Simulation numérique en temps réel d'un transformateur

Ing. P. MINTEN
Ir P. LEFEBVRE
ISICHt – Mons

Ce document traite de la simulation numérique en temps réel d'un transformateur monophasé réel et dont le noyau a été considéré comme linéaire. La simulation est réalisée à l'aide d'une carte à porte logique programmable (FPGA) et en virgule flottante grâce au logiciel « NI LabView FPGA ».

Mots clés **:** *simulation numérique, temps réel, modélisation numérique, dérivée numérique d'un signal, virgule flottante, virgule fixe, carte logique programmable (FPGA), convertisseur de puissance ferroviaire.*

This document reports a numerical simulation in real time of a single-phase transformer (the core is supposed to be linear). The model is supported by a Field Programmable Gate Array (FPGA) card, realized in floating point and coded through NI LabView FPGA software.

Keywords **:** *numerical simulation, real time, numerical model, numerical derivative of a signal, floating point, fixed point, Field Programmable Gate Array (FPGA), railway inverter.*

1. Introduction

Une simulation est une représentation modélisée d'un phénomène. Elle permet de prévoir les conséquences de celui-ci. On peut tout d'abord distinguer deux types de simulation. La première analogique peut être réalisée grâce à des cartes électroniques utilisant des composants passifs (résistances, condensateurs, inductances) qui reproduisent les équations mathématiques d'un modèle. La seconde numérique utilisent des composants actifs tels que des transistors, des portes logiques, etc. Dans cette dernière solution, on a alors souvent recours à des langages de hauts niveaux et des compilateurs pour réaliser la programmation du support, qui peut être des cartes électroniques, des ordinateurs, etc. On appelle en générale le support de la simulation : le simulateur.

Dans le cadre du travail, ce simulateur sert à la validation des électroniques de contrôle de convertisseurs de puissances ferroviaires. Ceux-ci permettent de transformer l'énergie disponible à la caténaire et de l'adapter au niveau de tension des appareils utilisateurs comme présenté à la figure 1 ci-dessous.

Ces convertisseurs sont constitués de :

- IGBT[1];
- un transformateur ;
- des filtres ;
- une électronique de contrôle ;
- etc.

[1] « Insulated Gate Bipolar Transistor » en anglais

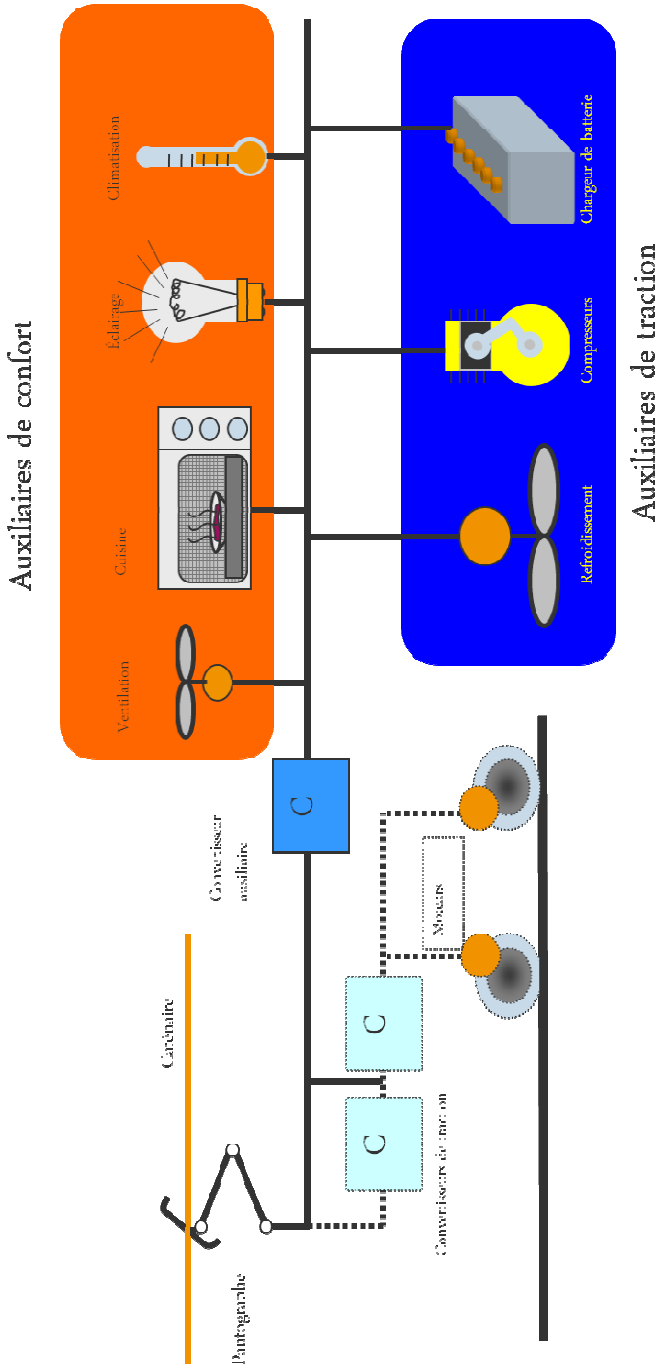

Figure 1 : Vue électrique schématique d'un train [Interne Alstom]

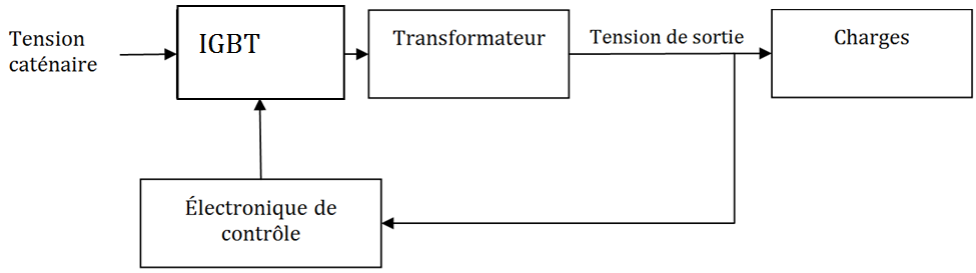

Figure 2: Schéma d'un convertisseur de puissance

On observe bien le rôle de l'électronique de contrôle à la figure 2 ci-dessus. Elle permet la régulation de la tension malgré les variations de la tension d'entrée.

