RENPAR'15 / CFSE'3 / SympAAA'2003
La Colle sur Loup, France, 15 au 17 octobre 2003

SPIN, un micro-réseau d'interconnexion à commutation de paquets respectant la norme VCI. Concepts généraux et validation.

Herve Charlery - Emmanuelle Encrenaz - Alain Greiner - Adrijean Andriahantenaina - Laurent Mortiez

Département ASIM (Laboratoire LIP6)
Université Pierre et Marie CURIE
4, place Jussieu, 75252 Paris Cedex 05
e-mails : {Prenom.Nom}@lip6.fr

Résumé
Le micro-réseau SPIN (pour Scalable Programmable Integrated Network) est un réseau d'interconnexion à commutation de paquets pour systèmes intégrés sur puce développé au LIP6. Cette technologie fournit un mécanisme de communication très général entre les différents composants virtuels connectés dans le système. De plus, la bande passante croit linéairement avec le nombre de processeurs intégrés.
La mise au point d'un tel interconnect a néanmoins soulevé des problèmes tels que la présence d'interblocages, et la façon de répartir les abonnées autour du réseau.
On décrira plus particulièrement les fonctionnalités et l'architecture interne des *wrappers* VCI/SPIN qui permettent de fournir aux composants virtuels une interface de communication respectant le standard VCI (Virtual Component Interface).
Ensuite, on proposera une solution afin de supprimer les risques d'interblocages, et enfin on présentera les premiers résultats d'une comparaison avec un bus système.

Mots-clés : spin, wrapper, vci, micro-réseau, vérification

1. Introduction

Le concept `SPIN` [1][2] d'architecture de communication à haut débit pour systèmes intégrés sur puce dérive de l'expérience acquise dans le domaine des calculateurs parallèles. Ces machines ont de gros besoins en bande passante. Elles utilisent souvent des réseaux d'interconnexion multi-étages qui mettent en oeuvre des liaisons point-à-point et des routeurs, comme substitut au traditionnel "bus système" [3].
Dans les systèmes intégrés, le bus de communication est souvent le goulot d'étranglement, et cette tendance ne peut que s'accentuer avec l'augmentation des capacités d'intégration[4].
Il y a deux raisons majeures à cela. La première est qu'il ne permet pas de communications parallèles, et la seconde, qu'il ralentit le système du fait des techniques de circuiterie de type multi-émetteurs.
La technologie de micro-réseau à commutation de paquets `SPIN` est une réponse possible à ces problèmes. Non seulement des communications simultanées sont possibles, mais en plus, on peut monter en fréquence, vu que l'on utilise exclusivement des liaisons point-à-point.
Cependant pour permettre une migration aisée d'une architecture à base de bus vers une architecture utilisant un réseau commuté, il faut permettre la réutilisation des composants existants (IP cores). Par conséquent, le réseau doit fournir aux concepteurs de systèmes la même interface et le même type de service que le bus traditionnel. Le standard VCI [5] (`Virtual Component Interface`), normalisé par le consortium VSIA, définit un protocole de communication de type "espace mémoire adressable partagé", qui peut être implémenté aussi bien par un bus système traditionnel de type AMBA [6] ou PIBUS [7], que par un réseau d'interconnexion commuté.
Pour passer d'un interconnect à un autre, il suffit d'utiliser des convertisseurs de protocoles, appelés `wrapper`. Dans le cas des interconnects cités précédemment, ce sont des wrapper VCI/AMBA, des wrapper VCI/PI ou alors des wrapper VCI/SPIN. Ces wrappers sont de type initiateur ("M-wrapper") ou cible ("T-wrapper"), selon qu'ils sont reliés respectivement à un initiateur ou à une cible. Pour ce qui est de l'interface VCI, celle-ci présente de nombreux avantages. En effet, VCI introduit un découplage

