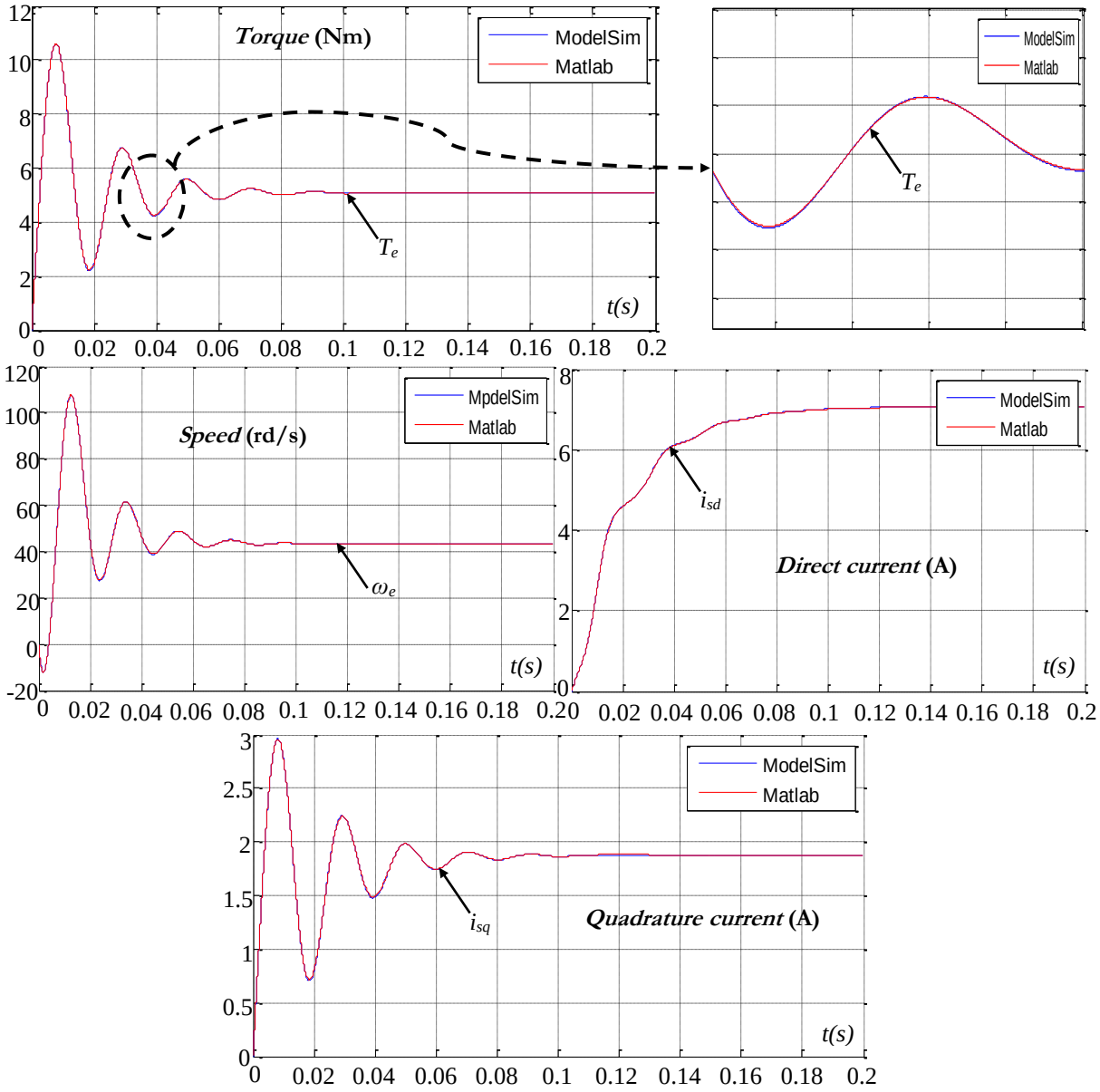


Figure 3.12: Comparison between Simulink continuous-time simulation results and ModelSim VHDL simulation results

d- Design/synthesis/place/route

Now as the global FPGA architecture is functionally validated, the developed design is then synthesised, placed and routed. Since the Xilinx FPGA device is targeted, the ISE tool has been then used. The consumed FPGA resources and the maximum clock frequency that can be used are evaluated and the time/area performances can therefore be analysed.

e- Time/Area evaluation

When the architecture is synchronized with a 100 MHz clock frequency, the computation time of the whole embedded synchronous machine real-time simulator is 0.96 µs. The developed architecture consumes only 5% of the available Zynq FPGA cells. Also, It uses 18 DSP48E1s units (8%), 1366 LUT (2%) and 1962 Flip-Flops (1%). As for the whole architecture including the controller, the machine simulator IP and the simulator IPs of the

inverter and load, it uses 7% of the available 13300 slices, 4% of LUTs, 2% of Flip- Flops and 9% of the available 220 DSP48E1s units.

2.4. Experimentations

a- HIL tests

The HIL tests are used here to validate the functionality of the embedded synchronous machine real-time simulator. The ChipScope tool has been used. The latter is used to probe the internal signals on the one hand and to configure the design on the other hand. The implemented design must be associated with specific cores (see Figure 3.13). These cores aim to control the USB-JTAG transmission with the Host-PC (Integrated Controller – ICON core), to probe data (Integrated Logic Analyzer – ILA core) and monitor the implemented design (Virtual Input/Output -VIO core), [XIL].

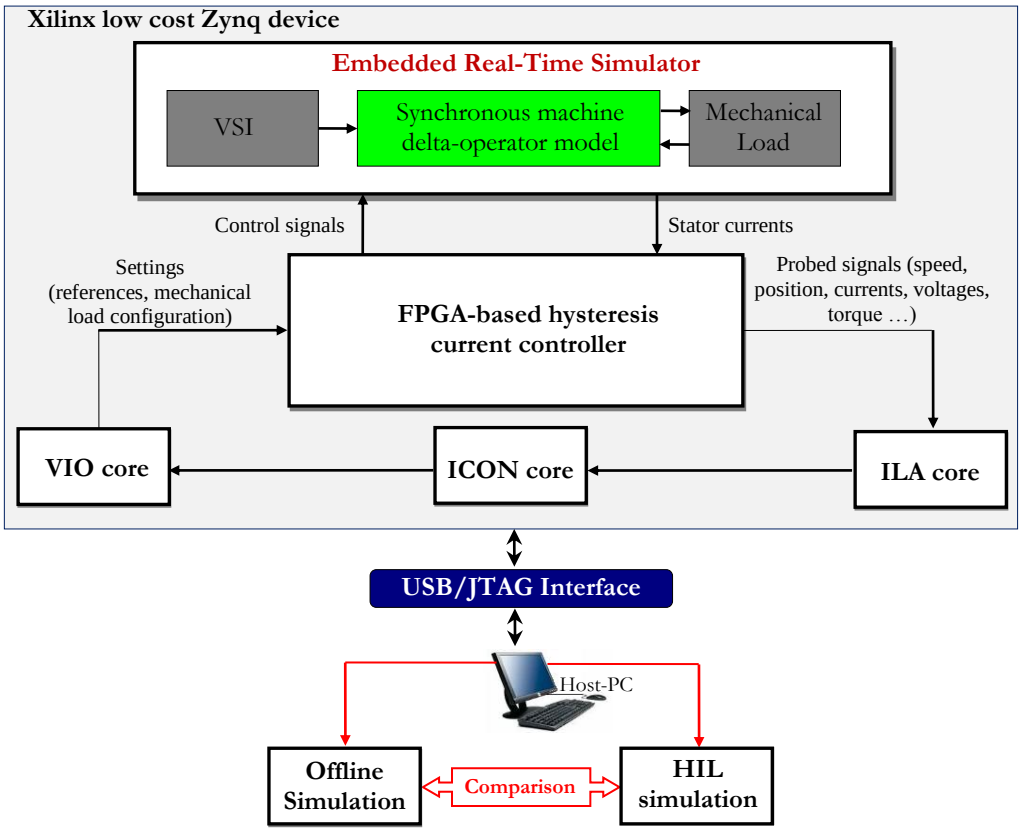


Figure 3.13:HIL validation of the synchronous machine delta-operator model.

Figure 3.14, Figure 3.15 and Figure 3.16 present respectively the waveforms of the electromagnetic torque, the stator currents, the rotor position and the rotor speed. These responses are obtained for the d-q component references of the stator current vector i_{sd}* and i_{sq}* equal respectively to 0A and 2A, an hysteresis controller bandwidth set to 0A, a time-step Δt equals to 1µs and a sampling period T_s equal to 100µs.When compared to offline responses presented in Figure 3.10, real-time responses confirm the good functionalities of the developed FPGA-based embedded synchronous machine real-time simulator IP.

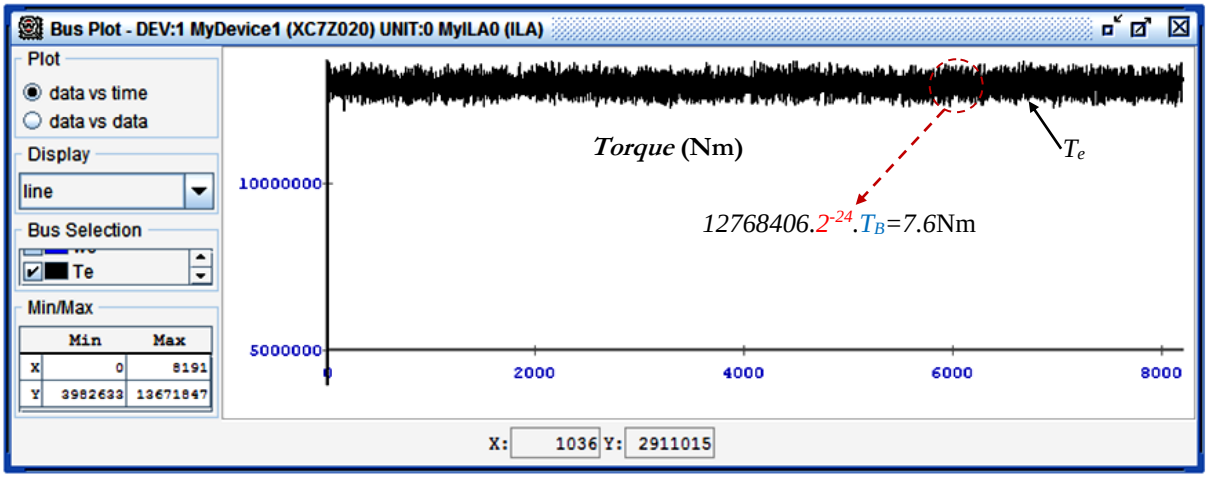


Figure 3.14: Waveform of the electromagnetic torque – HIL testing

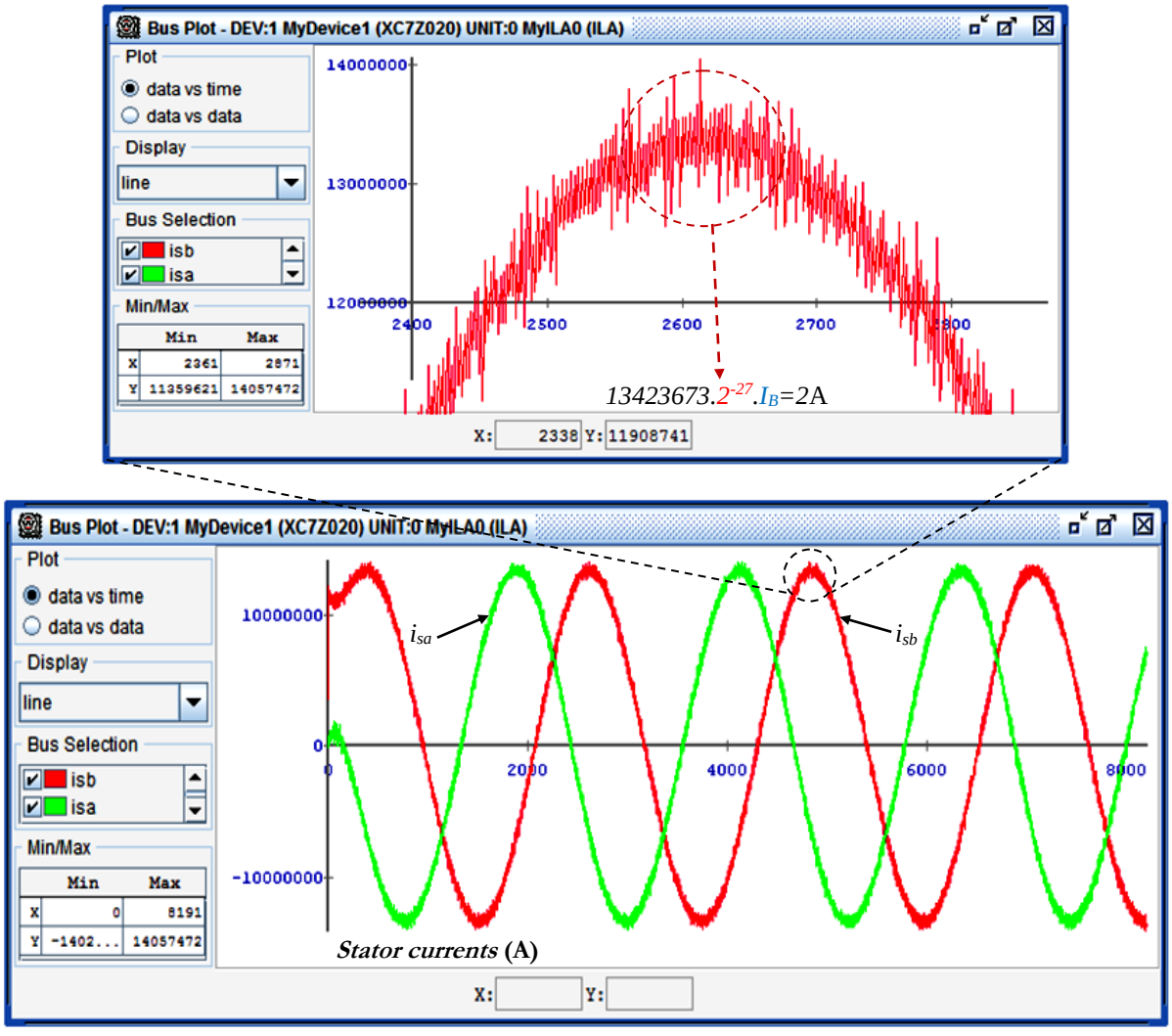


Figure 3.15: Waveforms of the stator currents – HIL testing

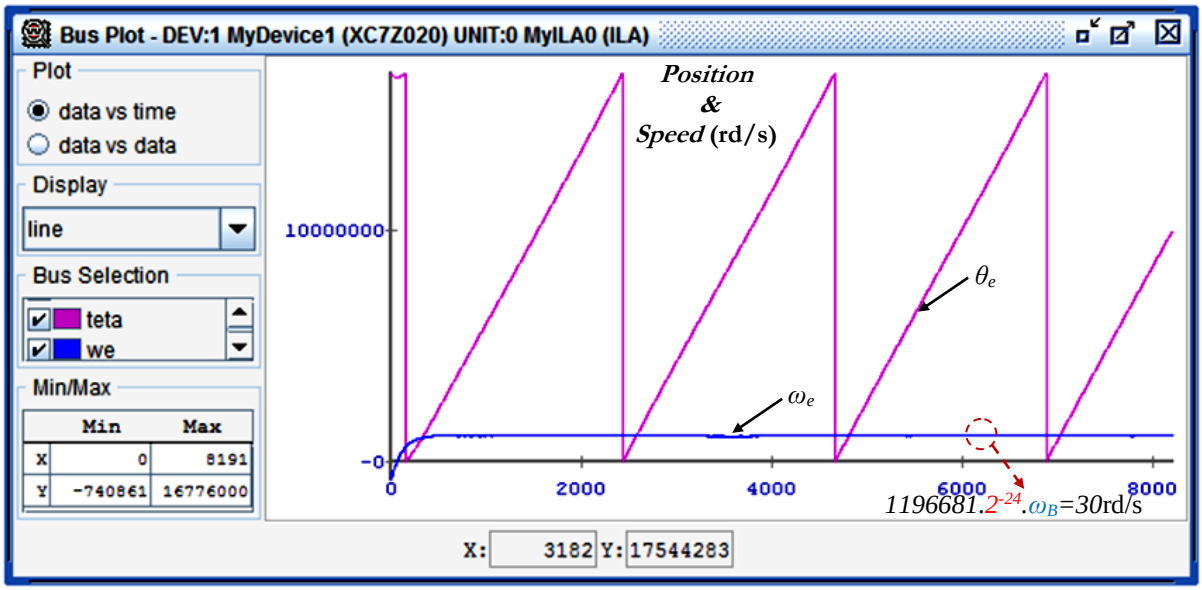


Figure 3.16: Waveforms of rotor position (blue) and rotor speed (red) – HIL testing

3. FPGA-based embedded real-time simulator of a 3-phase induction machine

Staying always with of the case of AC machines, the development of an embedded real-time simulator of a squirrel-cage induction machine is discussed in this section. The development of this simulator IP has been also achieved with the help of the proposed design guidelines.

For the same motivations explained before, the bi-phase model with external transformations is chosen to be implemented. This model is obtained here after Clarke transformations. The proposed α-β based model assumes that the machine is symmetrical, the saturations are neglected and the flux induction is sinusoidal. By choosing the stator currents and rotor fluxes as state variables, the state-space model of the induction machine in the stationary reference frame linked to the stator axis is given by the following equations (relation 3.11).

$$\begin{cases} s.i_{s\alpha} = -\gamma.i_{s\alpha} + \dfrac{M_{sr}}{\sigma L_s L_r T_r}.\emptyset_{r\alpha} + \dfrac{M_{sr}}{\sigma L_s L_r}.p.\omega_e.\emptyset_{r\beta} + \dfrac{1}{\sigma l_s}.v_{s\alpha} \\ s.i_{s\beta} = -\gamma.i_{s\beta} - \dfrac{M_{sr}}{\sigma L_s L_r}.p.\omega_e.\emptyset_{r\alpha} + \dfrac{M_{sr}}{\sigma L_s L_r T_r}.\emptyset_{r\alpha} + \dfrac{1}{\sigma l_s}.v_{s\beta} \\ s.\emptyset_{r\alpha} = \dfrac{M_{sr}}{T_r}.i_{s\alpha} - \dfrac{1}{T_r}.\emptyset_{r\alpha} - p.\omega_e.\emptyset_{r\beta} \\ s.\emptyset_{r\beta} = \dfrac{M_{sr}}{T_r}.i_{s\beta} + p.\omega_e.\emptyset_{r\alpha} - \dfrac{1}{T_r}.\emptyset_{r\beta} \\ s.\omega_e = \dfrac{p}{J}\dfrac{M_{sr}}{T_r}\left(\emptyset_{r\alpha}.i_{s\beta} - \emptyset_{r\beta}.i_{s\alpha}\right) - \dfrac{f}{J}.\omega_e - \dfrac{1}{J}.T_L \end{cases} \quad (3.11)$$

With $T_r = \frac{L_r}{R_r}; \sigma = 1 - \frac{{M_{sr}}^2}{L_r L_s}; \gamma = \frac{R_s + \frac{M_{sr}}{L_r T_r}}{\sigma L_s}$

The expression of the electromagnetic torque T_e is given in (3.12).

$$T_e = p\frac{M_{sr}}{T_r}\left(\emptyset_{r\alpha}.i_{s\beta} - \emptyset_{r\beta}.i_{s\alpha}\right) \tag{3.12}$$

The parameters of the induction machine under simulation are given in Appendix-D.

With respect to the followed design guidelines and the structure of the proposed IP-library, the α-β based induction machine model has been partitioned in three main modules. The first module corresponds to the *Clarke* transformation, the second one solves the electrical and mechanical equations and the last one corresponds to the *Clarke*$^{-1}$ transformation.

A digital realization based on delta-operator is chosen to be developed. As explained above, this operator is preferred since it allows using small simulation time-step and fixed-point format with limited data precision without affecting the stability. This is very important in the context of this work since it efficiently contribute to increase the accuracy and reduce the consumed resources.

To satisfy timing and algorithm constraints, the time-step is set to 1.4µs and the chosen fixed-point data format is set to 28Q24 (24 bits in the fractional part) for electrical and mechanical modules and 16Q15 bits for the rest of modules. To validate these choices, an offline fixed-point simulation is made.

Figure 3.17, Figure 3.18 and Figure 3.19 show respectively the responses of the electromagnetic torque, α-β stator currents and α-β rotor fluxes. These responses are obtained for sinusoidal input voltages.

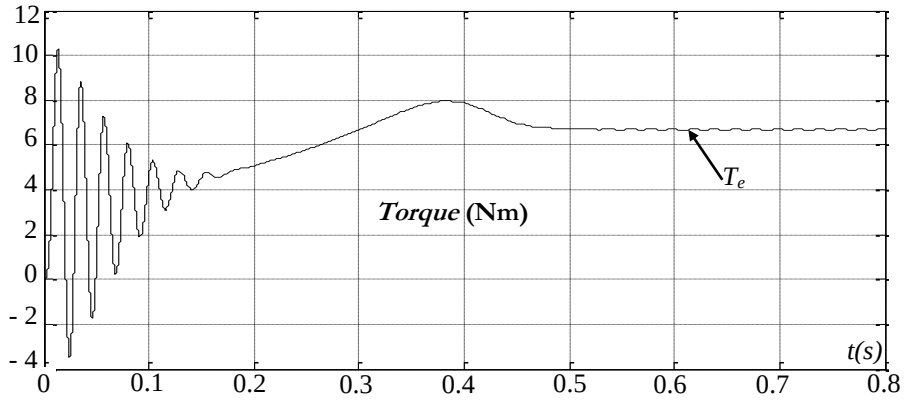


Figure 3.17: Response of the electromagnetic torque T_e

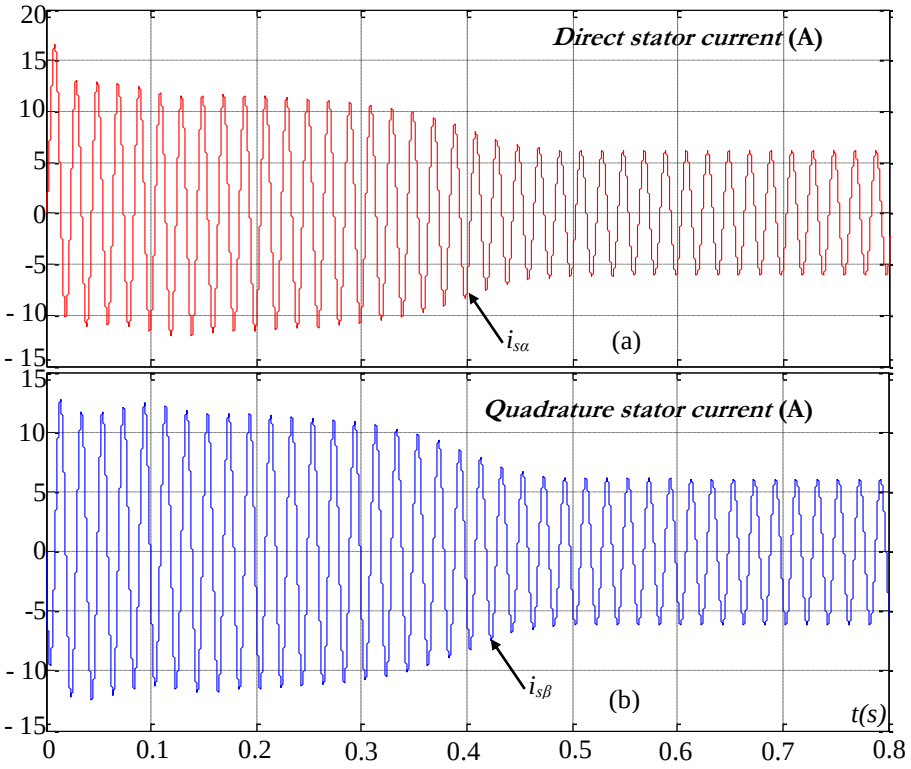


Figure : 3.18: Simulation results: (a) $i_{s\alpha}$ stator current (b) $i_{s\beta}$ stator current

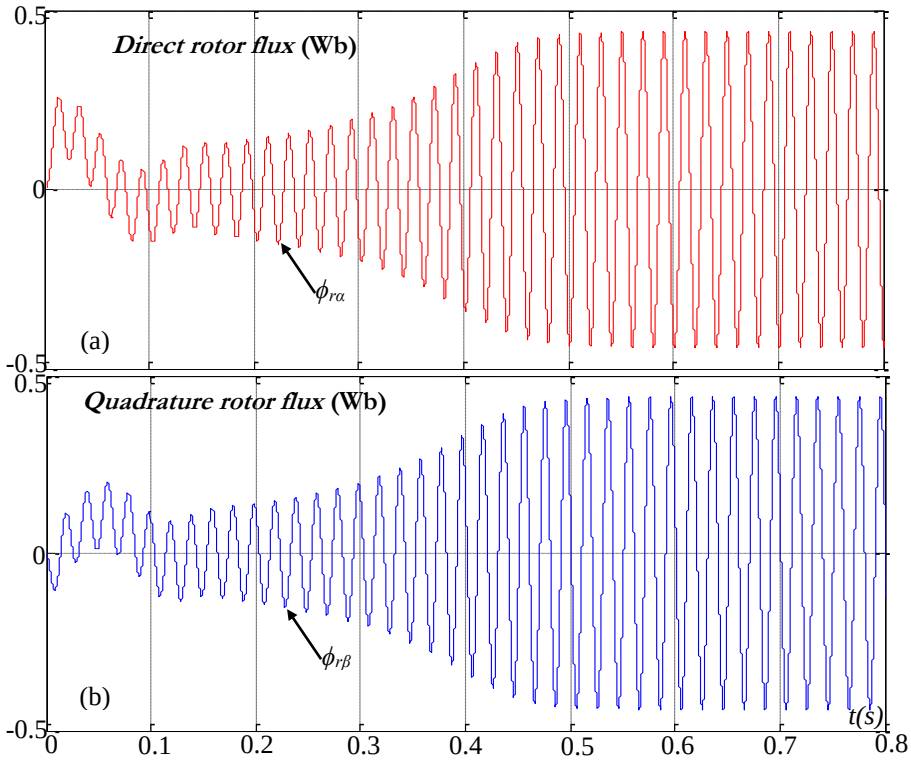


Figure 3.19: Simulation results: (a) $\phi_{r\alpha}$ rotor flux (b) $\phi_{r\beta}$ rotor flux waveforms

After having validated the developed digital realization, its corresponding FPGA-based architecture has been developed. To find the appropriate architecture that is able to perform the developed realization with regards to area/cost constraints, the A^3 and pipeline techniques have been adopted.

In order to verify the good functionality of developed architecture, an offline simulation is firstly done using ModelSim tool and compared to that achieved with the same input waveforms using Matlab/simulink tool. The simulation results presented in Figure 3.20 show a slight difference between Matlab/Simulink results and those of ModelSim. Therefore, the good functionality of the developed architecture of the embedded real-time induction machine simulator IP based on delta-operator is validated.

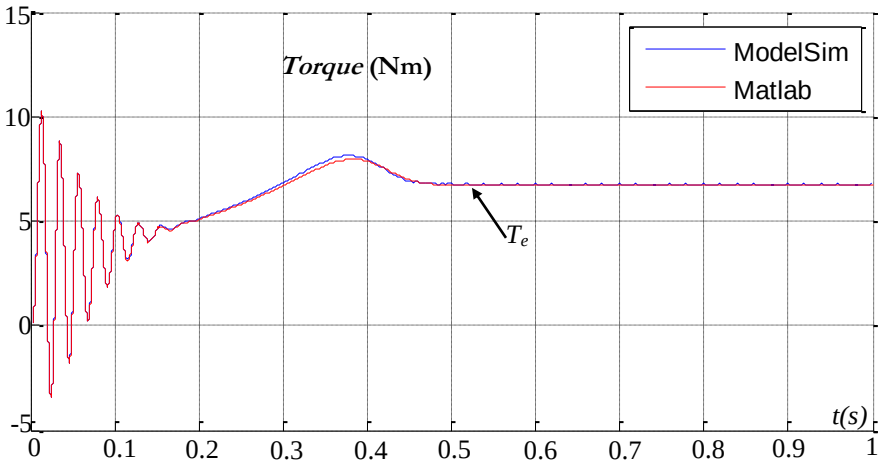


Figure 3.20: Matlab and ModelSim simulation results

Table 3.3 shows the obtained time/area performances of the developed FPGA-based architecture when using the Xilinx XC7Z020 Zynq device and 100 MHz clock frequency.The obtained time/area performances were satisfied the expected performances and the HIL validation has been initiated.

Table 3. 3: Time/area performances of the embedded real-time induction machine simulator

Module	Latency	Computation time
Clark transformation	7	70 ns
Electrical and mechanical model	99	*990* ns
Clark^{-1} transformation	8	80 ns
Total consumed resources 1272 out of 13300 (9%)		
LUTs	4%	
Flip-Flops	4%	
DSP48E1s	9%	

In the following, the real-time simulation results are presented. The same operating conditions (input voltages and load conditions) as during the fixed-point simulation are maintained. Figure 3.21, Figure 3.22 and Figure 3.23 present respectively the real-time responses of the electromagnetic torque, α-β stator currents and α-β rotor fluxes. As shown from all these figures, the FPGA-based real-time simulation results are similar to those of the offline Matlab/Simulink results presented in Figures 3.17, 3.18 and 3.19. The developed embedded induction machine real-time simulator IP is then ready to be added to the IP-library.

