ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE ÉLECTRIQUE
M.Ing.

PAR
FRANÇOIS PERREAULT

DÉVELOPPEMENT D'UNE INTERFACE ENTRE UNE COMMANDE EXTERNE ET UN SIMULATEUR EN TEMPS RÉEL

MONTRÉAL, LE 28 AVRIL 2006

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE:

M. Louis-A. Dessaint, directeur de mémoire
Département de génie électrique à l'École de technologie supérieure

M. Bruno De Kelper, codirecteur
Département de génie électrique à l'École de technologie supérieure

M. Roger Champagne, président du jury
Département de génie logiciel et des TI à l'École de technologie supérieure

M. Pierre Mercier, membre du jury
Consultant, iOMEGAt

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 21 AVRIL 2006

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

DÉVELOPPEMENT D'UNE INTERFACE ENTRE UNE COMMANDE EXTERNE ET UN SIMULATEUR EN TEMPS RÉEL

François Perreault

SOMMAIRE

Le projet de recherche consiste à réaliser une interface entre un simulateur en temps réel d'entraînements électriques et une commande externe. L'interface est réalisée pour des modèles d'entraînements électriques à valeur moyenne et des modèles d'entraînements électriques détaillés. Les parties de commande des modèles sont implémentées en temps réel avec la carte de prototypage DS1104 de la compagnie dSPACE. L'implémentation en temps réel des parties de l'électronique de puissance et de la machine des modèles est réalisée avec deux ordinateurs personnels, le logiciel xPC Target et deux cartes d'E/S PCI-MIO-16E-4 de la compagnie National Instruments.

Les modèles à valeur moyenne comportent des onduleurs triphasés basés sur des sources de courants contrôlées par des références de courant provenant de la commande. Les signaux de ces modèles ne présentent donc pas de discontinuités, ce qui permet de réaliser une simulation en temps réel simple. La simulation en temps réel avec commande externe des modèles à valeur moyenne est validée grâce à la comparaison des résultats obtenus avec des résultats connus de simulation en temps différé.

Les modèles détaillés comportent des onduleurs triphasés basés sur des interrupteurs contrôlés par des signaux d'impulsions provenant de la commande. Les signaux de ces modèles présentent donc des discontinuités qui peuvent survenir à l'intérieur des pas de calcul de la simulation et qui doivent être corrigées. La simulation en temps réel avec commande externe des modèles détaillés qui fonctionne avec la méthode des commutations précises du professeur De Kelper [11] nécessite le développement d'un pilote spécifique pour que les cartes PCI-MIO-16E-4 puissent traiter les signaux d'impulsions provenant de la commande. Le pilote développé est intégré dans le simulateur avec une S-function, qui génère des signaux de passage par zéro pour les impulsions de la commande. L'engin de calcul de la simulation est modifié pour faire le traitement de ces signaux de passage par zéro. La comparaison des résultats obtenus avec des résultats connus de simulation en temps différé démontre que l'engin de calcul n'est pas adapté correctement pour le traitement des signaux de passage par zéro associés aux impulsions de la commande. Les recommandations proposent des modifications dans l'engin de calcul pour le traitement des commutations multiples et des commutations simultanées.

INTERFACE DEVELOPMENT BETWEEN AN EXTERNAL CONTROLLER AND A REAL-TIME SIMULATOR

François Perreault

ABSTRACT

The main objective of the research project is to create an interface between an electric drive real-time simulator and an external controller. The interface is developed for the simulation of averaged-value models and detailed models. The implementation of the model's controller part is done with the dSPACE DS1104 controller board. The real-time implementation of the power electronics and electric machine is done with two personal computers, the xPC Target software and two National Instruments PCI-MIO-16E-4 I/O boards.

Averaged-value models use a three-phase inverter based on current sources controlled by current references fed by the external controller. Signals from these models don't present discontinuities so they can be easily simulated in real-time. The validation is done by the comparison between the results obtained for the real-time simulation with an external controller and the results obtained for the off-line simulation.

Detailed models use a three-phase inverter based on switches controlled by pulses fed by the external controller. Signals from these models presents discontinuities that can occur between two fixed time steps and that have to be corrected. The real-time simulator uses the accurate switching method developed by Professor De Kelper [11] to achieve the detailed models real-time simulation with an external controller. A specific PCI-MIO-16E-4 I/O board pulse handling driver has been developed for the pulses signals fed by the external controller. The driver's integration in the simulator is done through a S-function. The driver's S-function generates zero-crossing signals related to the controller's pulses signals. The simulation engine has been modified to handle these zero-crossing signals. The comparison between the results obtained for the real-time simulation with an external controller and the results obtained for the off-line simulation shows that the simulation engine is not well adapted to handle the zero-crossing signals related to the controller's pulses signals. Recommendations are to adapt the simulation engine to handle multiple switching and simultaneous switching.

REMERCIEMENTS

Tout d'abord, je tiens à remercier le professeur Louis-A. Dessaint, directeur de mémoire, pour m'avoir offert l'opportunité de réaliser une maîtrise et pour son soutien tout au long de mon programme. Je tiens aussi à remercier le professeur Bruno De Kelper, codirecteur, pour tout le support et l'aide qu'il m'a apporté dans la réalisation du projet de recherche. Finalement, je tiens à remercier ma famille et mes amis pour le support moral qu'ils m'ont apporté et pour m'avoir aidé à persévérer dans la réalisation de ma maîtrise.

Je tiens particulièrement à remercier le Fonds Nature et Technologies pour le financement qu'il m'a accordé afin de poursuivre mes études aux cycles supérieurs.

TABLE DES MATIÈRES

Page

LISTE DES FIGURES

Page

LISTE DES TABLEAUX

Page

LISTES DES ABRÉVIATIONS ET SIGLES

ASIC	Application-Specific Integrated Circuit
BIOS	Basic input output system
CAN	Convertisseur analogique à numérique
CC	Courant continu
CNA	Convertisseur numérique à analogique
DAQ-STC	Data Acquisition Timing Controler
dq0	Axes du référentiel dq arbitraire (d=direct, q=quadrature, 0=homopolaire)
DSP	Digital signal processor
E/S	Entrée/sortie
F	Coefficient de la friction visqueuse combinée du rotor et de la charge, $N \cdot m \cdot s$
FOC	Field oriented control (Commande à flux orienté)
GPCT	General Purpose Counter/Timer Module
H	Constante de l'inertie combinée du rotor et de la charge
i_{abc}	Vecteur des courants statoriques de la machine synchrone à aimants permanents, A
i_{abc}^*	Vecteur des références des courants statoriques de la machine synchrone à aimants permanents, A
i_{abcs}	Vecteur des courants statoriques de la machine asynchrone, A
i_{abcs}^*	Vecteur des références des courants statoriques de la machine asynchrone, A
I_{cc}	Courant CC moyen de la source idéale, A
J	Coefficient de l'inertie combinée du rotor et de la charge, $kg \cdot m^2$

L_d, L_q	Composantes orthogonales dq de l'inductance des enroulements statoriques de la machine synchrone à aimants permanents, H
L_m	Inductance de magnétisation de la machine asynchrone, H
L_s, L'_r	Inductance totale statorique et inductance totale rotorique de la machine asynchrone, H
MLI	Modulation de largeur d'impulsion
N	Vitesse angulaire mécanique du rotor, RPM
p	Nombre de paires de pôles
PCI	Peripheral component interconnect
PFI	Programmable Function Input
PI	Proportionnel et intégrale
P_{pertes}	Pertes de l'onduleur triphasé, W
P_{sortie}	Puissance de sortie de l'onduleur triphasé, W
R	Résistance des enroulements statoriques de la machine synchrone à aimants permanents, Ω
RAM	Random-access memory
R'_r, L'_{lr}	Résistance et inductance de fuite rotorique de la machine asynchrone, Ω, H
R_s, L_{ls}	Résistance et inductance de fuite statorique de la machine asynchrone, Ω, H
RTI	Real-Time Interface
RTW	Real-Time Workshop
SPS	SimPowerSystems
t	temps, s
T	période, s

TCP/IP	Transmission controp protocol / Internet protocol
T_e	Couple électromagnétique, N · m
TET	Temps d'exécution de tâche
T_m	Couple mécanique, N · m
v_d, i_d	Composantes directes de la tension et du courant statorique de la machine synchrone à aimants permanents, V, A
V'_{dr}, i'_{dr}	Composantes directes de la tension et du courant rotorique de la machine asynchrone, V, A
V_{ds}, i_{ds}	Composantes directes de la tension et du courant statorique de la machine asynchrone, V, A
$V_{entrée}$	Tension d'entrée de l'onduleur triphasé, V
v_q, i_q	Composantes en quadrature de la tension et du courant statorique de la machine synchrone à aimants permanents, V, A
V'_{qr}, i'_{qr}	Composantes en quadrature de la tension et du courant rotorique de la machine asynchrone, V, A
V_{qs}, i_{qs}	Composantes en quadrature de la tension et du courant statorique de la machine asynchrone, V, A
ω_m	Vitesse angulaire mécanique du rotor, rad/s
ω_r	Vitesse angulaire électrique du rotor $(\omega_m \times p)$, rad/s
λ	Flux induit par les aimants permanents du rotor dans les phases du stator de la machine synchrone à aimants permanents, Wb
φ'_{qr}, φ'_{dr}	Composante en quadrature et directe du flux rotorique de la machine asynchrone, Wb
φ_{qs}, φ_{ds}	Composante en quadrature et directe du flux statorique de la machine asynchrone, Wb
φ_s	Flux statorique de la machine asynchrone, Wb

τ_r	Constance de temps du rotor, s
θ_e	Position angulaire du flux rotorique, rad
θ_m	Position angulaire mécanique du rotor, rad
θ_r	Position angulaire électrique du rotor $(\theta_m \times p)$, rad

INTRODUCTION

Le projet de recherche consiste à réaliser une interface entre un simulateur en temps réel d'entraînements électriques et une commande externe. La simulation en temps réel avec commande externe est réalisée pour des modèles d'entraînements électriques à valeur moyenne et détaillés. Les principaux objectifs sont de réaliser l'interface et de valider le fonctionnement de la simulation en temps réel.

La première partie du projet consiste à faire la simulation en temps réel avec commande externe de modèles d'entraînements électriques à valeur moyenne. Cette première partie a comme principal objectif la familiarisation des outils de travail. La carte DS1104 de la compagnie dSPACE sert d'outil de prototypage pour les parties de commande des modèles. Deux ordinateurs personnels, le logiciel xPC Target et deux cartes d'E/S PCI-MIO-16-E-4 de la compagnie National Instruments servent d'outils pour la simulation en temps réel des parties de l'électronique de puissance et de la machine des modèles. La validation des résultats de la simulation en temps réel est réalisée par une comparaison avec des résultats de simulation en temps différé.

La deuxième partie du projet, qui consiste à faire la simulation en temps réel avec commande externe de modèles d'entraînements électriques détaillés, constitue la problématique principale du projet. La simulation de cette partie fonctionne avec la méthode des commutations précises développée par le professeur Bruno De Kelper de l'École de technologie supérieure. La réalisation de cette partie nécessite le développement d'une interface spécifique pour la réception des signaux d'impulsions provenant de la commande externe, car ces signaux sont asynchrones aux pas de calcul de la simulation en temps réel. La simulation en temps différé des modèles détaillés avec la méthode des commutations précises sert à la validation de la simulation en temps réel avec commande externe des modèles détaillés.

Le chapitre 1 du mémoire présente une revue de la littérature sur les travaux effectués dans le domaine de la simulation en temps réel avec commande externe. La première partie du projet est présentée par les chapitres 2, 3 et 4. Le chapitre 2 présente de façon détaillée les modèles d'entraînement électrique à valeur moyenne utilisés pour la réalisation de cette partie. Dans le chapitre 3, la stratégie employée pour l'implémentation en temps réel avec commande externe de ces modèles est exposée et les outils utilisés sont présentés. Finalement, le chapitre 4 fait la validation de la simulation en temps réel avec commande des modèles à valeur moyenne par la comparaison des résultats obtenus avec des résultats de simulation en temps différé.

La deuxième partie du projet est présentée par le chapitre 5, qui présente les problématiques inhérentes de la simulation en temps réel avec commande externe des modèles détaillés. Le chapitre 5 présente ensuite la stratégie employée pour la réalisation de cette partie et expose les travaux effectués. Ce chapitre se termine par la présentation et l'analyse des résultats obtenus.

CHAPITRE 1

REVUE DE LA LITTÉRATURE

La recherche bibliographique a démontrée que peu de travaux traitent de la simulation en temps réel d'entraînement électrique avec commande externe. Cependant, quelques articles traitent du sujet. Champagne, Dessaint et Fortin-Blanchette [12] présentent une approche de simulation en temps réel d'entraînement électrique de machine asynchrone qui permet l'interconnexion d'une commande externe. L'entraînement électrique proposé est composé d'une source de puissance, d'une machine asynchrone, d'une charge et d'un convertisseur dont la fréquence de commutation peut atteindre 4 kHz. L'approche utilisée pour la simulation présente un modèle de simulation sous forme d'un modèle d'état global réalisé à l'aide des équations d'états de la source de puissance, du convertisseur et de la machine asynchrone. Ils démontrent le fonctionnement de leur approche comme outil de prototypage en réalisant l'interface entre une commande externe et la simulation en temps réel exécutée dans le simulateur Hypersim. La commande utilisée est composée d'un régulateur de vitesse et d'une commande vectorielle. Elle reçoit comme seule entrée la vitesse électrique du rotor (ω_r).

Le-Huy, Sybille et Giroux [13] présentent une simulation en temps réel d'un entraînement électrique pour une machine à courant continu avec commande externe. Le montage de simulation qu'ils utilisent comprend douze entraînements électriques contrôlés par des commandes simulées, puis un treizième contrôlé par une commande externe. L'entraînement électrique est un entraînment à quatre quadrants. La commande externe est composée d'un régulateur de vitesse avec une boucle interne de courant et d'une logique de commutation pour les thyristors. La commande reçoit comme entrées la vitesse mécanique du rotor et le courant d'armature. La simulation est réalisée avec le simulateur en temps réel Hypersim.

Abourida, Dufour et Bélanger [14] présentent un outil de simulation en temps réel de systèmes d'électronique de puissance et d'entraînements électriques qui permet l'interconnexion de commande externe. Comparativement aux travaux exposés précédemment, leur outil se base sur l'utilisation d'ordinateurs personnels pour la simulation en temps réel. Leur outil utilise aussi des techniques de simulation qui permettent en autre la compensation en temps réel d'événements se produisant à l'intérieur du pas de calcul. L'outil est utilisé dans Simulink et permet de faire l'interconnexion d'équipements externes via des cartes d'E/S. Les auteurs ne divulguent pas de détails techniques quant aux méthodes utilisées et ne démontrent pas le fonctionnement de leur outil pour la simulation en temps réel avec commande externe.

Notre projet de recherche consiste à réaliser l'interface entre une commande externe et un simulateur en temps réel d'entraînements électriques. Le but final est d'obtenir un produit dont les caractéristiques sont très similaires à celles de l'outil proposé par Abourida, Dufour et Bélanger [14]. Le simulateur en temps réel du projet de recherche est aussi basé sur l'utilisation d'ordinateurs personnels, ce qui rend le produit plus accessible que les autres méthodes présentées qui utilisent le simulateur en temps réel Hypersim. De plus, la simulation en temps réel dans le projet de recherche est basée sur une méthode développée par le professeur Bruno De Kelper [11] qui permet la correction des événements se produisant à l'intérieur du pas de calcul. L'outil de travail utilisé est aussi Simulink et l'interconnexion de la commande externe est réalisée avec des cartes d'E/S. Compte tenu des faibles divulgations de la part d'Abourida, de Dufour et de Bélanger [14], il est difficile de juger de l'originalité du projet de recherche. Cependant, le projet permet de valider la capacité de la méthode du professeur De Kelper [11] à faire de la simulation en temps réel avec commande externe.

CHAPITRE 2

MODÈLES À VALEUR MOYENNE

2.1 Introduction

La première partie du projet de recherche consiste à réaliser la simulation en temps réel de modèles à valeur moyenne avec commande externe. Pour valider la simulation en temps réel, les résultats obtenus sont comparés à des résultats de simulations en temps différé. Cette validation est faite au chapitre 4.

Le présent chapitre présente une description détaillée des deux modèles de simulation à valeur moyenne utilisés dans la première partie du projet de recherche. Le premier modèle est un entraînement électrique de machine asynchrone à commande à flux orienté et le deuxième est un entraînement électrique de machine synchrone à aimants permanents.

2.2 Modèles des entraînements électriques

Les modèles de simulation utilisés sont les modèles à valeur moyenne AC3 et AC6 de la librairie Electric Drives du SPS de MATLAB. Le modèle AC3 est l'entraînement électrique de machine asynchrone à commande à flux orienté et le modèle AC6 est l'entraînement électrique de machine synchrone à aimants permanents. Les sections 2.2.1 et 2.2.2 présentent la description détaillée des éléments de chacun des modèles.

La caractéristique principale des modèles à valeur moyenne est qu'ils comportent un onduleur triphasé non standard. L'onduleur des modèles à valeur moyenne est composé de sources de courant contrôlées par les références des courants statoriques provenant du régulateur de courant de la commande. La génération d'impulsions par la commande

avec un hystérésis de courant pour la commutation des interrupteurs d'un onduleur standard n'est pas nécessaire puisque l'onduleur des modèles à valeur moyenne fonctionne avec les références des courants statoriques.

Les modèles à valeur moyenne permettent d'utiliser un pas de calcul plus grand, puisque la complexité est réduite grâce à l'élimination des discontinuités à traiter dans les signaux. Ces modèles permettent aussi de faire une simulation en temps réel beaucoup plus simple, puisque la correction des événements se produisant à l'intérieur du pas de calcul n'est pas nécessaire. En effet, ces corrections sont nécessaires seulement lorsque les impulsions pour la commutation des interrupteurs sont présentes, puisqu'elles peuvent survenir à l'intérieur du pas de calcul.

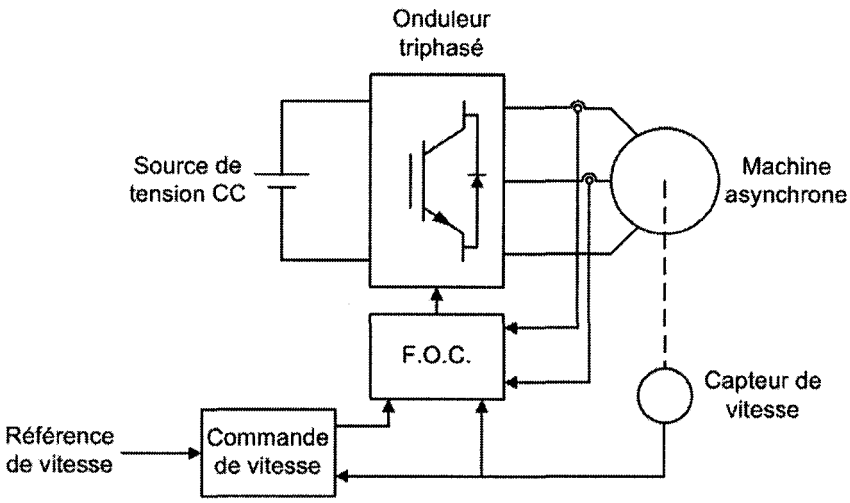


Figure 1 Entraînement électrique de machine asynchrone à commande à flux orienté

2.2.1 Modèle AC3

Le modèle AC3 est un entraînement électrique de machine asynchrone à commande à flux orienté. Il comprend une source triphasée, un convertisseur de puissance, une machine asynchrone et une commande composée d'une commande de vitesse et d'une

commande à flux orienté (FOC). Le convertisseur de puissance est composé d'un redresseur triphasé, d'un hacheur de freinage dynamique et d'un onduleur triphasé. Tous les éléments du modèle de l'entraînement électrique AC3 se retrouvent sous forme de blocs dans les libraires SPS et Electric Drives de Simulink.

Afin de réduire la complexité du modèle, la partie du convertisseur de puissance constituée de la source triphasée, du redresseur et du hacheur est substituée par une source CC idéale. Le schéma bloc du modèle AC3 avec la source CC idéale est présenté à la figure 1 et le schéma Simulink est présenté à la figure 2.

Les sections 2.2.1.1 à 2.2.1.4 présentent une description détaillée des quatre principaux éléments du modèle, soit la machine asynchrone, l'onduleur triphasé, la commande de vitesse et la commande à flux orienté.

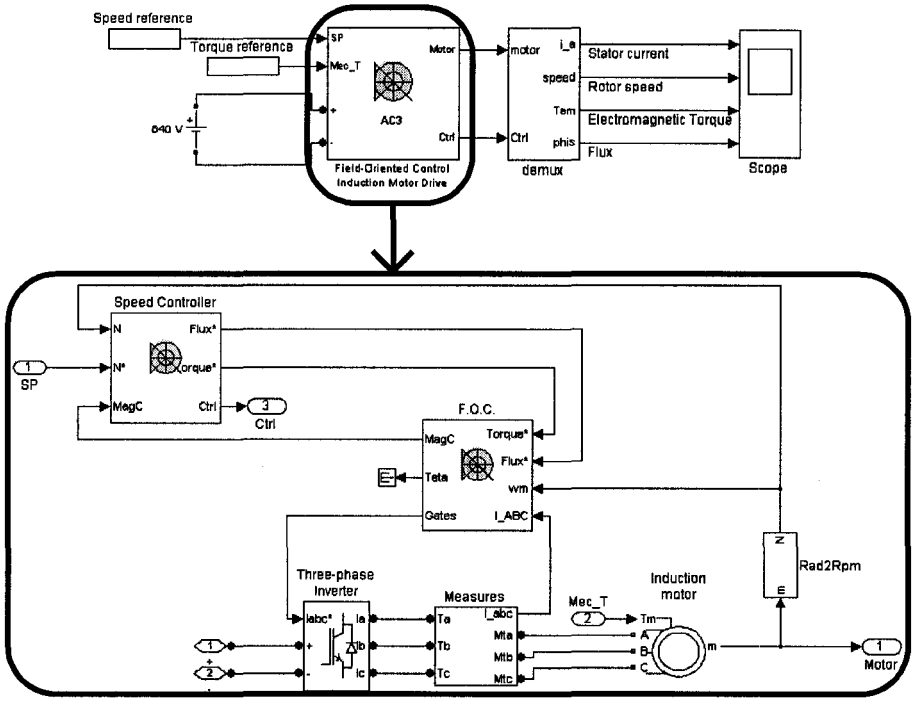


Figure 2 Schéma Simulink du modèle AC3 à valeur moyenne

2.2.1.1 Machine asynchrone

La machine asynchrone utilisée dans le modèle AC3 est un moteur d'induction triphasé à cage d'écureuil. La machine asynchrone du SPS est basée sur le modèle de Krause [25] présenté par les équations (2.1) à (2.13). Les équations (2.1) à (2.11) modélisent la partie électrique et les équations (2.12) à (2.13) modélisent la partie mécanique. Dans les équations électriques, les paramètres et variables électriques du rotor sont référés au stator. De plus, les quantités statoriques et rotoriques sont exprimées dans le référentiel dq. La figure 3 présente le schéma électrique de la machine asynchrone dans le référentiel dq.

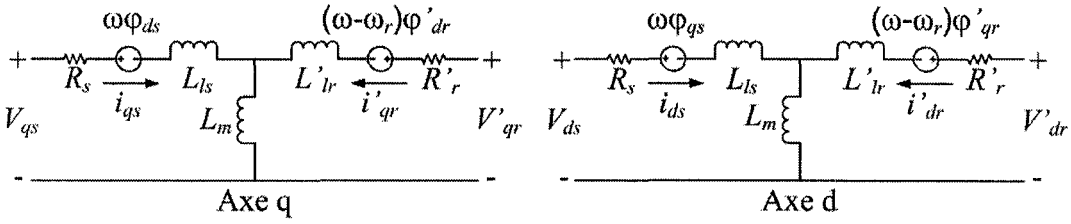


Figure 3 Schéma électrique de la machine asynchrone

Dans la partie électrique, les composantes de la tension statorique et du flux statorique sont exprimées par les équations (2.1), (2.2), (2.6) et (2.7). Les équations (2.3), (2.4), (2.8) et (2.9) expriment les composantes de la tension rotorique et du flux rotorique référées au stator. Finalement, l'équation (2.5) exprime le couple électromagnétique, l'équation (2.10) exprime l'inductance totale statorique et l'équation (2.11) exprime l'inductance totale rotorique. Pour la partie mécanique, les équations (2.12) et (2.13) expriment la dynamique de l'accélération et de la vitesse de la machine asynchrone.

$$V_{qs} = R_s i_{qs} + \frac{d}{dt}\varphi_{qs} + \omega_m \varphi_{ds} \tag{2.1}$$

$$V_{ds} = R_s i_{ds} + \frac{d}{dt}\varphi_{ds} - \omega_m \varphi_{ds} \tag{2.2}$$

$$V'_{qr} = R'_r i'_{qr} + \frac{d}{dt}\varphi'_{qr} + (\omega_m - \omega_r)\varphi'_{dr} \tag{2.3}$$

$$V'_{dr} = R'_r i'_{dr} + \frac{d}{dt}\varphi'_{dr} - (\omega_m - \omega_r)\varphi'_{qr} \tag{2.4}$$

$$T_e = 1,5p(\varphi_{ds} i_{qs} - \varphi_{qs} i_{ds}) \tag{2.5}$$

$$\varphi_{qs} = L_s i_{qs} + L_m i'_{qr} \tag{2.6}$$

$$\varphi_{ds} = L_s i_{ds} + L_m i'_{dr} \tag{2.7}$$

$$\varphi'_{qr} = L'_r i'_{qr} + L_m i_{qs} \tag{2.8}$$

$$\varphi'_{dr} = L'_r i'_{dr} + L_m i_{ds} \tag{2.9}$$

$$L_s = L_{ls} + L_m \tag{2.10}$$

$$L'_r = L'_{lr} + L_m \tag{2.11}$$

$$\frac{d}{dt}\omega_m = \frac{1}{2H}(T_e - F\omega_m - T_m) \tag{2.12}$$

$$\frac{d}{dt}\theta_m = \omega_m \tag{2.13}$$

Une transformation de référentiel est nécessaire puisque les entrées et les sorties du modèle de la machine asynchrone sont dans le référentiel abc. Pour convertir les tensions triphasées à l'entrée du modèle du référentiel abc en deux composantes orthogonales dans le référentiel dq, la transformation de Park est utilisée. Pour convertir les composantes orthogonales de courants à la sortie du modèle du référentiel dq en courants triphasés dans le référentiel abc, la transformation inverse de Park est utilisée. Le référentiel dq utilisé est un référentiel tournant en phase avec la position électrique du rotor. Les équations (2.14) et (2.15) présentent les transformations des tensions lignes-lignes statoriques et rotoriques du référentiel abc vers le référentiel dq. Les équations (2.16) à (2.19) présentent les transformations des courants de lignes statoriques et rotoriques du référentiel dq vers le référentiel abc.

$$\begin{bmatrix} V_{qs} \\ V_{ds} \end{bmatrix} = \frac{2}{3}\begin{bmatrix} \cos\theta_r & -\cos\left(\theta_r + \frac{2\pi}{3}\right) \\ \sin\theta_r & -\sin\left(\theta_r + \frac{2\pi}{3}\right) \end{bmatrix}\begin{bmatrix} V_{abs} \\ V_{bcs} \end{bmatrix} \tag{2.14}$$

$$\begin{bmatrix} V'_{qr} \\ V'_{dr} \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & \frac{1}{2} \\ 0 & -\frac{\sqrt{3}}{2} \end{bmatrix}\begin{bmatrix} V'_{abr} \\ V'_{bcr} \end{bmatrix} \tag{2.15}$$

$$\begin{bmatrix} i_{as} \\ i_{bs} \end{bmatrix} = \begin{bmatrix} \cos\theta_r & \sin\theta_r \\ \cos\left(\theta_r - \frac{2\pi}{3}\right) & \sin\left(\theta_r - \frac{2\pi}{3}\right) \end{bmatrix}\begin{bmatrix} i_{qs} \\ i_{ds} \end{bmatrix} \tag{2.16}$$

$$\begin{bmatrix} i'_{ar} \\ i'_{br} \end{bmatrix} = \begin{vmatrix} 1 & 0 \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{vmatrix}\begin{bmatrix} i'_{qr} \\ i'_{dr} \end{bmatrix} \tag{2.17}$$

$$i_{cs} = -i_{as} - i_{bs} \tag{2.18}$$

$$i'_{cr} = -i'_{ar} - i'_{br} \tag{2.19}$$

La figure 4 présente la fenêtre de configuration des divers paramètres électriques et mécaniques du modèle de simulation de la machine asynchrone. La machine utilisée opère avec une puissance apparente nominale de 149,2 kVA, une tension nominale de 460 V et une fréquence de 60 Hz. La résistance statorique est de 14,85 mΩ, l'inductance de fuite statorique est de 0,3027 mH, la résistance rotorique référée au stator est de 9,295 mΩ, l'inductance de fuite rotorique référée au stator est de 0,3027 mH et l'inductance mutuelle est de 10,46 mH. La machine est constituée de deux paires de pôles, l'inertie combinée du rotor et de la charge est de 3,1 $\mathrm{kg \cdot m^2}$ et le coefficient de friction visqueuse combiné du rotor et de la charge est de 0,08 $\mathrm{N \cdot m \cdot s}$.

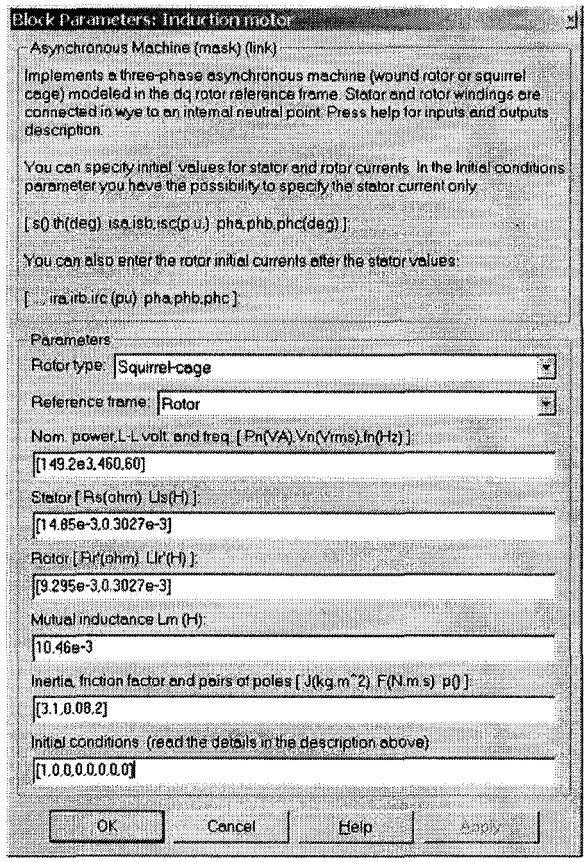


Figure 4 Paramètres de la machine asynchrone

2.2.1.2 Onduleur triphasé

L'onduleur triphasé est composé de sources contrôlées de courant et de tension. La figure 5 présente le schéma bloc de l'onduleur. Le côté CC comporte une source de courant, puis le côté CA comporte deux sources de courant et deux sources de tension. La source de courant du côté CC est contrôlée selon l'équation (2.20), ce qui permet de représenter le comportement du courant CC moyen de la source idéale. Du côté CA, les sources de courant représentent les courants triphasés moyens appliqués au stator de la machine asynchrone. Les courants appliqués au stator sont égaux aux références des courants statoriques (i_{abcs}*) provenant de la commande à flux orienté. Des petits

courants sont ajoutés pour compenser l'effet de la charge triphasée. Cette charge est nécessaire puisque les sources de courants de l'onduleur sont en série avec la machine asynchrone. L'onduleur utilise seulement les références i_{as}* et i_{bs}* pour les sources de courant contrôlées. La référence de courant i_{cs}* est obtenue selon l'équation (2.21). Les sources de tension sont utilisées pour la représentation de la saturation de l'onduleur. La saturation de l'onduleur n'est pas utilisée dans le projet.

$$I_{cc} = \frac{P_{sortie} + P_{pertes}}{V_{entrée}} \tag{2.20}$$

$$i_{cs}^* = -i_{as}^* - i_{bs}^* \tag{2.21}$$

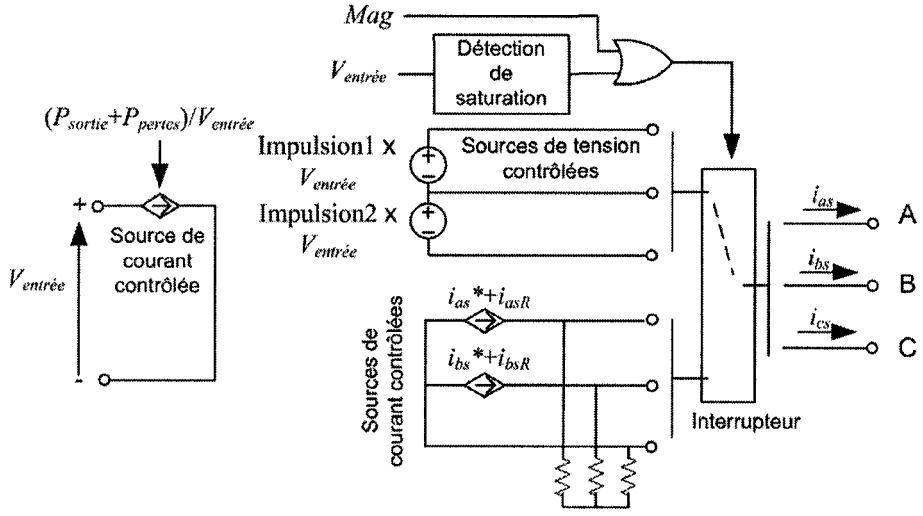


Figure 5 Onduleur triphasé

2.2.1.3 Commande de vitesse

La commande de vitesse utilisée est basée sur un régulateur de type PI. Les sorties du régulateur PI sont les références de couple et de flux appliquées à la commande à flux orienté. La référence de couple est déterminée à partir du traitement proportionnel et

intégral sur l'erreur entre la référence de vitesse et la vitesse réelle de la machine asynchrone. D'autre part, la référence de flux est déterminée dans une table de correspondance à partir de la référence de vitesse. La figure 6 présente le schéma bloc de la commande de vitesse.

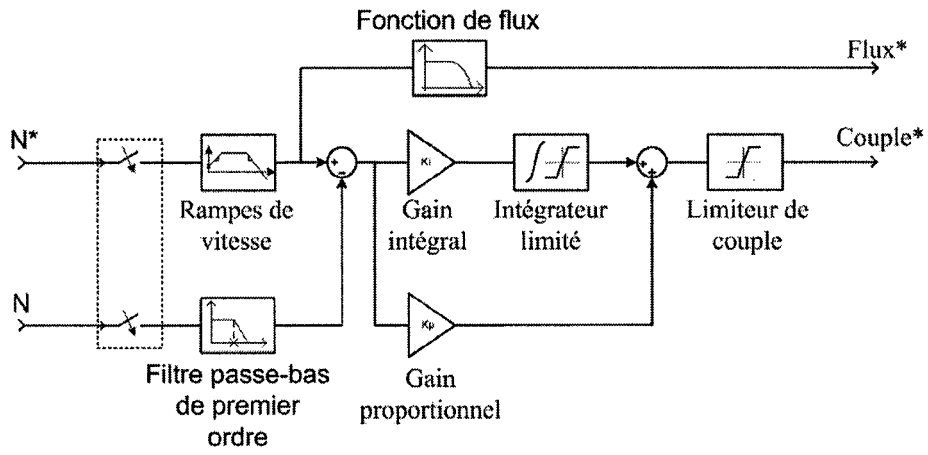


Figure 6 Commande de vitesse

Les rampes de vitesse appliquées à la référence de vitesse permettent de limiter les pentes d'un signal de type échelon, afin de réduire l'effort de commande lors d'un changement de référence. Deux saturations sont présentes dans la commande de vitesse. L'intégrateur possède une saturation pour éviter que sa valeur de sortie soit trop élevée lorsque l'erreur ne peut être corrigée. La deuxième saturation appliquée à la sortie de la deuxième sommation permet d'éviter une référence de couple trop élevée.

La fonction de flux détermine une référence de flux pour la commande à flux orienté dans une table de correspondance à partir de la référence de vitesse. La table de correspondance maintient le flux nominal pour une vitesse nulle jusqu'à la vitesse nominale de 1800 RPM, puis décroît pour une vitesse supérieure à la vitesse nominale dans la zone de défluxage pour maintenir le courant nominal sous la puissance nominale.

Il faut noter que la commande de vitesse est activée seulement lorsque le contrôle de magnétisation est terminé, soit lorsque le flux dans la machine est suffisamment élevé. Ce contrôle de magnétisation est fait par la commande à flux orienté. Les détails du contrôle de magnétisation sont dans la section 2.2.1.4.

2.2.1.4 Commande à flux orienté

La commande à flux orienté utilisée est basée sur la commande vectorielle appliquée à la machine asynchrone présentée par Bose [26]. La figure 7 présente le schéma bloc de la commande à flux orienté.

Dans le modèle à valeur moyenne, les sorties de la commande à flux orienté sont des références pour les courants statoriques triphasés (i_{abcs}*). Ces références de courant sont fournies à l'onduleur. La commande détermine les références de courant à partir des références de flux et de couple provenant de la commande de vitesse, des courants statoriques triphasés (i_{abcs}), ainsi que de la vitesse angulaire méc.anique du rotor (ω_m).

