Réseau d'Interconnexion pour les Systèmes sur Puce : le Réseau HERMES

Séverine Riso, Lionel Torres, Gilles Sassatelli, Michel Robert

Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier
Université de Montpellier II / CNRS
161, rue Ada – 34392 Montpellier Cedex 5, France
Email: <lastname>@lirmm.fr

Fernando Moraes

FACIN PUCRS
Av. Ipiranga, 6681 – Prédio30/ BLOCO 4
90619-900-Porto Alegre-Brasil-
Email: moraes@inf.pucrs.br

RESUME

Les connexions dédiées et les bus ne seront probablement plus utilisées dans cinq ou dix ans, à cause de leur manque de flexibilité et de leur faible scalabilité. Ces interconnexions seront remplacées par des réseaux sur puce (NOC Network On Chip) à commutation de paquet et une synchronisation de type GALS (Globalement Asynchrone Localement Synchrone). Apres un rapide état de l'art, nous étudions un réseau maillé à commutation de paquet appelé Hermes. Nous montrerons les performances obtenues par ce type de réseau, notamment en terme de latence et de débit.

1. Introduction

L'interconnexion de composants électroniques a été de tout temps une préoccupation majeure dans la réussite et la conception d'un système électronique performant.

S'il y a une décennie, l'interconnexion se faisait plus au niveau composant élémentaire par l'intermédiaire des technologies PCB (circuits imprimés) ; nous sommes maintenant face à une problématique nouvelle, où l'interconnexion se fait à l'intérieur du circuit (intra-chip) avec des dizaines de cœurs (IP Intelectual Property) à connecter. A titre d'exemple, l'ITRS prévoit en 2010 l'intégration de systèmes électroniques de 4 milliards de transistors pour des fréquences proches de 10 GHz [1].

De ce fait, le goulot d'étranglement des SOCs (*System On Chip*) réside dans la gestion des communications entre les blocs.

La communication entre les cœurs peut se faire par des interconnexions de type point à point (fils dédiés). Cette solution est adéquate pour des communications entre cœurs demandant des débits de transfert élevés. Une autre solution utilisée traditionnellement est de connecter l'ensemble des cœurs par un ou plusieurs bus. Mais cette approche (comme nous le détaillerons dans l'état de l'art) montre ses limites dans les SOCs actuels [2]-[5].

C'est pourquoi les réseaux sur puce (appelés NOC pour Network On Chip) semblent être une solution appropriée pour gérer la communication entre les blocs. La difficulté de la conception d'un NOC réside dans un compromis entre une Qualité de Service (QoS) optimale, une bande passante élevée et une flexibilité d'utilisation importante, tout en limitant la consommation d'énergie et de surface de silicium.

Comme le montre R. Lauwereins [6] avec les variations de la surface de la zone isochrone en fonction de la fréquence du circuit, la synchronisation d'un circuit avec une seule source d'horloge deviendra très difficile voire impossible. Le modèle de synchronisation des futures puces sera probablement de type « Globalement Asynchrone Localement Synchrone » (GALS) : avec la possibilité d'utiliser plusieurs domaines isochrones (phases et domaines d'horloge différents).

L'objectif de ce papier est de montrer qu'il est possible d'envisager de nouvelles topologies d'interconnexion permettant de s'affranchir d'une part de la synchronisation globale d'ensemble du circuit et d'autre part d'augmenter sensiblement les performances externes en termes de débit. Après un aperçu de l'état de l'art de ce domaine, nous proposerons une nouvelle architecture de réseau d'interconnexion appelée Hermes. Le paragraphe 4 met en évidence les performances (latence, débit) de ce type de réseau en le comparant à d'autres réseaux connus tel que le SPIN et le Pi-Bus.

2. Structures d'interconnexion dans les SOCs

Trois techniques majeures d'interconnexions sont, aujourd'hui, disponibles. Il s'agit de :

- La communication point à point
- Le bus partagé
- Une technique émergente: réseaux sur puces (NOC, Network On Chip).

Les caractéristiques principales de ces différentes approchent sont énumérées dans le tableau 1.

