UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMMANDE PRÉDICTIVE D'UN ONDULEUR DE TENSION SUR FPGA

MÉMOIRE PRÉSENTÉ
COMME EXIGENCE PARTIELLE DE LA

MAÎTRISE EN GÉNIE ÉLECTRIQUE

PAR
ACHILLE BLAISE TCHOUZIN MBATKAM

OCTOBRE 2022

UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

MAÎTRISE EN GÉNIE ÉLECTRIQUE (M. Sc. A.)

Direction de recherche :

Daniel Massicotte, directeur de recherche	Université du Québec à Trois-Rivières

Jury d'évaluation

Loic Michel	Évaluateur externe
Alben Cardenas Gonzalez	Université du Québec à Trois-Rivières

Résumé

Le modèle de contrôle prédictif des convertisseurs de puissance a été un sujet important de recherche ces dernières années avec l'augmentation rapide de l'utilisation des contrôleurs numériques dans les systèmes de contrôle. Cette stratégie de contrôle est considérée comme une méthode avancée de contrôle des systèmes qui a été largement implémentée dans l'industrie. Dans ce projet, un modèle pour simuler un contrôle prédictif appliqué à un onduleur à deux niveaux a été développé et cela s'est effectué en deux étapes. La première étape consistait à développer un modèle Simulink pour tester le contrôle prédictif et analyser ses performances. Des périodes d'échantillonnage de $25\ \mu s$ et $1\ \mu s$ pour l'algorithme de contrôle prédictif ont été testées, dans le but d'étudier de façon préventive l'effet de la fréquence sur la performance du contrôleur. Nous avons pu constater qu'une fréquence d'échantillonnage élevée entraînait une performance nettement meilleure. En seconde étape, l'algorithme de contrôle prédictif a été implémenté sur un réseau de portes programmables (FPGA – *Field-Programmable Gate Array*) pour des fins expérimentales. La librairie Xilinx System Generator (XSG) a été utilisée pour modifier et adapter le modèle de contrôle prédictif Simulink à des blocs XSG pour que ce modèle soit exécuté sur un FPGA. Ce qui a permis de mieux paralléliser le modèle et de réduire de façon considérable la propagation des données. Le contrôleur sera ensuite ajusté dans une configuration expérimentale sur FPGA et pourra se connecter avec un système physique réel. Pour cela, le modèle sera importé dans l'environnement de simulation temps réel RT-Lab de Opal-RT.

Remerciements

Tout d'abord, j'aimerais remercier mon regretté superviseur de projet, Monsieur Pierre Sicard, de m'avoir proposé ce projet intéressant. Ensuite, j'aimerais remercier mon superviseur de projet, Monsieur Daniel Massicotte, d'avoir accepté de m'encadrer dans ce projet. Ce projet a pu être réalisé grâce à l'appui de la Chaire de recherche sur les signaux et l'intelligence des systèmes haute performance appuyés par le FRQNT, CRSNG-RDC, Prompt, Fondation canadienne pour l'innovation, la CMC microsystèmes pour le support en équipements et logiciels, ainsi que les partenaires industriels Opal-RT, Hydro-Québec et Nutaq. J'aimerais aussi remercier Michel Lemaire, étudiant au doctorat, pour ses précieux conseils, son assistance et son partage de connaissances sur la plate-forme Opal-RT tout au long des travaux du projet. Finalement, j'aimerais remercier ma famille, mon épouse Rosine Tchuissa, qui n'a cessé de me soutenir tout au long de ces quatre années, mes enfants, Ashrone Dagang et Eden-Maël Tchouzin, pour leur patience.

Table des matières

Liste des tableaux

Liste des figures

Liste des symboles

CA: Courant alternatif

CC: Courant continu

CPU: Central Processor Unit

DSP: Digital Signal Processor

f.c.é.m: force contre-électromotrice

FPGA: Field-programmable Gate Array

HIL: Hardware-In-the-Loop testing

LQR: Linear-Quadratic Regulator

MPC: Model Predictive Control

PID: Proportionnel-Intégral-Dérivé

XSG: Xilinx System Generator

Chapitre 1 - Introduction

La commande des systèmes est une branche de l'ingénierie qui utilise la théorie du contrôle pour concevoir des systèmes avec le comportement désiré. Pour atteindre le comportement désiré, on utilise les capteurs afin de mesurer des grandeurs physiques nécessaire à l'évaluation des performances de l'asservissement et éventuellement à la définition de nouveaux objectifs de contrôle. L'ingénierie des systèmes de commande est un ensemble d'activités, qui se concentre sur l'implémentation des systèmes de contrôle principalement dérivés de la modélisation mathématique de systèmes d'une gamme diversifiée (mécanique, électrique, fluide, chimique, etc.). Grâce aux avancées de progrès technologiques, cette discipline de l'ingénierie a attiré une attention signifiante au cours du 20^{e} siècle. À l'aide de la modélisation mathématique, elle cherche à comprendre les systèmes physiques, en termes d'entrées, de sorties et de divers composants avec des comportements différents. De plus, elle utilise des outils de conception des systèmes de commande pour implémenter des contrôleurs pour ces systèmes, et intégrer des contrôleurs dans des systèmes physiques utilisant la technologie disponible. De nos jours, un grand nombre d'industries utilise de nombreux systèmes de commande dans des applications de toutes sortes. Le dénominateur commun de tous les types de commandes est de maintenir un résultat désiré qui peut changer au cours du processus. Il existe une gamme variée de stratégies de contrôle telles que le contrôle robuste, le contrôle adaptatif, le contrôle intelligent, le contrôle stochastique, le contrôle prédictif, etc. Le contrôle prédictif ou commande prédictive est une conception qui consiste à prendre en compte, à l'instant présent, le comportement futur, en utilisant explicitement un modèle numérique du système afin de prédire la sortie dans le futur sur un horizon fini. Cependant, il n'existe pas une stratégie unique, mais plutôt tout un ensemble de méthodes de commande prédictive, assez similaires, bâtit autour de principes communs, mais présentant néanmoins quelques différences dans l'interprétation des concepts clés. Une des méthodes de commande prédictive, qui sera mise en œuvre dans ce projet, est la commande prédictive à base de modèle (MPC). Le développement du contrôle prédictif à

base de modèle (MPC) date d'environ trois décennies et est de ce fait considéré comme l'une des avancées les plus importantes dans le contrôle des processus [1]. Le MPC se base sur des modèles dynamiques du processus, généralement des modèles linéaires obtenus par l'identification du système. L'implémentation de cette méthode de contrôle nécessite un nombre élevé de calculs, ce qui a été précédemment un défi pour le développement. Elle était néanmoins plus utilisée dans les processus chimiques, car la constante de temps des modèles est très élevée [2]. Grâce aux avancées technologiques, ce n'est plus un problème de nos jours, car il existe des plates-formes de contrôle qui ont beaucoup de puissance de calcul, ce qui rend ce type de contrôle très intéressant pour bon nombre d'applications. La technologie moderne des circuits intégrés a développé les processeurs de signal numérique (DSP – *Digital Signal Processor*) et les réseaux de portes programmables (FPGA – *Field-Programmable Gate Array*), qui offrent une puissance de calcul substantielle à des prix abordables. Ces développements parallèles offrent une opportunité sans précédent pour l'implémentation pratique de techniques de contrôle avancées. En conséquence, les applications de système de commande sophistiquées utilisant des DSP et FPGA ont augmenté de façon exponentielle ces dernières années. Le MPC a la capacité d'anticiper les événements futurs et peut prendre des mesures de contrôle en conséquence. Les contrôleurs PID et LQR n'ont pas cette capacité prédictive [3, [4]. Le contrôle prédictif présente plusieurs avantages qui le rendent très adapté au contrôle des convertisseurs de puissance : les concepts sont intuitifs et faciles à comprendre ; ils peuvent être appliqués à une variété d'application ou systèmes ; les contraintes et les non-linéarités peuvent être facilement incluses ; les cas multi-variables peuvent être considérés et le contrôleur résultant est facile à mettre en œuvre. Le MPC a été implémenté dans plusieurs applications pratiques, et les résultats ont été très efficaces [5]. Quelques-unes de ces principales applications pratiques où il a récemment été utilisé sont : les systèmes de productions distribués, le filtrage actif et le conditionnement de puissance, les entrainements électriques, les énergies renouvelables non conventionnelles et les alimentations ininterruptibles [5].

Dans ce travail, le contrôle prédictif sera appliqué à une source de tension ondulée à deux niveaux. Cependant, si le seul critère pour le contrôle est le suivi de la consigne de courant, et que l'algorithme fonctionne avec une fréquence d'échantillonnage élevée, on s'attend à ce que la fréquence de commutation de l'onduleur soit très élevée dans cette étude ; ce qui pourra

être difficile pour un circuit physique à suivre cette fréquence de commutation. Avec une fréquence d'échantillonnage très élevée, les prélèvements des données seront plus efficaces ce qui aura pour but d'améliorer la boucle de rétroaction nécessaire à la MPC. Un autre problème concerne les pertes de commutation causées par une fréquence de commutation élevée, l'efficacité étant un facteur important dans la conversion de l'énergie. L'un des avantages avec le MPC est la simplicité d'ajouter plusieurs objectifs de contrôle qui découleront des résultats obtenus selon les objectifs prédéfinis initialement. D'une manière plus explicative, un contrôle MPC peut faire naître de nouveaux objectifs de contrôle comme la réduction des harmoniques, l'amélioration de la fréquence de commutation ou même un système de contrôle en cascade. En plus de la méthode de mise en œuvre, l'utilisation des ressources est un autre aspect important de l'implémentation du contrôle de systèmes. Les implémentations des contrôleurs prédictifs de courant pour les convertisseurs de puissance sont principalement basées sur des DSP et des microprocesseurs [17], [23], [24]. Cependant, les solutions microprogrammables présentent certaines limitations. L'un des inconvénients majeurs est le retard au moment du calcul de l'état de commutation optimal. Du fait de ce retard, l'instant auquel l'état de commutation est appliqué au convertisseur est en retard d'un cycle sur l'instant auquel les variables sont échantillonnées. Cette lacune a été clairement reconnue dans [23]–[27], où certaines modifications sont incluses afin de compenser le retard susmentionné. Pour faire face à ce problème, les implémentations basées sur FPGA représentent une alternative intéressante et efficace par rapport aux solutions logicielles telles que les microprocesseurs ou les DSP. L'implémentation du contrôle du système, basée sur l'utilisation des ressources FPGA est comparée dans [10] et [13] pour le FPGA Xilinx. Dans [10], l'horizon de prévision est pris en compte pour la comparaison de l'utilisation des ressources FPGA. Dans [13], des méthodes d'implémentation semi-parallèle et entièrement en série pour le MPC sont adoptées et l'utilisation des ressources est comparée pour la même application. Les tables de correspondance (LUT – *Lookup Tables*), les bascules et les slices DSP sont les principaux indices de logique numérique considérés pour l'utilisation des ressources dans les articles mentionnés ci-dessus. Le FPGA est un choix pragmatique pour la mise en œuvre du MPC en raison de sa capacité de traitement parallèle [6] – [7][8][9][10][11][12]. Dans [6], une étude comparative entre différentes plates-formes de contrôle numérique pour la mise en œuvre du MPC est présentée, y compris le FPGA. De

plus, une approche différente est adoptée pour la mise en œuvre du système basé sur FPGA. Dans [7], une plate-forme FPGA du système reconfigurable NI-CRIO (*National Instruments-Compact Reconfigurable I/O Modules*) est utilisée pour implémenter le MPC pour un convertisseur « back-to-back ». L'implémentation FPGA du MPC est réalisée à l'aide d'une conception basée sur un modèle (MBD – *Model-Base Design*) via MATLAB/Simulink dans [8] et [9], et le code FPGA a été obtenu à l'aide de la fonctionnalité de codeur du langage de description matérielle (HDL – *Hardware Description Language*) de Simulink. Dans [10], une implémentation FCS-MPC (*Finite control set-model predictive control*) à grande vitesse est présentée avec un algorithme de contrôle codé en C++ à l'aide de la boîte à outils « PROTOIP » et le code HDL a été généré via Xilinx Vivado HLS. Le générateur de système Xilinx (XSG – *Xilinx System Generator*) en tant que simulateur numérique adoptant la plate-forme MBD a été utilisé pour implémenter un contrôleur de courant prédictif basé sur FPGA dans [11]–[12][13][14][15][16]. Un XSG est une plate-forme intégrée avec MATLAB/Simulink et facilite la génération de code HDL pour le système développé.

Ce travail de recherche propose une implémentation basée sur FPGA d'un contrôleur de courant prédictif basé sur la méthode MPC pour un onduleur triphasé à deux niveaux [5]. Les références sont données en courant triphasé délivré au réseau. Ensuite, ils sont transformés en références actuelles dans le plan biphasé $\alpha\beta$ en utilisant la transformée de Clarke . Par conséquent, la fonction de coût contient l'erreur de suivi du courant de référence. Profitant de la nature discrète de l'onduleur, un modèle discret est utilisé pour prédire le comportement du système. La prédiction est effectuée pour chaque état de commutation de l'onduleur (sept pour un onduleur à deux niveau) avec un horizon d'une période d'échantillonnage $(k+1)$. Ensuite, le vecteur d'état de commutation qui minimise la fonction de coût est sélectionné et appliqué à l'onduleur pendant la même période d'échantillonnage. L'évaluation des performances souligne l'importance de définir un modèle précis de l'onduleur.

Le développement de ce travail se fera en 6 chapitres en commençant par une introduction du sujet. Au chapitre 2, on présentera la topologie de convertisseur utilisé (onduleur triphasé) et également de la charge. Cette présentation représente le centre du travail sur lequel il faudra appliquer l'objectif de contrôle car il varie en fonction de chaque topologie d'onduleur. Au chapitre 3, le concept de contrôle prédictif est présenté, et un modèle mathématique du circuit de puissance contenant l'algorithme de contrôle de l'onduleur, de la charge et de la prévision

est défini. Le chapitre 4 implémente un modèle Simulink du système obtenu à partir du modèle mathématique et illustre les résultats de la simulation du dit modèle. Le logiciel RT-Lab de Opal-RT et la librairie XSG, qui seront utilisés pour ajuster le modèle dans une configuration expérimentale, sont introduits au chapitre 5. La mise en œuvre se fera en deux étapes : une première étape sera effectuée sous environnement Matlab/Simulink et la seconde sera effectuée par implémentation sous FPGA à partir des données obtenues précédemment à la première étape. Toute cette mise en œuvre est présentée au chapitre 5 avec une combinaison des blocks Simulink et ceux de Xilinx. Les résultats de la simulation avec ce modèle Xilinx, combinés avec le matériel simulé, sont également présentés dans ce chapitre. Enfin, le modèle est ajusté à l'environnement RT-Lab et sera configuré pour être testé avec le matériel RT-Lab sous forme d'émulateurs. Ce processus est décrit au chapitre 6. Le chapitre 7, quant à lui, contient la conclusion et les travaux ultérieurs. Il est également à préciser qu'aucune implémentation physique n'a été réalisée faute de mise au point d'un banc d'essais respectant les normes de sécurité. Cependant, l'algorithme sera implémenté sur un simulateur pouvant recevoir toutes les commandes nécessaires à l'émulation d'un onduleur. Cet émulateur est intégré au RT-LAB et présente les comportements d'un onduleur réel.

Chapitre 2 - Circuit de l'onduleur

2.1 Circuit de puissance de l'onduleur

Le circuit de puissance de l'onduleur, qui sera étudié dans le cadre de ce projet, est composé de trois bras. Il est représenté à la Figure 2.1. L'alimentation électrique est fournie par une source de courant continu (CC), avec une tension V_{dc}. Cette source CC est par la suite transformée en courant alternatif (CA) en contrôlant le flux de courant à travers les interrupteurs de commutation $S1 - S6$.

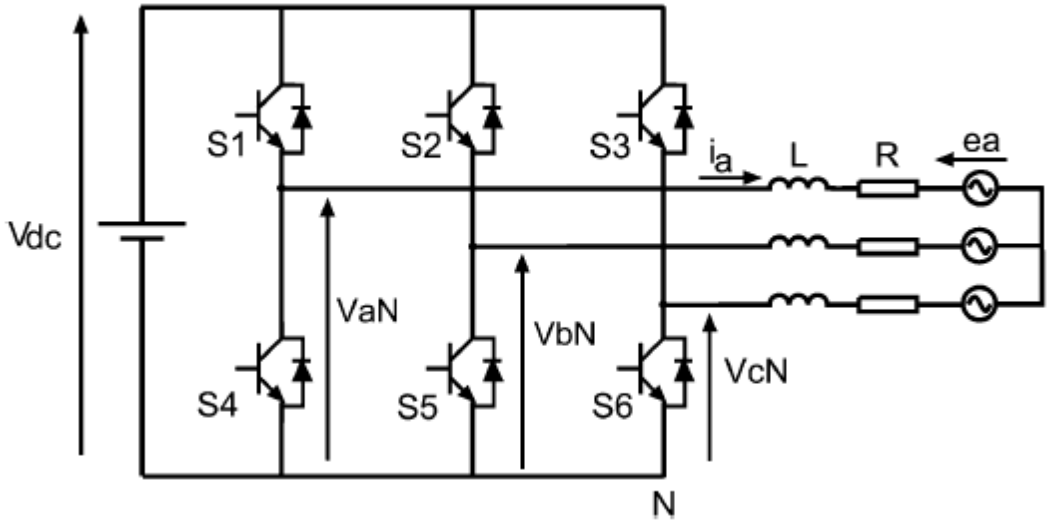


Figure 2.1 Circuit de puissance de l'onduleur [17]

Chaque bras de l'onduleur est couplé à l'une des trois phases de la charge en sortie. Les deux commutateurs de chaque bras fonctionnent en mode complémentaire, afin d'éviter de court-circuiter la source CC. Donc, les états de commutation de chaque bras peuvent alors être représentés par les trois signaux de commutation définis dans les équations (1) – (3) [5].

$$S_a = \begin{cases} 1, S_1 \; conduit \; et \; S_4 \; bloqué \\ 0, S_1 \; bloqué \; et \; S_4 \; conduit \end{cases} \quad (1)$$

$$S_b = \begin{cases} 1, S_2\ conduit\ et\ S_5\ bloqué \\ 0, S_2\ bloqué\ et\ S_5\ conduit \end{cases} \quad (2)$$

$$S_c = \begin{cases} 1, S_3\ conduit\ et\ S_6\ bloqué \\ 0, S_3\ bloqué\ et\ S_6\ conduit \end{cases} \quad (3)$$

Ces équations précédentes peuvent être exprimées sous forme vectorielle comme suit [5] :

$$\boldsymbol{S} = \frac{2}{3}\left(S_a + \boldsymbol{a}S_b + \boldsymbol{a}^2 S_c\right) \quad (4)$$

où

$$\boldsymbol{a} = e^{j2\pi/3} = -\frac{1}{2} + j\frac{\sqrt{3}}{2} \quad (5)$$

avec $\boldsymbol{a}$ le vecteur de tension unitaire dans l'équation précédente. Il représente le déplacement de phase de $120°$ entre les différentes phases. Étant donnée la tension source CC, V_{dc}, les tensions en sortie qui représentent les tensions phases-neutres sont déterminées à l'aide des équations 6, 7, 8 [5] ci-dessous:

$$V_{aN} = S_a V_{dc} \quad (6)$$

$$V_{bN} = S_b V_{dc} \quad (7)$$

$$V_{cN} = S_c V_{dc} \quad (8)$$

Il s'agit ici des tensions phase-neutre, où N est le point neutre, comme indiqué dans la Figure 2.1. Les différentes configurations de la charge triphasée seront créées par les différents états de commutation. Le vecteur de tension de la charge en sortie, créé par l'onduleur, peut être décrit par l'équation suivante :

$$\boldsymbol{v} = \frac{2}{3}\left(v_{aN} + \boldsymbol{a}v_{bN} + \boldsymbol{a}^2 v_{cN}\right) \quad (9)$$

Étant donné que les commutateurs de chaque bras fonctionnent de façon complémentaire, en évaluant toutes les combinaisons possibles d'états de commutation, il en ressort huit combinaisons valides. Les vecteurs de tension générés par ces états ou combinaisons valides sont présentés dans le Tableau 2-1

Vecteur de tension V	Sa	Sb	Sc	Valeur des vecteurs de tension
$\boldsymbol{V_0}$	0	0	0	0
$\boldsymbol{V_1}$	1	0	0	$\frac{2}{3}\,V_{dc}$
$\boldsymbol{V_2}$	1	1	0	$\frac{1}{3}\,V_{dc}+\frac{\sqrt{3}}{3}\,V_{dc}$
$\boldsymbol{V_3}$	0	1	0	$-\frac{1}{3}\,V_{dc}+\frac{\sqrt{3}}{3}\,V_{dc}$
$\boldsymbol{V_4}$	0	1	1	$-\frac{2}{3}\,V_{dc}$
$\boldsymbol{V_5}$	0	0	1	$-\frac{1}{3}\,V_{dc}-\frac{\sqrt{3}}{3}V_{dc}$
$\boldsymbol{V_6}$	1	0	1	$\frac{1}{3}\,V_{dc}-\frac{\sqrt{3}}{3}\,V_{dc}$
$\boldsymbol{V_7}$	1	1	1	0

Tableau 2-1 État de commutation et vecteurs de tension

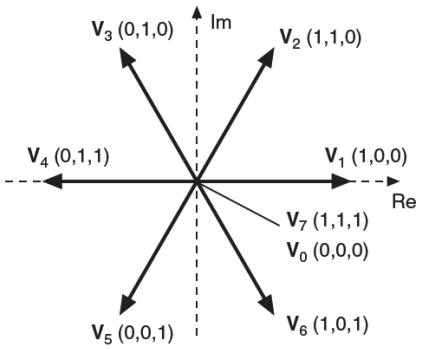


Figure 2.2 Vecteurs de tension générés par l'onduleur [17]

Les états de commutation obtenus de chaque vecteur de tension sont indiqués à la Figure 2.2. On peut constater sur cette figure que les vecteurs $\boldsymbol{V_0}$ et $\boldsymbol{V_7}$ ont la même valeur nulle. Cela donne donc un ensemble fini de sept vecteurs de tensions différents. Il convient ici de

mentionner qu'un modèle plus précis du convertisseur de puissance pourrait être utilisé pour des fréquences de commutation plus élevées. Quelques exemples de sujets d'intérêt pourraient être : modélisation du temps mort, la chute de tension dans les diodes ou la tension de saturation IGBT. Cependant, dans le but de garder le modèle de base aussi simple et de se concentrer sur la stratégie de contrôle, ces sujets n'ont pas été traités dans le cadre de ce travail, mais pourront être traités parmi les suggestions à venir.