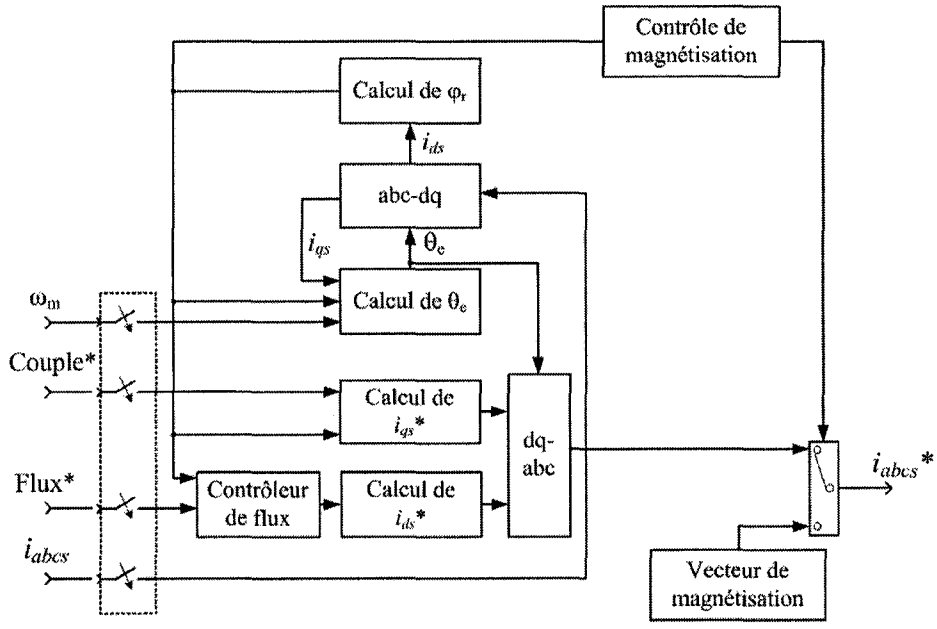


Figure 7 Commande à flux orienté

Le traitement des courants statoriques dans la commande à flux orienté est fait dans le référentiel dq. Une transformation de Park convertie les courants statoriques triphasés (i_{abcs}) en deux composantes de courants orthogonales dans un référentiel tournant synchrone au flux rotorique. Les composantes résultantes de la transformation sont la composante directe des courants statoriques (i_{ds}) et la composante en quadrature des courants statoriques (i_{qs}). L'équation (2.22) présente la transformation de Park, où l'angle utilisé est la position angulaire du flux rotorique (θ_e).

$$\begin{bmatrix} i_{ds} \\ i_{qs} \\ i_{0s} \end{bmatrix} = \frac{2}{3} \begin{bmatrix} \cos(\theta_e) & \cos\left(\theta_e - \frac{2\pi}{3}\right) & \cos\left(\theta_e + \frac{2\pi}{3}\right) \\ -\sin(\theta_e) & -\sin\left(\theta_e - \frac{2\pi}{3}\right) & -\sin\left(\theta_e + \frac{2\pi}{3}\right) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} i_{as} \\ i_{bs} \\ i_{cs} \end{bmatrix} \tag{2.22}$$

La position angulaire du flux rotorique (θ_e) est déterminée par l'intégrale de la somme de la vitesse angulaire mécanique du rotor (ω_m) et de la vitesse angulaire électrique du rotor (ω_r). Cette relation est présentée par l'équation (2.23). La vitesse angulaire mécanique est obtenue avec un capteur de vitesse. La vitesse angulaire électrique est estimée par l'équation (2.24). L'estimation est obtenue par le quotient du produit de la composante en quadrature des courants statoriques (i_{qs}) avec l'inductance de magnétisation (L_m) et du produit du flux rotorique (φ_r) avec la constance de temps du rotor (τ_r). L'inductance de magnétisation et la constance de temps du rotor sont des paramètres préalablement connus. Le flux du rotor est aussi une variable estimée. L'estimation du flux est obtenue par le produit de la composante directe des courants statoriques (i_{ds}), de l'inductance de magnétisation (L_m) et d'un système de premier ordre dont la constante de temps est celle du rotor. Cette relation est présentée par l'équation (2.25).

$$\theta_e = \int(\omega_r + \omega_m)dt \tag{2.23}$$

$$\omega_r = \frac{L_m \cdot i_{qs}}{\tau_r \cdot \varphi_r} \tag{2.24}$$

$$\varphi_r = \frac{L_m \cdot i_{ds}}{1 + \tau_r \cdot s} \tag{2.25}$$

Le contrôleur de flux est basé sur un régulateur de type PI. Il permet de contrôler la dynamique et l'erreur en régime permanent du flux rotorique. Il agit sur l'erreur entre la référence de flux provenant du contrôleur de vitesse et l'estimation du flux rotorique (φ_r). La sortie du contrôleur est une référence de flux contrôlée utilisée pour le calcul de la référence pour la composante directe des courants statoriques. Ce calcul est présenté par l'équation (2.26). La référence pour la composante en quadrature des courants statoriques est obtenue par l'équation (2.27), qui fait intervenir la référence de couple (T_e*) provenant de la commande de vitesse, l'estimation du flux rotorique (φ_r), l'inductance de fuite des enroulements rotoriques (L_r), l'inductance de magnétisation (L_m) et le nombre de paires de pôles de la machine asynchrone (p). Finalement, la transformation inverse de Park est utilisée pour déterminer les courants statoriques triphasés de référence (i_{abcs}*) à partir des composantes orthogonales de référence (i_{ds}* et i_{qs}*) et de la position angulaire du flux rotorique (θ_e). La transformation inverse de Park est définie par l'équation (2.28).

$$i_{ds}^* = \frac{\varphi_r^*}{L_m} \tag{2.26}$$

$$i_{qs}^* = \frac{2}{3} \cdot \frac{1}{p} \cdot \frac{L_r}{L_m} \cdot \frac{T_e^*}{\varphi_r} \tag{2.27}$$

$$\begin{bmatrix} i_{as}^* \\ i_{bs}^* \\ i_{cs}^* \end{bmatrix} = \begin{bmatrix} \cos(\theta_e) & -\sin(\theta_e) & 1 \\ \cos\left(\theta_e - \frac{2\pi}{3}\right) & -\sin\left(\theta_e - \frac{2\pi}{3}\right) & 1 \\ \cos\left(\theta_e + \frac{2\pi}{3}\right) & -\sin\left(\theta_e + \frac{2\pi}{3}\right) & 1 \end{bmatrix} \begin{bmatrix} i_{ds}^* \\ i_{qs}^* \\ i_{0s}^* \end{bmatrix} \tag{2.28}$$

Le contrôle de magnétisation contient la logique qui permet à la commande de passer du mode de magnétisation au mode normal de fonctionnement lorsque le flux initial dans la machine asynchrone est suffisant. En mode de magnétisation, le vecteur de magnétisation est appliqué directement à l'onduleur afin de créer rapidement un flux initial dans le moteur. En mode normal de fonctionnement, les références de courants statoriques sont appliquées à l'onduleur.

2.2.2 Modèle AC6

Le modèle AC6 est un entraînement électrique de machine synchrone à aimants permanents. Il comprend une source triphasée, un convertisseur de puissance, une machine synchrone à aimants permanents et une commande composée d'une commande de vitesse et d'une commande vectorielle. Le convertisseur de puissance est composé d'un redresseur triphasé, d'un hacheur de freinage dynamique et d'un onduleur triphasé. Tous les éléments du modèle de l'entraînement électrique AC6 se retrouvent sous forme de blocs dans les librairies SPS et Electric Drives de Simulink.

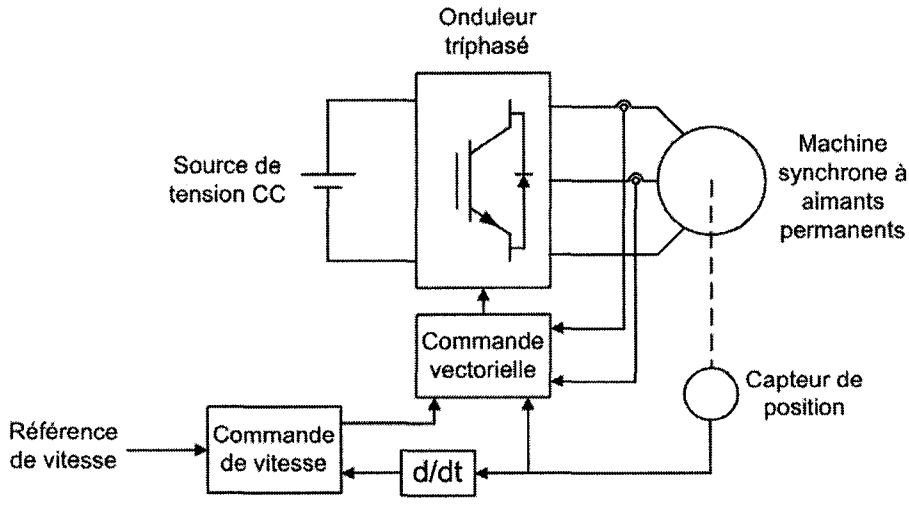


Figure 8 Entraînement électrique de machine synchrone à aimants permanents

Comme pour le modèle AC3, afin de réduire la complexité du modèle, la partie du convertisseur de puissance constituée de la source triphasée, du redresseur et du hacheur est substituée par une source CC idéale. Le schéma bloc du modèle AC6 avec la source CC idéale est présenté à la figure 8 et le schéma Simulink est présenté à la figure 9.

Les sections 2.2.2.1 à 2.2.2.4 présentent une description détaillée des quatre principaux éléments du modèle, soit la machine synchrone à aimants permanents, l'onduleur triphasé, la commande de vitesse et la commande vectorielle.

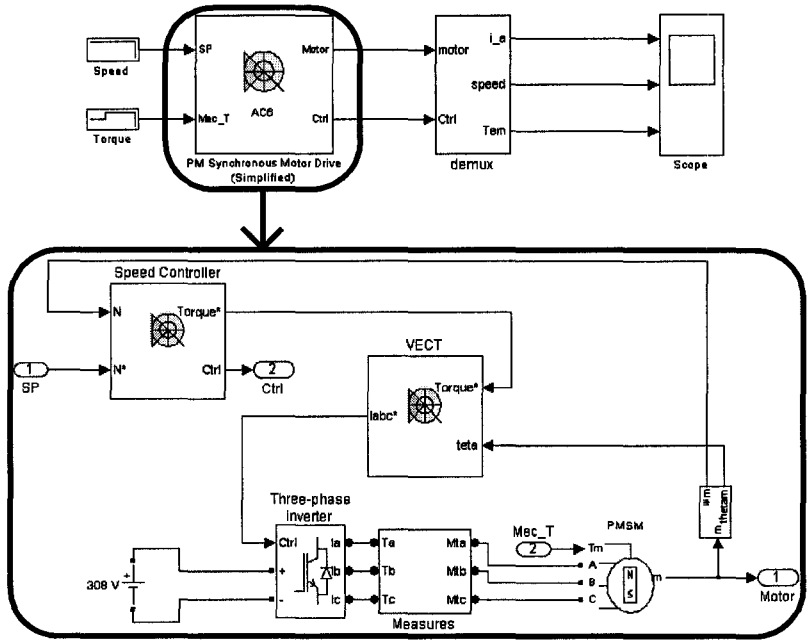


Figure 9 Schéma Simulink du modèle AC6 à valeur moyenne

2.2.2.1 Machine synchrone à aimants permanents

La machine utilisée dans le modèle AC6 est une machine synchrone triphasée à aimants permanents à distribution sinusoïdale de flux. Dans le SPS, la dynamique de cette machine est basée sur le modèle de Krause [25] présenté par les équations (2.29) à

(2.33). Les équations (2.29) à (2.31) modélisent la partie électrique et les équations (2.32) et (2.33) modélisent la partie mécanique. Les équations électriques sont exprimées dans le référentiel dq. La figure 10 présente le schéma électrique de la machine.

Dans la partie électrique, les composantes du courant statorique sont exprimées par les équations (2.29) et (2.30). L'équation (2.31) exprime le couple électromagnétique. Pour la partie mécanique, les équations (2.32) et (2.33) expriment la dynamique de l'accélération et de la vitesse de la machine.

$$\frac{d}{dt}i_d = \frac{1}{L_d}v_d - \frac{R}{L_d}i_d + \frac{L_q}{L_d}p\omega_r i_q \tag{2.29}$$

$$\frac{d}{dt}i_q = \frac{1}{L_q}v_q - \frac{R}{L_q}i_q - \frac{L_d}{L_q}p\omega_r i_d - \frac{\lambda p\omega_r}{L_q} \tag{2.30}$$

$$T_e = 1{,}5p[\lambda i_q + (L_d - L_q)i_d i_q] \tag{2.31}$$

$$\frac{d}{dt}\omega_r = \frac{1}{J}(T_e - F\omega_r - T_m) \tag{2.32}$$

$$\frac{d}{dt}\theta_r = \omega_r \tag{2.33}$$

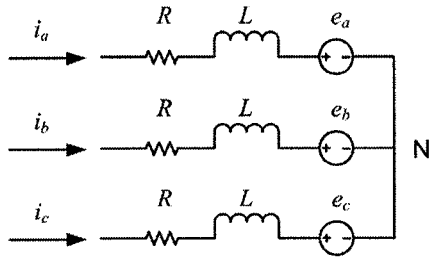


Figure 10 Schéma électrique de la machine synchrone à aimants permanents

La figure 11 présente la fenêtre de configuration des divers paramètres électriques et mécaniques du modèle de simulation de la machine synchrone à aimants permanents. La résistance des enroulements statoriques est de 0,2 Ω, les inductances statoriques dans le

référentiel dq sont de 8,5 mH et le flux induit par les aimants permanents du rotor dans les phases du stator est de 0,175 Wb. La machine est constituée de quatre paires de pôles, l'inertie combinée du rotor et de la charge est de 0,089 $kg \cdot m^2$ et le coefficient de friction visqueuse combiné du rotor et de la charge est de 0,005 $N \cdot m \cdot s$.

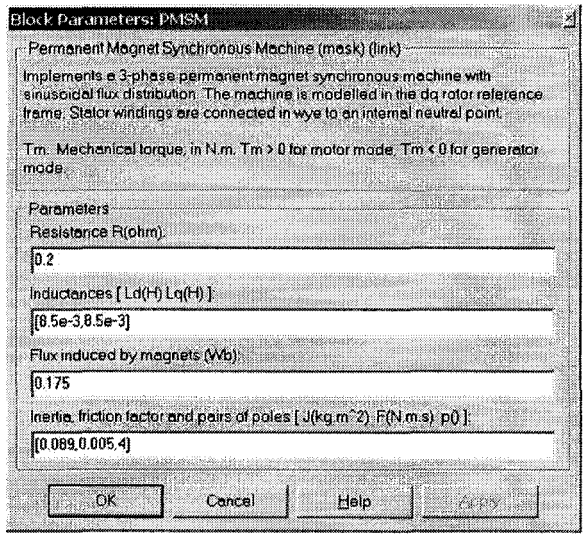


Figure 11 Paramètres de la machine synchrone à aimants permanents

2.2.2.2 Onduleur triphasé

L'onduleur triphasé du modèle AC6 est pratiquement identique à celui du modèle AC3. La seule différence se situe au niveau de la logique de sélection de l'interrupteur. La condition relative au contrôle de magnétisation présente dans le modèle AC3 est inexistante dans le modèle AC6, puisque la machine synchrone à aimants permanents ne nécessite pas une période de magnétisation au démarrage. La figure 12 présente le schéma bloc de l'onduleur. La section 2.2.1.2 présente les détails relatifs au fonctionnement de l'onduleur triphasé des modèles à valeur moyenne.

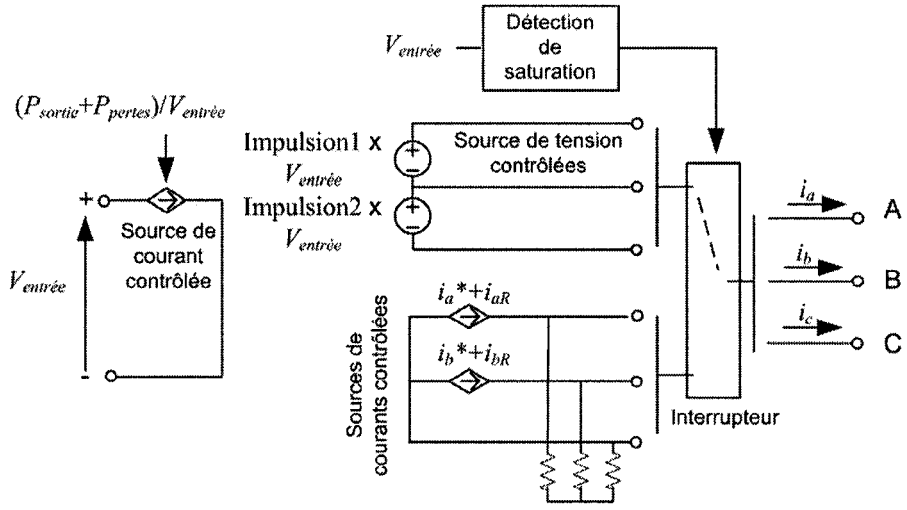


Figure 12 Onduleur triphasé

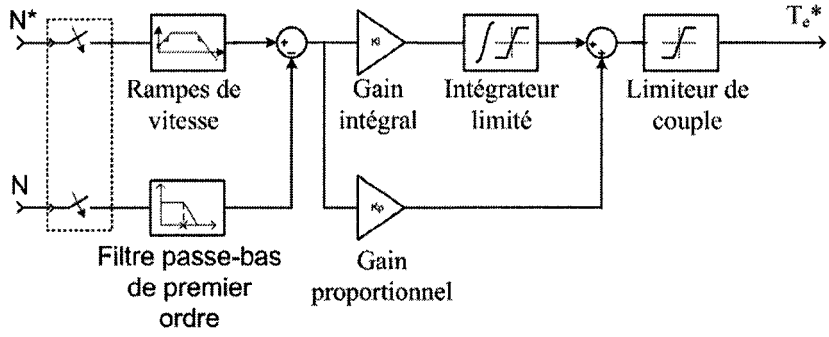


Figure 13 Commande de vitesse

2.2.2.3 Commande de vitesse

La commande de vitesse du modèle AC6 est aussi pratiquement identique à celle du modèle AC3. La seule différence avec le modèle AC3 dans ce cas-ci est l'absence de la sortie qui est une référence de flux. Puisque le rotor est composé d'aimants permanents, cette référence n'est pas utile dans le modèle AC6 car le flux rotorique ne peut être régulé. La commande de vitesse présente donc une seule sortie, qui est une référence de

couple pour la commande vectorielle. La section 2.2.1.3 présente les détails relatifs au fonctionnement de la commande de vitesse. La figure 13 présente le schéma bloc de la commande de vitesse.

2.2.2.4 Commande vectorielle

La commande vectorielle utilisée dans le modèle AC6 est présentée par Bose [26]. La figure 14 présente le schéma bloc de la commande vectorielle. Dans le modèle à valeur moyenne, les sorties de la commande vectorielle sont des références pour les courants statoriques triphasés (i_{abc}*). Ces références de courant sont fournies à l'onduleur. La commande détermine les références de courant à partir de la référence de couple provenant de la commande de vitesse et de la position angulaire mécanique du rotor (θ_r).

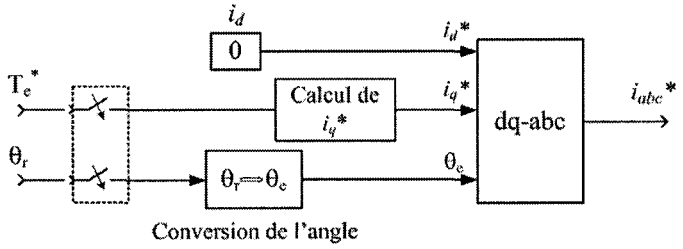


Figure 14 Commande vectorielle

$$i_q^* = \frac{2}{3} \cdot \frac{1}{p} \cdot \frac{T_e^*}{\lambda} \tag{2.34}$$

$$\theta_e = p \cdot \theta_r \tag{2.35}$$

Une transformation inverse de Park converti les références pour les deux composantes de courants orthogonales (i_d* et i_q*) du référentiel dq tournant synchrone au flux rotorique en références pour les courants statoriques triphasés (i_{abc}*). La transformation inverse de Park est présentée par l'équation (2.28) et l'angle utilisé est la position angulaire du flux

rotorique (θ_e). Cette position est obtenue avec la relation entre la position angulaire mécanique du rotor (θ_r) et le nombre de paires de pôles de la machine selon l'équation (2.35). La référence pour la composante directe des courants statoriques (i_d*) est posée nulle. De cette façon il n'y a pas de courant réactif induit dans la machine et les pertes sont diminuées. La référence pour la composante en quadrature des courants statoriques (i_q*) est obtenue par l'équation (2.34), qui fait intervenir la référence de couple (T_e*) provenant de la commande de vitesse, le flux induit par les aimants permanents du rotor (λ) et le nombre de paires de pôles de la machine (p).

2.3 Conclusion

Ce chapitre a présenté une description détaillée des deux modèles d'entraînement électrique à valeur moyenne utilisés dans la première partie du projet de recherche. Le premier modèle est un entraînement électrique de machine asynchrone à commande à flux orienté et le deuxième est un entraînement électrique de machine synchrone à aimants permanents. La démarche employée pour réaliser la simulation en temps réel avec commande externe de ces deux modèles est présentée dans le chapitre 3 et les résultats obtenus sont présentés dans le chapitre 4.

CHAPITRE 3

IMPLÉMENTATION DES MODÈLES À VALEUR MOYENNE

3.1 Introduction

Ce chapitre présente la démarche utilisée pour réaliser la simulation en temps réel des modèles à valeur moyenne avec commande externe. Tous les outils matériels et logiciels utilisés ainsi que les étapes nécessaires à la réalisation de cette partie du projet sont présentés.

La section 3.2 présente les outils et les étapes nécessaires à l'implémentation de la commande. Cette section présente la carte de prototypage DS1104 de dSPACE. Les étapes de génération, de téléchargement et de compilation du code C d'une application temps réel sont décrites en détail. La démarche complète de configuration d'un modèle Simulink pour l'implémentation du code C dans dSPACE est exposée. De plus, la méthode pour démarrer une application temps réel à l'aide du logiciel ControlDesk est expliquée.

La section 3.3 présente les outils et les étapes nécessaires à l'implémentation de l'électronique de puissance et de la machine, qui doivent être simulé en temps réel à l'aide d'un simulateur. Le simulateur en temps réel est réalisé grâce à l'outil xPC Target. La description de l'environnement et du fonctionnement de cet outil est faite en détail. L'environnement est composé de deux ordinateurs, soit l'hôte et la cible, ainsi que de deux cartes d'E/S PCI-MIO-16E-4 de National Instruments. La description présente entre autre la façon dont un modèle Simulink est généré en code C et compilé pour constituer une application cible qui est exécutée en temps réel. Les détails de configuration du modèle Simulink pour utiliser les cartes d'E/S et générer une application cible sont décrits.

Les résultats obtenus pour la simulation en temps réel avec commande externe des modèles AC3 et AC6 sont présentés dans les sections 4.4 et 4.5. Ces résultats sont validés par la comparaison avec les résultats de simulation en temps différé présentés dans les sections 4.2 et 4.3.

3.2 Implémentation de la commande

L'implémentation de la commande est réalisée à l'aide de la carte de prototypage DS1104 de la compagnie dSPACE. L'installation et la configuration de la carte DS1104 ont été réalisées à l'aide du document [7]. Les documents [8], [9] et [10] ont été nécessaires pour l'installation et l'utilisation des logiciels de dSPACE.

Cette carte comprend un processeur principal PowerPC de 250 MHz et un DSP esclave. Le DSP est dédié à la génération d'impulsions par modulation de largeur d'impulsion (MLI) et n'est pas utilisé dans le cadre du projet. La carte est munie de quatre entrées analogiques avec des convertisseurs analogiques à numériques (CAN) de 12 bits possédants des temps de conversion de 800 ns, de quatre entrées analogiques multiplexées avec un seul CAN de 16 bits possédant un temps de conversion de 2 µs, de huit sorties analogiques avec des convertisseurs numériques à analogiques (CNA) de 16 bits possédant des temps de conversion de 10 µs et de vingt E/S numériques. Les entrées et les sorties analogiques ont une plage de tension de -10 V à 10 V. L'accès aux signaux des E/S est réalisé avec le panneau CP1104. L'accès aux signaux analogiques se fait via des prises BNC et l'accès aux signaux numériques se fait via des prises Sub-D. L'annexe 1 présente les spécifications techniques de la carte DS1104 et du panneau CP1104.

La carte DS1104 est installée dans une fente PCI de l'ordinateur hôte. Elle est liée à MATLAB, Simulink et Real-Time Workshop par un outil nommé RTI (Real-Time Interface). Cet outil permet, lors du développement d'un modèle dans Simulink, l'accès aux E/S de la carte et l'implémentation du modèle Simulink dans la carte par la génération

automatique de code. Le schéma du montage matériel et logiciel complet utilisé dans la réalisation de la première partie du projet est présenté par la figure 15.

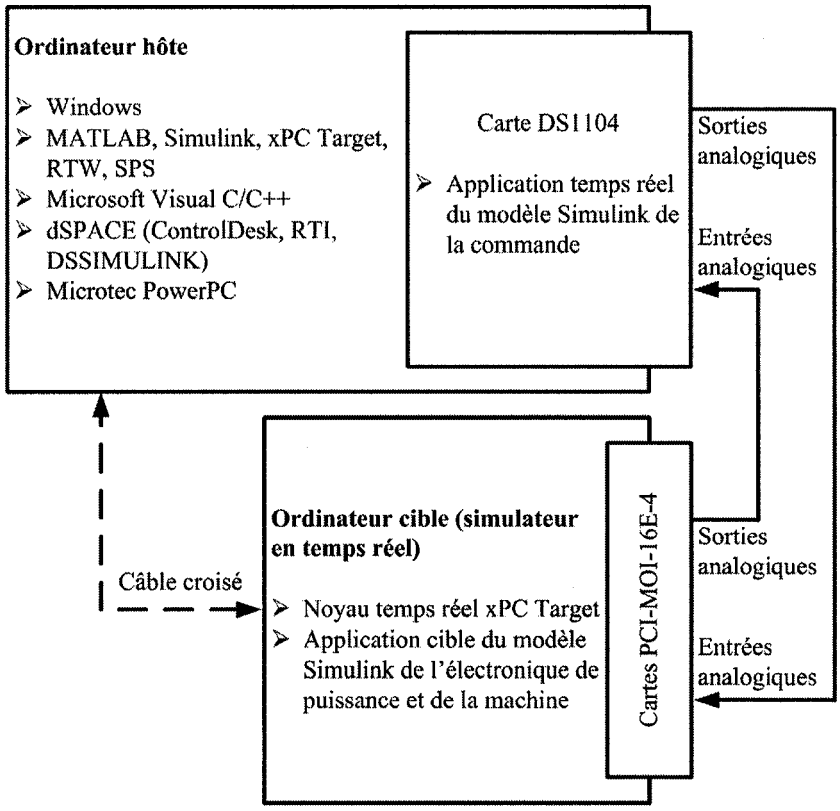


Figure 15 Schéma du montage

Lors de l'installation du logiciel RTI, la librairie dSPACE RTI1104 est incluse dans Simulink via une interface entre MATLAB et dSPACE nommée MLIB. La librairie dSPACE RTI1104 donne accès à toutes les fonctionnalités de la carte. Dans cette partie du projet on s'intéresse particulièrement aux E/S analogiques. Ces E/S sont représentées sous forme de blocs dans Simulink. On insère et connecte les blocs de la librairie au modèle Simulink comme n'importe quel autre bloc standard. Il suffit ensuite de configurer les différents blocs utilisés.

Pour l'implémentation d'un modèle Simulink dans la carte DS1104, RTI doit créer une application temps réel. Real-Time Workshop et RTI génèrent le code C d'une application temps réel à partir du modèle Simulink et des blocs d'E/S de dSPACE en utilisant le format de code temps réel de Real-Time Workshop. Le code est ensuite téléchargé dans la carte DS1104 et compilé par le compilateur Microtec PowerPC pour produire l'application temps réel exécutable. Les étapes du processus de génération d'une application temps réel sont présentées par la figure 16.

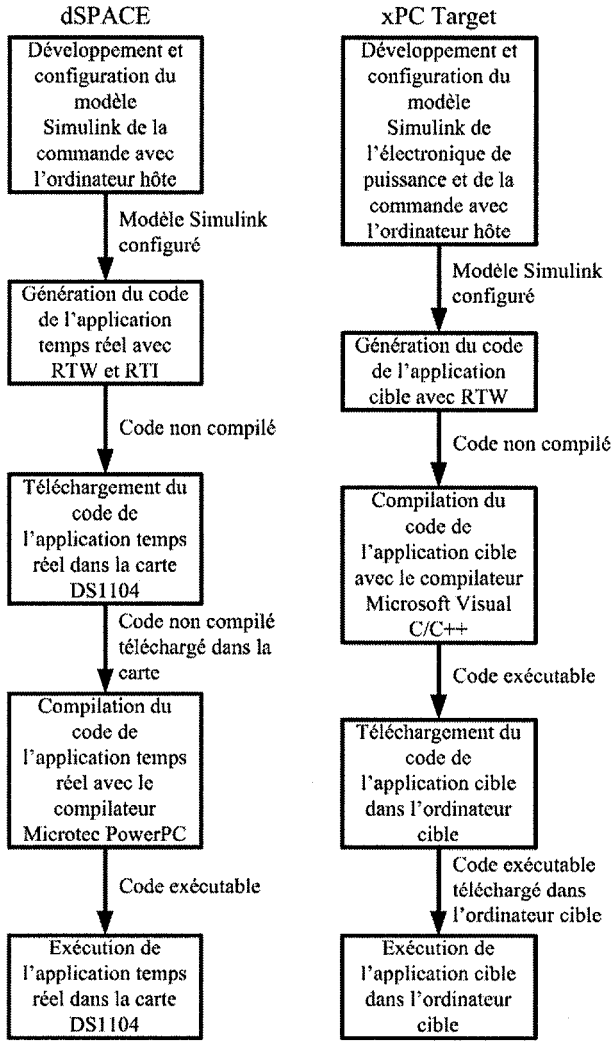


Figure 16 Processus de génération des applications cible et temps réel

Par défaut, l'application temps réel sur la carte démarre automatiquement à la suite de la compilation. Dans la configuration du modèle dans Simulink, on peut spécifier l'état initial de l'application suite à la compilation. Les trois états possibles sont RUN, PAUSE et STOP. Par la suite, on peut modifier l'état de l'application via une variable nommée *SimState*. Cette variable est créée lors de la génération de code et est seulement disponible via le logiciel ControlDesk. Ce logiciel fournit toutes les fonctionnalités de contrôle et d'affichage de la carte DS1104. ControlDesk agit indépendamment de MATLAB et Simulink après la génération du code. Les variables et données nécessaires à ControlDesk sont transmises de Simulink lors de la génération du code via une interface nommée DSSIMULINK. Dans le cadre du projet, ControlDesk est utilisé seulement pour contrôler la variable *SimState*. Lors de la configuration d'un modèle dans Simulink, on spécifie que l'état initial de l'application temps réel soit sur PAUSE. Par la suite, grâce au logiciel ControlDesk, on fait démarrer l'application temps réel au moment désiré via la variable *SimState*.

3.2.1 Développement et configuration des modèles

Dans le projet de recherche, les applications temps réel sont créées à partir des commandes des modèles AC3 et AC6 à valeur moyenne. Dans le cas d'AC3 la commande est constituée d'une commande de vitesse et d'une commande à flux orienté. Pour AC6 la commande est constituée d'une commande de vitesse et d'une commande vectorielle. Le chapitre 2 présente la description détaillée des commandes de chaque modèle. Les figures 22, 23, 25 et 26 présentent les schémas Simulink pour les commandes des modèles AC3 et AC6. Ces figures présentent les blocs d'E/S analogiques de la carte DS1104. Ces blocs sont disponibles dans la librairie dSPACE RTI1104 de Simulink. La figure 17 présente les volets sous lesquels sont disponibles les E/S analogiques de la carte DS1104.

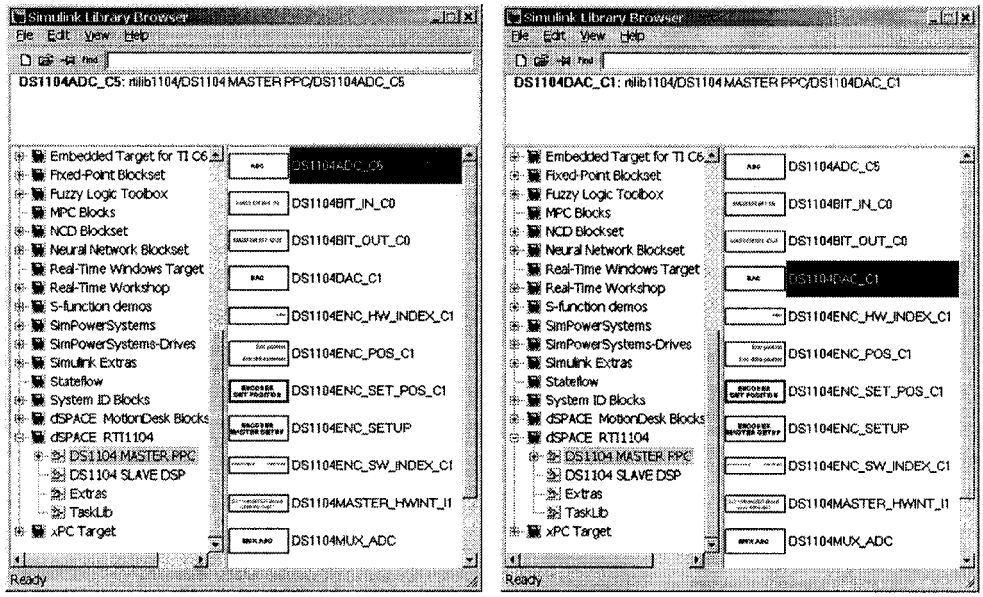


Figure 17 Blocs d'E/S analogiques de la carte DS1104

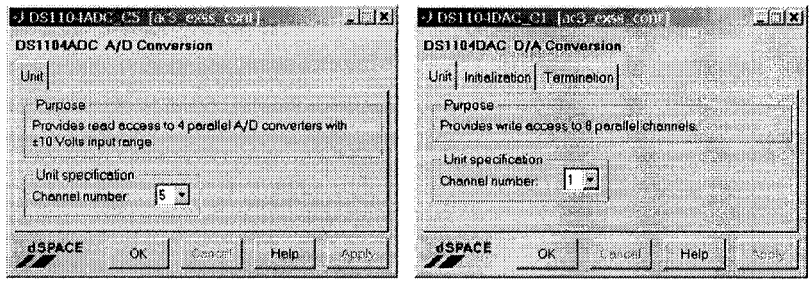


Figure 18 Fenêtres de configuration des blocs d'E/S analogiques

Les blocs d'entrées analogiques ne nécessitent pas de configuration particulière. La seule information requise est le numéro de l'entrée désirée. Les plages de tension des entrées sont de -10 V à 10 V et la période d'échantillonnage est celle du modèle, soit de 20 µs dans tous les cas. Les blocs de sorties analogiques, quant à eux, nécessitent un peu plus d'information. Ces blocs nécessitent le numéro de la sortie désirée, l'état initial et l'état final. Ces états sont fixés à 0 V pour toutes les sorties dans le cadre du projet. La plage de

tension des sorties est de -10 V à 10 V et la période d'échantillonnage est de 20 µs dans tous les cas. La figure 18 présente les fenêtres de configuration des blocs d'entrée et de sortie analogiques de dSPACE.

Le modèle de Simulink doit être configuré afin d'être généré en code C pour créer une application temps réel pour dSPACE. Pour faire cette configuration, il faut choisir *Configuration parameters* à partir du menu *Simulation* dans l'interface de Simulink. Ceci ouvre la fenêtre de configuration des paramètres qui est présentée à la figure 19. Dans l'onglet *Solver*, il faut inscrire 0 pour le temps de départ dans la case *Start time* et déterminer le temps d'arrêt dans la case *Stop time*. Ensuite, on choisi l'algorithme de calcul à pas fixe de type discret en sélectionnant *Fixed-step* dans la case *Type* et *Discrete* dans la case *Solver*. Par la suite, il faut inscrire la période d'échantillonnage de 20 µs dans la case *Fixed-step size*. Finalement, dans la case *Mode*, il faut spécifier que le modèle fonctionne en tâche unique en sélectionnant *Single Tasking*.

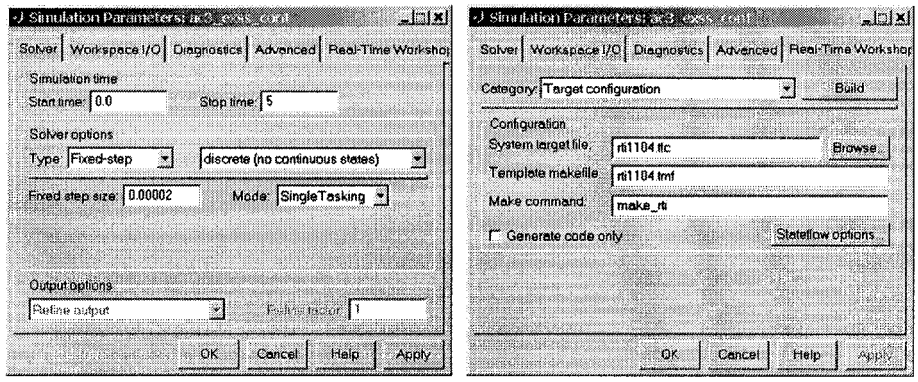


Figure 19 Fenêtres de configuration des paramètres de dSPACE

Dans l'onglet *Real-Time Workshop*, en appuyant sur *Browse*, on sélectionne *rti1104.tlc*, qui spécifie la génération du code C du modèle pour la carte DS1104. Par la suite, on sélectionne *RTI simulation options* dans la case *Category*. On spécifie que l'exécution est réalisée en temps réel en sélectionnant *real-time* dans la case *Execution mode*. Il faut aussi

spécifier que l'état initial de l'application temps réel soit pause en sélectionnant *pause* dans la case *Initial simulation state.*

Afin de lancer la génération, le téléchargement et la compilation du code, il suffit d'appuyer sur le bouton *build* de la fenêtre de configuration des paramètres sous l'onglet *Real-Time Workshop*, présenté à la figure 19. Ces étapes sont exécutées automatiquement avec Real-Time Interface (RTI), Real-Time Workshop (RTW) et le compilateur C Microtec PowerPC.

Lorsque les étapes de génération, de téléchargement et de compilation du code sont terminées, l'application temps réel sur la carte DS1104 est prête à être démarrée. Cette application est démarrée via la variable *SimState* qui est disponible dans ControlDesk. La figure 20 présente l'interface de ControlDesk.