	Point à Point	Bus Partagé	NOC
Parallélisme	++	-	++
Consommation	+	-	++
Scalabilité	--	+	++
Réutilisation	--	++	++
Qualité de Service	++	+	+

***Tableau 1** : Comparaison des interconnexions*

++ : très bon **+ : bon**
- : mauvais **-- : très mauvais**

2.1. La communication point à point

La communication point à point permet un parallélisme contrôlé. En effet, chaque connexion est dédiée et spécifique à la communication entre deux cœurs. Cela permet ainsi d'établir des connexions à fort débit et d'optimiser le nombre de transfert et éventuellement la consommation d'énergie. Les problèmes essentiels de ce type d'approche sont : la faible possibilité de réutilisation (du fait de sa spécificité), le faible nombre de cœurs qui peuvent adresser directement, son manque de scalabilité. Ce type de connexion étant déterministe, il est aisé de garantir une qualité de service optimale.

2.2 Les bus

Un bus partagé est un ensemble de connexion autour duquel communiquent plusieurs cœurs [9]. Il est réutilisable et souple d'utilisation. Un cœur peut être ajouté, déplacé ou supprimé facilement. Un bus partagé est uniquement composé de lignes de données, de lignes de contrôle et d'un arbitre. Les lignes de données transportent les informations entre une source et une cible. Les lignes de contrôle transportent les signaux de requête et d'acquittement ainsi que des informations concernant les données. L'arbitre désigne le maître en droit d'utiliser les services du bus. En effet, une seule communication à la fois peut avoir lieu à un instant donné.
Du fait de la longueur du bus et du nombre de cœurs connectés, des phénomènes électriques sont aussi à prendre en compte : phénomène résistif et capacitif important, consommation d'énergie (due à la distribution de l'arbre d'horloge), couplage entre lignes (crosstalk), etc … Ces phénomènes, parfois difficilement maîtrisables, sont un frein majeur à l'intégration de bus de longueur importante.

Il existe trois types de bus : les bus backplane, les bus processeur-mémoire et les bus entrée/sortie. Le bus backplane est le bus partagé grâce auquel les processeurs, les mémoires et les blocs entrées/sorties communiquent. Afin de faciliter les communications entre les processeurs et les mémoires, le bus processeur-mémoire possède une grande bande passante. Ce bus est optimisé pour les transferts de données vers les mémoires caches. Le bus entrée/sortie est un standard industriel. Il est habituellement plus long et plus lent que les précédents et permet d'adapter un grand nombre d'élément entrée/sortie. Il se connecte au bus processeur-mémoire ou au bus backplane.

Ces types de bus se retrouvent dans plusieurs structures d'interconnexions. La structure à un seul type de bus utilise des bus backplane tels que le PI-bus de OMI (Open Microprocessor systems Initiative) ou le SiliconBackplane de SONICS [10]. L'allocation de bande passante des cœurs dans le Silicon Backplane se fait par une technique de type TDMA (Time Division Multiple Access).

Les structures à deux types de bus utilisent des bus backplane ou des bus processeur-mémoires et des bus entrés/ sorties. Les différents bus sont reliés par un pont et possèdent leur propre arbitre. Le ST-bus de STmicroelectronics et le bus ARM de AMBA utilisent des structures à deux bus qui permettent de décharger la bande passante du bus principal.
Tous les cœurs partagent la même bande passante dans le système, les bus ont donc une scalabilité limitée à une vingtaine de cœurs. En utilisant des bus séparés ou hiérarchiques les architectures bus réduisent leurs contraintes. Mais les techniques utilisées deviennent de plus en plus complexes.

2.3 Les NOCs

Les réseaux sur puce apparaissent comme une solution pour interconnecter les cœurs à la place des bus. Un NOC est conçu sur un modèle simplifié de trois couches qui sont la couche physique, la couche architecture-contrôle et la couche protocole [3].

La couche physique est la couche de plus bas niveau d'un NOC. Elle permet de définir les canaux utilisés au niveau physique. Lors de la conception de cette couche il faut choisir la façon d'interconnecter les ports d'entrées et de sorties entre eux (type crossbar ou type switch) qui seront les meilleurs en fonction de la