Afin de s'assurer qu'elle correspond bien au cahier des charges du projet, elle est soumise à deux types de tests avec le simulateur. Les tests en boucle ouverte permettent de vérifier la spécification des entrées-sorties. Tandis que le test en boucle fermée permet de valider l'aspect régulation programmée sur l'électronique. Le simulateur représente donc le comportement des autres éléments du convertisseur de tension et réalise le test en boucle ouverte.

Actuellement la majorité de ces tests sont réalisés à l'aide de simulateur analogique. Cependant, il existe une volonté de passer à une version numérique et ce pour plusieurs raisons :

- La possibilité de réutiliser ;
- L'automatisation des différents tests ;
- La simulation des cas de défauts ;
- La simulation de comportements non linéaires ;
- La validation des résultats.

C'est donc le gain de temps qui est mis en avant.
La validation de la régulation ne peut se faire qu'avec un simulateur numérique dit « temps réel ». Cela signifie que le simulateur est capable d'interagir avec l'électronique de contrôle pendant la simulation. De cette façon, il est possible de tester celle-ci comme si elle était reliée à un véritable convertisseur de puissance. Ce type de simulation implique que le temps de réalisation de la communication et des calculs doit être inférieur au pas de calcul choisi. Dans notre cas, ce pas a été posé à une microseconde afin de rester le plus fidèle à la réalité. À l'inverse, des simulations en temps

différé impliquent que les stimuli sont prédéfinis et que le temps de réalisation des calculs peut être plus grand que le pas. Mais on ne peut pas tenir compte des réactions réelles de l'électronique. Ce type de simulation peut être réalisé par des logiciels du type « Matlab Simulink ». On voit donc bien la nécessité d'utiliser le temps réel, puisque l'objectif est de tester les réactions de l'électronique.

2. Un simulateur numérique [3] [5]

La réalisation d'un simulateur temps réel nécessite donc un matériel (hardware) capable de supporter des vitesses de calculs aussi rapides. Le choix pourrait se porter sur les cartes FPGA[2] ou les DSP[3]. Une carte FPGA est un ensemble de plusieurs millions de portes logiques non liées les unes aux autres. C'est lors de la programmation que celles-ci sont connectées entre elles pour réaliser les opérations demandées (addition, …). Les deux cartes FPGA sont également munies de multiplicateurs capables de réaliser des multiplications de manière aussi efficace qu'un DSP. Un DSP est un microprocesseur spécialisé dans la réalisation de tâche mathématique complexe. La question concerne le choix d'utiliser l'une ou l'autre technologie pour réaliser les calculs nécessaires à la simulation d'un CVS et les charges. Ce choix est principalement dû à la vitesse d'acquisition nécessaire à la réalisation d'un modèle en temps réel. La vitesse d'exécution a été posée à 1MHz de façon à pouvoir observer des phénomènes jusque 20kHz. La vitesse d'exécution des calculs doit donc être supérieure à l'échantillonnage, de manière à pouvoir réaliser un certain nombre d'additions, soustractions ou multiplications en série pendant cette microseconde. Or lorsque la vitesse dépasse les quelques MHz, les DSP sont fort sollicités car le processeur place les résultats dans des mémoires partagées externes. Par contre, une carte FPGA attribue par défaut des mémoires logiques internes rapides au stockage des résultats. Dès lors, la carte FPGA peut supporter des vitesses d'exécution plus rapides. De plus, une carte FPGA possède un grand nombre de multiplicateurs intégrés qui permettent d'exécuter plus d'opérations en une microseconde qu'un DSP.

---

[2] « Field Programmable Gate Array »

[3] « Digital Signal Processor »

3. Méthodes

La réalisation de ce travail est faite en plusieurs étapes et suivant la démarche se trouvant à la figure 3.

La première étape du travail a consisté à transcrire les divers phénomènes en équations. S'intéressant principalement au côté électrique, le schéma équivalent du transformateur a pu servir de base à l'élaboration du modèle [9].

Ensuite, ce modèle est traduit en équation mathématique [1][4][6]. Il a été utile de tester l'influence des équations les unes sur les autres grâce à Excel. De cette façon on a pu pré-simuler l'algorithme et le schéma de calcul avant d'entamer la partie programmation du simulateur.

Dès lors, il a été intéressant de comparer les résultats obtenu en « pré-simulation » et durant la simulation proprement dites afin de valider le programme du simulateur.

Une dernière partie consiste à essayer de déterminer de manière mathématique les limites de l'algorithme implémenté. Ce point ne sera pas abordé dans l'article.