2.2 Conclusion

Dans ce chapitre, le modèle d'un onduleur a été présenté ainsi que son mode de fonctionnement. Cette étude constituera une base pour appliquer notre MPC.

Chapitre 3 - Contrôleur prédictif

3.1 Introduction à la commande prédictive

Le contrôle prédictif se sert du modèle d'un système pour prédire le comportement futur dudit système. Un critère d'optimisation est défini, et le choix optimal est ensuite décidé en fonction de ce critère et des prévisions réalisées. La commande prédictive est un terme général qui englobe un ensemble de méthodes différentes. Une classification de différentes méthodes de contrôle prédictives proposées dans [5] est présentée à la Figure 3.1.

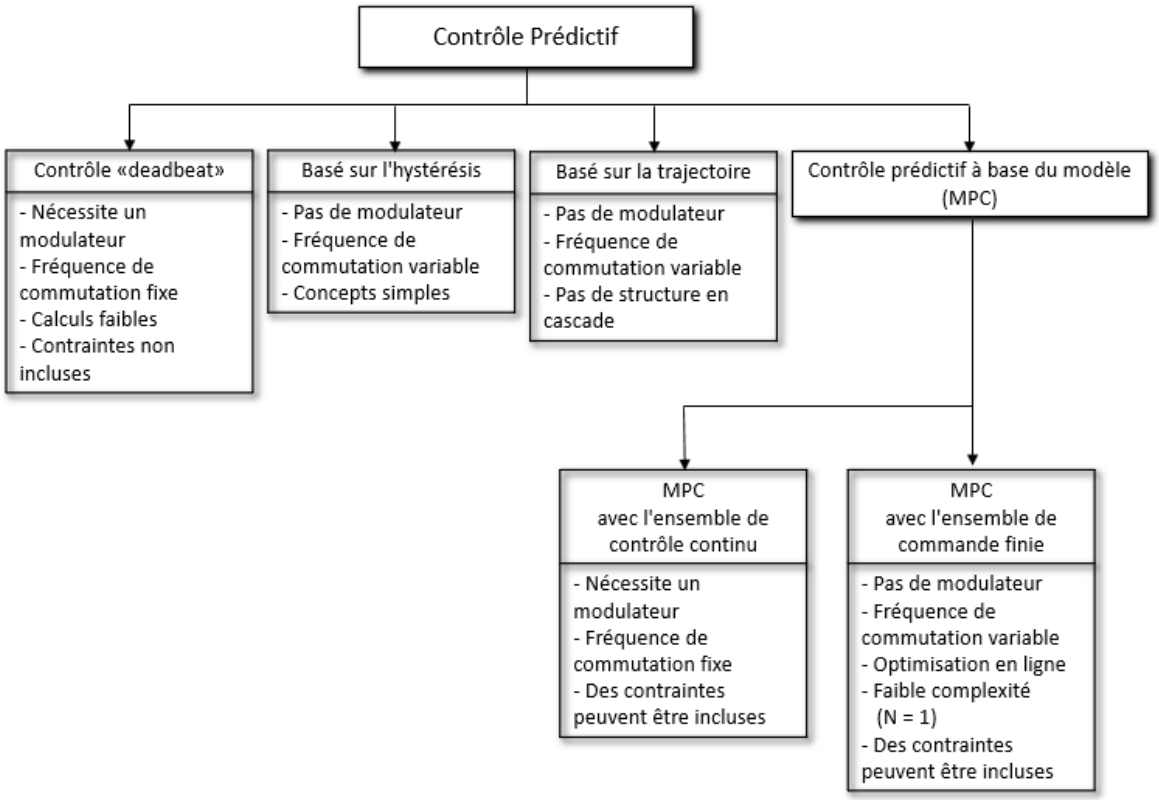


Figure 3.1 Classification des méthodes de contrôle prédictives (tiré et traduit de [5])

Les critères d'optimisation diffèrent d'un type de contrôle prédictif à un autre, ainsi que des caractéristiques du système à contrôler. On y distingue par exemple le contrôle basé sur l'hystérésis, le contrôle du Deadbeat, le contrôle de trajectoire ou le MPC. Pour ce qui est du contrôle basé sur l'hystérésis, le critère est de maintenir la variable contrôlée dans une limite de la bande d'hystérésis, alors que dans le cas d'un contrôle « Deadbeat », il est nécessaire de trouver l'action qui conduit à une erreur nulle dans l'instant d'échantillonnage suivant. Pour le contrôle basé sur la trajectoire, les variables contrôlées suivent une trajectoire prédéfinie. En ce qui concerne le contrôle prédictif basé sur le modèle (MPC – *Model Preditive Control*), le critère d'optimisation est une fonction de coût qui doit être minimisée, ce qui nous offre plus d'options ou possibilités d'utiliser différentes formes d'optimisation [18].

Ce projet se concentrera sur le contrôle prédictif basé sur le modèle avec un ensemble de contrôle fini et un horizon de prévision fini dans le temps $(k+1)$, c'est-à-dire un pas en avance. L'avantage majeur ici est que l'ensemble des commandes finies est facile à implémenter, ne nécessite aucun modulateur et fonctionne donc avec une fréquence de commutation variable. Le MPC est basé sur un nombre fini d'états de commutation et sur la sélection de la commutation optimale en minimisant une fonction de coût, et comprend la nature discrète des convertisseurs de puissance. Le MPC a démontré son extrême efficacité dans plusieurs applications pratiques et sa grande influence dans plusieurs domaines de recherche au cours des dernières décennies [5]. La Figure 3.2 présente le schéma général pour un MPC, où $\boldsymbol{x}^*(k)$ est la valeur de référence et $\boldsymbol{S}$ est la fonction de commutation ou matrice/vecteur de commutation.

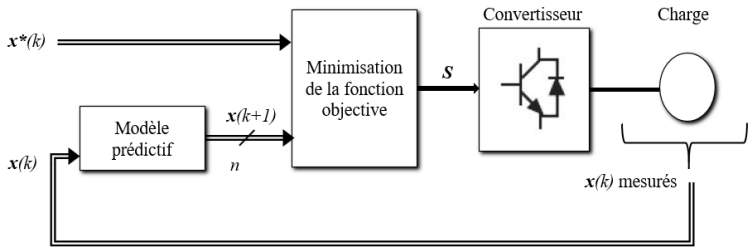


Figure 3.2 Schéma général d'un MPC pour les convertisseurs de puissance (tiré et traduit de [18]).

En se basant sur la valeur mesurée $\boldsymbol{x}(k)$, une prévision de la valeur suivante $\boldsymbol{x}(k+1)$ est réalisée. On compare ensuite cette valeur à la valeur de référence, et en minimisant la fonction de coût, l'état de commutation optimal, $\boldsymbol{S}$, est trouvé. Ce processus est répété pour chaque nouvel instant d'échantillonnage. La Figure 3.3 illustre ce principe de base du travail. Les valeurs futures du système sont prédites jusqu'à un horizon fini dans le temps $(k+N)$, en utilisant le modèle du système et les valeurs mesurées au temps k. L'état de commutation optimal est détecté, et ceci est appliqué jusqu'au prochain instant d'échantillonnage $\boldsymbol{x}(k+1)$ où les calculs sont effectués à nouveau.

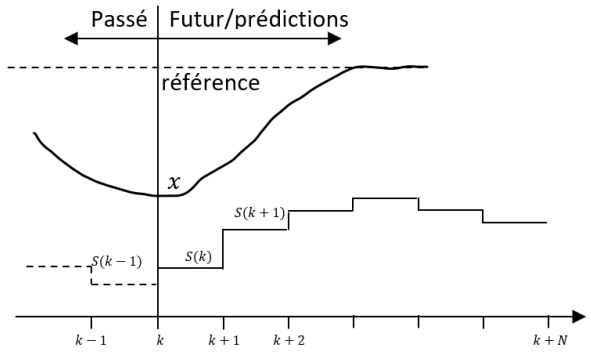


Figure 3.3 Principe de fonctionnement du modèle de contrôle prédictif (tiré et traduit de [18]).

3.2 Modèle du système

Pour le calcul des valeurs prédites des variables, un modèle mathématique du système à contrôler est nécessaire. Le système qui sera modélisé pour ce projet correspond au circuit de puissance de l'onduleur présenté à la Figure 2.1. Cette section présentera de façon détaillée un modèle mathématique du système utilisé pour le contrôle prédictif du courant.

3.2.1 Modèle de l'onduleur

Les principes de fonctionnement de l'onduleur ont été présentés dans le Chapitre 2. L'équation (9) a décrit la définition du vecteur de tension de charge en sortie. Cette équation peut également être exprimée comme suit :

$$\begin{bmatrix} v_{aN}(t) \\ v_{bN}(t) \\ v_{cN}(t) \end{bmatrix} = V_{dc} * \begin{bmatrix} S_a(t) \\ S_b(t) \\ S_c(t) \end{bmatrix} \tag{10}$$

où les différents états de commutation ont été définis dans les équations (1) – (3).

3.2.2 Modèle de la charge en sortie

Un modèle de la charge en sortie est défini pour la prévision du courant de charge. La charge résistive-inductive peut être décrite par l'équation suivante

$$L\frac{d\boldsymbol{i_0}}{dt} = \boldsymbol{v_0}(t) - R\boldsymbol{i_0} - \boldsymbol{e}(t) \tag{11}$$

où les variables L et R sont respectivement l'inductance et la résistance de la charge, $\boldsymbol{v_0}$ est le vecteur de tension généré par l'onduleur, $\boldsymbol{i_0}$ est le vecteur de courant de la charge et $\boldsymbol{e}$ est la force contre-électromotrice, f.c.é.m.. Avec une période d'échantillonnage égale à T_s, l'approximation suivante peut être effectuée :

$$\frac{d\boldsymbol{i_0}}{dt} \approx \frac{\boldsymbol{i_0}(k+1) - \boldsymbol{i_0}(k)}{T_S} \tag{12}$$

En substituant l'équation (11) dans (12), on obtient l'équation nécessaire pour calculer la valeur prédite du courant de charge au moment $(k+1)$.

$$\boldsymbol{i_0}\,(k+1) = \left(1 - \frac{R\boldsymbol{T_s}}{L}\right)\,\boldsymbol{i_0}(k) + \frac{T_S}{L}(\boldsymbol{v_0}(k) - \hat{\boldsymbol{e}}(k)) \tag{13}$$

où $\hat{\boldsymbol{e}}(k)$ est l'estimation de la f.c.é.m.

Le vecteur de tension de charge correspondant est calculé pour chaque état de commutation valide, en utilisant l'équation (10). L'extrapolation à partir des valeurs actuelles et passées peut être utilisée pour estimer la valeur actuelle de la f.c.é.m., $\boldsymbol{e}(k)$. L'approximation $\boldsymbol{e}(k-1) \approx \boldsymbol{e}(k)$ peut être utilisée, car le temps d'échantillonnage ($25\ \mu s$ et $1\ \mu s$) est suffisamment petit, et par conséquent, aucune extrapolation n'est nécessaire [17].

3.2.3 Fonction de coût

La fonction de coût représentera le comportement désiré du système. Lorsqu'elle est minimisée, on obtient l'état de commutation optimal du système. Plusieurs objectifs peuvent être mis en œuvre dans cette fonction. Pour ce qui concerne ce projet, l'objectif sera de contrôler le courant de sortie et de le maintenir proche de la valeur de référence. La fonction de coût sera donc l'erreur entre le courant de consigne et le courant mesuré, et devrait converger vers 0. De plus, un contrôle MPC peut faire naître de nouveaux objectifs de contrôle comme la réduction des harmoniques, l'amélioration de la fréquence de commutation ou même un système de contrôle en cascade.

3.2.3.1 Contrôle en courant

L'idée ici est d'affecter un coût à la fonction de coût, g, chaque fois qu'une valeur prédite s'écarte de la valeur de consigne. L'horizon de prévision dans cette étude étant de 1, c'est à dire un pas en avance, le contrôle de courant peut alors être décrit par l'équation :

$$g_{actuel} = \left|i^*_{o\alpha}(k+1) - i^p_{o\alpha}(k+1)\right| + \left|i^*_{o\beta}(k+1) - i^p_{o\beta}(k+1)\right| \quad (14)$$

où le terme (*) désigne les valeurs de référence et p les valeurs prédites. $i^*_{o\alpha}$ et $i^*_{o\beta}$ représentent les courants en coordonnées biphasées $\alpha\beta$ qui seront définies à la section 4.1.3 sur la transformation des coordonnées (abc/αβ) selon la transformée de Clarke. Ozan Gulbudak et Enrico Santi [6] proposent une fonction de coût basée sur la mise aux carrées des erreurs plutôt qu'en valeurs absolues selon l'équation (14). Des tests ont été réalisés montrant qu'une mise en valeur absolue offrait les mêmes performances. La motivation de proposer des erreurs en valeurs absolues est une réduction des ressources et de la latence en calcul.

3.3 Conclusion

Dans ce chapitre, la commande prédictive a été illustrée par les différentes méthodes qui la constituent, ainsi que son principe de fonctionnement. En plus, le schéma global du système à étudier a été présenté, de même que son modèle mathématique. Cette modélisation sera utilisée pour implémenter le système dans les chapitres suivants.

Chapitre 4 - Modélisation Simulink du système

4.1 Modèle Simulink – Modèle moyen

Pour la première phase du projet, la plate-forme Matlab-Simulink a été mise en contribution, pour simuler le circuit de puissance présenté à la Figure 2.1 avec l'algorithme du contrôle prédictif du chapitre 3 et d'analyser la pertinence des résultats. Le but sera de l'adapter, plus tard à la deuxième phase, dans l'environnement Xilinx System Generator (RT-XSG), afin de l'implanter par la suite dans le FPGA du simulateur temps réel. Le modèle Simulink du système est développé à partir du modèle théorique et mathématique présenté au chapitre précédent.

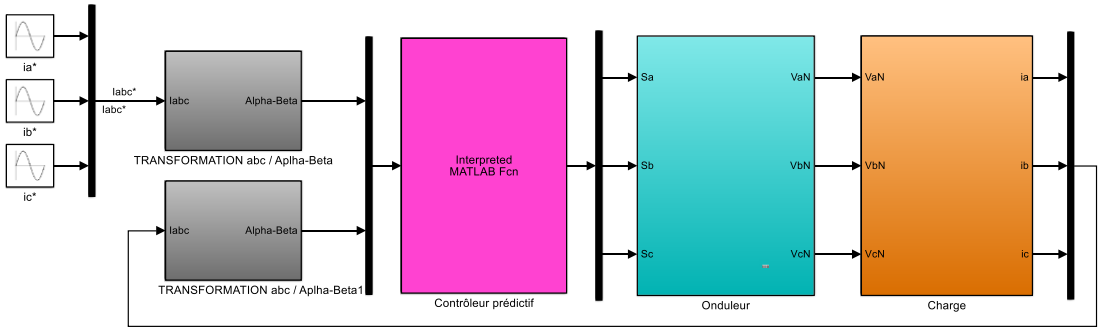


Figure 4.1 Modèle du système dans Simulink.

Pour des raisons de compatibilité avec la librairie XSG et l'environnement RT-LAB, la version Matlab 2011b (32bits) a été utilisée. Le modèle Simulink du système est illustré à la Figure 4.1. Il comprend les courants de consigne, les sous-systèmes pour la transformation des coordonnées des courants de consigne et mesurés de triphasé à biphasé (Transformée de

Clarke), un bloc fonction Matlab/Simulink contenant l'algorithme de contrôle prédictif, un sous-système onduleur et un sous-système charge inductive-résistive. Le sous-système de l'onduleur contient une constante correspondant à la source CC (bus CC) pour son alimentation. Dans la réalité, cette source CC varie en fonction de la charge et l'onduleur, et doit par conséquent être contrôlée. De plus, un fichier de paramètres a été implémenté dans le but d'avoir un bon aperçu des différents paramètres du système à étudier et pour les manipuler facilement.

4.1.1 Modélisation charge

La Figure A-1 de l'annexe A présente la modélisation de la charge résistive-inductive du système dans l'environnement Matlab/Simulink. Comme illustrée à la Figure 4.2, la connexion de la charge est en configuration étoile. La loi sur le courant de Kirchhoff peut ensuite être utilisée pour décrire ou exprimer les tensions de phase de la charge comme suit :

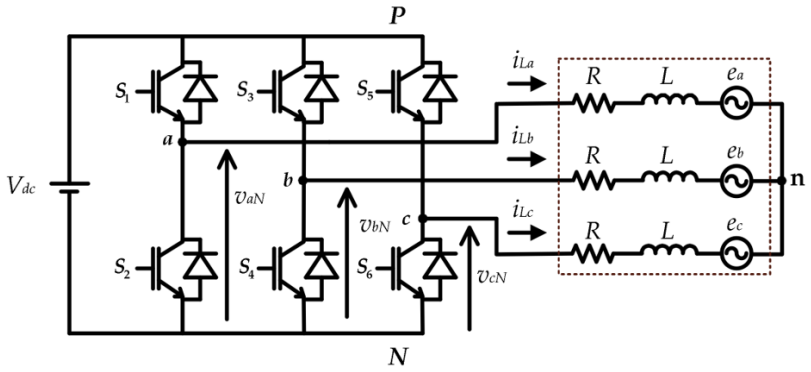


Figure 4.2 : Onduleur triphasé avec tension simple et charge [5]

$$v_{an} = v_{aN} - v_{nN} \tag{15}$$

$$v_{bn} = v_{bN} - v_{nN} \tag{16}$$

$$v_{cn} = v_{cN} - v_{nN} \tag{17}$$

La tension en mode commun peut alors être obtenue en ajoutant les tensions de phase.

$$v_{aN} + v_{bN} + v_{cN} = L\frac{d}{dt}(i_a + i_b + i_c) + R(i_a + i_b + i_c) + (e_a + e_b + e_c) + 3v_{nN} \quad (18)$$

On a la relation suivante : $i_a + i_b + i_c = 0$, lorsqu'une charge est branchée en topologie étoile. En supposant un équilibre de la force contre-électromotrice, f.c.é.m., $e_a + e_b + e_c$ sera également nul. On obtient une expression plus simplifiée pour la tension en mode commun.

$$v_{nN} = \frac{1}{3}(v_{aN} + v_{bN} + v_{cN}) \quad (19)$$

Cette relation peut être facilement implémentée dans le modèle Simulink. La f.c.é.m. est représentée avec les blocs « Sine Wave » et la dynamique de la charge en sortie à l'aide de fonctions de transfert linéaires en temps continu. La fonction de transfert est déterminée à partir de l'équation de base pour un circuit inductif-résistif, présentée ici par exemple pour la phase a :

$$v_{an} = L\frac{di_a}{dt} + Ri_a + e_a \quad (20)$$

et en appliquant la transformée de Laplace de cette équation, on obtient :

$$\frac{I_a}{V_{an} - E_a} = \frac{1}{Ls+R} \quad (21)$$

où s est la variable de Laplace. Dans l'équation ci-dessus, les lettres majuscules représentent les transformées de Laplace des variables minuscules respectives dans le domaine temporel. La sortie des fonctions de transfert est le courant de charge de chaque phase, tandis que leur entrée est une tension obtenue en soustrayant la force contre-électromotrice de la tension de charge correspondante. La force contre-électromotrice est considérée comme sinusoïdale

avec une amplitude et une fréquence constantes et est simulée par des blocs d'ondes sinusoïdales standard.

4.1.2 Modélisation de l'onduleur

L'onduleur du système est illustré par Figure A-2 de l'annexe A. Il prend en entrée les signaux de commutation optimaux et une constante, qui représente la tension V_{dc} de la source CC. Dans la plupart des cas pratiques, la tension du bus CC ne sera pas constante et devra être contrôlée. Le bloc V_{dc} de ce modèle sera donc substitué par un modèle décrivant la tension continue variable. En multipliant les signaux de commutation par la tension du bus CC, on calcule la tension de sortie de chaque bras inverseur par rapport au point négative N.

4.1.3 Transformation des coordonnées (abc/αβ) : Transformée de Clarke

Les courants de consigne en entrée et les courants de charge mesurés en sortie sont donnés en coordonnées triphasées abc représentant les trois phases. En transformant ces courants en coordonnées biphasées $\alpha\beta$ [18], le nombre de prévisions requis dans l'algorithme de contrôle prédictif peut être réduit. Les équations (22) et (23) démontrent la relation entre les deux systèmes de coordonnées.

$$i_\alpha = \frac{2}{3}\left(i_a - \frac{1}{2}i_b - \frac{1}{2}i_c\right) \tag{22}$$

$$i_\beta = \frac{2}{3}\left(\frac{\sqrt{3}}{2}i_b - \frac{\sqrt{3}}{2}i_c\right) \tag{23}$$

La Figure A-3 de l'annexe A présente la mise en œuvre pratique de ces équations obtenues par la transformée de Clarke [18] dans l'environnement Simulink.