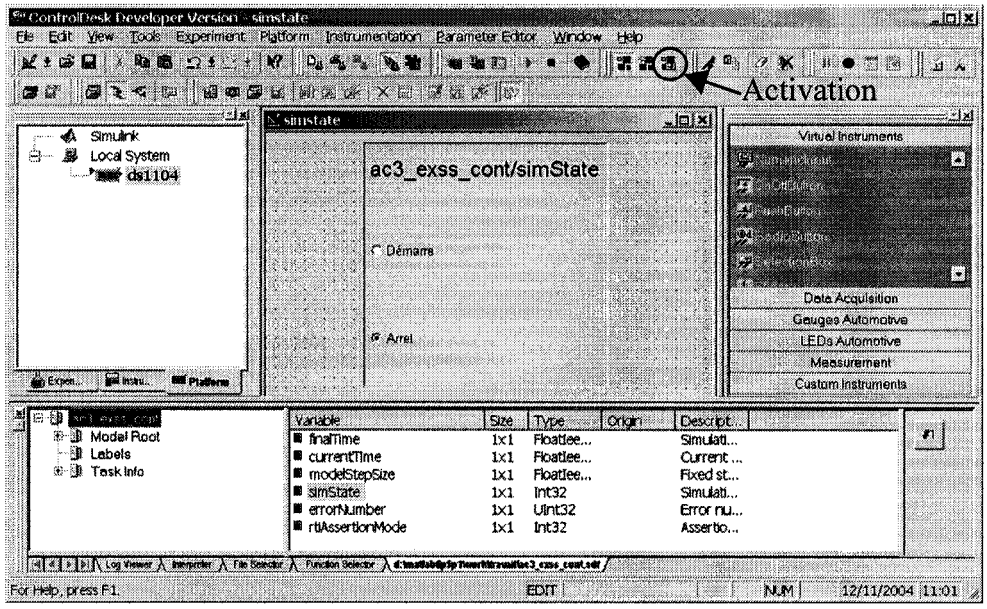


Figure 20 Interface de ControlDesk

Lors de la génération du code, DSSIMULINK crée un fichier dans ControlDesk contenant tous les signaux et variables du modèle Simulink. Ce fichier est présenté au bas de la figure 20. La variable *SimState* est contenue dans ce fichier. Pour utiliser cette variable, une interface de contrôle est développée dans la fenêtre de développement de ControlDesk. L'interface de contrôle comprend un seul bouton de type *RadioButton.* Ce bouton est sélectionné dans le sélecteur d'instruments et inséré dans la fenêtre de développement de ControlDesk. Sur la figure 20, le sélecteur d'instrument est situé à droite et la fenêtre de développement est située au centre. Le bouton est associé à la variable *SimState* en faisant glisser la variable sur le bouton. Il faut ensuite double cliquer sur le bouton pour accéder à sa configuration. La figure 21 présente la fenêtre de configuration du *RadioButton.*

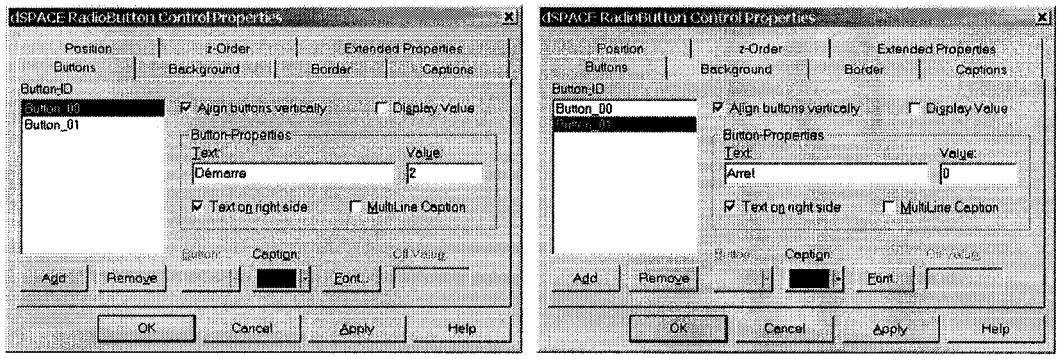


Figure 21 Fenêtres de configuration du *RadioButton*

Des valeurs sont associées aux états de la variable. La valeur 2 est associée à RUN, la valeur 1 à PAUSE et la valeur 0 à STOP. Le bouton utilisé est configuré pour sélectionner les états RUN et STOP. La figure 21 présente la configuration du bouton pour démarrer et arrêter la simulation. Pour démarrer l'application temps réel, le bouton d'activation doit tout d'abord être appuyé afin d'activer l'interface de contrôle dans ControlDesk. Ce bouton est identifié sur la figure 20. Lorsque l'interface de contrôle est

activée, l'application temps réel est démarrée en appuyant sur la case *Démarre* du *RadioButton*.

3.2.1.1 Modèle AC3

Comme présenté dans la section 2.2.1, le modèle AC3 comprend une commande composée d'une commande de vitesse et d'une commande à flux orienté (FOC). Cette commande nécessite quatre entrées analogiques de la carte DS1104 pour recevoir la vitesse mécanique du moteur (ω_m) et les courants statoriques triphasés (i_{abcs}) provenant du simulateur. Deux sorties analogiques sont nécessaires afin de transmettre les références des courants statoriques i_{as}* et i_{bs}* à l'onduleur triphasé du simulateur. La figure 22 présente le schéma Simulink de la commande du modèle AC3 à valeur moyenne.

Les quatre entrées analogiques non multiplexées de la carte DS1104, dont les numéros sont 5, 6, 7 et 8, sont utilisées. Les deux sorties analogiques utilisées sont les sorties 1 et 2. Il faut inscrire les numéros des entrées et des sorties dans les fenêtres de configuration des blocs comme présenté à la figure 18. Des gains sont ajoutés aux entrées et aux sorties pour faire correspondre les plages de valeurs. À partir des résultats de simulation en temps différé du modèle AC3 à valeur moyenne présentés dans la section 4.2 par la figure 36, la plage maximale de fonctionnement observée pour les courants est de -500 A à 500 A et la plage maximale pour la vitesse est de 0 RPM à 500 RPM. Dans le cas des E/S analogiques de dSPACE, elles fonctionnent sous une plage de valeur de -1 à 1 qui correspond à une plage de tension de -10 V à 10 V. Pour faire correspondre la plage des sorties, les références de courants sont divisées par un gain de 500 A, puis pour faire correspondre la plage des entrées, les courants et la vitesse mécanique sont multipliés par des gains de 500 A et 500 RPM.

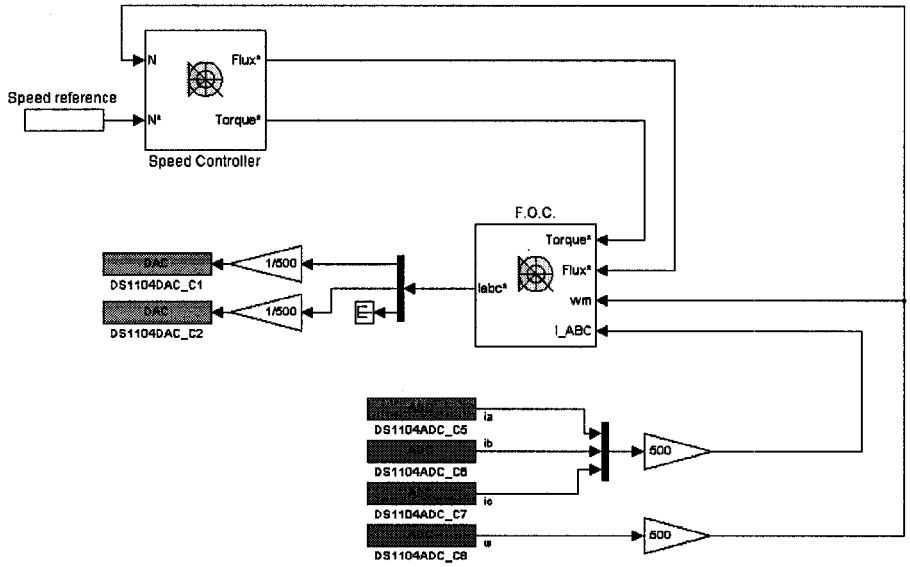


Figure 22 Schéma Simulink de la commande

Les résultats obtenus pour la simulation en temps réel du modèle AC3 à valeur moyenne avec commande externe sont présentés dans la section 4.4. Les premiers résultats n'étant pas satisfaisants, un modèle amélioré de la commande a été réalisé. Le schéma Simulink du modèle amélioré est présenté par la figure 23. Ce modèle comporte deux modifications par rapport au premier modèle.

La première modification est le passage de la vitesse mécanique de l'entrée analogique numéro 8 avec un CAN de 12 bits vers l'entrée analogique numéro 1 avec un CAN de 16 bits. Il faut changer le bloc d'entrée analogique pour le bloc d'entrées analogiques multiplexées. Pour configurer ce bloc il faut seulement sélectionner l'entrée numéro 1. Le passage de toutes les entrées vers les quatre entrées analogiques multiplexées avec le CAN de 16 bits n'est pas possible, puisque qu'après expérimentation, il a été observé que le temps de conversion était beaucoup trop élevé pour la simulation en temps réel.

La deuxième modification est le doublage de la résolution des signaux de sortie. Le principe utilisé est de séparer un signal en deux et de transmettre chaque partie sur une sortie différente. Puisque le signal est transmis sur deux sorties avec des CNA de 12 bits, la résolution du signal augmente à 24 bits. Le schéma de principe est présenté par la figure 24.

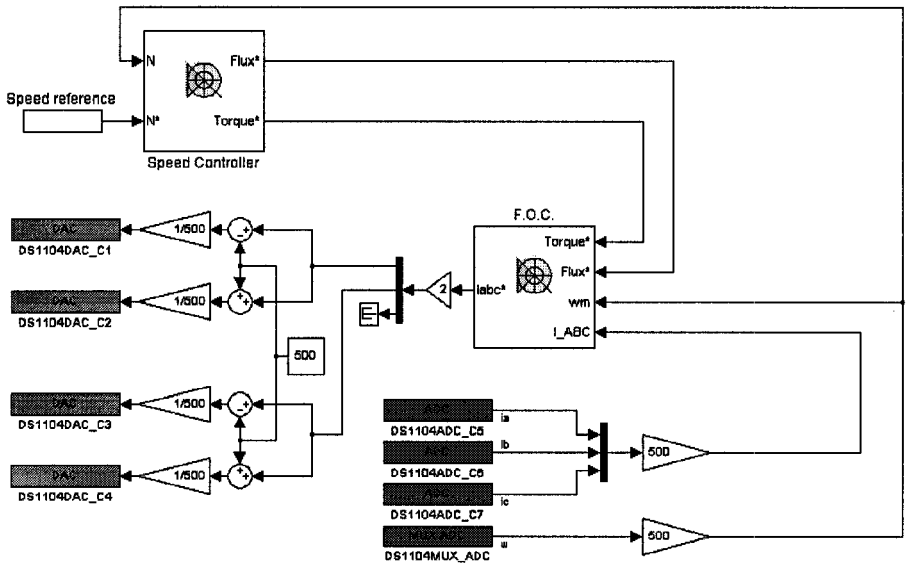


Figure 23 Schéma Simulink amélioré de la commande

Dans ce cas-ci, les signaux de sortie sont les références des courants statoriques. Les signaux sont premièrement multipliés par un gain de 2 afin de doubler leurs amplitudes et ainsi obtenir une plage maximale de courant de -1 000 A à 1 000 A. Un biais de 500 A est ensuite soustrait aux signaux pour obtenir une plage de -1 500 A à 500 A. Afin de faire correspondre les plages de sorties, les signaux sont divisés par un gain de 500 A. La plage résultante est de -3 à 1. Les sorties analogiques sont saturées pour les valeurs inférieures à -1, ce qui a pour effet de tronquer les alternances négatives et ainsi transmettre seulement les alternances positives des signaux. Pour obtenir les alternances négatives, la même démarche est utilisée en faisant l'addition du biais de 500 A. Les alternances positives des signaux sont transmises par les sorties 1 et 3, puis les

alternances négatives par les sorties 2 et 4. Les signaux sont par la suite reconstitués dans le simulateur et les détails sont présentés dans la section 3.3.1.1.

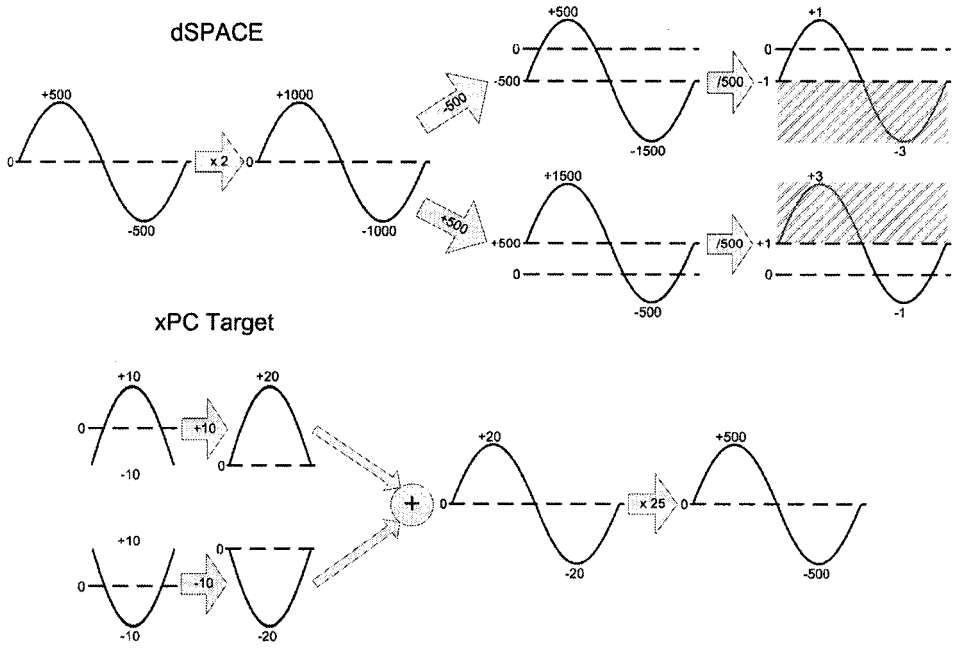


Figure 24 Schéma de principe du doublage de la précision

3.2.1.2 Modèle AC6

Comme présenté dans la section 2.2.2, le modèle AC6 comprend une commande composée d'une commande de vitesse et d'une commande vectorielle. Cette commande nécessite deux entrées analogiques de la carte DS1104 pour recevoir la vitesse mécanique du moteur (ω_m) et la position angulaire mécanique du rotor (θ_r) provenant du simulateur. Deux sorties analogiques sont nécessaires afin de transmettre les références des courants statoriques i_a* et i_b* à l'onduleur triphasé du simulateur. La figure 25 présente le schéma Simulink de la commande du modèle AC6 à valeur moyenne.

Les deux entrées analogiques 5 et 6 de la carte DS1104 sont utilisées. Les deux sorties analogiques utilisées sont les sorties 1 et 2. Des gains sont ajoutés aux entrées et aux

sorties pour faire correspondre les plages de valeurs. Les résultats de simulation en temps différé du modèle AC6 à valeur moyenne sont présentés dans la section 4.3 par les figures 37 et 38. La plage maximale de fonctionnement observée pour les courants est de -15 A à 15 A selon la figure 37. D'après la figure 38, la plage maximale pour la vitesse mécanique est de -1 rad/s à 32 rad/s et la plage maximale pour la position angulaire est de 0 rad à 32 rad. Pour faire correspondre la plage des sorties, les références de courants sont divisées par un gain de 15 A, puis pour faire correspondre la plage des entrées, la vitesse mécanique et la position angulaire sont multipliées par des gains de 32 rad/s et de 32 rad.

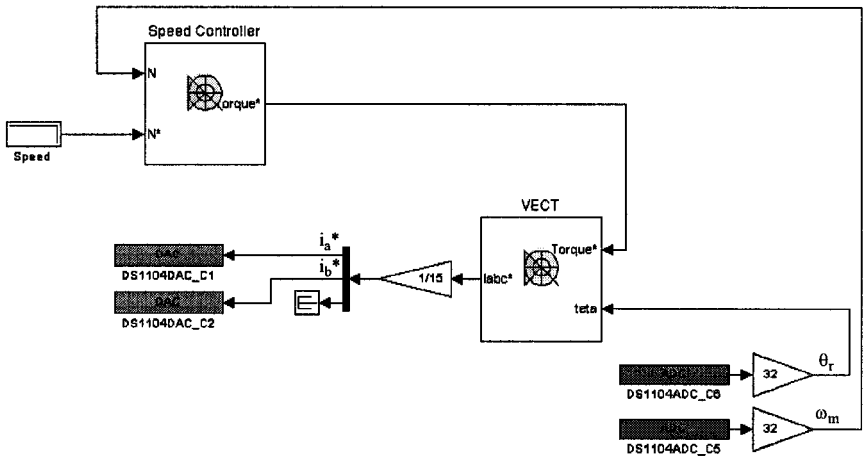


Figure 25 Schéma Simulink de la commande

Les résultats obtenus pour la simulation en temps réel du modèle AC6 à valeur moyenne avec commande externe sont présentés dans la section 4.5. Les premiers résultats n'étant pas satisfaisants, un modèle amélioré de la commande à été réalisé. Le schéma Simulink du modèle amélioré est présenté par la figure 26. Ce modèle comporte deux modifications par rapport au premier modèle.

La première modification est le doublage de la résolution des signaux de sortie. Le principe utilisé est le même que dans la section 3.2.1.1 comme présenté par la figure 24.

Les signaux de sortie sont les références des courants statoriques. Les signaux sont premièrement multipliés par un gain de 2 afin de doubler leurs amplitudes et ainsi obtenir une plage maximale de courant de -30 A à 30 A. Un biais de 15 A est ensuite soustrait aux signaux pour obtenir une plage de -45 A à 15 A. Afin de faire correspondre les plages de sorties, les signaux sont divisés par un gain de 15 A. La plage résultante est de -3 à 1. Puisque les sorties analogiques sont saturées pour les valeurs inférieures à -1, les alternances négatives sont tronquées et ainsi seulement les alternances positives des signaux sont transmises. Pour obtenir les alternances négatives, la même démarche est utilisée en faisant l'addition du biais de 15 A. Les alternances positives des signaux sont transmises par les sorties 1 et 3, puis les alternances négatives par les sorties 2 et 4. Les signaux sont par la suite reconstitués dans le simulateur et les détails sont présentés dans la section 3.3.1.2.

La deuxième modification est le doublage de la résolution des signaux d'entrée. Cette modification n'est pas possible dans le cas du modèle AC3 car les quatre sorties analogiques des cartes PCI-MIO-16E-4 sont utilisées. Dans le cas du modèle AC6, les deux signaux de sortie du simulateur peuvent être doublés. La commande reçoit donc la vitesse mécanique et la position angulaire sur quatre entrées et doit reconstituer ces signaux. Les deux parties de la vitesse mécanique sont reçues par les entrées 5 et 6, puis les deux parties de la position angulaire par les entrées 7 et 8.

Dans le cas de la vitesse mécanique, le signal à reconstituer possède une plage de -1 rad/ s à 32 rad/s. Les signaux reçus par les entrées 5 et 6 possèdent une plage de -1 à 1. L'entrée 5 reçoit le signal qui correspond à la plage de 16,5 rad/s à 32 rad/s et l'entrée 6 reçoit le signal qui correspond à la plage de -1 rad/s à 16,5 rad/s. Un biais de 1 est additionné au signal de l'entrée 5 et soustrait au signal de l'entrée 6 pour obtenir des plages respectives de 0 à 2 et de -2 à 0. Les deux signaux sont ensuite additionnés pour obtenir un seul signal possédant une plage de -2 à 2. Le signal est multiplié par un gain de 16,5/2 rad/s ce qui produit une plage de -16,5 rad/s à 16,5 rad/s. Finalement, la

reconstitution du signal est complète en additionnant un biais de 15,5 rad/s pour obtenir la plage de -1 rad/s à 32 rad/s. Cette démarche suit similairement le principe présenté par la figure 24 pour reconstituer un signal provenant de la commande dans le simulateur. La même démarche est utilisée pour reconstituer le signal de la position angulaire qui possède une plage de 0 rad à 32 rad.

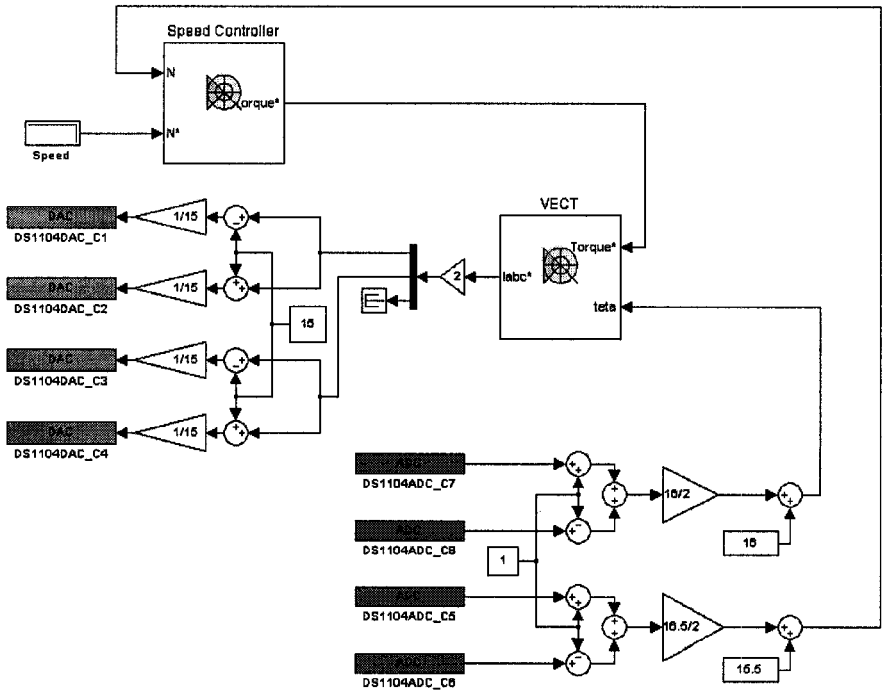


Figure 26 Schéma Simulink amélioré de la commande

3.3 Implémentation de l'électronique de puissance et de la machine

La simulation en temps réel de l'électronique de puissance et de la machine est réalisée avec l'ordinateur cible. Cet ordinateur constitue le simulateur en temps réel. Le schéma du montage matériel et logiciel complet utilisé dans la réalisation de la première partie du projet est présenté par la figure 15. Le simulateur communique avec la commande externe via deux cartes d'E/S PCI-MIO-16E-4 de National Instruments installées dans

les fentes PCI de l'ordinateur cible. Chaque carte possède seize entrées analogiques avec des CAN de 12 bits possédant un temps de conversion de 2 µs, deux sorties analogiques avec des CNA de 12 bits possédant un temps de conversion de 1 µs, huit E/S numériques et deux compteurs de 24 bits. Les entrées et les sorties analogiques ont une plage maximale de tension de -10 V à 10 V. L'accès aux signaux des E/S est réalisé avec des panneaux BNC-2110. L'accès aux signaux analogiques se fait via des prises BNC et l'accès aux signaux numériques se fait via un terminal de 30 broches. L'annexe 2 présente les spécifications techniques de la carte PCI-MIO-16E-4 sous la référence NI 6040E.

xPC Target est l'outil de simulation utilisé pour réaliser la simulation en temps réel des modèles à valeur moyenne. Les documents [3] et [4] ont été nécessaires pour l'utilisation d'xPC Target. Cet outil utilise des ordinateurs personnels standard. Son environnement est composé de l'ordinateur hôte et de l'ordinateur cible. L'ordinateur hôte sert d'outil de développement avec MATLAB et Simulink pour créer des modèles et l'ordinateur cible permet de réaliser les simulations en temps réel.

xPC Target permet l'addition des blocs d'E/S des cartes PCI-MIO-16E-4 dans les modèles. Il utilise Real-Time Workshop et le compilateur Microsoft Visual C/C++ pour créer du code exécutable. Le code est téléchargé de l'ordinateur hôte vers l'ordinateur cible qui est opéré par le noyau temps réel d'xPC Target. Après le téléchargement du code exécutable, l'application cible peut être exécutée en temps réel. Les étapes du processus de génération d'une application cible sont présentées par la figure 16.

Dans le projet de recherche, l'ordinateur hôte utilisé est muni d'un microprocesseur Intel Pentium 4 de 1,7 GHz et de 512 MB de mémoire RAM. L'ordinateur cible est muni d'un microprocesseur Intel Pentium 4 de 2,0 GHz et de 512 MB de mémoire RAM. D'un point de vue logiciel, la version 6.5.1 de MATLAB est utilisée. Cette version de MATLAB comprend la version 2.0.1 d'xPC Target, la version 5.1 de Simulink, la

version 5.1 de RTW et la version 3.0 de la librairie de SimPowerSystems (SPS). La version 6.0 du compilateur Microsoft Visual C/C++ est utilisée.

xPC Target ne requiert pas de système d'exploitation comme DOS, Windows ou Linux sur l'ordinateur cible. L'ordinateur cible est démarré avec une disquette de démarrage qui comprend le noyau temps réel d'xPC Target. Le noyau gère de manière optimale l'ordinateur cible pour la simulation en temps réel et n'accède pas le disque dur.

La disquette de démarrage est créée avec l'ordinateur hôte à l'aide d'un outil de configuration nommé xPC Target Setup. La figure 27 présente la fenêtre de l'outil de configuration, qui est obtenue en entrant la commande *xpcsetup* à l'invite de MATLAB.

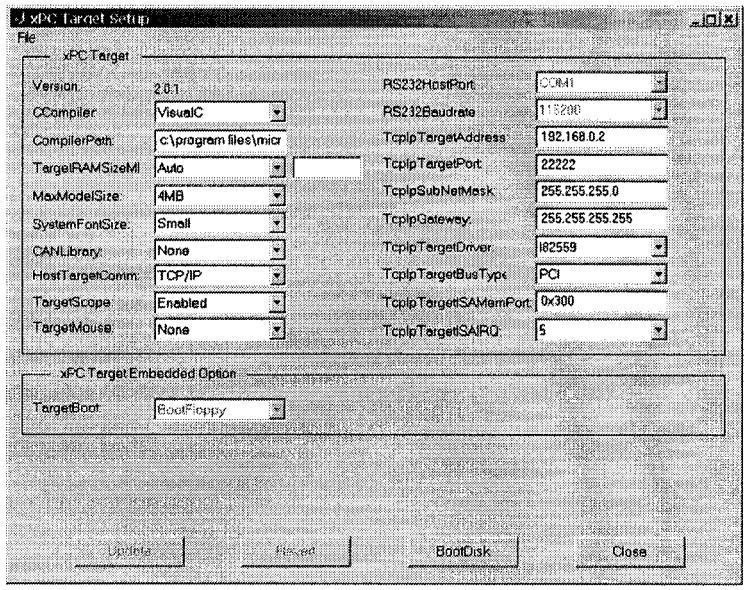


Figure 27 Fenêtre de configuation d'xPC Target

La configuration fournie à xPC Target l'information nécessaire sur les logiciels et le matériel utilisés. Les informations essentielles sont le compilateur C utilisé et son chemin d'accès, la taille de la mémoire RAM de l'ordinateur cible, la taille maximale de

la mémoire réservée pour l'application cible, le type de communication entre les ordinateurs, les adressages TCP/IP des ordinateurs, le pilote de la carte Ethernet de l'ordinateur cible et le type du bus de l'ordinateur cible. Les informations contenues dans la figure 27 sont les informations relatives au projet de recherche. La création de la disquette de démarrage est faite en appuyant sur le bouton *BootDisk* de la fenêtre xPC Target Setup.

Après son démarrage, le noyau affiche un message de bienvenue avec l'information sur la connexion entre l'ordinateur hôte et l'ordinateur cible. Le noyau active ensuite l'application de chargement et se met en attente pour le téléchargement de l'application cible à partir de l'ordinateur hôte. Lors du téléchargement, l'application de chargement reçoit le code de l'application cible, copie les différentes parties de code aux adresses désignées et prépare le démarrage de l'application cible.

Le code d'initialisation de l'application cible réserve le restant de la RAM inutilisée pour le noyau et la pile. La mémoire disponible est la RAM totale installée dans l'ordinateur cible moins les 4 MB réservés pour l'application cible. Dans le projet, la mémoire disponible pour le noyau et la pile est donc de 508 MB. Normalement, la plus grande partie de la pile est utilisée pour faire la sauvegarde des signaux durant la simulation.

L'application cible est capable d'exécuter des tâches en temps réel à très haute vitesse. Le temps d'exécution de tâche varie selon la grosseur et la complexité du modèle, ainsi que des composantes de l'ordinateur cible.

Le noyau temps réel d'xPC Target sauvegarde les données des signaux de l'application cible dans la RAM de l'ordinateur cible. L'acquisition des données est faite durant la simulation et la visualisation des signaux est faite lorsque la simulation est terminée. Les données sont recueillies pendant l'exécution des tâches et sont associées à un temps

précis. Lorsque la simulation est terminée, le téléchargement des données de l'ordinateur cible vers l'ordinateur hôte est fait pour ensuite visualiser les signaux.

La communication entre l'ordinateur hôte et l'ordinateur cible est faite à partir d'une connexion directe de type réseau via un câble Ethernet croisé. Les deux ordinateurs sont connectés au réseau avec des cartes Ethernet utilisant un protocole de communication TCP/IP. Puisque seulement deux cartes Ethernet sont supportées par xPC Target, la carte utilisée dans l'ordinateur cible du projet est la carte Pro/100S d'Intel, qui comporte un seul connecteur RJ45. Il n'y a pas de limitation pour la carte Ethernet de l'ordinateur hôte et la carte 3Com EtherLink XL 10/100 est utilisée. La connexion entre les deux ordinateurs permet de télécharger l'application cible de l'hôte vers la cible et de contrôler l'application cible. La connexion permet aussi de télécharger les données de l'ordinateur cible vers l'ordinateur hôte afin de visualiser les signaux.

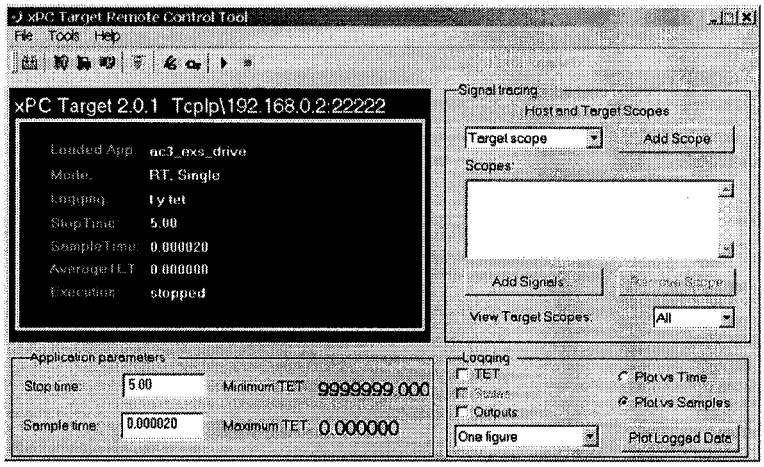


Figure 28 Fenêtre de contrôle d'xPC Target

Le téléchargement et le contrôle de l'application cible se fait de l'ordinateur hôte avec un outil nommé xPC Target Remote Control Tool. La fenêtre de contrôle, qui est présentée par la figure 28, est obtenue en entrant la commande *xpcrctool* à l'invite de MATLAB.

Cet outil permet de lancer la génération, la compilation et le téléchargement dans l'ordinateur cible du code de l'application cible à partir d'un fichier Simulink (.mdl). L'outil permet aussi le démarrage et l'arrêt de l'application cible, la modification de la période d'échantillonnage et du temps d'arrêt, ainsi que la réception de l'information sur les performances de l'application cible et du microprocesseur de l'ordinateur cible.

Le noyau d'xPC Target à deux modes pour l'exécution de la simulation en temps réel, soit le mode par interruption, qui est le mode par défaut, et le mode par polling. Le mode par interruption est le mode qui offre la plus grande flexibilité et qui devrait être utilisé pour exécuter des applications dont la période d'échantillonnage est relativement lente. Le désavantage de ce mode est qu'il présente une latence provenant du traitement des interruptions. Cette latence est d'environ 7 μs pour le microprocesseur utilisé. Par contre, le mode par polling qui est considéré comme un mode plus brute et primitif, ne possède pas cette latence car il ne permet aucune interruption. Ce mode est donc privilégié pour exécuter une application cible dont la période d'échantillonnage est proche des limites du microprocesseur, ce qui est le cas dans le projet de recherche.

xPC Target supporte une vaste gamme de cartes d'E/S [5]. Les cartes utilisées dans le projet ont été mentionnées précédemment et sont deux cartes PCI-MIO-16E-4 de National Instruments. Ces cartes ont été choisies pour leurs multiples fonctionnalités et leurs performances. Les pilotes de ces cartes sont accessibles via les blocs de la librairie d'xPC Target dans Simulink. On insère et connecte les blocs de cette librairie dans le modèle Simulink comme n'importe quel autre bloc standard. Plusieurs types de blocs sont disponibles, soit des E/S analogiques, des E/S numériques, des supports séries RS-232/422/485, des compteurs, des encodeurs incrémentaux, etc. Dans cette partie du projet, les blocs utilisés sont les E/S analogiques.

3.3.1 Développement et configuration des modèles

Comme mentionné précédemment, une application cible est créée à partir d'un modèle Simulink. Dans le projet, les modèles consistent en l'électronique de puissance et les machines des modèles AC3 et AC6 à valeur moyenne. L'électronique de puissance est constituée dans les deux cas d'une source de tension CC et d'un onduleur triphasé. Les figures 32 à 35 présentent les schémas Simulink de l'électronique de puissance et des machines des modèles AC3 et AC6 à valeur moyenne.

Ces modèles comportent les blocs de pilotes pour les E/S analogiques des cartes PCI-MIO-16E-4. Ces blocs sont disponibles dans la librairie d'E/S de xPC Target de Simulink. La figure 29 présente les volets sous lesquels sont disponibles les E/S analogiques des cartes PCI-MIO-16E-4.

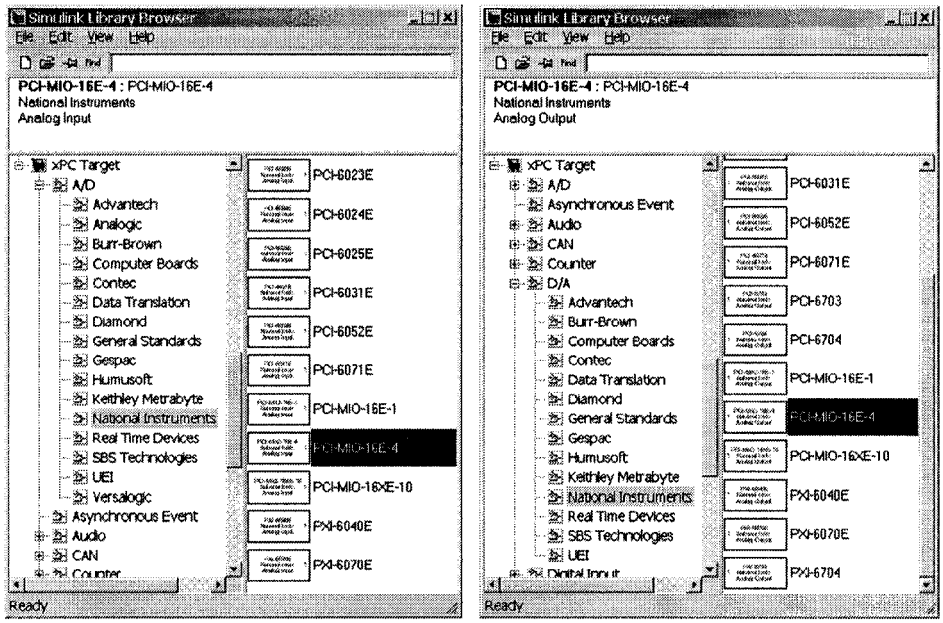


Figure 29 Blocs d'E/S analogiques de la carte PCI-MIO-16E-4

Le bloc d'entrée analogique doit être configuré avec les informations concernant le nombre d'entrées utilisées, les plages de tension désirées, le type de couplage des entrées, la période d'échantillonnage et l'emplacement de la carte sur le bus PCI. La figure 30 présente le bloc de configuration d'entrée analogique. Dans ce cas, il y a quatre entrées utilisées et leurs plages de tension sont de -10 V à 10 V. Le vecteur de couplage des entrées permet de déterminer si les entrées sont simples ou différentielles. Dans le projet, toutes les entrées analogiques sont simples. La période d'échantillonnage est de 20 µs et l'emplacement de la carte est dans la neuvième fente du bus PCI de l'ordinateur cible. Les deux cartes PCI-MIO-16E-4 sont situées sur le bus PCI de l'ordinateur cible, la première carte est dans la neuvième fente et la deuxième carte est dans la dixième fente. L'aide de MATLAB présente en détail toutes les informations sur la configuration du bloc d'entrée analogique de la carte PCI-MIO-16E-4.

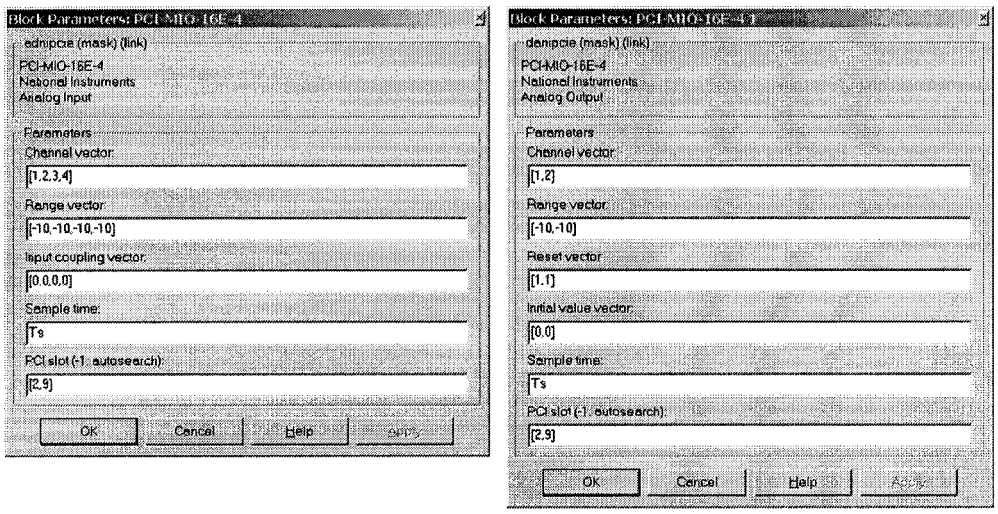


Figure 30 Fenêtres de configuration des blocs d'E/S analogiques

Pour les blocs de sortie analogique, les informations nécessaires concernent les numéros des sorties utilisés, les plages de tensions désirées, les valeurs initiales, le comportement des valeurs finales, la période d'échantillonnage et l'emplacements de la carte. La figure

30 présente le bloc de configuration de sortie analogique. Il y a deux sorties utilisées, les plages de tension sont de -10 V à 10 V, la période d'échantillonnage est de 20 µs et la carte est située dans la neuvième fente du bus PCI de l'ordinateur cible. Les valeurs initiales des sorties sont de 0 V et les valeurs finales sont mises à 0 V à la terminaison du modèle. L'aide de MATLAB présente en détail toutes les informations sur la configuration du bloc de sortie analogique de la carte PCI-MIO-16E-4.