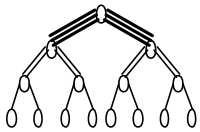

Figure 1a: Structure Fat Tree

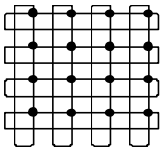

Figure 1b: Réseau torus

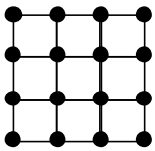

Figure 1c: Réseau maillé 2D

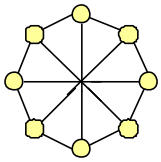

Figure 1d: Chordal Ring

***Figure 1:** Quelques topologies de NOC*

longueur des fils et de la surface[7][8].
La couche contrôle gère les communications entre les switchs. Elle détermine le type de synchronisation utilisé. La topologie (maillée, anneau, tore..) du réseau est définie à ce niveau ainsi que les algorithmes de routage et d'arbitrage.
La couche protocole est la couche de plus haut niveau. Elle gère les communications entre les cœurs. Le protocole de communication (OCP, VCI..) et donc les wrappers sont fixés à ce niveau.
De nombreuses topologies sont actuellement disponibles telles que les architectures maillées, *fat tree*, *torus* et *chordal ring* présentées dans la figure 1.
Les topologies matricielles (2D, maillées) semblent dominer le monde des réseaux sur puces, pour des raisons d'implémentation et de connexion (distances courtes de switch en switch, éliminant ainsi un certain nombre de phénomènes électriques en comparaison aux bus) :

- Simplicité de routage des données.
- Scalabilité de la structure.

Cependant, cette architecture présente des temps de latence souvent supérieurs à ceux du type *fat tree* ou *chordal ring*. Ces différentes topologies se retrouvent dans la littérature comme T. Maresceaux [*15*] qui profite d'un réseau 2D ou Octagon[*20*][*21*] et Proteo [*16*][*17*] qui sont construits à base d'un anneau ou encore le SPIN32 [*11*]-[*14*] qui utilise une topologie fat-tree. Une des approches qui a retenu notre attention est celle proposée par [*11*]-[*14*] pour le SPIN32.

Le SPIN (Scalable Programmable Integrated Network) est un réseau sur puce à commutation de paquet, utilisant un routage de type whormhole (à trou de vers). Il est basé sur une topologie en arbre quaternaire (Fat-tree).
La figure 2 illustre un arbre quaternaire avec 32 ports utilisé dans le SPIN32. Les liens sont bidirectionnels et full-duplex. Le routage des paquets est adaptatif. L'élément de base du SPIN est le routeur RSPIN. Il est composé de huit ports bidirectionnels, de buffers sur les ports d'entrée et de sortie et d'un crossbar 10x10. Le SPIN dispose de wrappers VCI entre le réseau et les IPs qui permettent les conversions entre les deux protocoles de communication VCI et SPIN. Le réseau SPIN32 est composé de 16 routeurs, soit deux rangées de huit routeurs RSPIN. Pour un procédé de fabrication 0.13micron de ST Microelectronics, le réseau occupe une surface totale de 3.9mm^2. La bande passante d'un tel réseau permet de fournir à une source et une cible une bande passante maximum de 3 GBit/s, dans une hypothèse de fonctionnement à 200MHz.

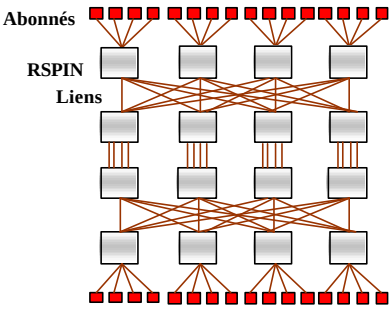

Figure 2: *Représentation du SPIN32*

Par la suite nous comparerons cette approche avec le réseau Hermes qui fait l'objet d'une présentation détaillée dans la section suivant.

3. Switch Hemes

Le projet Hermes utilise un réseau maillé à commutation de paquet dont l'élément de base est un switch présenté ci dessous [*19*].

3.1 Le switch

Le switch est composé de cinq ports bidirectionnels (Est, Ouest, Nord, Sud et Local). Le port Local est relié au cœur alors que les autres ports sont reliés aux switchs voisins. Chaque port d'entrée possède un buffer pour stocker les données provisoirement. Le bloc de contrôle est formé d'un arbitre et d'un routeur, comme le montre la figure 3.

Ce switch utilise un mode de routage de type *wormhole* car il requiert moins de mémoire et engendre une latence plus faible. Les paquets circulant dans le réseau contiennent les données et un en-tête pour les informations de routage. Cet en-tête est composé de l'adresse de destination et d'un compteur indiquant le nombre de mots présents dans le paquet. Les tailles des paquets, des mots et aussi des buffers sont paramétrables. Ce switch a été conçu sur un réseau maillé mais il peut être employé pour d'autres topologies (tore, tore replié et anneau). Chaque switch possède une adresse unique. Afin de simplifier le routage dans le réseau, cette adresse est exprimée en XY, où X représente sa position horizontale et Y sa position verticale.