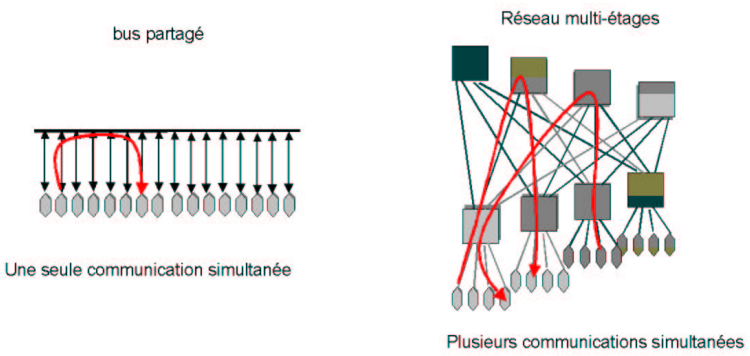

FIG. 1 – Schémas des communications selon le type d'interconnect

complet entre la conception ou le choix des composants du système (coeurs de microprocesseur, DSP ou coprocesseurs spécialisés), et le choix du dispositif matériel de communication (bus classique, bus hiérarchiques, réseau commuté, etc...)[1]. De plus, l'extrême simplicité du mécanisme de contrôle de flux défini par VCI facilite beaucoup la conception des interfaces d'accès au réseau. Enfin, VCI supporte les transactions éclatées.

Ce papier présente comment la technologie de réseau commuté SPIN peut être utilisée pour fournir un mécanisme d'interconnexion générique respectant la norme VCI. Nous présentons successivement les caractéristiques essentielles du réseau SPIN, et les principes généraux de l'interface VCI. Nous décrivons ensuite l'architecture des wrappers VCI/SPIN qui réalisent la conversion entre les deux protocoles.

Enfin, nous terminons par une description de la méthode générale de validation de cette architecture et les résultats d'un test de surcharge réalisé sur un réseau SPIN et un Pi-Bus, interconnectant chacun 32 abonnés.

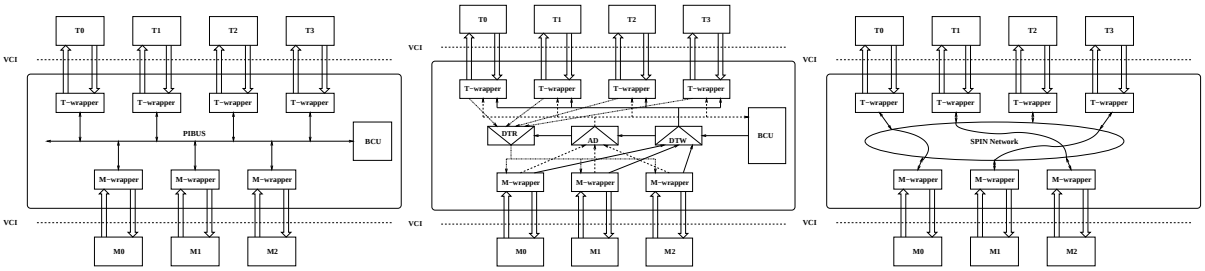

FIG. 2 – Schémas respectifs d'interconnexions PI/AMBA/SPIN avec VCI

2. Le réseau SPIN

2.1. Généralités

Le réseau SPIN est basé sur une topologie dite d'arbre quaternaire élargi ("fat-tree"). C'est à dire que chaque routeur RSPIN, noeud de l'arbre, possède quatre fils et quatre pères. Les chemins issus des pères sont équivalents, mais ceux issus des fils sont déterminés en fonction du numéro de destination de l'information à router.

Les liens du réseau SPIN sont des liens bidirectionnels point-à-point.

Le mécanisme de contrôle de flux utilisé est de type `CREDIT`. Le producteur utilise une estimation du taux de remplissage du consommateur, afin de réguler l'envoi des données. Ceci permet d'insérer des

[1] Cf Figures 2 : Schémas respectifs d'interconnections PI/AMBA/SPIN avec VCI

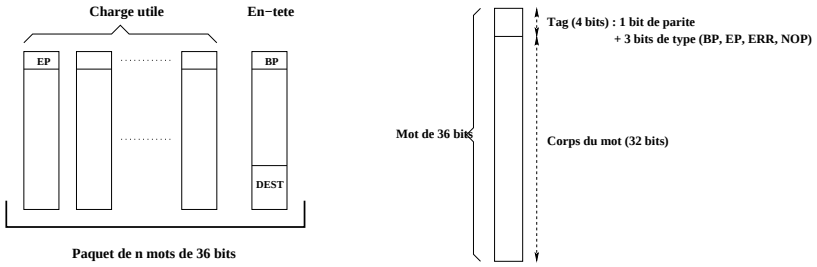

FIG. 3 – Structure d'un paquet et d'un mot de type SPIN

barrières de registres sans avoir de perte d'information, ce que ne permettrait pas un contrôle de flux de type `FIFO`.