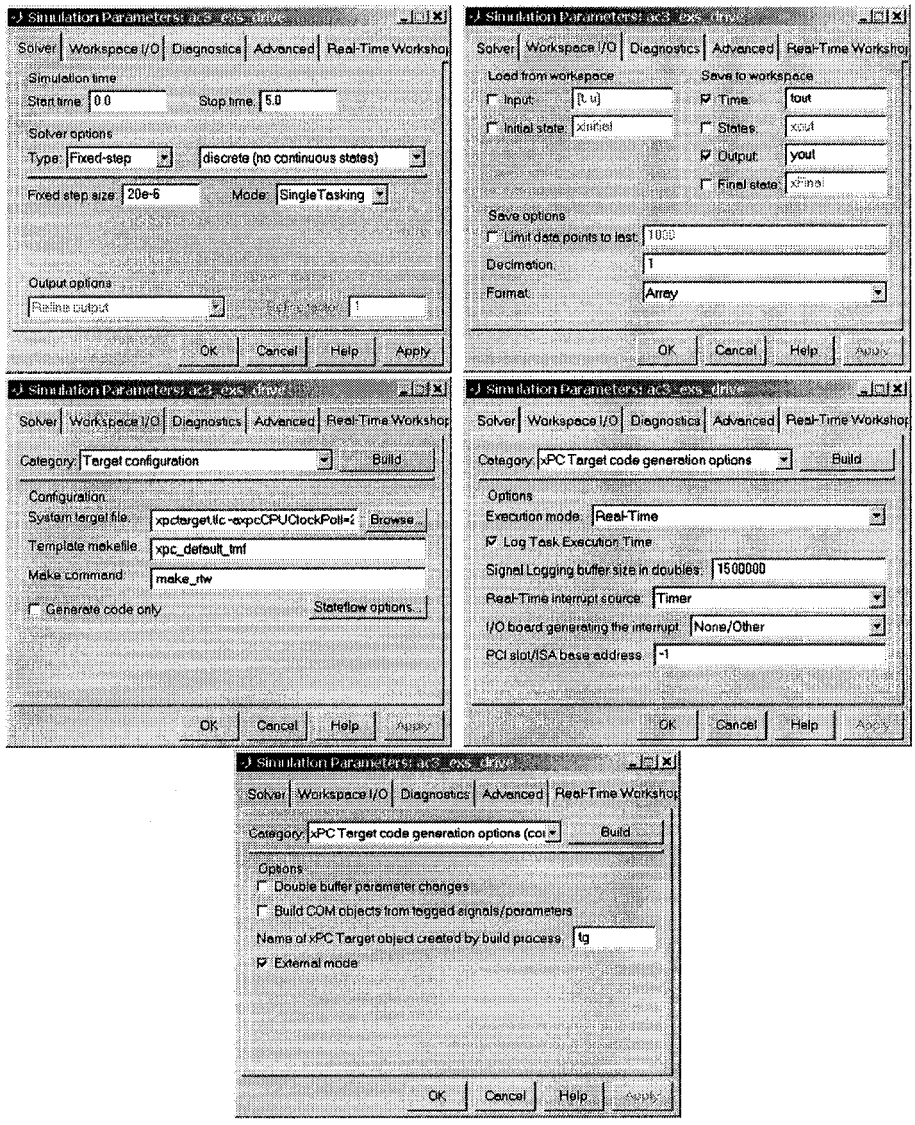


Figure 31 Fenêtres de configuration des paramètres d'xPC Target

Un modèle Simulink doit être configuré pour fonctionner avec xPC Target. La fenêtre de configuration des paramètres présentée à la figure 31 est obtenue en sélectionnant *Simulation parameters* dans le menu *Simulation* de la fenêtre Simulink du modèle. Dans l'onglet *Solver*, il faut inscrire les temps de départ et d'arrêt dans les cases *Start time* et *Stop time*. Dans ce cas la simulation débute à 0 s et se termine à 5 s. L'algorithme de calcul à pas fixe de type discret est choisi en sélectionnant *Fixed-step* dans la case *Type* et *Discrete* dans la case *Solver*. Par la suite, il faut inscrire la période d'échantillonnage de 20 µs dans la case *Fixed-step size*. Finalement, il faut spécifier que le modèle exécute seulement une tâche en sélectionnant *Single Tasking* dans la case *Mode*.

Dans l'onglet *Workspace I/O*, il faut sélectionner *Time* et *Output* dans la section *Save to workspace* pour sauvegarder les signaux de sortie et le temps lors de la simulation du modèle. De plus, dans la section *Save options*, il faut s'assurer que la case *Limit data points to last* ne soit pas sélectionnée afin de ne pas limiter le nombre de points acquisitionnés lors de la simulation. Il faut aussi sélectionner *Array* dans la case *Format* afin d'obtenir des matrices de données comprenant les signaux de sorties et le temps.

Dans l'onglet *Real-Time Workshop*, en appuyant sur *Browse*, on sélectionne *xpctarget.tlc* pour générer le code C du modèle pour xPC Target. Par défaut, le mode d'exécution de la simulation en temps réel est le mode par interruption. Pour sélectionner le mode par polling, il faut inscrire *-axpcCPUClockPoll=2000* à la suite de *xpctarget.tlc* dans la case *System target file*, où le nombre 2 000 correspond à la vitesse du microprocesseur de l'ordinateur cible en mégahertz (MHz).

La configuration du modèle se poursuit en sélectionnant *xPC Target code generation options* dans le menu déroulant *Category*. Il faut spécifier que l'exécution est réalisée en temps réel en sélectionnant *Real-Time* dans la case *Execution mode*. Il faut aussi spécifier le nombre maximum de points pouvant être acquisitionnés dans la case *Signal Logging buffer size in doubles*. Le nombre choisi est de 1 500 000 pour le cas demandant

le plus de points, soit dans le cas du modèle AC3 où il y a six signaux échantillonnés pendant 5 s avec une période d'échantillonnage de 20 µs.

3.3.1.1 Modèle AC3

Comme présenté dans la section 2.2.1, le modèle AC3 comprend une source CC idéale, un onduleur triphasé et une machine asynchrone. Le simulateur nécessite deux entrées analogiques d'une carte PCI-MIO-16E-4 pour recevoir les références des courants statoriques i_{as}* et i_{bs}* de la commande. Les deux entrées analogiques utilisées sont les entrées 1 et 2 de la première carte située dans la neuvième fente du bus PCI de l'ordinateur cible. Le simulateur nécessite aussi les quatre sorties analogiques des cartes pour transmettre la vitesse mécanique du moteur (ω_m) et les courants statoriques triphasés (i_{abcs}) à la commande. La figure 32 présente le schéma Simulink de l'électronique de puissance et de la machine du modèle AC3 à valeur moyenne.

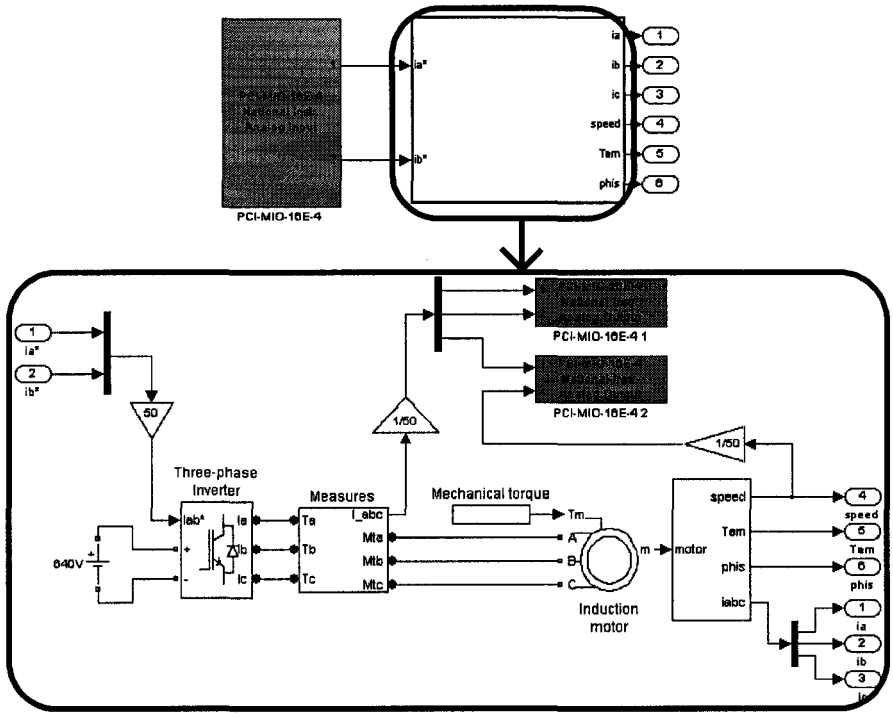


Figure 32 Schéma Simulink de l'électronique de puissance et de la machine

La configuration des blocs d'E/S analogiques du modèle est la même que celle présentée à la figure 30, à l'exception du nombre d'entrées analogiques qui est de deux. Des gains sont ajoutés aux entrées et aux sorties pour faire correspondre les plages de valeur tout en étant cohérent avec l'ajustement des plages de la commande. Comme mentionné dans la section 3.2.1.1, la plage maximale de fonctionnement observée pour les courants est de -500 A à 500 A et la plage maximale pour la vitesse est de 0 RPM à 500 RPM. Les plages de fonctionnement des E/S analogiques des cartes PCI-MIO-16E-4 sont de -10 V à 10 V. Pour faire correspondre la plage des entrées, les références de courants sont multipliées par un gain de 50 A/V, puis pour faire correspondre la plage des sorties, les courants et la vitesse mécanique sont divisés par des gains de 50 A/V et de 50 RPM/V.

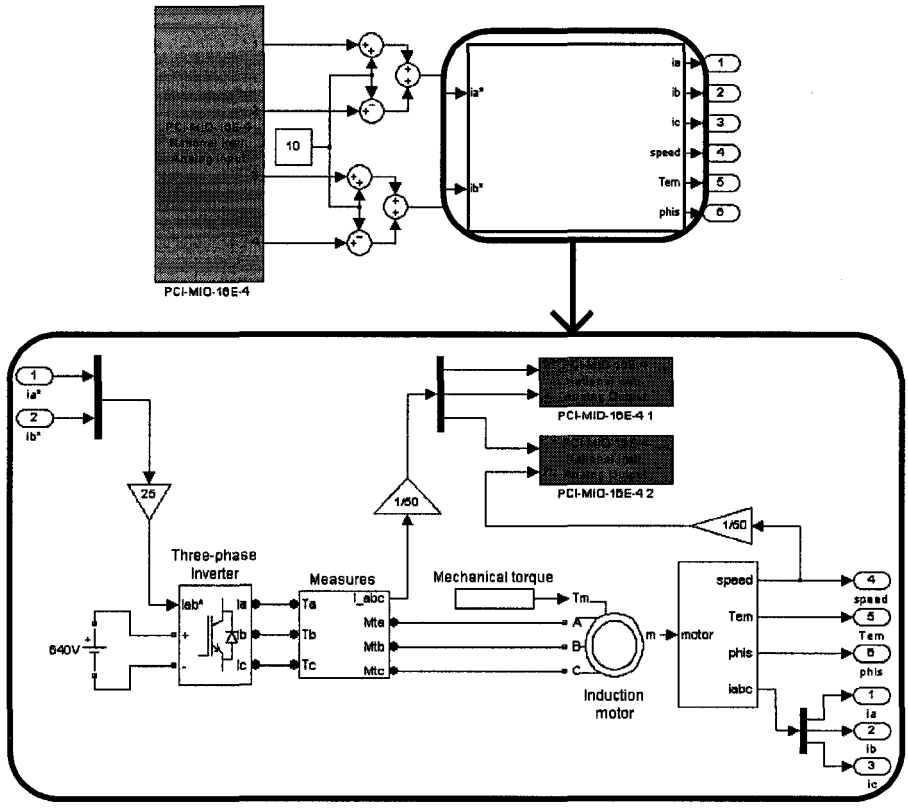


Figure 33 Schéma Simulink amélioré de l'électronique de puissance et de la machine

Les résultats obtenus pour la simulation en temps réel du modèle AC3 à valeur moyenne avec commande externe sont présentés dans la section 4.4. Comme spécifié dans la section 3.2.1.1, un modèle amélioré de la commande à été réalisé car les premiers résultats n'étaient pas satisfaisants. Un modèle amélioré de l'électronique de puissance et de la machine a aussi été réalisé pour être compatible avec la commande. Le schéma Simulink du modèle amélioré est présenté par la figure 33. Ce modèle comporte une modification par rapport au premier modèle.

La modification est le doublage de la résolution des signaux d'entrées. Le schéma de principe de la reconstitution des signaux provenant de la commande est présenté par la figure 24. Le modèle reçoit les références des courants statoriques i_{as}* et i_{bs}* sur quatre entrées et doit reconstituer ces signaux. Les deux parties de la référence i_{as}* sont reçues par les entrées 1 et 2, tandis que les deux parties de la référence i_{bs}* par les entrées 3 et 4.

Les deux signaux a reconstituer possèdent une plage de -500 A à 500 A. Les signaux reçus par les entrées possèdent une plage de -10 V à 10 V. Les entrées impaires reçoivent les signaux qui correspondent aux plages de 0 A à 500 A et les entrées paires reçoivent les signaux qui correspondent aux plages de -500 A à 0 A. Un biais de 10 V est additionné aux signaux des entrées impaires et soustrait aux signaux des entrées paires pour obtenir des plages respectives de 0 V à 20 V et de -20 V à 0 V. Les signaux 1 et 2 sont ensuite additionner pour obtenir un seul signal possédant une plage de -20 V à 20 V. La même opération est réalisée avec les signaux 3 et 4. Finalement, les deux signaux résultants sont multipliés par un gain de 25 A/V pour obtenir des plages de -500 A à 500 A. Les deux références de courants sont alors reconstituées. La configuration de blocs d'E/S analogiques est la même que celle présentée par la figure 30. Cependant, il faut ajouter un second bloc pour les sorties de la deuxième carte située dans la dixième fente du bus PCI de l'ordinateur cible. La configuration de ce bloc est la même que celle de la figure 30 à l'exception de l'emplacement de la carte. L'emplacement est modifié en inscrivant [2,10] dans la case *PCI Slot*.

3.3.1.2 Modèle AC6

Comme présenté dans la section 2.2.2, le modèle AC6 comprend une source CC idéale, un onduleur triphasé et une machine synchrone à aimants permanents. Le simulateur nécessite deux entrées analogiques pour recevoir les références des courants statoriques i_a* et i_b* de la commande. Les deux entrées analogiques 1 et 2 de la première carte PCI-MIO-16E-4 située dans la neuvième fente du bus PCI de l'ordinateur cible sont utilisées. Le simulateur nécessite aussi deux sorties analogiques pour transmettre la vitesse mécanique du moteur (ω_m) et la position angulaire du rotor (θ_r) à la commande. Les sorties 1 et 2 de la première carte sont utilisées. La figure 34 présente le schéma Simulink de l'électronique de puissance et de la machine du modèle AC6 à valeur moyenne.

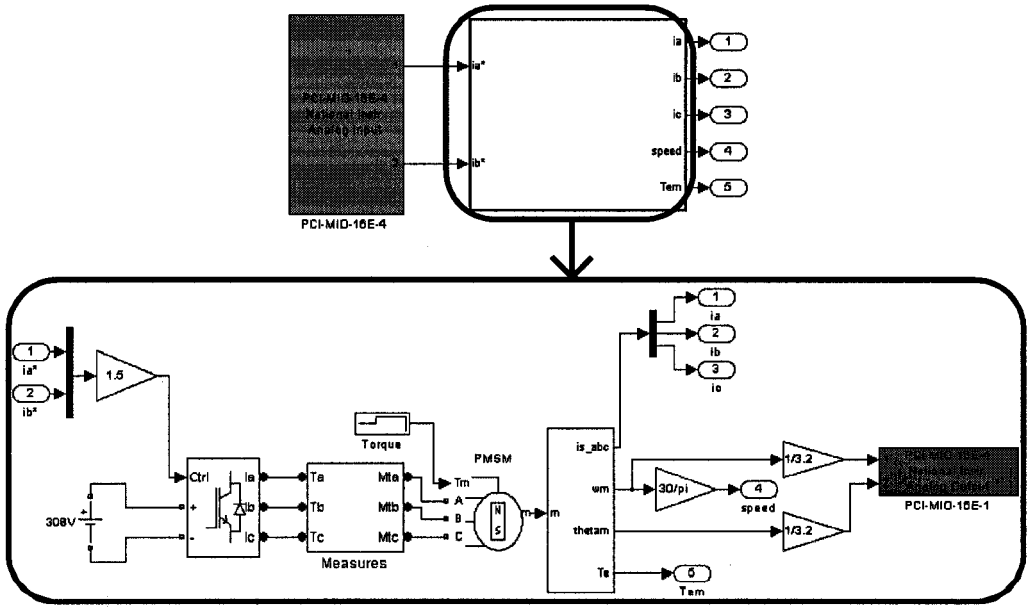


Figure 34 Schéma Simulink de l'électronique de puissance et de la machine

La configuration des blocs d'E/S analogiques du modèle est la même que celle présentée à la figure 30, mais pour seulement deux entrées analogiques. Des gains sont ajoutés aux

E/S du modèle pour faire correspondre les plages de valeur de façon cohérente avec l'ajustement des plages de la commande. Comme mentionné dans la section 3.2.1.1, la plage maximale de fonctionnement observée pour les courants est de -15 A à 15 A, la plage maximale pour la vitesse mécanique est de -1 rad/s à 32 rad/s et la plage maximale pour la position angulaire est de 0 rad à 32 rad. Les plages de fonctionnement des E/S analogiques des cartes PCI-MIO-16E-4 sont de -10 V à 10 V. Pour faire correspondre la plage des entrées, les références des courants sont multipliées par un gain de 1,5 A/V, tandis que pour faire correspondre la plage des sorties, la vitesse mécanique et la position angulaire sont divisés par des gains de 3,2 (rad/s)/V et de 3,2 rad/V.

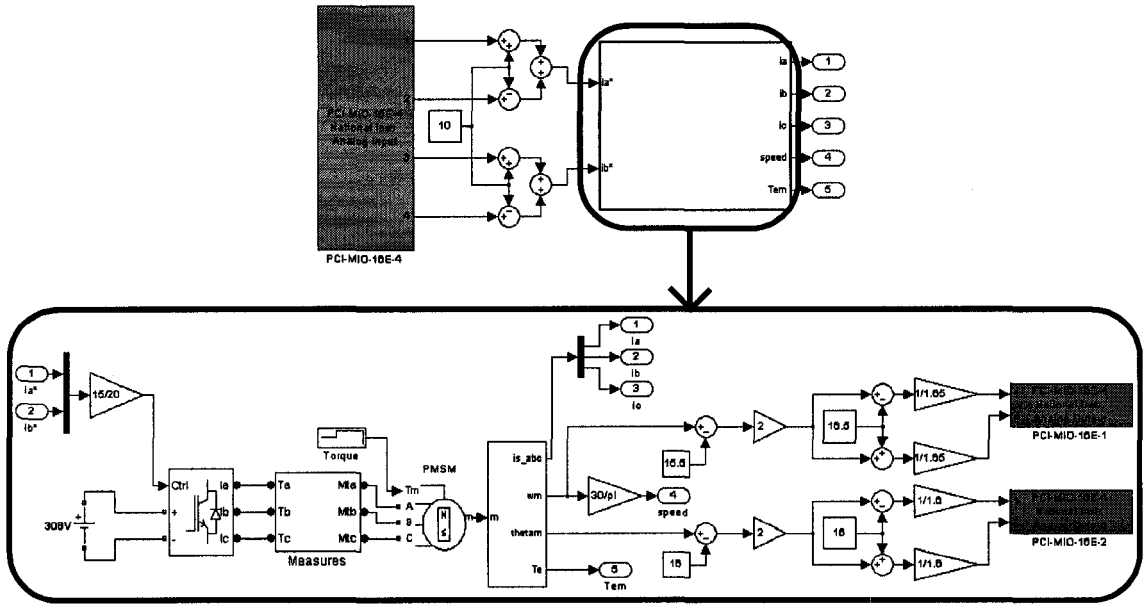


Figure 35 Schéma Simulink amélioré de l'électronique de puissance et de la machine

Les résultats obtenus pour la simulation en temps réel du modèle AC6 à valeur moyenne avec commande externe sont présentés dans la section 4.5. Comme spécifié dans la section 3.3.1.2, un modèle amélioré de la commande à été réalisé car les premiers résultats n'étaient pas satisfaisants. Un modèle amélioré de l'électronique de puissance et de la machine a aussi été réalisé pour être compatible avec la commande. Le schéma

Simulink du modèle amélioré est présenté par la figure 35. Ce modèle comporte deux modifications par rapport au premier modèle.

La première modification est le doublage de la résolution des signaux d'entrées. Le schéma de principe de la reconstitution des signaux provenant de la commande est présenté par la figure 24. Le modèle doit reconstituer les deux références des courants statoriques reçues sur quatre entrées. Les deux parties de la référence i_a* sont reçues par les entrées 1 et 2, tandis que les deux parties de la référence i_b* sont reçues par les entrées 3 et 4.

Les deux signaux à reconstituer possèdent une plage de -15 A à 15 A. Les signaux reçus par les entrées possèdent une plage de -10 V à 10 V. Les entrées impaires reçoivent les signaux qui correspondent aux plages de 0 A à 15 A et les entrées paires reçoivent les signaux qui correspondent aux plages de -15 A à 0 A. Un biais de 10 V est additionné aux signaux des entrées impaires et soustrait aux signaux des entrées paires pour obtenir des plages respectives de 0 V à 20 V et de -20 V à 0 V. Les signaux 1 et 2 sont ensuite additionner pour obtenir un seul signal possédant une plage de -20 V à 20 V. La même opération est réalisée avec les signaux 3 et 4. Finalement, les deux signaux résultants sont multipliés par un gain de 15/20 A/V pour obtenir des plages de -15 A à 15 A. Les deux références de courants sont alors reconstituées.

La deuxième modification est le doublage de la résolution des signaux de sortie. Le principe utilisé est le même que dans la section 3.2.1.1 comme présenté par la figure 24. Les signaux de sortie sont la vitesse mécanique et la position angulaire du rotor. Tout d'abord, un biais est soustrait aux signaux puisqu'ils ne sont pas symétriques par rapport à l'abscisse. Le biais est de 15,5 rad/s pour la vitesse et de 16 rad pour la position angulaire. Les plages de valeurs résultantes sont de -16,5 rad/s à 16,5 rad/s et de -16 rad à 16 rad. Les signaux sont ensuite multipliés par un gain de 2 pour doubler leurs amplitudes et ainsi obtenir des plages maximales de -33 rad/s à 33 rad/s et de -32 rad à 32

rad. Un biais de 16,5 rad/s est soustrait et additionner au signal de la vitesse pour obtenir deux signaux avec des plages respectives de -49,5 rad/s à 16,5 rad/s et de -16,5 rad/s à 49,5 rad/s. Dans le cas de la position angulaire, la même opération est effectuée avec un biais de 16 rad pour obtenir deux signaux avec des plages de -48 rad à 16 rad et de -16 rad à 48 rad. Les signaux de vitesse sont divisés par un gain de 1,65 (rad/s)/V, ce qui produit des plages de -30 V à 10 V et de -10 V à 30 V. Les mêmes plages sont obtenues pour les signaux de la position angulaire avec la division par un gain de 1,6 rad/V. Puisque les sorties analogiques sont saturées entre -10 V et 10 V, les plages de -30 V à -10V et les plages 10 V à 30 V sont tronquées. La partie supérieure de la vitesse est alors transmise par un signal et la partie inférieure par un autre signal. Le même résultat est obtenu pour la position angulaire. Les deux signaux de la vitesse sont transmis à la commande par les deux sorties analogiques de la première carte PCI-MIO-16E-4. Pour ce qui est des deux signaux de la position angulaire, ils sont transmis par les deux sorties analogiques de la deuxième carte PCI-MIO-16E-4. Les signaux sont par la suite reconstitués dans la commande et les détails sont présentés dans la section 3.2.1.2.

La configuration de blocs d'E/S analogiques est la même que celle présentée par la figure 30. Cependant, il faut ajouter un second bloc pour les sorties de la deuxième carte située dans la dixième fente du bus PCI de l'ordinateur cible. La configuration de ce bloc est la même que celle de la figure 30 à l'exception de l'emplacement de la carte. L'emplacement est modifié en inscrivant [2,10] dans la case *PCI Slot*.

3.4 Conclusion

Ce chapitre a présenté la démarche utilisée pour réaliser la simulation en temps réel avec commande externe des modèles d'entraînement électrique à valeur moyenne présentés dans le chapitre 2. Tous les outils matériels et logiciels utilisés ainsi que les étapes nécessaires à la réalisation de cette partie du projet ont été présentés.

Le prochain chapitre présente et valide les résultats obtenus pour la simulation en temps réel avec commande externe des modèles d'entraînement électrique à valeur moyenne. La validation de la simulation est réalisée par la comparaison des résultats obtenus avec des résultats de simulation en temps différé.

CHAPITRE 4

COMPARAISON ET ANALYSE DES RÉSULTATS

4.1 Introduction

Ce chapitre présente les résultats de simulation en temps différé et les résultats de simulation en temps réel avec commande externe des modèles à valeur moyenne AC3 et AC6. Les sections 4.2 et 4.3 décrivent les simulations réalisées et présentent les résultats de simulation en temps différé. Les sections 4.4 et 4.5 présentent les résultats de simulation en temps réel avec commande externe pour les premières versions et les versions améliorées des modèles. Les résultats des deux parties sont ensuite comparés et analysés pour faire la validation de la simulation en temps réel avec commande externe dans les sections 4.6 et 4.7.

4.2 Résultats de simulation en temps différé du modèle AC3

La simulation réalisée en temps différé et en temps réel avec commande externe pour le modèle AC3 à valeur moyenne est la même. La référence de vitesse est nulle du temps 0 s à 1 s, elle est de 500 RPM du temps 1 s à 2 s, puis elle est nulle du temps 2 s à 5 s. Le couple mécanique est nul du temps 0 s à 1,5 s, il est de 792 N · m du temps 1,5 s à 2,5 s et il est de -792 N · m du temps 2,5 s à 5 s.

La figure 36 présente les résultats de simulation en temps différé. Les résultats présentés sont le courant statorique (i_{as}), la vitesse mécanique du rotor (N), le couple électromagnétique (T_e) et le flux statorique (φ_s). Les mêmes résultats pour la simulation en temps réel avec commande externe sont présentés dans la section 4.4.

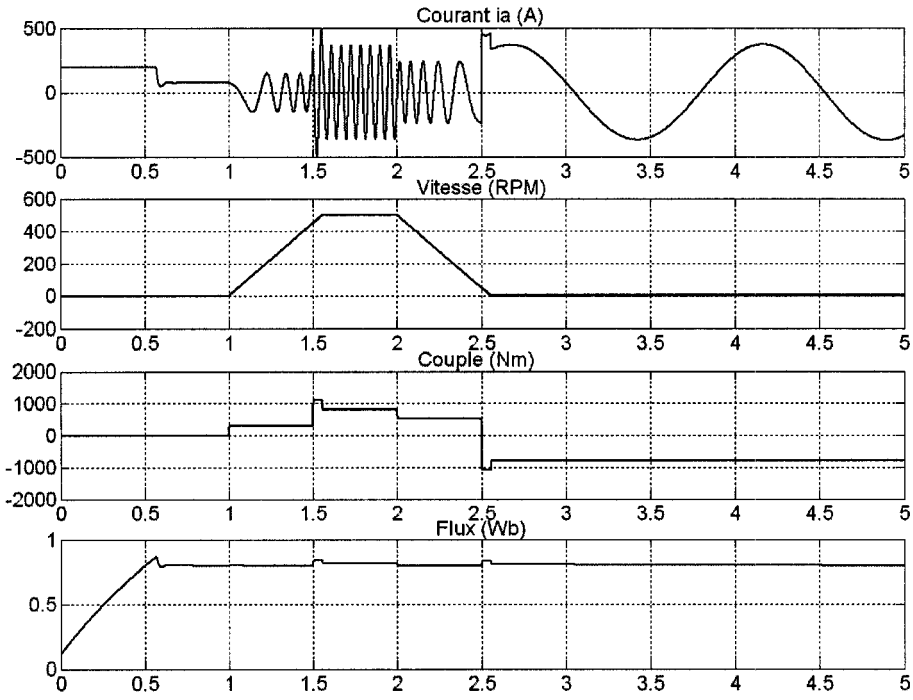


Figure 36 Résultats de simulation en temps différé du modèle AC3

4.3 Résultats de simulation en temps différé du modèle AC6

La simulation réalisée en temps différé et en temps réel avec commande externe pour le modèle AC6 à valeur moyenne est la même. La référence de vitesse est de 300 RPM du temps 0 s à 1 s, puis elle est nulle du temps 1 s à 3 s. Le couple mécanique est nul du temps 0 s à 0,5 s, il est de 11 N · m du temps 0,5 s à 1,5 s et il est de -11 N · m du temps 1,5 s à 3 s.

La figure 37 présente les résultats de simulation en temps différé. Les résultats présentés sont le courant statorique (i_a), la vitesse mécanique du rotor (N) et le couple électromagnétique (T_e). Les mêmes résultats pour la simulation en temps réel avec commande externe sont présentés dans la section 4.5. La figure 38 présente la vitesse mécanique du moteur (ω_m) et la position angulaire mécanique du rotor (θ_r). Ces résultats

sont utilisés dans les sections 3.2.1.2 et 3.3.1.2 pour déterminer les plages de fonctionnement des signaux et les gains pour les E/S des cartes PCI-MIO-16E-4 et DS1104.

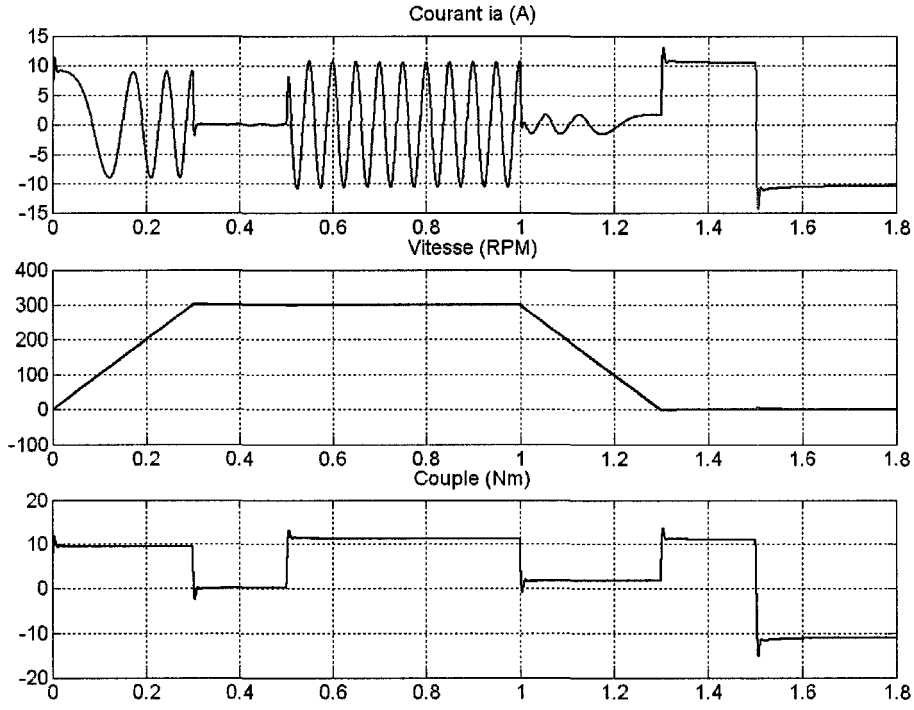


Figure 37 Résultats de simulation en temps différé du modèle AC6

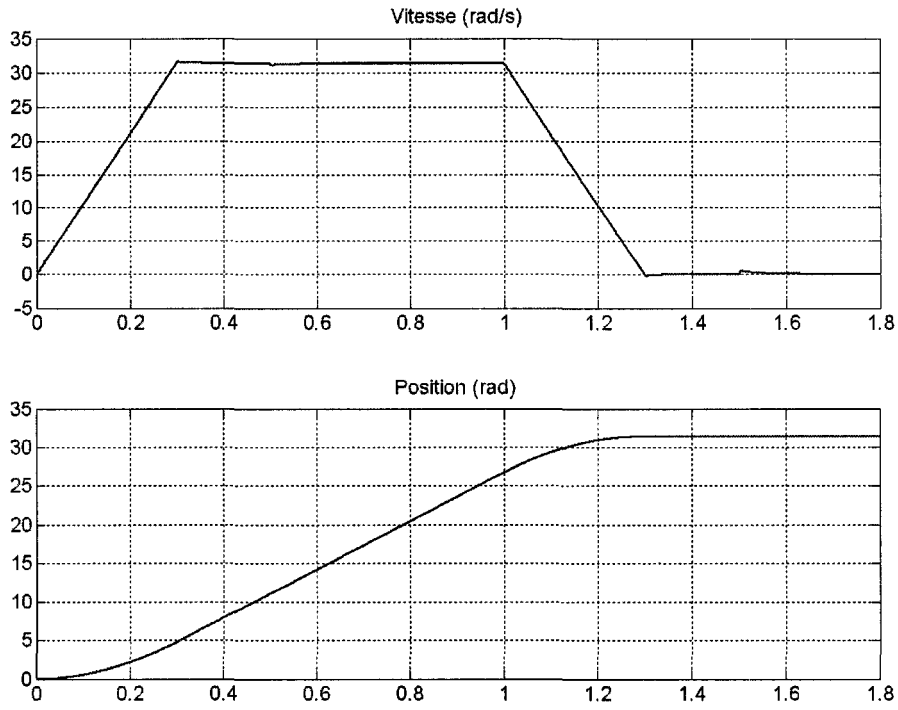


Figure 38 Vitesse et position du modèle AC6

4.4 Résultats de simulation en temps réel du modèle AC3

Cette section présente les résultats pour la simulation en temps réel avec commande externe du modèle AC3 à valeur moyenne. La figure 39 présente les résultats pour la première version du modèle et la figure 43 présente les résultats pour la version améliorée. Pour observer l'amélioration entre les deux versions, les figures 40 et 44 présentent un agrandissement d'une section des signaux de couple électromagnétique qui montrent le niveau de bruit. Les figures 41 et 45 présentent les spectres de fréquence de ces sections de signaux, ce qui démontre l'amélioration obtenue. Finalement, les figures 42 et 46 présentent le temps d'exécution des tâches en temps réel pour les deux versions.