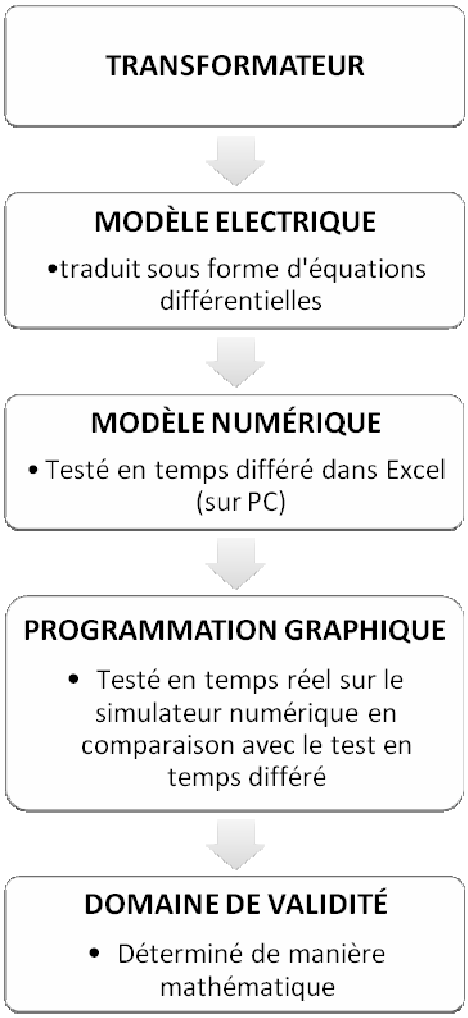

Figure 3 : Démarche adoptée dans le projet

4. Résultats

Cette partie présentera plus en détail la méthodologie choisie, associée à des résultats pratiques. Le modèle électrique choisi est le suivant :

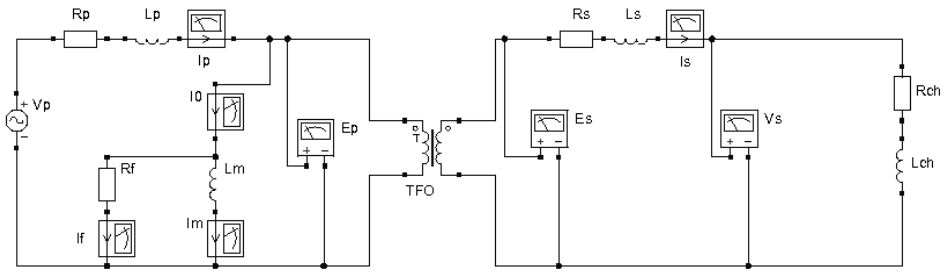

Figure 4 : Démarche adoptée – Modèle électrique

Le schéma équivalent peut être modélisé par le schéma fonctionnel suivant :

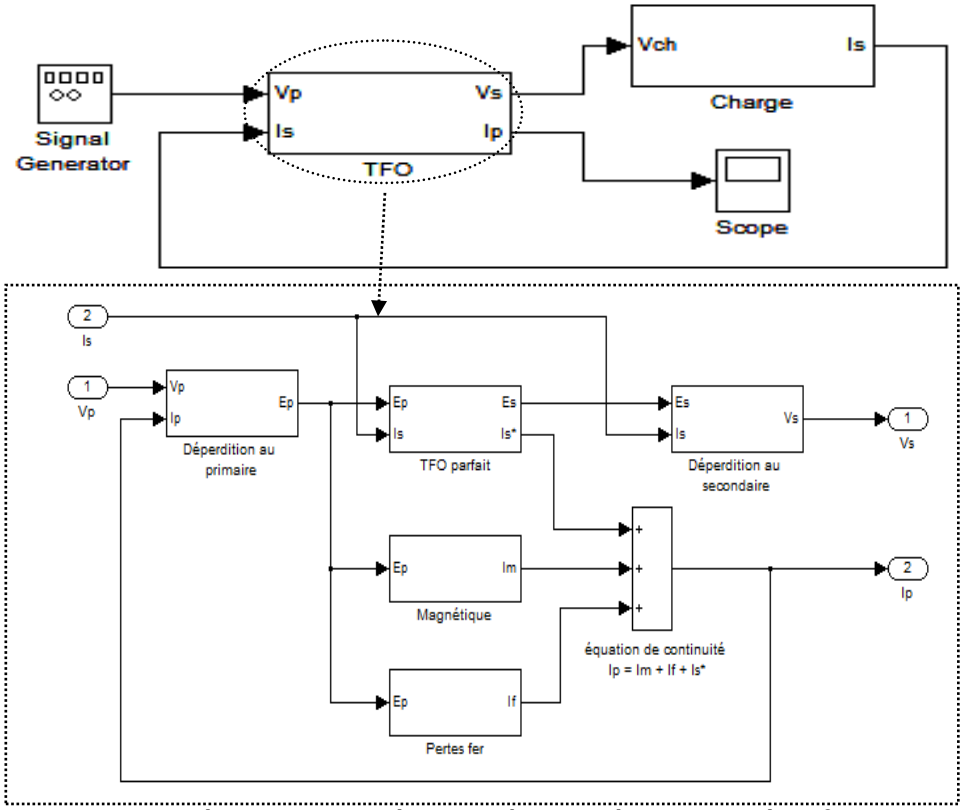

Figure 5 : Démarche adoptée – Modèle numérique - schéma fonctionnel d'un transformateur et d'une charge

Il est alors plus facile de se rendre compte que plusieurs fonctions sont similaires et peuvent être réalisées par des mêmes algorithmes. On distingue donc des fonctions « charges » générant un courant sous une différence de potentiel, et des fonctions « chute de tension ».