2.2. Structure d'un paquet SPIN

Les informations qui circulent sur le réseau SPIN sont des `paquets`[2]. Un paquet SPIN est une séquence de mots de 36 bits, dont le premier mot, appelé `en-tête` possède un marqueur de début de paquet (`BP`) et le dernier mot un marqueur de fin de paquet (`EP`).
L'ensemble des mots qui suivent le mot d'en-tête, forment la `charge-utile` du paquet. Le nombre de mots constituant un paquet n'est pas borné.
La largeur d'un mot est de 36 bits qui se répartissent en 32 bits de données, 3 bits d'étiquette qui permettent de typer les mots, et un bit de parité. Les marqueurs EP et BP sont transportés dans le champ étiquette.
Un troisième marqueur, `ERR`, sera utilisé pour indiquer une erreur durant le traitement de l'information. En effet, l'acquittement ne se fait pas par paquet, mais par mot.

2.3. Routage d'un paquet SPIN

SPIN met en oeuvre un routage distribué, dynamique, adaptatif, de type `wormhole`. Il est distribué parce que le choix de l'aiguillage d'un paquet se fait dans chaque routeur et non dans un organe central. Ce routage est dynamique parce que les chemins sont refermés après le dernier mot de chaque paquet, et sont réouverts en fonction des besoins de l'aiguillage. Les chemins issus des quatre pères étant équivalents, plusieurs chemins sont possibles pour aller d'un abonné à un autre. L'adaptativité se traduit par le choix fait en fonction de la congestion du réseau.
Le routage de type wormhole a pour but non seulement de limiter la latence, mais en plus de simplifier le routage d'un paquet. En effet, un paquet est perçu par le réseau comme étant une entité atomique, dont l'ensemble des mots suivent le chemin tracé par le premier. Ainsi donc, un paquet pourra se trouver à cheval sur plusieurs routeurs simultanément, sans qu'il n'y ait jamais de déséquencement.
Le premier mot d'un paquet SPIN contient nécessairement le numéro du port destinataire. Ce numéro, codé sur 8 bits est utilisé par le réseau pour acheminer le paquet vers le port destinataire. Chaque port est donc identifié par un numéro unique qui est imposé par la topologie du réseau, et ne peut donc être modifié.

3. L'interface VCI

La norme VCI fait l'hypothèse que tous les composants du système partagent un même espace d'adressage. Les maîtres (ou initiateurs) envoient des requêtes de lecture ou d'écriture vers des cibles, qui sont identifiées par les bits de poids fort de l'adresse, comme sur un bus. Il existe donc deux types de paquets VCI : Les maîtres envoient des paquets *requête* constitués de une ou plusieurs adresses (en cas de rafale). Les cibles renvoient des paquets *réponse*. Les paquets sont de longueur quelconque, et sont terminés par un marqueur de fin de paquet. Le paquet *réponse* possède la même longueur que le paquet *requête* auquel il est associé et toute requête reçoit un acquittement. Chaque mot d'un paquet *requête* VCI comporte une centaine de signaux parmi lesquels on trouve des signaux de contrôle de flux, de donnée

[2] Cf Figure 3 : Structure d'un paquet et d'un mot de type SPIN

(en cas d'écriture), d'adresse et de définition de la requête. Chaque mot d'un paquet *réponse*, comporte une quarantaine de signaux qui sont les signaux de contrôle de flux, de donnée, et d'acquittement pour la gestion des erreurs.
Le mécanisme d'accès au réseau d'interconnexion est extrêmement simple, puisque chaque composant a l'impression d'écrire dans une FIFO. Plusieurs initiateurs peuvent donc émettre simultanément des requêtes, et un même initiateur peut émettre une requête (n+1) sans attendre d'avoir reçu la réponse à la requête (n). Cette possibilité s'appuie sur un mécanisme d'étiquetage des paires (requête/réponse) et est utilisée par des processeurs *multi-thread* pour masquer la latence des accès mémoire.