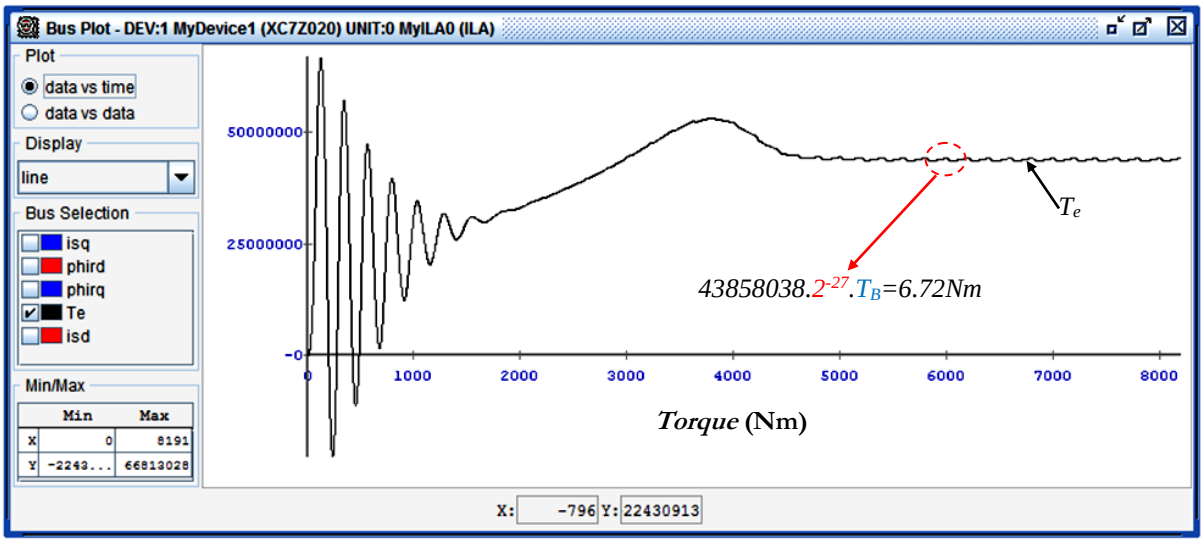


Figure 3.21: Real-time Waveform of the electromagnetic torque T_e

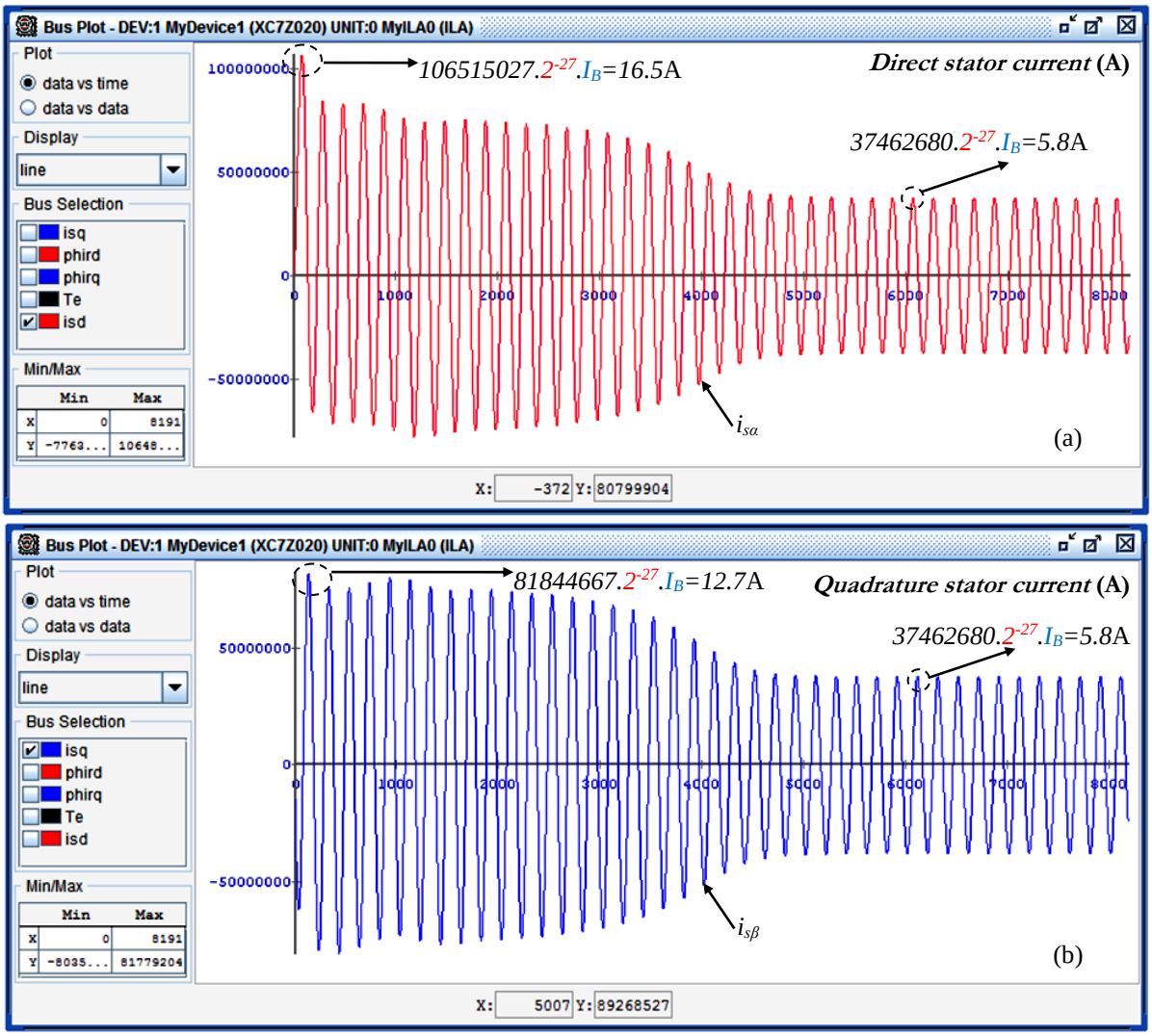


Figure 3.22: Real-time simulations results (a) $i_{s\alpha}$ (b) $i_{s\beta}$

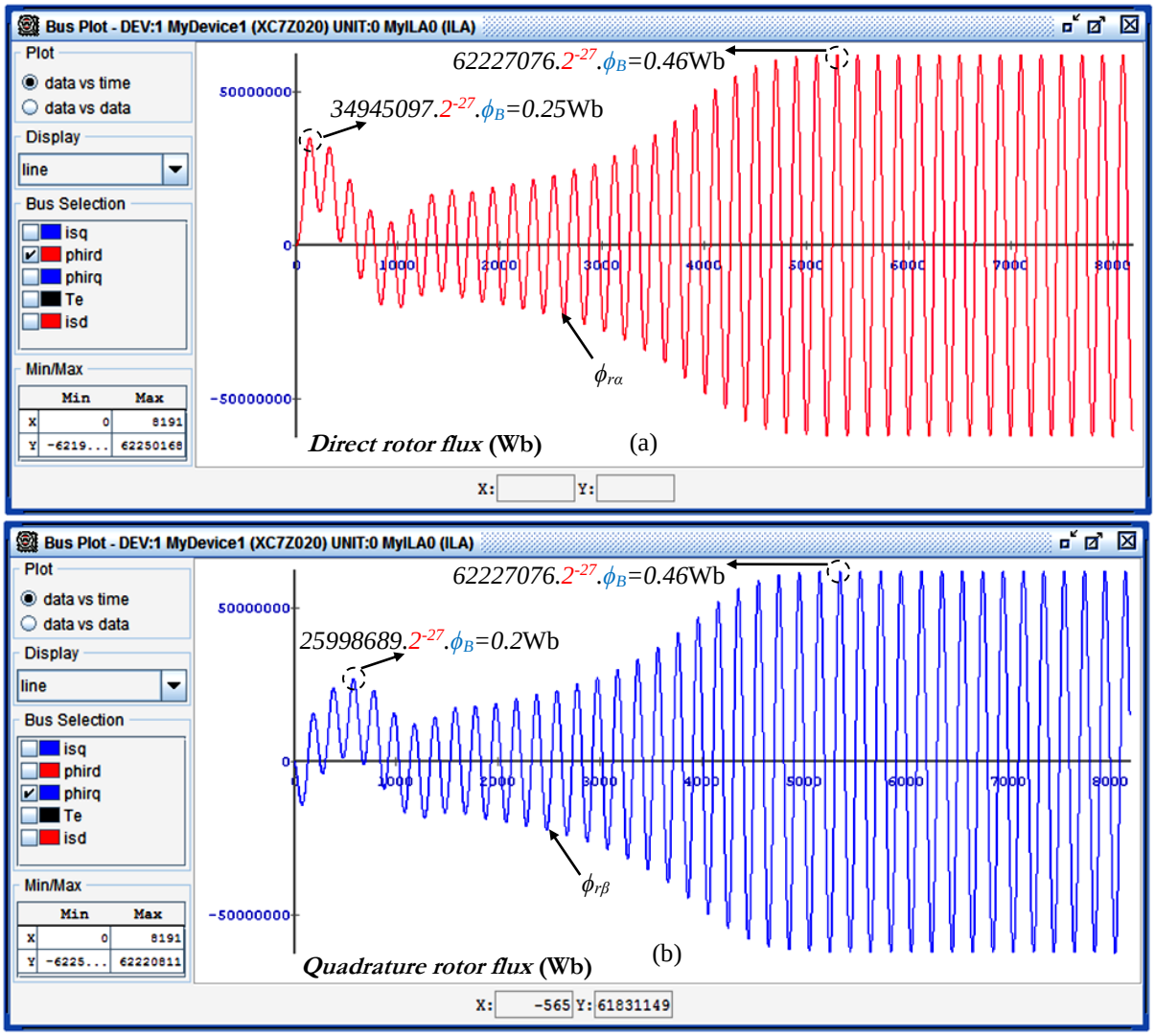


Figure 3.23: Real-time simulations results (a) $\phi_{r\alpha}$ (b) $\phi_{r\beta}$

4. FPGA-based embedded real-time simulator of a three-stage avionics alternator

In this section, the implementation in a low cost FPGA of an embedded real-time simulator IP of a three-stage avionics alternator is dealt with. The proposed simulator IP is intended to simulate a wide range of alternators with different power values and have small simulation time-step and moderate consumed resources.

Figure 3.24 shows the components of the three-stage avionics alternator to be simulated. This latter is composed of a DC-DC converter used to regulate the Main Generator (MG) output voltages through the exciter current, the Exciter Machine (EM) and a 3-phase Diode Rectifier (DR) which are mounted on the same shaft as the MG and an external nominal load connected to the MG through three dummy capacitors. More details about this system and the parameters of the used MG, EM and load are given in Appendix-E.

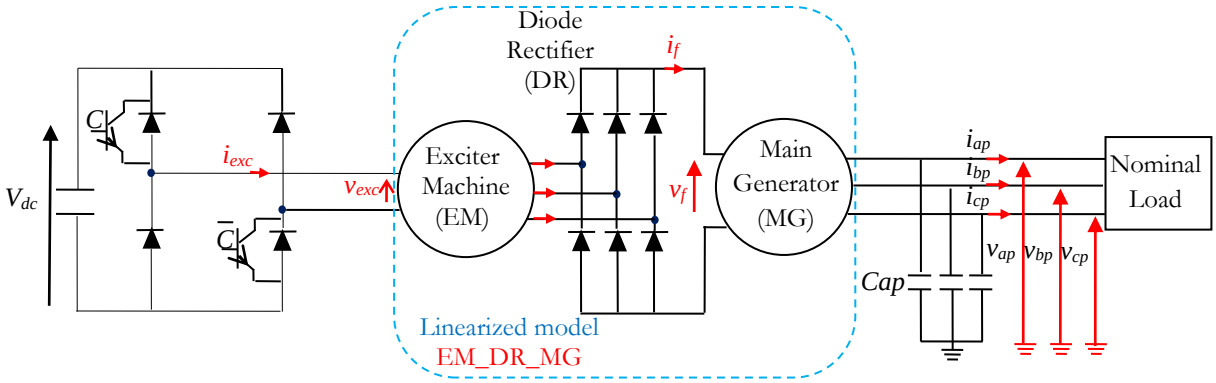


Figure 3.24: Structure of the three-stage avionics alternator

Always according to the proposed design guidelines and after the system specification, the next step consists in the system model selection. In order to find an optimum between the accuracy of the system model and its complexity, two main simplifications are achieved. The first one is related to the DC-DC converter. This latter has been modelled as a simple gain. The second simplification concerns the excitation system constituted by the EM and the DR. To reduce complexity and avoid adding dummy capacitors between these components, a linearization has been achieved.

The DR has been then linearized and its average-value model calculated in [ALI06] for each operating mode has been used.

The average value of the DR voltage is given in the following relation.

$$v_f = k_1 . M_{se} . \omega_{ee} . i_{exc} - k_4 . L_{de} . \omega_{ee} . i_f \qquad (3.13)$$

Where $k_4 = k_1 . k_3 + k_2$ is the correction factor due to the assumption that the DR supplies a fixed load ($i_{de} = k_3 . i_f$). k_1 depends on the rectifier operation mode and k_2 depends on the EM parameters (leakage inductance, magnetizing inductance of the d axis, etc.) and the operation mode of DR. k_1, k_2 and k_3 are assumed to be constant.

By associating these equations with those of the d-q based MG model given in Appendix-E, the equations of the linear model of the avionics alternator are obtained and given in the following relation. Note that the two last equations of this relation are used to model the presence of a damper cage in the rotor of the MG. The first one represents the damper winding located on the direct axis and the second one represents the winding located on the indirect axis. The accuracy of this linear model has been proved in previous studies ([BAR11]).

$$\begin{cases} i_{d1} = i_{dp} + Cap\,\omega_{ep} v_{qp} - Cap\dfrac{d_{v_{dp}}}{dt} \\ i_{q1} = i_{qp} + Cap\,\omega_{ep} v_{dp} - Cap\dfrac{d_{v_{qp}}}{dt} \\ v_{exc} = R_e i_{exc} + L_e\dfrac{d_{i_{exc}}}{dt} - k_s k_3 M_{se}\dfrac{d_{i_f}}{dt} \\ 0 = -v_{dp} - R_s i_{dp} + L_q\omega_{ep} i_{qp} - M_{sQ}\omega_{ep} i_Q - L_d\dfrac{d_{i_{dp}}}{dt} + M_{sf}\dfrac{d_{i_f}}{dt} + M_{sD}\dfrac{d_{i_D}}{dt} \\ 0 = -v_{qp} - R_s i_{qp} - L_d\omega_{ep} i_{dp} + M_{sf}\omega_{ep} i_f + M_{sD}\omega_{ep} i_D - L_q\dfrac{d_{i_{qp}}}{dt} + M_{sQ}\dfrac{d_{i_Q}}{dt} \\ 0 = -k_1 M_{se}.\omega_{ee}.i_{exc} - \alpha.i_f + L_f\dfrac{d_{i_f}}{dt} - M_{sf}\dfrac{d_{i_{dp}}}{dt} + M_{fD}\dfrac{d_{i_D}}{dt} \\ 0 = R_D i_D + L_D\dfrac{d_{i_D}}{dt} + M_{fD}\dfrac{d_{i_f}}{dt} - M_{sD}\dfrac{d_{i_{dp}}}{dt} \\ 0 = R_Q i_Q + L_Q\dfrac{d_{i_Q}}{dt} - M_{sQ}\dfrac{d_{i_{dp}}}{dt} \end{cases} \quad (3.14)$$

With $\alpha = k_4 L_{de}.\omega_{ee} + R_f i_f$.

Before starting the digital realization, the system model has been partitioned according to the proposed IP-library. To this purpose, three independent and reusable modules are extracted. The DC-DC converter module, the avionics alternator module described by equation 3.14 and the external nominal load module. In this section, the study is focused only on the avionics alternator IP module.

The Forward Euler discretization method has been adopted here. A 32Q27 fixed-point data format (with 27 fractional bits) has been used. Five bits are attributed to the dynamic part to avoid overflow, which can be appeared during load impact and load shedding.

The functional validation of the developed digital realization has been made in Matlab/Simulink environment. Figure 3.25 presents the obtained offline simulation results. They are obtained with an excitation voltage (v_{exc}) of the EM equal to 25.7V, an electrical speed (ω_{ee}) of the EM equal to 628.31rd/s and an electrical speed (ω_{ep}) of the MG equal to 314.15 rd/s. These results have been used in the FPGA implementation step to functionally validate the corresponding FPGA architecture.

This latter is presented in Figure 3.26. The global control unit activates firstly the DC-DC converter module. When the computation of the DC-DC converter module is achieved, the module of the load is activated. Finally, the EM_DR_MG module is activated. It is clear from the linear model (see equations above) that each equation is independent over each other. Therefore, they can be executed at the same time. For this reason, the linear model has been divided into six sub-modules that can be executed in parallel. Figure 3.27 presents the proposed sequential timing diagram for the three-stage avionics alternator FPGA-based architecture.

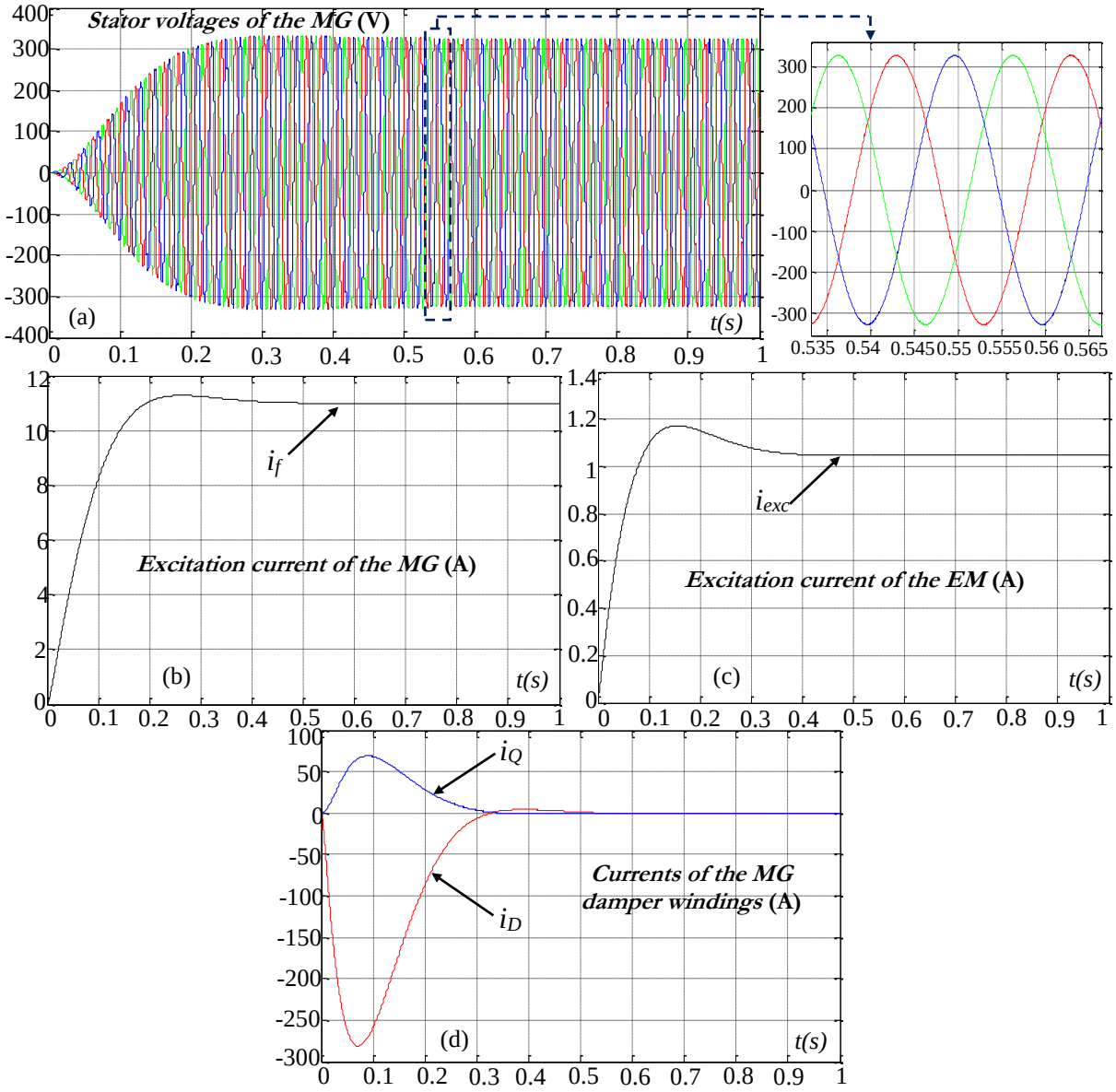


Figure 3.25: Offline simulation results of the developed digital algorithm a) Simple stator voltages of the main generator b) Excitation current of the main generator c) Excitation current of the exciter machine d) currents of the main generator damper windings

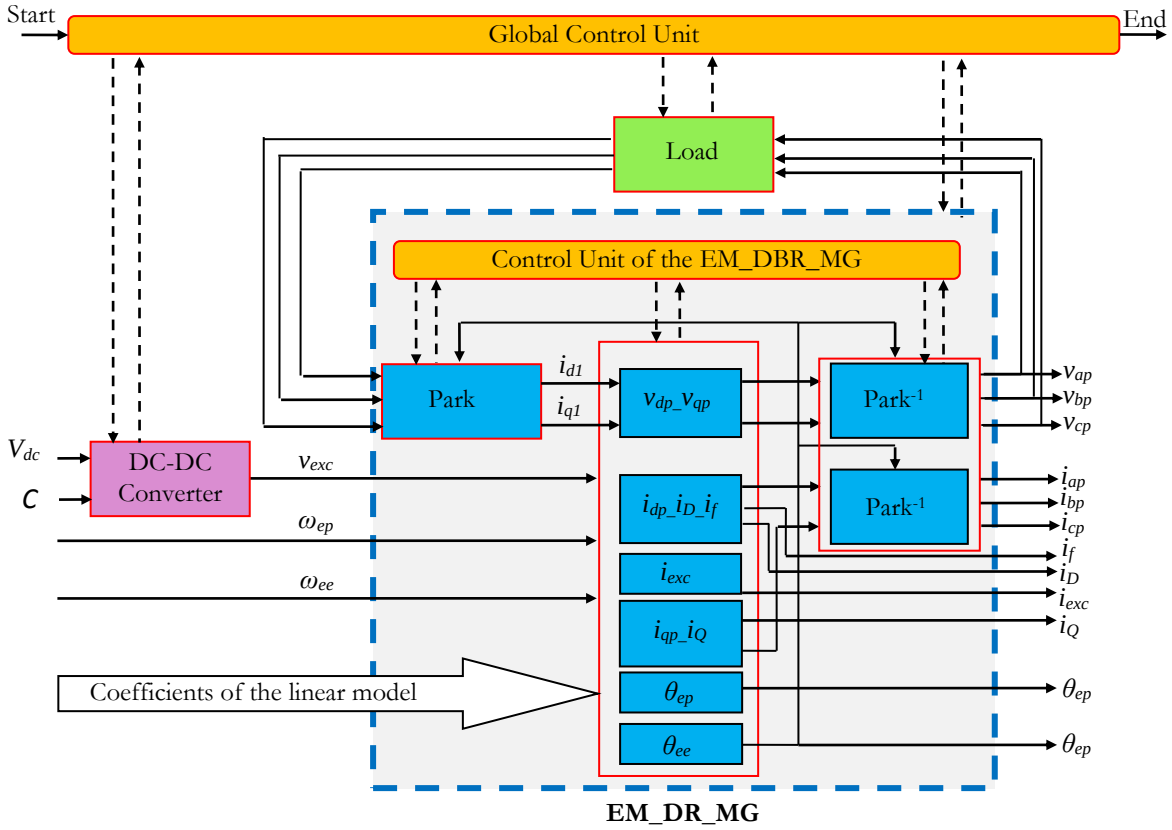


Figure 3.26: Global hardware architecture of the embedded three-stage avionics alternator real-time simulator

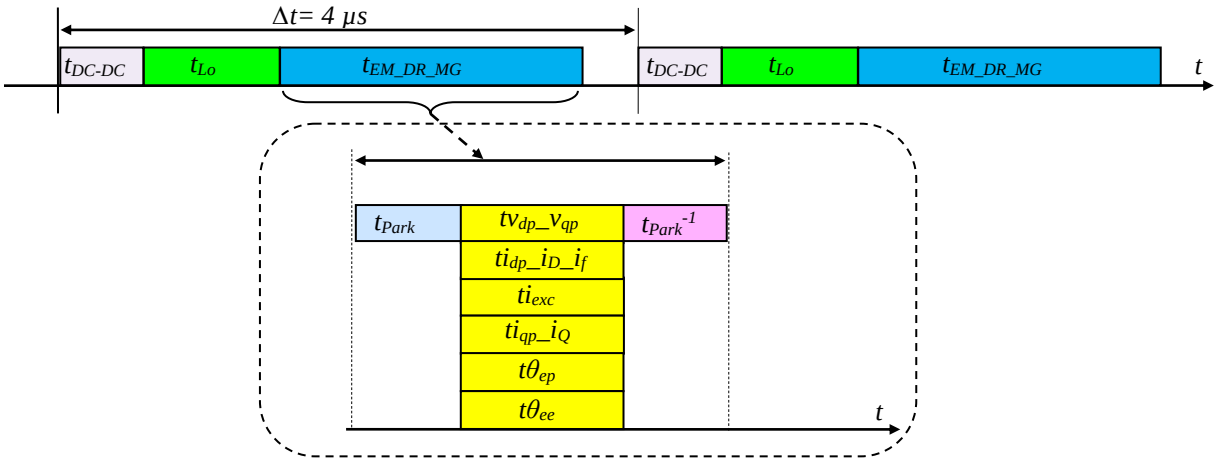


Figure 3.27: Sequential timing diagram of the developed FPGA-based architecture

To achieve the best compromise between computation time and consumed resources, the equations that have the same data-path like v_{dp} and v_{qp} have been gathered into a same module and executed in sequential mode.

In order to satisfy the implementation and timing constraints, the architecture of each extracted module has been factorized and pipelined. Figure 3.28 presents the designed hardware architecture of the $v_{dp}\_v_{qp}$ module.

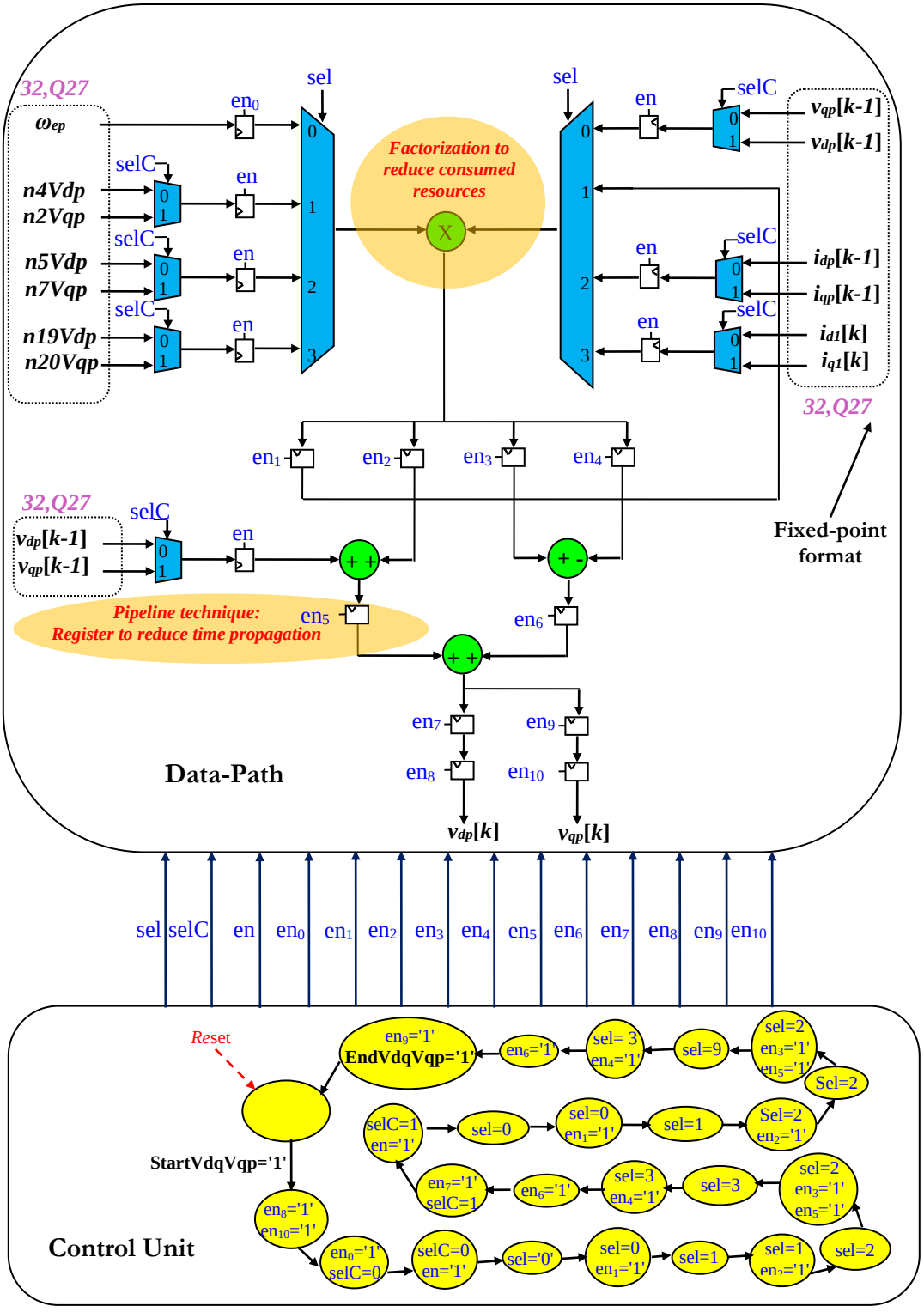


Figure 3.28: Hardware architecture of the Vdp_Vqp sub-module

Note that in order to develop a generic architecture able to simulate a wide range of alternators with different power values, all coefficients of the model are considered as signals that can be modified in real-time to simulate another avionics alternator with another power value.

After the VHDL-coding, a functional simulation of the design is done using ModelSim tools. Since they have been stimulated with the same inputs, the VHDL model and the

Matlab/Simulink model have been compared. Once this is done, an evaluation of the time/area performances is made. Table 3.5 presents the FPGA time/area performances of the designed FPGA-based architecture.

These results are obtained with a Xilinx XC7Z020 Zynq device and 100 MHz clock frequency. The whole computation time of the developed architecture T_c is 1.6 µs. The consumed resources are only 14% of the available ones. It is clear from the Table 3.4 that the developed FPGA-based architecture matches all the timing, modularity, algorithm and area/cost constraints.

Table 3. 4: FPGA Time/Area performances of the developed embedded real-time three-stage avionics alternator

Module	Latency	Computation time
Exciter-machine_Diode-rectifier_Main-generator	141	$t_{EM\_DR\_MG}$= 1410 ns
DC-DC Converter	1	$t_{DC\text{-}DC}$= 10 ns
Nominal Load	18	t_{Lo}= 180 ns
Computation time $T_c = t_{EM\_DR\_MG} + t_{Lo} + t_{DC\text{-}DC}$		T_c= 1.6 µs
Total consumed resources		1868 out of 13300 (14%)
LUTs		6%
Flip-Flops		5%
DSP48E1s		16%

Figure 3.29 presents the real-time waveforms of stator voltages of the main generator, the excitation current of the main generator, the excitation current of the exciter machine and currents of the main generator damper windings. These responses are obtained for the references of the excitation voltage of the exciter machine (v_{exc}), the electrical speed of the exciter machine (ω_{ee}) and the electrical speed of the main generator (ω_{ep}) equal respectively to 25.7V, 628.31rad/s and 314.15 rad/s.

When compared to offline responses (Figure 3.25), real-time responses (Figure 3.29) show the good functionalities of the developed FPGA-based embedded real-time simulator of the three-stage avionics alternator. This simulator IP is then added to the IP-Library.

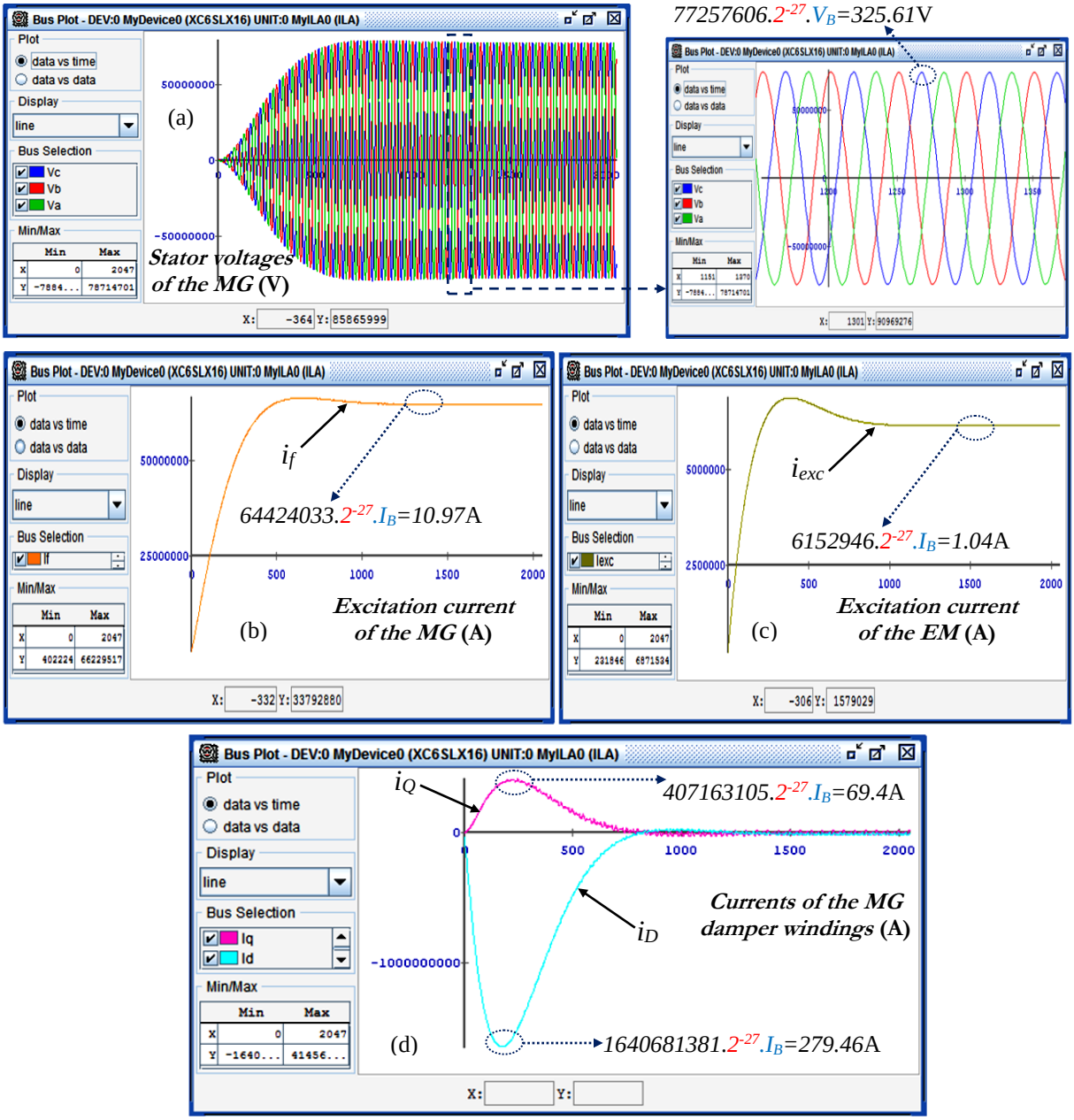


Figure 3.29: Real-time emulation results of the developed FPGA-based architecture (a) Simple stator voltages of the main generator (b) Excitation current of the main generator (c) Excitation current of the exciter machine (d) currents of the main generator damper windings

5. Conclusion

In this chapter the implementation in low cost FPGA of embedded real-time simulator IPs of electromagnetic elements was discussed. The proposed simulators IPs was mainly intended for embedded control applications but they can be also used for HIL applications. The implementation of three electromagnetic IPs modules has been studied. Thus, the embedded real-time simulator of a 3-phase synchronous machine, the one of a 3-phase induction machine and finally the one of a three-stage avionics alternator have been all implemented and added to the IP-library. The implementation has been achieved with the help of the proposed guidelines presented in the previous chapter.

The obtained results show the ability of all developed electromagnetic IPs to reproduce accurately the dynamic behaviors of the simulated electromagnetic elements. They show also that all developed simulator IPs enables the use of small simulation time-step and consumes moderate hardware resources. Their ability to address both embedded control and HIL applications was then confirmed.

In this chapter, the advantages of using delta operator rather than shift operator to design digital realizations of AC machines were also examined. It has been shown that delta-operator based realization is more convenient in the context of this work since it allows using small simulation time-step and limited fixed-point data word length without affecting the system stability.

In the next chapter the implementation in low cost FPGA of embedded real-time simulator IPs of switching elements will be discussed.

Chapter 4

Implementation in low cost FPGA of embedded real-time simulator IPs of switching elements

1. Introduction

This chapter is now discussing the development of the FPGA-based embedded real-time simulator IPs of the switching elements of electrical systems. As discussed in chapter 1, the modeling of a power converter can be achieved with two approaches: the modeling at system scale or the modeling at switch scale. This last approach has been privileged in this work since it allows the modeling of the power switches individually, leading then to accurate models. However, the main challenge is how to cope with their complexity having in mind that very short time steps are required and the FPGA resources are limited (due to the cost).

Three power converter topologies have been studied: a single-phase DC-AC converter, a 3-phase Voltage Source Inverter (VSI) and a 3-phase Diode Rectifier (DR). For all these IPs, the ADC-based modeling approach has been adopted. Here again, the design guidelines proposed previously have been followed.

2. FPGA-based embedded real-time simulator IP of a single-phase DC-AC power converter

2.1. Preliminary system specification

The topology of the simulated converter is a H-bridge DC-AC converter composed of SKM 50GB123D IGBT/Diode modules, 1200V/50A, operating at 20 kHz maximum switching frequency.

The developed simulator IP has been firstly validated in the context of HIL testing (See Figure 4.1) and the used FPGA target is a Xilinx XC7Z020 Zynq device. Indeed, to make a complete validation, this IP is connected to the simulator of a 1-phase AC load with sinusoidal back-EMF (the parameters are given in Table 4.1) and a current controller (based on a P+Resonant regulator). More details about these additional IPs are given in Appendix-F.