4.1.4 Fichier de paramètres

Les paramètres nécessaires à la simulation sont donnés dans un fichier de paramètres.m, qui se trouve à l'annexe A. Le Tableau 4-1 présente les noms de paramètres et les valeurs définies dans le fichier *.m. Les simulations de l'algorithme de prévision ont été faites avec les périodes d'échantillonnage T_s de 25 μs et 1 μs, pour analyser l'effet de la fréquence d'échantillonnage sur la performance de la méthode de contrôle. La fréquence d'échantillonnage du bloc fonction Matlab doit être définie sous les paramètres du sous-système.

Paramètres	**Nom des paramètres**	**Valeurs**
Tension du bus CC	V	540 V
Résistance de la charge	R	10 Ω
Inductance de la charge	L	10 mH
f.c.é.m. de la charge	e	100 V
Fréquence	f	60 Hz
Amplitude du courant de référence	Amp_Div	10 A
Période d'échantillonnage de l'algorithme de contrôle prédictif	Ts	25µs et 1µs

Tableau 4-1 Paramètres définis dans le fichier de parametres.m

En plus de définir les paramètres présentés dans le Tableau 4-1 dans le fichier de parametres.m, les différents vecteurs de tension y sont également calculés. De même, tous les états de commutation valides pour le convertisseur sont définis dans ledit fichier.

La définition de certains paramètres de configuration de simulation est nécessaire sous Simulink, en plus des paramètres définis par l'exécution du fichier des paramètres de

simulation.m. Le Tableau 4-2 contient les paramètres qui ont été configurés. Les autres paramètres de configuration de simulation restent inchangés.

Paramètres	Valeurs
Temps initial	0 s
Temps d'arrêt	0.18 s
Type de solveur	Pas fixe
Solveur	Ode 5 (Dormand-Prince)
Valeur du pas de calcul	1 µs

Tableau 4-2 Paramètres de configuration dans l'environnement de simulation

4.1.5 Algorithme de contrôle prédictif

L'algorithme de prévision est développé à l'aide du bloc fonction de Matlab. Il s'agit d'une version éditée du bloc « *Embedded Matlab Function* ». Le bloc fonction de Matlab utilise le vecteur de tension qui a été calculé dans le fichier des paramètres en fonction de la tension du bus CC. Pour passer les variables en entrées à cette fonction, les courants de références et mesurés ont été séparés en leurs parties réelles et imaginaires ; ce qui permet de simplifier les calculs.

Le *Code 1*, présenté à la page suivante, montre l'implémentation de l'algorithme de contrôle prédictif dans Matlab. Les paramètres en entrée de la fonction sont les parties réelles et imaginaires du courant de référence et du courant de charge mesuré. Aux lignes 12 et 14, ces deux courants sont stockés dans les variables ik_ref et ik. Le préfixe k indique l'instant d'échantillonnage dans le temps. À la ligne 6, les variables $\boldsymbol{x}\_old$ et $\boldsymbol{i}\_old$ sont initialisées en tant que variables persistantes. Si la variable n'existe pas la première fois, que vous émettez

l'instruction persistante, elle sera initialisée sous forme de matrice vide. Les variables persistantes sont similaires aux variables globales, à l'exception qu'elles sont locales à la fonction. Il en résulte que la valeur est retenue entre les appels de fonctions, car Matlab crée un stockage permanent pour la variable.

```
1     function [X] = controle(I_ref,I_ref_im,I_meas,I_meas_im)
2     %#codegen
3     % Variables définies dans le fichier de paramètres de simulation
4     global Ts R L v etats
5     % Vecteur optimal et courant mesuré à l'instant k-1
6     persistent x_old i_old
7     % Initialisation des valeurs
8     if isempty(x_old), x_old = 1; end
9     if isempty(i_old), i_old = 0 + 1j*0; end
10    g_opt = 1e10;
11    % Lire le courant de référence en entrée à l'instant
d'échantillonnage k
12    ik_ref = I_ref + 1j*I_ref_im;
13    % Lire le courant mesuré à l'instant d'échantillonnage k
14    ik = I_meas + 1j*I_meas_im;
15    % Estimation de la f.c.é.m
16    e = v(x_old) - L/Ts*ik - (R - L/Ts)*i_old;
17    % Sauvegarder le courant mesuré pour la prochaine itération
18    i_old = ik;
19    for i = 1:8 % Calcul de la fonction de coût pour chacun des 8
états de commutation valides (états 0 à 7)
20        % i-ème vecteur de tension pour la prévision de courant
21        v_o1 = v(i);
22        % Prévision du courant à l'instant k+1
23        ik1 = (1 - R*Ts/L)*ik + Ts/L*(v_o1 - e);
24        % Fonction de coût
25        g = abs(real(ik_ref - ik1)) + abs(imag(ik_ref - ik1));
26        % Selection de la valeur optimale des 8 possibilités
27        if (g < g_opt)
28            g_opt = g;
29            x_opt = i;
30        end
31    end
32    % Sauvegarde la valeur actuelle de x_opt
33    x_old = x_opt;
34
35    % États de commutation de sortie
36    Sa = etats(x_opt,1);
37    Sb = etats(x_opt,2);
38    Sc = etats(x_opt,3);
39
40    X=[Sa,Sb,Sc,g_opt];
```

Code 1 : Algorithme du contrôle prédictif

Dans la boucle « For » qui va de la ligne 19 à la ligne 31, la fonction de coût pour chacun des 8 états de commutation valides est calculée et l'état de commutation optimal selon le critère Eq. (14), c'est-à-dire celui dont la fonction de coût est minimale, est choisi (ligne 27). La prévision de la valeur suivante pour le courant de charge est effectuée à la ligne 23, selon l'équation (13). La fonction de coût est alors estimée à la ligne 25, correspondant à l'équation (14). La combinaison de commutation avec la fonction de coût (ou de coût) le plus bas est sélectionnée et définie comme sortie de la fonction.

4.2 Résultats de simulation sur Simulink

4.2.1 Simulation avec une période d'échantillonnage de 25µs

Les Figure 4.3 à Figure 4.7 présentent certains résultats de la simulation effectuée dans Simulink où la période d'échantillonnage pour l'ensemble du système a été fixée à 25 μs, et le bloc « Function block » contenant l'algorithme de contrôle prédictif a eu un échantillonnage de 25 µs.

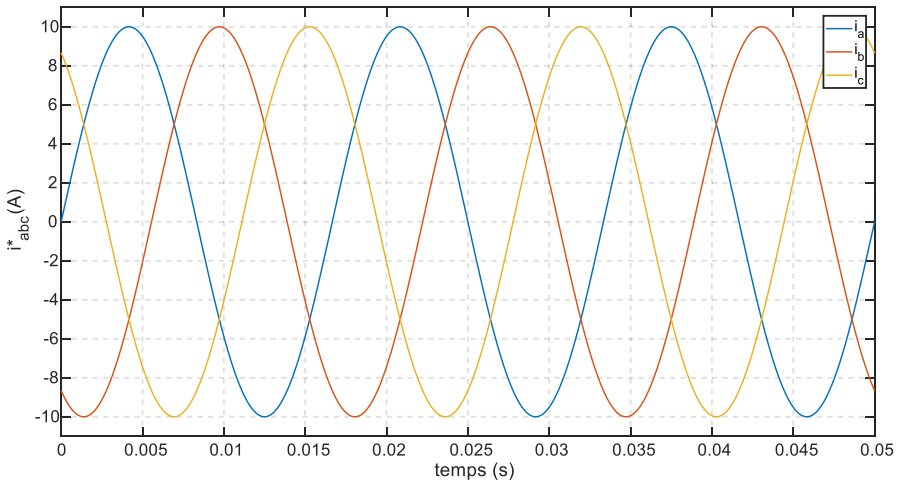


Figure 4.3 Courant de consigne triphasé *abc*

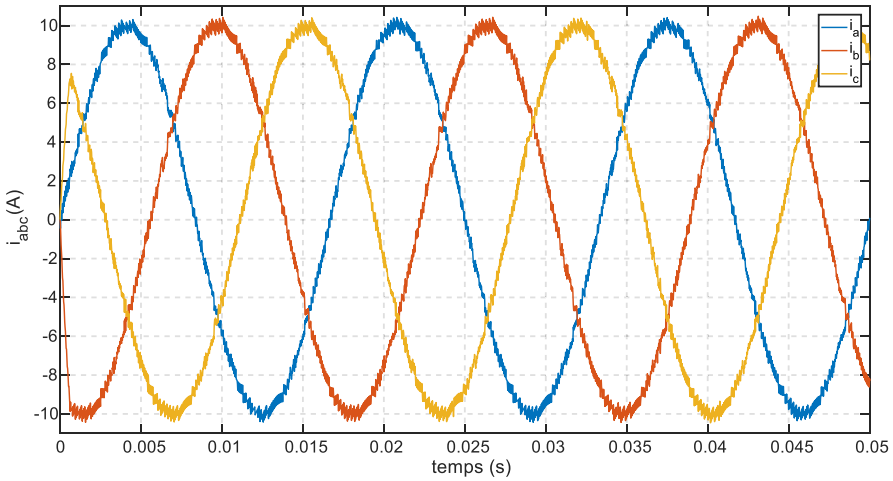


Figure 4.4 Courant de charge triphasé mesuré

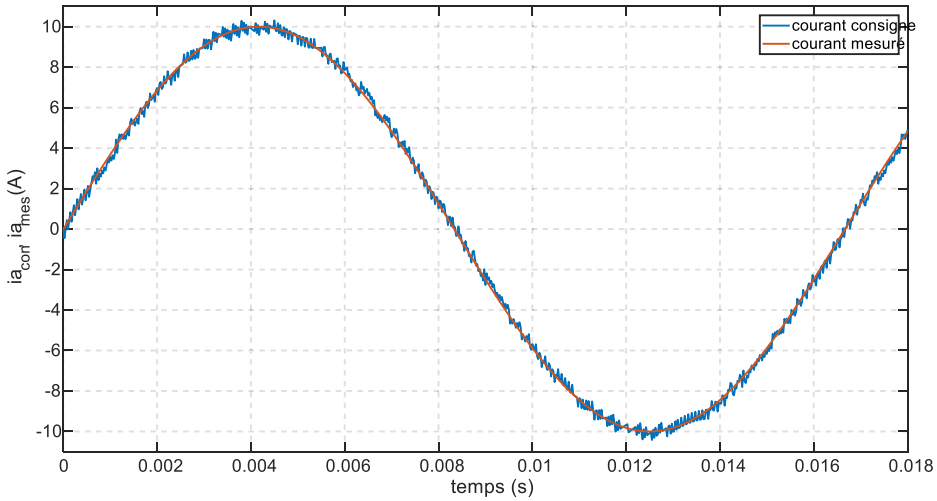
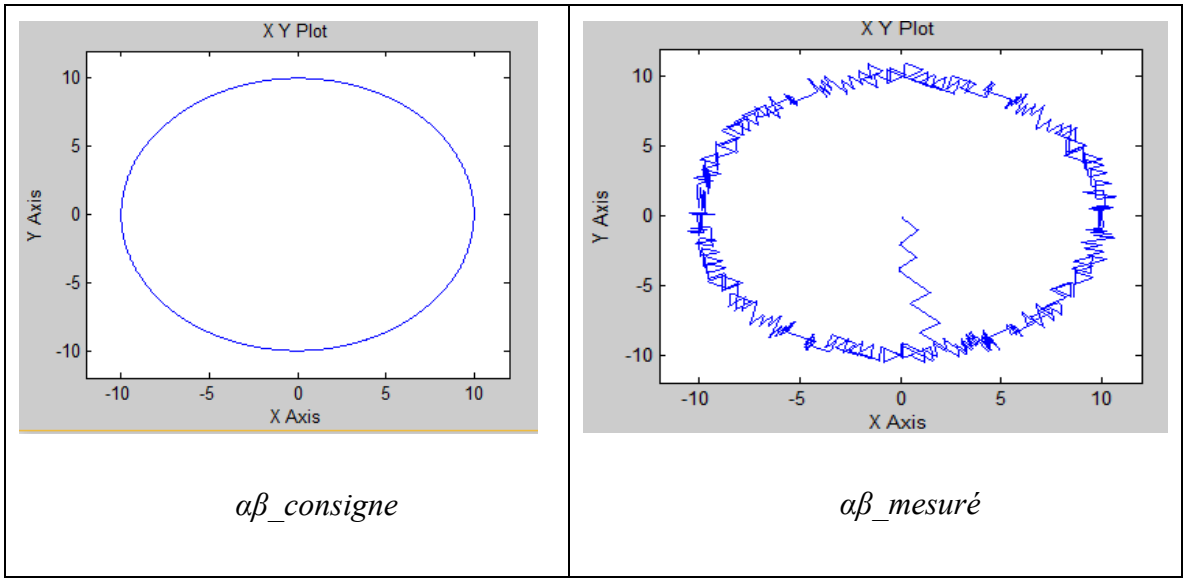


Figure 4.5 Courant référence ia_consigne et courant mesuré ia_mesure (Phase a)



αβ_consigne *αβ_mesuré*

Figure 4.6 αβ_consigne et αβ_mesuré dans le plan complexe αβ

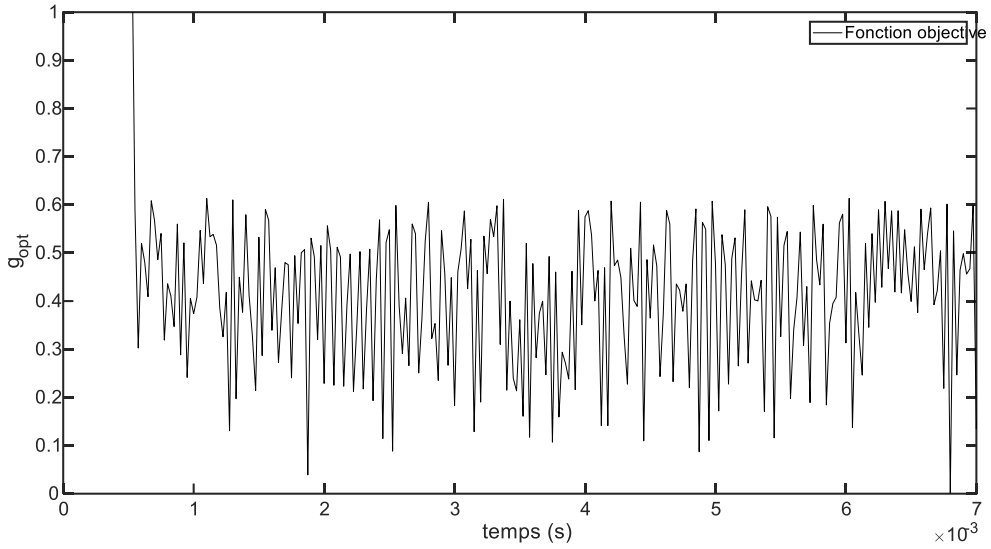


Figure 4.7 Fonction de coût, g.

La Figure 4.4 illustre le courant de sortie livré à la charge, qui a une forme sinusoïdale avec de l'ondulation. Comme on peut le constater à partir de la Figure 4.5, un bon suivi de la consigne est accompli. La fonction de coût est présentée dans la Figure 4.7. Cela montre l'écart entre le courant de sortie et celle de référence plus en détail. Comme prévu, le système

a besoin de temps pour se stabiliser, mais après environ 0.5 ms, la valeur de g sera centrée autour de 0.4.

4.2.2 Simulation avec une période d'échantillonnage de 1µs

La simulation de l'algorithme de contrôle de prévision a été effectuée avec un temps d'échantillonnage de 1 μs. Les Figure 4.8 à Figure 4.11 illustrent les résultats de cette simulation.

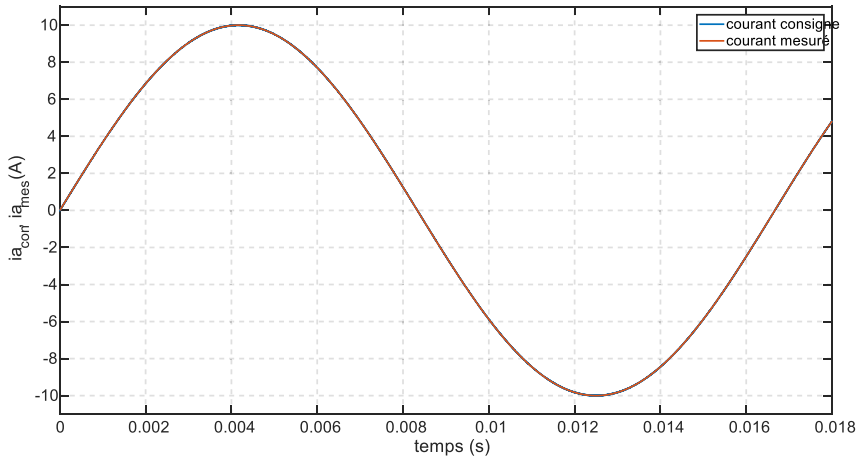


Figure 4.8 Courant référence ia_consigne et courant mesuré ia_mesure (Phase a)

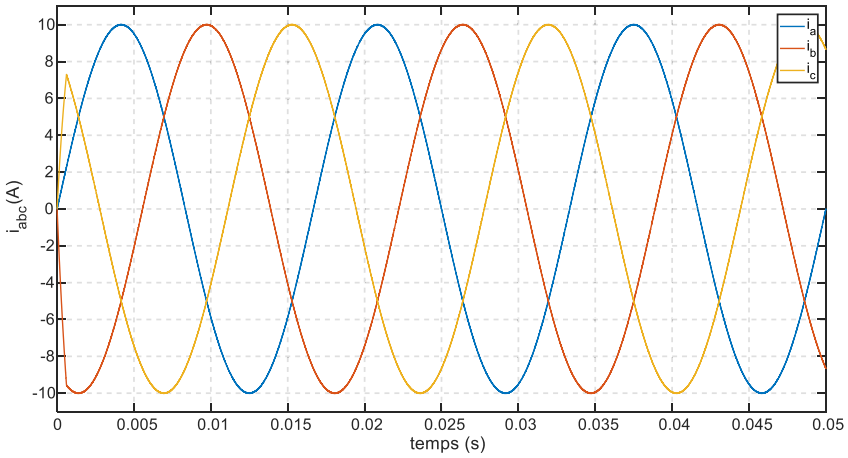


Figure 4.9 Courant Triphasé mesuré *Iabc*

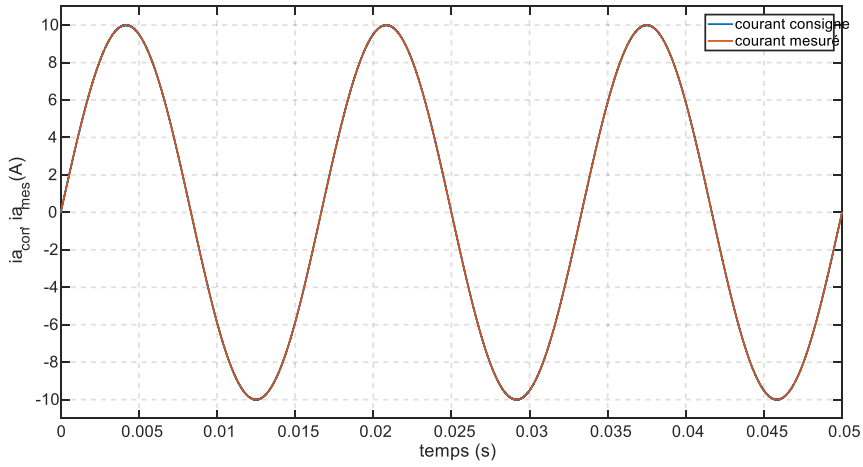


Figure 4.10 Courant référence ia_consigne et courant mesuré ia_mesure (Phase a)

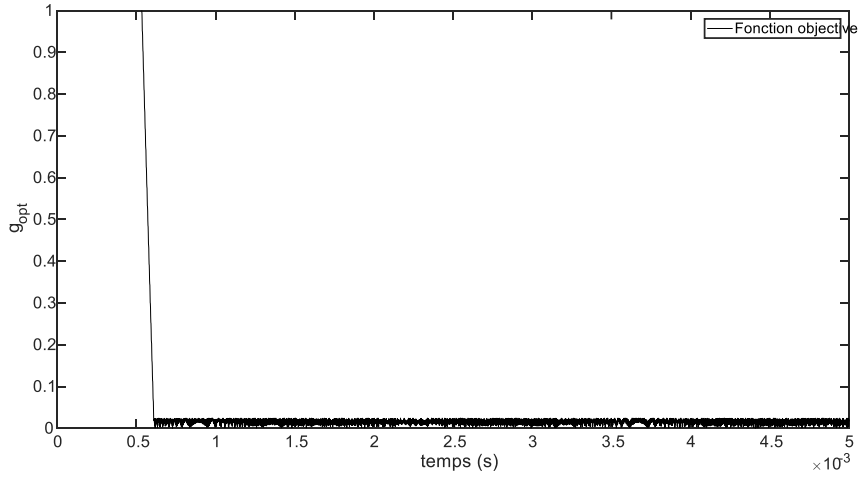


Figure 4.11 Fonction de coût, g

Comme on peut le constater, la Figure 4.8 montre une diminution considérable de l'ondulation du courant en sortie. La Figure 4.10 présente une amélioration significative du suivi du courant de consigne. Une valeur moyenne d'environ 0.015 de la fonction de coût est obtenue à la Figure 4.11. Cette valeur représente la déviation des courants mesurés par

rapport aux courants de consigne, ce qui démontre, de façon claire, une très bonne performance de la méthode de contrôle. Ces résultats étaient prévisibles, car pour une fréquence d'échantillonnage élevée, les opérations effectuées dans l'algorithme de prévision fonctionneront avec une précision accrue.