4. Les wrappers VCI / SPIN

4.1. Principes généraux

Nous appelons " wrapper " un composant matériel qui réalise une conversion entre deux protocoles de communication.
Puisqu'il existe deux types de paquets VCI (paquets " requête " et paquets " réponse "), il faut définir un mécanisme de traduction spécifique pour chacun de ces deux types. De plus une requête de lecture ne contient pas les mêmes informations qu'une requête d'écriture (puisqu'il n'y a pas de données dans une requête de lecture). Il en va de même pour les réponses. Il faut donc distinguer en pratique 4 types de paquets VCI :
- cas a) un paquet " requête de lecture " VCI de longueur N (rafale de N lectures) engendrera un paquet SPIN de longueur N+1 : un mot par adresse, plus un mot d'en-tête.
- cas b) un paquet " requête d'écriture " VCI de longueur N (rafale de N écritures) engendrera un paquet SPIN de longueur 2N + 1 : deux mots par adresse, plus un mot d'en-tête.
- cas c) un paquet " réponse lecture " VCI de longueur N engendrera un paquet SPIN de longueur N+1 : un mot par adresse, plus un mot d'en-tête.
- cas d) un paquet " réponse écriture " VCI de longueur N engendrera un paquet SPIN de longueur N+1 : un mot par adresse, plus un mot d'en-tête.

Puisqu'il existe deux types de composants VCI (des initiateurs et des cibles), il y a donc deux types de wrappers. Le wrapper de l'initiateur est responsable des conversions $\mathrm{VCI} \rightarrow \mathrm{SPIN}$ dans les cas a) et b), ainsi que des conversions $\mathrm{SPIN} \rightarrow \mathrm{VCI}$ dans les cas c) et d). Le wrapper de la cible est responsable des conversions $\mathrm{VCI} \rightarrow \mathrm{SPIN}$ dans les cas c) et d), ainsi que des conversions $\mathrm{SPIN} \rightarrow \mathrm{VCI}$ dans les cas a) et b).
Les requêtes et les réponses étant fondamentalement asynchrones, dans chacun des deux wrappers (initiateur et cible), il y aura deux automates indépendants pour traiter les paquets "requête" dans un sens, et les paquets "réponse" dans l'autre.

4.2. Le wrapper initiateur

Le wrapper initiateur en plus des conversions de format décrites à la section précédente, a deux autres fonctionnalités importantes. C'est lui qui est chargé de décoder les bits de poids fort de l'adresse et de déterminer le numéro de port destinataire à placer dans l'en-tête du paquet SPIN. Il effectue ce " transcodage " en fonction des " tables de routage " qui définissent la structure de la carte mémoire (la carte mémoire attribue à chaque composant cible un segment particulier de l'espace adressable. Ces tables de routage sont évidemment identiques pour tous les wrappers, puisqu'elles constituent une caractéristique globale du système, mais elles doivent être " câblées " dans chaque wrapper initiateur. Cette fonction de sélection de la cible, traditionnellement assurée par le contrôleur de bus, est ici répliquée dans tous les wrappers. C'est un des prix à payer pour avoir une architecture de communication " extensible ". Accessoirement, le wrapper initiateur doit indiquer dans l'entête du paquet SPIN son propre numéro de port, pour permettre au wrapper cible de connaître le numéro du port auquel il doit envoyer le paquet réponse.
Par ailleurs, le standard VCI " advanced " permet à un même initiateur d'envoyer une seconde requête sans attendre d'avoir reçu la réponse à la première. Il utilise pour cela les champs TRDID et PHTID du mot VCI, qui permettent de définir un "numéro de transaction". Evidemment, le wrapper ne doit pas transmettre deux requêtes avec le même numéro de transaction. Il gère donc une table des requêtes en attente, identifiées par leur numéro de transaction.