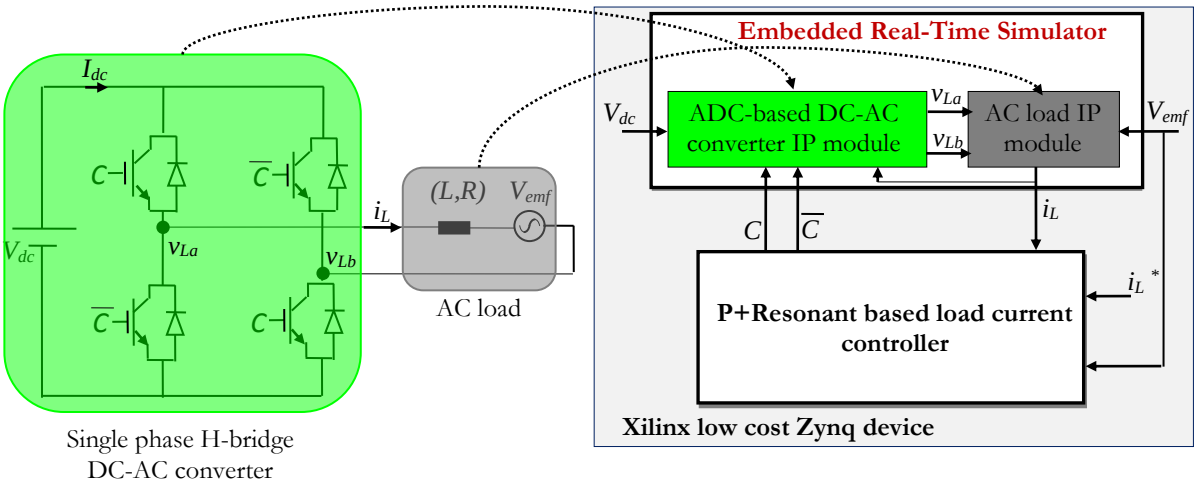


Figure 4. 1: Structure of the developed control system

Table 4. 1: Circuit parameters of system under study

Circuit parameters		Value
Resistance	(R)	4.5 Ω
Inductance	(L)	5 mH
DC bus voltage	(V_{dc})	500 V
Back EMF voltage	(V_{emf})	110 V_{RMS}
Back EMF frequency		50 Hz

2.2. Algorithm development

In the chosen validation context, the developed DC-AC converter simulator IP computes the line voltages v_{La} and v_{Lb} from the AC load current i_L, the dc voltage V_{dc} and the control signal C (see Figure 4.1).

a- Model selection

The adopted modeling approach is based on the two-valued ADC equivalent switch model. This approach is privileged since it allows representing each switch individually which allows a more accurate modeling of switching dynamics.

Furthermore, compared to other two-valued models like small/large resistor model, the ADC approach allows (with specific assumptions as will be seen after) the extraction of a constant conductance matrix. This has the credit of moderating the complexity of the algorithm since the inversion of this matrix can then be calculated on-time and offline.

Basically, an ADC equivalent model consists in representing a power switch by a small inductance (L_{sw}) when the switch state is ON and by a small capacitance (C_{sw}) when it is OFF, [PEJ94]. To solve the problem of overshoots and oscillations during commutations (due to LC circuit), a resistance (R_{sw}) is placed in series with the capacitance which acts as a damping element, [MAT10], [DAG12].

After the discretization of the L-circuit and the RC circuit (see digital realization section), a common equivalent circuit for each switch is extracted. This circuit is composed of a dependent current source in parallel with a conductance.

Figure 4.2 shows the obtained ADC-based equivalent model for the studied power converter. It is made by the association of the ADC-based model of each switch.

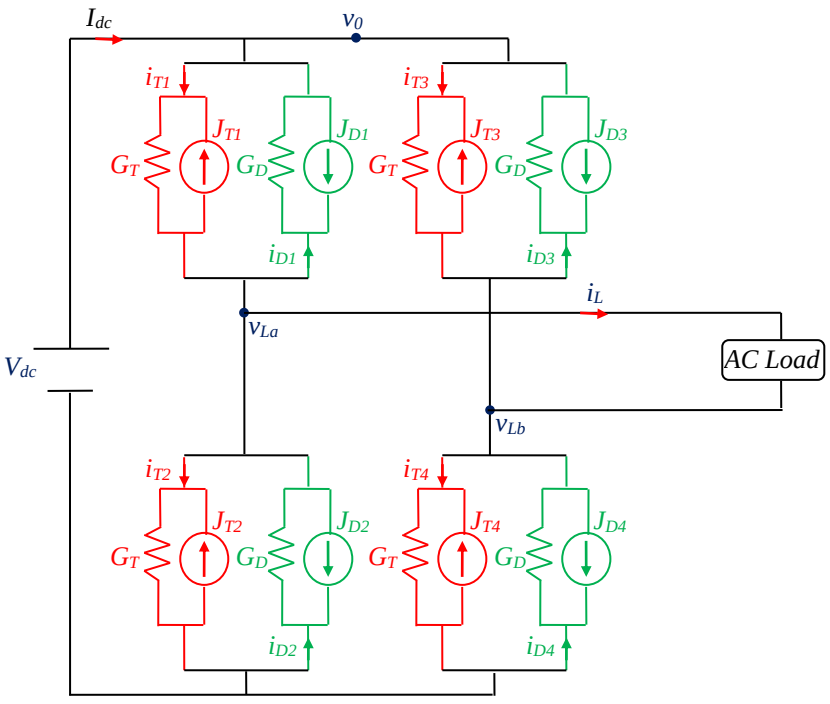


Figure 4. 2: Equivalent ADC-based model of the single phase DC-AC converter

In this equivalent circuit, J_{Ti} and J_{Di} are respectively the dependent currents sources of the i[th] controllable (IGBT) and uncontrollable (diode) switches. v_0, v_{La} and v_{Lb} are the node voltages. i_{Ti} and i_{Di} are the branch currents. G_T and G_D are the equivalent conductances. As it will be seen in the digital realization section, all these data are updated at each time-step Δt according to the switch state.

Then, the state of the IGBT is updated depending on the control signal, the voltage nodes and the branch current as given in the following logic equation.

$$s_T[k] = C[k].(i_T[k-1] \geq 0) \tag{4.1}$$

As for the diode, the state is S updated as follows:

$$s_D[k] = s_D[k-1].(i_D[k-1] \geq 0) + \overline{s_D[k-1]}\,.(V_D[k-1] \geq 0) \tag{4.2}$$

Thus, when the switch is ON:

$$\begin{cases} J_{T/D}[k] = -i_{T/D}[k-1] \\ G_{T/D} = \dfrac{\Delta t}{L_{sw}} \\ V_{T/D}[k] = f(v_0, v_{La}, v_{Lb}) \\ i_{T/D}[k] = G_{T/D}.v_{T/D}[k] + i_{T/D}[k-1] \end{cases} \tag{4.3}$$

and when it is OFF

$$\begin{cases} J_{T/D}[k] = G_{T/D}.V_{T/D}[k-1] \\ G_{T/D} = \dfrac{C_{sw}}{R_{sw}.C_{sw} + \Delta t} \\ V_{T/D}[k] = f(v_0, v_{La}, v_{Lb}) \\ i_{T/D}[k] = G_{T/D}.v_{T/D}[k] - J_{T/D}[k] \end{cases} \tag{4.4}$$

To extract the relations between voltages and currents of the whole power converter, a nodal analysis is made. The Modified Nodal Analysis (MNA), [PEJ94], can be applied to extract these relations as a matrix equation:

$$H.x[k] = b[k] \tag{4.5}$$

Where, H is the conductance matrix. The vector $x[k]$ gathers the node voltages $v_0[k]$ and $v_{Li(i=a,b)}[k]$ and the dc-current $I_{dc}[k]$. The vector $b[k]$ is built from combinations, at each voltage node, of $J_{Ti}[k]/J_{Di}[k]$ and $i_L[k]$. It also includes the dc-voltage $V_{dc}[k]$. The system solution is computed at each simulation time-step Δt by solving the matrix equation (4.6).

$$x[k] = H^{-1}.b[k] \tag{4.6}$$

The conductance of each power switch varies with regards to its state. To optimize the complexity and avoid online matrix inversion, one can define a relationship between L_{sw}, C_{sw}, R_{sw} and Δt that makes the conductance matrix constant whatever the switch state. This relation is then:

$$R_{sw} = \frac{L_{sw}}{\Delta t} - \frac{\Delta t}{C_{sw}} \tag{4.7}$$

b- Modular partitioning

As shown in Figure 4.3, the overall mathematical equations of the previously discussed ADC-based model have been divided into four independent and reusable modules. These modules have been organized hierarchically with respect to the proposed IP-library and are located in level 2 and level 3 (Figure 2.7, chapter 2). Each module is subdivided into sub-modules from the lower levels.

The first module updates the states (ON/OFF) of switches depending on the control signals provided by the controller and depending on the switch voltages $V_{T/Di(i=1\dots4)}[k\text{-}1]$ and currents $i_{T/Di(i=1\dots4)}[k\text{-}1]$. The second one computes the currents sources ($J_{Ti(i=1\dots4)}[k]$ and $J_{Di(i=1\dots4)}[k]$) of each switch according to its new state. The third one computes the $v_0[k]$ and $v_{Li(i=a,b)}[k]$ node voltages and $I_{dc}[k]$ current. The last module computes the currents ($i_{Ti(i=1\dots4)}[k]$ *and* $i_{Di(i=1\dots4)}[k]$) and the voltages ($V_{Ti(i=1\dots4)}[k]$ *and* $V_{Di(i=1\dots4)}$ $[k]$) of each switch.

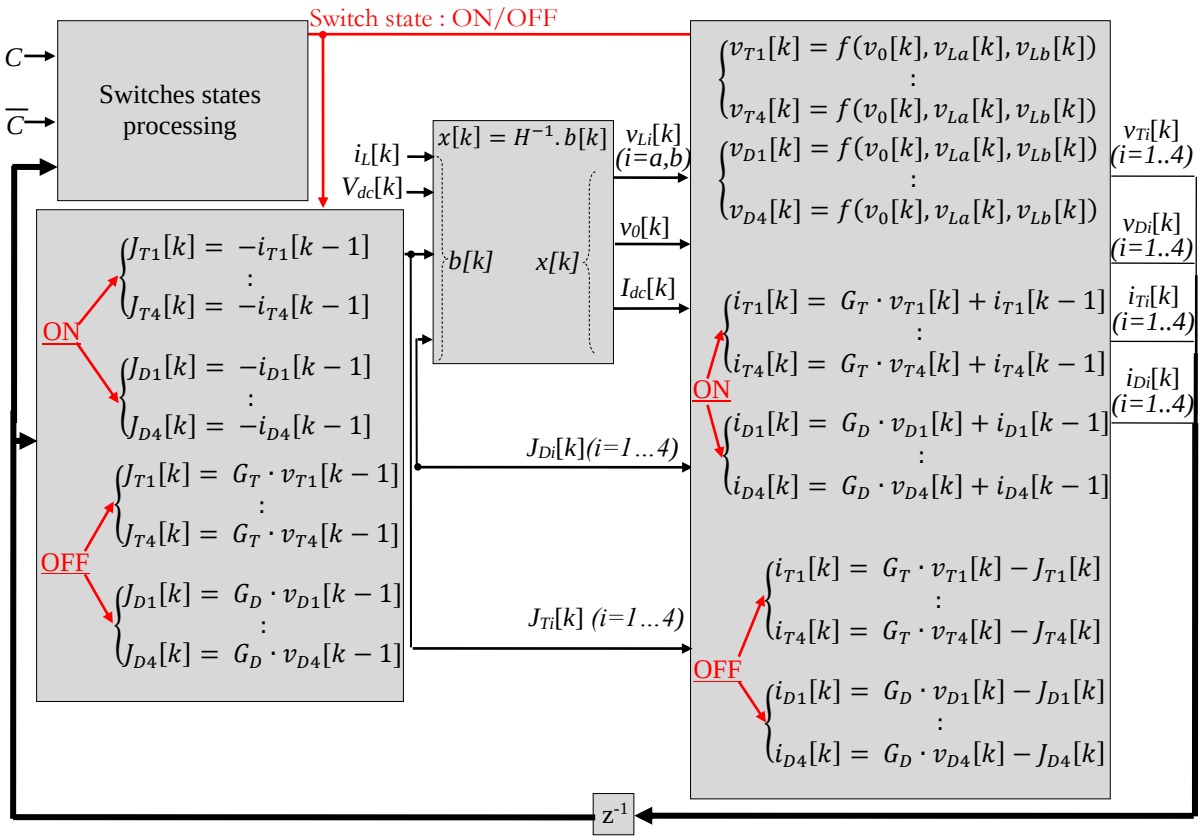


Figure 4. 3: ADC-based model of the single phase DC-AC converter – Modular partitioning

Note that with regards to this modular partitioning, the computation of equations of each module can be made in parallel since they are independent. As a consequence, the whole computation time is independent on the topology of the power converter and remains always the same as it will be seen for the case of 3-phase power converter (sections 3 and 4).

c- Digital realization

To satisfy timing constraints, the simulation time-step of the DC-AC converter real-time simulator IP has been set to 500ns. Indeed, the maximum switching frequency of the used power converter is equal to 20 kHz. Thus, the simulation time-step is therefore 100 times less than the corresponding minimum switching period and the inter simulation time-step errors are then neglected.

The discretization of the ADC-based model has been made using the Backward Euler method. This method is chosen since it allows reducing numerical oscillations during switching events [PEJ94]. As for data quantification, the chosen fixed-point data format is set to 24Q20 bits (with 20 fractional bits), which has been determined after simulation tests.

d- Algorithm validation

The algorithm validation is achieved with the help of Matlab/Simulink tools. The SimPowerSystems based model, with a variable time-step, is taken as a reference design. An open-loop validation is firstly made. Figure 4.4 compares, in the same test conditions, the waveform of the AC load current obtained with the ADC-based model to that obtained with SimPowerSystems based model. The compared waveforms are similar.

The developed algorithm has been also validated in closed-loop. Figure 4.5 gives the waveform of the AC load current in the case of a closed-loop test. A sinusoidal reference current (5A, 50Hz) has been applied at the beginning. Then, the amplitude of this current has been multiplied by 2 at 45ms. As it can be seen, here again, the closed-loop simulations results obtained with ADC-based model and its SimPowerSystems counterpart are similar. The developed algorithm is then validated.

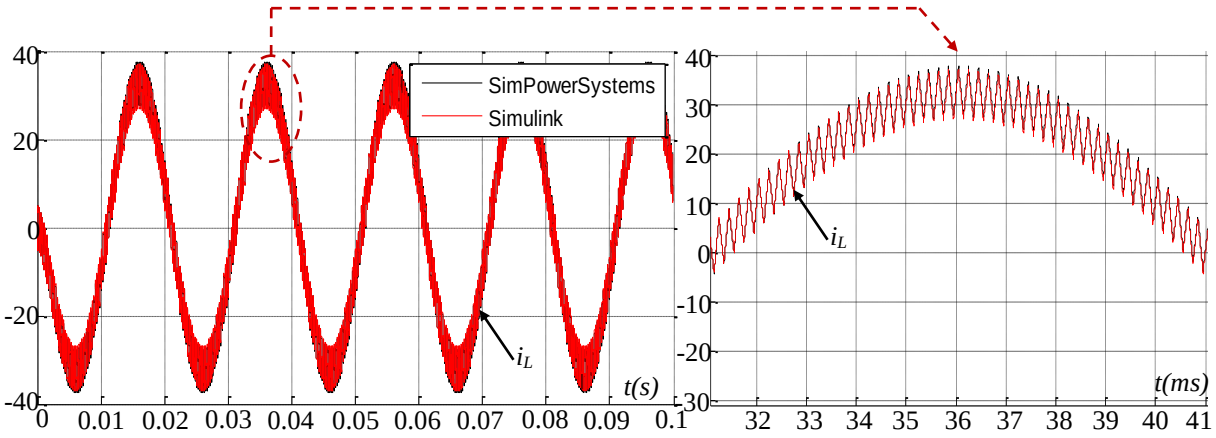


Figure 4. 4: AC load current waveform- open loop simulation

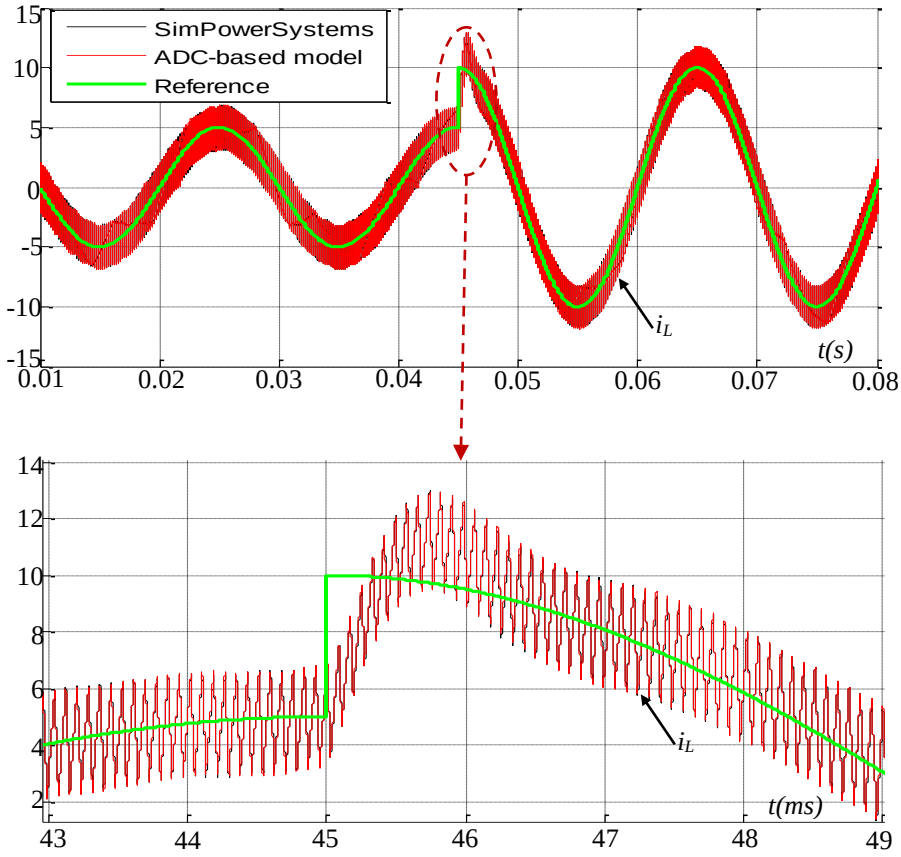


Figure 4. 5: AC load current waveform – closed loop simulation

2.3. FPGA implementation

a- FPGA architecture design

The FPGA-based architecture has been developed according to the adopted partitioning given in Figure 4.3. By this way, these modules are sequenced as follows. The processing of the states (ON/OFF) of switches is firstly achieved depending on the control signals C and its inverse provided by the controller and depending on the switch voltages $V_{T/Di(i=1...4)}[k\text{-}1]$ and currents $i_{T/Di(i=1...4)}[k\text{-}1]$. The dependent current sources $J_{Ti}[k]$ and $J_{Di}[k]$ are then computed. This is followed by making the necessary combinations to extract the vector $b[k]$ used to calculate the vector $x[k]$. Note that the matrix H^{-1} has been calculated offline since it remains constant whatever the switch states. Finally, the node voltages $v_0[k]$, $v_{Li(i=a,b)}[k]$ from this vector and the currents $J_{T/Di(i=1...4)}[k]$ are used to calculate the switch voltages $v_{T/Di(i=1...4)}[k]$ and currents $i_{T/Di(i=1...4)}[k]$.

As for the architecture of each module, the data-path of each module has been factorized and pipelined. The objective is always to find an optimum between the available hardware resources and the timing constraints with the preservation of the algorithm modularity. Figure 4.6 highlights a part of the designed FPGA architecture that corresponds to the computation of one element of the vector $x[k]$.

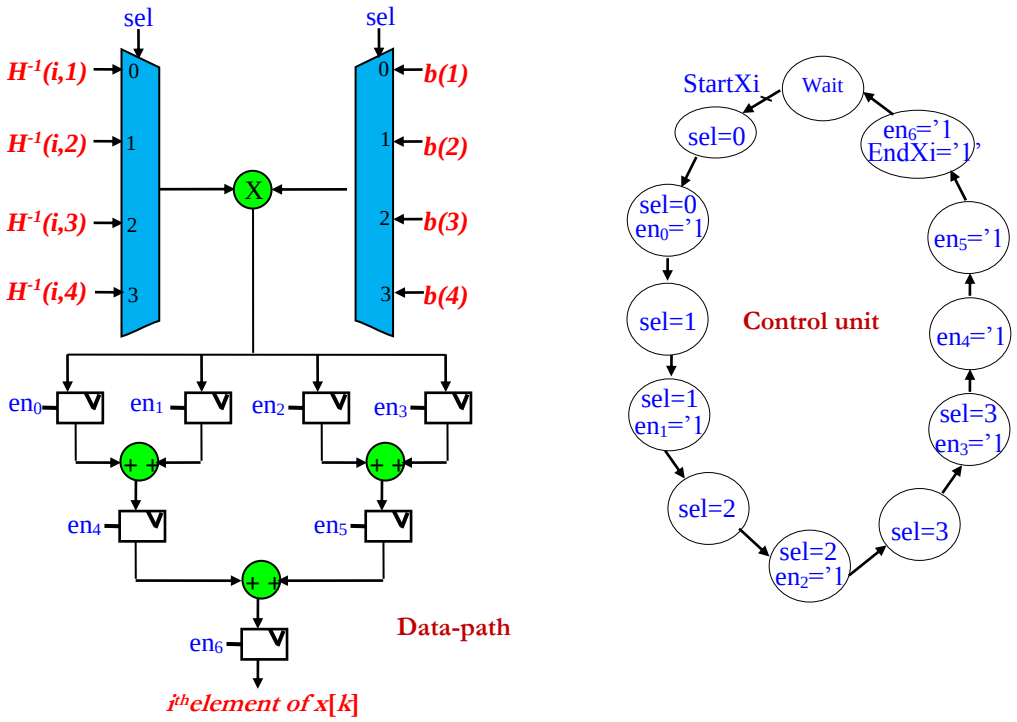


Figure 4. 6: Designed FPGA architecture of the i^{th} element of the vector $x[k]$

b- Coding & Functional validation

The developed VHDL architecture has been functionally tested in open-loop using ModelSim software tool. The obtained offline simulation results are then compared to those obtained with Matlab/SimPowerSystems.

The made open-loop test consists in applying a voltage reference to a PWM module (5 kHz switching frequency) that generates the control signals for the converter. Figure 4.7 presents the waveform of the AC load current. One can observe the same behavior of the Matlab/SimPowerSystems based model and the ModelSim model.

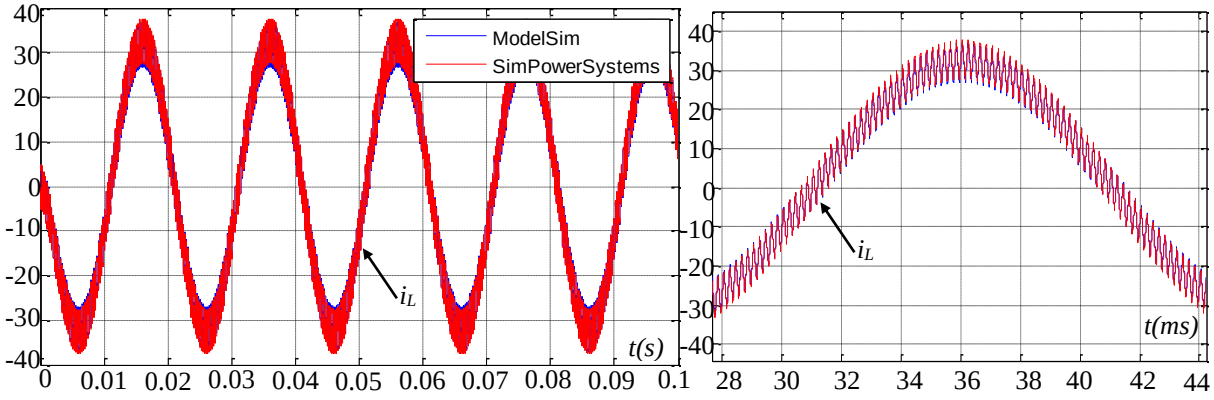


Figure 4. 7: AC Load current waveform- open loop offline simulation

c- Design/synthesis/place/route and Time/Area evaluation

Figure 4.8 shows an overview of the timing diagram. The IP modules of the DC-AC converter and the AC load are activated in parallel with two different rates. The first module is activated via StartDCAC signal every 500ns. The second one is turned via Start signal at each 1µs.

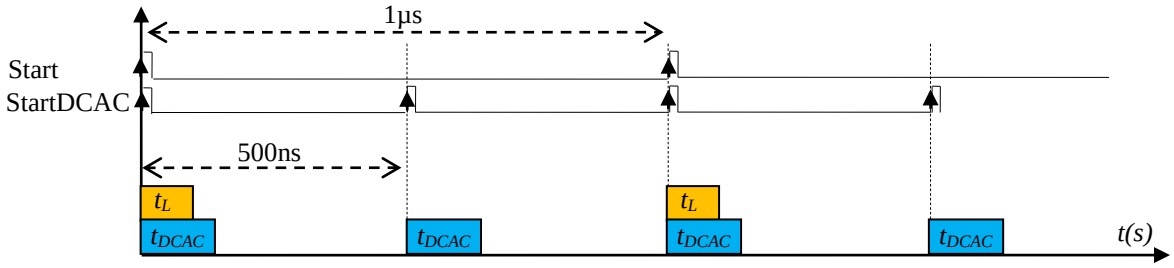


Figure 4. 8: Timing diagram of the developed simulator

With a 100MHz clock frequency, this computation time t_{DCAC} is equal to 380ns (latency equal to 38).

When using the Xilinx XC7Z020 Zynq device for the implementation of the DC-AC converter architecture, the resources consumption is evaluated at 8% of the component slices over 13300, 1% of Flip-Flops over 106400, 2% of LUTs over 53200, and 9% of DSP48E1s over 220.

The whole architecture including the P+Resonant controller and the embedded real-time simulator IPs of the DC-AC converter and the AC load uses 8% of the component slices over 13300, 4% of Flip-Flops over 106400, 4% of LUTs over 53200, and 20% of DSP48E1s over 220.

2.4. Experimentations

a- HIL tests

The objective here is to test and validate the functionalities of the developed single-phase DC-AC converter simulator IP when it is associated with its P+Resonant current controller. The real-time HIL validation is made using the Xilinx ChipScope analyzer.

Figure 4.9 illustrates the real-time HIL waveform of the P+Resonant-based AC load current. It can be seen that the load current follows properly its reference and has the same response to that obtained by SimPowerSystems. Therefore, the performance of the developed embedded real-time simulator IP is confirmed.

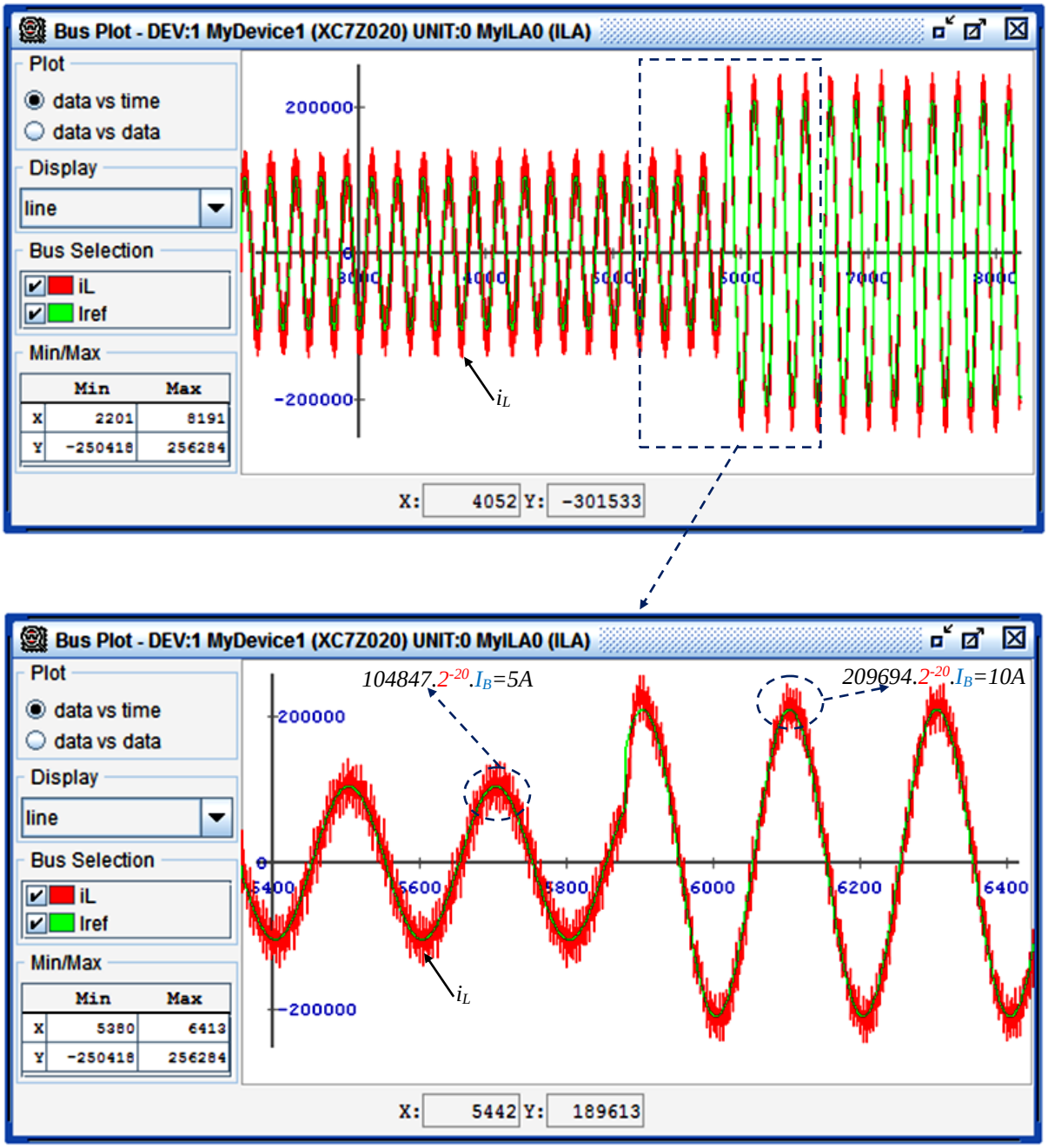


Figure 4. 9: P+Resonant based real-time HIL waveform AC load current

3. FPGA-based embedded real-time simulator IP of a 3-phase voltage source inverter

In this section the development of an ADC-based embedded real-time simulator IP of a 2-level 3-phase Voltage Source Inverter (VSI) is discussed. The latter is based on SKM 50GB123D IGBT/Diode switches that support 1200V/50A and can switch at 20 kHz maximum switching frequency.

As shown in Figure 4.10, to make a complete validation, this simulator is associated to the simulator IP of a 3-phase Y-connected load including a sinusoidal back-EMF (its parameters are given in Table 4.2) and all are applied in the context of HIL validation. To this purpose, a

PWM IP module is also used to generate the VSI control signals. All these IPs are implemented in the same Xilinx XC7Z020 Zynq device.

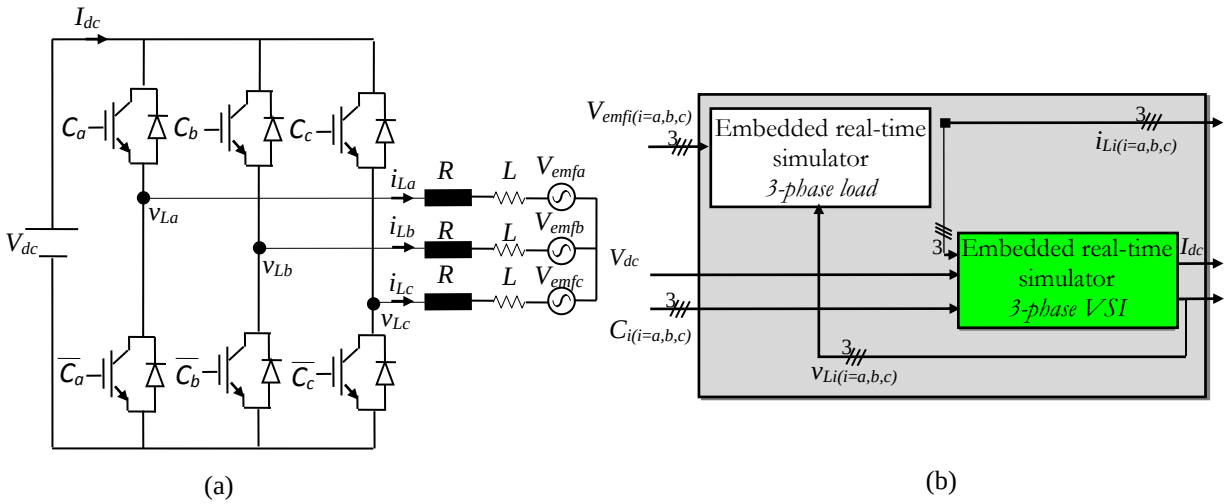


Figure 4. 10: Simulated power system (a) topology (b) structure

Table 4. 2: Circuit parameters of the power system under simulation

Circuit parameters	Value
Resistance (R)	50 Ω
Inductance (L)	100 mH
DC bus voltage (V_{dc})	220 V
Back EMF voltage (V_{emfi})	100 V_{RMS}
Back EMF frequency (Hz)	50 Hz

For the same previously discussed motivations, the VSI modeling has been achieved based on the ADC approach. Thus, to extract the overall electrical equations representing the VSI, its equivalent ADC circuit is firstly determined by replacing each switch by its equivalent ADC model (see Figure 4.11). Then, the relations between voltages and currents of the whole VSI are extracted from this circuit by applying the MNA method.