4.3 Modélisation du système avec la librairie SimPower System dans Simulink

La modélisation a également été effectuée avec la librairie « SimPower System », afin d'analyser le comportement de l'algorithme de contrôle prédictif en utilisant les composants réels. La Figure 4.12 montre l'implémentation du modèle avec les composants physiques réels. L'implémentation des différents sous-systèmes est présentée à l'annexe B.

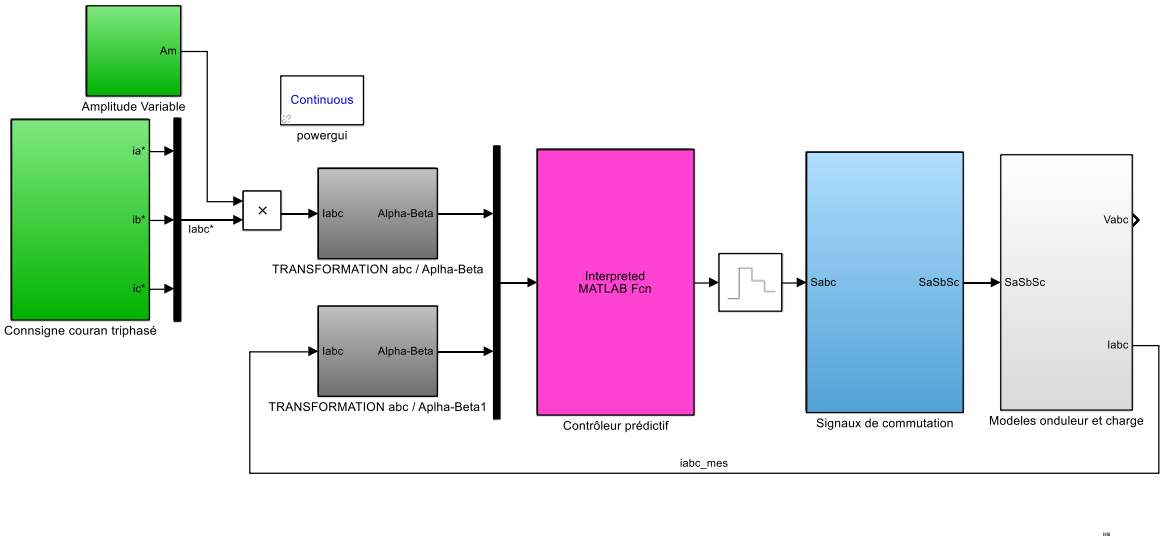


Figure 4.12 Implémentation de l'algorithme avec les composants physiques réels

4.4 Résultats de la simulation dans Simulink avec SimPower System

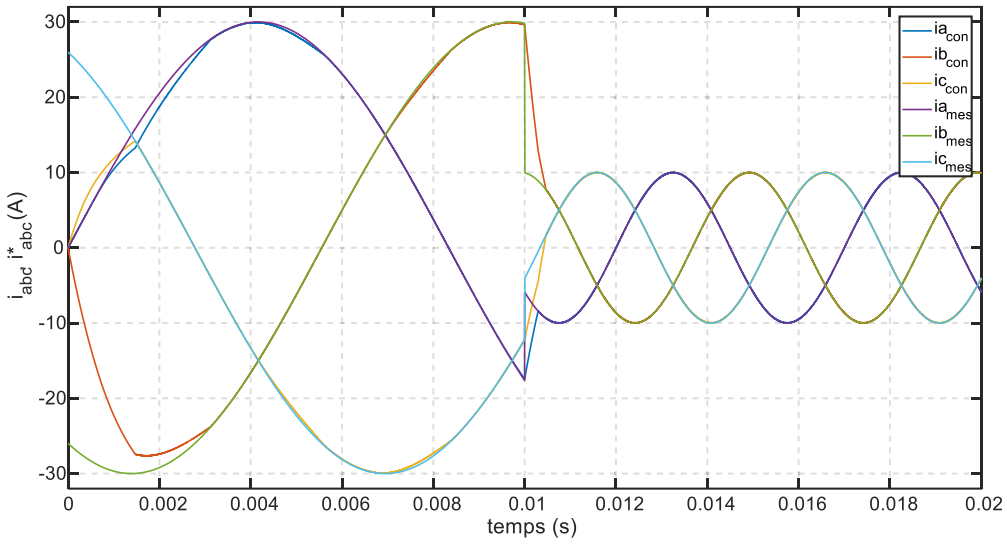
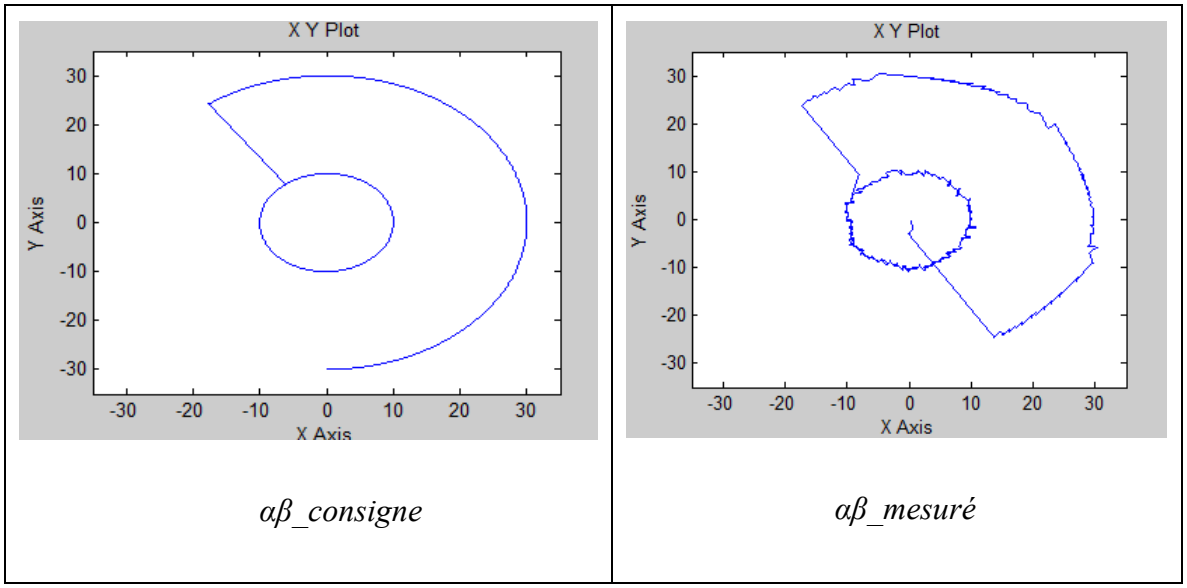


Figure 4.13 Courants de consigne abc/Courants de charge mesurés abc



αβ_consigne *αβ_mesuré*

Figure 4.14 αβ_consigne et αβ_mesuré dans le plan biphasé

Les résultats de la simulation de l'algorithme de contrôle prédictif avec les composants physiques réels sont présentés ci-dessus. Dans la Figure 4.13, on peut observer un très bon suivi des courants de consigne, même lorsqu'une variation de l'amplitude de $30A$ à $10A$ est

effectuée. La Figure 4.14 montre également le comportement des courants de consigne et mesuré dans le plan biphasé $\alpha\beta$.

4.5 Conclusion

Les résultats présentés dans ce chapitre sont convaincants et motivants pour exécuter l'algorithme de contrôle prédictif sur un FPGA, qui possède une puissance de calcul très élevée. Ceci est important pour une implémentation physique, car le calcul des états de commutation pourrait être effectué avec une vitesse très élevée. En revanche, certains facteurs doivent pris en considération dans un circuit physique avant de décider d'une fréquence de commutation. Comme on a pu le constater avec les résultats de simulation, une fréquence élevée produit un meilleur suivi de la référence, mais entraînera aussi des pertes de commutation très importantes. Le chapitre suivant décrira comment la méthode de contrôle prédictif testée dans Simulink sera adaptée pour une utilisation dans une configuration expérimentale.

Chapitre 5 - Méthodologie et implémentation du système

Dans ce chapitre, on se consacrera à la présentation des outils nécessaires à la mise en œuvre du système présenté dans les chapitres précédents. Dans un premier temps, la plateforme de travail sera présentée ainsi que la méthodologie utilisée et ensuite viendra l'implémentation de façon virtuel d'un onduleur triphasé avec contrôle MPC. L'onduleur sur lequel sera implémenté la MPC n'est autre qu'un simulateur présentant un comportement similaire à un onduleur physique (ou réel) : RT-LAB.

5.1 RT-Lab et Xilinx System Generator

5.1.1 Introduction à RT-Lab et XSG

Les simulations précédentes avec Matlab/Simulink présentent de très bons résultats et une très bonne performance de la commande. Ces résultats permettent de conclure que le contrôle prédictif peut être considéré comme une méthode de contrôle intéressante et prometteuse pour l'onduleur de puissance. Cependant, avant d'arriver à cette conclusion, la méthode devrait être implémentée et vérifiée expérimentalement ; ce qui nécessitera un système de contrôle pouvant fonctionner avec un temps de calcul court pour pouvoir suivre le système physique en temps réel. Pour atteindre ce défi, le logiciel de simulation RT-Lab de la compagnie OPAL-RT est utilisé, combiné avec la librairie Xilinx System Generator (XSG). Le but est d'adapter la méthode de contrôle prédictif qui sera utilisée dans une configuration expérimentale avec le simulateur temps réel. RT-Lab est une plate-forme temps réel distribuée qui permet la réalisation de la simulation en temps réel de modèles Simulink avec du matériel dans la boucle (HIL). Xilinx System Generator (RT-XSG) est une boîte à

outils Simulink qui permet aux ingénieurs de générer des modèles d'applications personnalisés et spécifiques qui peuvent être implémentés sur FPGA.

La version RT-Lab 11.0.2.410 est utilisée pour développer le modèle du système. Ce logiciel peut être installé sur un ordinateur Windows XP/Vista /7 ou Red Hat Linux, qui est la station de commande. Il s'agit ici de l'ordinateur qui sert d'interface utilisateur. Lorsque le modèle est configuré et adapté pour être utilisé dans RT-Lab, il est automatiquement codé en C et envoyé aux nœuds cibles pour l'exécution. Dans ce cas, les nœuds cibles sont des cœurs parallèles situés dans un simulateur en temps réel OP4510 HILBOX produit par la compagnie OPAL-RT Technologies. Le simulateur dispose de quatre cœurs disponibles et fonctionne sur le système d'exploitation en temps réel Intel Xeon E3 3.5GHz. Un cœur est réservé au système d'exploitation. La connexion physique entre l'onduleur et FPGA est établie en introduisant des blocs personnalisés dans le modèle RT-Lab/XSG, qui gère l'interface pour les périphériques d'Entrées/Sortie.

L'algorithme de contrôle prédictif sera exécuté sur le FPGA, car il nécessite un temps de calcul très court. Un module FPGA est un circuit intégré à usage général, qui est « programmé » par l'utilisateur à l'aide de la mémoire flash. Il se programme en téléchargeant un fichier de configuration appelé « bitstream » dans la mémoire d'accès aléatoire statique (SRAM) de la puce. Donc, à l'aide des blocs de la librairie XSG, un modèle Simulink de haut niveau peut facilement être transcrit dans un programme FPGA exécutable de bas niveau. En général, un langage de description de matériel (HDL) est utilisé pour décrire la configuration du FPGA et l'outil XSG servira à compiler ce code. XSG est un outil très efficace et utile pour les personnes qui connaissent bien la programmation de haut niveau et Matlab/Simulink, mais de connaissance en programmation de bas niveau requise pour

contrôler un FPGA. Cependant, afin de créer un programme fonctionnel, on devrait avoir une certaine connaissance du principe de fonctionnement et des limites du module FPGA.

5.1.2 Blocs obligatoires pour le modèle RT-XSG

Avant d'utiliser RT-XSG avec des blocs d'Entrée/Sortie, les quatre blocs suivants sont obligatoires dans la conception et doivent être configurés. Il s'agit des blocs « System Generator », « OPAL-RT FPGA Synthesis Manager », « Version » et « Hardware Configuration ». Ces derniers se trouvent dans la boîte à outils XSG.

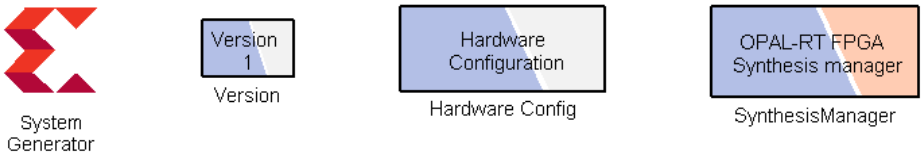


Figure 5.1 Blocs de configuration obligatoires

Le bloc « System Generator » est obligatoire pour toutes les conceptions Xilinx-Simulink. Dans ce bloc, plusieurs paramètres peuvent être définis, y compris le type de puce FPGA qui sera programmée, la langue FPGA qui devrait être utilisée ainsi que la fréquence d'horloge de la conception. Le jeton « System Generator » est l'ensemble des bibliothèques d'éléments et outils de base de Xilinx. Le Tableau 5-1 présente les paramètres définis pour ce bloc.

Compilation	**HDL Netlist**
Part	Kintex7 xc7k325t-3fbg676
Synthesis tool	XST
Hardware description language	VHDL
FPGA clock period (ns)	10
Simulink system period (sec)	5e-6

Tableau 5-1 Paramétrage du bloc System Generator

HDL Netlist est le type de compilation le plus couramment utilisé. Dans ce mode, les résultats de la génération sont une collection de fichiers HDL et EDIF, ainsi que quelques fichiers auxiliaires qui simplifient le traitement en aval. Cette partie décrit le type FPGA physique qui sera utilisé. L'outil de synthèse est l'outil utilisé pour synthétiser la conception. Le VHDL est sélectionné comme langue de description matérielle. La période d'horloge de la carte FPGA est donnée par la carte FPGA utilisée et est réglée sur 10 ns. La période du système Simulink doit être le « gcd », le plus grand diviseur commun, pour toute période d'échantillonnage définie sur n'importe quel bloc du modèle. Dans ce modèle, il a été réglé sur 5us.

Le bloc « Synthesis Manager » permet de générer un fichier de programmation pour le FPGA, et c'est à partir de ce bloc que le processus de génération est lancé. Le tableau de développement FPGA utilisé est le Xilinx OP716x PCIe-XSG (périphérique Kintex 7 XC7K325T). Pour générer le fichier de programmation, la case "Enable partition" doit être

cochée, avant de cliquer sur le bouton « Generate programming file », pour que le processus démarre.

Le bloc « version » est le bloc où le nom du fichier « bitstream » qui sera généré est défini. Chaque fois qu'un nouveau flux de bits est généré, le nom de la version doit être modifié pour éviter les problèmes avec les fichiers. Cela peut être fait simplement en donnant à chaque version un nombre et en augmentant chaque fois qu'un nouveau fichier est généré.

Le bloc « Hardware Configuration » décrit la configuration matérielle du module Entrées/Sorties de l'ordinateur cible.

5.2 Simulation avec XSG et Matériel de simulation

5.2.1 Passage de Simulink à Xilinx

Pour être implémenté sur un FPGA, le modèle Simulink du système doit être réadapté. C'est-à-dire que tous les blocs Simulink doivent être remplacés par des blocs équivalents de l'outil XSG. Les modèles de la charge et de l'onduleur seront remplacés par du matériel physique pendant les tests expérimentaux. Cependant, pour tester le nouveau modèle, les modèles Simulink de la charge et l'onduleur ont été utilisés dans la première étape de l'adaptation du modèle, afin de fournir des valeurs pour le courant de charge mesuré. Tous les autres calculs doivent être exécutés sur le FPGA et sont donc reconvertis avec des blocs Xilinx.

5.2.2 Formatage de données avec des blocs Xilinx

5.2.2.1 Les blocs « Gateway In / Out »

Les blocs de la librairie XSG de Simulink fonctionnent avec son propre format de données, qui est un format à virgule fixe. Les signaux Standard Simulink sont formatés en double. Ils doivent donc être convertis en virgule fixe avant d'être appliqués aux blocs Xilinx. La librairie Xilinx possède des blocs pouvant effectuer cette conversion. Notamment, le bloc « Gateway In » est utilisé pour la conversion d'un signal Simulink de format double, à format virgule fixe Xilinx. Un bloc « Gateway Out » est utilisé pour faire la conversion en sens inverse, c'est-à-dire convertir le signal du format point fixe Xilinx au format double à nouveau pour Simulink. Cela signifie que chaque partie du modèle qui sera exécutée sur le FPGA devra se trouver entre les blocs « Gateway In » et « Gateway Out ». Le format virgule fixe des signaux peut être choisi dans les paramètres de configuration du bloc d'entrée « Gateway In ». Lorsqu'on définit le format d'un signal en virgule fixe, le type arithmétique peut être défini comme Signé (Fix) ou Non-Signé (UFix). Dans le cas où le signal peut être à la fois négatif et positif, le format Signé doit être sélectionné. Pour un signal signé, le bit le plus significatif (MSB) sera utilisé pour indiquer le signe du signal. Le nombre de bits ainsi que le placement du point binaire sont également sélectionnés dans les paramètres. Lorsqu'on décide du format qui sera utilisé, deux paramètres doivent être prises en considération. Il s'agit de l'amplitude et la précision du signal. Pour être sûr que la grandeur totale du signal puisse être représentée, il doit y avoir suffisamment de bits dédiés avant le point binaire. La précision qui peut être utilisée pour le signal est déterminée par le nombre de bits après le point binaire. Pour cette simulation, l'ensemble des signaux est défini sous le format

Fix_24_10, ce qui signifie que la donnée est signée, contient 24 bits et a un point binaire avant le 10^{e} bit le moins significatif.

5.2.2.2 Le bloc « Reinterpret »

Le bloc « *Reinterpret* » réinterprète une donnée de son format d'origine à un autre format spécifié par l'utilisateur. En choisissant « force arithmetic type », le type du signal peut être changé en Signé, Non-Signé ou virgule flottante. La position du point binaire peut être modifiée, mais le nombre total de bits reste le même.

5.2.2.3 Le bloc « Slice »

Si l'on souhaite modifier le nombre de bits d'un signal, le bloc « *Slice* » peut être utilisé. Ce bloc extrait des bits spécifiques ou une plage de bits à partir d'un signal. Le nombre de bits à extraire peut être sélectionné, ainsi que le décalage du bit le moins significatif ou du bit le plus significatif.

5.2.2.4 Le bloc « Assert »

Le bloc « *Assert* » est utilisé pour faire valoir un taux et/ou un type de signal. Cela peut être utile dans les situations où l'intervention du concepteur est nécessaire pour résoudre les taux et/ou les types des signaux. Le type peut être configuré, soit signé, soit non signé, et la précision peut être spécifiée. L'assertion du taux d'échantillonnage peut également être effectuée, explicitement ou à partir d'un port d'entrée. Si les taux ou le type à l'entrée des blocs n'est pas le même que celui spécifié, un message d'erreur se produira.

5.2.2.5 Le bloc « Convert »

Le bloc « *Convert* » convertit un signal de son format actuel vers un autre format qui peut être spécifié par l'utilisateur. Les types de données pouvant être sélectionnés pour la sortie sont Booléen, Point-Fixe ou Point-Flottant. Si le format de point fixe est sélectionné, le nombre de bits et le point binaire peuvent être spécifiés par l'utilisateur. Le signal peut également être spécifié sous la forme Signée ou Non-Signée.

5.2.3 Le bloc comprenant les valeurs des composants

Comme mentionné plus haut, les modèles de la charge et de l'onduleur seront remplacés par du matériel dans la configuration expérimentale. Pour cela, le modèle sera ensuite adapté à l'environnement de simulation RT-Lab. Dans RT-Lab, il existe une console qui, lors de l'exécution du système, pourra communiquer avec le reste du modèle. Cela signifie que les valeurs mesurées peuvent être surveillées dans la console, et les variables peuvent également être modifiées pendant la simulation à partir de la console.

5.2.4 Transformation des coordonnées

Les courants de consignes et les courants mesurés doivent être transformés de coordonnées abc en coordonnées $\alpha\beta$ avant d'être appliqués à la fonction de contrôle prédictif. Cette transformation est connue sous le nom de transformée de Clarke. La méthode de transformation des coordonnées abc à $\alpha\beta$ implémentée avec XSG est très similaire à celui du modèle Simulink. Tous les blocs Simulink ont été remplacés par des blocs de la librairie Xilinx, comme on peut le constater à la Figure 5.1. La précision des données en sortie des blocs peut être spécifiée par l'utilisateur.