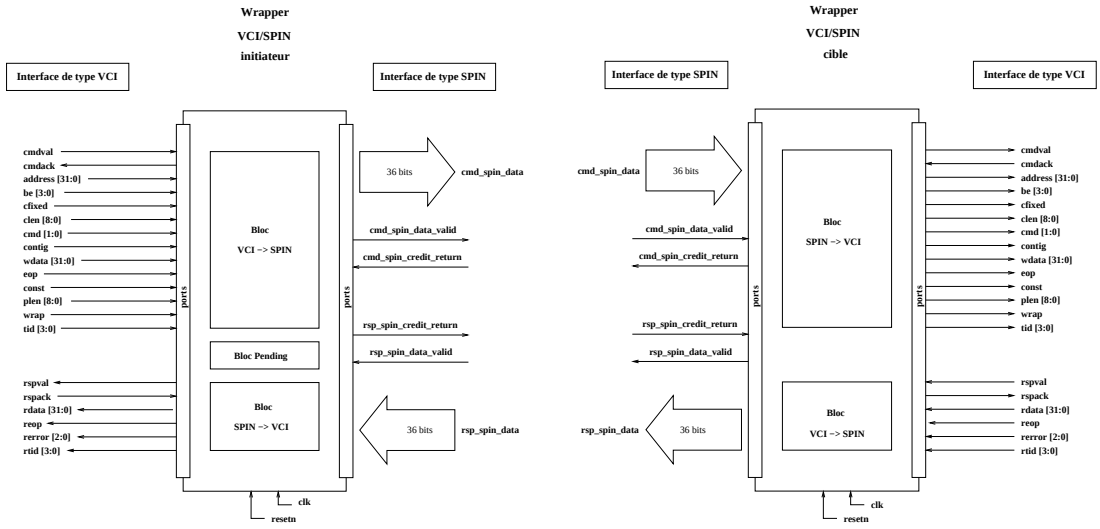

FIG. 4 – Les wrappers VCI/SPIN initiateur et cible

Le wrapper détecte les erreurs de routage, qui surviennent dans le cas où les bits de poids fort de l'adresse désignent une cible inexistante. Le mécanisme de gestion des erreurs de routage adopté se veut assez général et indépendant du type de l'interconnect.
Ce n'est donc pas le wrapper initiateur qui signale les erreurs en activant une interruption à destination de l'initiateur.
Il va simplement rediriger le paquet vers une cible par défaut qui elle renverra un paquet réponse signalant l'erreur. Ce sera ensuite à la charge de l'initiateur de la gérer.

4.3. Le wrapper cible

Le wrapper cible est plus simple, puisqu'il séquencialise les requêtes : Il attend d'avoir reçu le dernier mot de la réponse à la requête i avant de commencer à transmettre le premier mot de la requête i+1. On peut noter cependant qu'il faut absolument deux automates indépendant pour traiter les requêtes et les réponses si on veut éviter les situations de "dead-lock ".

5. Inter-blocages et vérification formelle

5.1. Généralités

Dans un réseau à commutation de paquets véhiculant simultanément des requêtes et des réponses sur le même médium, les paquets peuvent se trouver dans une configuration telle qu'aucun ne puisse arriver à destination, se bloquant les uns les autres. On se trouve alors en face d'un `inter-blocage` ("dead-lock").
Précédemment, on a vu que tout paquet requête donnait lieu à un paquet réponse.
On a émis l'hypothèse qu'en séparant le réseau en deux sous-réseaux, un pour les requêtes, et un pour les réponses, on supprimait ainsi les risques de dead-lock. Les requêtes et les réponses ne pouvant plus se bloquer les unes les autres.
Les techniques de vérification formelle ont été utilisées afin de valider cette hypothèse. Nous cherchons à montrer que le réseau ne distinguant pas les chemins des requêtes et des réponses contient des inter-blocages, et que celui les distinguant en est exempt. Pour la modélisation nous avons utilisé le langage `Promela`. Et pour la vérification ainsi que la simulation, le système de vérification `Spin` de Holzmann[8]. Le choix de ProMeLa (pour `Protocol Meta Language`) repose sur la nécessité de pouvoir modéliser des systèmes distribués, parallèles, asynchrones, dont les processus s'exécutent de façon concurrentes et communiquent à travers des canaux.
Une fois la modélisation réalisée, le système, est simulé et vérifié à l'aide des outils `spin`, et de son interface graphique `xspin`. Ces outils permettent entre autre de rechercher d'éventuels blocages, les