The same modular partitioning adopted for the single phase DC-AC converter has been maintained here. Although the number of equations of each module is much more important when compared to that of the single phase DC-AC converter model, the computation time remains the same. This is possible since the equations of each module are independent.

The corresponding electrical equations have been discretized using the Backward Euler method. To satisfy timing and algorithm constraints, the time-step and the fixed point format are maintained respectively at 500 ns and 24Q21 with 21 is the number of bits dedicated to the fractional part.

To validate the developed algorithm, an offline simulation test has been done and the results obtained with the ADC-based model and those obtained with the SimPowerSystems based reference model are compared.

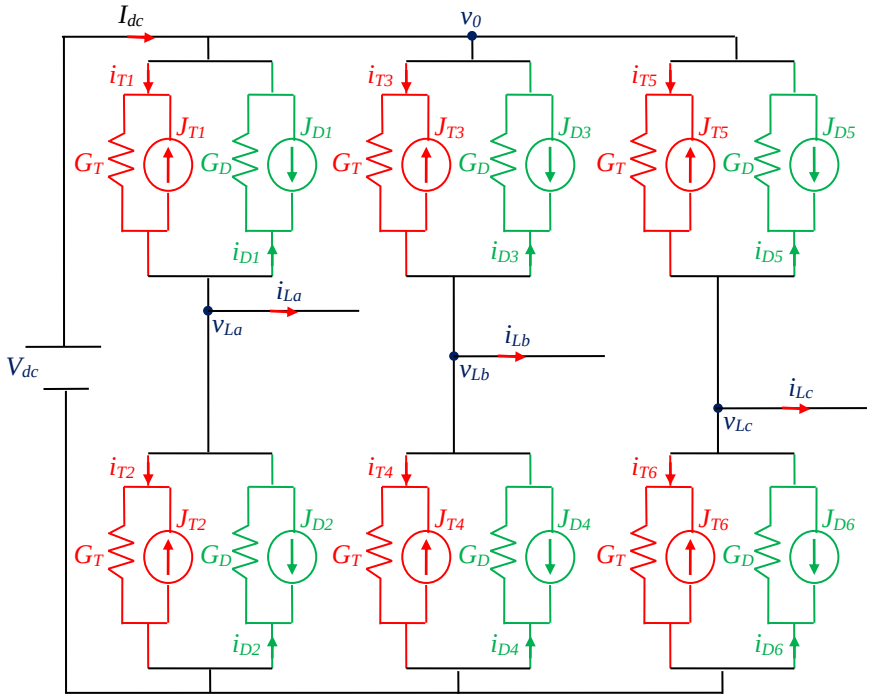


Figure 4. 11: Equivalent ADC-based model of the 3-phase voltage source inverter

Figure 4.12 and Figure 4.13 compares respectively the responses of the 3-phase load currents and the line-to-line voltages between phases calculated by the ADC-based model and the SimPowerSystems based model.

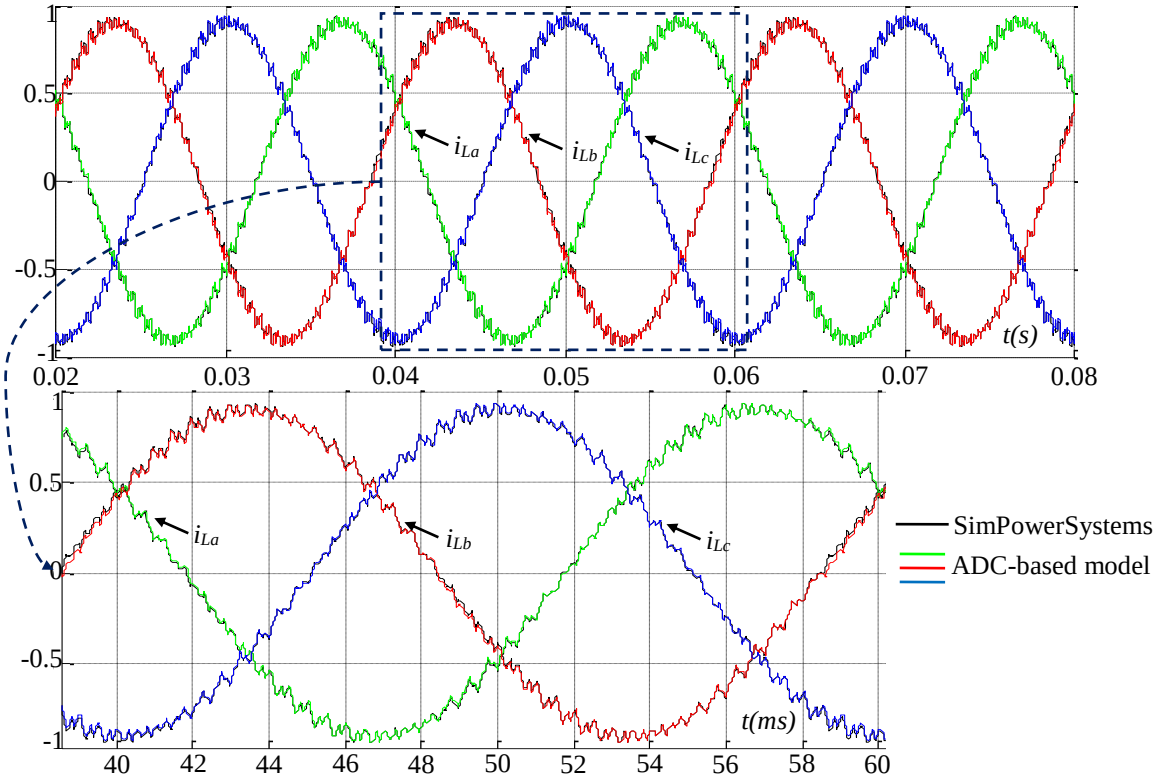


Figure 4. 12: 3-phase load currents of SimPowerSystems based model and ADC-based model

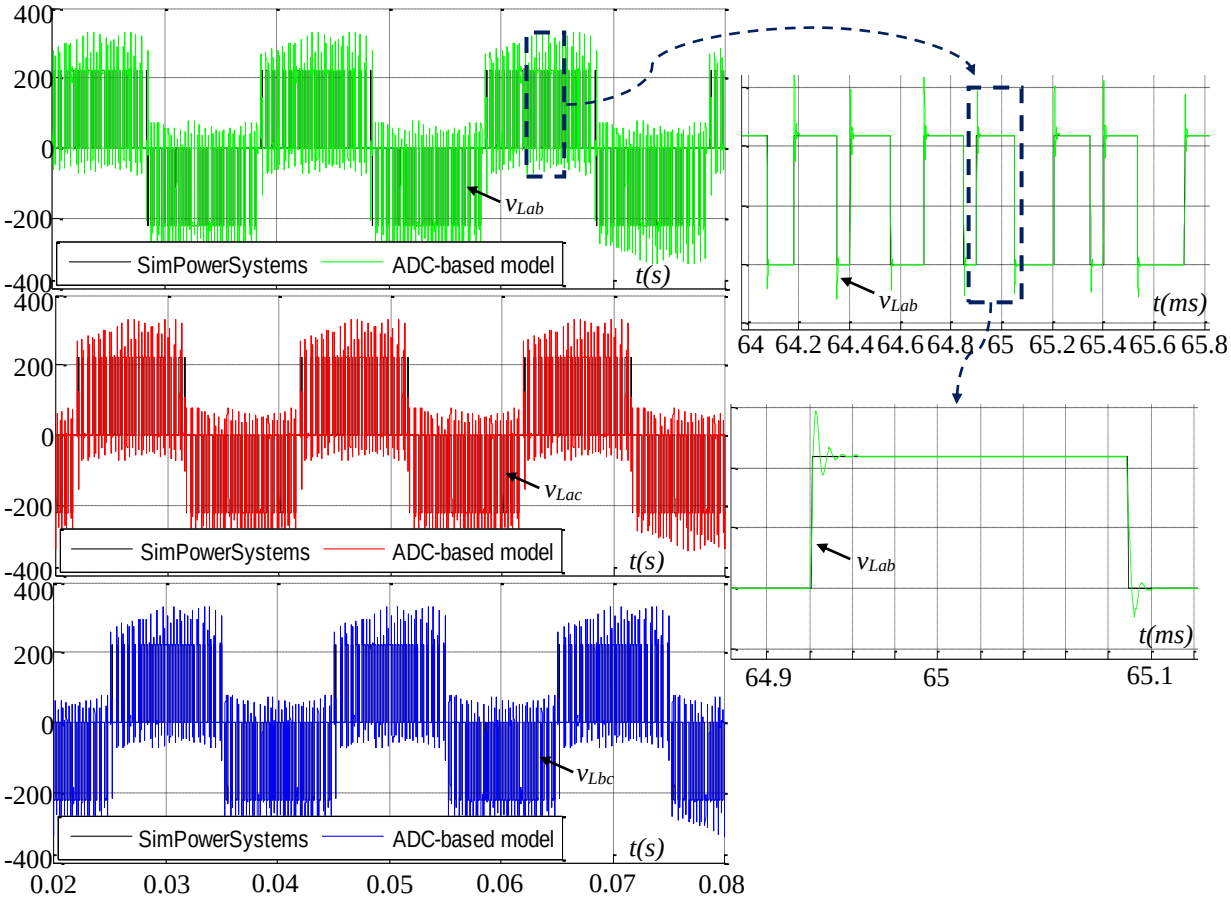


Figure 4. 13: Line-to-line voltages of SimPowerSystems based model and ADC-based model model

All these results are obtained for a PWM switching frequency equal to 2 kHz. As it can been seen, the obtained results show a close agreement between the ADC-based model and the SimPowerSystems based reference model. The developed algorithm is then validated and its corresponding FPGA-architecture has been developed, coded and implemented and its time/area performances have been evaluated.

Compared to the obtained time/area performances of the simulator IP of the single phase DC-AC converter, the computation time remains the same (380ns). As explained before, this is possible since the equations of each module are independent and have been executed in parallel. This is not the case for the consumed FPGA resources since the converter topology is much more complex and the number of implemented equations is much higher.

Table 4.3 shows the consumed FPGA resources for the embedded 3-phase VSI real-time simulator IP. The expected area/cost performances are then reached and the last step of the proposed design guidelines has been initiated.

Table 4. 3: Consumed FPGA resources for the embedded 3-phase VSI real-time simulator IP

LUTs	2927 out of 53200 (5 %)
Flip-Flops	4669 out of 106400 (4 %)
DSP48E1s	87 out of 220 (39 %)
Occupied slices	1374 out of 13300 (10%)

Figure 4.14 and Figure 4.15 present the HIL results of load currents i_{La}, i_{Lb} and i_{Lc} and line-to-line voltages v_{Lab}, v_{Lac} and v_{Lbc}. The same test conditions as during offline simulations are maintained. The obtained offline and real-time results are similar. The developed simulator IP is then validated and added to the IP-Library.

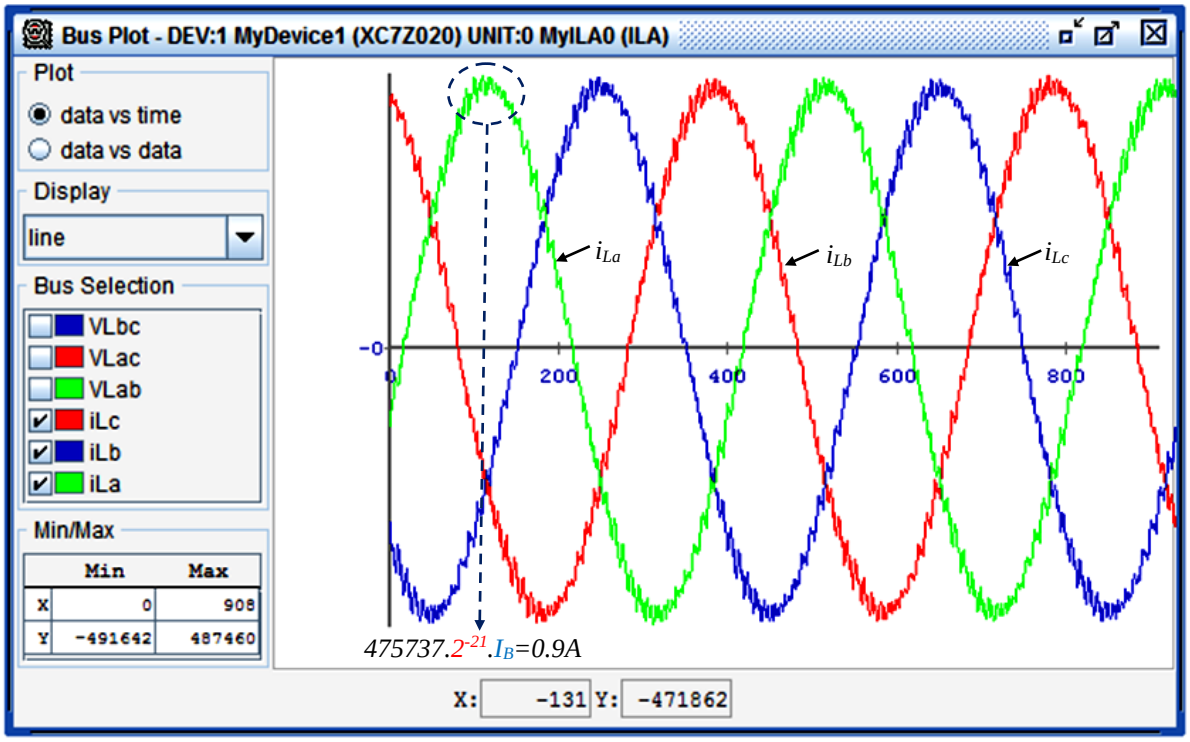


Figure 4. 14: Real-time waveform of the load currents

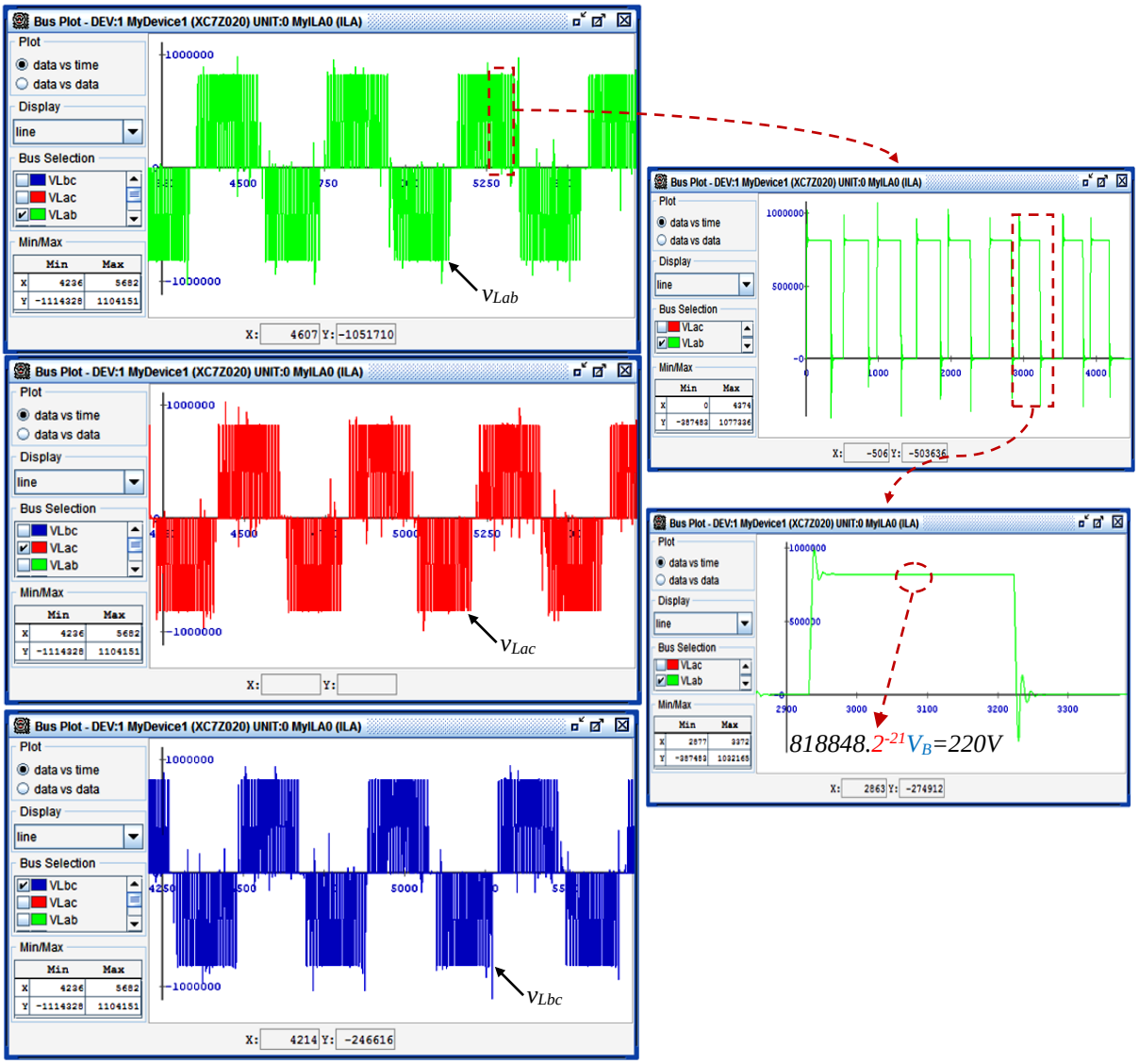


Figure 4. 15: Real-time waveforms of the line-to-line voltages

4. FPGA-based embedded real-time simulator IP of a 3-phase diode rectifier

In this section the development of an embedded real-time simulator IP of 3-phase Diode Rectifier (DR) is dealt with. To make a complete validation, this simulator IP has been associated with the simulator IP of a 3-phase grid and the one of a resistive load. The parameters of these additional IPs are given in Table 4.4.

Figure 4.16.a and Figure 4.16.b show the structure of the simulated power converter and its equivalent ADC-based circuit. Here again, the converter equations are extracted from this equivalent ADC-based circuit by applying the MNA method and partitioned into four independent and reusable modules with the same way of modular partitioning adopted in the previous sections (section 2 and 3).

As for the digital realization, the discretization is based on Backward Euler approximation. The chosen time-step and the fixed-point data format have been set respectively to 500ns and 24Q21.

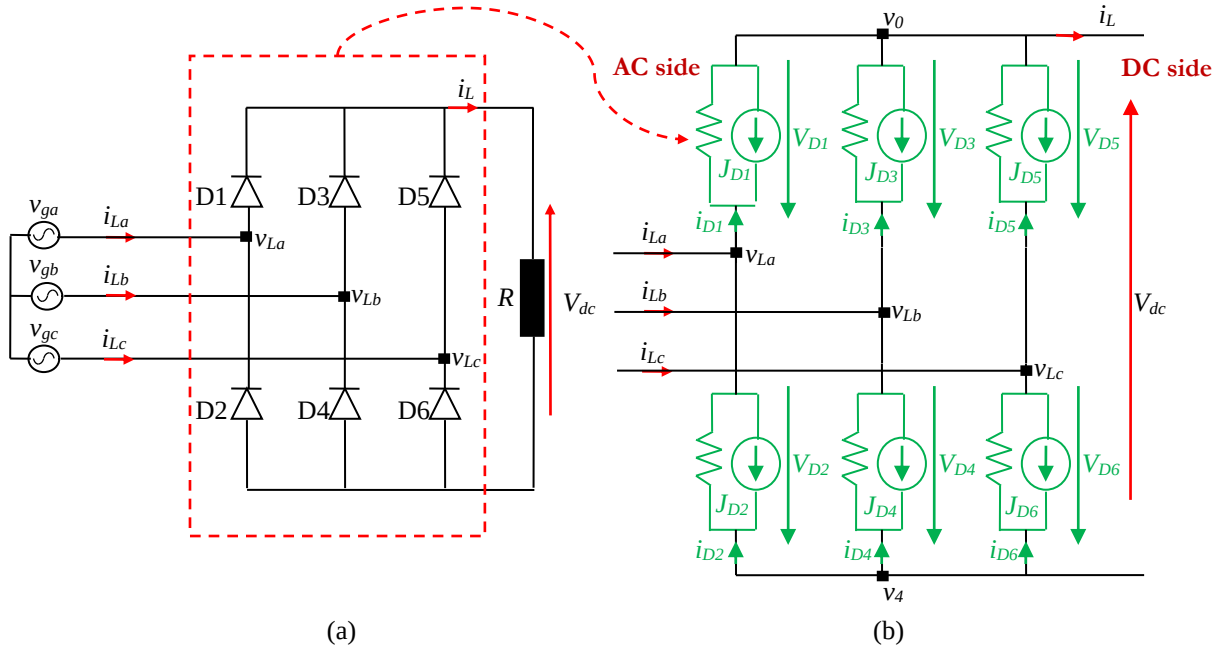


Figure 4. 16: (a) 3-phase diode rectifier topology (b) equivalent ADC-based model

Table 4. 4 : Circuit parameters of the 3-phase grid and the resistive load

Circuit parameters	Value
Load resistance (R)	22 Ω
Source voltage (v_{gi})	220 V_{RMS}
Frequency	50 Hz

To validate the developed digital realization, the obtained fixed-point simulation results are compared with those of the SimPowerSystems based model. As shown Figures 4.17, 4.18 and 4.19, the compared results are similar.

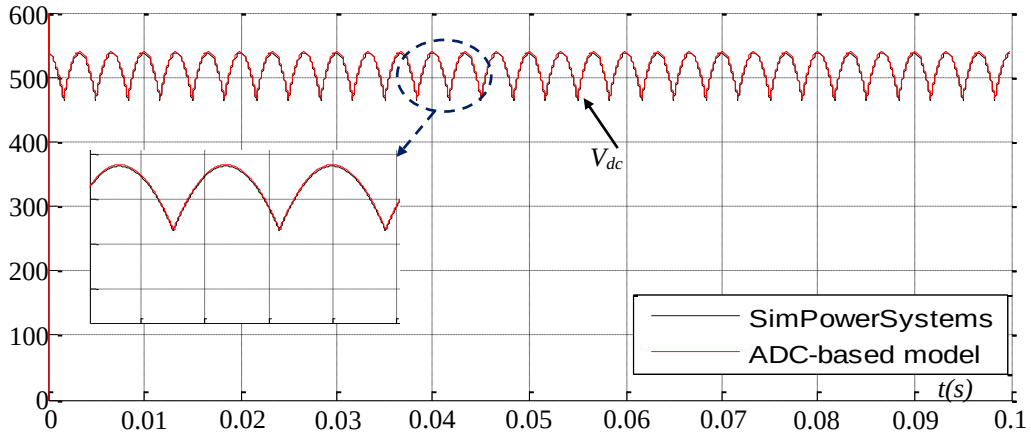


Figure 4. 17 : Diode rectifier DC-link voltage waveform - offline simulation

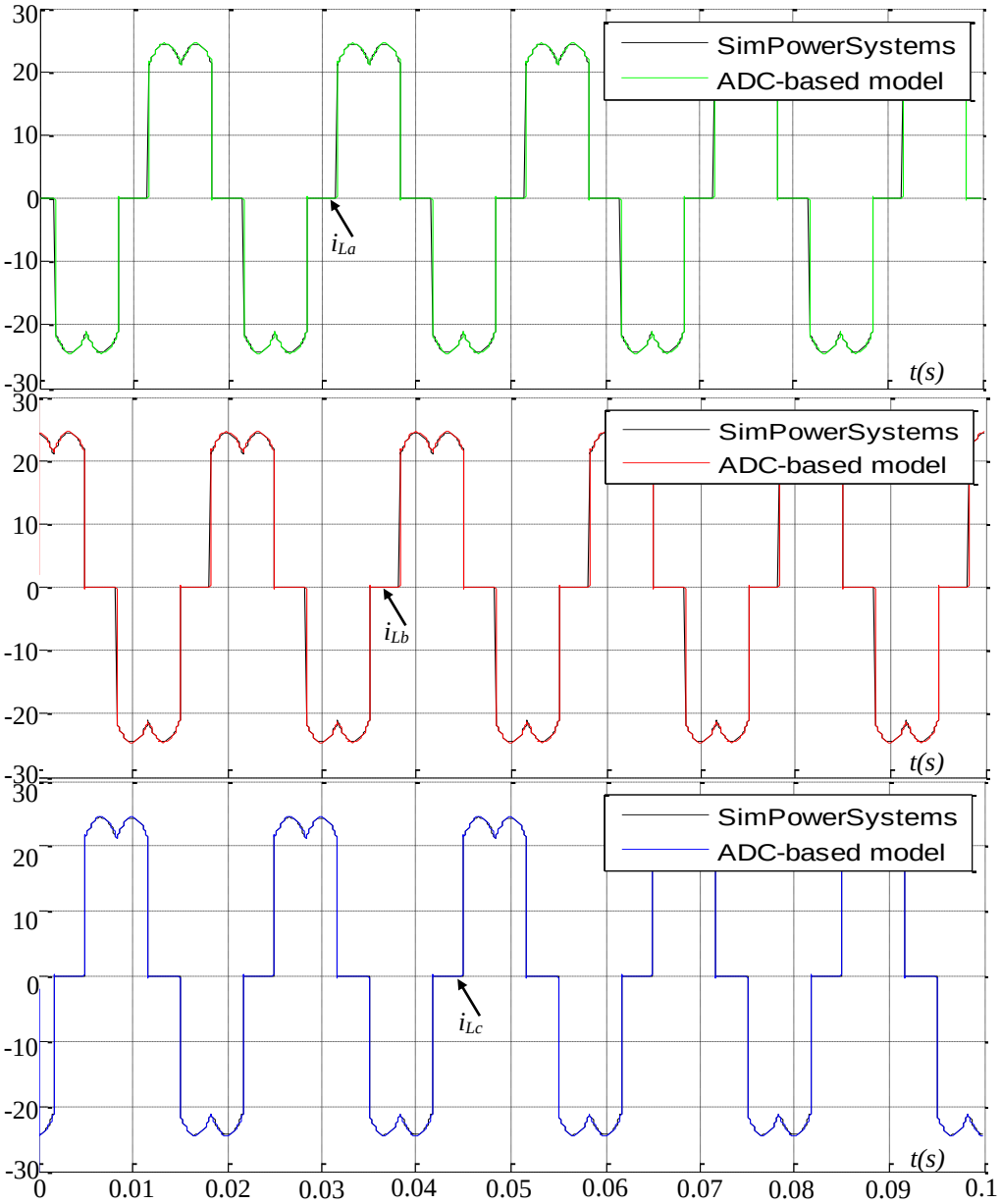


Figure 4. 18: Diode rectifier line currents waveforms - offline simulation

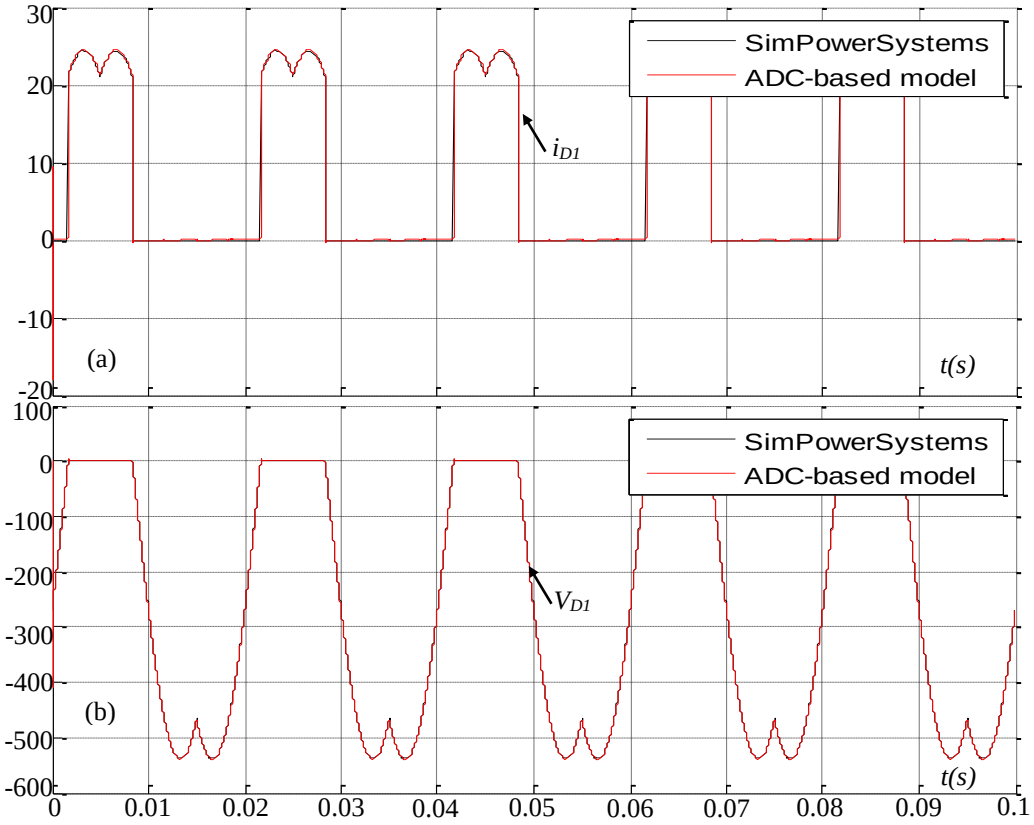


Figure 4. 19: (a) current of diode 1 (b) voltage of diode 1 - offline simulation

When it comes to the FPGA implementation, an FPGA-based hardware architecture that is dedicated to the developed digital algorithm has been designed and implemented on the low-cost Xilinx XC7Z020 Zynq device.

Table 4.5 summarizes the resources utilization of the developed embedded three-phase diode rectifier real-time simulator. As a recall, with the adopted modular partitioning, the computation time is independent of the converter topology and it equals here to 380ns.

Table 4. 5: Consumed FPGA resources for the embedded real-time 3-phase diode rectifier simulator

LUTs	2896 out of 53200 (5 %)
Flip-Flops	4634 out of 106400 (4 %)
DSP48A1s	39 out of 220 (17 %)
Occupied slices	1339 out of 13300 (10 %)

The following figures present the obtained HIL results. Figure 4.20 shows the real-time waveforms of the diode rectifier line currents. Figure 4.21 gives the real-time waveforms of the diode rectifier DC-link voltage and current and voltage of the diode D1.

As it can be seen from these figures, with the same operating conditions as during Matlab/Simulink offline simulations, the same behavior has been obtained.

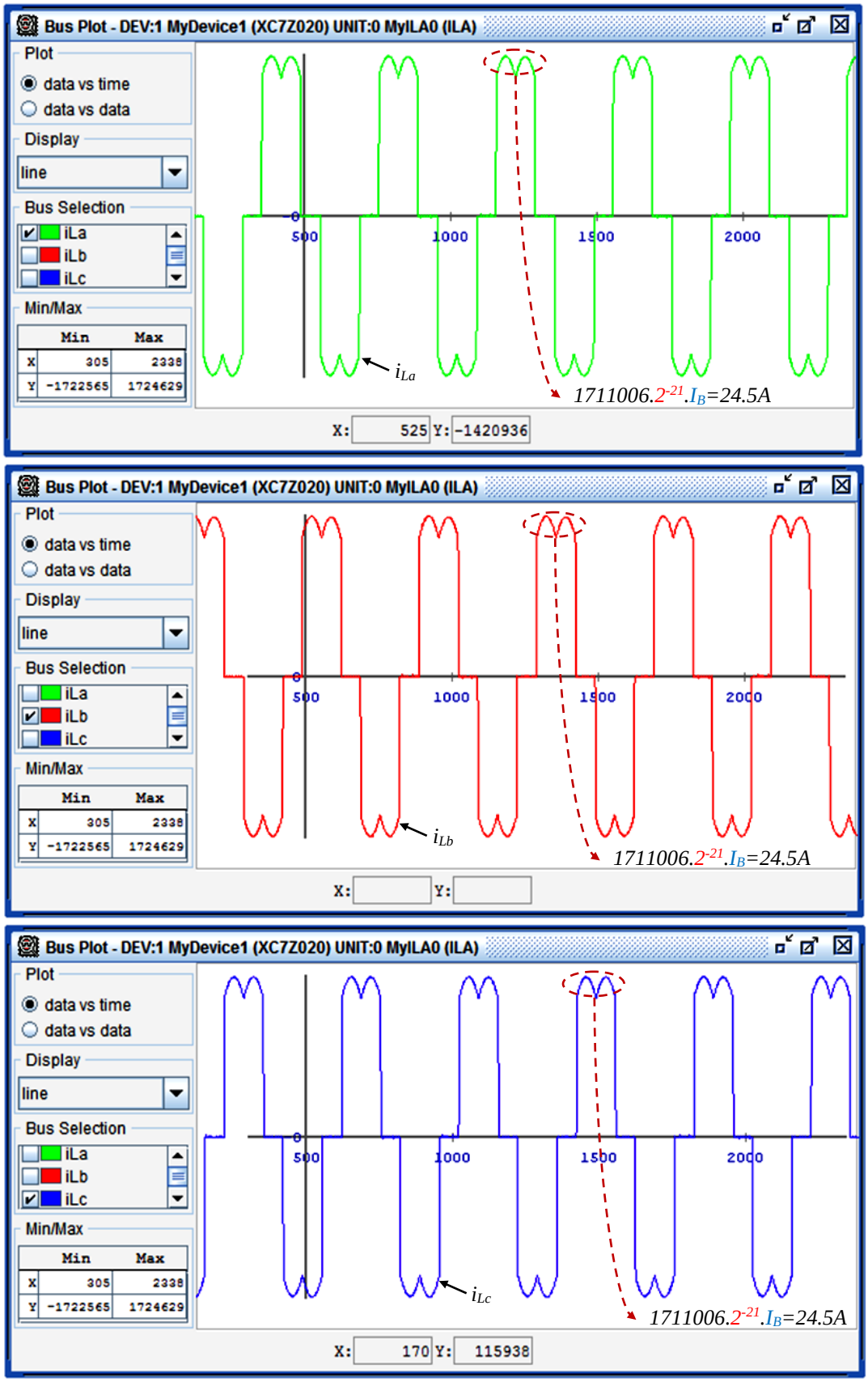


Figure 4. 20: Diode rectifier line currents waveforms - real-time simulation

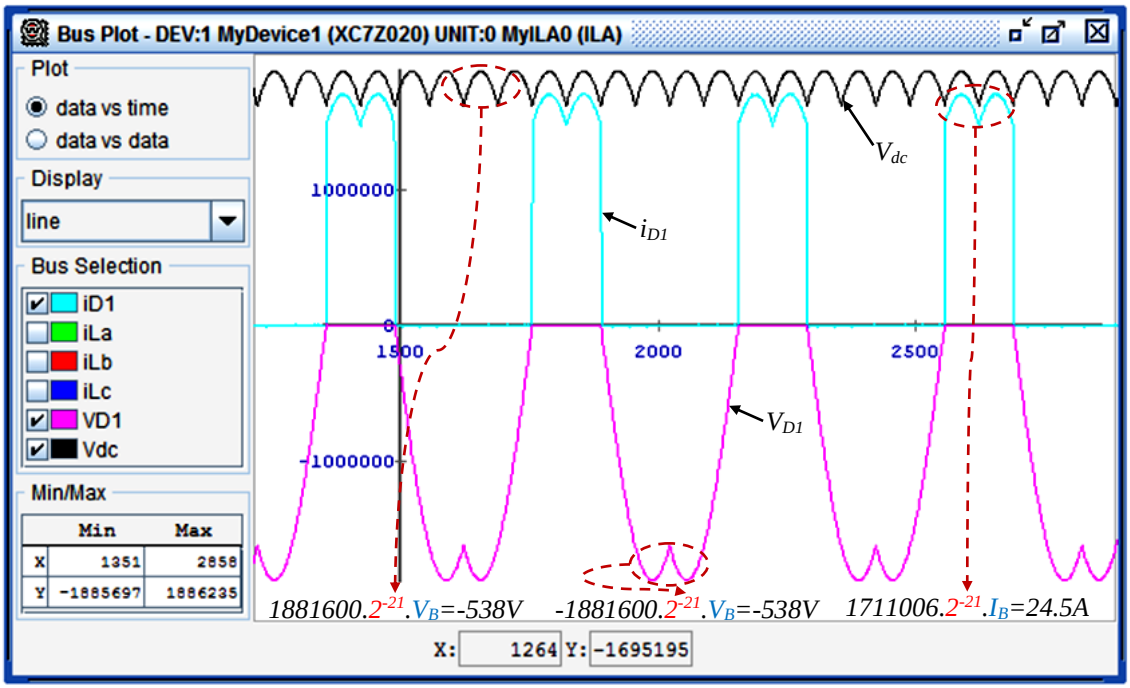


Figure 4. 21: Diode rectifier DC-link voltage and current and voltage of diode D1 - real-time simulation

5. Conclusion

In this chapter the FPGA implementation of embedded real-time simulator IPs of switching elements was discussed. Three power converter topologies have been studied: the embedded real-time simulator of a single phase DC-AC converter, the one of a 3-phase voltage source inverter and finally the one of a 3-phase diode rectifier have been all implemented and added to the IP-library.