Dans le cadre d'une fonction « charge », on peut écrire l'équation générale se rapportant à un composant inductif :

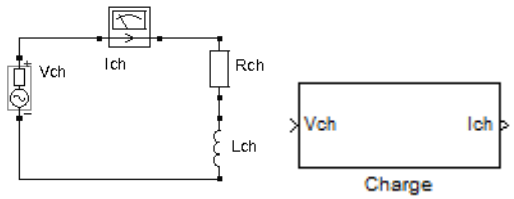

Figure 6 : schéma électrique et fonctionnel d'une charge inductive

$$\frac{di_{ch}}{dt} = \frac{v_{ch} - R_{ch} * i_{ch}}{L_{ch}}$$

Équation 1 : charge inductive

Pour résoudre ce type d'équation, il existe différentes méthodes avec leurs avantages et leurs inconvénients.

La première est la méthode d'Euler, qui est la plus simple et ne réalise qu'une seule approximation.

$$i_{ch\ i+1} = i_{ch\ i} + h * f(v_{ch\ i}, i_{ch\ i}) \quad (1)$$

$$f(x,y) = \frac{di_{ch}}{dt} \quad (2)$$

Équation 2 : Méthode d'Euler adaptée au cas pratique
Avec h : le pas d'intégration

Cependant, elle se réalise plus vite que les méthodes suivantes. La méthode d'Heun réalise deux approximations et demande donc deux fois plus temps que la méthode d'Euler. Et enfin la méthode de Runge Kunta (d'ordre 4) qui réalise quatre approximations et prend donc 4 fois plus de temps que la dite première méthode. Le choix s'est ici porté sur la méthode la plus rapide car le cahier des charges impose un pas d'intégration d'une microseconde.

Le but de la seconde fonction est le calcul d'une différence de tension, par exemple pour une charge inductive :

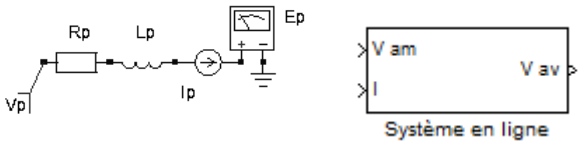

Figure 7 : Schéma électrique et fonctionnel d'un système en chute de tension

$$v_{am} - v_{av} = Ri + L\frac{di}{dt}$$

Équation 3 : Chute de tension sur un système RL

On doit donc à partir du courant circulant dans ce système en déduire la chute de tension. Il est donc nécessaire de réaliser une dérivation numérique. C'est là où réside la principale difficulté. La première méthode abordée est la plus simple. Intuitivement, on décline la dérivation en passant de l'infiniment petit aux accroissements entre les différents échantillons :

$$\frac{di}{dt} \approx \frac{\Delta i}{\Delta t}$$

Équation 4 : Approximation de la dérivée du courant

Cette méthode fonctionne mais provoque l'apparition d'oscillations à la montée. Pour mettre en évidence ces oscillations, nous utiliserons le circuit suivant :

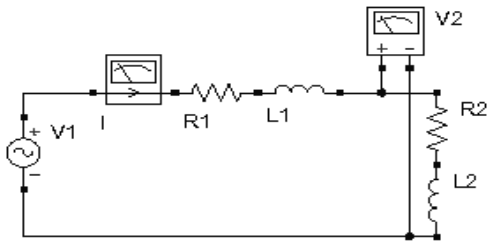

Figure 8 : schéma électrique du circuit servant à mettre en évidence les oscillations induites par la méthode de dérivation intuitive à la figure 9

L'impédance $Z_1 = R_1 + j\omega L_1$ sera résolue en tant que système « chute de tension » ; tandis que l'impédance $Z_2 = R_2 + j\omega L_2$ sera résolue en tant que

système « charge ». On peut alors mettre en évidence à la figure 9 que la dérivée du courant calculé par différence finie génère des oscillations.

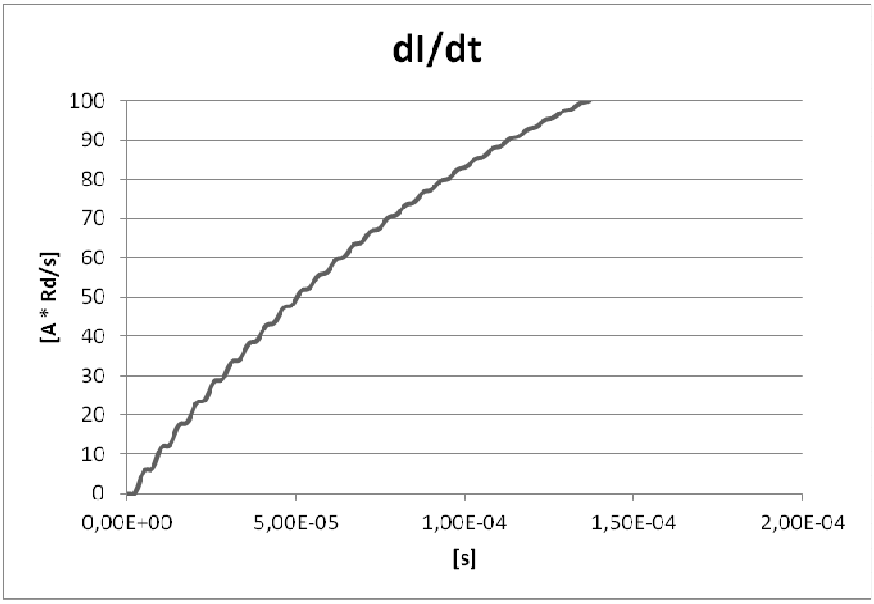

Figure 9 : Mise en évidence des oscillations dans la dérivée du courant

Ces oscillations peuvent conduire à une instabilité. En effet, ces oscillations au niveau de la dérivée du courant se répercutent dans la tension avale V_z. Cette dernière alimente la charge. Et la méthode d'Euler ayant l'inconvénient de propager et amplifier les erreurs, le courant résultant est donc lui-même perturbé. Il faut donc éliminer ces oscillations. La solution adoptée consiste à rejeter ce bruit par la mise en œuvre d'une dérivée filtrée [2].