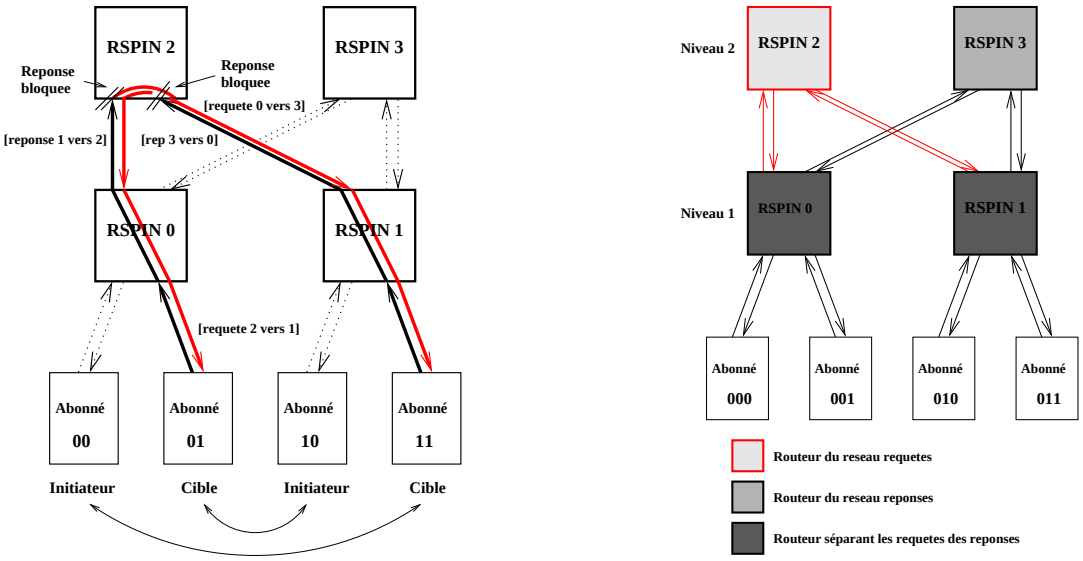

FIG. 5 – Schéma détaillé d'un inter-blocage et de la séparation requête/réponse

portions de codes jamais atteintes, de trouver les invariants, les cycles de non-progression et également de vérifier des propriétés LTL. L'absence d'inter-blocage peut être modélisée par une propriété LTL de type *chaque initiateur pourra toujours emettre un nouveau paquet* (ce qui implique que tout paquet finit par être acquitté).

5.2. Modélisation et analyse de l'inter-blocage

Cette étude s'est faite sur un réseau SPIN constitué de 4 routeurs RSPIN et de 4 abonnés. Ces routeurs sont connectés selon une structure dite en arbre binaire élargi.
Chaque routeur a deux ports père et deux ports fils. C'est moitié moins que la version originelle (SystemC) du routeur, mais cette simplification était nécessaire afin de réduire les risques d'explosion combinatoire. On part du principe que la solution trouvée pour une structure en arbre binaire élargi, s'appliquera également à une structure en arbre quaternaire élargi.
Afin de converger le plus rapidement possible vers un dead-lock, nous avons choisi de réduire les capacité de stockage des composants au minimum.
L'initiateur doit pouvoir recevoir des mots du paquet réponse sans attendre d'avoir envoyé tous les mots du paquet requête. De plus un initiateur ne peut s'adresser lui-même, ou adresser un autre initiateur.
Ces modèles de composants initiateur et cible, ont été instanciés à plusieurs reprises et connectés aux différents ports du réseau. La figure 5, schématise un inter-blocage faisant intervenir deux initiateurs et deux cibles. La taille des paquets a été fixée à 8 mots.
L'abonné initiateur 0 communique avec l'abonné cible 3, et l'initiateur 2 avec la cible 1. Au niveau du router "RSPIN 2", les deux réponses $1 \rightarrow 2$ et $3 \rightarrow 0$ sont bloquées par les deux requêtes, respectivement $0 \rightarrow 3$ et $2 \rightarrow 1$.
Les inter-blocages ont été mis en évidence par simulation et vérification.