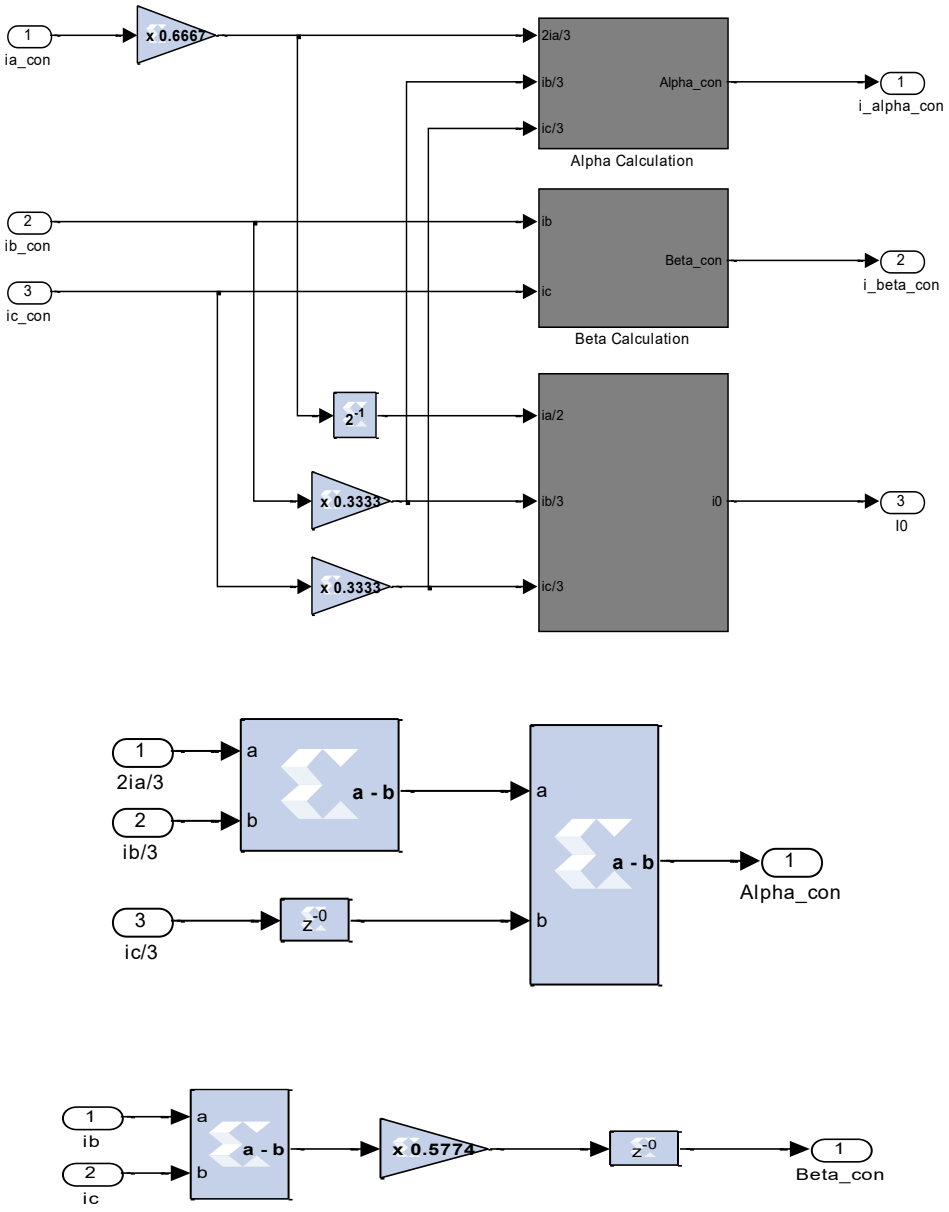


Figure 5.2 Transformation des coordonnées – (abc à αβ)

5.2.5 Le bloc de Mcode et l'algorithme de contrôle prédictif

Le bloc Xilinx MCode est un conteneur permettant d'exécuter une fonction MATLAB fournie par l'utilisateur dans Simulink. Comme mentionné précédemment, l'objectif est d'exécuter l'algorithme de contrôle prédictif sur le FPGA compte tenu de son nombre élevé de calcul. Pour ce faire, l'algorithme de prévision doit être adapté avec de nouveaux éléments

composés de blocs Xilinx. Un bloc Mcode est utilisé à cette fin. Elle exécute une fonction Matlab fournie par l'utilisateur, et le lien vers le fichier m. doit être défini comme paramètre dans le bloc. Lorsque le matériel est généré, le code est traduit de manière simple pour donner le même comportement dans le langage Description matérielle (HDL).

La Figure 5.3 présente le bloc Mcode, qui est une fonction prenant en entrée l'estimation de la fonction à minimiser pour chaque vecteur de tension et donne en sortie les signaux de commutations optimaux x_opt et la fonction de coût optimale, g.

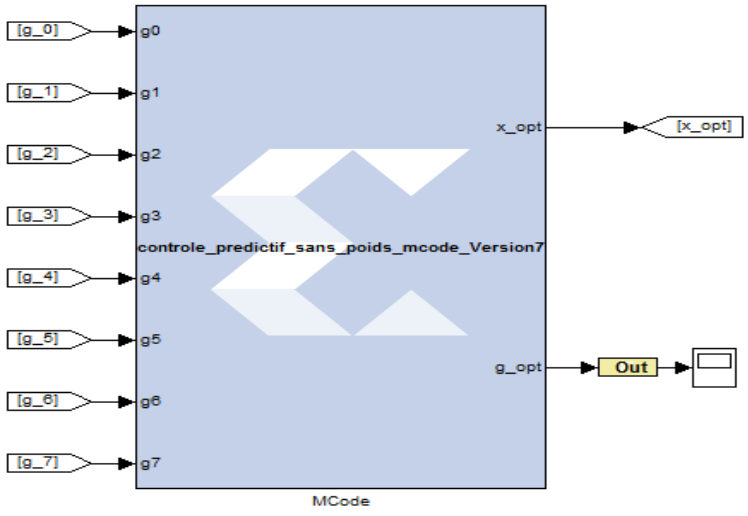


Figure 5.3 Bloc Mcode

Compte tenu de la difficulté à contrôler et gérer les délais dans le bloc Mcode, l'algorithme a été décomposé, parallélisé et remplacé par des opérations de calcul à l'aide des blocs de la librairie Xilinx. Avec cette nouvelle approche, le contrôle et la gestion des délais dans l'algorithme sont plus faciles. Ces délais permettent la synchronisation des données entre Matlab/Simulink et le système FPGA.

5.2.6 Calcul de la f.c.é.m.

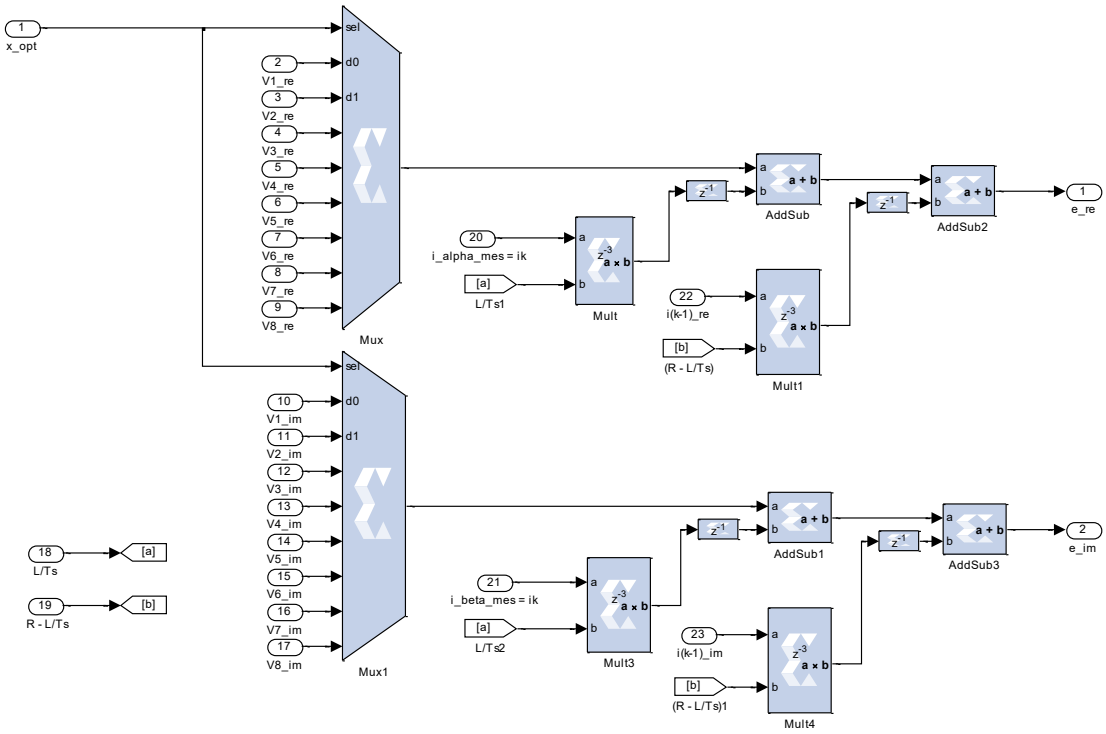


Figure 5.4 Calcul de la f.c.é.m. dans le plan αβ

Les calculs pour la f.c.é.m. de la charge en sortie, qui ont été utilisés dans l'algorithme de contrôle prédictif, sont effectués dans un bloc séparé. Les calculs des parties réelles et imaginaires s'effectuent séparément, mais de façon similaire avec une différence des vecteurs de tension en entrée des multiplexeurs. La Figure 5.4 montre le schéma des calculs des parties réelle et imaginaire de la f.c.é.m. Les entrées sont les valeurs des composants, les valeurs de tension, du signal d'état de commutation, x_opt, ainsi que les courants $\boldsymbol{i}\ (k)$ et $\boldsymbol{i}\ (k+1)$.

L'équation (13) utilisée pour obtenir le courant prédit de la charge peut être réarrangée pour donner une expression de la f.c.é.m.

$$\hat{\boldsymbol{e}}(k) = \boldsymbol{V_0}(k) - \frac{L}{T_s}\boldsymbol{i_0}(k+1) - \left(R - \frac{L}{Ts}\right)\boldsymbol{i_0}\ (k) \tag{24}$$

En tenant compte de l'hypothèse $\boldsymbol{e}\ (k+1) \approx \boldsymbol{e}\ (k)$, les courants dans l'équation peuvent être décalés d'un échantillonnage dans le temps. $\boldsymbol{i_0}\ (k+1)$ correspond alors au courant mesuré à l'instant k, et $\boldsymbol{i_0}\ (k)$ correspond au courant mesuré à l'instant $(k-1)$. Cela signifie que le port d'entrée nommé i_meas_alpha dans la Figure 5.4 prendra la composante α du courant mesuré comme entrée. Le signal alimentant le port d'entrée marqué $\boldsymbol{i}(k-1)$ dans la figure est le même courant mesuré, mais un bloc de registre est utilisé pour le retarder d'une étape de temps d'échantillonnage avant de passer au port d'entrée.

5.2.7 Le bloc de sélection des états

Ce bloc donne l'état de commutation optimal en sortie, désigné par x_opt, et la fonction de coût optimale g. Un bloc « *Slice* » est utilisé pour changer le signal x_opt en un format Ufix_3_0, qui est requis avant d'entrée sur dans le bloc « *Mux* ». Pour pouvoir utiliser les blocs de communication qui traitent la communication de sortie numérique avec le FPGA lorsque le matériel est inclus, le signal doit être d'un format UFix_1_0. Un bloc de réinterprétation « Reinterpret » est utilisé pour changer le signal sur non signé, avant qu'un bloc Slice extrait le bit pertinent. Le signal peut ensuite être transmis au modèle de l'onduleur Simulink, via un bloc Gateway Out.

La Figure 5.5 présente le contenu du bloc « *Switch_Selector »* pour chacune des phases a, b et c. Selon la valeur de x_opt, la bonne entrée droite du Multiplexeur est sélectionnée et transmise à la sortie.

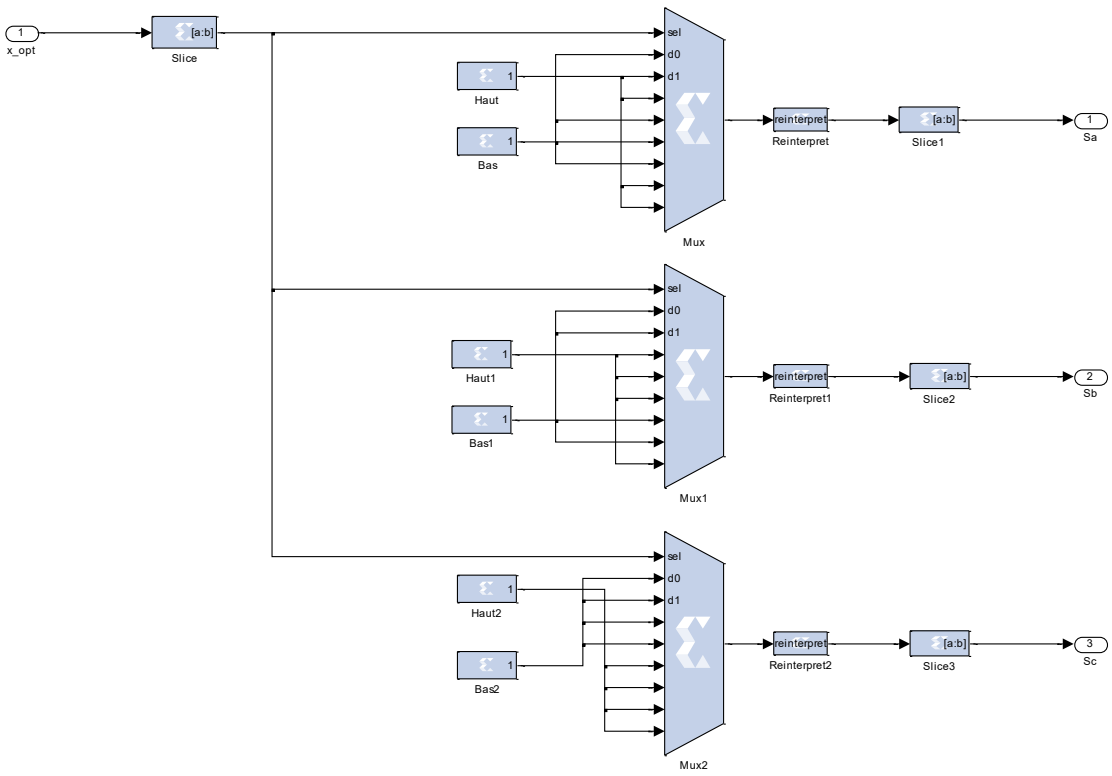


Figure 5.5 Génération des signaux de commutation des trois phases

5.2.8 Génération de vecteurs de tension

Au Chapitre 2, dans le Tableau 2-1, les huit états de commutation valides ainsi que les vecteurs de tension de sortie générés correspondants pour l'onduleur ont été présentés. Les vecteurs de tension sont divisés en deux variables, une pour la partie réelle et une pour la partie imaginaire, comme le montre la Figure 5.6. Le calcul des vecteurs de tension $V3$ alpha et beta est aussi présenté.

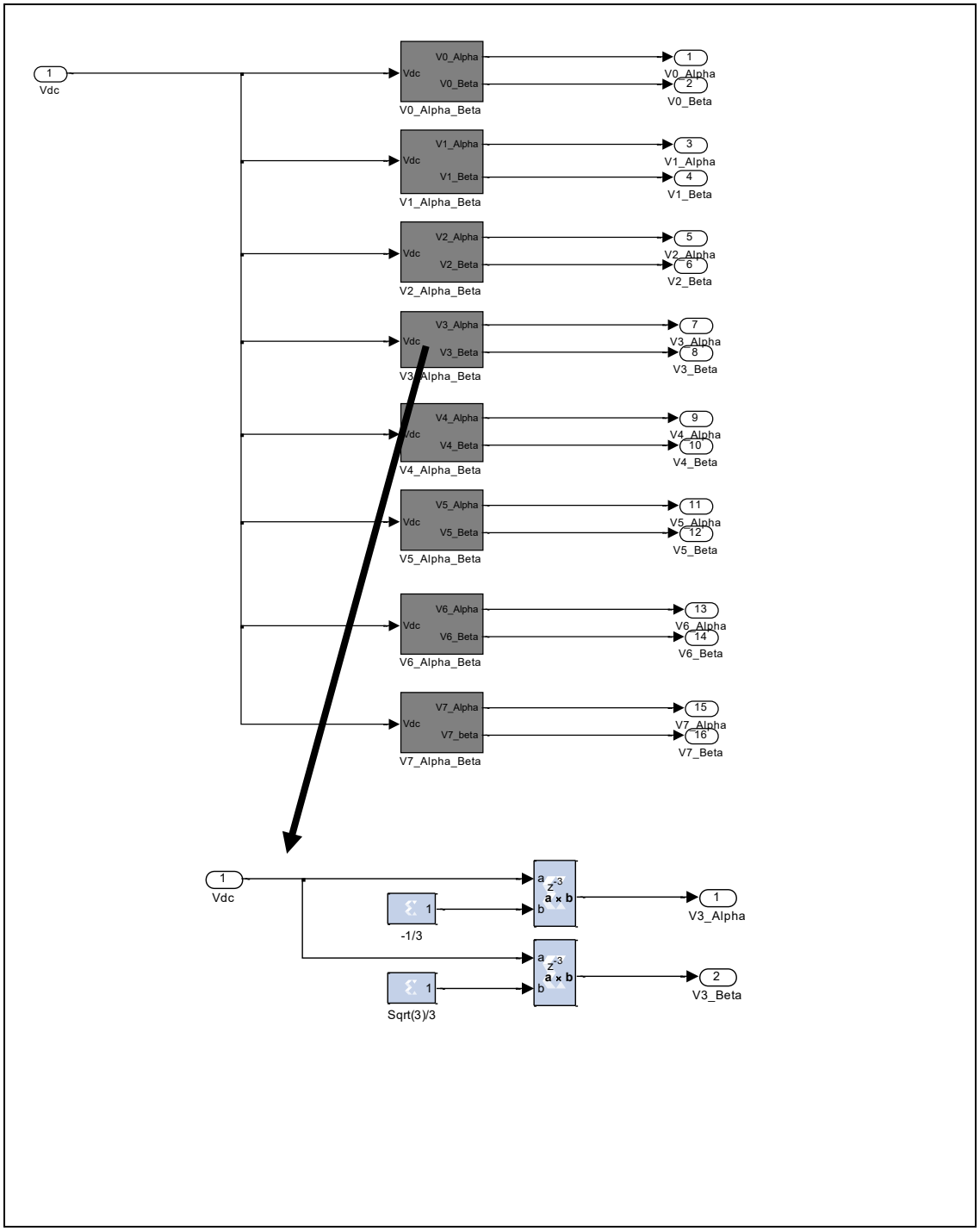


Figure 5.6 Calcul des vecteurs de tension

5.2.9 Les résultats de simulation avec XSG et matériel de simulation

5.2.9.1 Paramètres du modèle

La simulation a été effectuée avec les pas de temps de 25 μs et 1 μs. Le Tableau 4-1 du chapitre précédent présente les valeurs des paramètres définies pour le modèle du système. Le fichier des paramètres de simulation .m doit être exécuté pour fournir des valeurs au modèle de la charge.

5.2.9.2 Simulation XSG avec une période d'échantillonnage de 25 µs sans délais

La Figure 5.7 et la Figure 5.8 présentent les résultats de suivi courant-consigne / courant-mesuré et de la fonction de coût respectivement lorsqu'aucun délai n'est introduit dans le modèle du système implémenté sur FPGA. La période d'échantillonnage est de 25 μs.

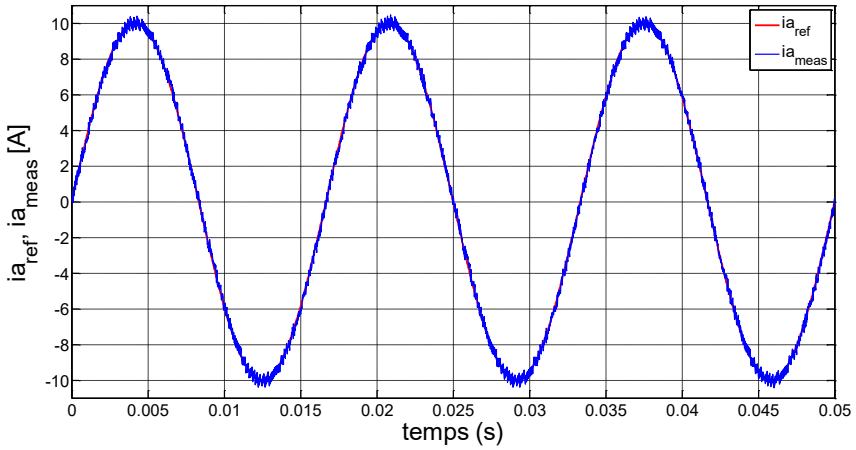


Figure 5.7 Courant consigne et courant de charge mesuré à 25μs sans délais

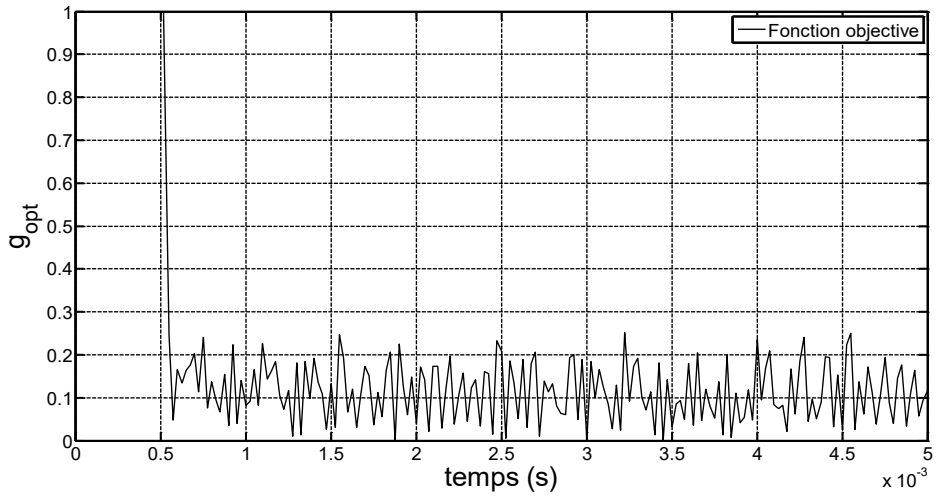


Figure 5.8 Fonction de coût, g à 25μs sans délais

5.2.9.3 Simulation XSG avec une période d'échantillonnage de 1 µs sans délais

La Figure 5.9 et la Figure 5.10 présentent les résultats de suivi courant-consigne / courant-mesuré et de la fonction de coût respectivement lorsqu'aucun délai n'est introduit dans le modèle du système implémenté sur FPGA. La période d'échantillonnage est de 1 μs.