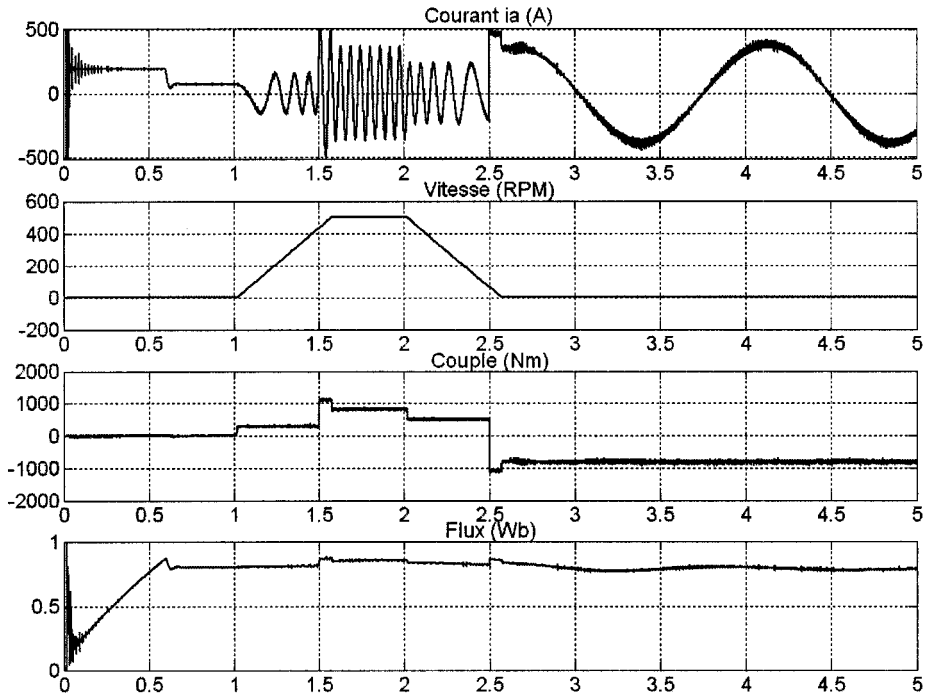


Figure 39 Résultats de simulation en temps réel du modèle AC3

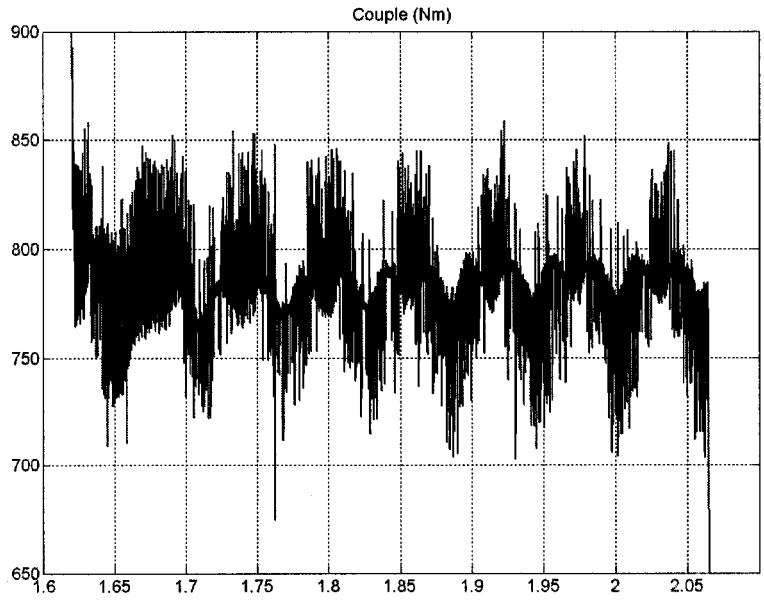


Figure 40 Agrandissement du couple

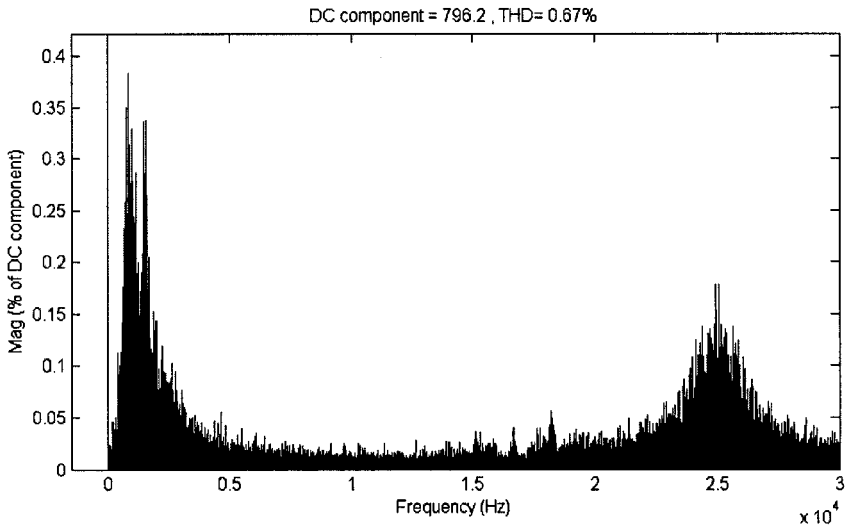


Figure 41 Spectre de fréquence

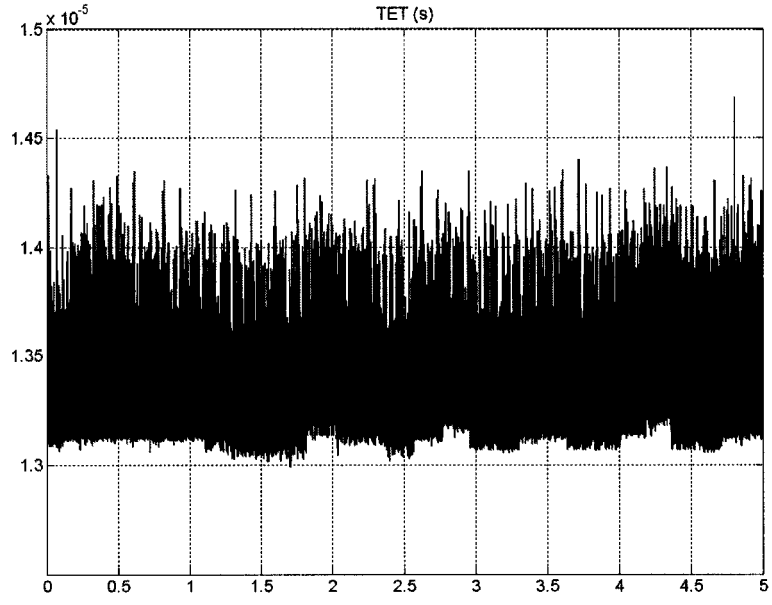


Figure 42 Temps d'exécution des tâches

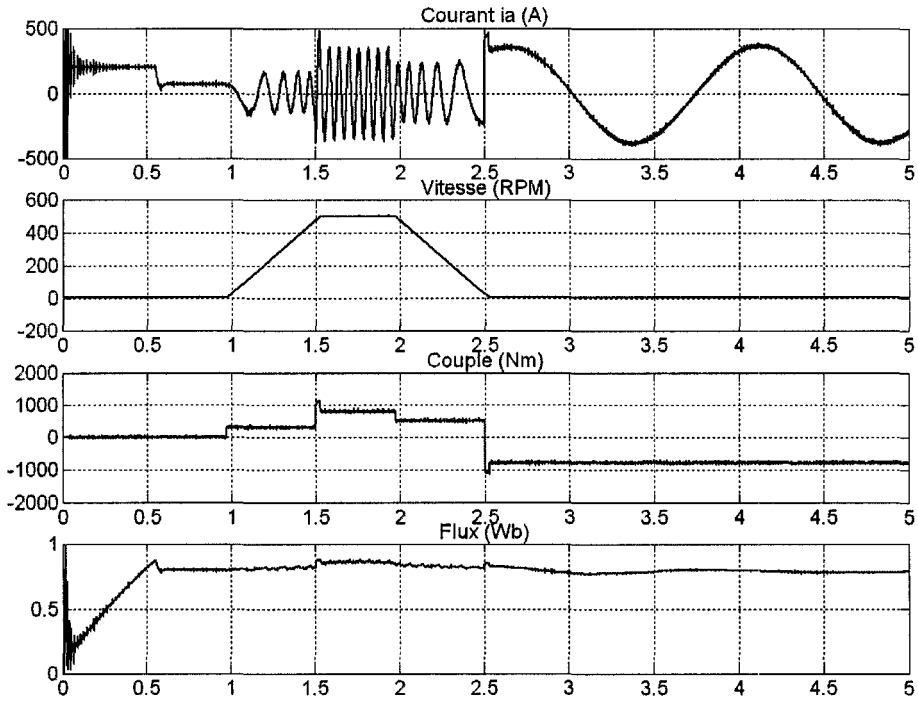


Figure 43 Résultats de simulation en temps réel du modèle AC3 amélioré

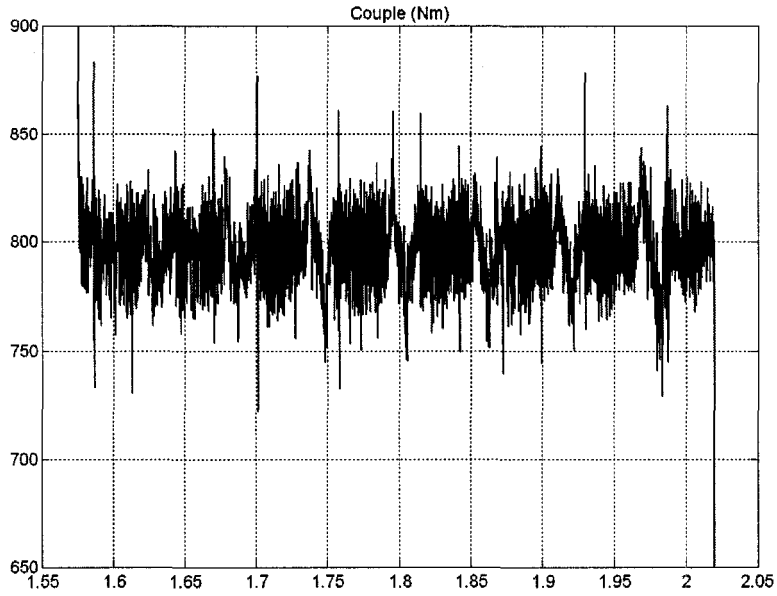


Figure 44 Agrandissement du couple

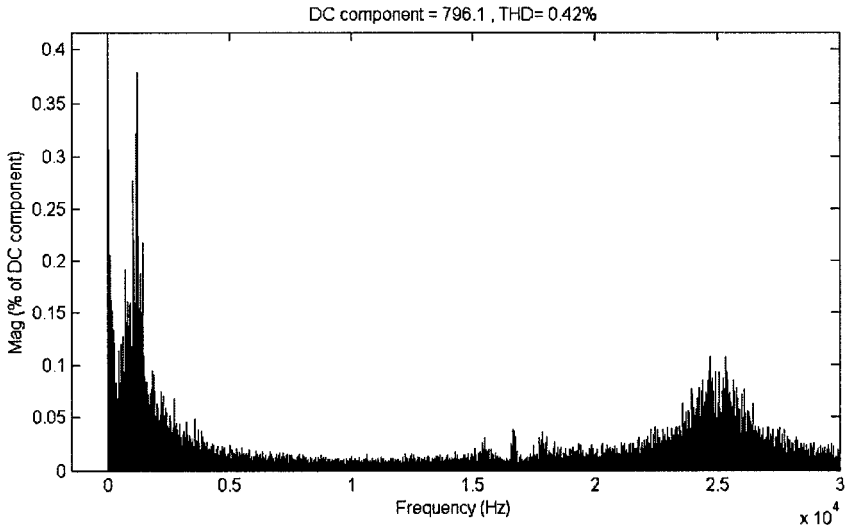


Figure 45 Spectre de fréquence

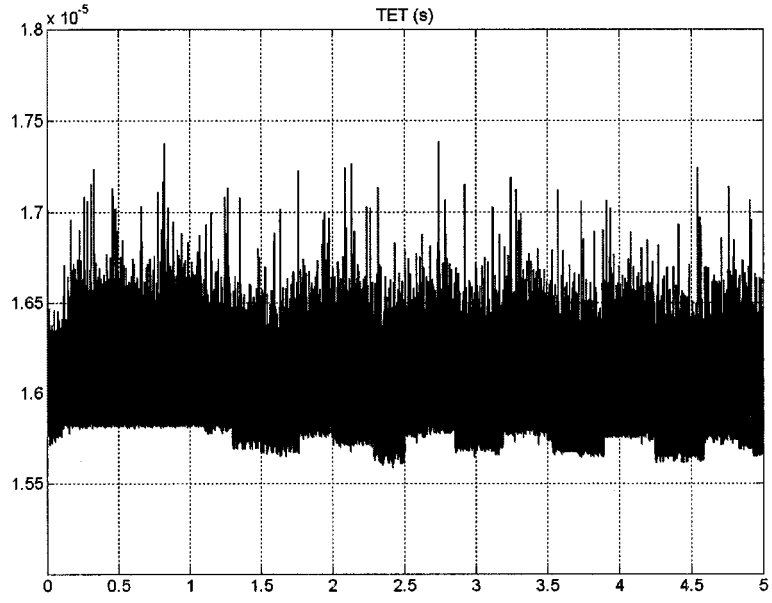


Figure 46 Temps d'exécution des tâches

4.5 Résultats de simulation en temps réel du modèle AC6

Cette section présente les résultats pour la simulation en temps réel avec commande externe du modèle AC6 à valeur moyenne. La figure 47 présente les résultats pour la première version du modèle et la figure 51 présente les résultats pour la version améliorée. Pour observer l'amélioration entre les deux versions, les figures 48 et 52 présentent un agrandissement d'une section des signaux de couple électromagnétique qui montrent le niveau de bruit. Les figures 49 et 53 présentent les spectres de fréquence de ces sections de signaux, ce qui démontre l'amélioration obtenue. Finalement, les figures 50 et 54 présentent le temps d'exécution des tâches en temps réel pour les deux versions.

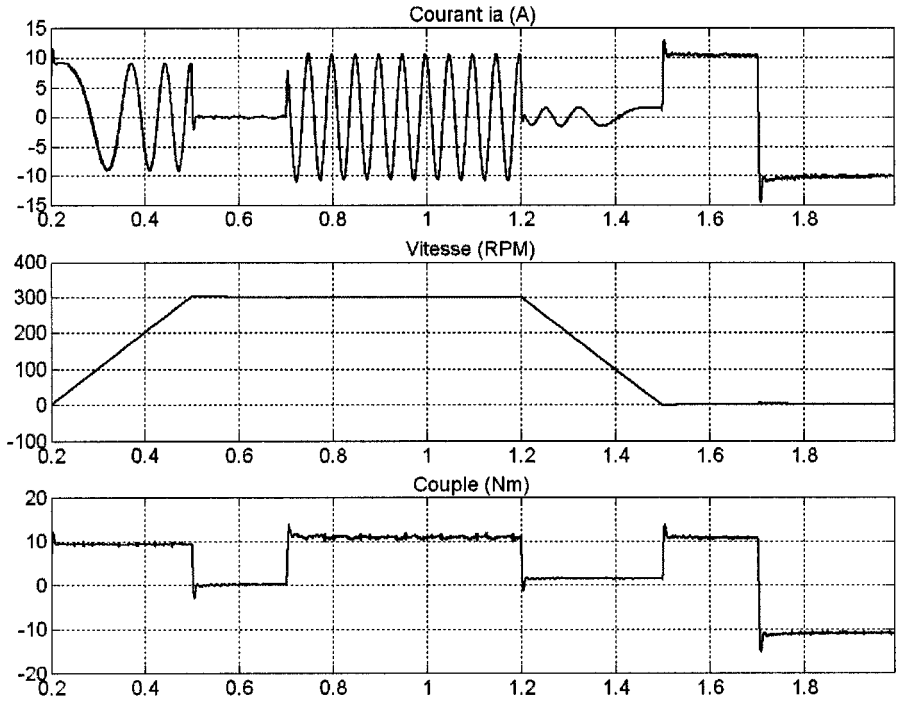


Figure 47 Résultats de simulation en temps réel du modèle AC6

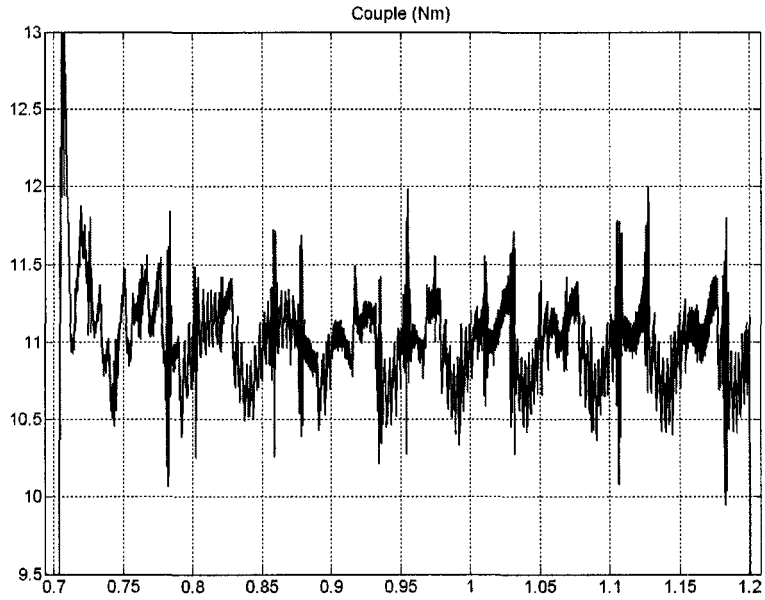


Figure 48 Agrandissement du couple

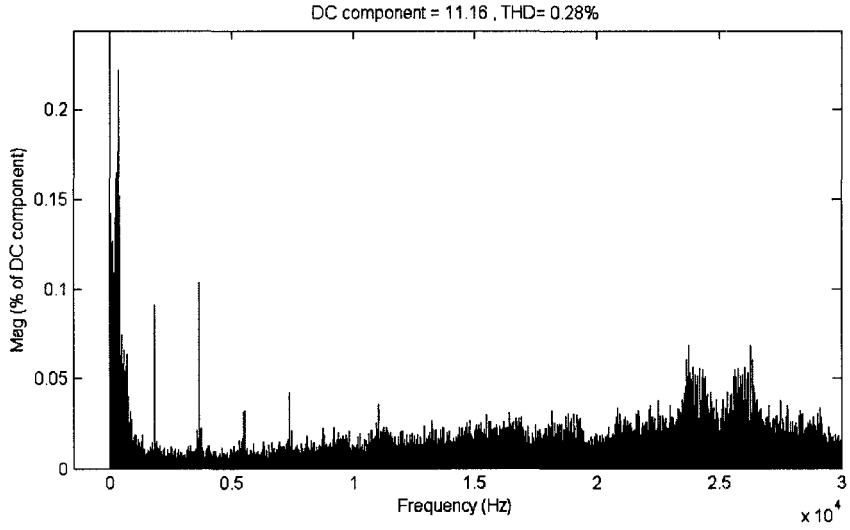


Figure 49 Spectre de fréquence

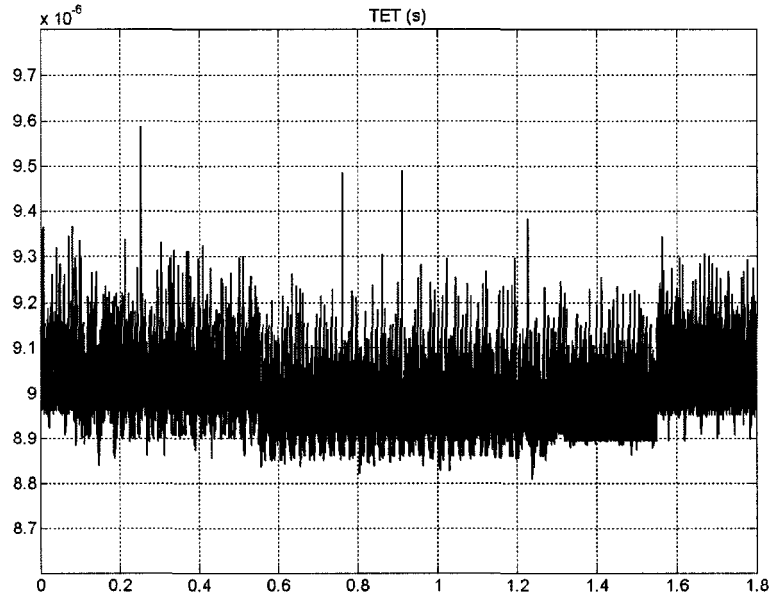


Figure 50 Temps d'exécution des tâches

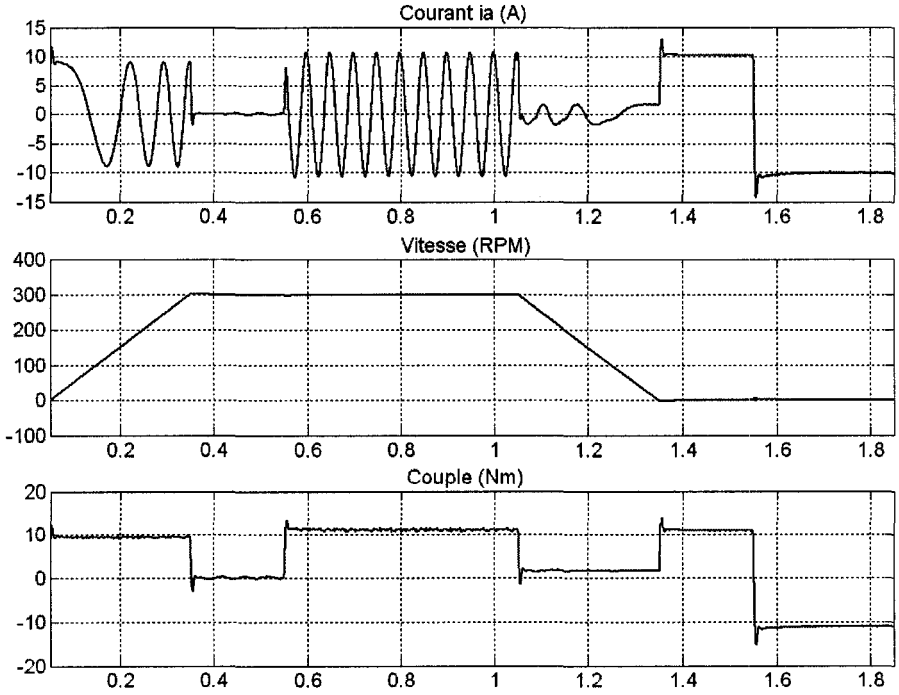


Figure 51 Résultats de simulation en temps réel du modèle AC6 amélioré

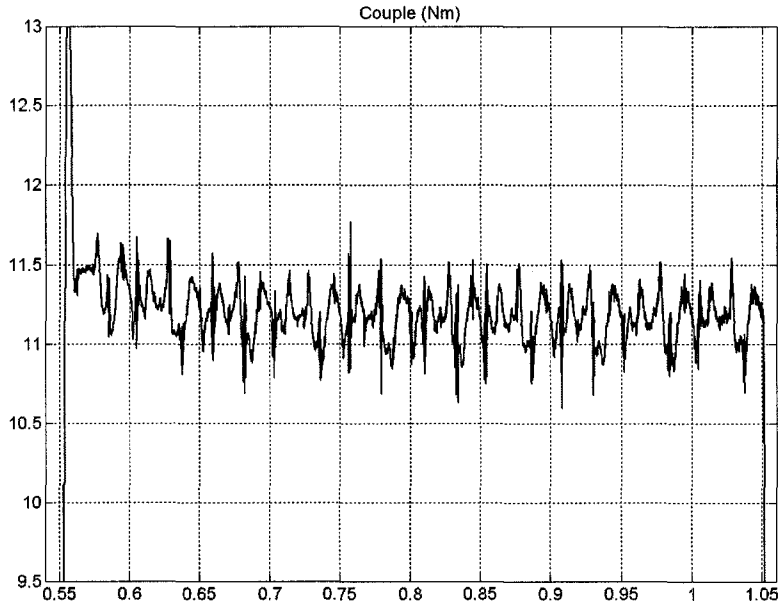


Figure 52 Agrandissement du couple

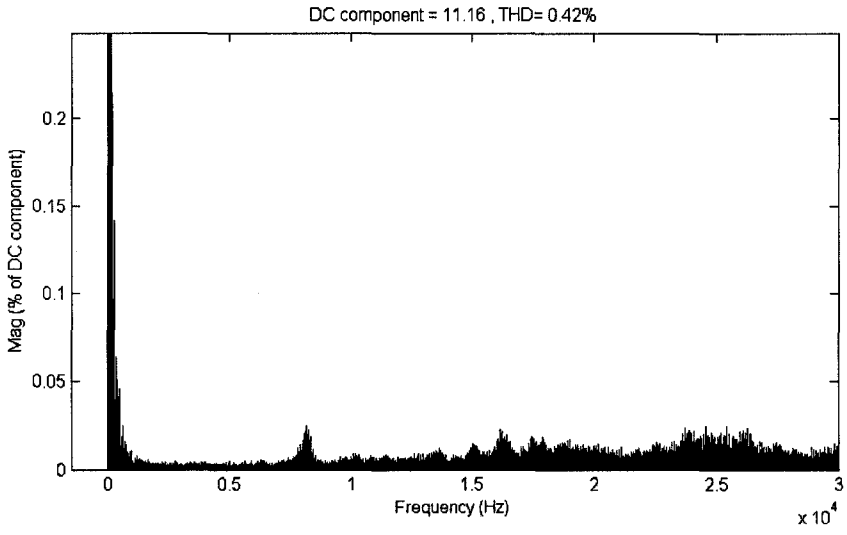


Figure 53 Spectre de fréquence

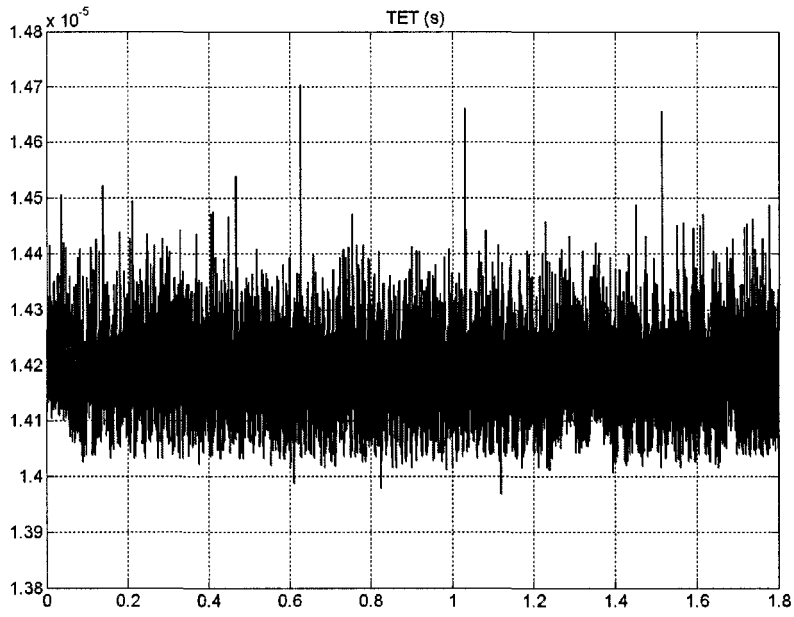


Figure 54 Temps d'exécution des tâches

4.6 Comparaison et analyse pour le modèle AC3

La comparaison entre les figures 36 et 39 montre des résultats très semblables. Tout d'abord, il faut négliger le délai entre les signaux de la figure 39 et ceux de la figure 36 car la méthode pour démarrer la simulation en temps réel avec commande externe occasionne un délai entre le simulateur et la commande. La zone transitoire de départ doit être négligée car elle se situe à l'intérieur du délai, dans lequel le simulateur reçoit des signaux nuls.

Par la suite, la figure 40 présente un agrandissement d'une section du couple qui montre un niveau de bruit très élevé. Ce bruit a été suspecté d'être généré en grande partie par la perte de résolution des CAN des cartes PCI-MIO-16E-4 et DS1104. Comme mentionné dans les sections 3.2.1.1 et 3.3.1.1, un modèle amélioré a été conçu. Ce modèle permet de doubler la résolution des références des courants statoriques i_{as}* et i_{bs}*, ainsi que

d'améliorer la résolution de la vitesse mécanique du moteur (ω_m). Le modèle permet donc de vérifier la contribution du bruit provenant de la perte de résolution des CAN des cartes PCI-MIO-16E-4. La figure 43 présentent les résultats obtenus pour ce modèle et sont aussi très semblables aux résultats de la figure 36. De plus, les signaux de la figure 43 présentent un niveau de bruit moins élevé que ceux de la figure 39. L'agrandissement de la section du couple présenté par la figure 44 montre plus précisément le niveau de bruit moins élevé par rapport à la figure 40. Pour confirmer la réduction du niveau de bruit, une analyse spectrale a été effectuée sur les sections du couple électromagnétique. Les figures 41 et 45 présentent les spectres de fréquences des deux sections. La comparaison entre les deux spectres montre que les amplitudes des harmoniques du signal de couple du modèle amélioré sont réduites par rapport à celle du premier modèle. Ceci confirme qu'une partie du bruit est généré par la perte de résolution des CAN des cartes PCI-MIO-16E-4.

Par contre, l'ajout de deux entrées analogiques dans le modèle amélioré pour le doublage de la résolution des références des courants statoriques augmente les temps d'exécution des tâches (TET) en temps réel. La figure 42 présente les TET pour la simulation de la première version du modèle et la figure 46 présente les TET pour la version améliorée. Le doublage de la résolution des références des courants statoriques dans le modèle amélioré produit une augmentation du TET moyen de presque 3 µs par rapport au premier modèle. Les valeurs des TET moyens obtenus pour tous les modèles AC3 et AC6 sont présentés dans le tableau I.

Tableau I

Temps d'exécution de tâche moyen pour AC3

	AC3	AC3 amélioré
TET moyen (µs)	13,5	16,2

4.7 Comparaison et analyse pour le modèle AC6

Comme dans le cas du modèle AC3, la comparaison entre les figures 37 et 47 montre des résultats très semblables. Encore une fois, il faut négliger le délai entre les signaux de la figure 47 et ceux de la figure 37 car la méthode pour démarrer la simulation en temps réel avec commande externe occasionne un délai entre le simulateur et la commande. Dans ce cas-ci, la zone transitoire occasionnée par le délai a été retirée.

Par la suite, la figure 48 présente un agrandissement d'une section du couple qui montre un niveau de bruit élevé. Comme mentionné dans les sections 3.2.1.2 et 3.3.1.2, un modèle amélioré a été conçu. Ce modèle permet de doubler la résolution des références de courants statoriques (i_{ab}*), de la vitesse mécanique du moteur (ω_m) et de la position angulaire mécanique du rotor (θ_r). Le modèle permet donc de vérifier la contribution du bruit provenant de la perte de résolution des CAN des cartes PCI-MIO-16E-4 et de la carte DS1104. La figure 51 présentent les résultats obtenus pour ce modèle et sont aussi très semblables aux résultats de la figure 37. De plus, les signaux de la figure 51 présentent un niveau de bruit moins élevé que ceux de la figure 47. L'agrandissement de la section du couple présenté par la figure 52 montre plus précisément le niveau de bruit moins élevé par rapport à la figure 48. Pour confirmer la réduction du niveau de bruit, une analyse spectrale a été effectuée sur les sections du couple électromagnétique. Les figures 49 et 53 présentent les spectres de fréquences des deux sections. La comparaison entre les deux spectres montre que les amplitudes des harmoniques du signal de couple du modèle amélioré sont réduites par rapport à celle du premier modèle. Ceci confirme qu'une partie du bruit est généré par la perte de résolution des CAN des cartes PCI-MIO-16E-4 et de la carte DS1104.

Dans ce cas-ci, l'ajout de deux entrées analogiques et de deux sorties analogiques dans le modèle amélioré pour le doublage de la résolution des signaux d'entrée et de sortie produit une augmentation plus importante des TET comparativement à l'augmentation

observée pour le modèle AC3. La figure 50 présente les TET pour la simulation de la première version du modèle et la figure 54 présente les TET pour la version améliorée. Le doublage de la résolution des références des courants statoriques, de la vitesse mécanique et de la position mécanique dans le modèle amélioré produit une augmentation du TET moyen de 5 µs par rapport au premier modèle. Les valeurs des TET moyens obtenus sont présentés dans le tableau II.

Tableau II

Temps d'exécution de tâche moyen pour AC6

	AC6	AC6 amélioré
TET moyen (µs)	9,1	14,2

4.8 Conclusion

Ce chapitre a présenté les résultats de simulation en temps différé et les résultats de simulation en temps réel avec commande externe des modèles d'entraînement électrique à valeur moyenne présenté dans le chapitre 2. La comparaison et l'analyse des résultats ont démontrées la validité de la simulation en temps réel avec commande externe. Ce chapitre conclut donc la première partie du projet. Le prochain chapitre présente les travaux effectués pour la réalisation de la deuxième partie du projet de recherche, soit la simulation en temps réel avec commande externe des modèles d'entraînement électrique détaillés.

CHAPITRE 5

IMPLÉMENTATION DES MODÈLES DÉTAILLÉS

5.1 Introduction

La deuxième partie du projet de recherche consiste à réaliser la simulation en temps réel de modèles détaillés avec commande externe. Ce chapitre présente la démarche utilisée pour réaliser cette simulation. Une première partie introduit les caractéristiques de cette simulation et expose les stratégies développées pour en faire la réalisation. Les parties suivantes présentent le développement réalisé et les résultats obtenus.

5.2 Simulation en temps réel des modèles détaillés

Les modèles détaillés présentent des onduleurs triphasés possédant des ponts de six interrupteurs qui génèrent des discontinuités. D'autre part, les commandes des modèles détaillés possèdent des hystérésis de courant qui génèrent des signaux de référence pour les onduleurs sous forme d'impulsions. Puisque les impulsions sont générées à une fréquence élevée et qu'elles sont asynchrones à la simulation, elles peuvent survenir à l'intérieur d'un pas de calcul de la simulation en temps réel. La principale caractéristique de la simulation en temps réel des modèles détaillés est de faire la correction des événements se produisant à l'intérieur des pas de calcul.

La simulation en temps réel est réalisée à l'aide d'un simulateur développé par le professeur Bruno De Kelper de l'École de technologie supérieure dans le cadre de son projet de doctorat. Ce simulateur, qui utilise la méthode des commutations précises, est conçu pour corriger les événements se produisant à l'intérieur des pas de calcul de la simulation en temps réel et fonctionne dans l'environnement de MATLAB et d'xPC

Target. La principale contribution du projet de maîtrise est de faire la réalisation et l'intégration d'une interface entre le simulateur et la commande externe.

L'interface doit être intégrée au simulateur et doit traiter les signaux provenant de la commande externe. L'interface doit déterminer deux informations essentielles pour faire le traitement des impulsions des signaux de la commande, soit les instants précis où les signaux changent d'états ainsi que les états correspondants aux changements. L'interface matérielle est composée des deux cartes PCI-MIO-16E-4. Puisqu'il n'existe pas de pilote dans xPC Target permettant un tel traitement, un pilote spécifique a été développé.

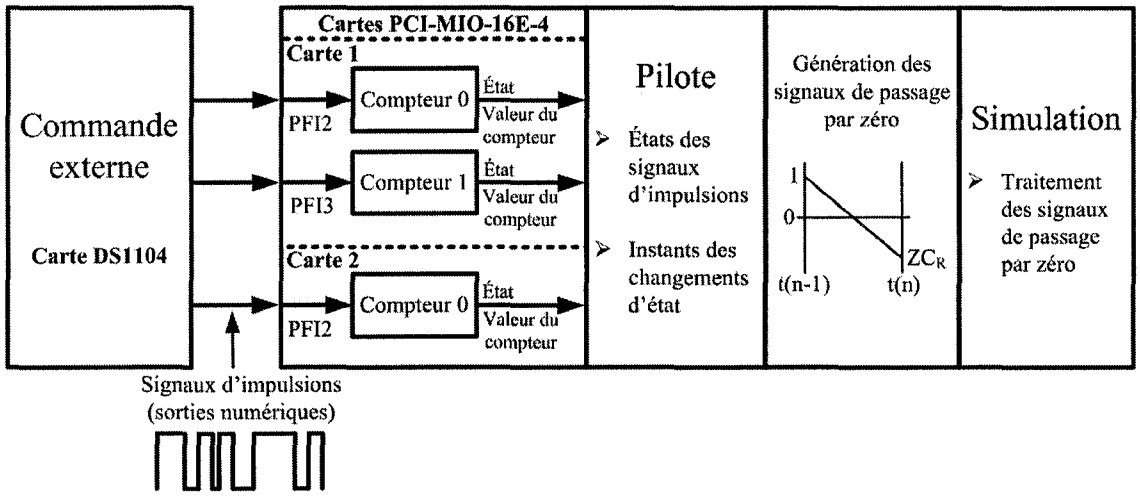


Figure 55 Schéma de principe de la simulation en temps réel avec commande externe

La pilote développé utilise les compteurs des cartes pour déterminer les informations nécessaires. Trois des six signaux d'impulsions provenant de la commande sont connectés à trois des quatre compteurs disponibles sur les cartes PCI-MIO-16E-4. Les trois autres signaux sont générés de façon logicielle puisqu'ils sont complémentaires. Grâce aux fonctionnalités des compteurs, le pilote génère les instants où les signaux changent d'états ainsi que les états correspondants aux changements. Les détails sur le développement du pilote se trouvent dans la section 5.3. La figure 55 présente le schéma

de principe de la stratégie utilisée pour la réalisation de la simulation en temps réel avec commande externe des modèles d'entraînement électrique détaillés.

Lors de l'intégration du pilote, les informations déterminées doivent être traitées pour corriger les événements se produisant à l'intérieur des pas de calculs de la simulation en temps réel. Le simulateur utilise la technologie de détection de passage par zéro de MATLAB pour détecter les discontinuités. Les informations doivent être traduites sous formes de signaux de passage par zéro pour que le simulateur puisse traiter les impulsions provenant de la commande externe comme les autres discontinuités. Des modifications sont apportées au pilote pour faire la génération de signaux de passage par zéro. Des modifications sont aussi apportées au simulateur pour faire le traitement des signaux de passage par zéro du pilote. La section 5.4 présente tous les détails concernant l'intégration du pilote dans le simulateur, la génération et le traitement des signaux de passage par zéro.

5.3 Développement du pilote

Les cartes PCI-MIO-16E-4 sont munies d'un circuit intégré spécialisé (ASIC) nommé DAQ-STC qui comporte une multitude de fonctionnalités. Dans le but d'obtenir le pilote désiré, les registres des DAQ-STC sont configurés afin d'utiliser les fonctionnalités nécessaires. Le pilote a été réalisé à l'aide des documents [1] et [2] de National Instruments sur la programmation des registres du DAQ-STC de la carte PCI-MIO-16E-4. Le pilote développé détermine les instants où les signaux changent d'états ainsi que les états correspondants aux changements. Le tableau III présente la description des registres utilisés et l'annexe 3 présente la description des bits de ces registres.

Tableau III

Description des registres[1]

Registre	Addresse (*i*=0/1)	Fonction
G*i*_Mode_Register	26/27	Ce registre permet de définir le mode de fonctionnement du compteur. Le registre défini entre autre le fonctionnement du compteur par rapport au signal G*i*_GATE.
G*i*_Command_Register	6/7	Ce registre permet de commander le compteur. Il permet entre autre d'armer et de désarmer le compteur, de faire le chargement du compte à partir des registres de chargement, ainsi que de choisir le sens du compte.
G*i*_Input_Select_Register	36/37	Ce registre détermine la source des signaux G*i*_GATE et G*i*_SOURCE. Il spécifie quelques caractéristiques du signal G*i*_GATE ainsi que la polarité des signaux G*i*_OUT et G*i*_SOURCE.
G*i*_Autoincrement_Register	68/69	Ce registre de 8 bits contient une valeur fixe ajoutée au contenu du registre de chargement A après chaque chargement du compteur pour maintenir une valeur incrémentée du compte.
Interrupt_A_Enable_Register	73	Ce registre permet de sélectionner les signaux et les conditions du groupe A qui déclenchent une demande d'interruption.

1.L'indice *i* représente l'indice d'un compteur (compteur 0 ou compteur 1). Les deux compteurs de la première carte sont utilisés et le compteur 0 de la deuxième carte est utilisé.

Tableau III (suite)

Registre	Addresse (*i*=0/1)	Fonction
Interrupt_B_Enable_Register	75	Ce registre permet de sélectionner les signaux et les conditions du groupe B qui déclenchent une demande d'interruption.
Interrupt_A_Ack_Register	2	Ce registre permet de remettre à 0 les conditions d'interruptions et de faire la reconnaissance des demandes d'interruptions du groupe A.
Interrupt_B_Ack_Register	3	Ce registre permet de remettre à 0 les conditions d'interruptions et de faire la reconnaissance des demandes d'interruptions du groupe B.
IO_Bidirection_Pin_Register	57	Ce registre permet de sélectionner la direction en entrée ou en sortie des dix broches PFI.
G*i*_Load_A_Registers	28-29/32-33	Ce registre de chargement contient une valeur modifiable qui peut être chargée dans le compteur lorsqu'il est arrêté. Ce registre comprend une partie haute de 8 bits et une partie basse de 16 bits.
Joint_Status_1_Register	27	Ce registre fourni le statut de quelques signaux et conditions. Il fourni entre autre le statut des signaux G0_GATE et G1_GATE.
G*i*_HW_Save_Registers	8-9/10-11	Ce registre sauvegarde la valeur du compte sur les fronts actifs du signal G*i*_GATE. Ce registre comprend une partie haute de 8 bits et une partie basse de 16 bits.

Le pilote conçu utilise les fonctionnalités des modules des compteurs d'usage général (GPCT) des DAQ-STC. Le GPCT consiste en deux compteurs (compteur 0 et compteur 1) indépendants de 24 bits associés à des registres de chargement et de sauvegarde. Les compteurs comportent trois signaux d'entrée, soit la source (G*i*_SOURCE), la grille (G*i*_GATE) et le contrôle de sens du compte (G*i*_UP_DOWN). Le G*i*_UP_DOWN permet de sélectionner le sens du compte en incrémentation ou en décrémentation. Dans le cadre du projet, le sens du compte des compteurs est incrémental. Le signal G*i*_SOURCE, qui représente l'horloge du compteur, est relié à la base de temps interne du DAQ-STC, soit le signal G_IN_TIMEBASE1 qui est un signal de forme carrée d'une fréquence de 20 MHz. Les compteurs incrémentent donc avec la fréquence de l'horloge qui est de 20 MHz. Finalement, le signal G*i*_GATE est relié à un signal d'impulsion provenant de la commande. Les trois signaux d'impulsion de la commande sont connectés à trois des quatre signaux G*i*_GATE des compteurs via les broches PFI (Programmable Function Inputs). Les deux compteurs de la première carte sont utilisés et le compteur 0 de la deuxième carte est utilisé.

Le pilote doit tout d'abord effectuer l'initialisation des registres des compteurs. Le tableau IV présente la configuration des registres pour l'initialisation des cartes. Le pilote doit ensuite configurer les registres des cartes pour obtenir les fonctionnalités désirées. Le tableau V présente la configuration des registres pour le pilote. Selon la configuration des registres, les signaux G*i*_GATE permettent, via des registres de sauvegarde et de statut, de déterminer les états et les instants des changements d'état des signaux d'impulsion. Les bits G*i*_Gate_St situés dans les registres Joint_Status_1_Register fournissent les états des signaux G*i*_GATE. Les instants des changements d'état sont fournis par les registres G*i*_HW_Save_Registers, qui contiennent la sauvegarde des comptes. La sauvegarde d'un compte est effectuée de façon matérielle par la carte lorsqu'il survient un changement d'état sur le signal G*i*_GATE correspondant. La sauvegarde des comptes est faite sur les fronts montants et descendants des signaux G*i*_GATE. Les bits G*i*_Gate_St et les registres G*i*_HW_Save_Registers sont lus à la fin de chaque pas de calcul de la simulation

afin de déterminer les nouveaux états et les instants de changement d'état des signaux d'impulsions.

Au début de chaque pas de calcul, les valeurs des comptes sont remises à zéro. De cette façon, les valeurs des comptes recueillies à la fin d'un pas de calcul représentent les délais entre le début du pas et les événements. Les valeurs des comptes sont des valeurs binaires d'une résolution de 24 bits qui incrémentent à la fréquence des compteurs de 20 MHz, soit une période de 50 ns. Pour obtenir les temps des événements en secondes, les valeurs des comptes sont multipliées par la période des compteurs. Cette relation est présentée par l'équation (5.1).

$$t_{\text{événement}} = \text{Valeur du compte} \cdot T_{\text{compteur}} \tag{5.1}$$

Tableau IV

Initialisation des registres[1]

Registre	Valeur (*i*=0/1)	Explication
G*i*_Mode_Register	0x0000/0x0000	Initialisation de tous les bits du registre.
G*i*_Command_Register	0x0000/0x0000	Initialisation de tous les bits du registre.
G*i*_Input_Select_Register	0x0000/0x0000	Initialisation de tous les bits du registre.
G*i*_Autoincrement_Register	0x0000/0x0000	Initialisation de tous les bits du registre.
Interrupt_A_Enable_Register	0x0000/0x0000	Initialisation de tous les bits du registre.

1.L'indice *i* représente l'indice d'un compteur (compteur 0 ou compteur 1). Les deux compteurs de la première carte sont utilisés et le compteur 0 de la deuxième carte est utilisé.

Tableau IV (suite)

Registre	Valeur (*i*=0/1)	Explication
Interrupt_B_Enable_Register	0x0000/0x0000	Initialisation de tous les bits du registre.
G*i*_Command_Register	0x0100/0x0100	Synchronisation du signal G*i*_GATE avec le signal G*i*_SOURCE.
Interrupt_A_Ack_Register	0xC060	Remise à 0 des conditions d'interruptions G0_Gate_Interrupt_St et G0_TC_St, ainsi que des conditions d'erreur G0_TC_Error_St et G0_Gate_Error_St.
Interrupt_B_Ack_Register	0xC006	Remise à 0 des conditions d'interruptions G1_Gate_Interrupt_St et G1_TC_St, ainsi que des conditions d'erreur G1_TC_Error_St et G1_Gate_Error_St.