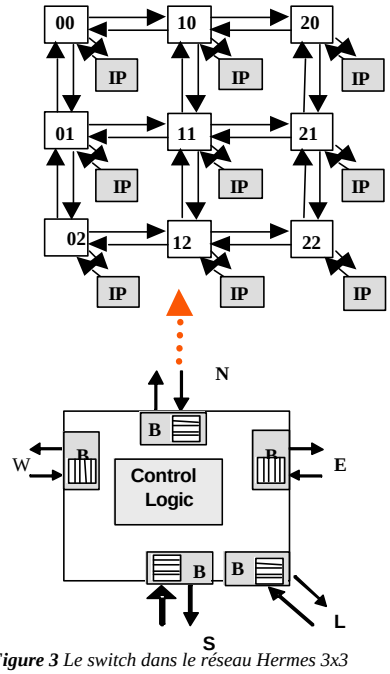

Figure 3 *Le switch dans le réseau Hermes 3x3*

3.2. Le bloc contrôle

Quand un switch reçoit un nouvel en-tête de paquet, l'algorithme d'arbitrage est exécuté. Cet algorithme est utilisé pour garantir l'accès à un port de sortie lorsqu'un ou plusieurs ports d'entrées requièrent simultanément une connexion. Il utilise un algorithme à priorité du type « round robin ». C'est-à-dire que le port le plus prioritaire deviendra le moins prioritaire au coup suivant. Cette méthode garantie que toutes les requêtes seront traitées et évite un conflit de type *starvation* (les données restent bloquées dans le buffer du switch).

Une fois que l'algorithme d'arbitrage est exécuté et si la requête est accordée, alors un algorithme de routage XY est accompli afin de connecter le port d'entrée au bon port de sortie. L'arbitre attend dix cycles d'horloges s'il ne reçoit pas de réponse du routeur pour traiter une nouvelle requête. Cette période correspond au temps nécessaire pour exécuter l'algorithme de routage.

Un algorithme de routage doit libérer le circuit de *deadlock* (la perte des données après un certain nombre de coup d'horloge). Les algorithmes de routage se classent dans trois catégories [18]. Dans un routeur, un algorithme non-adaptatif renvoie un paquet vers un seul nœud. Cet algorithme route donc un paquet de sa source à sa cible par un chemin prédéterminé. Un algorithme partiellement adaptatif permet de router les paquets parmi un sous-ensemble de chemin déterminé suivant certaines règles. Donc cet algorithme ne permet pas de router les paquets à travers tous les chemins d'un réseau. Un algorithme adaptatif autorise un paquet à être renvoyé vers tous les nœuds de destination. Donc cet algorithme accepte de router les paquets à travers tous les chemins d'un réseau.
Un algorithme de routage pleinement adaptatif a tendance à créer des *deadlock* dans les réseaux [19]. Lors d'un deadlock, les données tournent dans le réseau, sans trouver leur cible. Le réseau Hermes peut être utilisé avec des algorithmes de routage soit non adaptatif soit partiellement adaptatif. L'algorithme de routage reste en mode XY, c'est-à-dire que le paquet est routé en priorité selon l'axe horizontal.

3.3. Les buffers

Lorsqu'un mot est bloqué dans un routeur donné, les performances du réseau en sont affectées, tant que les mots appartenant au même paquet sont bloqués dans d'autres routeurs. Pour diminuer ces effets sur les performances du réseau, des buffers sont placés sur les ports d'entrées. Ces buffers sont des FIFOs dont la taille est paramétrable.

La taille des buffers est un paramètre important qui contribue à diminuer les blocages du réseau. En contre partie les buffers de tailles trop importantes ajoutent un coût en surface silicium non négligeable. La section suivante présente différente validations mettant en exergue ces paramètres.

4. Validation

La topologie du réseau dépend de la manière de connecter les switchs entre eux. Le réseau Hermes étudié connecte les switchs entre eux avec des liens bi-directionnels. Dans un réseau maillé, que nous utilisons pour notre étude, le nombre de ports par switch dépend de son emplacement dans le réseau.
Cette partie présente des résultats de simulations VHDL obtenus avec l'outil NCLaunch de Cadence. Deux définitions importantes doivent être données avant de présenter les résultats de simulation :

La Latence
La latence correspond au temps de propagation d'un signal. Par extension, la latence est le temps, en coups d'horloge, que met un paquet pour être routé de sa source à sa destination.