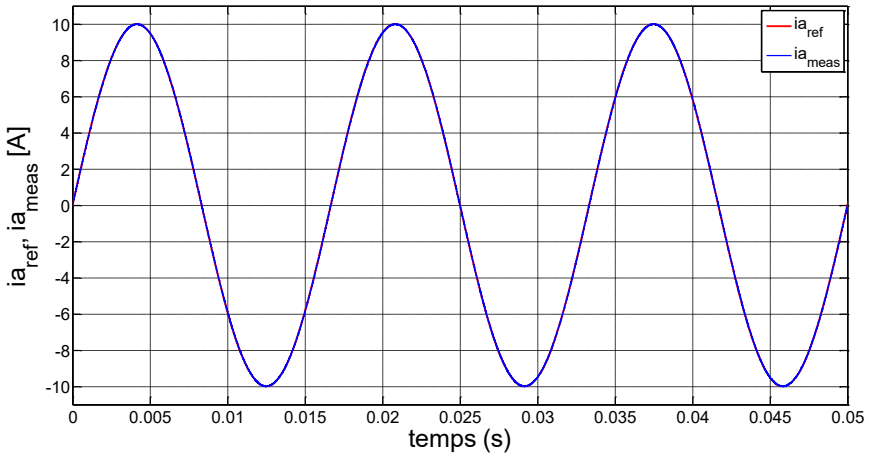


Figure 5.9 Courant consigne et courant de charge mesuré à 1μs sans délais (pas de synchronisation)

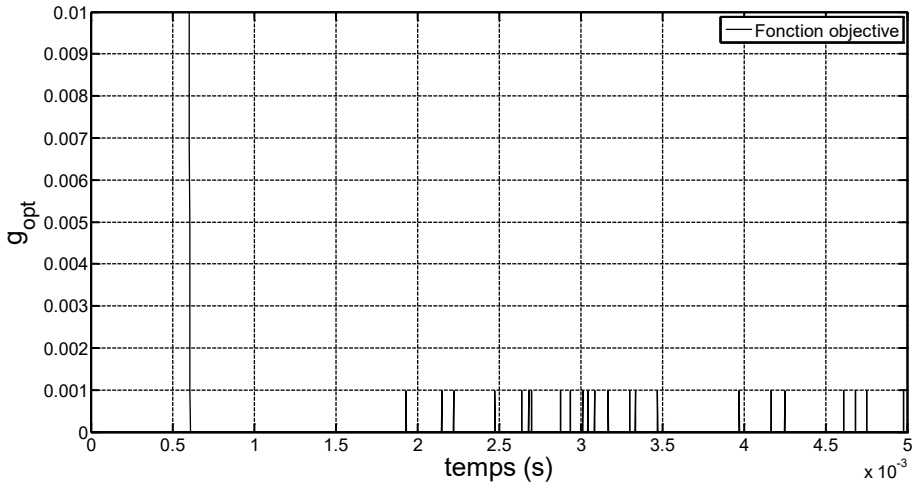


Figure 5.10 Fonction de coût, g à $1\mu s$ sans délais (pas de synchronisation)

5.2.9.4 Simulation XSG avec une période d'échantillonnage de 25 µs avec délais(synchronisation)

La Figure 5.11 et la Figure 5.12 présentent les résultats de suivi courant-consigne / courant-mesuré et de la fonction de coût respectivement avec délai. La période d'échantillonnage est de 25 μs.

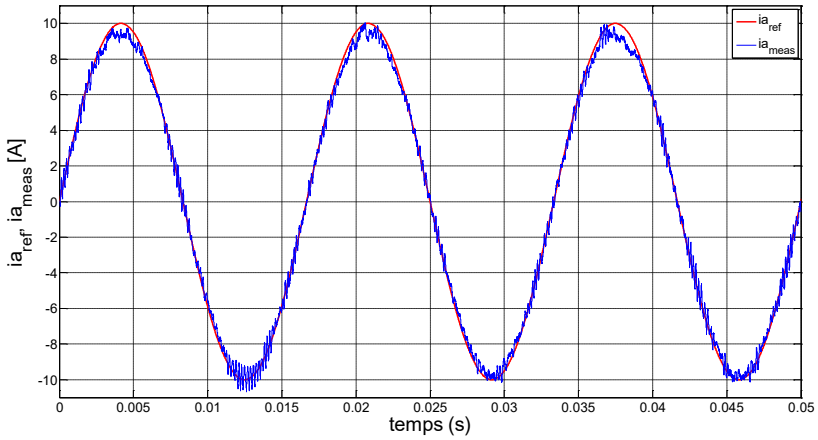


Figure 5.11 Courant consigne et courant de charge mesuré à $25\mu s$ avec délais(synchronisation)

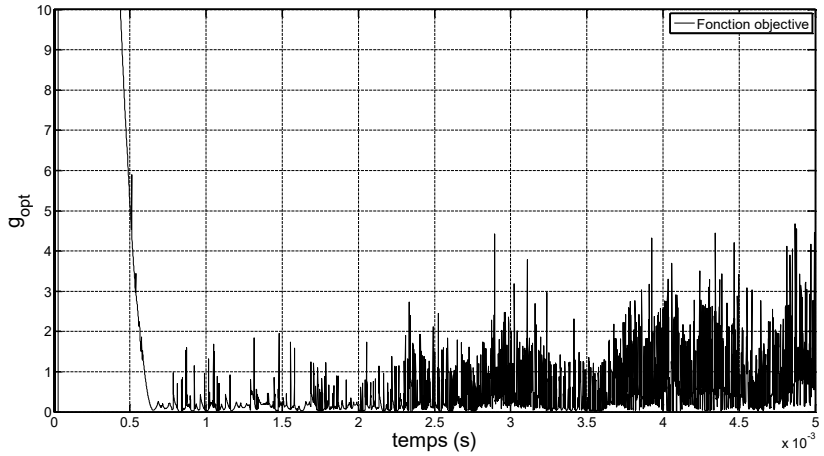


Figure 5.12 Fonction de coût, g évalué à 25μs avec délais(synchronisation)

5.2.9.5 Simulation XSG avec une période d'échantillonnage de 1 µs avec délais

La Figure 5.13 et la Figure 5.14 présentent les résultats de suivi courant-consigne / courant-mesuré et de la fonction de coût respectivement avec délai. La période d'échantillonnage est de 1 μs.

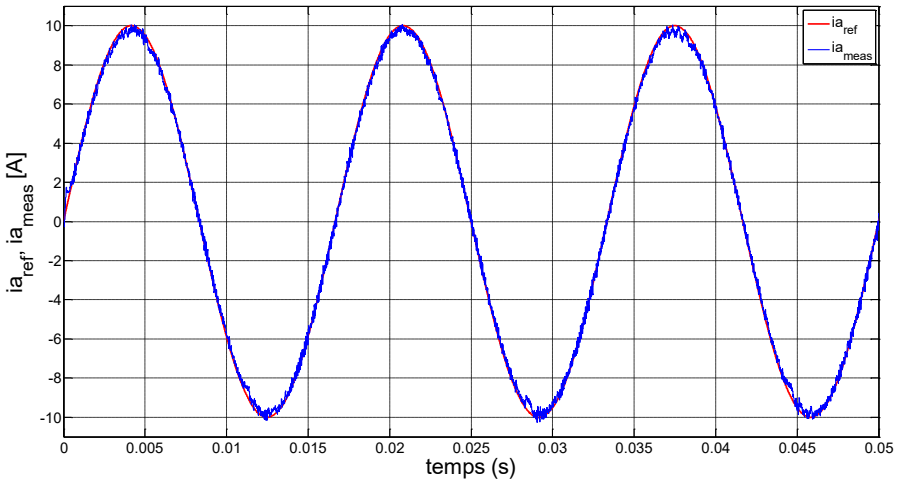


Figure 5.13 Courant consigne et courant de charge mesuré à 1μs avec délais(synchronisation)

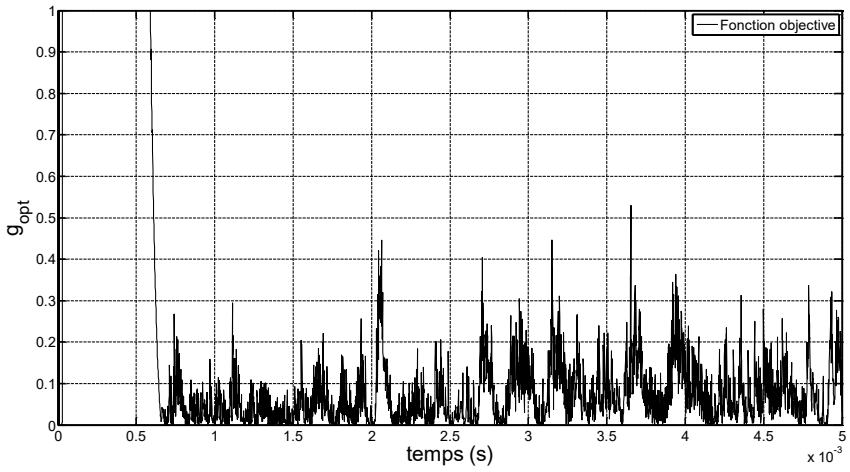


Figure 5.14 Fonction de coût, g à $1\mu s$ avec délais (synchronisation)

5.2.10 Interprétation et analyse des résultats

D'après les résultats de simulation ci-dessus, on peut constater qu'un suivi précis du courant de consigne est réalisé. Ce suivi du courant de référence est très précis lorsque la fréquence de commutation est élevée. Il est encore plus précis lorsqu'aucun délai n'est introduit dans le modèle. Malheureusement, le modèle nécessite des délais pour fonctionner sur un FPGA et éviter des « Timing Error ». On peut aussi constater que le courant mesuré du modèle XSG possède plus d'ondulation que celui de Simulink. En comparant la fonction de coût aux résultats de la simulation de contrôle de courant avec le modèle Simulink, on constate que la fonction de coût aura une valeur plus élevée. La fonction de coût du modèle Simulink, ayant la même fréquence d'échantillonnage de $1\ \mu s$, avait une valeur moyenne d'environ 0.015. En étudiant la Figure 5.14, la valeur maximale se situe autour de 0.4. Les variations de la valeur de la fonction de coût sont également plus importantes dans le modèle Xilinx. Les deux modèles fonctionnent avec les mêmes paramètres et la fréquence d'échantillonnage, en effectuant uniquement le contrôle de courant de l'onduleur.

Une première raison de la différence de performance du modèle Xilinx, par rapport au modèle de Simulink, pourrait s'expliquer par le fait qu'un certain retard ou délai est introduit dans le modèle en raison des principes de fonctionnement du FPGA [20]. Les blocs sont synchronisés par le signal d'horloge et l'algorithme de contrôle prédictif implémenté dans le bloc Mcode, sera traduit en langage VHDL, qui décidera comment exécuter la fonction sur une carte physique. Cela peut provoquer un léger décalage des signaux, ce qui peut entrainer une augmentation d'erreur entre les valeurs désirée et obtenue.

Une seconde raison de la différence entre le modèle Simulink et Xilinx est que les blocs Xilinx fonctionnent avec des données au format virgule fixe. Cela limite la précision des signaux et peut affecter les performances dans une certaine mesure. La précision utilisée pour différentes données doit être évaluée, car un nombre plus élevé de bits et une meilleure précision se font au prix de plus de données à traiter, ce qui encore une fois nécessitera un temps de calcul plus long.

Une troisième raison pourrait être la boucle de rétroaction du modèle après la propagation des données, car le modèle possède plusieurs délais. Indépendamment des performances moindres par rapport au modèle Simulink, le suivi du courant de consigne du modèle Xilinx est toujours très bon.

5.3 Conclusion

Ce chapitre a présenté les plateformes logicielles qui sont utilisées pour implémenter le modèle du système sur FPGA. Le modèle Simulink du chapitre précédent a été transcrit avec les blocs propres à ces plateformes et les résultats des simulations analysés et comparés. Le chapitre suivant présentera l'adaptation du modèle à l'environnement de simulation temps

réel RT-Lab, qui convient à travailler avec du matériel physique connecté (version émulateur de ce système).

Chapitre 6 - RT-Lab et implémentation expérimentale via un émulateur

6.1 Adaptation du modèle dans l'environnement RT-Lab

Dans le chapitre précédent, un modèle de l'algorithme de prévision a été implémenté à l'aide de la librairie Xilinx, ce qui permet de générer les fichiers nécessaires à l'exécution du modèle sur un onduleur réel. Les modèles Simulink pour l'onduleur et la charge ont été utilisés pour déterminer les courants mesurés en sortie, et un sous-système « Paramètres » a été utilisé comme interface utilisateur pour le réglage des variables clés. Pour substituer la charge et l'onduleur simulées par du matériel physique, le modèle doit être une nouvelle fois modifié pour contenir certains blocs de communication avec le matériel. La connexion physique à l'onduleur et au FPGA sera réalisée en ajoutant des blocs personnalisés dans le modèle, qui gèrent l'interface des périphériques d'Entrées/Sorties. Dans le modèle précédent, le sous-système « Paramètres » était utilisé pour spécifier les valeurs des variables. Cependant, il doit être adapté à un environnement capable de communiquer avec le FPGA pendant le déroulement de la simulation, afin d'ajuster facilement les variables. Pour analyser le comportement du système pendant la simulation, il est important d'avoir accès aux mesures de courant de sortie et aux variables de performance. L'environnement de simulation RT-Lab, présenté au Chapitre 5, rend cette configuration possible.

Tout modèle Simulink peut être adapté dans l'environnement RT-LAB, mais certaines modifications doivent être effectuées afin de repartir le modèle et le transférer dans l'environnement de simulation. La réalisation de calcul distribué d'un modèle complexe dépend de la séparation de ce modèle en petits sous-systèmes synchronisés pour fonctionner

en parallèle. Pour adapter le modèle dans l'environnement RT-LAB, l'utilisateur doit modifier le diagramme (fichier.mdl) du modèle, en le séparant en sous-systèmes et insérant des blocs de communication « OpComm ». Lorsque le modèle est divisé en sous-systèmes, certaines étapes doivent être suivies. Une console doit être créée et le nom de ce sous-système doit commencer par le préfixe « SC_ ». La console sert d'interface utilisateur avec le système pendant la simulation. Elle contient tous les blocs Simulink liés à l'acquisition et à la visualisation des données (oscilloscope, commutation manuelle, etc.). Elle est utilisée pour afficher les signaux acquis et donne la possibilité de définir des signaux de contrôle pendant la simulation. La console peut être utilisée à partir de la station de commande. Elle remplacera le sous-système « Paramètres » ainsi que les outils de visualisation utilisés pour surveiller les signaux dans le modèle Simulink/Xilinx.

De même, tout modèle doit également intégrer un sous-système appelé maître, avec un nom commençant par le préfixe « SM_ ». Il est responsable du calcul en temps réel du modèle et de la synchronisation globale du réseau. Chaque modèle ne peut comporter qu'un seul maître. En plus du maître, plusieurs sous-systèmes esclaves peuvent être ajoutés au modèle pour distribuer les calculs sur plusieurs cœurs dans le simulateur. Dans ce modèle, la plupart des calculs sont effectués sur le FPGA et un seul cœur est utilisé dans le simulateur. La tâche principale du sous-système maître dans le modèle RT-Lab sera la communication d'Entrées/Sorties entre le matériel et le FPGA, vers la station de contrôle et la console. La Figure 6.1 illustre l'adaptation du modèle Simulink dans l'environnement RT-Lab.

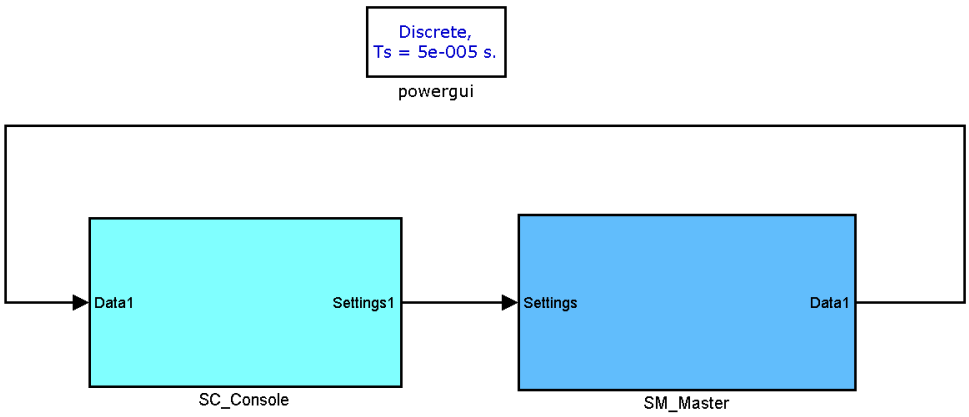


Figure 6.1 Adaptation du modèle Simulink dans l'environnement RT-Lab

6.2 La console

Une fois que le modèle est subdivisé en sous-systèmes *console* et *master*, les blocs spéciaux appelés « OpComm » doivent être inclus dans un modèle RT-Lab. Les blocs « OpComm » interceptent tous les signaux entrants avant de les envoyer aux blocs de calcul dans un sous-système donné. Aucun signal ne peut entrer dans un sous-système sans passer d'abord par ce bloc. Le bloc « OpComm » sert à plusieurs objectifs: transmet au simulateur RT-Lab des informations sur la taille et le type des données transférées d'un sous-système à un autre, définit les groupes et les paramètres d'acquisition de données, il s'assure que toutes ses entrées soient mises à jour avant de mettre à jour ses sorties, spécifie le temps d'échantillonnage pour le sous-système dans lequel il se trouve et définit le temps d'échantillonnage de communication d'un sous-système de calcul avec un autre sous-système de calcul.

Il convient de rappeler que la communication entre la console et les sous-systèmes de calcul sur FPGA n'est pas en temps réel. Pour exécuter en même temps plusieurs sous-systèmes sur le simulateur RT-LAB, chaque sous-système doit avoir un bloc « OpComm »

pour la communication en temps réel entre les cœurs, et un bloc « OpComm » pour la communication avec la console.

Pour ce modèle, le bloc « OpComm » dans le bloc console reçoit comme données en entrée les courants de charge mesurés en sortie et la fonction de coût. Ces signaux sont ensuite transmis via un autre bloc « OpComm », puis envoyés pour la visualisation, où les valeurs peuvent être analysées et sauvegardées dans l'espace de travail. Les sorties de la console incluent les courants de référence triphasés, la tension de la source CC, V_{dc}, les valeurs des composants données comme (T_s/L) et R. Au cours de l'exécution de la méthode de contrôle, ces variables peuvent être modifiées. Ceci est un gain de temps important par rapport aux simulations en cours d'exécution dans Simulink, car de nombreuses valeurs et combinaisons peuvent être testées pour les variables clés en temps réel, et l'impact peut être analysé directement.

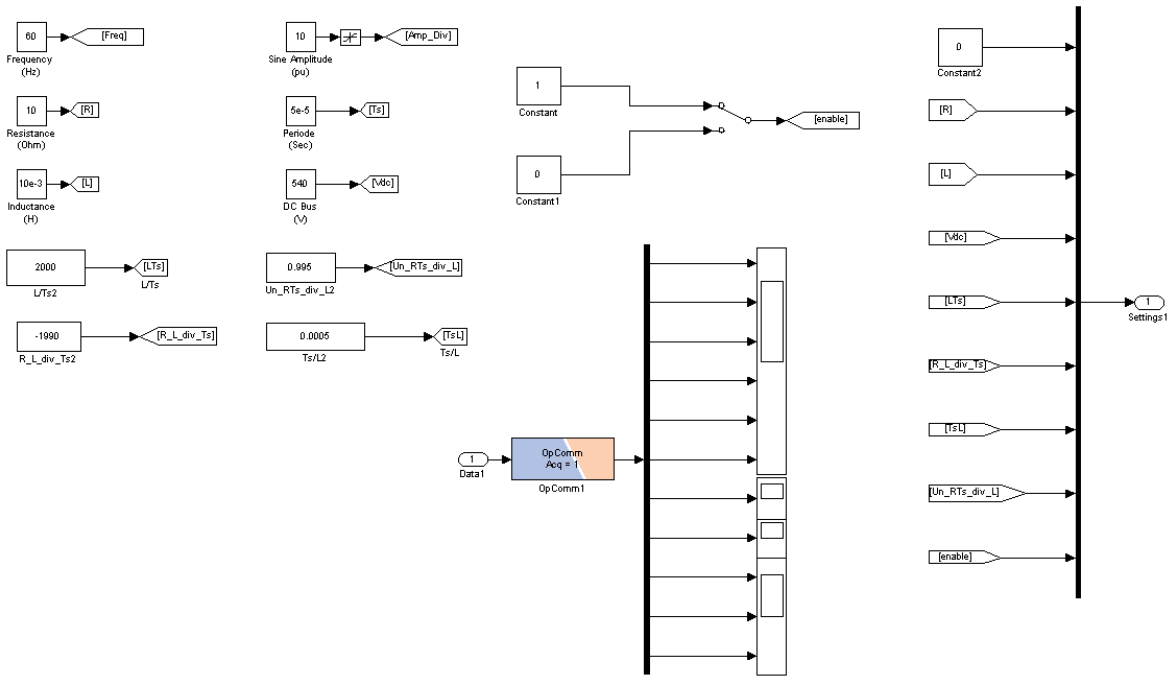


Figure 6.2 Sous-système Console

6.3 Le Master

Les échanges de données entre le sous-système console et le modèle FPGA se font à travers le sous-système « master » de RT-Lab, en utilisant les blocs spécialisés tels que DataOut Recv, DataIn Send, DataIn et DataOut. La Figure 6.3 présente un aperçu de ce sous-système. En général, les blocs spécialisés sont utilisés par RT-Lab et RT-XSG pour effectuer les échanges de données entre le CPU et le modèle FPGA. Le bloc l'OpCtrOP7160EX1, présenté à la Figure 6.4, est le bloc le plus important, car il permet de gérer la programmation d'une plate-forme de conception FPGA OP7160. Ce bloc est présenté dans la figure ci-dessous. Il permet également d'initialiser et de choisir le mode de synchronisation matérielle, ainsi que de lier les blocs d'envoi et de réception et d'Entrées/Sorties à cette carte en particulier. Dans les paramètres de ce bloc, le nom de fichier « bitstream » généré par le bloc « Xilinx Synthesis Manager », doit être indiqué. Ce fichier .bin sera appliqué au FPGA. Le fichier doit être placé dans le même répertoire que le modèle RT-Lab de la console.