Tableau V

Configuration des registres[1]

Registre	Valeur (*i*=0/1)	Explication
IO_Bidirection_Pin_Register	0x0000	Sélection de toutes les broches PFI en entrée.
G*i*_Load_A_Registers	0x0000-0x0000 / 0x0000-0x0000	Affectation de la valeur 0 dans le registre de chargement A.

1.L'indice *i* représente l'indice d'un compteur (compteur 0 ou compteur 1). Les deux compteurs de la première carte sont utilisés et le compteur 0 de la deuxième carte est utilisé.

Tableau V (suite)

Registre	Valeur (*i*=0/1)	Explication
G*i*_Mode_Register	0x007F/0x007F	• Les fronts montants et descendants du signal G*i*_GATE sont actifs. • Sélection du mode front. Le compteur est contrôlé par les fronts actifs du signal G*i*_GATE.
G*i*_Command_Register	0x0120/0x0120	• Synchronisation du signal G*i*_GATE avec le signal G*i*_SOURCE. • Sélection du sens du compte en incrémentation.
G*i*_Input_Select_Register	0x0180/0x0200	• Sélection d'une broche PFI comme source du signal G*i*_GATE (PFI2 pour le compteur 0 et PFI3 pour le compteur 1). • Sélection du signal G_IN_TIMEBASE1 comme source du signal G*i*_SOURCE. • Sélection du front montant pour le front actif du signal G*i*_SOURCE. • Les autres options sont mises hors fonction.
Interrupt_A_Enable_Register	0x0000	Les signaux et conditions du groupe A ne génèrent pas d'interruption.
Interrupt_B_Enable_Register	0x0000	Les signaux et conditions du groupe B ne génèrent pas d'interruption.

Tableau V (suite)

Registre	Valeur (*i*=0/1)	Explication
G*i*_Mode_Register	0x007F/0x007F	• Le registre de chargement A est toujours utilisé pour le chargement du compteur. • Le compteur n'est pas chargé lors d'une condition sur le signal G*i*_GATE. • Sélection du niveau haut pour le niveau actif du signal G*i*_GATE. • À la fin du compte, le compteur recommence à compter à partir de 0. • Il n'y a pas de désarmement matériel du compteur. • Le signal G*i*_GATE ne démarre pas et n'arrête pas le compteur.

5.4 Intégration du pilote dans le simulateur

Le pilote développé pour les compteurs des cartes PCI-MIO-16E-4 est intégré dans le simulateur avec une S-function. Une S-function décrit un bloc Simulink en langage machine et peut être écrite en plusieurs langages. Dans le projet, la S-function est écrite en langage C et doit être compilée comme un fichier MEX. Une S-function utilise une syntaxe d'appel de fonction qui permet d'interagir avec l'engin de calcul de la simulation. La S-function est intégrée dans Simulink par un bloc S-function, qui permet de faire interagir la S-function avec les autres blocs de Simulink.

L'implémentation du pilote se fait à travers les diverses fonctions de la S-function. La description des fonctions utilisées pour le pilote est présentée par le tableau VI. Comme mentionné dans le tableau, la fonction mdlStart permet de faire l'initialisation et la configuration des registres du pilote pour obtenir les fonctionnalités désirées. La fonction

mdlOutputs, qui est appelée à tous les pas de calculs de la simulation, met à jour les trois signaux de sortie de la S-function qui représentent les états des trois signaux d'impulsions provenant de la commande externe. Les signaux de sortie de la S-function sont mis à jour avec des variables nommées MODE. Chaque signal de sortie est associé à sa propre variable MODE. Ces variables sont aussi mises à jour dans la fonction mdlOutputs et alternent entre les valeurs 0 et 1 lorsqu'il y a détection de passage pas zéro sur leur signal de passage par zéro respectif. Un algorithme de détection de passage par zéro est donc implémenté dans la fonction mdlOutputs pour traiter les signaux de passage par zéro générés par la fonction mdlZeroCrossings. La figure 56 présente la séquence d'appel des fonctions durant la simulation.

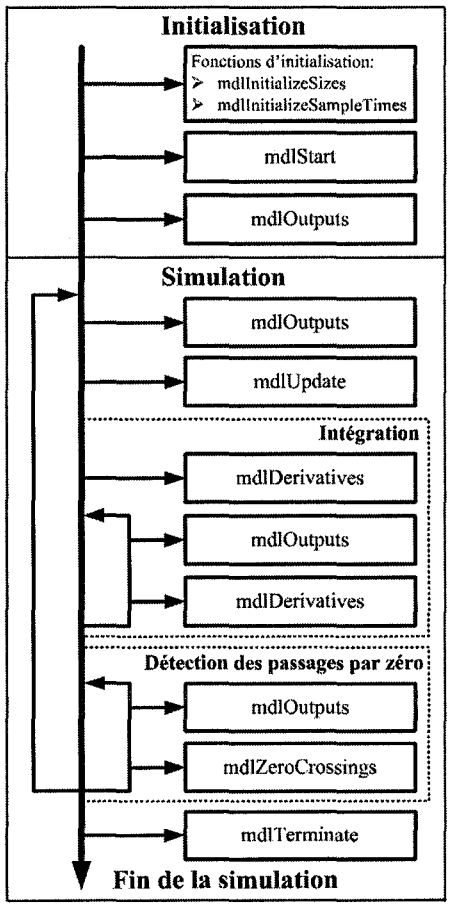


Figure 56 Séquence d'appel des fonctions de la simulation

Les signaux de passage par zéro ont aussi l'utilité de fournir les instants des changements d'état des signaux d'impulsions. Les détails sur la génération des signaux de passage par zéro sont présentés dans la section 5.4.1. Ces signaux doivent aussi être traités dans l'engin de calcul de la simulation. Les détails sur le traitement des signaux de passage par zéro sont présentés dans la section 5.4.2.

Tableau VI

Fonctions de la S-function

Fonction	Description
mdlInitializeSizes	Cette fonction spécifie le nombre d'entrées, de sorties, d'états, de paramètres et autres caractéristiques de la S-function. Elle est la première fonction appelée par Simulink.
mdlInitializeSampleTimes	Cette fonction spécifie les périodes d'échantillonnage auxquelles la S-function opère.
mdlStart	Cette fonction initialise les vecteurs d'état. Elle permet aussi de faire d'autres initialisations requises par la S-function. Pour le pilote, elle est utilisée pour faire l'initialisation et la configuration des registres des cartes PCI-MIO-16E-4, comme présenté dans les tableaux IV et V.
mdlOutputs	Cette fonction est appelée à tous les pas de calcul de la simulation par Simulink pour mettre à jour les signaux de sortie de la S-function et pour stocker les résultats dans les matrices de signaux de sortie. Pour le pilote, elle permet de mettre à jour l'information sur l'état des signaux d'impulsions provenant de la commande.

Tableau VI (suite)

Fonction	Description
mdlZeroCrossings	Cette fonction est appelé à tous les pas de calcul de la simulation par Simulink pour mettre à jour le vecteur des signaux de passage par zéro. Pour le pilote, elle permet de générer les signaux de passage par zéro associés aux changements d'état des signaux d'impulsions provenant de la commande.
mdlTerminate	Cette fonction performe les actions nécessaires à la terminaison de la simulation. Pour le pilote, elle permet d'effectuer la réinitialisation des cartes PCI-MIO-16E-4.

5.4.1 Génération des signaux de passage par zéro

La fonction mdlZeroCrossings est utilisée pour fournir à Simulink les signaux pour lesquels la détection des passages par zéro est nécessaire. Normalement, ce sont des signaux continus à l'entrée de la S-function ou des signaux générés à l'interne qui croisent le zéro lorsqu'une discontinuité survient dans la fonction mdlOutputs. Conséquemment, les signaux de passage par zéro sont utilisés pour localiser les discontinuités dans le pas de calcul selon les points où ils croisent le zéro. Pour fournir les signaux de passage par zéro à Simulink, la fonction mdlZeroCrossings met à jour le vecteur de passage par zéro ssGetNonsampleZCs(S).

Pour le pilote, les signaux de passage par zéro associés aux changements d'état des signaux d'impulsions provenant de la commande sont générés à l'interne de la fonction mdlZeroCrossings de la S-function. Les signaux générés croisent le zéro lorsqu'il y a un changement d'état sur leur signal d'impulsion respectif. Les points où il y a croisement du zéro représentent les instants dans un pas de calcul où s'est produit les changements

d'état. La figure 57 présente le schéma du principe utilisé pour la génération des signaux de passage par zéro.

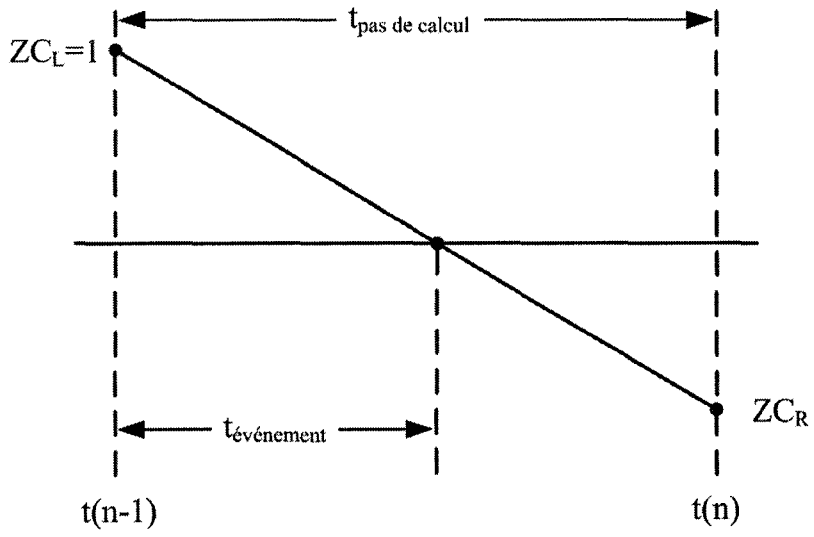


Figure 57 Signal de passage par zéro

$$K = \frac{t_{événement}}{t_{pas\ de\ calcul}} - 1 \tag{5.2}$$

$$ZC_R = \frac{K \cdot ZC_L}{K + ZC_L} \tag{5.3}$$

Selon la figure 57, dans un signal de passage par zéro, il y a deux points particuliers aux extrémités de chaque pas de calcul, soit le point gauche (ZC_L) au début du pas de calcul et le point droit (ZC_R) à la fin du pas de calcul. Ces deux points sont reliés par une droite. Au début de chaque pas de calcul, le ZC_L est placé à 1. Si un événement survient à l'intérieur d'un pas de calcul, le ZC_R est calculé pour que l'interpolation de la droite au niveau de l'abscisse représente le temps où l'événement s'est produit. Tout d'abord, la pente de la droite (K) entre le ZC_L et le ZC_R est obtenue par l'équation (5.2). Cette pente est calculée avec la proportion du pas de calcul où est survenu l'événement et utilise le

temps de l'événement déterminé par l'équation (5.1). Ensuite, le ZC_R est déterminé par l'équation (5.3), qui représente l'équation de la droite.

Dans le cas où il ne survient pas d'événement à l'intérieur d'un pas de calcul, le ZC_R est calculé pour représenter la proportion du temps restant dans le pas de calcul. Ce calcul est présenté par l'équation (5.4). Cette opération est réalisée car le temps d'exécution est normalement plus rapide que le temps du pas de calcul. Cette situation est présentée par la figure 58, qui montre un temps d'exécution plus court que le temps du pas de calcul. Le temps de l'événement utilisé par l'équation (5.4) est le compte du compteur à la fin du temps d'exécution, ce qui permet d'indiquer la proportion du temps restant dans le pas de calcul. Par exemple, selon la figure 58, le temps d'exécution correspond à 40% du temps du pas de calcul réel. Le signal de passage par zéro résultant dans le cas où il n'y a pas d'événement à l'intérieur du pas de calcul est présenté par la figure 59, où le ZC_R représente la proportion du temps restant dans le pas de calcul, soit 60%.

$$ZC_R = 1 - \frac{t_{événement}}{t_{pas\ de\ calcul}} \tag{5.4}$$

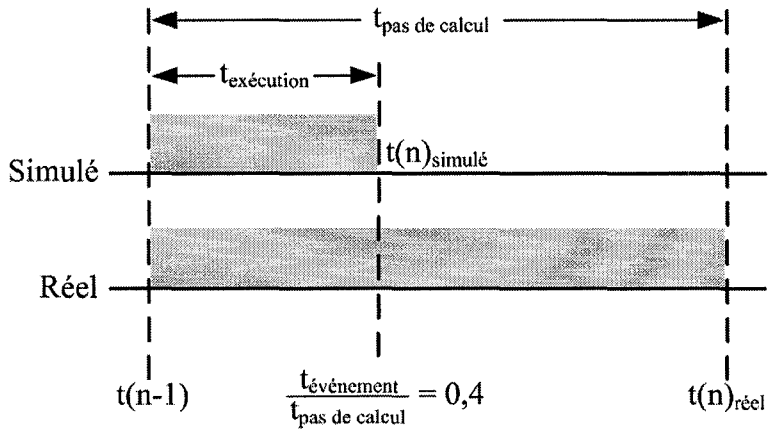


Figure 58 Pas de calcul et temps d'exécution

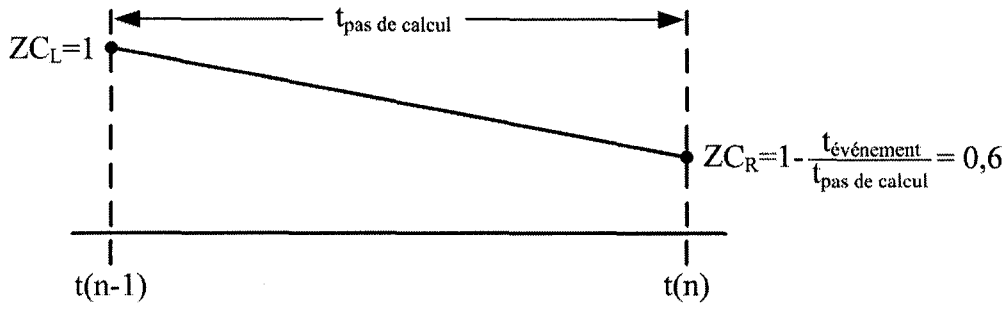


Figure 59 Signal de passage par zéro sans événement

5.4.2 Traitement des signaux de passage par zéro

Durant la simulation, l'engin de calcul de Simulink gère les appels de fonction du modèle. Le modèle Simulink appelle ensuite les fonctions des blocs Simulink et des S-functions selon la demande de l'engin de calcul. L'engin de calcul appelle tout d'abord les fonctions d'initialisation et de démarrage telles que mdlInitializeSizes, mdlInitializeSampleTimes et mdlStart. Par la suite, il démarre la boucle de simulation et fait des appels aux fonctions telles que mdlOutputs, mdlUpdate, mdlDerivatives et mdlZeroCrossings. Il termine ensuite la simulation en appelant les fonctions de terminaison mdlTerminate. La séquence d'appel des fonctions durant la simulation est présentée par la figure 56.

L'engin de calcul fait un appel aux fonctions mdlOutputs avant de démarrer la boucle de simulation pour déterminer les états initiaux des signaux. La boucle de simulation est ensuite démarrée au temps 0. La simulation avance tout d'abord jusqu'à la fin du premier pas de calcul (t_R), comme le présente la figure 60. Tous les pas de calcul de la simulation sont associés à un temps de départ (t_L) et à un temps de fin (t_R). La simulation vérifie ensuite le vecteur de passage par zéro pour déterminer si un ou plusieurs événements sont survenus à l'intérieur du pas de calcul.

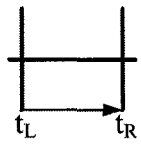


Figure 60 Avancement de la simulation à l'intérieur d'un pas de calcul

Si aucun événement n'est détecté à l'intérieur du pas de calcul, comme le présente l'exemple de la figure 61, la simulation qui a avancée à l'instant t_R ne nécessite pas de correction et débute alors un second pas de calcul. La figure 62 présente le signal de passage par zéro d'un pas de calcul pour lequel aucun événement n'a été détecté. Dans ce cas la simulation a avancé à l'instant t_R et le ZC_R correspond à la proportion du temps restant dans le pas de calcul.

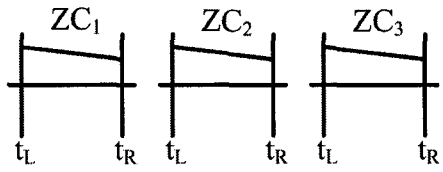


Figure 61 Aucune détection d'événement à l'intérieur d'un pas de calcul

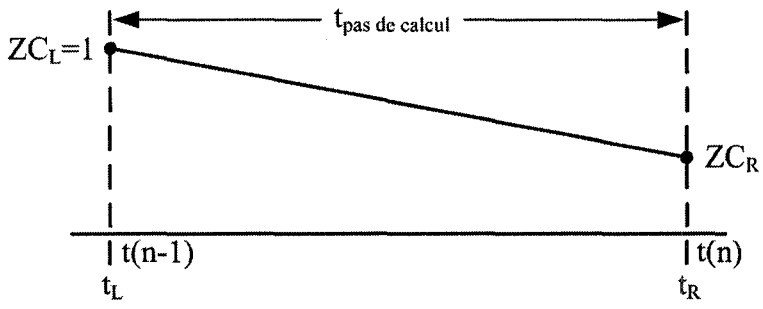


Figure 62 Signal de passage par zéro sans événement

Si un ou plusieurs événements sont détectés à l'intérieur du pas de calcul, comme le présente l'exemple de la figure 63, une logique détermine l'instant du premier événement détecté parmi tous les événements survenus. Dans l'exemple de la figure 63, le premier événement est détecté sur le signal de passage par zéro ZC_1. La figure 64 présente l'agrandissement du signal de passage par zéro ZC_1 dans lequel c'est produit le premier événement. La simulation qui avait avancé à l'instant t_R, retourne à l'instant du premier événement ($t_{événement}$), comme présenté par les figures 65 et 66. La simulation est par la suite corrigée en considérant le nouvel état du signal correspondant au premier événement.

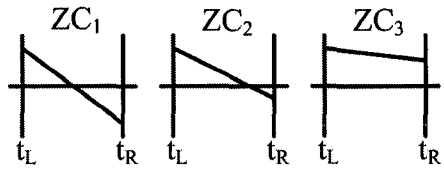


Figure 63 Détection de plusieurs événements à l'intérieur d'un pas de calcul

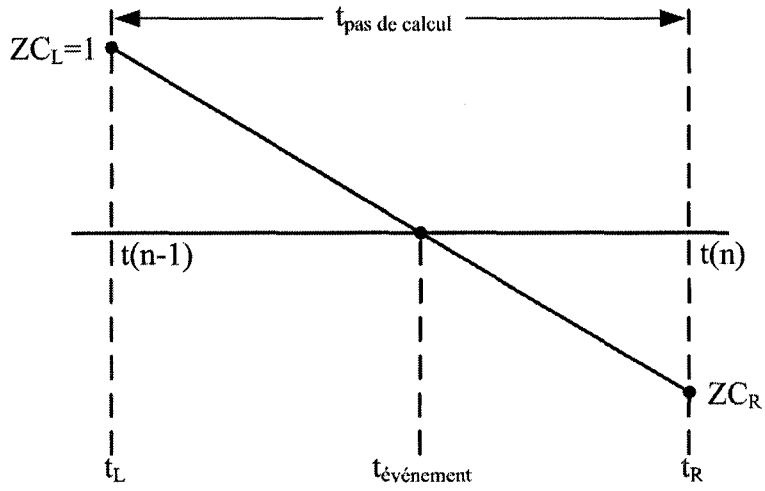


Figure 64 Signal de passage par zéro avec événement

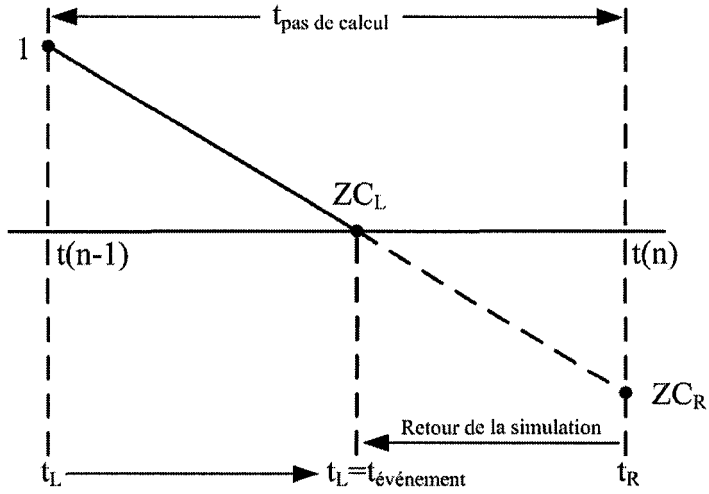


Figure 65 Retour de la simulation à l'instant $t_{événement}$

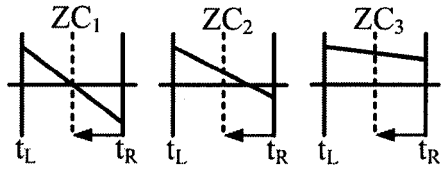


Figure 66 Retour de la simulation

Comme présenté par la figure 65, la simulation place le nouveau t_L à l'instant du premier événement ($t_{événement}$). De plus, les nouvelles valeurs de ZC_L à cet instant correspondent aux valeurs d'interpolation sur les droites où est survenu le premier événement. La même situation se produit pour les autres signaux de passage par zéro comme le présente la figure 66. Les signaux de passage par zéro associés aux signaux d'impulsions provenant de la commande nécessitent une correction avant que la simulation puisse avancer de nouveau à l'instant t_R. Cette correction est nécessaire afin que les droites entre les nouvelles valeurs des ZC_L et les prochaines valeurs des ZC_R croisent le zéro adéquatement afin de correspondre aux instants des changements d'états dans le cas où d'autres impulsions se produiraient. La correction de ces signaux de passage par zéro se

fait au nouveau t_L comme présenté par la figure 67. Pour que les droites croisent le zéro correctement, les nouvelles valeurs des ZC_L doivent correspondre à la proportion restante du pas de calcul et non aux valeurs d'interpolation sur les droites. Les nouvelles valeurs des ZC_L sont déterminées selon l'équation (5.4), qui présente le calcul utilisé pour déterminer la proportion restante du pas de calcul. Dans ce cas-ci, le temps de l'événement utilisé dans l'équation est le nouvel instant t_L. L'exemple de la figure 68 présente les trois signaux de passage par zéro associés aux signaux d'impulsions de la commande suite à la correction.

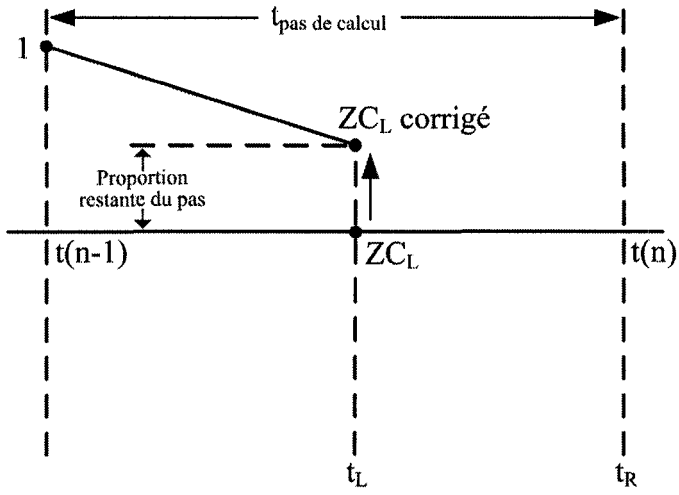


Figure 67 Correction d'un signal de passage par zéro

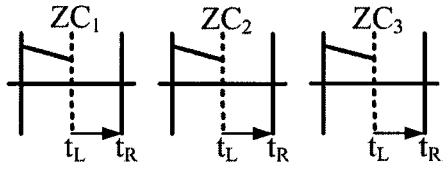


Figure 68 Correction des signaux de passage par zéro

Par la suite, la simulation avance à nouveau jusqu'à t_R et recommence le même traitement s'il y a détection d'événements. La simulation débute un second pas de calcul seulement lorsqu'elle a avancée à l'instant t_R et qu'elle ne détecte plus d'événement dans le pas de calcul. La figure 69 présente le signal de passage par zéro avec la détection d'un deuxième événement à l'intérieur du même pas de calcul. Cette figure démontre le fonctionnement adéquat de la génération des signaux de passage par zéro lorsque la correction des valeurs des ZC_L est réalisée.

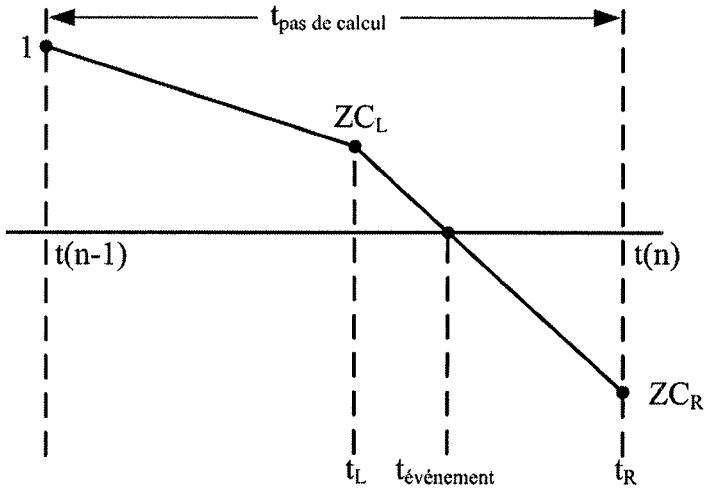


Figure 69 Signal de passage par zéro avec un deuxième événement

Le pas de calcul de la simulation en temps réel est fixe contrairement au temps d'exécution qui est variable et normalement plus court. Ceci représente la problématique majeure de la simulation en temps réel avec commande externe car les impulsions provenant de la commande peuvent survenir à l'intérieur du pas de calcul, mais après la fin du temps d'exécution. La figure 70 présente le schéma de comparaison entre le temps d'exécution et le pas de calcul pour la simulation en temps réel. Ce schéma montre le temps d'exécution sur l'axe du temps simulé et le pas de calcul sur l'axe du temps réel. S'il n'y a pas de modification apportée à l'engin de calcul, les impulsions qui surviennent dans cette période ne sont pas traitées. Afin de traiter ces impulsions, une

logique a été ajoutée dans la fonction mdlZeroCrossings de la S-function du pilote pour vérifier l'état des signaux d'impulsions au début des pas de calcul. Ceci permet de déterminer si les états des signaux d'impulsions au début d'un pas de calcul sont différents des états des signaux d'impulsions à la fin du temps d'exécution du pas de calcul précédent. En cas de différence, les changements d'état correspondants sont forcés, ce qui produit des passages par zéro au début du pas de calcul. La simulation est alors corrigée avec une erreur.

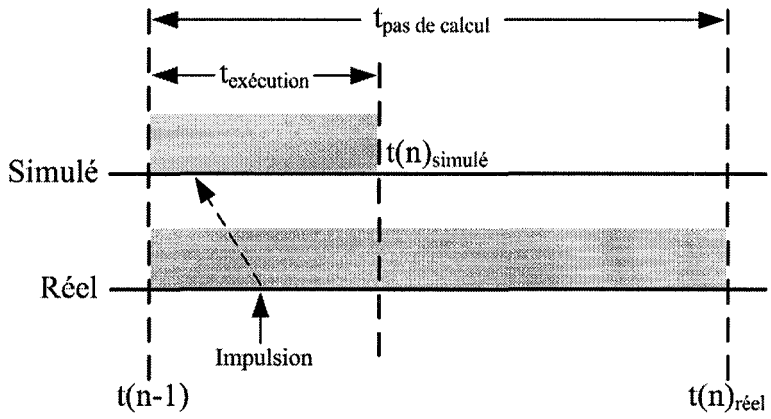


Figure 70 Correction d'une impulsion à l'intérieur du temps d'exécution

Dans la situation présentée par la figure 70, lorsqu'une impulsion survient à l'intérieur du temps d'exécution, la correction de la simulation s'effectue au temps correspondant à cette impulsion. Par contre, dans la situation présentée par la figure 71, lorsqu'une impulsion survient à l'intérieur du pas de calcul mais après la fin du temps d'exécution, la logique ajoutée permet de faire la correction de la simulation au début du nouveau pas de calcul. L'instant de la correction de la simulation correspond à la fin du temps d'exécution et non au temps correspondant à l'impulsion, ce qui introduit une erreur dans la simulation.

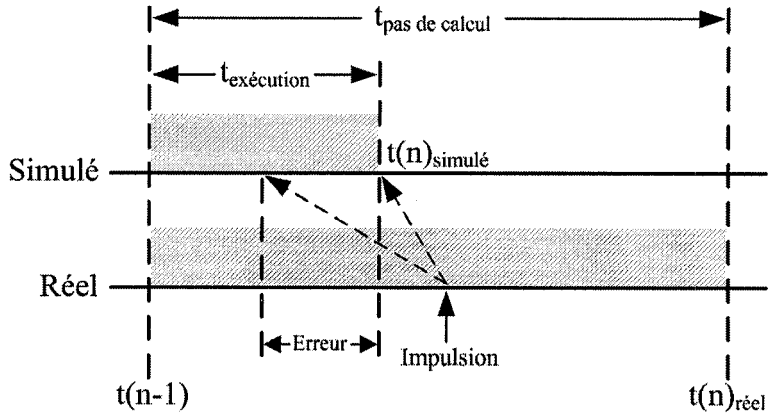


Figure 71 Correction d'une impulsion à l'extérieur du temps d'exécution

5.5 Résultats et analyse

Cette section présente les résultats obtenus pour la simulation en temps réel des modèles détaillés avec commande externe. La simulation a été réalisée à partir du modèle AC3 détaillé. Afin de comparer les résultats obtenus pour la simulation en temps réel du modèle, une simulation en temps différé a été réalisée. La partie de l'électronique de puissance et de la machine asynchrone est simulée avec la méthode des commutations précises du professeur De Kelper dans les deux cas. La figure 72 présente le schéma Simulink utilisé pour réaliser la simulation en temps différé du modèle AC3 détaillé avec commutations précises.

La commande du modèle détaillé est pratiquement identique à celle du modèle à valeur moyenne comme présentée dans la section 2.2.1. La seule différence se situe au niveau de la commande à flux orienté. La différence est que la commande fournit des signaux d'impulsions pour les interrupteurs de l'onduleur triphasé, contrairement à la commande du modèle à valeur moyenne qui fournit des références de courant aux sources de courant contrôlées de l'onduleur non standard. La figure 73 présente la commande à flux

orienté du modèle détaillé. La génération des signaux d'impulsions est réalisée avec des hystérésis de courant.

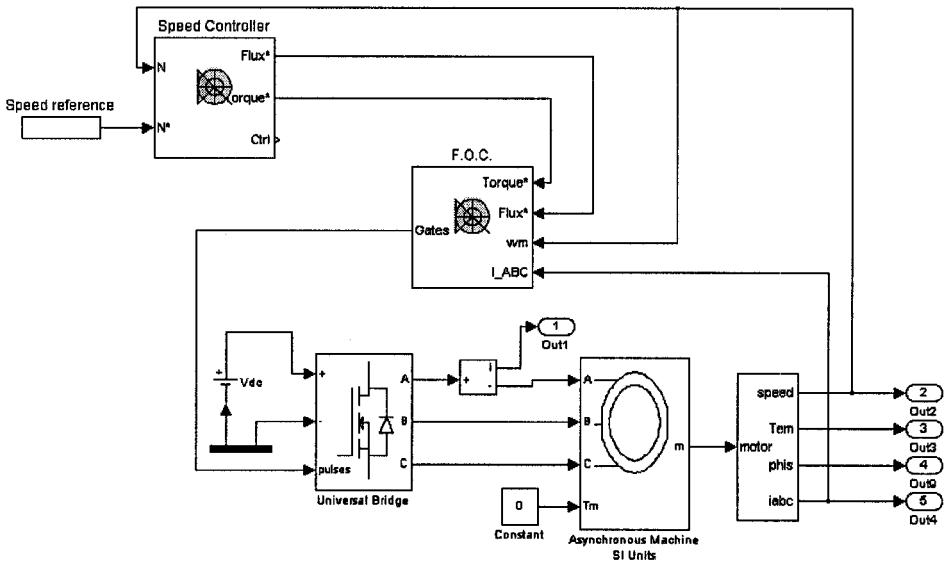


Figure 72 Schéma Simulink de simulation en temps différé

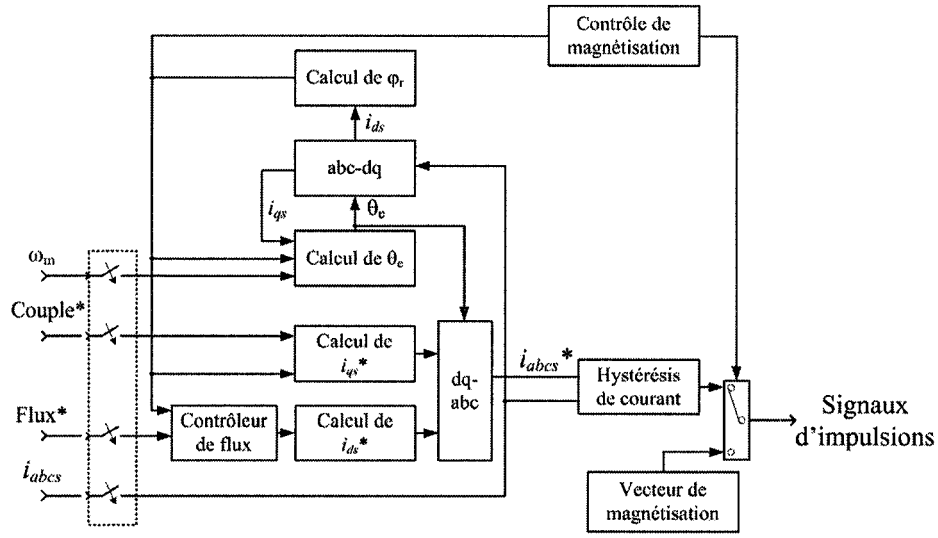


Figure 73 Commande à flux orienté

La simulation réalisée est la même pour les deux cas. La référence de vitesse est de 55 RPM et le couple mécanique est nul pour toute la durée de la simulation. La figure 74 présente les courants statoriques (i_{abcs}) obtenus pour la période de 0 s à 0,04 s de la simulation en temps différé. La figure 75 présente les signaux d'impulsions 1, 3 et 5 de la commande pour la même période de temps. Les signaux 2, 4 et 6 sont complémentaires. Les résultats montrent que les courants sont élevés durant la période de magnétisation pour faire augmenter le flux dans la machine. La fréquence des impulsions pendant cette période est relativement faible. La fréquence des impulsions augmente considérablement à partir de 0,015 s.

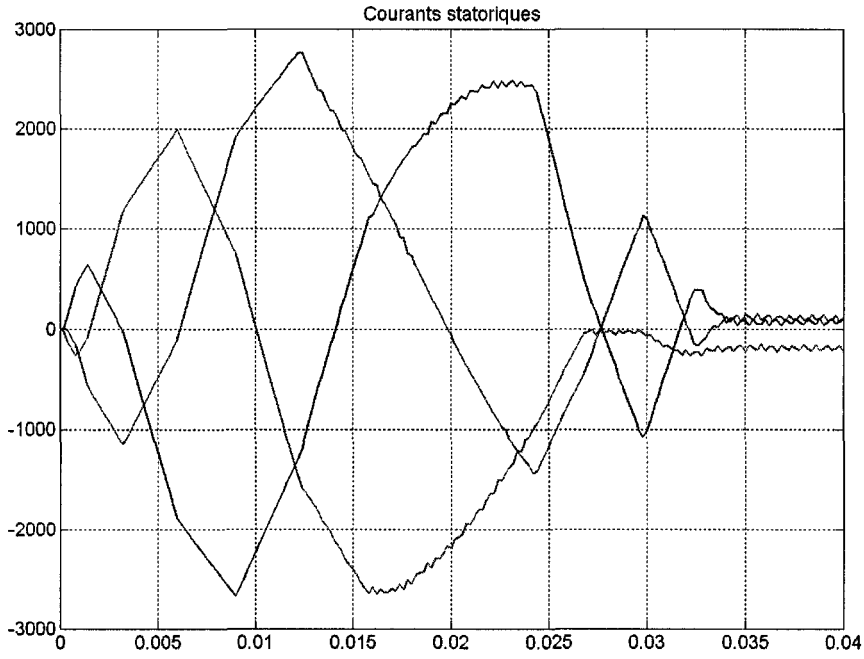


Figure 74 Résultats de simulation en temps différé

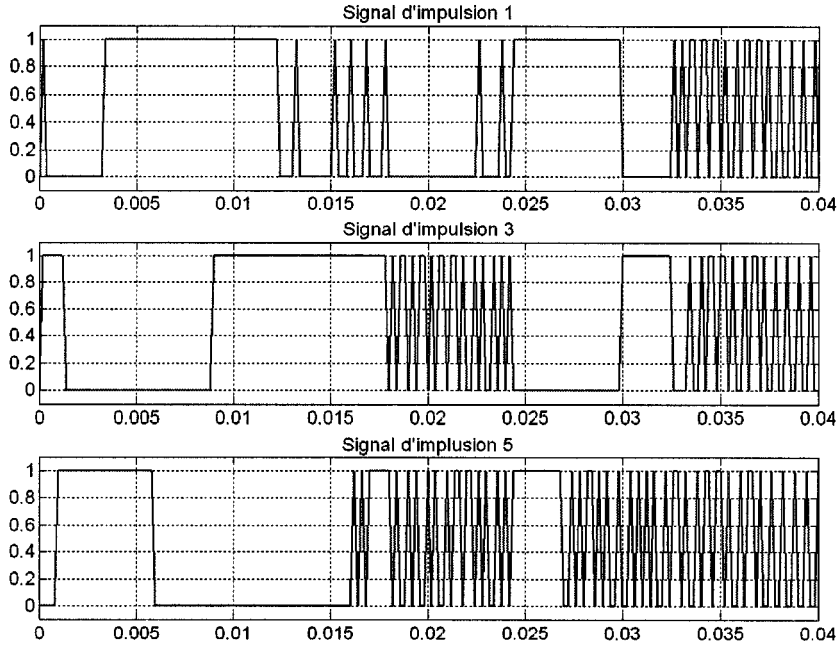


Figure 75 Signaux d'impulsions de la commande

Pour la simulation en temps réel du modèle détaillé avec commande externe, la commande est implémentée dans la carte DS1104 de la même façon que présentée dans la section 3.2. La figure 76 présente le schéma Simulink de la commande du modèle AC3 détaillé. Contrairement au modèle à valeur moyenne qui utilisent des sorties analogiques pour transmettre les références des courants statoriques, le modèle détaillé utilisent les sorties numériques pour transmettre les signaux d'impulsions. Les blocs de Simulink pour les sorties numériques de la carte DS1104 nécessitent comme information les numéros des sorties utilisées et leurs niveaux initiaux. Les sorties numériques utilisées sont les sorties 1, 2 et 3. Le signal d'impulsions 1 est transmis par la sortie 1, le signal d'impulsions 3 est transmis par la sortie 2 et le signal d'impulsions 5 est transmis par la sortie 3. Les niveaux initiaux des trois sorties sont de 0 V. Des blocs de conversion de type sont nécessaires avant les blocs de sortie numérique pour convertir le type double des signaux en type booléen. Les entrées analogiques 5 et 6 sont utilisées pour recevoir les courants statoriques i_{as} et i_{bs}. Le courant i_{cs} est obtenu selon l'équation (2.18).