5.3. Topologie solution

La séparation des requêtes et des réponses se traduit par la division du réseau originel en deux sous-réseaux, un pour les requêtes, en rouge, et un autre pour les réponses, en noir.
Cette division est réalisée par les routeurs de premier niveau, dont les ports supérieurs sont spécialisés pour les requêtes ou les réponses [3]. Les ports supérieurs gauche des routeurs de premier niveau sont destinés aux requêtes et ceux de droite aux réponses.
Cette topologie est en cours de validation.

[3] Cf Figure 6 : Schéma détaillé d'un inter-blocage et de la séparation

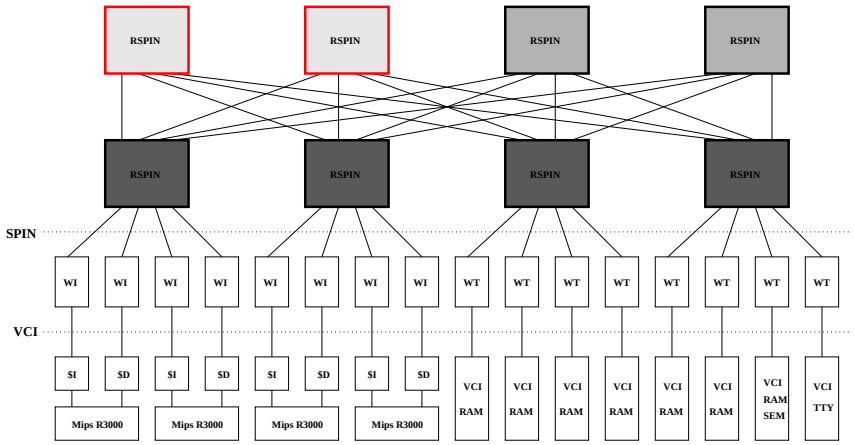

FIG. 6 – Système de test multiprocesseurs

6. La validation par simulation SystemC

La validation de l'architecture s'est déroulée en deux étapes. La première concerne la simulation d'une application réelle et est explicitée ci-après. La deuxième sera détaillée par la suite dans la section 8. Elle met en oeuvre des initiateurs dont l'objectif n'est pas d'exécuter un programme, mais de surcharger le réseau en envoyant des requêtes, simultanément, à plusieurs cibles.
Pour la simulation de l'application réelle, on utilise un outil développé au laboratoire LIP6 : le simulateur CASS [9][10]. Ce simulateur prend en entrée des modèles écrits en C/C++, de niveau RTL, permettant une simulation *cycle-true / bit-true*.
L'architecture des wrappers a été validée par la simulation d'une application parallèle multi-threads réalisant un calcul de Transformation de Fourrier Rapide (FFT), sous le contrôle d'un petit système d'exploitation multiprocesseurs et multi-threads. Cette application a été simulée sur une architecture matérielle [4] composée de huit routeurs RSPIN, de 16 wrappers initiateurs et cibles (respectivement WI et WT), de 4 processeurs de type MIPS R3000 doté de caches à interfaces VCI, de RAMs et d'une fenêtre d'affichage. La durée de la simulation pour ce système comportant 44 composants, est d'environ 80 secondes sur une machine Solaris à 450MHz pour 200 000 cycles simulés.
Ce système a également été validé à l'aide du simulateur SystemC. Pour ce faire, les modèles ont du être légèrement adaptés. Tout comme CASS, SystemC permet une simulation *cycle-true / bit-true*. Il est cependant plus lent, mais permet de mieux exporter les modèles, étant quasiment un standard pour ce qui est de la modélisation et simulation de System-On-chip.

7. Comparaison avec le Pi-Bus

La comparaison entre le Pi-Bus et le réseau SPIN ne s'est pas faite sur une application concrète comme celle présentée à la section validation. Cette comparaison [2] repose sur des tests de surcharge réalisés à l'aide de composants envoyant des rafales de requêtes sur plusieurs cibles. Ces composants sont appelés `GAP`, pour Générateur Analyseur de Paquets. La carte de test se compose de 32 abonnés équitablement répartis. On aura alternativement un initiateur (GAP) et une cible (RAM).
Les courbes représentées à la figure 7, représentent le nombre de cycles nécessaires pour l'envoi et la réception de 100000 paquets de 8 mots, en fonction de la charge offerte. Cette charge offerte (en pourcentage), est directement liée à l'intervalle moyen (en nombre de cycles) entre deux envois de paquets. Soit IM cet intervalle moyen et L la longueur d'un paquet. La valeur de la charge offerte est $L/(L + IM)$.
Le résultat de cette comparaison est que la saturation se produit plus tardivement avec un réseau SPIN. En effet, pour ce dernier, elle est obtenu à partir d'une charge de 28%, alors qu'avec un PI-Bus, c'est à partir de 4%.