The developed real-time simulators have been validated in the context of HIL testing. For each developed simulator, real-time simulation results have been provided and compared with those of offline simulations. The obtained real-time simulation results demonstrate clearly the effectiveness of the developed simulator IPs in terms of accuracy and their ability to be embedded within low-cost FPGA-based controllers using moderate consumed resources and small time-steps.

In the next chapter a complete embedded control application will be presented. Thus, embedded real-time simulator IPs of PWM rectifier (switching element) and RL filter (electromagnetic element) will be developed and applied in the context of fault-tolerant control of a grid-connected Voltage Source Rectifier (VSR). An experimental validation will be also provided.

Chapter 5

Embedded Real-Time Simulator IPs of PWM Rectifier and 3-phase RL-Filter

Application to a Fault-Tolerant Control of a Grid-Connected Voltage Source Rectifier

1. Introduction

In this chapter, author proposes a complete application where both electromagnetic and switching simulator IPs are deployed. They are applied in the context of fault-tolerant control of grid-connected voltage source rectifier. These simulators are the one of 3-phase 2-level PWM rectifier and the one of a 3-phase RL-filter. They are both embedded within an FPGA-based PWM rectifier controller and used as a grid current estimator. When a fault appears on the grid current sensors, this estimator is inserted in the control closed-loop and the measurements are then replaced by their estimates. The ability of this estimator to improve the system reliability by maintaining the service continuity in the case of faults has been validated through HIL tests and experiments.

The content of this chapter is organized according to the proposed design guidelines, giving at the end the FPGA time/area analysis, the HIL and the experimental results.

2. Preliminary system specification

Figure 5.1 presents the developed control system. The power stage is composed of: *(i)* 3-phase voltage sources from the grid (230V,50Hz), *(ii)* an auto-transformer, *(iii)* a 3-phase RL-filter (R=0.8Ω, L=20mH), *(iv)* a 20 kVA 3-phase 2-level PWM rectifier composed of IGBT/Diode switches, *(v)* a capacitor for the DC-link (1100µF/800V), *(vi)* a resistive load (100Ω/2.5A) and *(vii)* a contactor to connect and disconnect this load.

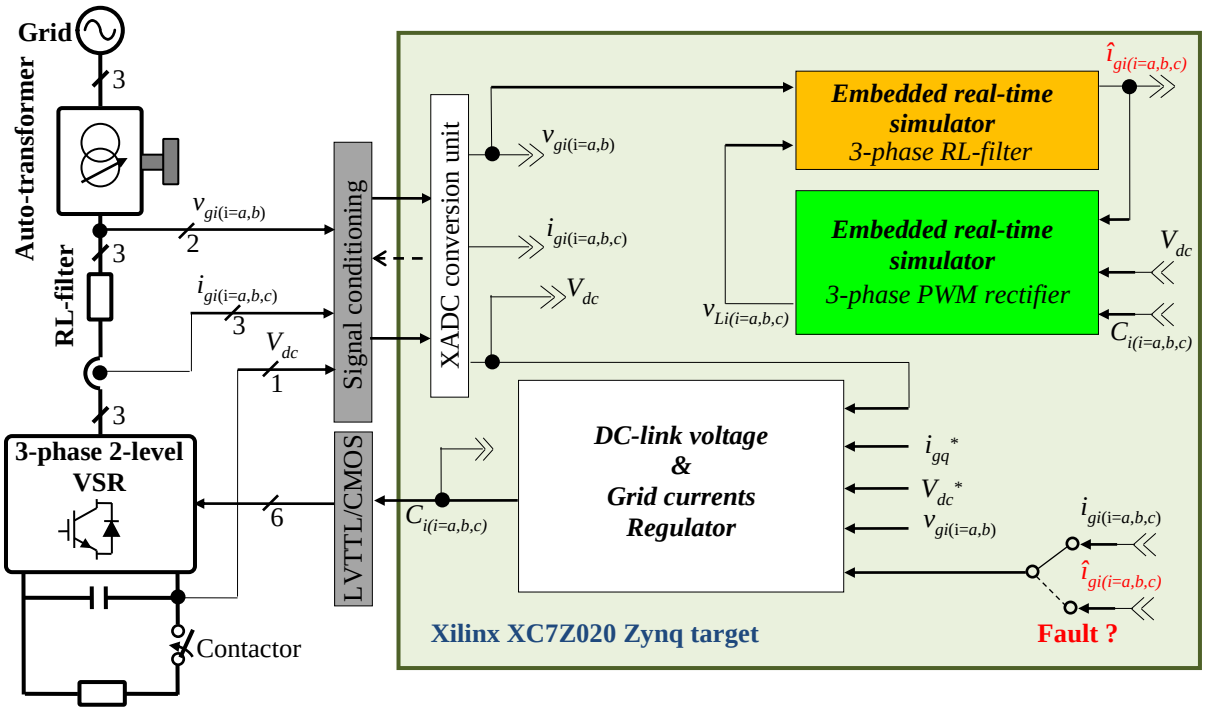


Figure 5. 1: Structure of the developed control system

The used device is the Xilinx FPGA Zynq SoC (XC7Z020 ZedBoard). Indeed, such low cost SoC devices are certainly one of the most appropriate technological solutions for current embedded applications. As a recall, it combines the software flexibility ensured by the integrated processors (a dual-core Cortex-A9 ARM processor) and hardware performances brought by the FPGA logic blocks (53200 LUTs and 106400 Flip-Flops). This is of course in

addition to other resources like 560 KB RAM blocks, 220 DSP48E1, analog peripherals and up to 200 high speed I/O blocks.

As shown in Figure 5.1 the following modules are implemented: *(i)* the analog to digital conversion module which is based on the integrated XADC, *(ii)* a DC-link voltage and grid currents regulator, *(iii)* the ADC-based embedded real-time simulator module of the PWM rectifier and *(iv)* the embedded real-time simulator of the 3-phase RL-filter.

Due to their fast dynamics which need very short time-steps, all these modules have been implemented fully in hardware. This approach leaves space for implementing advanced control technique like model predictive control within one of the processor cores, but it is beyond the scope of this work.

Since the study will be focusing on the embedded real-time simulator of the PWM rectifier and the one of the 3-phase RL-filter, author supposes that the other modules have already been developed and extracted from the IP-Library. They are summarized in the following.

2.1. DC-link voltage and grid currents regulator

The chosen control strategy is the Direct Sliding Mode Power Control (DSMPC), [JIA11]. Its principle is summarized in Figure 5.2.

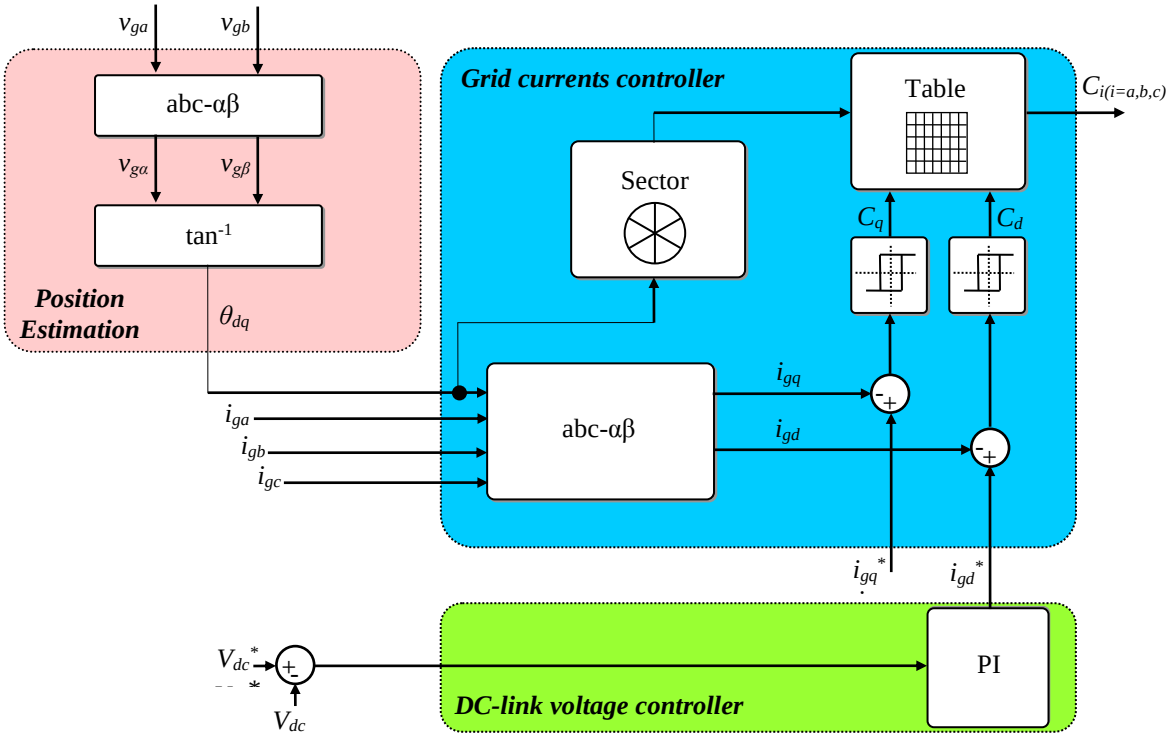


Figure 5. 2: Synoptic of the DSMP controller

The main objective is to keep the DC-link voltage V_{dc} equal to its reference V_{dc}^*, with sinusoidal current absorption and controlled power factor. To do so, two control loops are implemented. The outer loop is based on PI regulator for the DC-link voltage. The inner loop controls the grid currents in the dq synchronous reference frame. As shown in Figure 5.2 the position θ_{dq} of the grid voltage vector is computed after an *abc-αβ* coordinate transformation and a tan^{-1} trigonometric function. It is then used to compute the grid voltage sector and the i_{gd} and i_{gq} currents (*abc-dq*). Two hysteresis controllers are used to compute the logical signals

C_d and C_q according to the current error on d-axis (resp. q-axis). Finally, a switching table allows the application of the control signals $C_{i(i=a,b,c)}$.

The discretization of the PI regulator is achieved with Tustin method and the sampling period is set to 50µs. The used base values to normalize the variables are 563V for voltages and 4A for currents. These values are chosen by considering the nominal voltage/current values and also the gains introduced by the sensors and the XADC conversion unit.

As for data quantification, the fixed-point format is set to 20Q12 (20 total bit number and 12 bits in the fractional part).

The obtained total latency of the controller is equal to 42. The corresponding computation time is then equal to 420ns (with a 100MHz system clock).

As for hardware resources occupation, the controller uses 7 (3.18%) DSP48E1 units (for multiplications), 1336 (2.51%) LUTs and 1034(0.97%) Flip-Flops. Additional details about these time/area performances are given later on in Table 5.1.

2.2. XADC conversion unit

The analog to digital conversion is achieved using the on-chip XADC block, [XIL]. The latter includes a dual 12-bit, 1MSPS ADC which has been configured to operate in simultaneous sampling mode. To ensure the conversion of all signals ($i_{gi(i=a,b,c)}$, $v_{gi(i=a,b)}$ and V_{dc}), an off-chip analog multiplexer has been used. Figure 5.3 highlights the synoptic of the implemented XADC conversion module associated to its corresponding process.

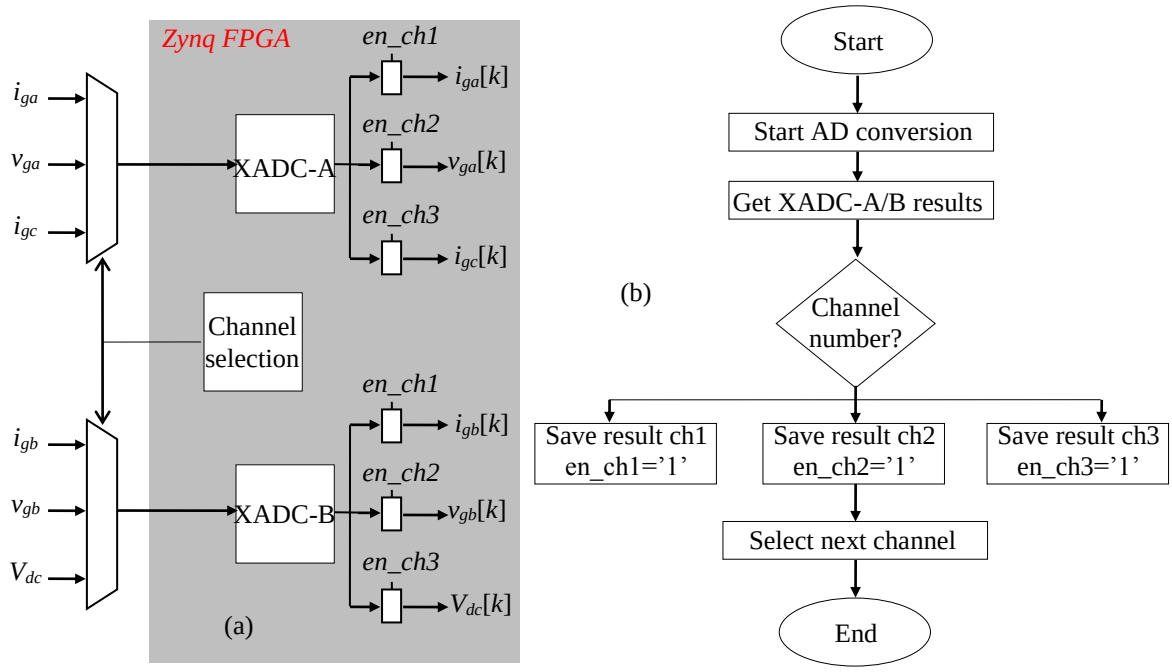


Figure 5. 3: (a) Synoptic of the XADC conversion unit; (b) conversion process

The experimentally measured conversion time is equal to 1.18µs. To take into account the settling time of the external analog multiplexer, the sampling period of this XADC conversion unit has been set to 2µs. The used hardware resources to control the XADC and to rescale the data are: 39 (0.07%) LUTs and 197(0.18%) Flip-Flops.

3. Algorithm development

Always with respect to the given design guidelines proposed in chapter 2, this section deals with the development and the validation of the algorithm of the embedded PWM rectifier real-time simulator and the one of 3-phase RL-filter simulator.

3.1. Algorithm of the PWM rectifier simulator

As a reminder, this simulator computes the line voltages $v_{Li(i=a,b,c)}$ from the estimated currents $\hat{\imath}_{gi(i=a,b,c)}$, the measured voltage V_{dc} and according to the control signals $C_{i(i=a,b,c)}$(see Figure 5.1).

The adopted modeling approach is based on the ADC equivalent model. Figure 5.4-(a,b) shows the obtained equivalent ADC-based circuit for the studied power converter. The same modular partitioning proposed in chapter 4 is adopted here. Figure 5.4-c shows the modules that have been defined for the ADC-based model and that are located in level 2 and level 3 of the IP-library. Each module is subdivided into sub-modules from the lower levels.

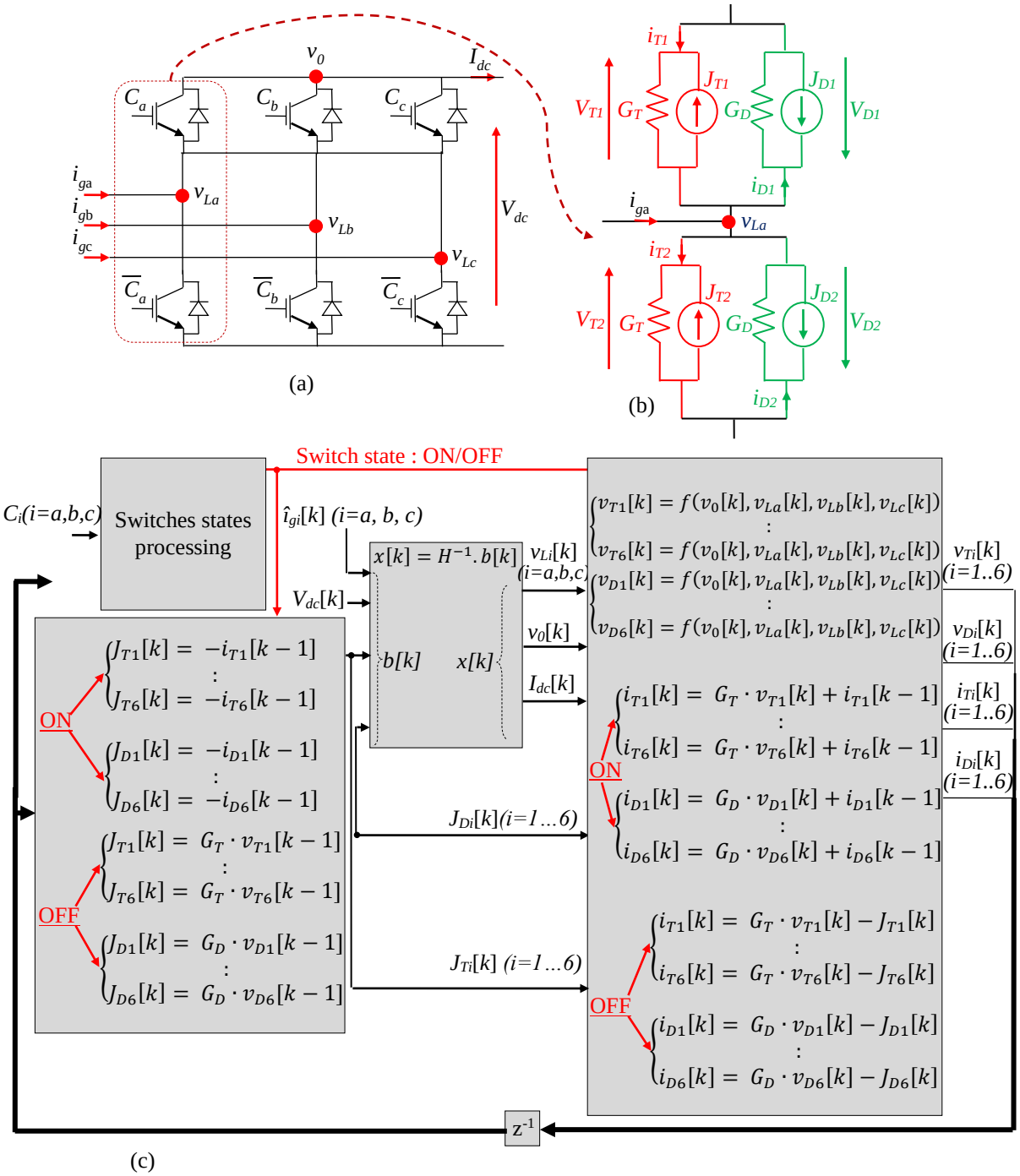
To satisfy the timing constraint, the simulation time-step has been set to 500 ns. Indeed, the maximum switching frequency of the used power converter is equal to 20 kHz. Thus, the simulation time-step is therefore 100 times less than the corresponding minimum switching period and inter-simulation time-step errors are then neglected.

The R_{sw}, L_{sw} and C_{sw} parameters have been tuned manually in this work and set to 7.5Ω, 160µH and 1.6nF. As a perspective, analytical tuning methods (eg.[SON14]) will be investigated.

As for the discretization, the derivative terms of the L-circuit and RC-circuit of the ADC model have been discretized using the Backward Euler method. As recall, this method is privileged since it contributes to reduce numerical oscillations during switching events.

Regarding the normalization, here again, the same base values for normalization as those of the controller are used (563V, 4A).

Finally, the chosen fixed-point format for these normalized data has been set to 32Q28 (32 total bit number and 28 bits in the fractional part). Four bits are attributed to the integer part to avoid overflow.



$$x[k] = [v_0[k] \quad v_{La}[k] \quad v_{Lb}[k] \quad v_{Lc}[k] \quad I_{dc}[k]]^T = H^{-1}.\, b[k]$$

$$= \begin{bmatrix} 3(G_T+G_D) & -G_T-G_D & -G_T-G_D & -G_T-G_D & 1 \\ -G_T-G_D & 2(G_T+G_D) & 0 & 0 & 0 \\ -G_T-G_D & 0 & 2(G_T+G_D) & 0 & 0 \\ -G_T-G_D & 0 & 0 & 2(G_T+G_D) & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}^{-1} . \begin{bmatrix} J_{T1}[k]-J_{D1}[k]+J_{T3}[k]-J_{D3}[k]+J_{T5}[k]-J_{D5}[k] \\ -J_{T1}[k]+J_{D1}[k]+J_{T2}[k]-J_{D2}[k]+i_{ga}[k] \\ -J_{T3}[k]+J_{D3}[k]+J_{T4}[k]-J_{D4}[k]+i_{gb}[k] \\ -J_{T5}[k]+J_{D5}[k]+J_{T6}[k]-J_{D6}[k]+i_{gc}[k] \\ V_{dc}[k] \end{bmatrix} \quad (5.1)$$

Figure 5. 4: (a) power converter topology; (b) 1-leg equivalent ADC-based circuit; (c) synoptic of the ADC-based model

3.2. Algorithm of the 3-phase RL-filter simulator

As shown in Figure 5.1 this embedded simulator computes the grid currents $i_{gi(i=a,b,c)}$ from the measured grid voltages $v_{gi(i=a,b,c)}$ and the line voltages $v_{Li(i=a,b,c)}$ processed by the PWM rectifier simulator. The corresponding discrete-time equations are obtained using a Backward Euler approximation and are expressed as follows:

$$\begin{cases} i_{ga}[k] = a_1.(\,v_{ga}[k] - v_{La}[k]) + a_2.i_{ga}[k-1] \\ i_{gb}[k] = a_1.(\,v_{gb}[k] - v_{Lb}[k]) + a_2.i_{gb}[k-1] \\ i_{gc}[k] = a_1.(\,v_{gc}[k] - v_{Lc}[k]) + a_2.i_{gc}[k-1] \end{cases} \quad (5.2)$$

Where, $a_1 = \frac{\Delta t}{\Delta t.R+L}$ and $a_2 = \frac{L}{\Delta t.R+L}$.

This simulator is synchronized with the PWM rectifier one and have the same simulation time-step, Δt = 500ns. The same base values for normalization as those of the controller and PWM rectifier simulator are used here (563V, 4A). Also, the same fixed-point format as the PWM rectifier simulator is chosen (32Q28).

To validate these developed algorithms, discrete-time and fixed-point offline simulations have been made with the help of Matlab/Simulink tools. To start with, Figure 5.5 presents the obtained waveforms during ON/OFF and OFF/ON commutations of the switches voltages and currents.

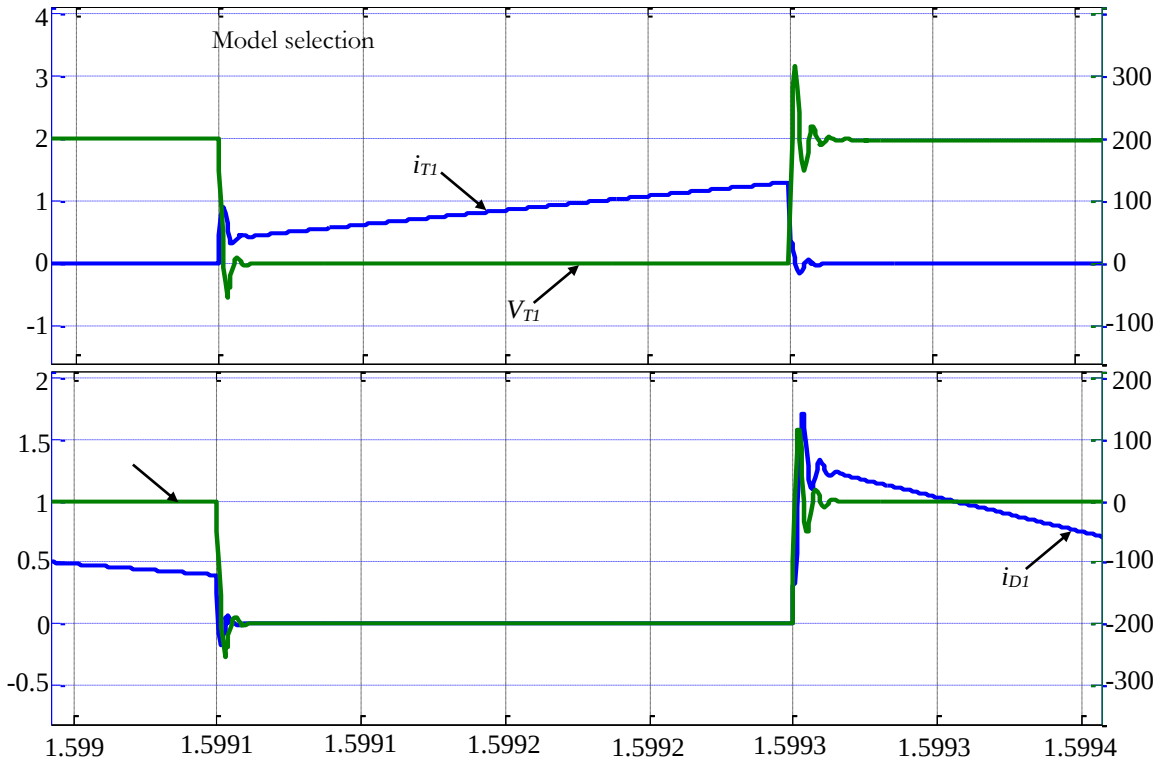


Figure 5. 5: Offline simulation results during switches commutation

(h: 50µs/div; v: 100V/div, 1A/div for IGBT, 0.5A/div for diode)

Figure 5.6 presents the obtained waveforms after having applied the developed embedded real-time simulators in the simulation model of the whole control system (Figure 5.1). It highlights the waveform, before and after load connection, of V_{dc}, v_{ga}, measured i_{ga} and

estimated $\hat{\imath}_{ga}$ (which are superposed). It can be seen that, when injecting these estimated grid currents to the controller, the load disturbance is correctly compensated and V_{dc} remains equal to its reference (set here to 200V). Also, unit power factor operation is obtained (i_{gq}^{*} reference being set to 0A), since the grid current and voltage remain in phase.

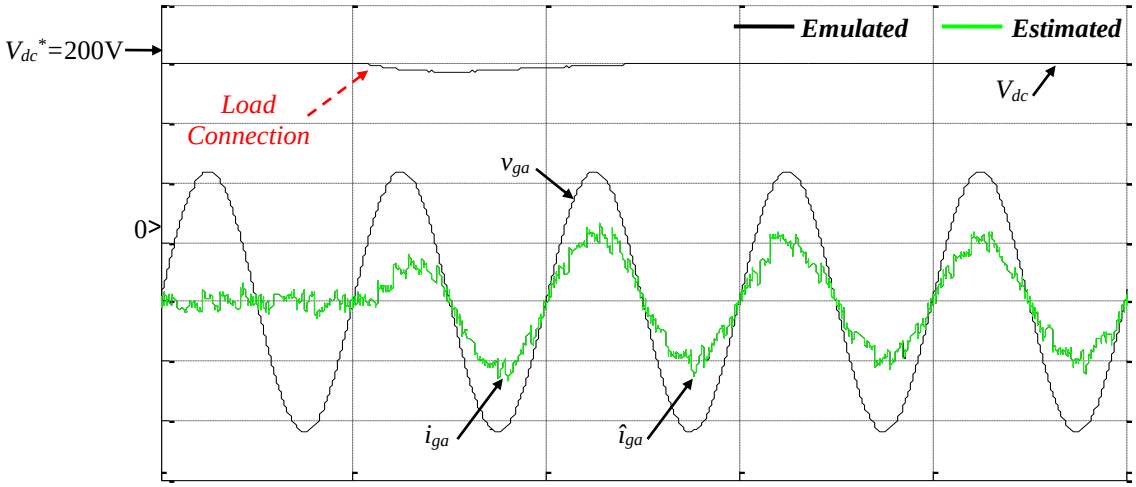


Figure 5. 6: Closed loop offline simulation results during load connection (h: 20ms/div; v: 50V/div, 2.5A/div)

4. FPGA implementation

4.1. FPGA architecture design

a- Architecture of the PWM rectifier simulator

Due to the request of very short simulation time-step (500ns), there is no other option than implementing the ADC-based embedded real-time simulator IP fully in hardware and then designing a fully dedicated architecture. The challenging issue is to find the best compromise between the hardware resources available for this IP and the corresponding timing performances (latency and system clock frequency). To do so, practical rules are given in the chapter 2 (FPGA implementation constraints).

Thus for this simulator IP, to optimize the timing performances, each of the modules of the designed FPGA architecture is synchronized with a control unit. Also, to reduce the propagation delays inside the FPGA target and then achieve a high operating clock frequency, the architecture is fully pipelined by inserting registers between successive arithmetic operators. Besides, to optimize the use of the hardwired DSP48E1 units, each module has been factorized according to the previously discussed A^3 methodology. The level of factorization must be carefully defined because it does not only depend on the availability of the hardware resources but also on the latency which impacts the total computation time.

Figure 5.7 highlights a part of the designed FPGA architecture that corresponds to the computation of one element of the vector $x[k]$ (according to equation 5.1). It shows the data-path and its control unit, both being duplicated 5 times so as to compute the whole elements of this vector.

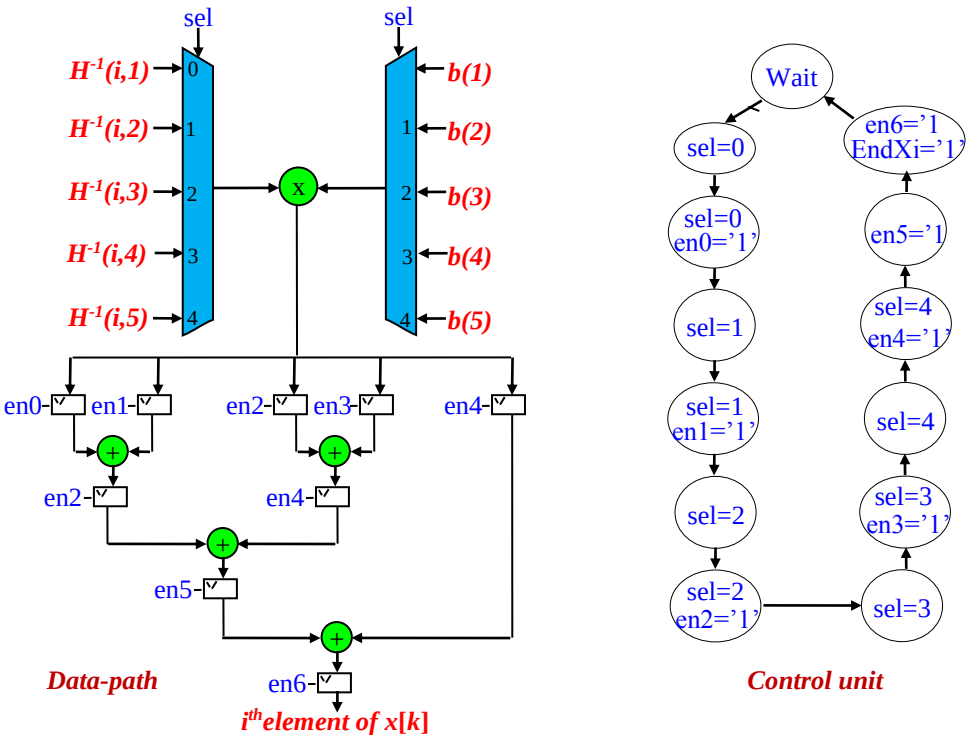


Figure 5. 7: Designed FPGA architecture of the ith element of the vector x[k]

b- Architecture of the 3-phase RL-filter simulator

Figure 5.8 presents the FPGA architecture of the 3-phase RL-filter simulator. The latter has been also factorized to optimize the use of multipliers and pipelined to achieve high operating clock frequency.