L'entrée analogique 8 est utilisée pour recevoir la vitesse mécanique du rotor (ω_m). Les gains appliqués aux entrées ont été ajustés pour faire correspondre les plages de valeurs avec les plages de la simulation de l'électronique de puissance et de la machine. Ces gains ont été déterminés grâce aux plages de valeurs observées dans les résultats de la simulation en temps différé, comme expliqué dans la section 3.2.1.1.

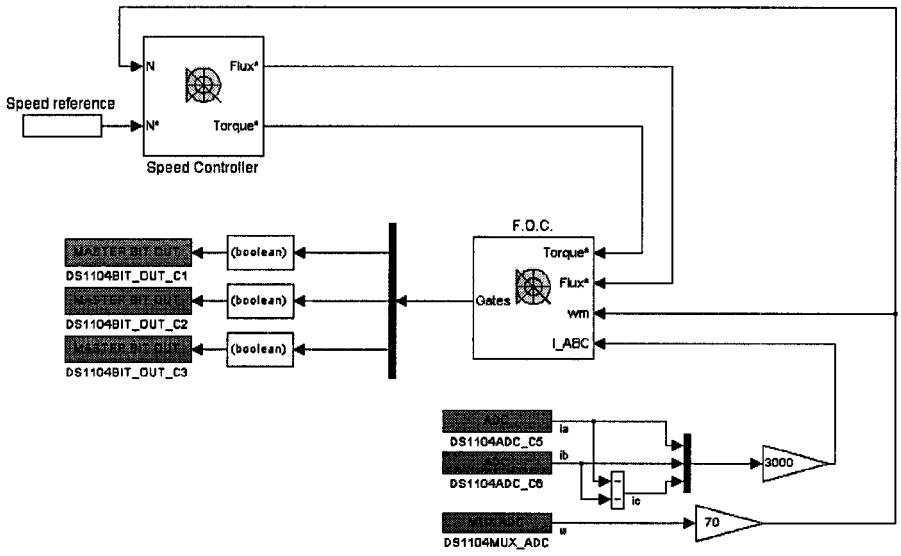


Figure 76 Schéma Simulink de la commande

La figure 77 présente le schéma Simulink de l'électronique de puissance et de la machine asynchrone. Des blocs S-function sont utilisés dans le modèle pour intégrer la S-function du pilote des cartes PCI-MIO-16E-4. La S-function est intégrée avec deux blocs. Le bloc Pilote_carte_1 est le pilote pour les deux compteurs de la première carte et le bloc Pilote_carte_2 est le pilote pour le compteur 0 de la deuxième carte. Les blocs des sorties analogiques des cartes PCI-MIO-16E-4 utilisés sont configurés de la même façon que dans la section 3.3.1.1. Les sorties de la première carte transmettent les courants statoriques i_{as} et i_{bs}. Les plages de fonctionnement de ces sorties sont de -10 V à 10 V et leurs valeurs initiales sont nulles. La première sortie de la deuxième carte transmet la vitesse mécanique du rotor (ω_m). La plage de fonctionnement de cette sortie est de 0 V à

10 V et sa valeur initiale est nulle. Les gains appliqués aux sorties ont été ajustés pour faire correspondre les plages de valeurs avec les plages de la commande externe.

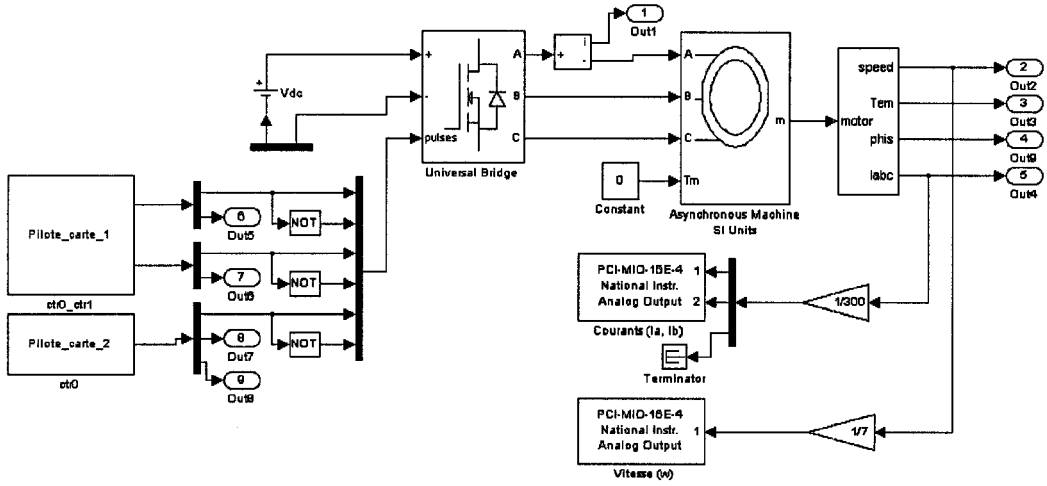


Figure 77 Schéma Simulink de l'électronique de puissance et de la machine

Comme mentionné précédemment, la simulation réalisée en temps réel avec commande externe pour le modèle AC3 détaillé est la même que celle en temps différé. Le pas de calcul utilisé pour obtenir le fonctionnement de la simulation pour la période de 0 s à 0,04 s est de 200 µs. La figure 78 présente les courants statoriques (i_{abcs}) obtenus pour cette période. La comparaison de ces résultats avec ceux de la figure 74 montrent quelques similitudes, mais démontre que le système n'est pas fonctionnel. En effet, durant la période de magnétisation, où la fréquence des impulsions est relativement faible, le comportement des courants statoriques est très similaire au comportement des courants de la simulation en temps différé. Par contre, la simulation n'est plus fonctionnelle à partir de 0,015 s, qui correspond au temps où la fréquence des impulsions augmente considérablement. Il est évident que les impulsions provenant de la commande ne sont pas toutes traitées. Par contre, le début de la simulation démontre que la correction des événements est fonctionnelle.

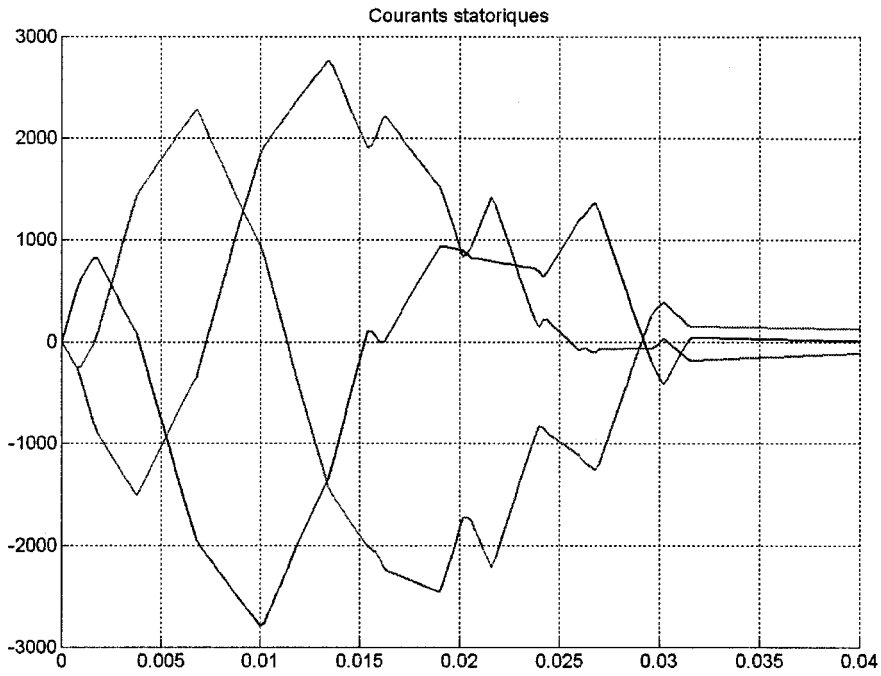


Figure 78 Résultats de simulation en temps réel

La figure 79 présente l'état des variables MODE et les signaux de passage par zéro associés aux signaux d'impulsions provenant de la commande pour la période de 0 s à 0,02 s. Les variables MODE représentent l'état des signaux d'impulsions. L'observation des variables MODE démontre que certaines impulsions ne sont pas traitées. En effet, par la comparaison des variables MODE avec les impulsions de la figure 75, il peut être observé que l'impulsion survenant au temps de 12,5 ms sur le signal d'impulsion 1 de la figure 75 n'est pas détectée par le compteur 0 de la carte 1 sur la figure 79. De plus, l'observation des signaux de passage par zéro à partir du temps de 12 ms sur la figure 79 montre que la simulation n'est plus fonctionnelle, car ces signaux sont incohérents. Grâce à la correction des impulsions, les signaux de passage par zéro ne devraient pas croiser le zéro .

La figure 79 présente aussi les TET de la simulation, qui sont multipliés par un gain de 25 000 afin d'être visualisés. La plage de variation des TET observée pour cette période est de 47 µs à 180 µs. D'après la figure 79, les TET de 47 µs sont obtenus lorsqu'il n'y a pas de traitement d'impulsion. Lors du traitement d'une impulsion, la plage de variation des TET est de 105 µs à 180 µs. Cette situation est inquiétante car le traitement d'une impulsion nécessite entre 58 µs et 133 µs. La figure 79 montre aussi que le TET moyen augmente considérablement à partir du temps de 12 ms, lorsque la simulation n'est plus fonctionnelle.

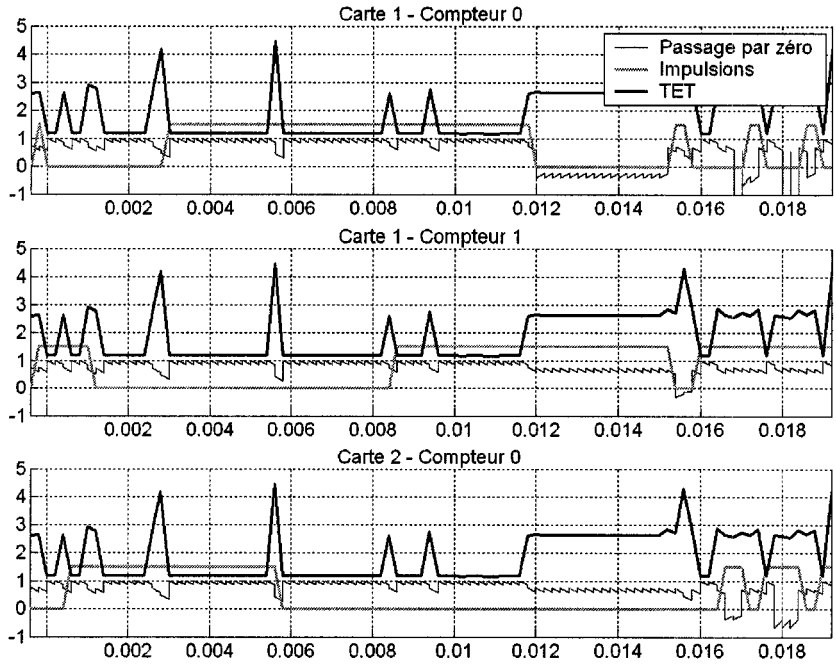


Figure 79 États des impulsions, passage par zéro et TET

Les résultats démontrent clairement que l'engin de calcul de la simulation n'est pas adapté correctement pour le traitement des signaux d'impulsions provenant de la commande. Une raison qui peut être à l'origine du problème est que l'engin de calcul exécute la boucle de simulation une seule fois par pas de calcul. Il n'est donc pas adapté

pour traiter des commutations multiples, c'est à dire lorsque plusieurs impulsions se produisent à l'intérieur d'un pas de calcul.

Par exemple, si l'engin de calcul n'a pas détecté d'événement durant le temps d'exécution, la simulation attend la fin du pas de calcul pour débuter un second pas de calcul. Comme le présente la figure 80, s'il survient plus d'une impulsion sur un même signal durant cette attente, seulement la dernière impulsion sera traitée par la logique mentionnée précedemment. Dans ce cas, la simulation est erronée car elle ne traite pas toutes les impulsions. Selon le comportement des résultats à partir du temps de 0,015 s pour une fréquence élevée des impulsions, il est très probable que cette situation soit à l'origine du problème.

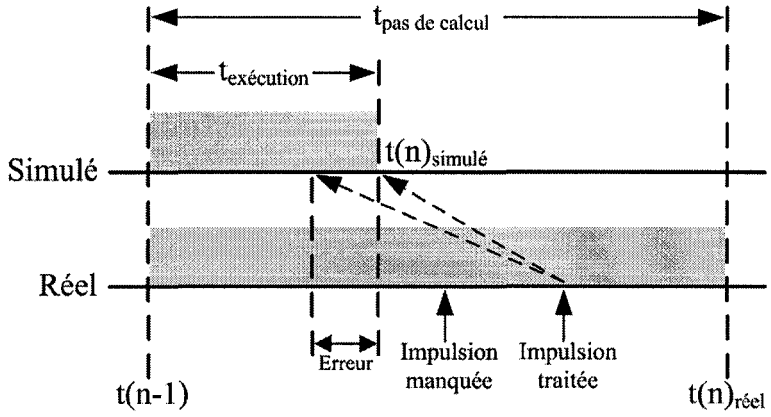


Figure 80 Impulsions multiples

Pour corriger ce problème, il faudrait modifier l'engin de calcul afin de permettre plusieurs passages dans la boucle de simulation. Cette solution permettrait de réduire le temps d'attente entre la fin du temps d'exécution et la fin du pas de calcul, comme présenté par la figure 81. La simulation pourrait ainsi détecter et traiter plus d'une impulsion dans le pas de calcul. Par contre, l'implémentation d'une telle solution n'est pas simple car l'engin devrait gérer le nombre de passages dans la boucle de simulation à

chaque pas de calcul. De plus, une autre problématique à considérer est que le temps d'exécution d'une boucle de simulation est variable, comme démontré par les TET obtenus à la figure 79.

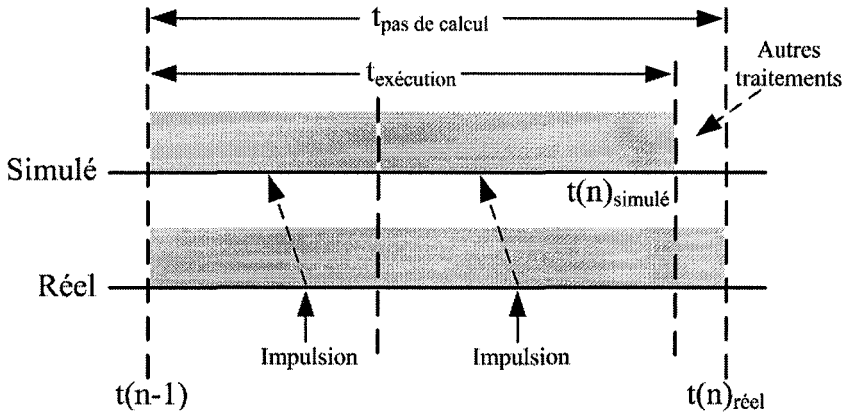


Figure 81 Réduction du temps d'attente

Comme le présente la figure 81, l'engin de calcul devrait aussi limiter la fin du temps d'exécution pour laisser un temps d'attente minimal nécessaire à la simulation pour effectuer d'autres traitements, telle que la sauvegarde des données des signaux. Dans le cas où des impulsions se produiraient dans ce temps d'attente minimal, la logique en place corrigerait la simulation avec une légère imprécision au début du pas de calcul suivant, en supposant que le risque que plusieurs impulsions surviennent sur un même signal dans cette période est très faible. Une meilleure solution serait de développer un pilote permettant de générer des interruptions pour les impulsions se produisant à l'intérieur de cette période, comme présenté par la figure 82. Cette solution assurerait le traitement de toutes les impulsions et permettrait de corriger la simulation avec plus de précision.

Le pilote par interruptions permettrait aussi de traiter les impulsions se produisant près de la fin du temps d'exécution, comme le présente la figure 82. Ces impulsions sont

problématiques car la simulation n'a pas le temps nécessaire pour les traiter avant la fin du temps d'exécution. Le pilote fournirait donc l'information nécessaire pour traiter ces impulsions avec précision au début du pas de calcul suivant.

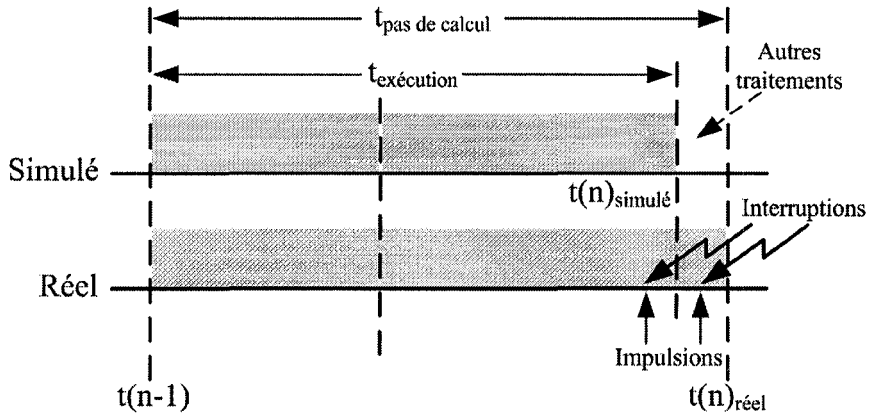


Figure 82 Traitement des impulsions par interruption

Finalement, un autre problème de la simulation est qu'elle ne traite pas les commutations simultanées, soit lorsque plusieurs événements se produisent au même instant. Par exemple, le changement d'état d'un signal d'impulsion entraîne le changement d'état d'un interrupteur dans le convertisseur, ce qui peut ensuite produire une chaîne de changements d'état des interrupteurs du convertisseur. Tous ces changements ce produisent au même instant dans la simulation. Le traitement d'une telle situation est problématique car le temps d'exécution est limité. Le traitement de plusieurs événements dans un même pas de calcul entraîne l'augmentation du temps d'exécution. La simulation devrait donc limiter le nombre d'événements à traiter dans cette situation, afin de ne pas excéder la limite du temps d'exécution.

5.6 Conclusion

Ce chapitre a présenté la démarche utilisée pour réaliser la simulation en temps réel avec commande externe des modèles d'entraînement électrique détaillés. Une première partie a introduit les caractéristiques de cette simulation et exposé les stratégies développées pour en faire la réalisation. Les parties suivantes ont présenté le développement du pilote pour les cartes PCI-MIO-16E-4, l'intégration du pilote dans le simulateur et les travaux effectués sur la génération et le traitement des signaux de passage par zéro. La dernière partie a présenté les résultats obtenus et les problématiques non résolues de la simulation en temps réel avec commande externe des modèles détaillés. Finalement, des solutions afin de pouvoir résoudre ces problématiques ont été proposées.

CONCLUSION

Les travaux effectués dans ce projet de recherche consistaient à réaliser une interface entre un simulateur en temps réel d'entraînements électriques et une commande externe. Les objectifs du projet étaient de réaliser l'interface et de valider la simulation en temps réel. La simulation en temps réel avec commande externe a été réalisée en deux parties. La première partie consistait à réaliser la simulation pour des modèles d'entraînements électriques à valeur moyenne et la deuxième partie consistait à réaliser la simulation pour des modèles d'entraînements électriques détaillés.

La première partie sur la simulation en temps réel avec commande externe des modèles à valeur moyenne avait comme principal objectif la familiarisation des outils de travail. Cette partie décrit en détails les deux modèles d'entraînements électriques à valeur moyenne utilisés ainsi que la démarche d'implémentation en temps réel. L'implémentation des parties de commande des modèles est réalisée avec la carte de prototypage DS1104 de la compagnie dSPACE. L'implémentation des parties de l'électronique de puissance et de la machine est réalisée avec xPC Target dans un ordinateur personnel munis des cartes d'E/S PCI-MIO-16E-4 de National Instruments. La comparaison des résultats obtenus avec les résultats de simulation en temps différé valide le fonctionnement de la simulation en temps réel avec commande externe des modèles à valeur moyenne. Cette première partie a permis d'atteindre l'objectif de familiarisation des outils matériels et logiciels utilisés pour la réalisation du projet. Elle a aussi permis de démontrer un potentiel intéressant pour la simulation de systèmes possédant de multiples entraînements électriques.

La deuxième partie, qui présente les caractéristiques particulières de la simulation en temps réel avec commande externe des modèles détaillés, introduit la nécessité de développer un pilote afin d'obtenir un fonctionnement spécifique des cartes d'E/S PCI-MIO-16E-4 pour la réception des signaux d'impulsions provenant de la commande.

Cette partie décrit les détails sur le développement et l'intégration du pilote, ainsi que les travaux effectués sur le traitement des signaux d'impulsions provenant de la commande par la génération et le traitement des signaux de passage par zéro. La comparaison des résultats obtenus pour la simulation en temps réel avec commande externe des modèles détaillés avec les résultats de simulation en temps différé démontre des lacunes dans le fonctionnement du simulateur.

Le projet ne s'est pas terminé dans sa totalité. Les objectifs de réaliser l'interface et de valider la simulation en temps réel avec commande externe pour les modèles détaillés n'ont pas été atteints. Plusieurs parties du projet, comme la simulation en temps réel avec commande externe des modèles à valeur moyenne, le développement du pilote et l'intégration du pilote dans le simulateur, ont nécessité beaucoup plus de temps qu'envisagé initialement. Les travaux sur le traitement des signaux de passage par zéro associés aux signaux d'impulsions provenant de la commande n'ont pas été complétés. Malgré tout, les travaux effectués dans ce projet de recherche constituent une bonne base pour la poursuite des travaux en considérant que le pilote pour les cartes PCI-MIO-16E-4 a été développé et intégré dans le simulateur, qu'une partie des travaux sur la génération et le traitement des signaux de passages par zéro a été réalisée, puis que les problèmes reliés à la simulation ont été identifiés.

RECOMMANDATIONS

Les résultats obtenus pour la simulation en temps réel avec commande externe des modèles détaillés démontrent clairement que l'engin de calcul de la simulation n'est pas adapté correctement pour le traitement des signaux d'impulsions provenant de la commande. Comme présenté à la fin de la section 5.5, des modifications devraient être réalisées dans l'engin de calcul afin de permettre le traitement des commutations multiples et des commutations simultanées.

La solution proposée pour le traitement des commutations multiples est de permettre plusieurs passages dans la boucle de simulation dans un même pas de calcul. Cette solution aurait pour effet de réduire le temps d'attente entre la fin du temps d'exécution et la fin du pas de calcul, pour ainsi permettre le traitement des impulsions qui se produisent après la fin du temps d'exécution dans la situation actuelle. Relativement à cette solution, pour traiter les impulsions se produisant trop près de la fin du temps d'exécution ou dans la période d'attente, la solution proposée est de développer un pilote permettant de générer des interruptions. Ce pilote permettrait de traiter ces impulsions avec précision au début du pas de calcul suivant.

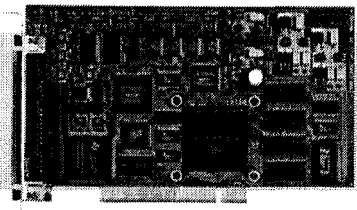
Pour les commutations simultanées, la solution proposée est de permettre le traitement des événements se produisant au même instant. Cependant, cette solution devrait limiter le nombre d'événements à traiter afin de ne pas excéder la limite du temps d'exécution.

ANNEXE 1

SPÉCIFICATIONS DE LA CARTE DS1104

DS1104 R&D Controller Board

Cost-effective system for controller development



Key Features

- **Single-board PCI hardware for use in PCs**
- **Set of intelligent I/O on-board**
- **Incremental encoder interface**
- **Serial interface (UART)**

Description

Application Areas

The real-time hardware based on PowerPC technology and its set of I/O interfaces make the controller board an ideal solution for developing controllers in various fields, such as drives, robotics, aerospace and automotives.

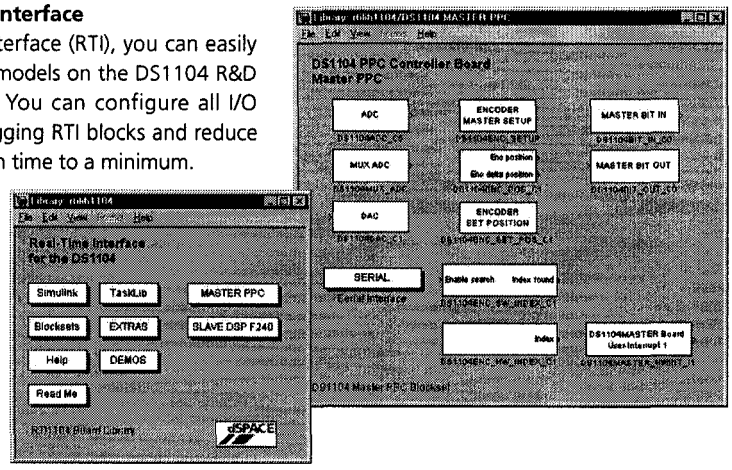
Key Benefits

The DS1104 upgrades your PC to a powerful development system for rapid control prototyping („R&D" stands for research & development). Real-Time Interface provides Simulink® blocks for graphical configuration of A/D, D/A, digital I/O lines, incremental encoder interface and PWM generation, for example. The board can be installed in virtually any PC with a free 5-V PCI slot.

Real-Time Interface

Using Real-Time Interface

With Real-Time Interface (RTI), you can easily run your function models on the DS1104 R&D Controller Board. You can configure all I/O graphically by dragging RTI blocks and reduce the implementation time to a minimum.



Technical Details

Parameter		Specification
Processor	PowerPC Type	■ PPC 603e
	CPU clock	■ 250 MHz
	Cache	■ 2 x 16 KB
Memory	Global memory	■ 32 MB SDRAM
	Flash memory	■ 8 MB
Timer	4 general-purpose timers	■ 32-bit down counter ■ Reload by hardware ■ 80-ns resolution
	1 sampling rate timer (decrementer)	■ 32-bit down counter ■ Reload by software ■ 40-ns resolution
	1 time base counter	■ 64-bit up counter ■ 40-ns resolution
Interrupt controller		■ 5 timer interrupts ■ 2 incremental encoder index line interrupts ■ 1 UART interrupt ■ 1 slave DSP interrupt ■ 1 slave DSP PWM interrupt ■ 5 A/D converter (end of conversion) interrupts ■ 1 host interrupt ■ 4 external interrupts (user interrupts)
A/D converter	Channels	■ 4 multiplexed channels equipped with one sample & hold A/D converters ■ 4 parallel channels each equipped with one sample & hold A/D converter
	Resolution	■ Multiplexed channels: 16 bit ■ Parallel channels: 12 bit
	Input voltage range	■ ±10 V
	Conversion time	■ Multiplexed channels: 2 µs [1] ■ Parallel channels: 800 ns [1]
	Offset error	■ ±5 mV
	Gain error	■ Multiplexed channels: ±0.25% ■ Parallel channels: ±0.5%ing
	Offset drift	■ 4 ppm/K
	Gain drift	■ 25 ppm/K
	Signal-to-noise ratio	■ Multiplexed channels: >80 dB ■ Parallel channels: >65 dB
D/A converter	Channels	■ 8 channels
	Resolution	■ 16-bit
	Output range	■ ±10 V
	Settling time	■ Max. 10 µs (full-scale, accuracy ½ LSB)
	Offset error	■ ±1 mV
	Gain error	■ ±0.1%
	Offset drift	■ 13 ppm/K
	Gain drift	■ 25 ppm/K
	Signal-to-noise ratio	■ >80 dB
	I_{max}	■ ±5 mA
Digital I/O	Channels	■ 20-bit parallel I/O ■ Single bit selectable for input or output
	Voltage range	■ TTL input/output levels
	I_{outmax}	■ ±5 mA

[1] Speed and timing specifications describe the capabilities of the hardware components and circuits of our products. Depending on the software complexity, the attainable overall performance figures can deviate significantly from the hardware specifications.

Parameter		Specification
Digital incremental encoder interface	Channels	■ 2 independent channels ■ Single-ended (TTL) or differential (RS422) input (software programmable for each channel)
	Position counters	■ 24-bit resolution ■ Max. 1.65 MHz input frequency, i.e., fourfold pulse count up to 6.6 MHz ■ Counter reset or reload via software
	Sensor supply voltage	■ 5 V/0.5 A
Serial interface	Configuration	■ Single UART (universal asynchronous receiver and transmitter) with FIFO ■ PLL-driven UART for accurate baud rate selection ■ RS232/RS422/RS485 compatibility
	Baud rate	■ Up to 115.2 Kbaud (RS232) ■ Up to 1 Mbaud (RS422/RS485)
Slave DSP	Type	■ Texas Instruments TMS320F240 DSP
	Clock rate	■ 20 MHz
	Memory	■ 64K x 16 external code memory ■ 28K x 16 external data memory ■ 4K x 16 dual-port memory for communication ■ 32 KB flash memory
	I/O channels	■ 10 PWM outputs ■ 4 capture inputs ■ 1 serial peripheral interface
	Input voltage range	■ TTL input/output level ■ A/D converter inputs: 0 ... 5 V
	Output current	■ Max. ±13 mA
Host interface		■ Requires one 33 MHz / 32-bit 5-V PCI slot
Physical characteristics	Physical size	■ 178 x 107 mm (7.0 x 4.2 in)
	Ambient temperature	■ 0 ... 55 °C (32 ... 131 °F)
	Cooling	■ Active cooling by fan
	Power consumption	■ 18.5 W
	Power supply	■ +5 V ±5%, 2.5 A ■ +12 V ±5%, 0.3 A ■ -12 V ±5%, 0.2 A

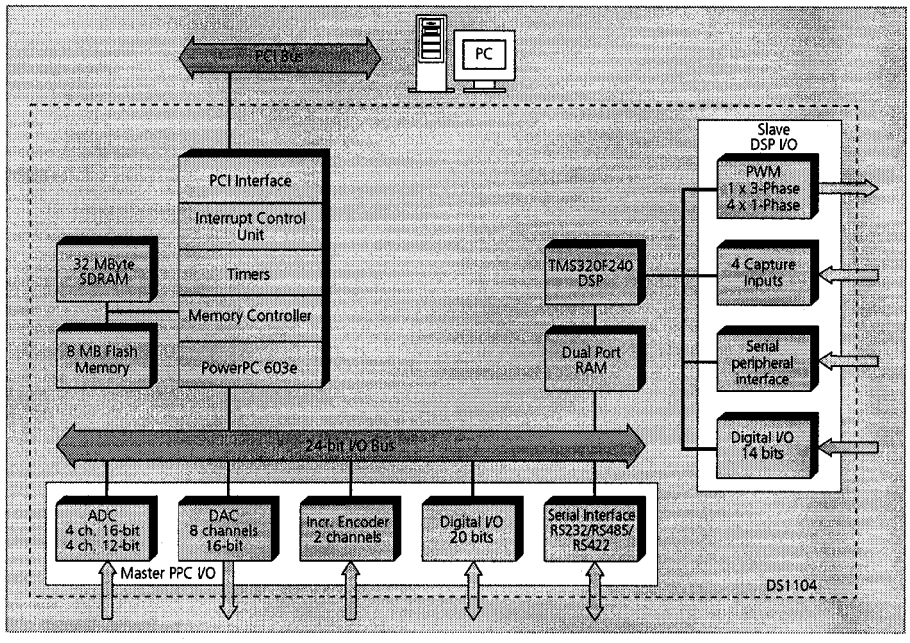
Order Information

Product	Order Number
DS1104 R&D Controller Board	■ DS1104

Relevant Software and Hardware

Software		Order Number
Included	■ DS1104 Real-Time Library	–
	■ Experiment and Platform Manager for hardware management	–
Required	■ Microtec C Compiler (p. 139)	■ CCPPPC
	■ Real-Time Interface (p. 120)	■ RTI
Optional	■ ControlDesk Standard – Operator Version (p. 140)	■ CS_O
	■ ControlDesk Standard – Developer Version (p. 140)	■ CS_D
	■ MLIB/MTRACE (p. 178)	■ MLIB/MTRACE
	■ CLIB (p. 177)	■ CLIB

Hardware		Order Number
Optional	■ Connector Panel (p. 260)	■ CP1104
	■ Combined Connector/LED Panel (p. 260)	■ CLP1104



Block Diagram

Induction Motor Control

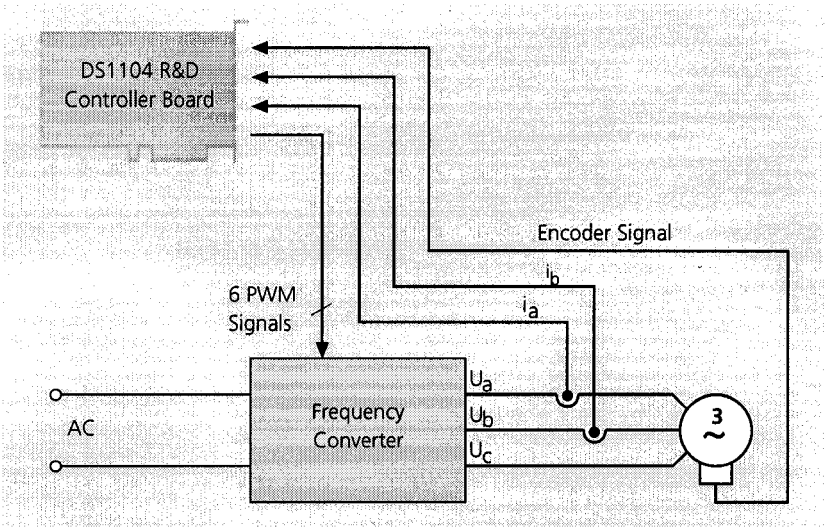
Use Case

Drive Control

In this use case, an induction motor controller is developed with the DS1104. The slave DSP system was designed for applications in drive control, and the PowerPC's calculation power supports convenient simulation and a smooth development process. In combination with Simulink, the board makes it easy to verify and optimize control algorithms and parameters.

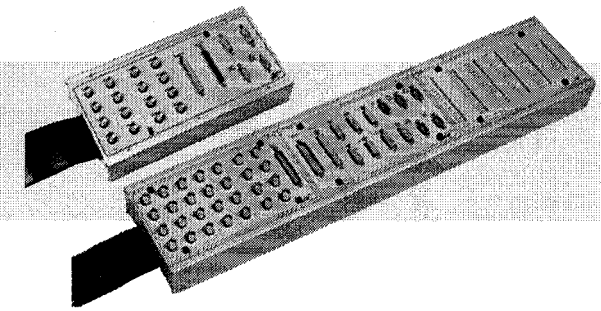
Determining Values

One of the board's incremental encoder interfaces picks up the encoder signal of the motor, while two A/D converters are required to analyze the motor currents. The controller board calculates the control algorithm on the basis of the measured values and determines the corresponding pulse width modulation (PWM). The three-phase PWM signals are generated on the board's DSP subsystem and determine the converter's output voltage and frequency.



Connector and LED Panels

Easy access to I/O signals with BNC and Sub-D connectors



Key Features

- **Access analog signals via BNC connectors**
- **Access digital signals via Sub-D connectors**
- **Sub-D connectors are low-density and grouped according to I/O channels or functional units**
- **LED panels indicate the status of the board's digital signals**

Overview

			Connector Panel	Connector/ LED Combi Panel
Board	BNC Connectors	Sub-D Connectors	Width	Width
DS1103	■ 20 ADC inputs ■ 8 DAC outputs	■ Digital I/O ■ Slave DSP I/O ■ Incremental encoder interfaces ■ CAN interface ■ Serial interfaces	14 MU[1]	21 MU[1]
DS1104	■ 8 ADC inputs ■ 8 DAC outputs	■ Digital I/O ■ Slave DSP I/O ■ Incremental encoder interfaces ■ Serial interfaces	9 MU[1]	14 MU[1]

[1] MU = measurement unit (1.2" / 30.5 mm)

Order Information

Products	Order Number
Connector Panel for DS1103 PPC Controller Board	■ CP1103
Connector Panel for DS1104 R&D Controller Board	■ CP1104
Connector/LED Combi Panel for DS1103 PPC Controller Board	■ CLP1103
Connector/LED Combi Panel for DS1104 R&D Controller Board	■ CLP1104

ANNEXE 2

SPÉCIFICATIONS DE LA CARTE PCI-MIO-16E-4

NI 6040E Family Specifications

This document lists the I/O terminal summary and specifications for the devices that make up the NI 6040E family of devices. This family includes the following devices:

- NI PXI-6040E
- NI PCI-MIO-16E-4 (NI 6040E)

For the most current edition of this document, refer to `ni.com/manuals`. For more information about using your E Series device, refer to the *E Series Help* at `ni.com/manuals` or on your NI-DAQ CD. Refer to the *DAQ Quick Start Guide* for more information about accessing documents on the NI-DAQ CD.

Note With NI-DAQmx, National Instruments has revised its terminal names so they are easier to understand and more consistent among NI hardware and software products. The revised terminal names used in this document are usually similar to the names they replace. For a complete list of Traditional NI-DAQ terminal names and their NI-DAQmx equivalents, refer to the *Terminal Name Equivalents* table in the *E Series Help*.

Table 1. I/O Terminal Summary

Terminal Name	Terminal Type and Direction	Impedance Input/ Output	Protection (Volts) On/Off	Source (mA at V)	Sink (mA at V)	Rise Time (ns)	Bias
AI <0..15>	AI	100 GΩ in parallel with 100 pF	25/15	—	—	—	±200 pA
AI SENSE	AI	100 GΩ in parallel with 100 pF	25/15	—	—	—	±200 pA
AI GND	—	—	—	—	—	—	—
AO 0	AO	0.1 Ω	Short-circuit to ground	5 at 10	5 at –10	20 V/µs	—

Table 1. I/O Terminal Summary (Continued)

Terminal Name	Terminal Type and Direction	Impedance Input/ Output	Protection (Volts) On/Off	Source (mA at V)	Sink (mA at V)	Rise Time (ns)	Bias
AO 1	AO	0.1 Ω	Short-circuit to ground	5 at 10	5 at –10	20 V/µs	—
AO EXT REF	AI	10 kΩ	25/15	—	—	—	—
AO GND	—	—	—	—	—	—	—
D GND	—	—	—	—	—	—	—
+5 V	—	0.1 Ω	Short-circuit to ground	1 A	—	—	—
P0.<0..7>	DIO	—	V_{CC} + 0.5	13 at (V_{CC} – 0.4)	24 at 0.4	1.1	50 kΩ pu
AI HOLD COMP	DO	—	—	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
EXTSTROBE*	DO	—	—	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 0/(AI START TRIG)	AI/DIO	10 kΩ	V_{CC} + 0.5/±35	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	9 kΩ pu, 10 kΩ pd
PFI 1/(AI REF TRIG)	DIO	—	V_{CC} +0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 2/(AI CONV CLK)*	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 3/ CTR 1 SOURCE	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 4/CTR 1 GATE	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
CTR 1 OUT	DO	—	—	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 5/(AO SAMP CLK)*	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 6/(AO START TRIG)	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 7/(AI SAMP CLK)	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 8/ CTR 0 SOURCE	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
PFI 9/CTR 0 GATE	DIO	—	V_{CC} + 0.5	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu

Table 1. I/O Terminal Summary (Continued)

Terminal Name	Terminal Type and Direction	Impedance Input/ Output	Protection (Volts) On/Off	Source (mA at V)	Sink (mA at V)	Rise Time (ns)	Bias
CTR 0 OUT	DO	—	—	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu
FREQ OUT	DO	—	—	3.5 at (V_{CC} – 0.4)	5 at 0.4	1.5	50 kΩ pu

* Indicates active low

AI = Analog Input DIO = Digital Input/Output pd = pull-down
AO = Analog Output DO = Digital Output pu = pull-up
AI/DIO = Analog Input/Digital Input/Output

Note: The tolerance on the 50 kΩ pull-up and pull-down resistors is large. Actual value might range between 17 kΩ and 100 kΩ.