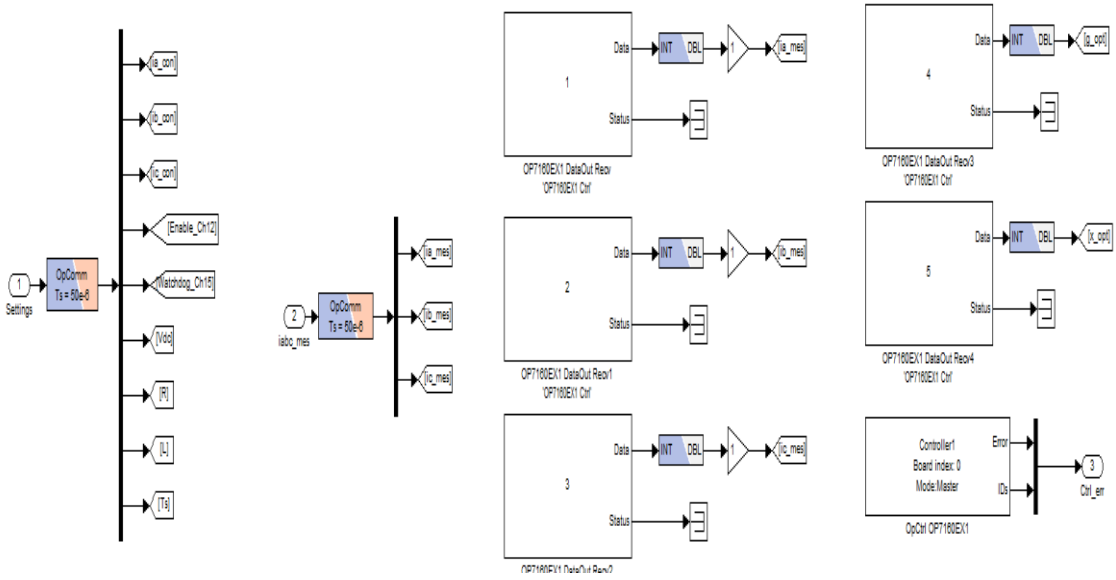


Figure 6.3 Aperçu du sous-système « master »

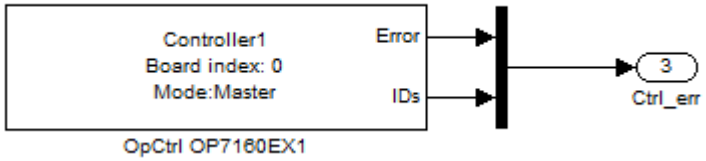


Figure 6.4 Bloc OpCtrlOP7160EX1

Dans le sous-système master se trouvent les blocs d'envoi et de réception, qui sont du type « OP7160EX1 DataIn Send » et « OP7160EX1 DataOut Recv ». Les signaux d'entrée de la console passent par un bloc de communication « OpComm », avant d'être envoyés à un bloc « DataIn Send ». Le bloc « DataIn Send » communique avec un bloc « DataIn » dans le modèle FPGA. Les formats de données double et uint32 sont acceptés en entrée et peuvent être configurés dans les paramètres du bloc. Tous les signaux d'entrée sont initialement en format double, mais seuls les signaux d'activation sont envoyés aux blocs « DataIn Send » dans ce format. Les autres signaux sont convertis au format uint32 à l'aide d'un bloc de reconversion de double à entier. Dans ce bloc, le nombre total de bits, le point binaire et le type de format numérique peuvent être spécifiés. Ceci est fait dans le but de contrôler le signal, afin de le retransférer avec la bonne valeur après son transfert vers le bloc « DataIn » dans le modèle FPGA. Le bloc « DataIn » effectue la conversion des données de uint32 vers le format de données UFix33_0 du Générateur de système. À partir de ce format, les données souhaitées doivent être extraites des 32 bits les moins significatifs, le point binaire doit être défini si nécessaire et le signe du signal doit être défini. Des opérations simples sont ajoutées dans le sous-système maître, pour le calcul des signaux de sortie, comme le montre la Figure 6.5 ci-dessous :

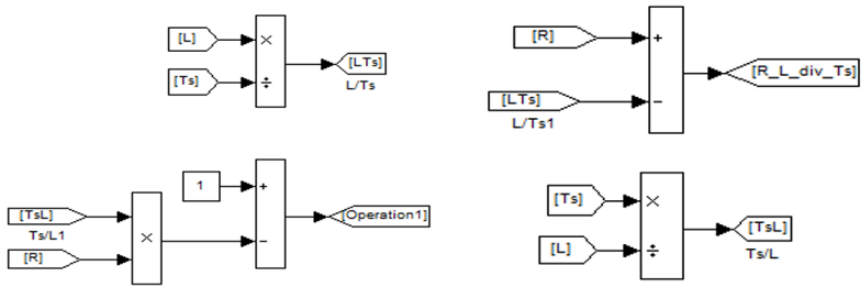


Figure 6.5 Opération dans le sous-système master (SM_)

Les courants mesurés, les signaux de commutation, la fonction de coût sont définis comme signaux d'entrée dans le sous-système master. Le but est de les transférer par la suite au sous-système console pour l'analyse et la sauvegarde. Cet échange de données se fait par l'utilisation de blocs « DataOut Recv ». Ces derniers reçoivent leurs signaux à partir du bloc « DataOut » placé dans le modèle du contrôleur sur FPGA. Les données en sortie de ce bloc peuvent être configurées en format double ou uint32. La Figure 6.6 montre un exemple de l'un des blocs « DataOut Recv » du modèle et le traitement des données avant leur envoi à la console.

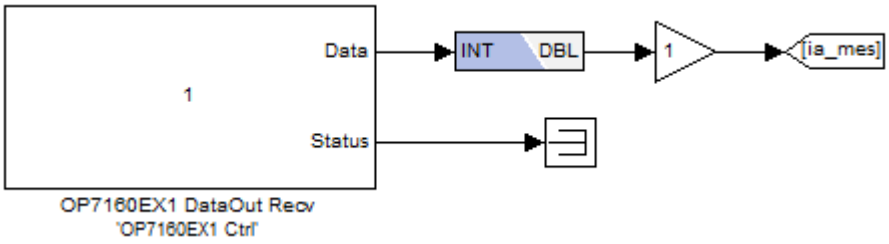


Figure 6.6 Bloc DataOut Recv

Le traitement des informations après leur réception par le bloc « DataOut Recv » est fonction du format du signal à l'entrée du bloc « DataOut » envoyant les valeurs. La Figure 6.6 présente un exemple; dans ce cas, le courant mesuré de la phase a. Ici, le signal a, à l'origine, le format Fix_24_10 avant d'être envoyé au bloc « DataOut », et pour cette raison,

cette combinaison de bits est utilisée dans les paramètres du bloc « Rescale », le reconvertissant dans la bonne échelle de représentation.

6.4 Blocs de communication dans le modèle FPGA

6.4.1 Le Bloc « DataOut »

Le modèle contient des modules pour la communication avec la console et des modules pour la communication avec le système physique. La communication avec le système physique se fait par l'intermédiaire des ports d'entrées pour les données d'entrée analogique et les ports de sorties pour ce qui est des données de sortie numérique. Le bloc « DataOut » envoie des données au bloc « DataOut Recv » dans le sous-système maître, avant leur transfert vers la console. Les entrées du bloc « DataOut » se composent des trois courants de phase mesurés, de la valeur de la fonction de coût g, et le vecteur de tension optimal x_opt. La Figure 6.7 présente les blocs « AnalogIN » et « DataOUT ».

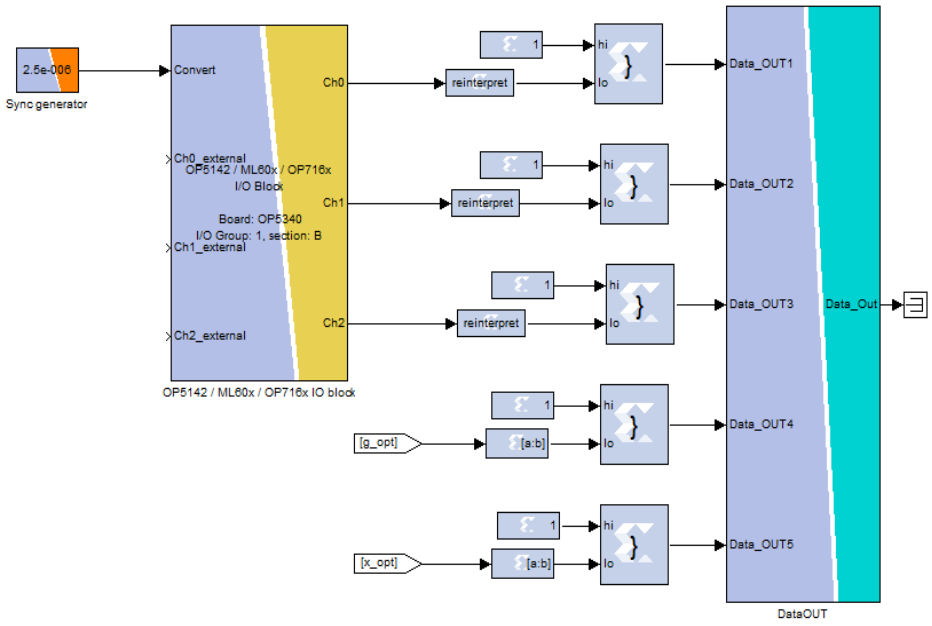


Figure 6.7 Blocs AnalogIn et DataOUT

Chacun des ports d'entrée du module « DataOut » est au format UFix33_0 où les 32 premiers bits représentent les données et le bit 33 (bit le plus significatif) est le signal valide indiquant quand les informations sont mises à jour. Le bit 33 peut être vu comme un signal d'écriture dans le tampon, que ce soit un registre ou un FIFO, dans le bloc DataOUT. Chacun de ces tampons est vidé et transféré vers le modèle de CPU au début de chaque étape de calcul. Un bloc « Constant » est utilisé pour configurer le premier bit à 1, et le bloc de concaténation « Concat » fusionne ce bit avec le reste du signal envoyé à l'entrée. Le point binaire doit être supprimé de chaque signal avant le bloc « DataOut ». Pour les courants mesurés, les données arrivent du bloc d'entrées analogiques au format Fix_24_10. Un bloc de réinterprétation est utilisé pour convertir le type arithmétique à Non-signé et de définir le point binaire sur la position 0. Les données sont ensuite envoyées à l'entrée au format UFix_24_0.

Le port de sortie « DataOUT » du module « DataOut » est utilisé uniquement pour la simulation hors ligne et n'a pas besoin d'être connecté. C'est un vecteur de 16 signaux au format uint32. Chacun de ces 16 signaux représente un port de sortie sur le bloc reconfigurable « OpCtrlReconfigurableIO » dans le modèle de CPU RT-LAB.

6.4.2 Le bloc « Analog Input »

La Figure 6.7 présente également le module d'entrée analogique utilisé pour recevoir les courants mesurés de la carte d'interface entre le FPGA et l'onduleur. Il s'agit d'un module Xilinx de type entrées/sorties OP5142/ML605/OP716x. Ce module donne accès à tous les modules d'entrées/sorties contrôlés par la carte OP5142, la carte de développement ML605 et la carte OP716x. Le bloc « Hardware Synchronization », introduit au Chapitre 5.1.2, détermine tous les modules d'interface disponibles, en se basant sur le type et la direction du

signal demandé. Le type de signal peut être configuré comme analogique ou numérique, tandis que la direction est soit entrée ou sortie. Les ports d'entrée marqués avec « Chx_External » peuvent être utilisés pour la simulation hors ligne. Il convient de noter que les ports d'entrée et de sortie des blocs dépendent de la carte d'interface sélectionnée.

À l'entrée du bloc « Analog Input » (« Convert »), un bloc « Synch Generator » est connecté. Ce bloc génère un train d'impulsions de synchronisation avec la période spécifiée. La largeur de l'impulsion est égale à la période d'horloge de la carte FPGA et donne un signal de sortie du type entier Non-Signé avec la valeur 1. Une période de 2.5 μs est sélectionnée. Les sorties analogiques délivrent des signaux reçus du module de conversion analogique-à-numérique. Les données aux ports de sortie sont au format Fix_24_10.

6.4.3 Le bloc de sortie numérique (digital output block)

Le bloc, utilisé pour la communication des sorties numériques, est du même type que celui utilisé pour l'entrée analogique. Cependant, la sélection du type d'Entrées/Sorties est définie comme sortie numérique. La sélection de l'interface matérielle est configurée sur « Emplacement #4 du panneau avant, section B ». Ce bloc gère la communication entre le FPGA et la carte d'interface qui délivre les signaux de commutation à l'onduleur. Les données d'entrée dans ce bloc doivent être du type UFix_1_0. Pour les signaux de commutation arrivant du sous-système de sélection des états dans le modèle, ceci est réalisé en passant d'abord les données à travers un bloc « Reinterpret » pour changer le format des données en Non-Signé. Un bloc Slice est ensuite utilisé pour extraire le seul bit pertinent. Le Tableau 6-1 présente un aperçu des canaux du port d'entrée auquel les différents signaux doivent être connectés. Les autres ports d'entrée ne sont pas utilisés et reçoivent une entrée constante de 0.

Canal	Signal
0	S_a
1	S_b
2	S_c
3	S_a *inversé*
4	S_b *inversé*
5	S_c *inversé*

Tableau 6-1 Port d'entrée du bloc « Digital Out »

6.4.4 Le bloc « DataIn »

Ce bloc représente la liaison d'entrée du FPGA via le bus de connexion des signaux. Les données peuvent provenir du modèle CPU ou d'un FPGA. Seize ports d'entrée sont fournis à l'utilisateur pour les échantillons de données et les transferts de signaux de contrôle. L'une des fonctions de ce bloc est d'effectuer la conversion des données de uint32 au format de données « System Generator » UFix33_0. Il appartient à l'utilisateur d'extraire les données souhaitées parmi les 32 bits les moins significatifs et de réinterpréter ces bits au format souhaité (signé ou non signé avec ou sans point binaire).

Chaque signal d'entrée passe par le même bloc pour le reformatage des données. Le nombre spécifique de bits et la position du point binaire dépendent de ce qui a été spécifié dans le modèle CPU. La Figure 6.8 montre un exemple de mise en œuvre des valeurs d'entrée.

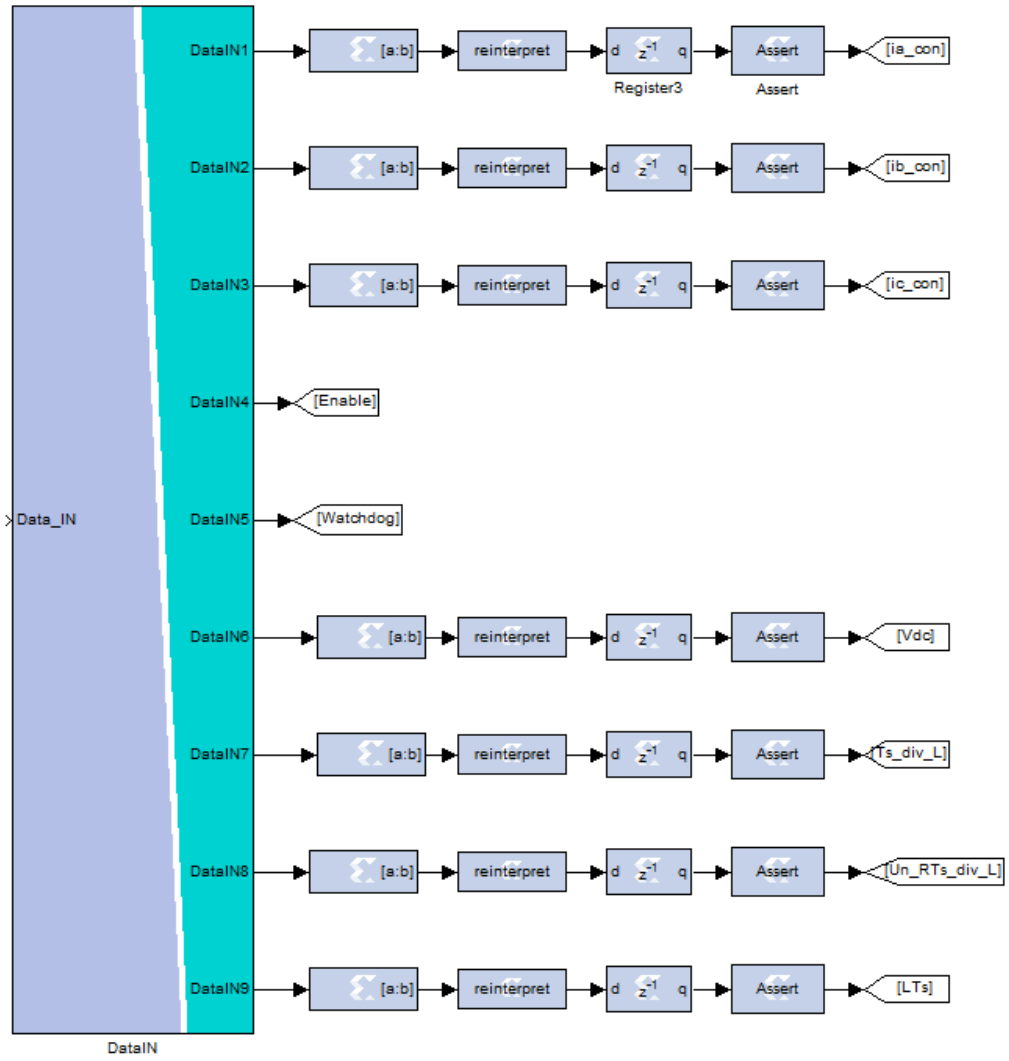


Figure 6.8 Bloc « DataIN » et signaux de sortie

Le signal de sortie du bloc « DataIn » est au format UFix_33_0. Un bloc Slice est utilisé pour extraire les 24 bits pertinents et délivre un signal UFix_24_0. En utilisant un bloc « Reinterpret », le nombre de bits fractionnaires est défini sur dix (10) et le format est modifié en Signé. Il en résulte un signal Fix_24_10. Le bloc « Register » est appliqué à chaque entrée du modèle, pour fournir un délai. Le taux d'échantillonnage est défini par le bloc « Assert ».

6.5 Implémentation expérimentale

L’implémentation expérimentale fera partir des travaux futurs avec la mise en place d’un banc d’essai physique composé entre autres d’un bus CC, d’un onduleur, une charge résistive-inductive, des capteurs pour mesurer les courants de sorties et du simulateur temps réel OP4500 de Opal-RT.

6.6 Conclusion

Dans ce chapitre, les étapes de l’adaptation du modèle à l'environnement RT-Lab pour fonctionner avec du matériel connecté, a été présentée. Elle permet ainsi de remplacer la charge simulée et l’onduleur par du matériel. La connexion physique à l'onduleur et au FPGA a été ajoutée en introduisant des blocs personnalisés dans le modèle, qui gère l'interface pour les périphériques d'E/S. Cette procédure sera utilisée lors de l’implémentation expérimentale au cours des travaux futurs.

Chapitre 7 - Conclusion et travaux futurs

7.1 Conclusion

Dans ce travail, un schéma de commande de prévision simple a été présenté pour un onduleur triphasé. Le contrôle du courant prédictif a été testé dans un modèle Simulink avec une période d'échantillonnage de $25\,\mu s$. Différentes périodes d'échantillonnage pour la commande prédictive ont été testées pour analyser l'impact de la fréquence de la prévision sur la performance de la méthode de contrôle. Des simulations avec une période d'échantillonnage de $25\,\mu s$ pour le bloc « Matlab Function », contenant l'algorithme de contrôle prédictif, ont donné un bon suivi de la consigne du courant, avec une certaine ondulation dans le courant de sortie. La fonction de coût (erreur) s'est stabilisée autour d'une valeur d'environ 0.4. Ce qui confirme une erreur de suivi de référence faible. Des simulations ont été effectuées avec une période d'échantillonnage pour l'algorithme de contrôle prédictif ajusté à $1\,\mu s$. Cela a amélioré significativement le suivi du courant de consigne et une valeur moyenne d'environ 0.015 pour la fonction de coût, qui ici représente l'erreur entre les courants de consigne et mesuré, a été réalisée. À partir de ces résultats, il est clair que la fréquence d'échantillonnage de l'algorithme de contrôle prédictif affectera la performance de la commande. Cela a contribué à la motivation pour l'implémentation de l'algorithme de contrôle sur un FPGA, qui peut effectuer une grande quantité de calculs à très grande vitesse.