Un paquet
Dans notre cas, un paquet est composé d'un en-tête avec l'adresse de la cible et le nombre de mot (ou flit) à envoyer puis les données découpées en mot.

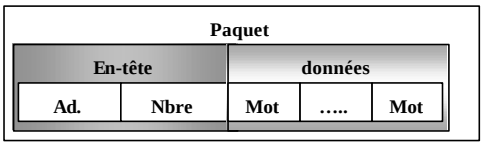

Figure 4 : *Composition d'un paquet du réseau Hermes*

4.1. Variation de la latence

Le but de cette simulation est de visualiser les variations de la latence en fonction de la taille des buffers et de la taille des paquets pour différent nombre de switchs traversés.

Le protocole
Cette simulation se place dans le cadre d'une matrice 5x5 de switch Hermes. Dans un premier temps 30 mots dans un même paquet traversent le réseau d'une source vers une cible.

Les résultats
La figure 5 montre les résultats obtenus. La latence varie de façon

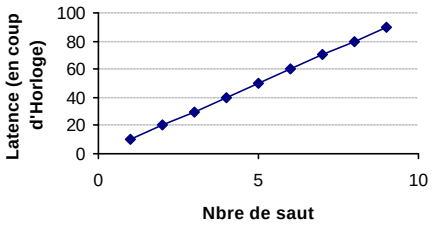

Figure 5 : *Courbe de variation de la latence(saut=nombre de switch traversés)*

linéaire. Ceci s'explique par le fait qu'un pipeline se créé dès le traitement des données de l'en-tête. Le temps d'exécution des algorithmes de routage et d'arbitrage est le même pour tous les ports. La latence ne varie ni en fonction de la taille des buffers ni

en fonction de la taille des paquets, mais uniquement en fonction du nombre de switch traversé.

4.2. Taille des buffers

Le protocole

Dans ce paragraphe nous cherchons à vérifier que la taille des buffers influe sur le débit de données dans le réseau Hermes. Comme précédemment le réseau se présente sous forme d'une matrice 5x5. Tous les cœurs émettent des données à des cibles choisies aléatoirement. Pour différentes tailles de paquet, nous faisons varier la taille des buffers et mesurons le temps nécessaire pour recevoir tous les paquets. Ces données sont découpées en 2 paquets de 15 mots puis en 15 paquets de 2 mots.

Les résultats

Sur la figure 6, nous remarquons que le réseau met moins de temps pour acheminer 2 paquets de 15 mots que 15 paquets de 2 mots. L'établissement de la connexion ne se fait qu'une seule fois pour un paquet de taille importante et s'opère plusieurs fois pour des paquets de faible dimension. Avec des paquets de taille importante nous augmentons les chances de collision.

Pour des buffers de 6 mots ou plus, le temps de délivrance ne varie presque plus. Si les buffers sont trop petits, le switch ne peut plus recevoir de nouveau mots jusqu'a ce que le port de destination soit choisi. C'est pourquoi, la taille minimum des buffers doit être égale au nombre d'opérations d'écriture possible pendant l'exécution des algorithmes d'arbitrage et de routage. Dans le réseau Hermes, ces algorithmes utilisent 10 coups d'horloges et chaque opération d'écriture prend 2 coups d'horloge. Considérant que le mot d'en-tête est dans le buffer pour être traité, la taille minimum des buffers est alors de six mots. L'augmentation de la taille du buffer entraîne un accroissement de la surface du NOC. Ainsi, en dimensionnant un buffer, le concepteur doit faire un compromis entre le temps de délivrance et la surface.