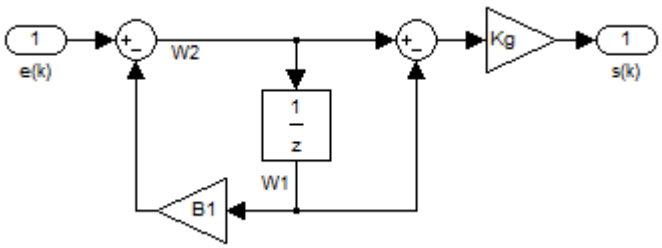

Figure 10 : Schéma fonctionnel numérique de la dérivée associée à un filtre d'ordre 1

On retrouve à la figure 10 ci-dessus le schéma fonctionnel, où l'opérateur $\frac{1}{z}$ est l'opérateur retard en numérique.

L'étape suivante du travail a consisté à tester les algorithmes et méthodes de calcul avant insertion dans le simulateur. En effet, la phase programmation est relativement complexe et longue. C'est pour cette raison que les méthodes de calculs seront testées grâce à un tableur. Il a d'abord testé les petits algorithmes de base, telle qu'une charge RL alimentée en continu ou en alternatif. Ensuite des algorithmes de plus en plus complexes ont été testés, tel qu'un transformateur monophasé. Ainsi les divers problèmes se sont posés un à un et ont pu être résolus aisément.

Le transformateur étant précédé d'un pont de semi-conducteur pouvant servir d'hacheur, la tension est alors un signal modulé en largeur d'impulsion ayant la même valeur efficace qu'une sinusoïde. Grâce à une simulation sur Excel[4], on peut alors déduire les courbes suivantes :

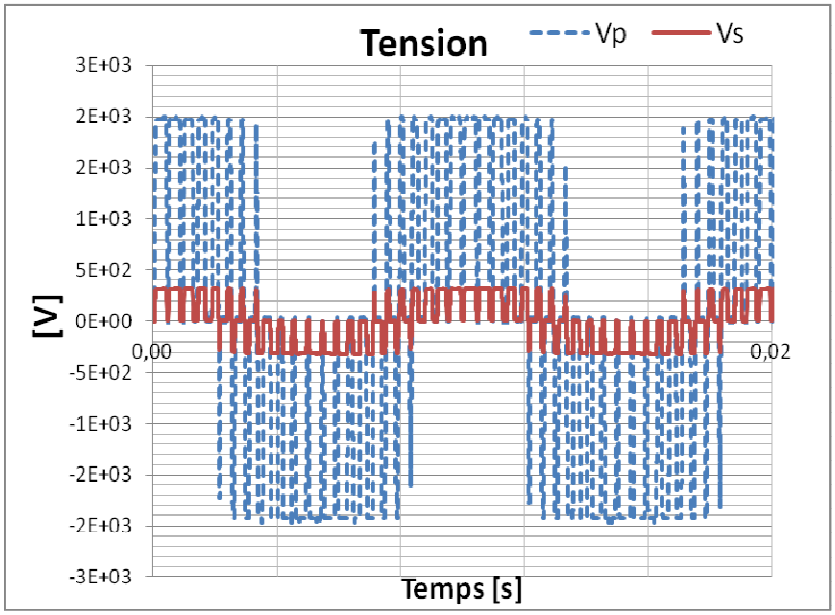

Figure 11 : Tensions du transformateur suite à une tension PWM au primaire

[4] Cette simulation met en pratique les différentes équations présentées précédemment.