[4] Cf Figure 6 : Système de test multiprocesseurs

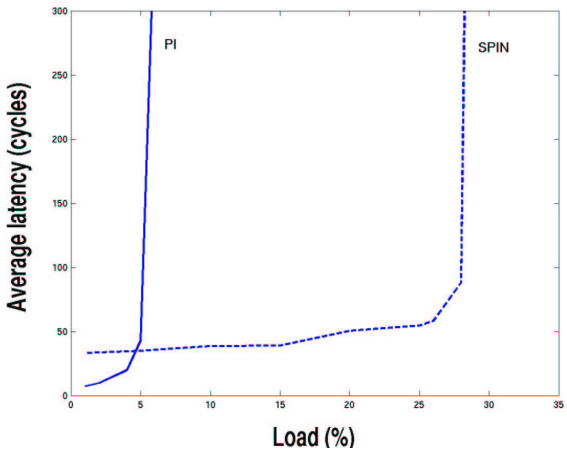

FIG. 7 – Courbe représentant la latence moyenne en fonction de la charge

8. Conclusion

On a présenté le réseau SPIN et vu qu'il était bien adapté pour transporter les paquets VCI sans dégradation de la sémantique VCI. Actuellement, afin de réduire le trafic sur le réseau, on étudie à la fois la prise en compte des signaux d'optimisation VCI, et le regoupement en cluster, d'initiateurs et de cibles liés.
Lors de la comparaison avec le PI-Bus, on a vu que le parallélisme coûtait très cher sur de petites architectures ou sur des réseaux peu chargés. La latence devenait alors un handicap.
Face au problème des inter-blocages, la solution de séparation des sous-réseaux requête et réponse est encore en cours de validation. Les simulations ont néanmoins été encouragentes.
Enfin, l'ensemble de ces travaux se trouve dans le domaine public, sous licence GPL.

Bibliographie

1. Pierre Guerrier et Alain Greiner – A Generic Architecture for On-Chip Packet-Switched Interconnections – Proceeding of the Design Automation and Test in Europe Conference 2000 (DATE 2000), March 2000.
2. Adrijean Adriahantenaina, Herve Charlery, Alain Greiner, Laurent Mortiez et Cesar Albenes Zeferino – SPIN : a Scalable, Packet Switched, On-chip Micro-network – Proceeding of the Design Automation and Test in Europe Designers' Forum 2003 (DATE 2003), Munich, March 2003.
3. John Hennessy et David Patterson – Computer Architecture, A Quantitative Approach - 2nd Edition – Morgan Kaufmann Publishers, San Francisco, CA, USA, 1996.
4. Hugo de Man – Education for the Deep Submicron Age : Business As Usual ? – Proceedings of the 34th Design Automation Conference, Anaheim, USA, March 1997.
5. Virtual Socket Interface Alliance – Virtual Component Interface Standard - Draft Specification, v. 2.2.0 – http ://www.vsia.com (document access may be limited to members only), August 1997.
6. ARM Company Ltd – Advanced Microprocessor Bus Architecture Specification – http ://www.arm.com/Pro+Peripherals/AMBA/, 1997-99.
7. Open Microprocessor Systems Initiative – PI-Bus Draft Standard Specification – ftp ://www.omimo.be/ftp/standard/omi324.ps, 1994.
8. Gerard J. Holzmann – The Model Checker Spin – IEEE transaction on software engineering, vol. 23, n?5, May 1997.
9. Denis Hommais et Frederic Petrot – Efficient Combinational Loops Handling for Cycle Precise Simulation of System on a Chip – IEEE Euromicro Conference, pp. 51-54, 1998.
10. Frederic Petrot et al. – Cycle-Precise Core based Hardware/Software System Simulation with Predictable Event Propagation – IEEE Euromicro Conference, pp. 182-187, 1997.