Specifications

The following specifications are typical at 25 °C unless otherwise noted.

Analog Input

Input Characteristics

Number of channels ..... 16 single-ended or 8 differential (software-selectable per channel)

Type of A/D converter (ADC) ..... Successive approximation

Resolution ..... 12 bits, 1 in 4,096

Maximum sampling rate

Single-channel scanning ..... 500 kS/s

Multiple-channel scanning ..... 250 kS/s

Input signal ranges

Range (Software-Selectable)	Input Range	
	Bipolar	Unipolar
20 V	±10 V	—
10 V	±5 V	0 to 10 V
5 V	±2.5 V	0 to 5 V
2 V	±1 V	0 to 2 V

Range (Software-Selectable)	Input Range	
	Bipolar	Unipolar
1 V	±500 mV	0 to 1 V
500 mV	±250 mV	0 to 500 mV
200 mV	±100 mV	0 to 200 mV
100 mV	±50 mV	0 to 100 mV

Input coupling ..... DC

Maximum working voltage (signal and common-mode) ..... Each input should remain within ±11 V of ground

Overvoltage protection

Powered on ..... ±25 V

Powered off ..... ±15 V

Inputs protected ..... AI <0..15>, AI SENSE

FIFO buffer size ..... 512 samples (S)

DMA

Channels 3

Data sources/destinations ..... Analog input, analog output, counter/timer 0, or counter/timer 1

Data transfers ..... Direct memory access (DMA), interrupts, programmed I/O

DMA modes ..... Scatter-gather (single-transfer, demand-transfer)

Configuration memory size ..... 512 words (1 word = 8 bits)

Accuracy Information

Nominal Range (V)	Absolute Accuracy							Relative Accuracy	
	% of Reading		Offset (mV)	Noise + Quantization (mV)		Temp Drift (%/°C)	Absolute Accuracy at Full Scale (mv)	Resolution (mV)	
	24 Hours	1 Year		Single Pt.	Averaged			Single Pt.	Averaged
±10	0.0672	0.0714	7.38	4.64	0.846	0.0010	15.373	6.27	1.11
±5	0.0272	0.0314	3.70	2.32	0.423	0.0005	5.697	3.14	0.557
±2.5	0.0672	0.0714	1.86	1.16	0.211	0.0010	3.859	1.57	0.278
±1	0.0672	0.0714	0.757	0.464	0.085	0.0010	1.556	0.627	0.111
±0.5	0.0672	0.0714	0.389	0.269	0.042	0.0010	0.789	.0339	0.056
±0.25	0.0672	0.0714	0.205	0.134	0.021	0.0010	0.405	0.169	0.028
±0.1	0.0672	0.0714	0.095	0.076	0.010	0.0010	0.176	0.088	0.013
±0.05	0.0672	0.0714	0.058	0.056	0.006	0.0010	0.100	0.064	0.008
0 to 10	0.0272	0.0314	3.70	2.32	0.423	0.0005	7.269	3.14	0.557
0 to 5	0.0672	0.0714	1.86	1.16	0.211	0.0010	5.645	1.57	0.278
0 to 2	0.0672	0.0714	0.757	0.464	0.085	0.0010	2.271	0.627	0.111
0 to 1	0.0672	0.0714	0.389	0.269	0.042	0.0010	1.146	0.339	0.056
0 to 0.5	0.0672	0.0714	0.205	0.134	0.021	0.0010	0.583	0.169	0.028
0 to 0.2	0.0672	0.0714	0.095	0.076	0.010	0.0010	0.247	0.088	0.013
0 to 0.1	0.0672	0.0714	0.058	0.056	0.006	0.0010	0.135	0.064	0.008

Absolute Accuracy = (% of Reading × Voltage) + Offset + Noise + (Temp Drift × Voltage)

Note: Temp Drift applies only if ambient is greater than ±10 °C of previous external calibration. Accuracies are valid for measurements following an internal E Series Calibration. Averaged numbers assume dithering and averaging of 100 single-channel readings. Measurement accuracies are listed for operational temperatures within ±1 °C of internal calibration temperature and ±10 °C of external or factory calibration temperature. NI recommends a one-year calibration interval. The Absolute Accuracy at Full Scale calculations were performed for a maximum range input voltage (for example, 10 V for the ±10 V range) after one year, assuming 100 pt averaging of data.

Transfer Characteristics

Relative accuracy

Dithered ..... ±0.5 least significant bits (LSB) typ

Undithered ..... ±1.5 LSB max

Differential nonlinearity (DNL)..... ±0.5 LSB typ, ±1 LSB max

No missing codes ..... 12 bits, guaranteed

Offset error

Pregain error after calibration.....±16 µV max

Pregain error before calibration.....±4.0 mV max

Postgain error after calibration.....±0.8 mV max

Postgain error before calibration .....±200 mV max

Gain error (relative to calibration reference)

After calibration (gain = 1).....±0.02% of reading max

Before calibration.....±2.5% of reading max

Gain ≠ 1 with gain error adjusted to 0 at gain = 1.....±0.02% of reading max

Amplifier Characteristics

Input impedance

Normal powered on .....100 GΩ in parallel with 100 pF

Powered off .....820 Ω min

Overload .....820 Ω min

Input bias current .....±200 pA

Input offset current .....±100 pA

CMRR, all input ranges, DC to 60 Hz

Range	CMRR
10 to 20 V	85 dB
5 V	95 dB
100 mV to 2 V	100 dB

Dynamic Characteristics

Bandwidth

Small Signal (–3 dB) .....600 kHz

Large Signal (1% THD) .....350 kHz

Settling time to full-scale step

Range	Accuracy[1]		
	±0.012% (±0.5 LSB)	±0.024% (±1 LSB)	±0.098% (±4 LSB)
All	4 µS typ, 8 µS max	4 µS max	4 µS max

[1] Accuracy values valid for source impedances <1 kΩ.

System noise (LSB_{rms}, not including quantization)

Range	Dither Off	Dither On
1 to 20 V	0.2	0.5
500 mV	0.25	0.5
200 mV	0.5	0.7
100 mV	0.9	1.0

Crosstalk (DC to 100 kHz)

Adjacent channels ..... –75 dB

All other channels ..... –90 dB

Stability

Offset temperature coefficient

Pregain ..... ±5 µV/°C

Postgain ..... ±240 µV/°C

Gain temperature coefficient ..... ±20 ppm/°C

Analog Output

Output Characteristics

Number of channels ..... 2 voltage

Resolution ..... 12 bits, 1 in 4,096

Max update rate

Waveform Generation			
FIFO Mode		Non-FIFO Mode	
Internally Timed	Externally Timed	1 Channel	2 Channels
1 MS/s	950 kS/s	800 kS/s, system-dependent	400 kS/s, system-dependent

Type of D/A converter (DAC) ..... Double-buffered, multiplying

FIFO buffer size ..... 512 Samples (S)

Data transfers ..... DMA, interrupts, programmed I/O

DMA modes ..... Scatter-gather (single-transfer, demand-transfer)

Accuracy Information

Nominal Range (V)		Absolute Accuracy					
		% of Reading			Offset	Temp Drift	Absolute Accuracy at Full Scale (mV)
Positive Full Scale	Negative Full Scale	24 Hours	90 Days	1 Year	(mV)	(%/°C)	
10	–10	0.0177	0.0197	0.0219	5.93	0.0005	8.127
10	0	0.0177	0.0197	0.0219	3.49	0.0005	5.685

Absolute Accuracy = (% of Reading × Voltage) + Offset + (Temp Drift × Voltage)

Note: Temp drift applies only if ambient is greater than ±10 °C of previous external calibration.

Transfer Characteristics

Relative accuracy, or integral nonlinearity (INL)

After calibration ..... ±0.3 LSB typ, ±0.5 LSB max

Before calibration ..... ±4 LSB max

DNL

After calibration ..... ±0.3 LSB typ, ±1.0 LSB max

Before calibration ..... ±3 LSB max

Monotonicity ..... 12 bits, guaranteed after calibration

Offset error
- After calibration ..... ±1.0 mV max
- Before calibration ..... ±200 mV max

Gain error (relative to internal reference)
- After calibration ..... ±0.01% of output max
- Before calibration ..... ±0.5% of output max

Gain error
(relative to external reference) ..... 0 to 0.67% of output max, not adjustable

Voltage Output

Ranges ..... ±10 V, 0 to 10 V, ±AO EXT REF, 0 to AO EXT REF (software-selectable)

Output coupling ..... DC

Output impedance ..... 0.1 Ω max

Current drive ..... ±5 mA max

Protection ..... Short-circuit to ground

Power-on state ..... 0 V (± 200 mV)

External reference input
- Range ..... ±11 V
- Overvoltage protection
  - Powered on ..... ±25 V
  - Powered off ..... ±15 V
- Input impedance ..... 10 kΩ
- Bandwidth (–3 dB) ..... 1 MHz

Dynamic Characteristics

Settling time for full-scale step ..... 3 µs to ±0.5 LSB accuracy[1]

Slew rate ..... 20 V/µs

Noise ..... 200 μV_{rms}, DC to 1 MHz

[1] Accuracy values are valid for source impedances <1 kΩ. Refer to the *Multichannel Scanning Considerations* section in the *E Series Help* for more information.

Glitch energy (at mid-scale transition)
- Reglitching disabled ..... ±20 mV
- Reglitching enabled ..... ±4 mV
- Duration ..... 1.5 µs

Stability

Offset temperature coefficient ..... ±50 µV/°C

Gain temperature coefficient
- Internal reference ..... ±25 ppm/°C
- External reference ..... ±25 ppm/°C

Digital I/O

Number of channels ..... 8 input/output

Compatibility ..... 5 V TTL

Digital logic levels on P0.<0..7>

Level	Min	Max
Input low voltage	0 V	0.8 V
Input high voltage	2 V	5.0 V
Input low current (V_{in} = 0 V)	—	–320 µA
Input high current (V_{in} = 5 V)	—	10 µA
Output low voltage (I_{OL} = 24 mA)	—	0.4 V
Output high voltage (I_{OH} = –13 mA)	4.35 V	—

Power-on state ..... Input (high-impedance)

Data transfers ..... Programmed I/O

Transfer rate (1 word = 8 bits)
- Maximum with NI-DAQ, system-dependent ..... 50 kwords/s

Constant sustainable rate ..... 1 to 10 kwords/s, typ

Timing I/O

Number of channels ..... 2 up/down counter/timers, 1 frequency scaler

Resolution
- Counter/timers ..... 24 bits
- Frequency scaler ..... 4 bits

Compatibility ..... 5 V TTL/CMOS

Base clocks available
- Counter/timers ..... 20 MHz, 100 kHz
- Frequency scaler ..... 10 MHz, 100 kHz

Base clock accuracy ..... ±0.01%

Max source frequency
- Up/down counter/timers ..... 20 MHz

Min source pulse duration ..... 10 ns

Min gate pulse duration ..... 10 ns, edge-detect mode

Data transfers ..... DMA, interrupts, programmed I/O

DMA modes ..... Scatter-gather (single-transfer, demand-transfer)

Triggers

Analog Trigger

Source ..... AI <0..15>, external trigger (PFI 0/AI START TRIG)

Purpose
- Analog Input ..... Start, reference, and pause trigger, sample clock
- Analog Output ..... Start and pause trigger, sample clock
- Counter/timers ..... Source, gate

Level
- Internal ..... ±Full-scale
- External ..... ±10 V

Slope ..... Positive or negative (software-selectable)

Resolution ..... 8 bits, 1 in 256

Hysteresis ..... Programmable

Bandwidth (–3 dB) ..... 650 kHz, internal; 3 MHz, external

External input (PFI 0/AI START TRIG)
- Impedance ..... 10 kΩ
- Coupling ..... DC
- Protection
  - When configured as a digital signal ..... –0.5 to V_{CC} + 0.5 V
  - When configured as an analog trigger signal or disabled ..... ±35 V
  - Powered off ..... ±35 V

Digital Trigger

Purpose
- Analog Input ..... Start, reference, and pause trigger, sample clock
- Analog Output ..... Start and pause trigger, sample clock
- Counter/timers ..... Source, gate

External sources ..... PFI <0..9>, RTSI <0..6>

Compatibility ..... 5 V TTL

Response ..... Rising or falling edge

Pulse width ..... 10 ns min

RTSI Bus (PCI Only)

Trigger lines 7

PXI Trigger Bus (PXI Only)

- Trigger lines 6
- Star trigger 1

Calibration

Recommended warm-up time ..... 15 minutes

Calibration interval ..... 1 year

External calibration reference ..... >6 and <10 V

Onboard calibration reference

DC level ..... 5.000 V (±3.5 mV), over full operating temperature, actual value stored in EEPROM

Temperature coefficient ..... ±5 ppm/°C max

Long-term stability ..... ±15 ppm/$\sqrt{1{,}000\ \text{h}}$

Bus Interface

Type ..... Master, slave

Power

Bus Requirement

+5 VDC (±5%) ..... 1.0 A

Note Excludes power consumed through +5 V available at the I/O connector.

I/O Connector Power

Power available at I/O connector ..... +4.65 to +5.25 VDC at 1 A

Physical

Dimensions (not including connectors)

NI PXI-6040E ..... 16 by 10 cm (6.3 by 3.9 in.)

NI PCI-MIO-16E-4 ..... 17.5 by 10.7 cm (6.9 by 4.2 in.)

I/O connector ..... 68-pin male 0.050 D-type

Maximum Working Voltage

Maximum working voltage refers to the signal voltage plus the common-mode voltage.

Channel-to-earth ..... 42 V, Installation Category II

Channel-to-channel ..... 42 V, Installation Category II

Environmental

Operating temperature ..... 0 to 55 °C

Storage temperature ..... –20 to 70°C

Relative humidity ..... 10 to 90%, noncondensing

Maximum altitude ..... 2,000 m

Pollution Degree (indoor use only) 2

Safety

The NI 6040E devices meet the requirements of the following standards for safety and electrical equipment for measurement, control, and laboratory use:

- IEC 61010-1, EN 61010-1
- UL 3111-1, UL 61010B-1
- CAN/CSA C22.2 No. 1010.1

Note For UL and other safety certifications, refer to the product label, or visit `ni.com/hardref.nsf`, search by model number or product line, and click the appropriate link in the Certification column.

Electromagnetic Compatibility

Emissions ..... EN 55011 Class A at 10 m
FCC Part 15A above 1 GHz

Immunity ..... EN 61326:1997
A2:2001, Table 1

CE, C-Tick, and FCC Part 15 (Class A) Compliant

Note For EMC compliance, operate this device with shielded cabling.

CE Compliance

This product meets the essential requirements of applicable European Directives, as amended for CE marking, as follows:

Low-Voltage Directive (safety) 73/23/EEC

Electromagnetic Compatibility
Directive (EMC) 89/336/EEC

Note Refer to the Declaration of Conformity (DoC) for this product for any additional regulatory compliance information. To obtain the DoC for this product, visit `ni.com/hardref.nsf`, search by model number or product line, and click the appropriate link in the Certification column.

370722B-01 Sep03

ANNEXE 3

DESCRIPTION DES BITS DES REGISTRES DE LA CARTE PCI-MIO-16E-4

Note: Dans les tableaux, l'indice *i* représente l'indice d'un compteur (0 ou 1) et l'indice *I* représente l'indice de l'autre compteur. Par exemple, si *i*=0, alors *I*=1.

Tableau VII

G*i*_Command_Register

Bit	Nom	Fonction
15	G*I*_Disarm_Copy	Le bit désarme le compteur *I* lorsqu'il est mis à 1. Le bit est remis à 0 automatiquement.
14	G*I*_Save_Trace_Copy	Le bit fait la sauvegarde du compte du compteur *I* dans le registre G*I*_Save_Registers s'il est mis à 1. Ce bit et le bit G*I*_Save_Trace doivent être à 0 pour que le registre de sauvegarde suive le compteur.
13	G*I*_Arm_Copy	Le bit arme le compteur *I* lorsqu'il est mis à 1. Le bit est maintenu à 1 et le compteur est armé tant que le compteur n'est pas désarmé, soit de façon matérielle ou par le bit G*I*_Disarm.
12	G*i*_Bank_Switch_Enable	Si le compteur *i* n'est pas armé, le bit sélectionne la banque de registre pour l'écriture: 0: Banque X 1: Banque Y Si le compteur *i* est armé, le bit met en fonction le changement de banque s'il est mis à 1.
11	G*i*_Bank_Switch_Mode	Le bit sélectionne la source qui contrôle le changement de banque de registre de chargement, si le changement de banque est en fonction: 0: G*i*_GATE 1: Logiciel
10	G*i*_Bank_Switch_Start	Le bit indique le changement de banque de registre de chargement sur la condition sélectionnée avec le bit G*i*_Bank_Switch_Mode lorsqu'il est mis à 1. Le bit est remis à 0 automatiquement.
9	G*i*_Little_Big_Indian	Le bit sélectionne la partie du registre de chargement ou de sauvegarde utilisée pour la reconnaissance automatique d'interruption: 0: Partie basse du registre 1: Partie haute du registre
8	G*i*_Synchronized_Gate	Le bit met en fonction la synchronisation du signal G*i*_GATE avec le signal G*i*_SOURCE lorsqu'il est mis à 1.

Tableau VII (suite)

Bit	Nom	Fonction
7	G*i*_Write_Switch	Le bit met en fonction le changement d'écriture dans les registres de chargement du compteur *i*. L'écriture dans le registre A est: 0: Inconditionnellement dirigée vers le registre A 1: Dirigée vers le registre de chargement inactif
6-5	G*i*_Up_Down	Les bits sélectionnent le mode du compte: 0: Sélection logicielle du compte en décrémentation 1: Sélection logicielle du compte en incrémentation 2: Sélection matérielle du compte contrôlée par le signal G*i*_UP_DOWN. Décrémentation pour un niveau bas et incrémentation pour un niveau haut. 3:Sélection matérielle du compte contrôlée par la valeur interne du bit G*i*_Gate_Polarity. Décrémentation pour un niveau actif du signal G*i*_GATE et incrémentation pour un niveau inactif du signal G*i*_GATE.
4	G*i*_Disarm	Le bit désarme le compteur *i* lorsqu'il est mis à 1. Le bit est remis à 0 automatiquement.
3	G*i*_Analog_Trigger_Reset	Le bit remet à 0 les registres d'hystérésis dans le circuit analogique du déclenchement. Le bit doit être mis à 1 lors de l'armement du compteur *i* pour utiliser le déclenchement analogique en mode hystérésis. Le bit est remis à 0 automatiquement.
2	G*i*_Load	Le bit charge le contenu du registre de chargement sélectionné dans le compteur *i* lorsqu'il est mis à 1. Le bit est remis à 0 automatiquement.
1	G*i*_Save_Trace	Le bit fait la sauvegarde du compte du compteur *i* dans le registre G*i*_Save_Registers s'il est mis à 1. Ce bit et le bit G*i*_Save_Trace_Copy doivent être à 0 pour que le registre de sauvegarde suive le compteur.
0	G*i*_Arm	Le bit arme le compteur *i* lorsqu'il est mis à 1. Le bit est maintenu à 1 et le compteur est armé tant que le compteur n'est pas désarmé, soit de façon matérielle ou par le bit G*i*_Disarm.

Tableau VIII

G*i*_Input_Select_Register

Bit	Nom	Fonction
15	G*i*_Source_Polarity	Le bit sélectionne le front actif du signal G*i*_SOURCE du compteur *i*: 0: Front montant 1: Front descendant
14	G*i*_Output_Polarity	Le bit sélectionne la polarité du signal d'impulsion G*i*_OUT en mode TC ou le niveau initial du signal G*i*_OUT en mode bascule de la sortie. Le mode est sélectionné avec le bit G*i*_Output_Mode. 0: Impulsion active haute ou niveau initial bas 1: Impulsion active basse ou niveau initial haut
13	G*i*_OR_Gate	Le bit détermine s'il y a un OU logique entre le signal G*i*_GATE et le signal G*I*_OUT: 0: Non 1: Oui
12	G*i*_Gate_Select_Load_Source	Le bit met en fonction la sélection du registre de chargement par le signal G*i*_GATE lorsqu'il est mis à 1. Lorsque la fonction est activé, le registre A est sélectionné pour un niveau actif et le registre B est sélectionné pour un niveau inactif. De plus, le bit G*i*_Reload_Source_Switching est ignoré. Cette fonction peut seulement être utilisée lorsque le bit G*i*_Gating_Mode est en mode niveau.
11-7	G*i*_Gate_Select	Les bits sélectionnent la source du signal G*i*_GATE: 1-10: PFI<0..9> 11-17: RTSI_TRIGGER<0..6> 18: Le signal d'entrée analogique interne START2 19: Le signal de sortie analogique interne UI2_TC 20: Le signal G*I*_TC 21: Le signal d'entrée analogique interne START1 31: Logique basse

Tableau VIII (suite)

Bit	Nom	Fonction
6-2	G*i*_Source_Select	Les bits sélectionnent la source du signal G*i*_SOURCE: 0: Le signal interne G_IN_TIMEBASE1 1-10: PFI<0..9> 11-17: RTSI_TRIGGER<0..6> 18: Le signal interne IN_TIMEBASE2 19: Le signal G*I*_TC 31: Logique basse
1	G*i*_Write_Acknowledges_Irq	Le bit cause un accès d'écriture dans les registres de chargement pour remettre à 0 le bit G*i*_TC_St et l'interruption associée de détection de l'erreur de latence du circuit lorsqu'il est mis à 1. Le bit ne doit pas être mis à 1 si le bit G*i*_Read_Acknowledges_Irq est à 1.
0	G*i*_Read_Acknowledges_Irq	Le bit cause un accès dans les registres de sauvegarde matérielle pour remettre à 0 le bit G*i*_Gate_Interrupt_St et l'interruption associée de détection de l'erreur de latence du circuit lorsqu'il est mis à 1. Le bit ne doit pas être mis à 1 si le bit G*i*_Write_Acknowledges_Irq est à 1.

Tableau IX

G*i*_Mode_register

Bit	Nom	Fonction
15	G*i*_Reload_Source_Switching	Si le bit G*i*_Gate_Select_Load_Source est mis à 0, le bit met en fonction la sélection du registre de chargement de la façon suivante: 0: Utilise toujours le même registre de chargement 1: Alterne entre les deux registres de chargement
14	G*i*_Loading_On_Gate	Le bit détermine si le signal G*i*_GATE cause le chargement du compteur: 0: Le signal ne cause pas le chargement du compteur 1: Le chargement du compteur s'effectue sur le front actif du signal qui arrête le compteur, à moins que le mode front soit utilisé et que G*i*_Trigger_Mode_For_Edge_Gate soit mis à 3. Dans ce cas, le compteur est chargé sur tous les fronts actifs du signal.
13	G*i*_Gate_Polarity	Le bit sélectionne la polarité du signal G*i*_GATE: 0: Actif haut 1: Actif bas
12	G*i*_Loading_On_TC	Le bit détermine le comportement du compteur à la fin du compte (TC): 0: Recommence à compter à 0 à la fin du compte 1: Chargement du compteur à la fin du compte
11-10	G*i*_Counting_Once	Les bits déterminent si le compteur est désarmé de façon matérielle quand le compteur est arrêté à cause d'une condition matérielle: 0: Pas de désarmement matériel 1: Désarmement à la fin du compte (TC) qui arrête le compte 2: Désarmement sur le signal G*i*_GATE qui arrête le compte 3: Désarmement à la fin du compte ou sur le signal G*i*_GATE qui arrête le compte, peu importe lequel arrive en premier

Tableau IX (suite)

Bit	Nom	Fonction
9-8	G*i*_Output_Mode	Les bits sélectionnent le mode du signal G*i*_OUT: 0: Réservé 1: Mode TC. Le signal de fin du compte (TC) apparaît sur le signal G*i*_OUT 2: Mode alterne sur TC ou G*i*_GATE. Le signal G*i*_OUT change d'état sur les fronts du signal TC ou sur les fronts actifs du signal G*i*_GATE
7	G*i*_Load_Source_Select	Si le compteur *i* est désarmé, le bit sélectionne le registre initial de chargement: 0: Registre de chargement A 1: Registre de chargement B
6-5	G*i*_Stop_Mode	Les bits sélectionnent la condition pour laquelle le compteur arrête: 0: Arrête sur une condition du signal G*i*_GATE 1: Arrête sur une condition du signal G*i*_GATE ou sur le premier TC, peu importe lequel arrive en premier 2: Arrête sur une condition du signal G*i*_GATE ou sur le second TC, peu importe lequel arrive en premier 3: Réservé Peu importe la configuration de ce registre, le compteur peut toujours être arrêté de façon logicielle par le bit G*i*_Disarm. La condition du signal G*i*_GATE qui arrête le compteur est déterminée par G*i*_Gating_Mode dans le cas de mode niveau, ou par la combinaison de G*i*_Gating_Mode et de G*i*_Trigger_Mode_For_Edge_Gate dans le cas de mode front. Les sélections 1 et 2 sont valides seulement si G*i*_Trigger_Mode_For_Edge_Gate est mis à 2.

Tableau IX (suite)

Bit	Nom	Fonction
4-3	G*i*_Trigger_Mode_For_Edge_Gate	Les bits sélectionnent le mode du déclenchement, si Gi_Gating_Mode est mis à 0: 0: Le premier front du signal G*i*_GATE démarre le compte, le second front arrête le compte 1: Le premier front du signal G*i*_GATE arrête le compte, le second front démarre le compte 2: Un front du signal G*i*_GATE démarre le compte à moins que le compte soit déjà démarré, dans ce cas le front est ignoré. Les sélections valides pour G*i*_Stop_Mode dans ce cas sont 1 et 2, mais seulement le signal TC arrête le compte. 3: Le signal G*i*_GATE est utilisé seulement pour le chargement et la sauvegarde. Il ne peut arrêter le compte. Les sélections 0 et 1 sont valides seulement si G*i*_Stop_Mode est mis à 0. Les sélections 0, 1 et 2 sont valides seulement si G*i*_Gating_Mode est mis à 2 ou 3. La sélection 3 est valide seulement si G*i*_Gating_Mode est différent de 0.
2	G*i*_Gate_On_Both_Edges	Le bit met en fonction l'utilisation des deux fronts (montant et descendant) du signal G*i*_GATE pour générer des interruptions et/ou contrôler le compteur s'il est mis à 1.
1-0	G*i*_Gating_Mode	Les bits mettent en fonction et sélectionnent le mode du signal G*i*_GATE: 0: Le mode du signal G*i*_GATE n'est pas en fonction 1: Mode niveau 2: Mode front. Front montant si G*i*_Gate_Polarity est mis à 0 et front descendant si G*i*_Gate_Polarity est mis à 1. 3: Mode front. Front montant si G*i*_Gate_Polarity est mis à 1et front descendant si G*i*_Gate_Polarity est mis à 0. Si G*i*_Gating_Mode est mis à 0, le niveau du signal G*i*_GATE est disponible seulement pour la sélection du sens du compte.

Tableau X

Interrupt_A_Enable_Register

Bit	Nom	Fonction
15-10	Reserved	
9	Pass_Thru_0_Interrupt_Enable	Le bit met en fonction l'interruption pour le signal Pass_Thru_0 s'il est mis à 1.
8	G0_Gate_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition G0_Gate s'il est mis à 1.
7	AI_FIFO_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AI_FIFO s'il est mis à 1.
6	G0_TC_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition G0_TC s'il est mis à 1.
5	AI_Error_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AI_Error s'il est mis à 1.
4	AI_STOP_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AI_STOP s'il est mis à 1.
3	AI_START_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AI_START s'il est mis à 1.
2	AI_START2_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AI_START2 s'il est mis à 1.
1	AI_START1_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AI_START1 s'il est mis à 1.
0	AI_SC_TC_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AI_SC_TC s'il est mis à 1.

Tableau XI

Interrupt_A_Ack_Register

Bit	Nom	Fonction
15	G0_Gate_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le G0_Gate_Interrupt_St et reconnaît la demande d'interruption si le G0_Gate_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
14	G0_TC_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le G0_TC_St et reconnaît la demande d'interruption si le G0_TC_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
13	AI_Error_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AI_Overflow_St et le AI_Overrun_St, et reconnaît la demande d'interruption si le AI_Error_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
12	AI_STOP_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AI_STOP_St et reconnaît la demande d'interruption si le AI_STOP_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
11	AI_START_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AI_START_St et reconnaît la demande d'interruption si le AI_Error_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
10	AI_START2_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AI_START2_St et reconnaît la demande d'interruption si le AI_START2_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
9	AI_START1_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AI_START1_St et reconnaît la demande d'interruption si le AI_START1_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
8	AI_SC_TC_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AI_SC_TC_St et reconnaît la demande d'interruption si le AI_SC_TC_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
7	AI_SC_TC_Error_Comfirm	Si le bit est mis à 1, il remet à 0 le AI_SC_TC_Error_St. Le bit est remis automatiquement à 0.

Tableau XI (suite)

Bit	Nom	Fonction
6	G0_TC_Error_Comfirm	Si le bit est mis à 1, il remet à 0 le G0_TC_Error_St. Le bit est remis automatiquement à 0.
5	G0_Gate_Error_Comfirm	Si le bit est mis à 1, il remet à 0 le G0_Gate_Error_St. Le bit est remis automatiquement à 0.
4-0	Reserved	

Tableau XII

Interrupt_B_Enable_Register

Bit	Nom	Fonction
15-12	Reserved	
11	Pass_Thru_1_Interrupt_Enable	Le bit met en fonction l'interruption pour le signal Pass_Thru_1 s'il est mis à 1.
10	G1_Gate_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition G1_Gate s'il est mis à 1.
9	G1_TC_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition G1_TC s'il est mis à 1.
8	AO_FIFO_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AO_FIFO s'il est mis à 1.
7	AO_UI2_TC_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AO_UI2_TC s'il est mis à 1.
6	AO_UC_TC_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AO_UC_TC s'il est mis à 1.
5	AO_Error_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AO_Error s'il est mis à 1.
4	AO_STOP_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AO_STOP s'il est mis à 1.
3	AO_START_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AO_START s'il est mis à 1.
2	AO_UPDATE_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AO_UPDATE s'il est mis à 1.
1	AO_START1_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AO_START1 s'il est mis à 1.
0	AO_BC_TC_Interrupt_Enable	Le bit met en fonction l'interruption pour la condition AO_BC_TC s'il est mis à 1.

Tableau XIII

Interrupt_B_Ack_Register

Bit	Nom	Fonction
15	G1_Gate_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le G1_Gate_Interrupt_St et reconnaît la demande d'interruption si le G1_Gate_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
14	G1_TC_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le G1_TC_St et reconnaît la demande d'interruption si le G1_TC_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
13	AO_Error_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AO_Overrun_St et reconnaît la demande d'interruption si le AI_Error_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
12	AO_STOP_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AO_STOP_St et reconnaît la demande d'interruption si le AO_STOP_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
11	AO_START_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AO_START_St et reconnaît la demande d'interruption si le AO_Error_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
10	AO_UPDATE_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AO_UPDATE_St et reconnaît la demande d'interruption si le AO_UPDATE_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
9	AO_START1_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AO_START1_St et reconnaît la demande d'interruption si le AO_START1_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
8	AO_BC_TC_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AO_BC_TC_St et reconnaît la demande d'interruption si le AO_BC_TC_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
7	AO_UC_TC_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AO_UC_TC_St et reconnaît la demande d'interruption si le AO_UC_TC_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.

Tableau XIII (suite)

Bit	Nom	Fonction
6	AO_UI2_TC_Interrupt_Ack	Si le bit est mis à 1, il remet à 0 le AO_UI2_TC_St et reconnaît la demande d'interruption si le AO_UI2_TC_Interrupt_Enable est en fonction. Le bit est remis automatiquement à 0.
5	AO_UI2_TC_Error_Comfirm	Si le bit est mis à 1, il remet à 0 le AO_UI2_TC_St. Le bit est remis automatiquement à 0.
4	AO_BC_TC_Error_Comfirm	Si le bit est mis à 1, il remet à 0 le AO_BC_TC_Error_St. Le bit est remis automatiquement à 0.
3	AO_BC_TC_Trigger_Error_Comfirm	Si le bit est mis à 1, il remet à 0 le AO_BC_TC_Trigger_Error_St. Le bit est remis automatiquement à 0.
2	G1_TC_Error_Comfirm	Si le bit est mis à 1, il remet à 0 le G1_TC_Error_St. Le bit est remis automatiquement à 0.
1	G1_Gate_Error_Comfirm	Si le bit est mis à 1, il remet à 0 le G1_Gate_Error_St. Le bit est remis automatiquement à 0.
0	Reserved	

Tableau XIV

Joint_Status_1_Register

Bit	Nom	Fonction
15	AI_Last_Shiftin_St	
14	AO_UC_Q_St	
13	AO_UI2_Gate_St	
12	DIO_Serial_IO_In_Progress_St	
11	AO_External_Gate_St	
10	AI_External_Gate_St	
9-8	AI_SI2_Q_St	
7	AO_Start_Stop_Gate_St	
6	AO_BC_Gate_St	
5	AI_Start_Stop_Gate_St	
4	AI_SC_Gate_St	
3	G1_Gate_St	Le bit indique le statut du signal G1_GATE: 0: Inactif 1: Actif
2	G0_Gate_St	Le bit indique le statut du signal G0_GATE: 0: Inactif 1: Actif
1	G1_Bank_St	Le bit indique la banque du registre de chargement utilisée par le compteur 1: 0: Banque X 1: Banque Y
0	G0_Bank_St	Le bit indique la banque du registre de chargement utilisée par le compteur 0: 0: Banque X 1: Banque Y

BIBLIOGRAPHIE

1. National Instruments Corporation. (1998). *PCI E series register-level programmer manual* (Novembre 1998). Austin, TX.

2. National Instruments Corporation. (1999). *DAQ-STC technical reference manual* (Janvier 1999). Austin, TX.

3. The MathWorks, Inc. (2004). *Getting started with xPC Target* (Version 2). Natick, MA.

4. The MathWorks, Inc. (2004). *xPC Target user's guide* (Version 2). Natick, MA.

5. The MathWorks, Inc. (2004). *xPC Target I/O reference guide* (Version 2). Natick, MA.

6. The MathWorks, Inc. *Documentation: SimPowerSystems*, [En ligne]. http:// www.mathworks.com/access/helpdesk/help/toolbox/physmod/powersys/ powersys.html (Consulté le 20 janvier 2006).

7. dSPACE GmbH. (2004). *DS1104 hardware installation and configuration.* (Mars 2004). Paderborn, Allemagne.

8. dSPACE GmbH. (2004). *dSPACE software installation and management guide.* (Mars 2004). Paderborn, Allemagne.

9. dSPACE GmbH. (2004). *ControlDesk experiment guide.* (Mars 2004). Paderborn, Allemagne.

10. dSPACE GmbH. (2004). *RTI and RTI-MP implementation guide.* (Mars 2004). Paderborn, Allemagne.

11. De Kelper, B., Dessaint, L.-A., Al-Haddad, K., Nakra, H. (2002). A comprehensive approach to fixed-step simulation of switched circuits. *IEEE Transactions on Power Electronics*, 17, 216-224.

12. Champagne, R., Dessaint, L.-A., Fortin-Blanchette, H., Sybille, G. (2004). Analysis and validation of a real-time ac drive simulator. *IEEE Transactions on Power Electronics*, 19, 336-345.

13. Le-Huy, H., Sybille, G., Giroux, P., Soumagne, J.-C., Guay, F. (1999). Digital real-time simulation of a four-quadrant dc drive for static transfer switch testing. *Proceedings of the IEEE Power Engineering Society Winter Meeting*, 761-765.

14. Abourida, S., Dufour, C., Bélanger, J., Murere, G., Léchevin, N., Yu, B. (2002). Real-time PC-based simulator of electric systems and drives. *Proceedings of the IEEE Applied Power Electronics Conference and Exposition (APEC)*, 433-438.

15. Zhao, Z. M., Meng, S., Yue, X. N. (1999). A flexible virtual system for real-time simulation and evaluation of motor drives. *Proceedings of the IEEE International Conference on Power Electronics and Drive Systems (PEDS)*, 361-365.

16. Champagne, R., Dessaint, L.-A., Sybille, G., Khodabakhchian, B. (2000). An approach for real-time simulation of electric drives. *Proceedings of the Canadian Conference Electrical and Computer Engineering (CCECE).*

17. Dessaint, L.-A., Al-Haddad, K., Le-Huy, H., Sybille, G., Brunelle, P. (1999). A power system simulation tool based on simulink. *IEEE Transactions on Industrial Electronics*, 46, 1252-1254.

18. Sureshbabu, N., Seshagiri, S., Masrur, A., Powell, B. K. (1999). On real-time simulation of induction motors. *Proceedings of the American Control Conference*, 719-723.

19. Champagne, R., Dessaint, L.-A., Sybille, G., Casoria, S. (1999). Real-time simulation of electrical drives using the state variable approach. *Proceedings of the 3rd International Conference on Digital Power Systems Simulators (ICDS).*

20. Larose, C., Guerette, S., Guay, F., Nolet, A., Yamamoto, T., Enomoto, H., Kono, Y., Hasegawa, Y., Taoko, H. (2002). A fully digital real-time power system simulator based on PC-cluster. *Proceedings of the 7th International Conference on Modeling Simulation Electrical Machines and Converters Systems (Electrimacs'02).*

21. Kaddouri, A., Khodabakhchian, B., Dessaint, L.-A., Champagne, R., Snider,L. (1999). A new generation of simulation tools for electric drives and power electronics. *Proceedings of the 3rd IEEE International Conference On Power Electronics And Drive Systems (PEDS).*

22. Dufour, C., Bélanger, J. (2001). Discrete time compensation of switching events for accurate real-time simulation of power systems. *Proceedings of the 27th Annual Conference of the IEEE Industrial Electronics Society (IECON'01).*

23. Dezza, F.C., Cristaldi, L., Ferrero, A., Monti, A. (1996). Real-time virtual system for electric drive testing: basic concepts and implementation. *Proceedings of the 8th Electrotechnical Conference*, 1, 513-516.

24. Bélanger, J., Léchevin, N., Murere, G. (2000). Real-time simulation of averaged models power converter (Part 1, 2 & 3). *Application notes, Opal-RT Technologies.*

25. Krause, P.C., Wasynczuk, O. & Sudhoff, S.D. (2002). *Analysis of Electric Machinery and Drive Systems* (2e éd.). Piscataway, NJ: Wiley-IEEE Press.

26. Bose, B.K. (2001). *Modern Power Electronics and AC Drives*. Englewood Cliffs, NJ: Prentice-Hall.