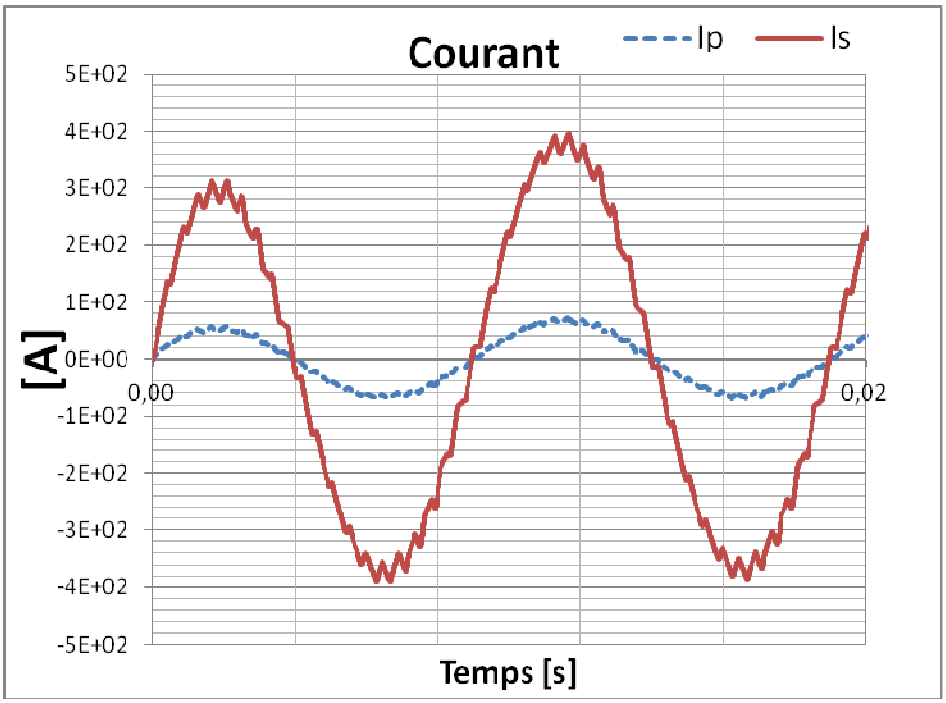

Figure 12 : Courants du transformateur suite à une tension PWM au primaire

On observe alors que les courants générés sont filtrés : les hautes fréquences ont été éliminées, et il reste une sinusoïde avec quelques harmoniques. En effet, une charge RL filtre les hautes fréquences provenant de la tension.

L'étape trois du projet consiste à implémenter les algorithmes dans le simulateur grâce à l'interface de programmation LabVIEW. Une contrainte a été l'architecture à adopter pour réaliser le plus efficacement les divers calculs. Par efficacement on entend le plus rapidement possible, mais également en utilisant le moins de ressources possibles de la carte FPGA (et donc utiliser le moins grand nombre de portes logiques).

La solution choisie a été d'adopter une structure sous forme de machine d'état. Chaque état réalise un certain nombre d'opérations mathématiques relatives aux méthodes de calculs choisies. Pour chaque état, il faut adopter une structure qui va permettre le transfert des résultats de l'un à l'autre. Mais il faut également utiliser des unités d'opération (addition, soustraction, multiplication) qui sont réutilisées à chaque état de la machine. On est alors

certains de ne pas utiliser plus de ressources que nécessaires pour réaliser les calculs. On effectue donc les étapes suivantes en boucle :

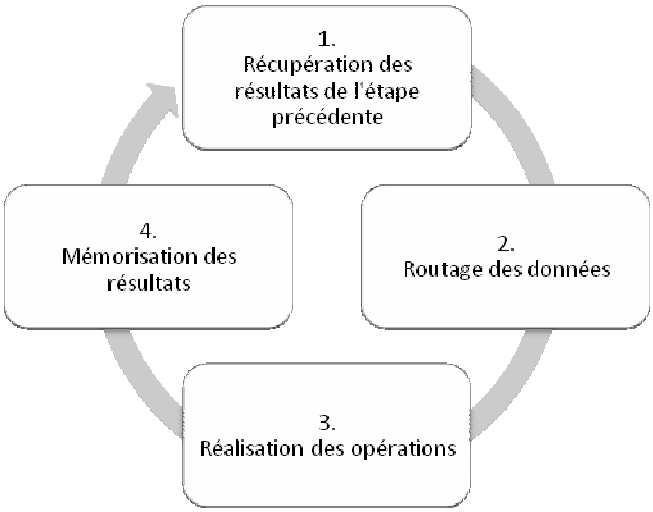

Figure 13 : Structure de la machine d'état

Cette structure répétitive peut donc être réalisée à l'aide d'une boucle. On peut la visualiser à la figure 14 comme étant le cadre extérieur. Cette boucle est même cadencée de façon à respecter la période d'échantillonnage précisément. On peut y observer ensuite un cadre intérieur. Celui-ci permet la réalisation du routage de données vers les différentes unités de calculs situées juste à droite. Cette zone de routage varie donc en fonction de l'étape. Et enfin on peut retrouver des petits carrés sur la boucle cadencée, nommés « cluster ». Ils permettent de réaliser la fonction de mémorisation des résultats.