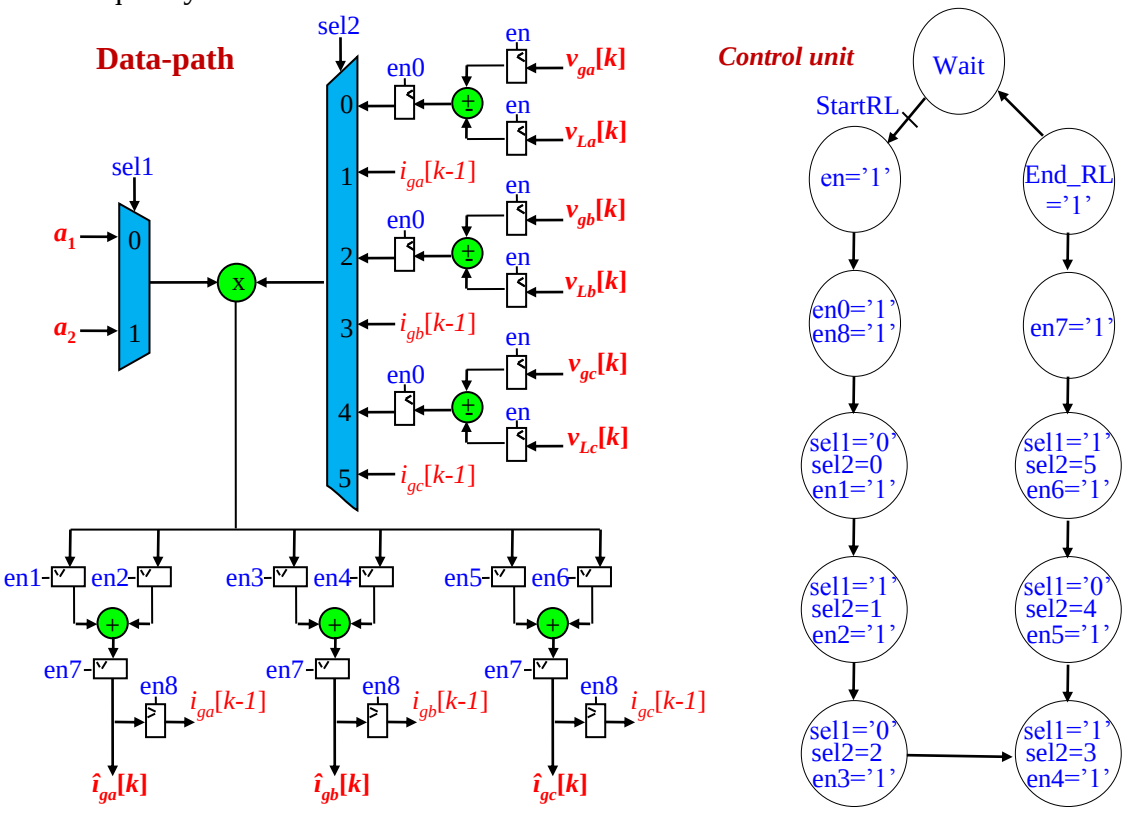


Figure 5. 8: FPGA-based architecture of the 3-phase RL-filter

4.2. Time/area evaluation

Before starting the time/area evaluation, Figure 5.9 gives an overview of the global timing diagram. One can see the three defined computation rates: 50µs for the controller, 2µs for the XADC conversion process and 500ns for the embedded real-time simulator of the 3-phase PWM rectifier (which is the same for the 3-phase RL-filter).

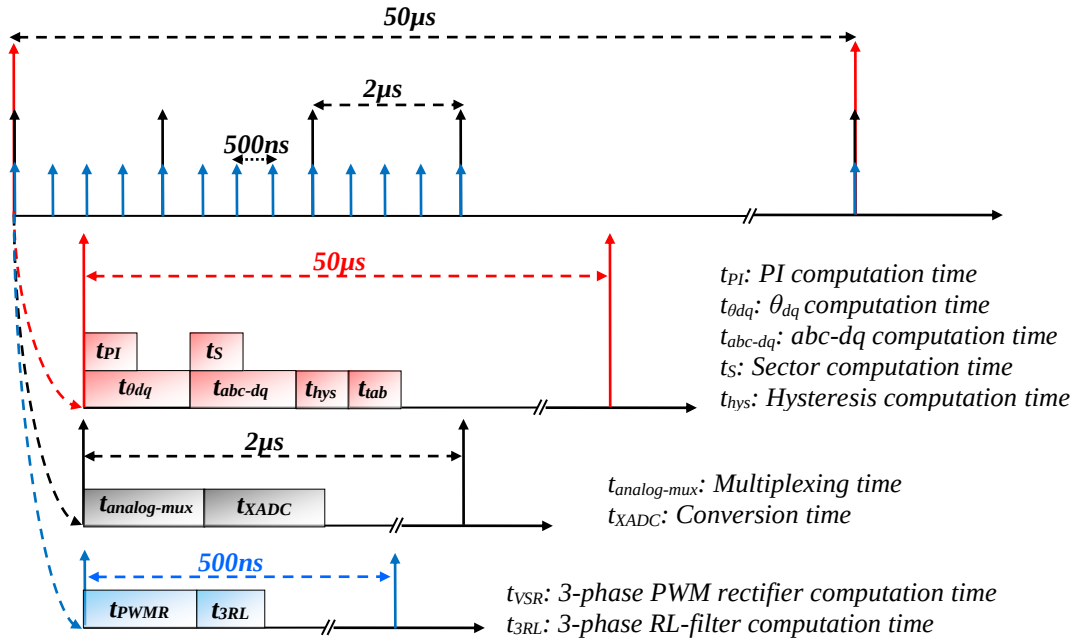


Figure 5. 9: Timing diagram

Then regarding the 3-phase embedded PWM rectifier real-time simulator, the obtained latency is evaluated to 38 which gives a computation time equal to 380ns (with a 100 MHz system clock). Also, the developed architecture uses 180 DSP48E1 units (81.81%), 3982 LUT (7.5%) and 6410 Flip-Flops (6.02%). Note that with the chosen level of factorization, this IP uses 44 32-bit multipliers to execute 88 multiplications.

As for 3-phase RL-filter, it uses four DSP48E1 units (each one integrates a 25x18 bit multiplier) to execute only one 32-bit multiplier. The additional used hardware resources are: 2(0.003%) LUTs and 15(0.014%) Flip-Flops. The total latency is equal to 15 and the computation time is equal to 150ns (with a 100MHz system clock).

Finally, the whole architecture including the controller, the XADC conversion unit, the 3-phase RL-filter and PWM rectifier embedded real-time simulators uses 19.68% of the available 13300 slices (11% of LUTs and 9% of Flip-Flops and 84.54% of the available 220 DSP48E1 units. Table 5.1 summarizes these time/area performances.

Table 5. 1: Timing/Area performances

Modules		Latency	Computation time
Grid current estimator	*PWM rectifier*	*38*	t_{PWMR}=380ns
	3-phase RL-filter	*15*	t_{3RL}=150ns
DC-link and grid currents controller	*PI*	*8*	t_{PI}=80ns
	Position estimation	*20*	$t_{\theta dq}$=200ns
	abc-dq	*16*	t_{abc-dq}=160ns

	Sector	*1*	t_S=10ns
	Hysteresis	*4*	t_{hys}=40ns
	Table	*2*	t_{tab}=20ns
Total consumed resources			*19.68%*
Total occupied DSP48E1			*84.54%*

5. Experimentations

5.1. HIL tests

In order to ensure a first realistic validation of the developed control system, an HIL validation test has been made. To this aim, an FPGA-based real-time emulator of the power system under control has been added to the design. Note that the same IP modules used to develop the embedded real-time estimators have been exploited to emulate this power system. To debug and to view the internal signals running in FPGA, the ChipScope analyzer has been used. In the following, author presents the closed-loop HIL results obtained after having injected the estimated grid currents to the controller.

Figure 5.10 compares the estimated and emulated grid currents during load connection (a) and at steady state (b). This test is validated since the estimated grid currents are similar to those of the power system emulator.

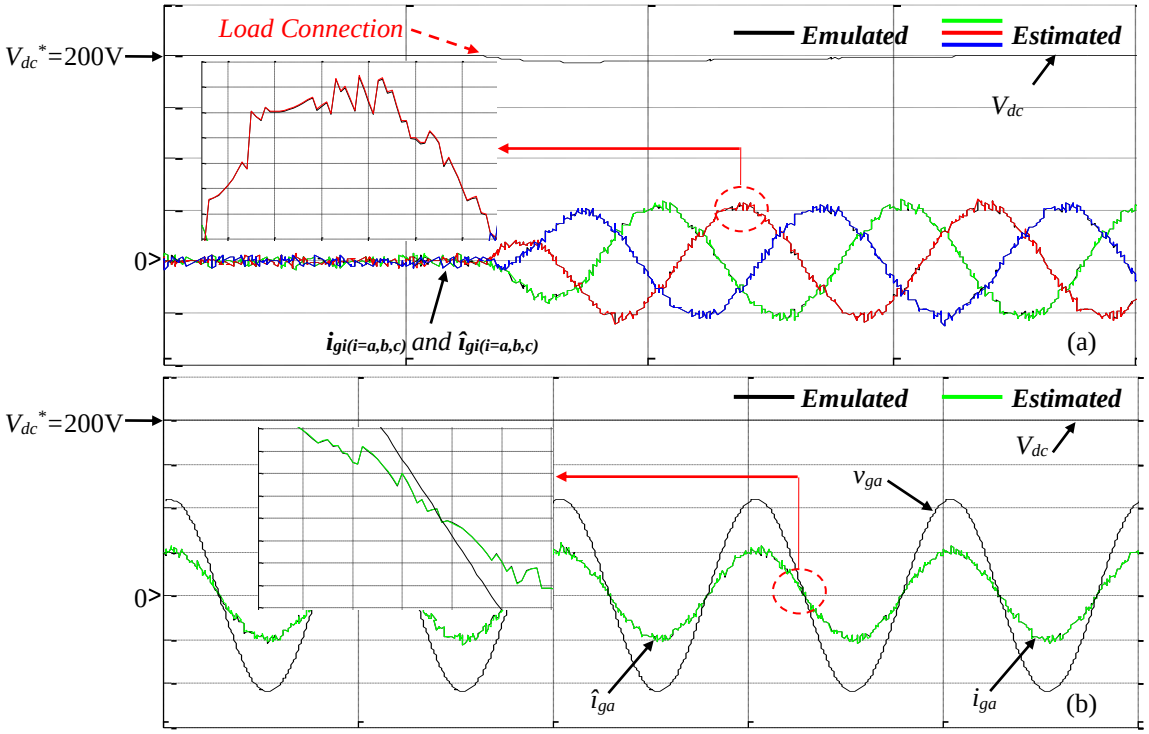


Figure 5. 10: Closed loop real-time HIL results during (a) load connection and (b) at steady state (h: 50ms/div; v: 50V/div, 2.5A/div)

5.2. Experimental validation

The developed experimental platform is shown in Figure 5.11 and the experimental tests were carried out according to the steps given thereafter.

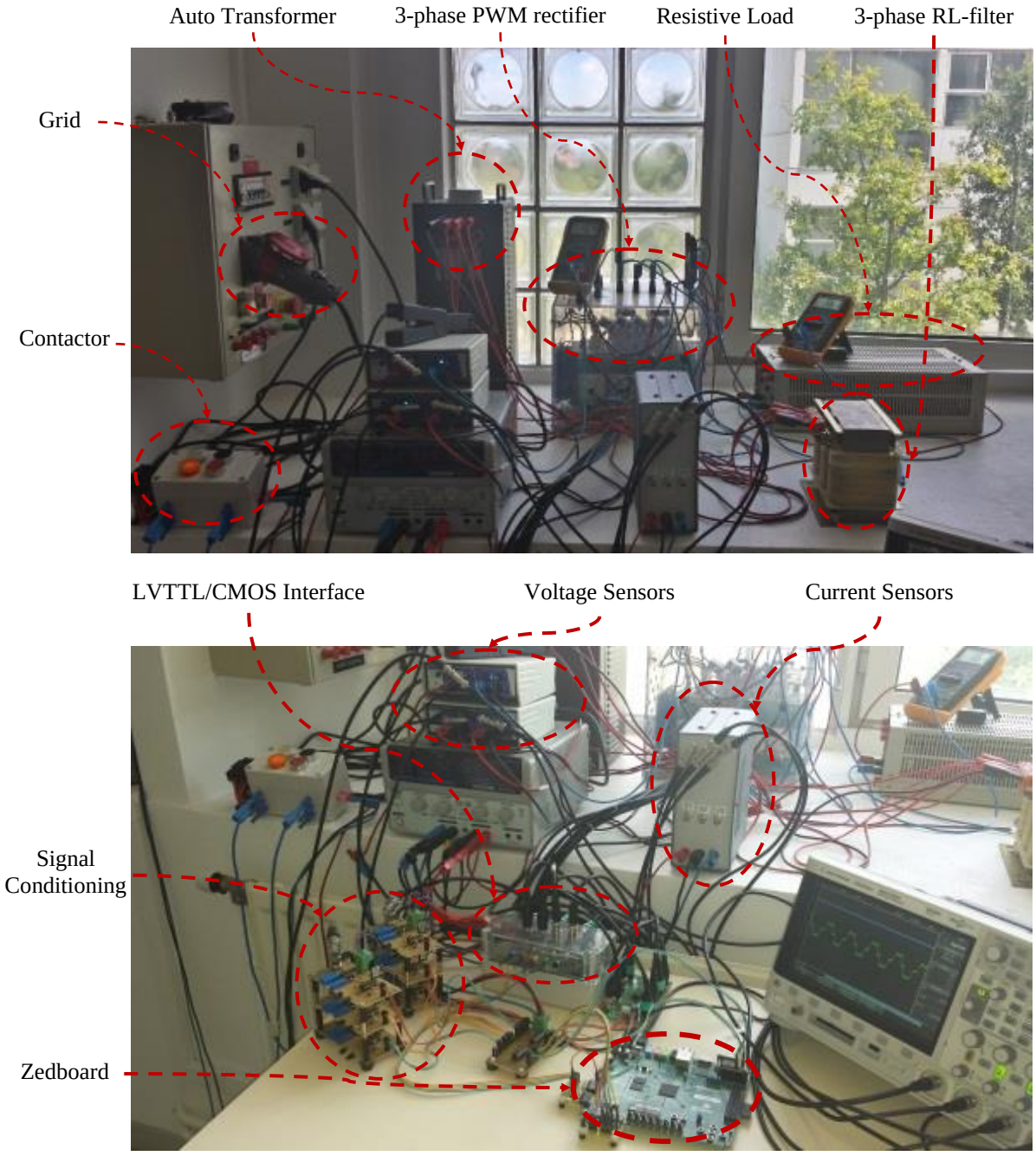


Figure 5. 11: Experimental setup

Step 1: The resistive load (in the DC side) is disconnected, and all the control signals applied to the power converter are set to zero. In this case, the power converter works as a simple diode voltage source rectifier. The magnitude of the DC-link voltage was set equal to 190V by acting on the auto-transformer ratio. The resistive load is alternatively connected and disconnected.

Figure 5.12 and Figure 5.13 show the good matching of the estimated and the measured grid current during this diode rectifier operating mode when respectively the load is connected and disconnected.

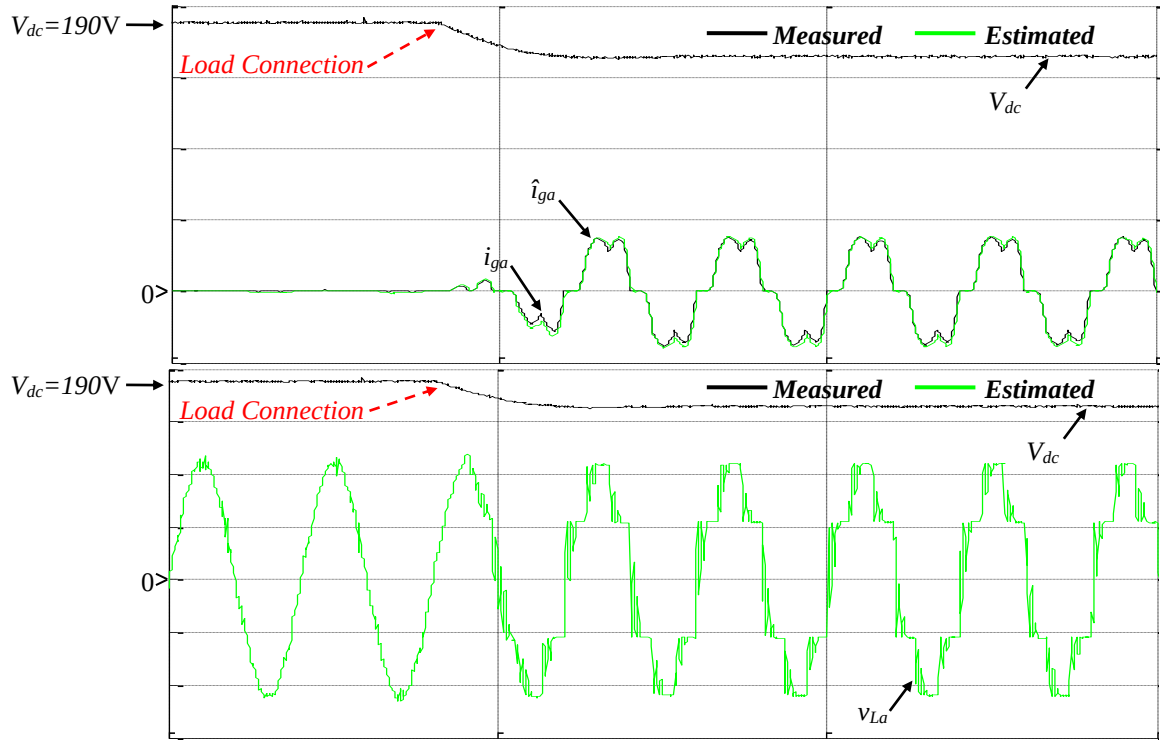


Figure 5. 12: Measured V_{dc}, measured and estimated i_{ga}, estimated v_{La} during diode PWM rectifier operation mode when the load is connected (h: 50ms/div; v: 50V/div, 2.5A/div)

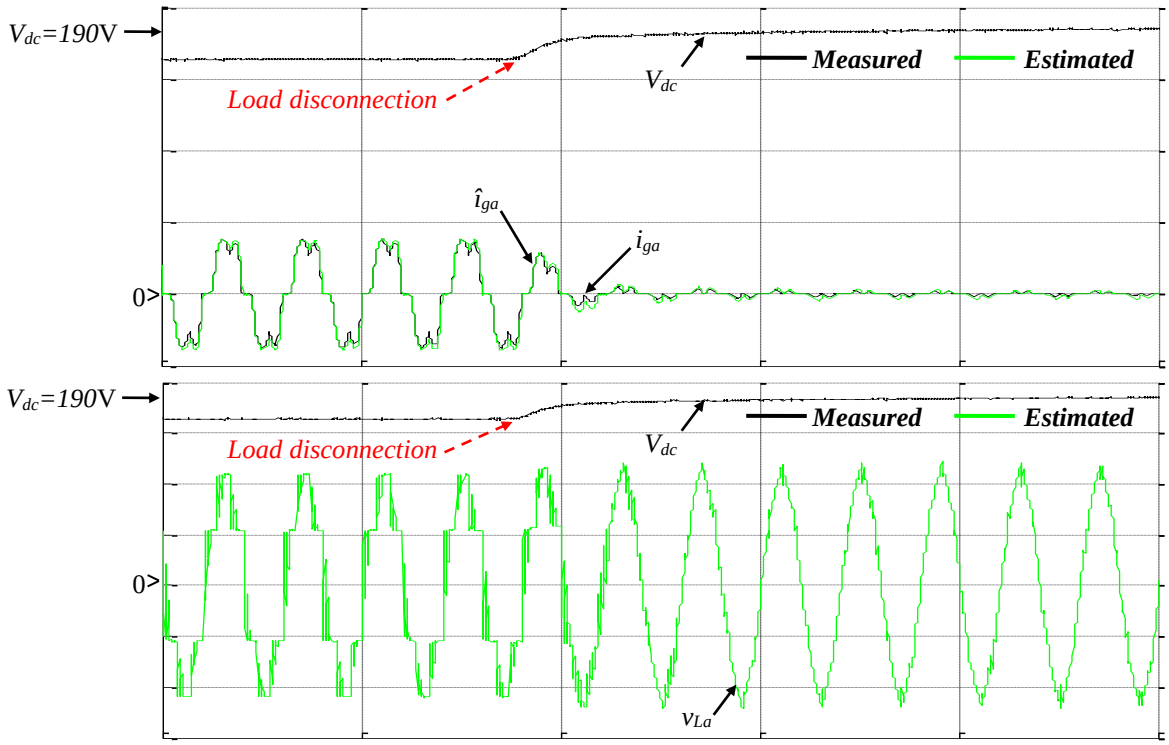


Figure 5. 13: Measured V_{dc}, measured and estimated i_{ga}, estimated v_{La} during diode rectifier operation mode when the load is disconnected (h: 50ms/div; v: 50V/div, 2.5A/div)

Step 2: The computed control signals are now applied to the power converter in order to impose the V_{dc} voltage equal to its reference (200V here). Up to this step, the measured grid currents are used by the controller.

Step 3: The resistive load is connected to the DC-link.

Step 4: The estimated grid currents processed are now used by the controller rather than the measured ones. Switching between measured and estimated currents is done by a simple switch that is used to model current sensor fault occurrence.

Figure 5.14 shows the responses of the DC-link voltage and the measured grid current before and after fault. During the latter, a slight disturbance is observed on the grid current and the DC-link voltage but these latter remain always well controlled.

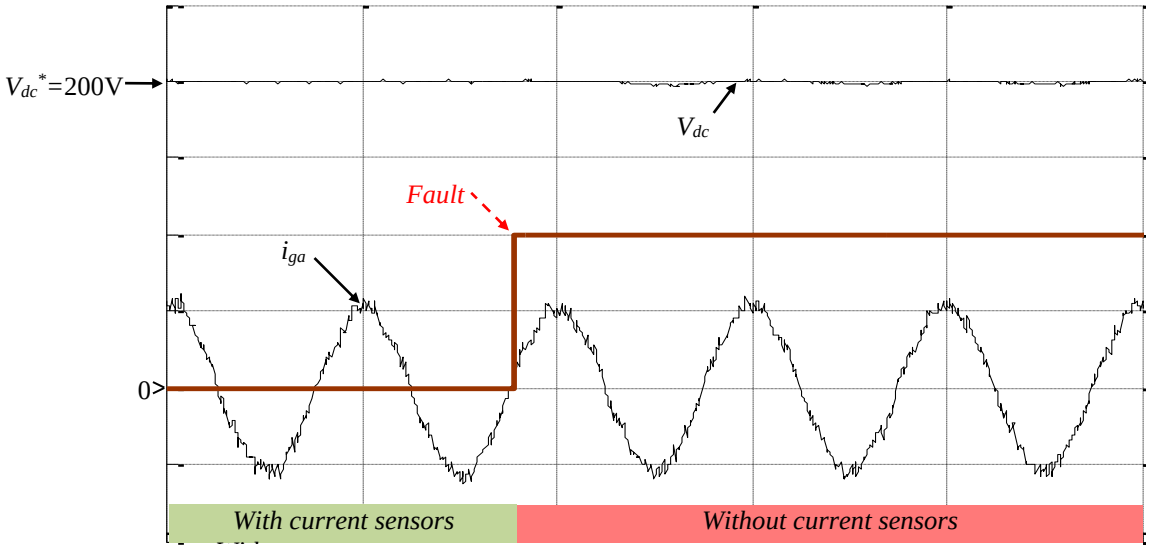


Figure 5. 14: Measured V_{dc} and i_{ga} before and after current sensor fault (h: 20ms/div; v: 50V/div, 2.5A/div)

Step 5: The resistive load is alternatively connected and disconnected (during fault). Figure 5.15 and Figure 5.16 give the waveforms of the measured and estimated grid currents.

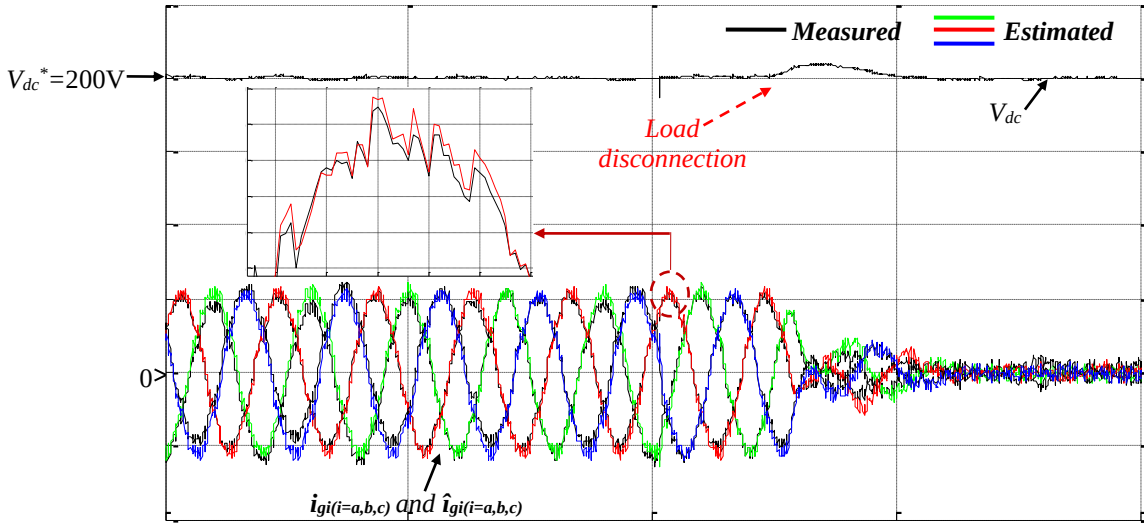


Figure 5. 15: Measured V_{dc}, measured and estimated $i_{gi(i=a,b,c)}$ when the load is disconnected (h: 50ms/div; v: 50V/div, 2.5A/div)

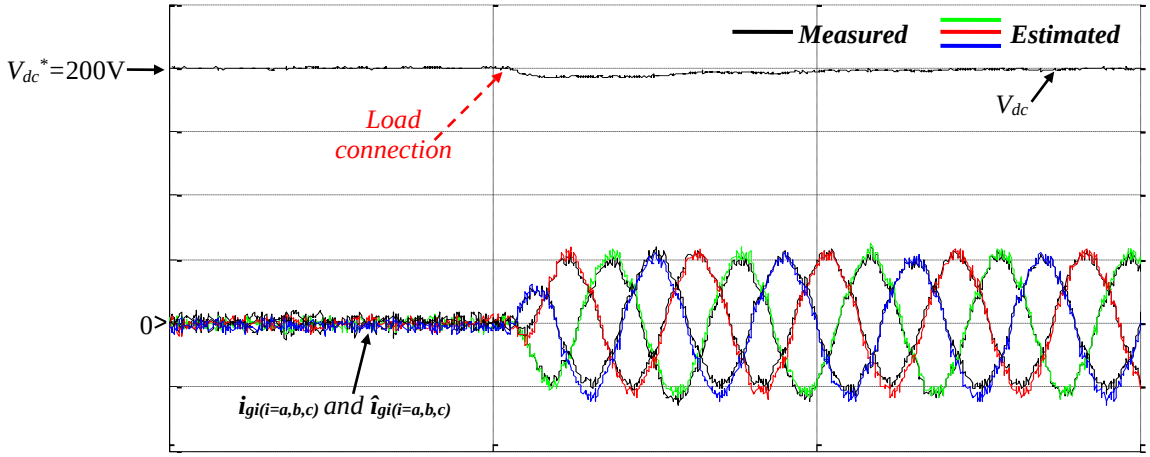


Figure 5. 16: Measured V_{dc}, measured and estimated $i_{gi(i=a,b,c)}$ when the load is connected (h: 50ms/div; v: 50V/div, 2.5A/div)

As shown from these figures, all the estimated and measured grid currents are quite similar. The obtained results demonstrate that the developed embedded real-time estimator can be used efficiently to maintain normal service in the case of current sensors fault occurrence.

Also, Figure 5.17 shows experimental waveforms of the power switches voltages and currents processed by the PWM rectifier embedded real-time simulator during ON/OFF and OFF/ON commutations. These waveforms are similar to those obtained during offline simulation (Figure 5.5).

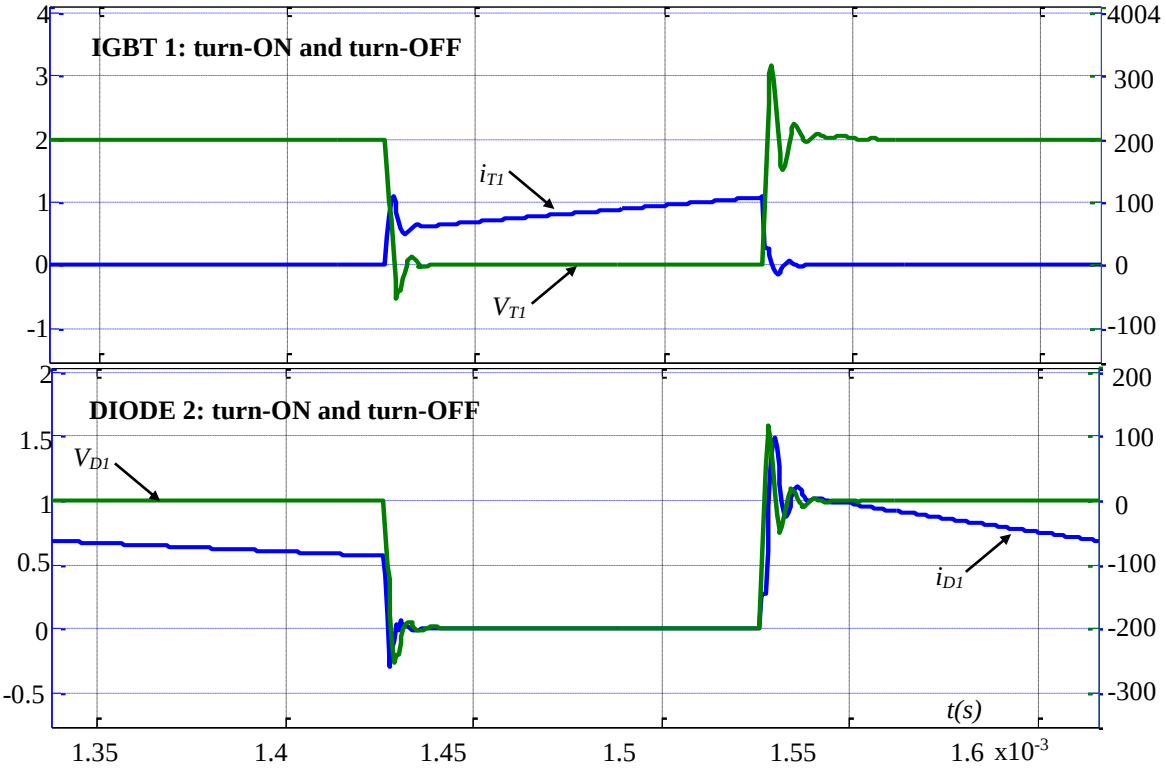


Figure 5. 17: Experimental results during switches commutation (h: 50µs/div; v: 100V/div, 1A/div for IGBT, 0.5A/div for diode)

Figure 5.18 and Figure 5.19 give experimental waveforms of respectively the line voltages v_{La}, v_{Lb} and v_{Lc} and the IGBT/Diode currents and voltages processed by the PWM rectifier embedded real-time simulator during steady state.