L'algorithme de contrôle prédictif développé avec Matlab/Simulink a été ajusté avec la librairie XSG, avec pour objectif de l'implanter dans une configuration expérimentale réelle

sur FPGA. Les simulations avec XSG ont été réalisées et ont abouti à un très bon suivi du courant de consigne, avec une erreur relative sensiblement proche 0.4. Comparativement aux résultats de simulation sur Simulink, le contrôle de courant avec XSG présentait une performance inférieure. Cela peut être dû aux principes de fonctionnement du FPGA, qui nécessitent en réalité un certain temps pour effectuer une opération mathématique. Pour cela, les délais doivent être introduits de façon adéquate dans le modèle pour la synchronisation des données. Une autre explication pourrait être que XSG fonctionne avec un format de données à virgule fixe, ce qui limite la précision des données, qui peuvent se propager à travers le modèle.

Le contrôleur a été adapté pour être implanté dans le logiciel de simulation RT-Lab, afin de communiquer avec le matériel externe. De plus, des modules d'entrées/sorties de RT-XSG ont été utilisés dans le modèle pour établir la liaison d'échange avec le matériel externe. Des essaies ont été effectués pour établir des échanges de données entre la console dans RT-Lab et la sortie numérique et la communication d'entrée analogique du modèle vers l'onduleur.

7.2 Travaux futurs

Les travaux futurs incluront les tests du contrôleur prédictif sur un système physique réel, adaptation de l'algorithme de contrôle prédictif pour une limitation de la fréquence des signaux de commutations et étendre l'horizon de prévision à $k + 2$. Des travaux antérieurs ont indiqué qu'un horizon de prévision plus long peut contribuer à une meilleure performance du système en régime permanent [1].

Références

[1] J. Rodriguez, M.P.K., J. R. Espinoza, P. Zanchetta, H. Abu-Rub, H. A. Young, & C. A. Rojas, *State of the Art of Finite Control Set Model Predictive Control in Power Electronics.* IEEE Transactions on Industrial Informatics, 2013. **9**(2): p. 1003 - 1016.

[2] M. Nikolaou, *Model predictive controllers: A critical synthesis of theory and industrial needs, Advances in Chemical Engineering*, vol. 26. Academic Press, 2001.

[3] E. F. Camacho and C. Bordons, *Model Predictive Control*. NewYork: SpringerVerlag, 1999.

[4] J. M. Maciejowski, *Predictive Control With Constraints*. Englewood Cliffs, NJ: Prentice-Hall, 2002.

[5] Cortes, P., Kazmierkowski, M. P., Kennel, R. M., Quevedo, D. E. & Rodriguez, J., *Predictive Control in Power Electronics and Drives.* IEEE Transactions on Industrial Electronics, 2008. **55**(12): p. 4312-4324.

[6] O. Gulbudak and E. Santi, *FPGA-based model predictive controller for direct matrix converter*, IEEE Trans. Ind. Electron., vol. 63, no. 7, pp. 4560-4570, Jul. 2016.

[7] Z. Zhang, F. Wang, T. Sun, J. Rodríguez and R. Kennel, *FPGA-based experimental investigation of a quasi-centralized model predictive control for Back-to-Back converters,* IEEE Trans. Power Electron., vol. 31, no. 1, pp. 662-674, Jan. 2016.

[8] Z. Ma, S. Saeidi and R. Kennel, *FPGA implementation of model predictive control with constant switching frequency for PMSM drives*, IEEE Trans. Ind. Informat., vol. 10, no. 4, pp. 2055-2063, Nov. 2014.

[9] S. Wendel, A. Dietz and R. Kennel, *FPGA based finite-set model predictive current control for small PMSM drives with efficient resource streaming*, Proc. IEEE Int. Symp. Predictive Control Electr. Drives Power Electron. (PRECEDE), pp. 66-71, Sep. 2017.

[10] B. Stellato, T. Geyer and P. J. Goulart, *High-speed finite control set model predictive control for power electronics,* IEEE Trans. Power Electron., vol. 32, no. 5, pp. 4007-4020, May 2017.

[11] V. K. Singh, R. N. Tripathi and T. Hanamoto, *FPGA-based implementation of finite set-MPC for a VSI system using XSG-based modeling*, Energies, vol. 13, no. 1, pp. 260, Jan. 2020.

[12] T. J. Vyncke, S. Thielemans and J. A. Melkebeek, *Finite-set model-based predictive control for flying-capacitor converters: Cost function design and efficient FPGA implementation, IEEE Trans. Ind. Informat.,* vol. 9, no. 2, pp. 1113-1121, May 2013.

[13] P. M. Sanchez, O. Machado, E. J. B. Peña, F. J. Rodriguez and F. J. Meca, *FPGA-based implementation of a predictive current controller for power converters*, IEEE Trans. Ind. Informat., vol. 9, no. 3, pp. 1312-1321, Aug. 2013.

[14] V. K. Singh, R. N. Tripathi and T. Hanamoto, *Xilinx system generator based modelling of finite state MPC,* Proc. 9th Int. Power Electron. Conf., pp. 1698-1703, May 2018.

[15] V. K. Singh, R. N. Tripathi and T. Hanamoto, *Implementation Strategy for Resource Optimization of FPGA-Based Adaptive Finite Control Set-MPC Using XSG for a VSI System*, IEEE Journal of Emerging and Selected Topics in Power Electronics, vol. 9, no. 2, pp. 2066-2078, April 2021.

[16] Q. Chen, X. Luo, L. Zhang and S. Quan, *Model Predictive Control for Three-Phase Four-Leg Grid-Tied Inverters*, in IEEE Access, vol. 5, pp. 2834-2841, 2017.

[17] Rodriguez, J., Pontt, J., Silva, C.A., Correa, P., Lezana, P., Cortes, P. & Ammann, U., *Predictive Current Control of a Voltage Source Inverter.* Industrial Electronics, IEEE Transactions on, 2007. **54**(1): p. 495 - 503.

[18] Rodriguez, J.C., P., *Predictive control of power converters and electrical drives*. 2 ed. 2012, United Kingdom: John Wiley & sons.

[19] Wilson, A., Rojas, C. A., Rivera, M., Rodriguez, J., Espinoza, J. R., Wheeler, P. W. & Empringham, L. , *A Comparative Assessment of Model Predictive Current Control and Space Vector Modulation in a Direct Matrix Converter.* IEEE Transactions on Industrial Electronics, 2013. **60**(2): p. 578-588.

[20] Rodriguez, J., Silva, Cortez, P., C. & Flores, A., *Delay Compensation in Model Predictive Current Control of a Three-Phase Inverter*, in IEEE Transactions on Industrial Electronics. 2011, IEEE. p. 1323 - 1325.

[21] B. Stellato, T. Geyer, and P. J. Goulart, *High-Speed Finite Control SetModel Predictive Control for Power Electronics,* IEEE Transactions on Power Electronics, vol. 32, pp. 4007–4020, May 2017.

[22] T. Dorfling, H. Mouton, T. Geyer, P. Karamanakos, *Long-Horizon Finite-Control-Set Model Predictive Control With Nonrecursive Sphere Decoding on an FPGA,* IEEE Transactions on Power Electronics, PP(99) 1–1, Nov 2019.

[23] H. Abu-Rub, J. Guzinski, Z. Krzeminski, and H. A. Toliyat, "*Predictive current control of voltage-source inverters*," IEEE Trans. Ind. Electron., vol. 51, no. 6, pp. 585–593, Jun. 2004.

[24] J. Rodríguez and P. Cortés, *"Predictive control of a three-phase neutralpoint clamped inverter,"* in Predictive Control of Power Converters and Electrical Drives. Hoboken, NJ, USA: Wiley-IEEE Press, 2012, pp.65–79.

[25] H. T. Moon, H. S. Kim, and M. J. Youn, *"A discrete-time predictive current control for PMSM,"* IEEE Trans. Power Electron., vol. 18, no. 1, pp. 464–472, Jan. 2003.

[26] T. Geyer, G. Papafotiou, and M. Morari, *"Model predictive direct torque control-part I: Concept, algorithm, and analysis,"* IEEE Trans. Ind. Electron., vol. 56, no. 6, pp. 1894–1905, Jun. 2009.

[27] G. Papafotiou, J. Kley, K. G. Papadopoulos, P. Bohren, andM.Morari, *"Model predicitive direct torque control-part II: Implementation and experimental evaluation,"* IEEE Trans. Ind. Electron., vol. 56, no. 6, pp. 1906–1915, Jun. 2009.

Annexe A – Modélisation Simulink du système – Modèle moyen

Code A.1 Fichier des paramètres du contrôleur prédictif

```
clear all;
% Paramètres requis pour l'algorithme de contrôle du système
global Ts R L v etats

% Periode de simulation de l'algorithme de prévision [S]
%Ts = 50e-6;
Ts = 25e-6;
%Ts = 1e-6;

% Paramètres de la charge
R=10;                % Résistance [Ohm]
L=10e-3;             % Inductance [H]
e = 100;             % f.c.é.m [V]
f_e = 60*(2*pi);     % Fréquence de la f.c.é.m [rad/s]
Vdc = 540;           % Tension du bus CC [V]

% Références du Courant
I_ref_peak = 10;
f_ref = 60*(2*pi);

%
ik_ref = 0 + 1j*0;
ik = 0 + 1j*0;
ik1 = 0 + 1j*0;
g = 0;
x_opt = 0;

% Vecteurs de Tension
v0 = 0;
v1 = (2/3)*Vdc;
v2 = (1/3)*Vdc + (1j*sqrt(3)/3)*Vdc;
v3 = (-1/3)*Vdc + (1j*sqrt(3)/3)*Vdc;
v4 = (-2/3)*Vdc;
v5 = (-1/3)*Vdc - (1j*sqrt(3)/3)*Vdc;
v6 = (1/3)*Vdc - (1j*sqrt(3)/3)*Vdc;
v7 = 0;
v = [v0 v1 v2 v3 v4 v5 v6 v7];

% États de commutation
etats = [0 0 0;1 0 0;1 1 0;0 1 0;0 1 1;0 0 1;1 0 1;1 1 1];
```

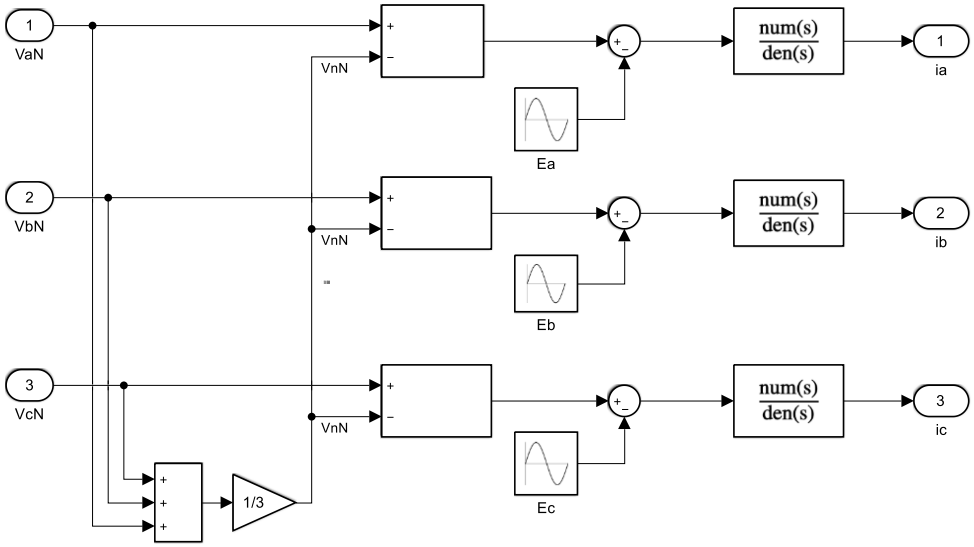


Figure A-1 Modèle de la charge du système

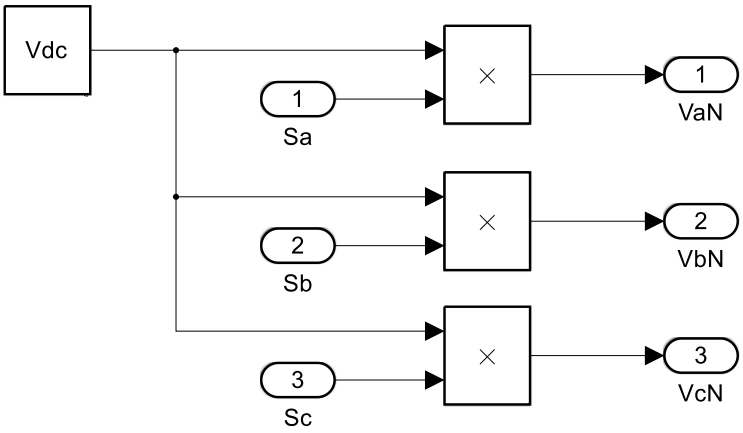


Figure A-2 Modèle de l'onduleur

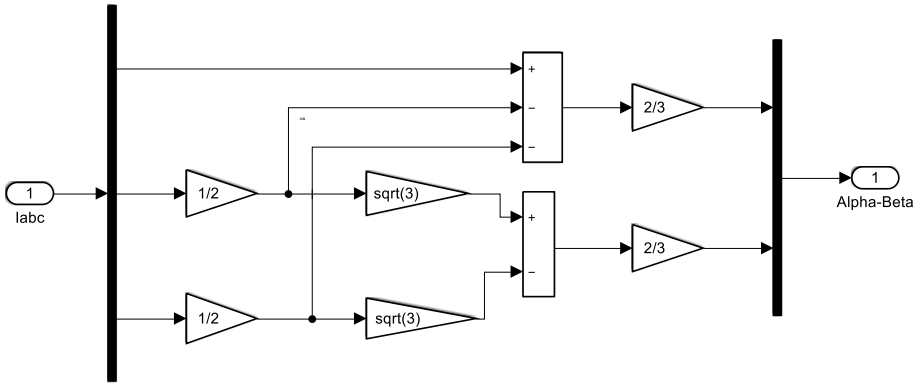


Figure A-3 Transformation des coordonnées abc en coordonnées alpha-bêta

Annexe B – Modélisation Simulink du système – SimPower system

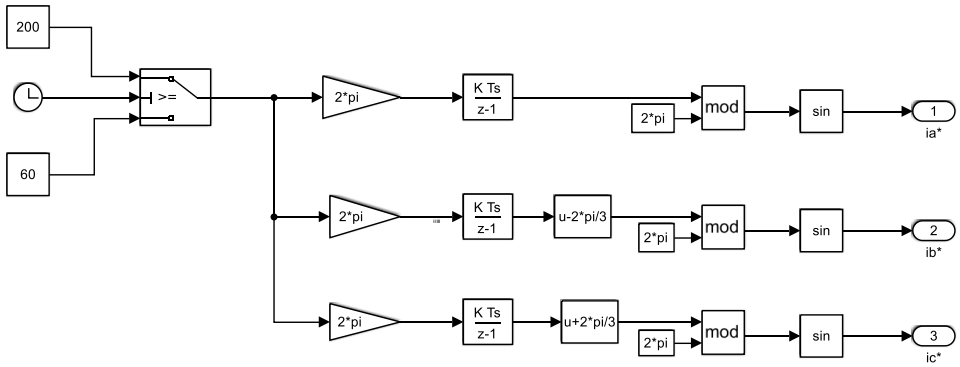


Figure B-1 Génération des courants de consigne triphasés

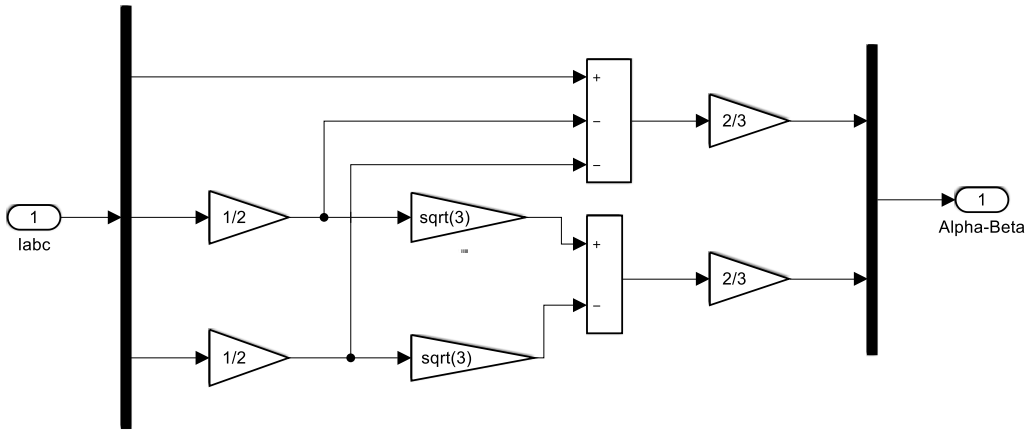


Figure B-2 Transformation des coordonnées abc en coordonnées alpha-bêta

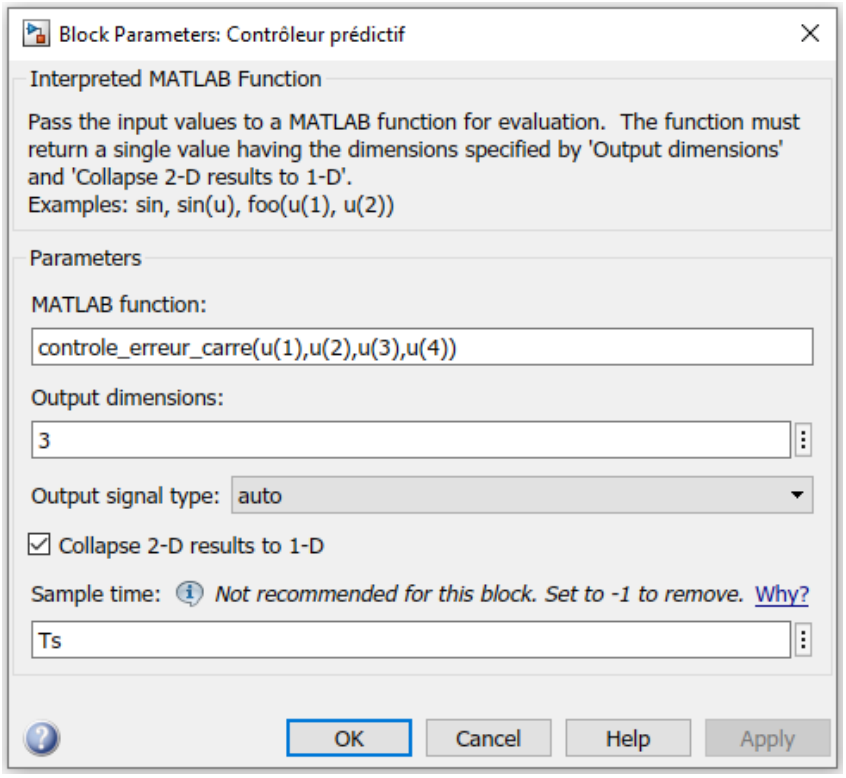


Figure B-3 Bloc de configuration du contrôleur prédictif

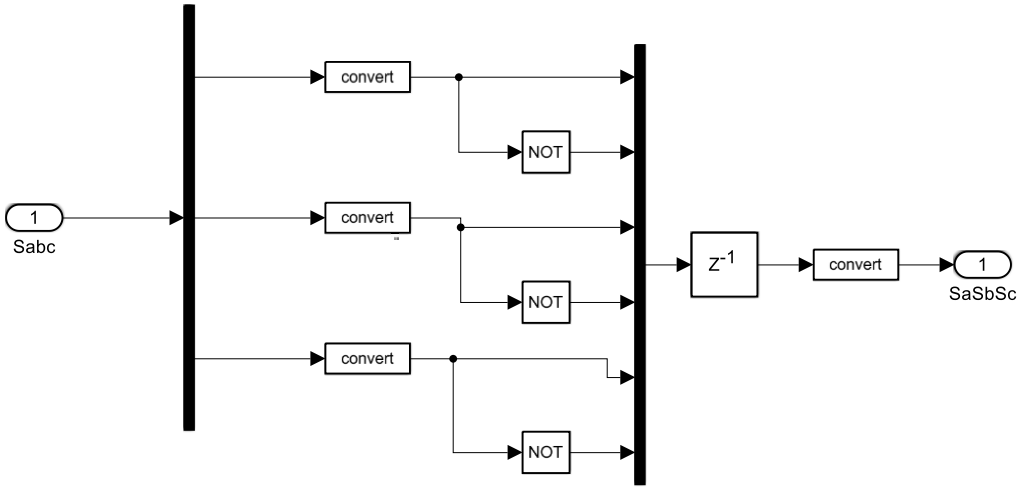


Figure B-4 Signaux de commutation

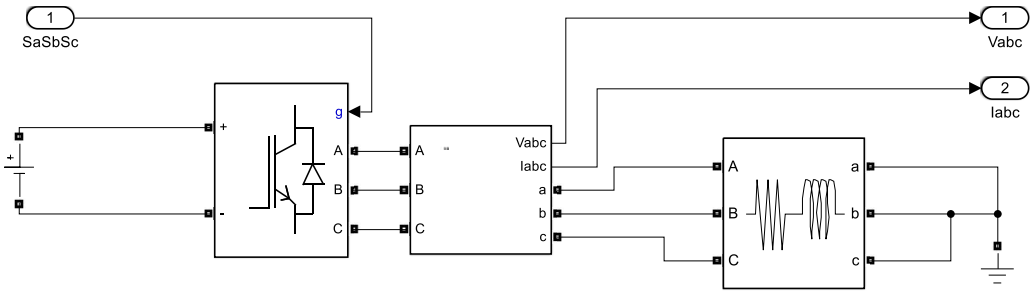


Figure B-5 Onduleur et charge du système