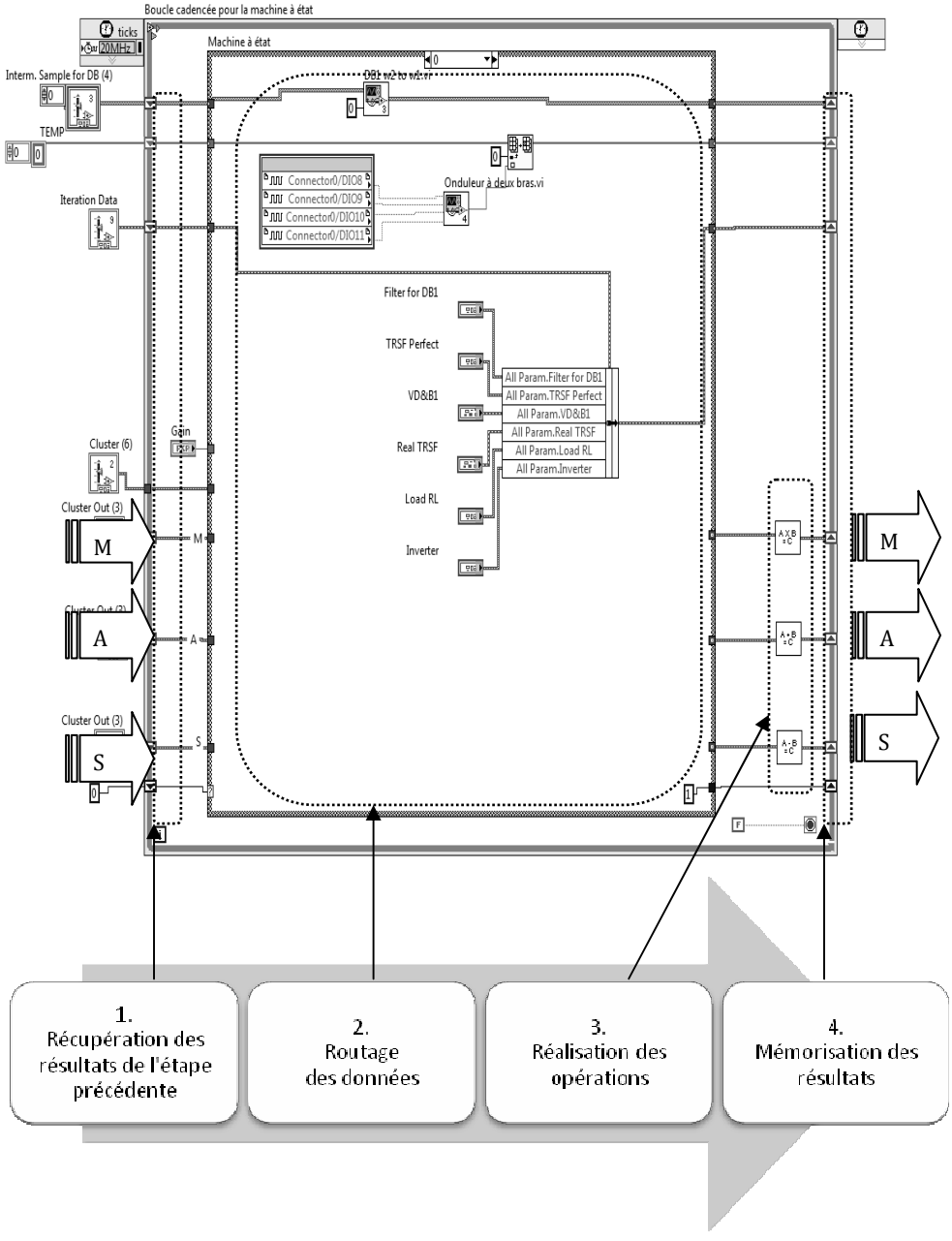

Figure 14 : Machine d'état - organisation pratique

Enfin la dernière étape a été de réaliser des tests en temps réel sur le simulateur et comparer les résultats obtenus par rapport au temps différé.

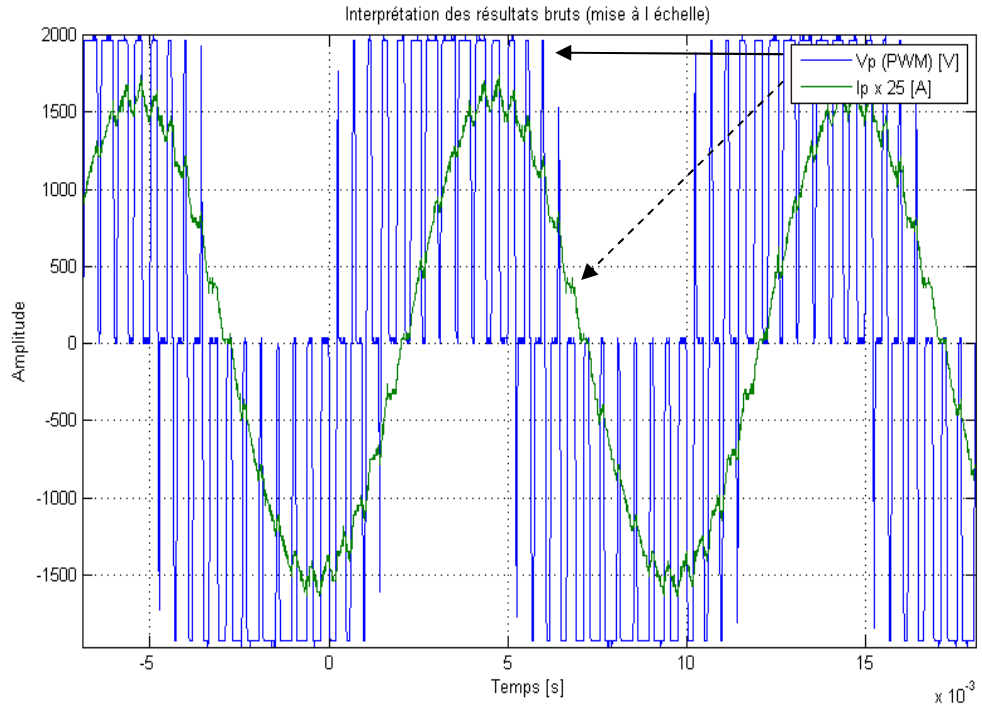

Figure 15 : Simulation en temps réel sur le simulateur

Les tests réalisés dans Excel et en temps réel ayant les mêmes paramètres, on peut comparer les deux courbes de courant.