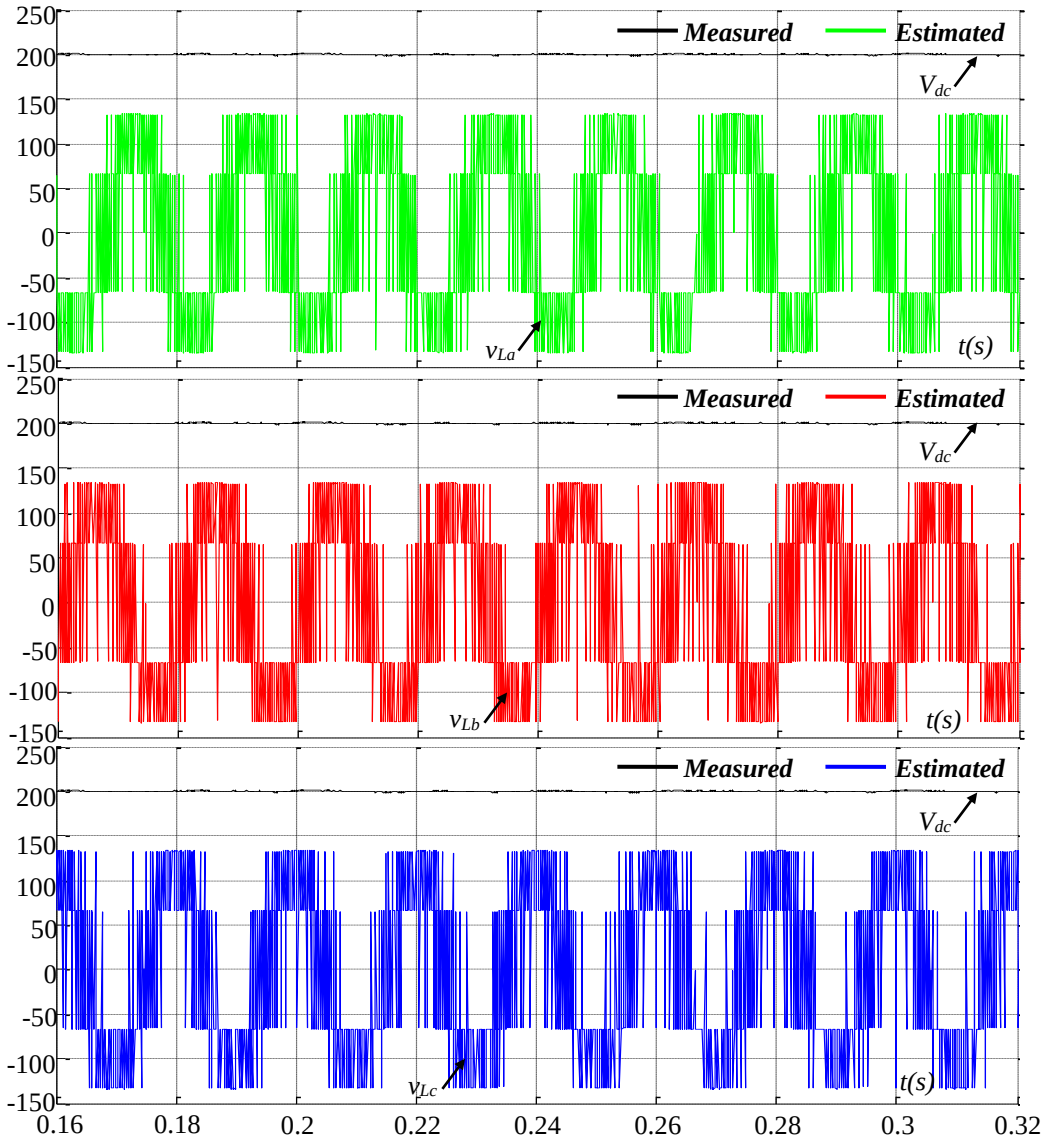


Figure 5. 18: Measured V_{dc} estimated $v_{Li(i=a,b,c)}$ during steady state

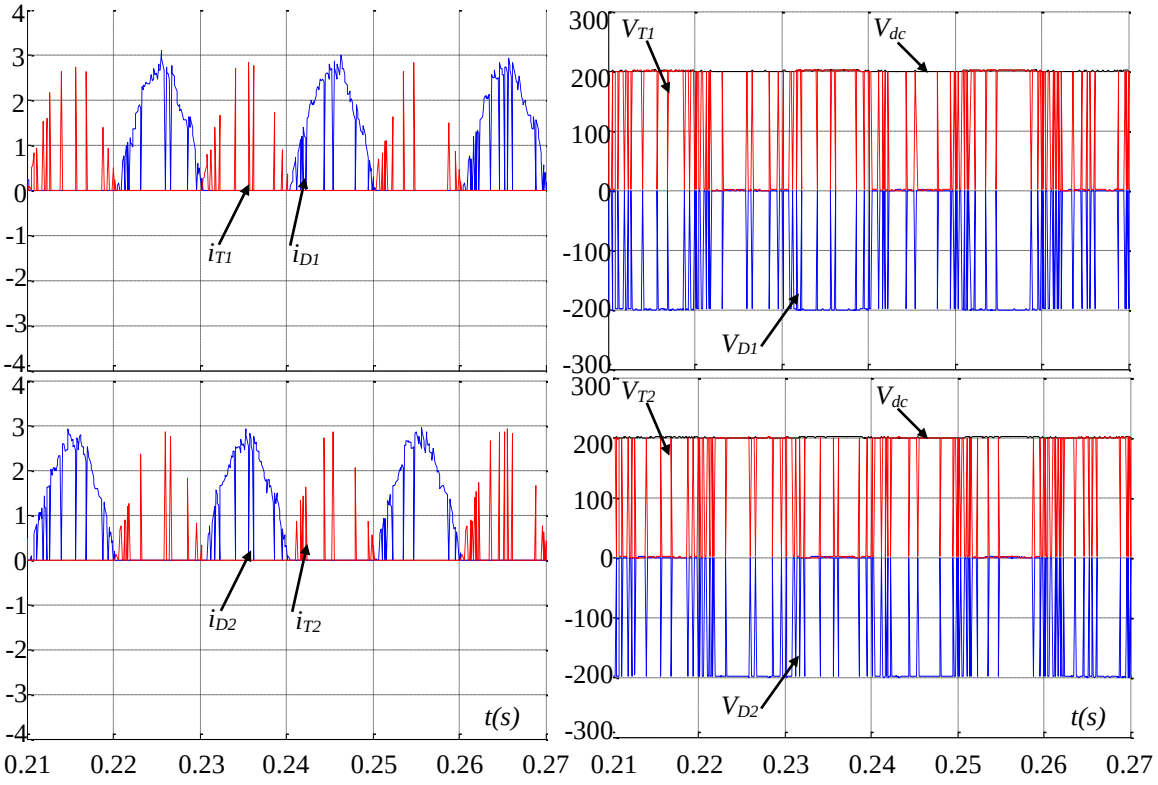


Figure 5. 19: Estimated IGBT/Diode currents and voltages during steady state

The ability of the embedded real-time simulators to ensure sensorless current control with a reduced number of sensors is also demonstrated.

Figure 5.20 and Figure 5.21 compare the measured and estimated grid currents obtained during startup and steady state with and without grid current sensors.

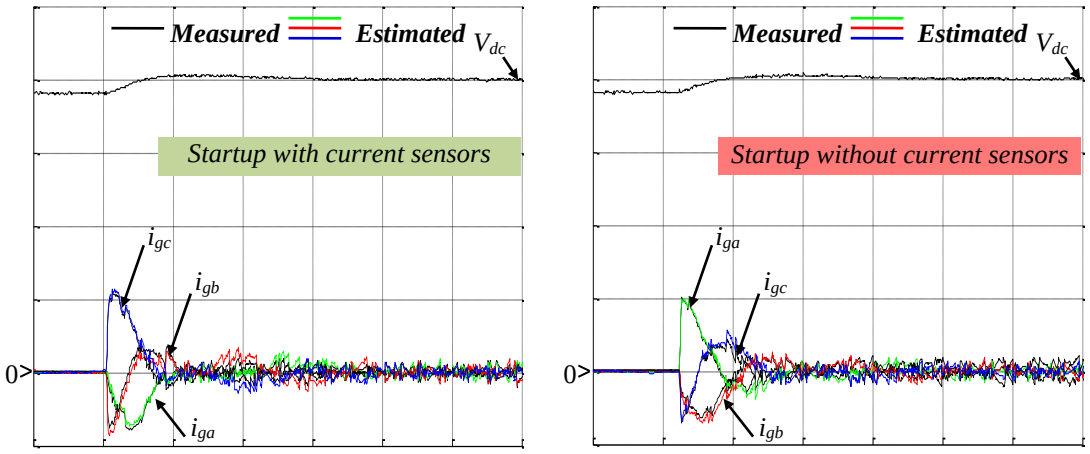


Figure 5. 20 : Measured V_{dc}, measured and estimated $i_{gi(i=a,b,c)}$ during startup with and without grid current sensors (h: 1ms/div; v: 50V/div, 2.5A/div)

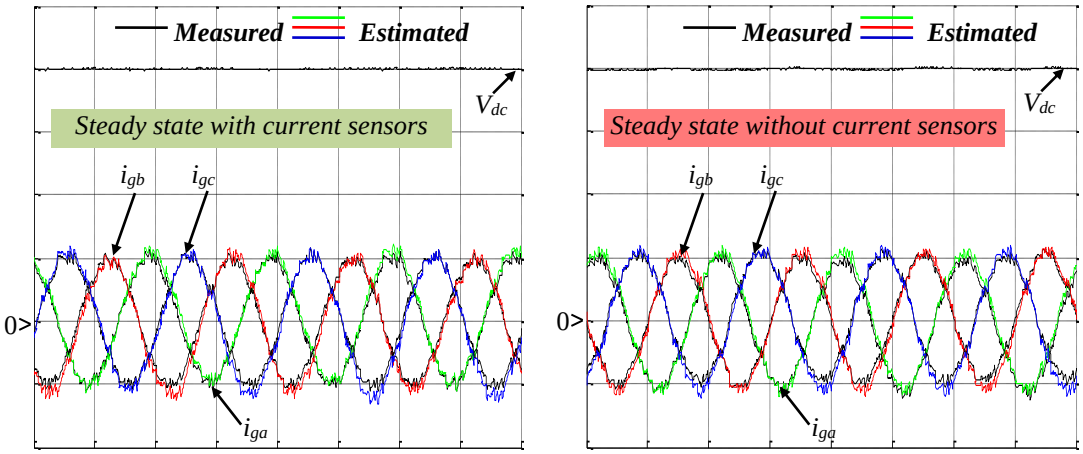


Figure 5. 21: Measured V_{dc}, measured and estimated $i_{gi(i=a,b,c)}$ during steady state with and without grid current sensors (h: 1ms/div; v: 50V/div, 2.5A/div)

6. Conclusion

This chapter has proposed the implementation in low cost FPGAs of embedded real-time simulators applied in the context of fault-tolerant control of grid-connected voltage source rectifier. Embedded real-time simulator IPs of 3-phase grid-connected PWM rectifier and 3-phase RL-filter have been developed and both are implemented within the rectifier controller in the same Zynq FPGA SoC device to estimate the 3-phase grid currents. The latter are injected to the controller when a fault appears in the current sensors.

The proposed design guidelines were used in order to efficiently develop these IPs always having in mind the constraints they bring. The ability of the developed embedded real-time simulators to ensure the service continuity and maintain the power converter control in case of faults was then validated through a closed-loop HIL testing and experiments.

Regarding the perspectives of this work, further improvements are intended such as including the dead time and implementing additional tasks like online identification algorithms to cope with parameter variations. From the controller perspective, a parallel work is in progress where predictive control strategies of the power converter will be tested. Conversely to a fully hardware FPGA implementation (case of this work) these controllers will be implemented in software using the ARM processor Zynq FPGA SoC device.

General conclusion and perspectives

1. General conclusion

This thesis has dealt with FPGA-based embedded real-time simulation of electrical systems. The objective was to develop an IP-library of embedded real-time simulator IPs that simulate different elements of an electrical system. These simulator IPs were designed to address not only HIL applications but above all low cost embedded control applications, keeping in mind the additional constraints they imply.

To develop these IPs, design guidelines were proposed to be followed in order to get the best compromise between their complexity, their expected accuracy and the size (thus the cost) of the FPGA on which these IPs have to be implemented. With the help of these design guidelines, an IP-library of simulator modules that simulate different power converters based on the ADC technique in addition to electromagnetic IP modules were proposed.

As a proof of concept, a complete application where both electromagnetic and switching IPs are deployed and applied in the context of an embedded control application was proposed. The chosen application consists of a fault-tolerant control of a grid-connected voltage source rectifier. Thus, an embedded real-time simulator IP of the 3-phase PWM rectifier was associated with the one of a 3-phase RL-filter and both were implemented within the rectifier controller to estimate the grid currents. These currents are injected in the controller in the case of a current sensor fault. The ability of this estimator to guarantee the service continuity in the case of faults was validated through HIL tests and experiments.

At the beginning of this thesis report, a state of the art real-time simulation of electrical systems was presented. In this chapter, author has discussed the advantages of a real-time simulation over an offline one and presented an overview of the latest application trends. The case of HIL applications has been focused on. Finally, a deeper discussion about the modeling of electrical systems, its digital realization and its digital implementation were made. This chapter was followed by a state of the art FPGA-based embedded real-time simulation of electrical systems. At first, the applications where embedded real-time simulators can be encountered were discussed. Then, the benefits of using FPGAs as embedded digital systems were presented. Also, the most important constraints linked to the FPGA-based embedded real-time simulators were emphasized. Finally, design guidelines to be followed to efficiently design the FPGA-based embedded real-time simulator IPs always having in mind the stringent constraints they imply were proposed.

In chapter 3, the implementation in low cost FPGA of embedded real-time simulator IPs of electromagnetic elements was discussed. The implementation of three electromagnetic IPs modules was studied. Thus, the embedded real-time simulator of a 3-phase DC-excited synchronous machine, the one of a 3-phase induction machine and finally the one of a three-stage avionics alternator were all implemented and added to the IP-library. The obtained results have shown the ability of all the developed simulators IPs to reproduce accurately the dynamic behaviors of the simulated electromagnetic elements. They have shown also that all the developed simulator IPs enables the use of small simulation time-step and consumes moderate hardware resources. Their ability to address both embedded control and HIL applications was then confirmed. In this chapter, the advantages of using delta operator rather than shift operator to design digital realizations of AC machines were also examined. It has been shown that delta-operator based realization is more convenient in the context of this work since it allows using small simulation time-step and limited fixed-point data word length without affecting the system stability.

In chapter 4, the FPGA implementation of embedded real-time simulator IPs of static converters was discussed. The implementation of three IPs was studied: the embedded real-time simulator of a single phase DC-AC converter, the one of a 3-phase voltage source inverter and finally the one of a 3-phase diode rectifier were all implemented and added to the IP-library. The developed real-time simulators were validated in the context of HIL testing. For each developed simulator, real-time simulation results were provided and compared with those of offline simulations. The obtained real-time simulation results have clearly demonstrated the effectiveness of the developed simulator IPs in terms of accuracy and their ability to be embedded within low-cost FPGA-based controllers using moderate consumed resources and small time-steps.

In chapter 5, a complete embedded control application was presented. Thus, embedded real-time simulator IPs of PWM rectifier (switching element) and RL filter (electromagnetic element) was developed and applied in the context of fault-tolerant control of a grid-connected voltage source rectifier. They are both embedded within an FPGA-based PWM rectifier controller and used as a grid current estimator. The ability of this estimator to guarantee the service continuity and to maintain normal services in the case of current sensor faults was validated through closed-loop HIL tests and experiments.

2. Perspectives

Regarding the perspectives of this work, further improvements are intended at different step of the development process of an embedded real-time simulator IP.

When focusing on the algorithm development and particularly on the model selection step, it is interesting to study the use of more detailed system models that include more nonlinear phenomena such as saturation and saliency effects. At the digital realization step, an in-depth analysis of the advantage of using floating-point data format for cost-sensitive embedded real-time simulation applications is still required.

Additional perspectives related to the FPGA architecture design are to be focused on. Conversely to a fully hardware FPGA implementation (case of this work), it is also important to evaluate the value-added of FPGAs for implementing an embedded simulator IP using a Hardware/Software co-design approach.

From application perspective, it is interesting to evaluate the benefits of using the developed ADC-based embedded real-time power converter simulator IP for estimation of conducting and switching losses. As for the developed electromagnetic simulator IPs, it is also important to study their advantages to be used for online identification of the system parameters in order to cope with the parameter variations.

Appendices

Appendix-A: Current-voltage relations of an ADC equivalent circuit

In this section the extraction of current-voltage relations of an ADC equivalent circuit with the MNA method is presented. As introduced previously, this method consists in representing an electrical circuit of *n* nodes and *m* independent voltage sources with a matrix equation of the form:

$$b = \begin{bmatrix} b_{11} \\ b_{21} \\ \vdots \\ b_{n+m,1} \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1,n+m} \\ h_{21} & h_{22} & \cdots & h_{2,n+m} \\ \vdots & \vdots & \vdots & \vdots \\ h_{n+m,1} & h_{n+m,2} & \cdots & h_{n+m,n+m} \end{bmatrix} . \begin{bmatrix} x_{11} \\ x_{21} \\ \vdots \\ x_{n+m,1} \end{bmatrix} = H.x$$

Where *H* is the conductance matrix, *x* is the vector of unknown variables typically node voltages and currents through independent voltage sources, *b* is the vector of known variables typically voltages sources and combinations, at each voltage node, of independent current sources.

Extracting current-voltage relations of an ADC equivalent circuit consists then in determining the conductance matrix *H* and the *x* and *b* vectors. In the following, the steps to be followed to extract this matrix and vectors are presented and illustrated throughout a simple example. The chosen ADC equivalent circuit is the one of a half-wave rectifier that supplies a resistive load (Figure A.1).

Step 1: As illustrated in Figure A.1, the first step consists in selecting a reference node (usually ground), numbering remaining nodes, labeling currents through each current source and assigning a name to current through each voltage source.

The studied circuit contains two voltage nodes which are labeled V_1 and V_2 and one independent voltage source labeled *E*. The current through this latter is labeled i_E. This circuit contains also one current source J_D. Note that, the number of current sources is dependent on the way the load is modeled. When this latter is modeled as an external load, load current i_L has to be considered as a current source. Indeed, in this case, the load will not appear in the matrix *H* as a conductance (*1/R*) as explained thereafter but rather in the vector *b* as a current source.

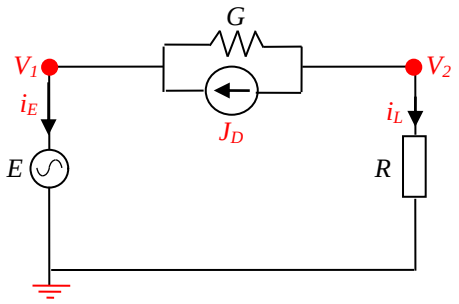


Figure A. 1: ADC equivalent circuit of a half-wave rectifier that supplies a resistive load

Step 2: Determine the *H* matrix, *x* and *b* vectors as follows:

The H matrix:

- Is (*n*+*m*)x(*n*+*m*) matrix and contains only known quantities.
- The *(n)*x*(n)* part of the matrix in the upper left:
  - Has only passive elements
  - Each element connected to ground appear only on the diagonal
  - Elements not connected to ground are both on the diagonal and off-diagonal terms

- h_{ii} is equal to the sum of the conductance connected to the node i
- h_{ij} is equal to minus the sum of the conductance between node i and node j

- The rest of the H matrix (not included in the (n)x(n) upper left part) contains only 1, -1 and 0.

$$h_{ij} = \begin{cases} 1\text{: if positive terminal of voltage source number } j \text{ is connected to node number } i \\ -1\text{: if negative terminal of voltage source number } j \text{ is connected to node number } i \\ 0 : \text{else} \end{cases}$$

Applying all these sub-steps to the studied circuit, the extract conductance matrix is then as follows:

$$H = \begin{bmatrix} G & -G & 1 \\ -G & G + 1/R & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{matrix} \\ n=2 \\ m=1 \end{matrix} \qquad H = \begin{bmatrix} G & -G & 1 \\ -G & G & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{matrix} \\ n=2 \\ m=1 \end{matrix}$$

(a) *(b)*

Figure A. 2: (a) Conductance H matrix with internal load modeling (b) conductance H matrix with external load modeling

The x vector:

- Is an $(n+m)$x1 vector that contains unknown quantities (node voltages and currents through the independent voltage sources).
- The top n elements are the n node voltages.
- The bottom m elements represent the currents through the m independent voltage sources.

$$x = \begin{bmatrix} V_1 \\ V_2 \\ i_E \end{bmatrix} \begin{matrix} \\ n=2 \\ m=1 \end{matrix}$$

The b vector:

- Is an $(n+m)$x1 vector that contains only known quantities.
- The top n elements are either zero or the sum and difference of independent current sources for each node (convention: + if the current source is going in the node).
- The bottom m elements represent the m independent voltage sources.

$$b = \begin{bmatrix} J_D \\ -J_D \\ E \end{bmatrix} \begin{matrix} \\ n=2 \\ m=1 \end{matrix} \qquad x = \begin{bmatrix} J_D \\ -J_D - i_L \\ E \end{bmatrix} \begin{matrix} \\ n=2 \\ m=1 \end{matrix}$$

(a) *(b)*

Figure A. 3: (a) b vector with internal load modeling (b) b vector with external load modeling

Step 3: After having extracted the conductance H matrix and the b and x vectors, the last step consists in solving the system equation by a simple matrix inversion:

(a)

$$x = \begin{bmatrix} V_1 \\ V_2 \\ i_E \end{bmatrix} = H^{-1}.b = \begin{bmatrix} 0 & \frac{0}{R} & \frac{1}{G.R} \\ 0 & \frac{}{G.R+1} & \frac{}{G.R+1} \\ 1 & \frac{G.R}{G.R+1} & \frac{-G}{G.R+1} \end{bmatrix}.\begin{bmatrix} J_D \\ -J_D \\ E \end{bmatrix}$$

(b)

$$x = \begin{bmatrix} V_1 \\ V_2 \\ i_E \end{bmatrix} = H^{-1}.b = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & \frac{1}{G} & 0 \end{bmatrix}.\begin{bmatrix} J_D \\ -J_D - i_L \\ E \end{bmatrix}$$

Figure A. 4: System equation with internal load modeling (b) system equation with external load modeling

Appendix-B: Parameters of the delta-operator based synchronous machine model

In this section the expressions of the parameters of the normalized delta-operator based synchronous machine model are presented.

$$d_1 = \frac{-2.R_S}{2.L_{sd} + R_S.\Delta t}, \quad d_2 = \frac{2.L_{sq}.\omega_B}{2.L_{sd} + R_S.\Delta t}, \quad d_3 = \frac{2.V_B}{(2.L_{sd} + R_S.\Delta t).I_B}$$

$$d_4 = \frac{L_{sq}.\omega_B.\Delta t}{2.L_{sd} + R_S.\Delta t}, \quad d_5 = \frac{V_B.\Delta t}{(2.L_{sd} + R_S.\Delta t).I_B}, \quad d_6 = \frac{-2.R_S}{2.L_{sq} + R_S.\Delta t}$$

$$d_7 = \frac{-2.L_{sd}.\omega_B}{2.L_{sq} + R_S.\Delta t}, \quad d_8 = \frac{-2.M_{sr}.\omega_B.I_{rd}}{(2.L_{sq} + R_S.\Delta t).I_B}, \quad d_9 = \frac{2.V_B}{(2.L_{sq} + R_S.\Delta t).I_B}$$

$$d_{10} = \frac{-L_{sd}.\omega_B.\Delta t}{2.L_{sq} + R_S.\Delta t}, \quad d_{11} = \frac{-M_{sr}.\omega_B.I_{rd}.\Delta t}{(2.L_{sq} + R_S.\Delta t).I_B}, \quad d_{12} = \frac{V_B.\Delta t}{(2.L_{sq} + R_S.\Delta t).I_B}$$

$$d_{13} = \frac{\omega_B}{\theta_B}, d_{14} = \frac{\omega_B.\Delta t}{2.\theta_B}, \quad d_{15} = \frac{3.p.(L_{sd} - L_{sq}).I_B^2}{2.C_B}, \quad d_{16} = \frac{3.p.M_{sr}.I_{rd}.I_B}{2.C_B}$$

$$d_{17} = \frac{-2.f}{2.J + f.\Delta t}, \quad d_{18} = d_{19} = \frac{2.p.C_B}{(2.J + f.\Delta t)\times \omega_B}, \quad d_{20} = d_{21} = \frac{p.C_B.\Delta t}{(2.J + f.\Delta t).\omega_B}$$

Where: V_B, I_B, ω_B, θ_B and C_B are the base quantities that are used for the normalization. Their numerical values are listed in Table B.1.

Table 3. 5: Base quantities used for the normalization of the delta-operator based synchronous machine model

Voltage base value	Current base value	Angular speed base value	Angular position base value	Torque base value
V_B = 936V	I_B = 20A	ω_B = 419 rd/s	$\theta_B = 2\pi$	C_B=10 Nm

Appendix-C: Poles of the delta-operator based synchronous machine model

In this section, the way-to-calculate the pole of the delta-operator based synchronous machine model is presented. To start with, let's recall that the continuous-time based model assumes the classical decoupling between mechanical and electrical modes. The state-space continuous-time electrical model is then:

$$\begin{cases} \dot{x} = A.x + B.u \\ y = C.x + D.u \end{cases} \quad \text{(C.1)}$$

Where, $x=y=\begin{bmatrix} i_{sd} \\ i_{sq} \end{bmatrix}$, $u=\begin{bmatrix} V_{sd} \\ V_{sq} \\ I_{rd} \end{bmatrix}$, matrices A, B, C, and D are 4-by-4, 4-by-2, 4-by-4, and 4-by-2 respectively.

$$A(\omega_e) = \begin{bmatrix} \frac{-R_s}{L_{sd}} & \frac{L_{sq}}{L_{sd}}.\omega_e \\ \frac{-L_{sd}}{L_{sq}}.\omega_e & \frac{-R_s}{L_{sq}} \end{bmatrix} = \begin{bmatrix} c_1 & c_2.\omega_e \\ c_4.\omega_e & c_5 \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{1}{L_{sd}} & 0 & 0 \\ 0 & \frac{1}{L_{sq}} & -\frac{M_{sr}}{L_{sq}} \end{bmatrix}, C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, D = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The poles p_{ci} of the continuous-time model are then the eigenvalues of the state matrix $A(\omega_e)$. They are equal to two and their expressions are given bellow.

$$\begin{cases} p_{c1} = \frac{c_5 + c_1}{2} + i\frac{\sqrt{-(c_1 + c_5)^2 + 4.c_1.c_5 - 4.c_2.c_4.\omega_e}}{2} \\ p_{c2} = \frac{c_5 + c_1}{2} - i\frac{\sqrt{-(c_1 + c_5)^2 + 4.c_1.c_5 - 4.c_2.c_4.\omega_e}}{2} \end{cases} \quad \text{(C.2)}$$

These continuous-time poles are function of the angular speed ω_e. The latter has been selected to be the worst-case value (corresponds to the continuous-time pole having the greatest module). It has been fixed to the maximal value.

Before calculating the poles of the delta-operator based model, the ones p_{ti} of the discrete-time implicit trapezoidal based model is firstly obtained from the p_{ci} ones using the following relation.

$$s \to \frac{2}{\Delta t}.\frac{z-1}{z+1} \quad \text{(C.3)}$$

Based on C.3, their general expression is obtained and given in relation C.4.

$$p_{ti} = \frac{p_{ci} + \frac{2}{\Delta t}}{\frac{2}{\Delta t} - p_{ci}} \tag{C.4}$$

Now as the expression of the poles of shift-operator based model are obtained, the p_{di} ones of the delta-operator based model can be determined using the following delta transformation:

$$z \to 1 + \gamma . \Delta t \tag{C.5}$$

The poles of the delta-operator based model have then the following expression:

$$p_{di} = \frac{1}{\frac{1}{p_{ci}} - \frac{\Delta t}{2}} \tag{C.6}$$

Appendix-D: Parameters of the induction machine

Table D.1 gives the parameters of the simulated induction machine.

Table D. 1: Induction Machine Parameters

1 KW, 230V, 50 Hz, 3 Phases, 2 poles	
Stator resistance R_s= 7.2 Ω	Rotor resistance R_r= 1.35 Ω
stator inductance L_s = 0.28 H	rotor inductance L_r= 0.075 H
Number of pole pairs p =2	Mutual inductance M_{sr}= 0.118 H
Rotor inertia J=0.006 Kg.m²	Viscous friction coefficient f=0.046 Kg.m²/s

The base values are obtained from the maximal values by using the following equations:

$$I_B = \sqrt{2}.I_m \tag{D.1}$$

$$V_B = \sqrt{2}.V_m \tag{D.2}$$

$$\omega_B = 2.PI.f_m \tag{D.3}$$

$$\Phi_B = \frac{V_m}{\omega_B} \tag{D.4}$$

$$C_B = \frac{\Phi_B}{I_B} \tag{D.5}$$

Where:

I_m : the phase maximal current.

V_m : the phase to neutral maximal voltage.

f_m : the maximal frequency of the induction motor.

Appendix-E: Components of the three-stage avionics alternator

In this section a detailed description of the three-stage avionics alternator components, their models and their parameters is presented.

1. Main generator

The main generator is a salient-pole synchronous machine with damper rotor windings. In this work, the first studies were conducted for a machine of 11.2 kVA. The numerical values of its parameters are given in Table E.1.

Table E. 1: Main Generator parameters

Stator resistance R_s= 10.5 Ω	d-q stator inductance L_d=63.6mH L_q=38.6 mH
Rotor resistance R_f= 10.5 Ω	Rotor inductance L_f=695 mH
d-q resistance of the damper windings R_D=0.86mH R_Q=0.99mH	d-q inductance of the damper windings L_Q=0.0236 mH L_D=0.0685 mH
Mutual inductance between the rotor and the damper winding of the d axis M_{fD}=6.7mH	d-q Mutual inductance between the stator and the damper winding M_{sD}=2mH M_{sQ}=0.9 mH
Mutual inductance between the rotor and the stator winding of the d axis M_{sf}=200.5mH	Number of pole pairs p_{mg}=2

As for its modeling, the d-q based model with external Park transformations has been used. To improve the accuracy of this model by taking into account the influence of the load variations on the stator voltages, two modeling methods can be used [BAR11]. The first one is called modeling with internal load. It consists in including the used load in the state matrix of the main generator model. Besides its acceptable accuracy, this method is not suitable since it does not guarantee the stability of the closed-loop system for all types of load. The second method is called modeling with external load because this latter is not included in the state matrix. It consists in adding dummy capacitors between the main generator and the load. The advantage of this method is that the stator voltages are represented as state variables (ie. outputs) and the load currents as disturbance inputs as shown in Figure E.1.

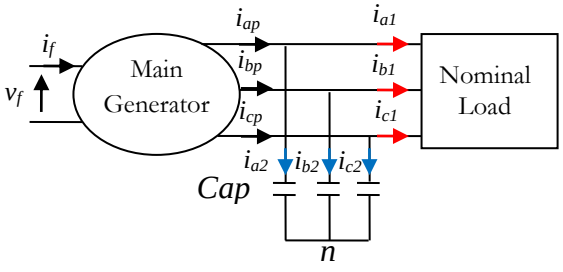


Figure E. 1: Principle of modeling with internal load

This second method is used in this work. The value of the added dummy capacitors is set to Cap=1µF so that its impedance ($1/(Cap.\omega_{ep})$ = 3180 Ω) is very large compared to the load impedance (14 Ω).

The electrical equations of the main generator in the d-q coordinates and after adding the three phase dummy capacitors are [BAR11]:

$$\begin{cases} i_{d1} = i_{dp} + Cap.\omega_{ep}v_{qp} - Cap.\dfrac{d_{v_{dp}}}{dt} \\ i_{q1} = i_{qp} + Cap.\omega_{ep}v_{dp} - Cap.\dfrac{d_{v_{qp}}}{dt} \\ 0 = -v_{dp} - R_s i_{dp} + L_q\omega_{ep}i_{qp} - M_{sQ}\omega_{ep}i_Q - L_d\dfrac{d_{i_{dp}}}{dt} + M_{sf}\dfrac{d_{i_f}}{dt} + M_{sD}\dfrac{d_{i_D}}{dt} \\ 0 = -v_{qp} - R_s i_{qp} - L_d\omega_{ep}i_{dp} + M_{sf}\omega_{ep}i_f + M_{sD}\omega_{ep}i_D - L_q\dfrac{d_{i_{qp}}}{dt} + M_{sQ}\dfrac{d_{i_Q}}{dt} \\ v_f = R_f i_f + L_f\dfrac{d_{i_f}}{dt} - M_{sf}\dfrac{d_{i_{dp}}}{dt} + M_{fD}\dfrac{d_{i_D}}{dt} \\ 0 = R_D i_D + L_D\dfrac{d_{i_D}}{dt} + M_{fD}\dfrac{d_{i_f}}{dt} - M_{sD}\dfrac{d_{i_{dp}}}{dt} \\ 0 = R_Q i_Q + L_Q\dfrac{d_{i_Q}}{dt} - M_{sQ}\dfrac{d_{i_{dp}}}{dt} \end{cases} \quad \text{(E.1)}$$

Where :

i_{d1} *and* i_{q1} *:* d-q currents of the nominal load

v_{qp} *and* v_{qp}*:* d-q stator voltages of the main generator

i_{dp} *and* i_{qp}*:* d-q stator currents of the main generator

i_f *and* v_f: Excitation current and voltage of the main generator

ω_{ep} *:* Electrical speed of the main generator

Cap: Dummy capacitor between the main generator and the load

2. Nominal load

The used external load is a nominal load of impedance equal to 14 Ω. It consumes the nominal apparent power of the main generator with a power factor of 0.8. It has been modelled by an inductance L_n in parallel with a resistance R_n. The values of these parameters have been obtained according to the following equations and are given in Table E.2:

$$R_n = \frac{{U_{RMS}}^2}{P_n} \quad \text{(E.2)}$$

$$L_n = \frac{{U_{RMS}}^2}{Q_n\omega_{\text{ep}}} \quad \text{(E.3)}$$

Where :

U_{RMS} : RMS voltage

P_n : Nominal active power

Q_n: Nominal reactive power

ω_{ep} : Electrical speed of the main generator

Table E. 2: Nominal load parameters

Active power consumed by the load P_n=8.96 KW	Reactive power consumed by the load Q_n=6.72kV AR
Load Resistance R_n=17.8571Ω	Load Inductance L_n=75.8 mH

3. Exciter Machine

The exciter machine is a reverse wound-rotor synchronous machine. It is a low salience machine without dampers. It has 8 poles. Therefore, when its rotor is driven at the speed of 1500 tr/min, the fundamental frequency of its stator voltages is of 100 Hz (twice as large as that of the stator voltage of the main generator which is equal to 50 Hz). Its parameters are given in Table E.3.

Table E. 3 : Exciter machine parameters

Stator resistance R_{se}=*0.26 Ω*	d-q stator inductance L_{de}=5.8 mH L_{qe}=3.1 mH
Rotor resistance *Re*= 24.5 Ω	d-q rotor inductance L_e=1.75 H
Mutual inductance M_{se}=89 mH	Number of pole pairs p_{em}=4

Appendix-F: IPs modules of the 1-phase AC load and P+Resonant current controller

To make a complete validation of the ADC-based embedded real-time simulator IP of the DC-AC converter, this latter has been associated with the ones of a 1-phase AC load and a P+Resonant based current controller. In this section, these additional IPs are presented and their time/area performances are summarized.

1. Embedded real-time simulator IP of the 1-phase AC load

This embedded simulator computes the load current i_L from the sinusoidal Back EMF voltage E and the line voltages v_{La} and v_{Lb} processed by the DC-AC simulator.