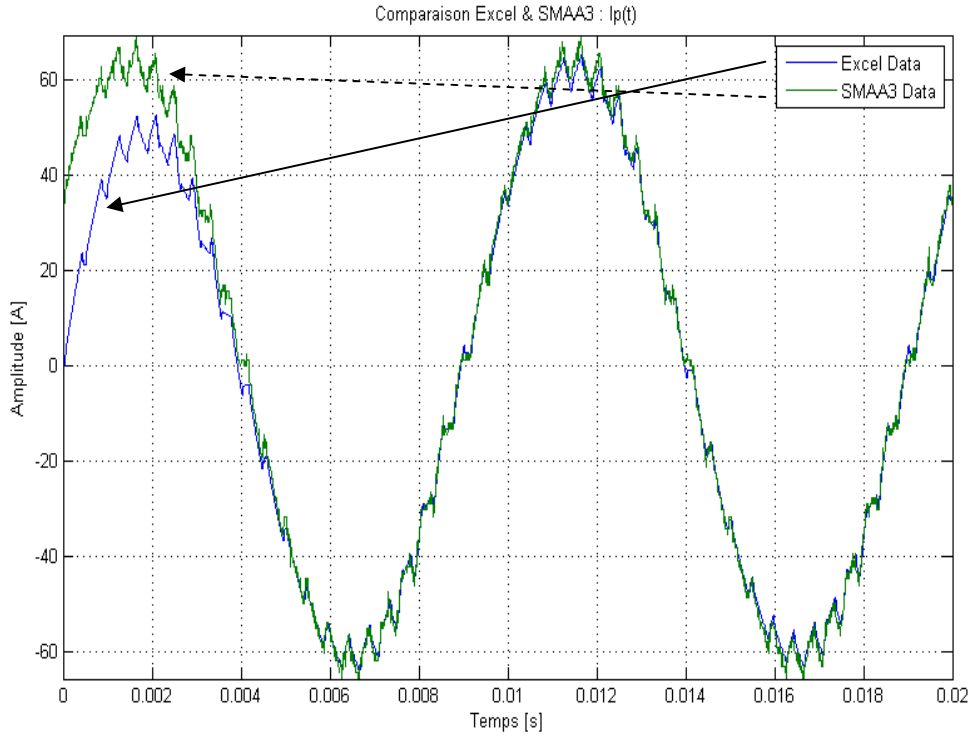

Figure 16 : Comparaison entre les résultats et la simulation Excel

On observe bien à la figure 16 la présence d'un transitoire sur la courbe provenant d'Excel. En effet, les conditions initiales y sont nulles ; alors que dans le simulateur, la mesure des données a été faite en régime. Ensuite, la simulation Excel tend vers le régime permanent et vient littéralement coller à la courbe pratique (celle du simulateur). À une petite différence près : l'amplitude n'est pas tout à fait la même. Cette légère différence peut facilement s'expliquer par des erreurs de mesures sur le courant primaire ; mais également sur le stimulus, qui est utilisé par la suite dans Excel pour générer les données.

D'autres tests ont également été réalisés afin de vérifier la cohérence : essai en variant la fréquence, en variant les paramètres, des transformées de Fourier,…

Cependant, il peut arriver parfois que lors de changement brusque d'obtenir des valeurs intermédiaires dépassant la valeur maximum codifiable par le format standard utilisé sur la carte FPGA. Par conséquent, le résultat devient instable. Le format natif de celle-ci est la virgule fixe. Facile à manipuler mais dès que les valeurs peuvent devenir grandes, l'augmentation du format

ralentit l'exécution des opérations. La solution trouvée fut d'évoluer vers un format en virgule flottante qui pose moins de problème pour les grands nombres. Suite à l'étude entre diverses possibilités d'implémentation [7] [8] et de tests en termes de rapidités et ressources utilisées, il a été décidé d'utiliser les algorithmes de Coregen[5].

5. Conclusion

La majeur partie du travail a consisté à modéliser et à implémenter la solution de la manière la plus adéquate et performante possible, en faisant des compromis.

Le travail aura donc permis d'effectuer un pas dans le développement du projet de développement d'un simulateur numérique. L'utilisation d'un transformateur monophasé n'est cependant pas suffisante mais permettra le développement d'un transformateur triphasé devant prendre en compte d'éventuels défauts dans le circuit, ou de charges déséquilibrées. Une piste pour résoudre ce dernier point serait d'utiliser la décomposition en composantes symétriques grâce à la transformée de Fortescue.

[5] Coregen un logiciel Xilink qui permet la génération de cœur de calcul dédié au FPGA à partir d'algorithme prédéfini. Ces cœurs peuvent ensuite être réutilisés dans l'algorithme développé sous LabVIEW FPGA.

6. Sources

[1] *BASTIEN J. & MARTIN J-N, Introduction à l'analyse numérique*, Dunod, Paris, 2003, Chapitre 5 : « équation différentielles » pp 195-204

[2] *BASTIDE G. ET SAGNES G.*, 1982, *Dérivation numérique d'une collection de points*, Revue Phys. Appl., 17 novembre, pp. 769-774

[3] Dr. *DICK C.* (Hunt Engineering), *FPGAs : The high-end alternative for DSP Applications*, page consultée le 21/01/2011 http://www.hunteng.co.uk/pdfs/tech/dsp1736fpga.pdf

[4] *FORTIN A., Analyse numérique pour ingénieurs*, deuxième édition (2001), chapitre 7 : « Equations différentielles » pp 396-401

[5] Hunt Engineering, *Choosing FPGA or DSP for your application*, page consultée le 21/01/2011 http://www.hunteng.co.uk/info/fpga-or-dsp.htm

[6] *HURÉ* J-M (LUTH/Observatoire de Paris Meudon, et Université Paris 7 Denis Diderot) & Pelat D. (LUTh/Observatoire de Paris Meudon), *Méthodes numériques : éléments d'un premier parcours*, Version 2 (Novembre 2002), Chapitre 3 « Dérivation » pp 27-33, Chapitre 7 « équations aux dérivées ordinaires » pp 69-84

[7] National Instruments, *Importing External IP into LabVIEW FPGA* http://zone.ni.com/devzone/cda/tut/p/id/7444

[8] National Instruments, Aide et support : http://www.ni.com

[9] *WILDI TH. ET SYBILLE G.*, 2005 (4ième édition), *Électrotechnique*, De Boeck, Chapitre 30 : Transformateur, pp 440-477