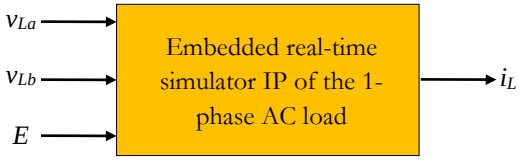


Figure F. 1: Embedded real-time simulator IP of the 1-phase AC load

The relation F.1 presents the implemented discrete-time equation. The latter is obtained after the Forward Euler discretization of the RLE circuit.

$$i_L[k] = a_1.(v_{La}[k-1] - v_{Lb}[k-1] - E[k]) + a_2.i_L[k-1] \qquad \text{(F.1)}$$

Where, $a_1 = \frac{\Delta t}{L}$ and $a_2 = 1 - \frac{\Delta t.R}{L}$.

The chosen time-step has been set to 1µs and the chosen fixed-point format is 24Q23 (24 total bit number and 23 bits in the fractional part).

Figure F.2 presents the developed FPGA-based architecture corresponding to the equation F.1. When using the Xilinx XC7Z020 Zynq device for the implementation of this simulator IP, the resources consumption is evaluated at 1% of the component slices over 13300 (2 (0.007%) LUTs and 6 (0.01%) Flip-Flops). The computation time is equal to 60ns (with a 100MHz system clock).

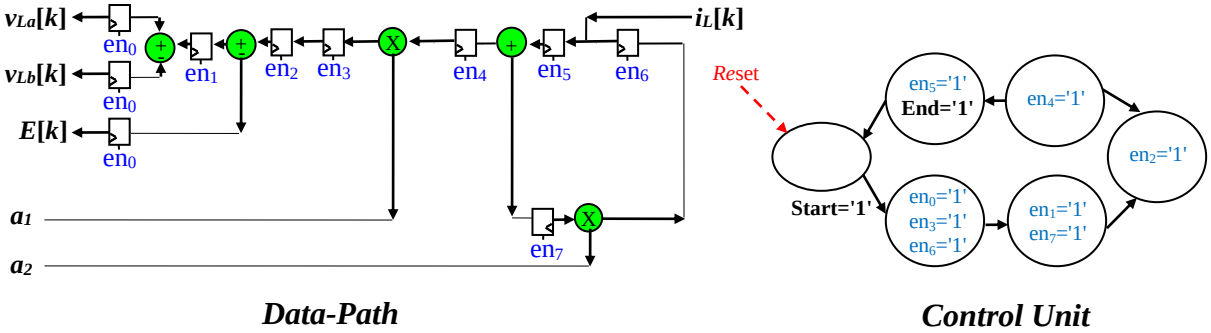


Figure F. 2: FPGA-based architecture of the 1-phase AC load

2. AC Load current controller

The goal of this controller consists in regulating the current of the AC load. To this purpose, the PI regulator can be used. However, this latter is well adopted for the control of constant quantities (e.g. DC current control) and has less performance for the control of AC quantities. For a three phase system, the rotating d-q synchronous frame can be used to

transform the AC currents to DC currents which can be well regulated with PI controller. This latter will be used in the future with a real-time simulator of a three phase system. In the proposed work, a P+Resonant control strategy can be an appropriate solution since it has a significantly improved steady state performance because of its increased closed-loop disturbance rejection at the resonant frequency [KON09].

The P+Resonant controller transfer function expressed in terms of converter voltage reference V^* and current error ε is:

$$C(s) = \frac{V^*(s)}{\varepsilon(s)} = K_p.\frac{K_i}{s^2 + \omega^2} \qquad \text{(F.2)}$$

Where K_p, K_i and ω are respectively the proportional gain, the integral gain and the fundamental frequency in [rad/s].

To find the parameters of this controller, the closed-loop transfer function is firstly determined considering the delays introduced by PWM process (t_{pwm}/2), the digital controller computation time (t_c), and the analog-to-digital converter conversion time (t_{adc}) for measuring the feedback load current (see Figure F.3). To take into account this conversion time, a delay has been deliberately introduced in the closed loop of the implemented real-time models.

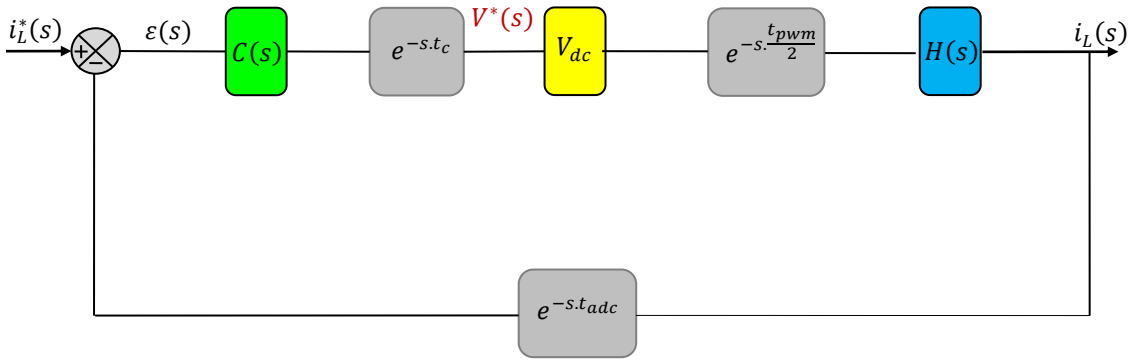


Figure F. 3: Block diagram of the closed-loop system

Where :

$i_L^*(s)$: the reference input current

$C(s)$: the transfer function of the controller

$H(s)$: the transfer function of the AC load

$i_L(s)$: the regulated output

The discretization of the P+Resonant is achieved with Backward method and the sampling period is set to 40µs. The fixed-point format is set to 31Q23 (31 total bit number and 23 bits in the fractional part). The proportional gain is determined to keep the system phase margin equal to 45 degree and the integral gain is selected so as to obtain a (near) zero steady-state error. In our case the proportional and integral coefficients of the discrete-time controller have been set respectively to 9.51 and 2.45.

The treatment is synchronized with the PWM carrier according to Figure F. 4. The voltage reference (from the regulator) is refreshed at each PWM vertex instants. Therefore, taking into account time delays (T=t_{adc}+t_c), the feedback current is sampled each t_{pwm}/2-T. It is worth noticing that the delay T is too short that the measured current can be considered as the

instantaneous mean value. The PWM period t_{pwm} has been set to 80µs and t_{adc}=1µs. As for the computation time t_c, it is equal to 0.15µs with a 100MHz system clock.

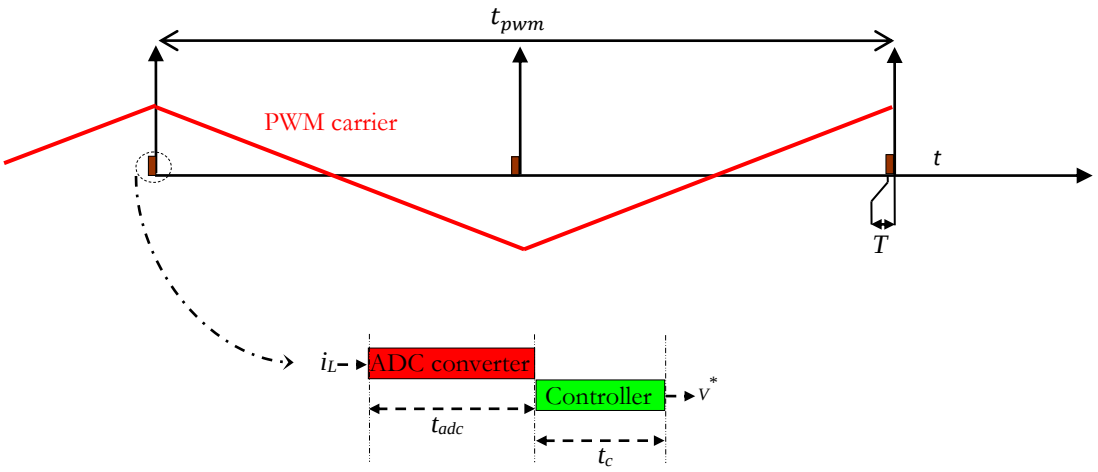


Figure F. 4: Delays caused by the analog-to-digital conversion, the PWM process and the digital controller

As for hardware resources occupation, the controller uses 4 (6.89%) DSP48A1s units (for multiplications), 213 (0.78%) LUTs and 403 (0.73%) Flip-Flops.

Bibliography

[ALT] Altera online documentation available on:http://www.altera.com

[ALI06] D.C. Aliprantis, S.D. Sudhoff, and B.T. Khun,“ A brushless exciter model incorporating multiple rectifier modes and preisach’s hysteresis theory,” *IEEE Transaction On Energy Conversion*, vol. 21, no. 1 :pp. 136–147, 2006.

[ABD09] A. Abdulatif, M. Shaban, “A Matlab / Simulink Based Tool for Power Electronic Circuits,” *in proceeding of world academy of science, engineering and Technology*, vol. 49, pp. 274-278, 2009.

[BEL07] J. Bélanger, V. Lapointe, C. Dufour, L. Schoen, “eMEGAsim: An Open High-Performance Distributed Real-Time Power Grid Simulator,” *in proceedings ICPS conference*, Bangalore, India, 2007.

[BEN10] A. K. Ben Salem, S. B. Othman, S. Ben Saoud,“Field Programmable Gate Array - Based System-on-Chip for Real-Time Power Process Control,” *American Journal of Applied Sciences*, vol.7, no.1, pp 127-139, 2010.

[BEL10] J. Belanger, L. Snider, G. Nanjundaiah,“Today’s Power System Simulation Challenge: High-performance, Scalable, Upgradable, Affordable COTS-Based Real-Time Digital Simulators,” *in proceeding PEDES conference*, New Delh, 2010.

[BEL-10] J. Bélanger, P. Venne, J.-N. Paquin,“The What, Where and Why of Real-Time Simulation,” OPAL-RT Technologies, www.opal-rt.com/technical-document/what-where-and-why-real-time-simulation, October 2010.

[BAR11] A. Barakat, S. Tnani, G. Champenois, and E. Mouni, “Monovariable and multivariable voltage regulator design for a synchronous generator modeled with fixed and variable loads”. *IEEE Transaction On Energy Conversion*, vol 26, no. 3, 2011.

[BAH11] I. Bahri, “SW/HW Co-Design methodology for AC Drive applications”, *PhD Thesis*, UCP, France-Tunisia, 20011.

[BER12] D.R. Bertenshaw, A.C. Smith, C.W.HO, T. Chan, M. Sasic, “ Detection of stator core faults in large electrical machines,” *IET Power Electronics*., vol. 6, no. 6, pp. 295 - 301, Jul. 2012.

[BAC13] T. Ould Bachir, C. Dufour, J. Bélanger, J. Mahseredjian, J.P. David, "A fully automated reconfigurable calculation engine dedicated to the real-time simulation of high switching frequency power electronic circuits," *Mathematics and Computers in Simulation*, vol. 91, pp. 167-177, May 2013.

[BAB14] A.S. Babel, E.G.Strangas, “Condition-based monitoring and prognostic health management of electric machine stator winding insulation,” *in proceeding ICEM*, Berlin , Germany, 2014.

[CHA04] R. Champagne, L. A. Dessaint, H. Fortin-Blanchette and G. Sybille, “Analysis and Validation of a Real-Time AC Drive Simulator,” *IEEE Transactions On Power Electronics*, vol. 19, pp. 336-345, 2004.

[CHA08] L. Charaabi, E.Monmasson, I.Slama-Belkhodja,“FPGA-based real-time emulation of induction motor using fixed point representation,” *in proceeding IECON conference*, 2009.

[COR09] P. Cortés, G. Ortiz, J.I. Yuz, J.Rodríguez, S. Vazquez, and L.G. Franquelo,“ Model Predictive Control of an Inverter With Output LC Filter for UPS Applications ,” *IEEE Transaction On Industrial Electronics* , vol. 56, no. 6, June. 2009.

[CAL13] T. Caldognetto, S. Buso, P. Mattavelli, “Digital Controller Development Methodology Based on Real-Time Simulations with LabVIEW FPGA Hardware-Software

Toolset," *Journal of Computational Intelligence and Electronic Systems*, vol.17, no.2, pp.110-117, December, 2013.

[CHE12] Y. Chen, V. Dinavahi, "Digital Hardware Emulation of Universal Machine and Universal Line Models for Real-Time Electromagnetic Transient Simulation," *IEEE Transactions On Industrial Electronics*, vol. 59, no. 2, pp. 1300-1309, February 2012.

[CHE13] Y. Chen and V. Dinavahi," Multi-FPGA Digital Hardware Design for Detailed Large-Scale Real-Time Electromagnetic Transient Simulation of Power Systems," IEEE IET, vol. 7, no. 5, pp. 451–463, Juin. 2013.

[DOM69] Dommel, H.W, "Digital computer solution of electromagnetic transients in single- and multi- phase networks," *IEEE Transaction On Power Apparatus and Systems*, vol. 88, no.4, pp. 388–399. 1969.

[DIN00] V. Dinavahi, "Real-time digital simulation of switching power circuits," Ph.D. dissertation, Univ. Toronto, Toronto, ON, Canada, 2000.

[DEH05] A. Dehkordi, A. M. Gole, and T. L. Maguire, "Permanent magnet synchronous machine model for real-time simulation," *in proceeding IPST conference*, Montreal, QC, Canada, June. 2005.

[DUF06] C. Dufour, S. Abourida, J. Bélanger, V. Lapointe, "Real-Time Simulation of Permanent Magnet Motor Drive on FPGA Chip for High-Bandwidth Controller Tests and Validation*," in proceedings ISIE conference*, Montreal, Que, July 2006.

[DUF07] C. Dufour, J. Bélanger, S. Abourida , V. Lapointe , "FPGA-Based Real-Time Simulation of Finite-Element Analysis Permanent Magnet Synchronous Machine Drives," *in proceedings PESC conference*, Orlando, FL, 2007.

[DUF07EPE] C. Dufour, J. Bélanger, S. Abourida, V. Lapointe, "Real-Time Simulation of Finite-Element Analysis Permanent Magnet Synchronous Machine Drives on a FPGA card," *in proceedings EPE conference*, Aalborg, September, 2007.

[DUF07PESC] C. Dufour, J. Belanger, S. Abourida, V. Lapointe, "FPGA-Based Real- Time Simulation of Finite-Element Analysis Permanent Magnet Synchronous Machine Drives," *in proceeding PESC conference*, pp. 944 – 950, 2007.

[DEL11] D. U. C. Delgado and D. R. E. Trejo, "An Observer-based Diagnosis Scheme for Single and Simultaneous Open-switch Faults in Induction Motor Drives", *IEEE Transaction on Power Electronics*, vol. 58, no. 2, pp. 671 - 679, 2011.

[DAGI11] M. Dagbagi, L. Idkhajine, E. Monmasson, L. Charaabi, and I. Slama-Belkhodja: "FPGA Implementation of a Synchronous Motor Real-Time Emulator based on Delta Operator". *In Proceeding ISIE'2011 Conference*, pp. 1581-1586, Gdańesk, Poland. 2011.

[DAGS11] M. Dagbagi, L. Charaabi, L. Idkhajine, E. Monmasson, and I. Slama-Belkhodja: "Digital Implementation using Delta Operator for FPGA based Induction Motor Emulator". *In Proceeding, SSD'2011 Conference*, pp. 1-6, Sousse, Tunisia. 2011.

[DAG12] M. Dagbagi, L. Idkhajine, E. Monmasson, and I. Slama-Belkhodja: "FPGA Implementation of Power Electronic Converter Real-Time Model". *In Proceeding SPEEDAM'2012 Conference*, pp. 658-663, Sorrento, Italy.

[DEB12] M. Debbou, M. Abdellatif, M. Pietrzak-David, "Analytical redundancy for service continuity of Doubly Fed Induction Machine speed drive," *in proceeding ICEM*, pp. 858-864, Marseille, France, September .2012.

[DAGS12] M. Dagbagi, L. Idkhajine, E. Monmasson, and I. Slama-Belkhodja: "FPGA Implementation of Power Electronic Converter Real-Time Model". *In Proceeding SPEEDAM'2012 Conference*, pp. 658-663, Sorrento, Italy. 2012.

[DAGP13] M. Dagbagi, L. Idkhajine, M.Tientcheu-Yamdeu, E. Monmasson, and I. Slama-Belkhodja: "FPGA-based Brushless Synchronous Generator Real-Time Emulator Prototype for Digital Controllers Testing". *In Proceeding PCIM'2013 Conference*, pp. 1707-1714, Nuremberg, Germany. 2013.

[DAGI13] M. Dagbagi, A. Hemdani, L. Idkhajine, M. W. Naouar, E. Monmasson, and I. Slama-Belkhodja: "FPGA-based Real-Time Hardware-In-the-Loop validation of a 3-phase PWM Rectifier controller". *In Proceeding IECON'2013 Conference*, pp. 5374 - 5379, Vienna, Austria. 2013.

[DAGJ13] M. Dagbagi: "Implantation sur une cible FPGA d'un simulateur temps réel d'un redresseur MLI triphasé". *In Proceeding SEEDS-JCGE'2013 Conference*, Saint Nazaire, France. 2013.

[DAGP15] M. Dagbagi, L. Idkhajine, E. Monmasson: " Simulation temps-réel des systèmes électriques – Implantation sur cible FPGA du simulateur temps réel d'un redresseur MLI triphasé". *Revue 3EI*, n° 80, pp. 36-42, Avril 2015.

[DAGI15] A. Hemdani, M. Dagbagi, M. W. Naouar, L. Idkhajine, I. Slama-Belkhodja, and E. Monmasson: "Indirect Sliding Mode Power Control for Three Phase Grid Connected Power Converter". *IET Power Electronics Journal*, vol. 8, No. 6, pp. 977-985, June 2015.

[DAGT15] M. Dagbagi, A. Hemdani, L. Idkhajine, M. W. Naouar, E. Monmasson, and I. Slama-Belkhodja: "ADC-based Embedded Real-Time Simulator of a Power Converter Implemented in a Low Cost FPGA: Application to a Fault-Tolerant Control of a Grid-Connected Voltage Source Rectifier". *IEEE Transaction on Industrial Electronics. (Accepted).*

[GOL84] A. M. Gole, R. W. Menzies, H. M. Turanli, and D. A.Woodford, "Improved Interfacing of Electrical Machine Models to Electromagnetic Transients Programs," *IEEE Transaction On Power Apparatus and Systems*, vol. PAS-103, no. 9, pp. 2446-2451,1984.

[GAN93] Gang. L, Gevers. M,"Comparative Study of Finite Wordlength Effects in Shift and Delta Operator Parameterizations," *IEEE Transaction On Automatic Control*, vol. 38, no. 5, pp.803-807, 1993.

[GAN93_2] Gang. L, Gevers. M,"Roundoff noise minimization using delta-operator realizations," *IEEE Transaction On Signal Processing*, vol. 41, no. 2, pp.629-637, 1993.

[GRA99] T. Grandpierre, C Lavrenne and Y. Sorel, "Optimized rapid prototyping for real-time embedded heterogeneous multiprocessor," *in Proc. CODES'99 7th International Workshop on Hardware/ Software Co-Design Conf*., pp. 74-78.

[GON14] O. Goni, A. Sanchez, E. Todorovich, A. de Castro, "Resolution analysis of switching converter models for hardware-in-the-loop", IEEE Transaction on Industrial Informatics, vol. 10, no. 1, pp. 1162-1170, Mai, 2014.

[HUI94] S. Y. R. Hui and S. Morrall, "Generalised Associated Discrete Circuit Model for Switching Devices," *in proceeding IEE Science*, Measurement and Technology, vol. 141, no. 1, pp. 57–64, 1994.

[HAS14] A. Hasanzadeh, C.S. Edrington, N. Stroupe, T. Bevis, "Real-Time Emulation of a High-Speed Microturbine Permanent-Magnet Synchronous Generator Using Multiplatform Hardware-in-the- Loop Realization," *IEEE Transactions On Industrial Electronics*, vol. 61, no. 6, pp.3109-3118, June 2014.

[HEM15] A. Hemdani , " Intégration d'une charge intelligente à base d'un redresseur MLI triphasé dans un micro-réseau", PhD Thesis, ENIT, Tunisia, 2015.

[IDK10] L.Idkhajine," Fully FPGA-based Sensorless Control for Synchronous AC Drive using an Extended Kalman Filter,"PhD Thesis, Cergy Pontoise University, 2010.

[JIN97] H. Jin, "Behavior-mode Simulation of Power Electronic Circuits," *IEEE Transaction On Power Electronics*, vol. 12, no. 3, pp. 443-452, 1997.

[JIA11] H. Jiabing, S. Lei, H. Yikang and Z. Q. Zhu "Direct active and reactive power regulation of grid connected DC/AC converters using sliding mode control approach," *IEEE Transactions on Power Electronics*, vol.26, pp. 210-222, January 2011.

[JIA14] L. Jiadai, V. Dinavahi, "A Real-Time Nonlinear Hysteretic Power Transformer Transient Model on FPGA," *IEEE Transaction On Industrial Electronics*, vol. 61, no. 7, pp.3587-3597, Jul.2014.

[JIM15] O. Jimenez et al., "Implementation of an FPGA-based on-line hardware-in the-loop emulator using high-level synthesis tools for resonant power converters applied to induction heating appliances", *IEEE Transaction on Industrial Electronics*, vol. 62, no. 4, pp. 2206-2214, April, 2015.

[KAU97] Kauraniemi. J, Laakso. T. I, "Roundoff noise analysis of modified Delta operator direct form structures," *IEEE Transaction On Circuits and Systems*, vol. 4, pp.2365-2368, 1997.

[KAR09] S. Karimi,P.Poure, andS.Saadate, "Fast power switch failure detection for fault tolerant voltage source inverters using FPGA," *IET Power Electronics*, vol. 2, no. 4, pp. 346–354, Jul. 2009.

[KON09] W.Y.Kong, D. G. Holmes, B. P. McGrath, "Improved Stationary Frame AC Current Regulation using Feed forward Compensation of the Load EMF," *in proceeding APEC2009 conference*, Washington, DC, 2009.

[KIF11] A. Kiffe, S. Geng, T. Schulte, J. Maas, "Real-Time Simulation of Power Electronic Circuits based on Discrete Averaging Method," *in proceeding IECON conference*, Melbourne, VIC, 2011.

[KAM13] M. Kaminski, T. Orlowska-Kowalska, "FPGA Implementation of ADALINE-Based Speed Controller in a Two-Mass System," *IEEE Transactions On Industrial Informatics*, vol. 9, no. 3, pp. 1301-1311, August 2013.

[LI93] G. Li, M. Gevers, " Roundoff Noise Minimisation Using Delta Operator Realisations," *IEEE Transaction On Signal Processing*, vol. 42, no, February 1993.

[LI20] LI. G, Wu. J, Chen. S, Chu. J, "Optimal finite precision state- Estimate feecback controller realizations of discretre-time systems," *IEEE Transaction On Automatic Control*, vol. 45, no. 8, pp.1550-1554, 2000.

[LUC11] O. Lucía, I. Urriza, L.A. Barragán, D. Navarro, O. Jiménez, J.M. Burdío, "Real-Time FPGA-Based Hardware-in-the-Loop Simulation Test Bench Applied to Multiple-Output Power Converters," *IEEE Transactions On Industry Applications*, vol. 47, no. 2, pp. 853-860, March-April 2011.

[MAR89] J. R. Marti and J. Lin, "Suppression of numerical oscillations in the EMTP," IEEE Trans. Power Syst., vol. 9, no. 1, pp. 71–72, Feb. 1989.

[MID90] H. Middleton and G. C. Goodwin, Digital Control and Estimation: A Unified Approach. Englewood Cliffs, NJ: Prentice- Hall, 1990.

[MAR97] J. R. Marti and K. W. Louie, "A phase-domain synchronous generator model including saturation effects," IEEE Transaction On Power Systems, vol. 12, no. 1, pp. 222–229, Feb. 1997.

[MAR06] C.C. Marouchos," The Switching Function: Analysis of Power Electronic Circuits," IET. 2006. 297 pages.

[MAT09] M. Matar, R. Iravani, "FPGA implementation of a modified two- layer network equivalent for real-time simulation of electromagnetic transients," *in proceeding IPST conference*, Kyoto, Japan, Jun. 2009.

[MAT10] M. Matar, R. Iravani, "FPGA Implementation of the Power Electronic Converter Model for Real-Time Simulation of Electromagnetic Transients," *IEEE Transaction On Power Delivery*, vol. 25, no. 2, pp.852-860, April 2010.

[MEN10] P.M.Menghal, A. Jaya Laxmi, "Real Time Control of Electrical Machine Drives: A Review," *in proceeding ICPCES conference*, Allahabad, 2010.

[MAJ11] D. Majstorovic, I. Celanovic, N.D. Teslic, N. Celanovic, V.A. Katic, "Ultralow-Latency Hardware-in-the-Loop Platform for Rapid Validation of Power Electronics Designs," *IEEE Transactions On Industrial Electronics* , vol. 58, no. 10, pp. 4708-4716, October 2011.

[MAT11] M. Matar, R. Iravani "Massively Parallel Implementation of AC Machine Models for FPGA-Based Real-Time Simulation of Electromagnetic Transients", *IEEE Transaction On Power Delivery*, vol. 26, no. 2, April 2011.

[MYA11] A. Myaing, V. Dinavahi, "FPGA-Based Real-Time Emulation of Power Electronic Systems With Detailed Representation of Device Characteristics," *IEEE Transaction On Industrial Electronics*, vol. 58, no. 1, pp. 358- 368, January 2011.

[MAT11IPST] M. Matar, M. Saeedifard, A. Etemadi, R. Iravani, "An FPGA-Based Hardware-in-the-Loop Simulator for Multilevel Converter Systems," *in proceeding IPST conference*, in Delft, The Netherlands, June, 2011.

[MON11] E. Monmasson, L. Idkhajine, M. N. Cirstea, I. Bahri, A.Tisan, M. W. Naouar, "FPGAs in industrial control applications," *IEEE Transaction On Industrial Informatics*, vol.7, no.2, pp.224-243, May 2011.

[MON12] E. Monmasson, I.Bahri, L. Idkhajine, A. Maalouf, W. M. Naouar," Recent Advancements in FPGA-based controllers for AC Drives Applications," *in proceeding OPTIM conference*, Brasov, May, 2012.

[MAR12] S. Mariethoz, A. Domahidi, M. Morari,"High-bandwidth explicit model predictive control of electrical drives," *IEEE Transactions on Industry Applications*, vol. 48, no. 6, pp.1980-1992, 2012.

[MAT13] M. Matar, R. Iravani, "The Reconfigurable-Hardware Real-Time and Faster-Than-Real-Time Simulator for the Analysis of Electromagnetic Transients in Power Systems," *IEEE Transaction On Power Delivery*, vol. 28, n° 2, pp. 619-627, April 2013.

[NAO07] M. W. Naouar, E. Monmasson, A. Naassani, I. Slama-Belkhodja, N. Patin, "FPGA-based current controllers for AC machine drives– A review," *IEEE Transactions On Industrial Electronics*, vol. 54, no. 4, pp. 1907-1925, August 2007.

[NAO07P] M. W. Naouar, "Commande numérique à base de composants FPGA d'une machine synchrone", *PhD Thesis*, UCP-ENIT, France-Tunisia, 2007.

[NOD11] T. Noda, S. Filizadeh, A. R. Chevrefils, M. Matar, R. Iravani, C. Dufour, J. Belanger, M. O. Faruque,“ Interfacing Issues in Real-Time Digital Simulators ,” *IEEE Transaction On Power Delivery*, vol . 26, no. 2, April 2011.

[NUS14] P. Nussbaumer, M.A. Vogelsberger, T. Wolbank, “Induction Machine Insulation Health State Monitoring Based on Online Switching Transient Exploitation,” *IEEE Transaction On Industrial Electronics* , vol. pp, no. 29, October, 2014.

[OPA] OPAL-RT website on:http://www.opal-rt.com

[ORI06] R. Orizondo, R. Alves, “UPFC Simulation and Control Using the ATP/EMTP and MATLAB/Simulink Programs,” *in proceeding PES conference*, Latin America, Venezuela, 2006.

[OUL13] T. Ould Bachir, C. Dufour, J. Bélanger, J. Mahseredjian, J.P. David, "A fully automated reconfigurable calculation engine dedicated to the real-time simulation of high switching frequency power electronic circuits," *Mathematics and Computers in Simulation*, vol. 91, pp. 167-177, May 2013.

[PEJ94] P. Pejovic and D. Maksimovic, "A Method for Fast Time-Domain Simulation of Networks with Switches," *IEEE Transactions on Power Electronics*, vol. 9, no. 4, pp. 449–456, 1994.

[PEL12] P. Peltoniemi, P. Nuutinen, J. Pyrhonen , “Observer-Based Output Voltage Control for DC Power Distribution Purposes,” *IEEE Transactions On Power Electronics*, vol. 28, no.4, pp.1914-1926, 2012.

[QUN11] AN Quntao, SUN Lizhi, ZHAO Ke, “Switching function model-based fast-diagnostic method of open-switch faults in inverters without sensors”, *IEEE Transaction on Power Electronics*, vol. 26, no. 1, pp. 119- 126, 2011.

[QUE12] L. Quéval, H. Ohsaki, “Study on the implementation of the phase-domain model for rotating electrical machines,” 15th International Conference on Electrical Machines and Systems (ICEMS2012), Sapporo, Japan, October 2012.

[RTD] RTDS website on:http://www.rtds.com

[RIC02] F. Ricci and H. Le-Huy, “An FPGA-based rapid prototyping platform for variable-speed drives,” *in Proc. IEEE IECON’02 Conf.*, 2002, pp. 1156–1161.

[RAJ05] A.D. Rajapakse, A.M. Gole, P.L. Wilson, “Electromagnetic Transients Simulation Models for Accurate Representation of Switching Losses and Thermal Performance in Power Electronic Systems,” *IEEE Transaction On Power Delivery*, vol. 20, n° 1, pp. 319-327, January 2005.

[RAK11] M. Rakotozafy, P. Poure, S. Saadate, C. Bordas, L. Leclere, " Real-time digital simulation of power electronics systems with Neutral Point Piloted multilevel inverter using FPGA, " *Electric Power Systems Research*, vol. 81, no. 2, pp. 687–698, 2011.

[REZ13] Mohammad Reza, “An Improved State Space Average Model of Buck DC-DC Converter with all of the System Uncertainties”, in International Journal on Electrical Engineering and Informatics, vol 5, no 1, March 2013.

[RAJ13] S. Rajendran, U. Govindarajan, S. Senthilvadivelu, S. B. Uandai, “Intelligent sensor fault-tolerant control for variable speed wind electrical systems ,” *IET Power Electronics*, vol. 6, no. 7, pp. 1308 - 1319, August. 2013.

[RAK14] A.M. Rakotozafy,“ Simulation Temps réel de dispositifs électrotrchniques ,”PhD Thesis, Lorraine University, 2014.

[SAL94] L. Salazar, G. Joos, " PSPICE Simulation of Three-Phase Inverters by Means of Switching Functions, " *IEEE Transaction On Power Electronics*, vol. 9, no. 1, pp. 35-42, 1994.

[SIM09] SimPowerSystems, User's Guide, Version 5, The MathWorks Inc., 2009.

[SEU12] T. C. Seung, W. Qiuwei , A.H Nielsen, J. Ostergaard, I.K Park , "Real-Time Hardware-in-the-Loop (HIL) Testing for Power Electronics Controllers," *in proceeding APPEEC conference*, Shanghai, March 2012.

[SAN12] A. Sanchez, A. de Castro, and J. Garrido, "A comparison of simulation and hardware-in-the- loop alternatives for digital control of power converters, " *IEEE Transaction On Industrial Informatics*, vol. 8, no. 3, pp. 491 –500, Aug. 2012.

[SHA13] M. Shahbazi, P. Poure, S. Saadate, ; M.R. Zolghadri "FPGA Based Fast Detection With Reduced Sensor Count for a Fault- Tolerant Three-Phase Converter," *IEEE Transactions On Industrial Informatics*, vol. 9, no. 3, pp.1343-1350, August 2013.

[SHAH13] M. Shahbazi, P. Poure, S. Saadate, M.R. Zolghadri, "FPGA Based Reconfigurable Control for Fault-Tolerant Back-to-Back Converter Without Redundancy," *IEEE Transactions On Industrial Electronics* , vol. 60, no. 8, pp. 3360-3371, August 2013.

[SUT13] T. Sutikno, N.R.N. Idris, A. Jidin, M.N. Cirstea, "An Improved FPGA Implementation of Direct Torque Control for Induction Machines," *IEEE Transactions On Industrial Informatics*, vol. 9, no. 3, pp. 1280-1290, August 2013.

[SON14] Y. Song, L. Chen, Y. Chen, and S. Huang, "A General Parameter Configuration Algorithm for Associate Discrete Circuit Switch Model," *In Proceedings POWERCON'2014 Conference*, Chengdu, China.

[SCH14] A. Schmitt, J. Richter, U. Jurkewitz, M. Braun, "FPGA-based real time simulation of nonlinear permanent magnet synchronous machines for power hardware-in-the-loop emulation systems", in proceeding IECON '14 conference, 2014.

[WAN10] L. Wang et al., "Methods of interfacing rotating machine models in transient simulation programs," *IEEE Transaction On Power Delivery*, vol. 25, no. 2, pp. 891-903, Apr. 2010.

[XIL] Xilinx online documentation available on:http://www.xilinx.com

[YIN13] Ying Zhu, Ming Cheng, Wei Hua, Bangfu Zhang," Sensorless Control Strategy of Electrical Variable Transmission Machines for Wind Energy Conversion Systems," *IEEE Transactions On Magnetics*, vol. 49, no.7, pp. 3383 - 3386, 2013.

[ZHA05] T. Zhao, Q. Wang, "Application of MATLAB/SIMULINK and PSPICE Simulation in Teaching Power Electronics and Electric Drive System," *in proceedings of the Eighth International Conference on Electrical Machines and Systems*, pp 2037–2041, Nanjing, China. 2005.

[ZOE14] C. Zoeller, M. Vogelsberger, P. Nussbaumer, T. Wolbank , "Insulation monitoring of three phase inverter-fed AC machines based on two current sensors only," *in proceeding ICEM*, Berlin , Germany, 2014.