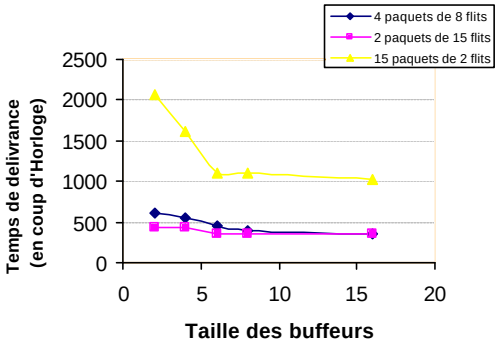

Figure 6 : *Temps de délivrance en fonction de la taille des paquets*

4.3. Latence moyenne

Le protocole

La troisième expérience concerne le comportement du NOC en fonction de sa charge. La matrice 4x4 utilisée est séparée en deux. D'un coté, il y a huit sources et de l'autre huit cibles. Tous les maîtres envoient des requêtes vers tous les esclaves. Soit, G le temps entre deux requêtes émises par la même source et GM la moyenne des G pour tous les maîtres. Si L est la taille des paquets, alors la charge se défini par : L / (L+GM) [13]. Pour une charge à 100%, GM doit être nul. Pour l'expérience, 64 paquets de 16 mots circulent dans le réseau construit avec des buffers de 3 mots. Ceci dans le but de saturer le plus vite possible le réseau.

Les résultats

La figure 7 présente la latence moyenne en fonction de la charge pour le réseau Hermes, le SPIN32 et le PI-bus. Les résultats du SPIN32 sont obtenus pour un transfert de 100.000 paquets dans le réseau. Les latences du Pi-Bus et du SPIN32 divergent au-delà d'une certaine charge, ce qui n'est pas le cas pour le réseau Hermes.

Pour de très petites charges, le Pi-bus a une latence plus petite que le SPIN32 et le réseau Hermes. Cependant la saturation du PI-BUS intervient à partir de 4%, avant celle du SPIN32 qui apparaît pour une charge supérieure à 28%. Cependant ces résultats doivent être confirmés pour des transferts de paquets plus important pour le NOC Hermes.

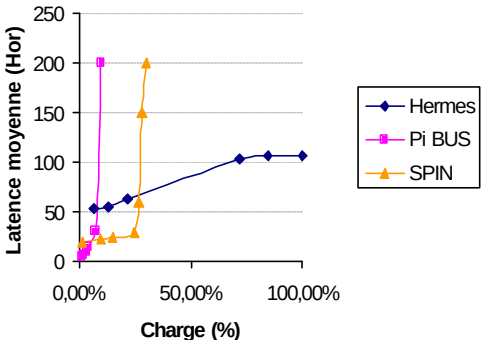

Figure 7 : *Latence moyenne en fonction de la charge*

5. Evaluation de temps de délivrance

La qualité de service est un critère important à prendre en compte dans le cas de l'utilisation d'un NOC. En effet, dans notre approche GALS, le problème essentiel est de pouvoir prendre en compte des contraintes sur un temps de délivrance maximum d'un cœur vers un autre. Pour illustrer ce problème, considérons un réseau maillé 5*5 avec 25 sources et 25 cibles, chaque source émet 20 paquets de 20 mots, dans le cas d'un bus partagé idéal, le temps de transfert de l'ensemble des données serait de 10000 cycles. En considérant des buffers de 8 mots, le tableau 2 résume les performances du réseau.

	Trafic 1	***Trafic 2***	***Trafic 3***	*Moyenne*
Moyenne	***201***	***195***	***190***	*195*
Ecart type	***121***	***109***	***109***	*113*
Minimum	***51***	***51***	***51***	*51*
Maximum	***768***	***717***	***787***	*757*
Temps total	***3035***	***2816***	***3114***	*2988*

Tableau 2: *Temps de délivrance du NOC en coup d'horloge*

Les trafics 1, 2 et 3 correspondent à des trafics envoyés de manière aléatoire. Globalement, nous constatons un facteur 3 en terme de débit (2988 cycles contre 10000 cycles pour un bus idéal). Cependant, on constate des dispersions importantes en terme de temps d'arrivée des paquets. En effet, 3% des paquets ont un temps de délivrance supérieur à la moyenne. Ceci peut être pénalisant pour certaines applications. Il est donc nécessaire de mettre en œuvre des techniques supplémentaires pour atténuer cette dispersion et surtout borner le temps de délivrance d'un ensemble de paquet. Ces techniques sont appelées qualité de service.

6. Implémentation

Un premier prototype de ce NOC a été réalisé sur un FPGA du type Xilinx virtex 2. Avec des tailles de buffer de 8 mots et une matrice 2x2.
Le tableau 3 montre qu'il est tout à fait envisageable d'utiliser ce réseau dans le cadre d'un SOC. En effet, le nombre de porte équivalente est relativement faible.

	FPGA Xilinx Virtex II		**CMOS 0.35µm**
	LUTs	**FFs**	**ASIC (# portes)**
1 Switch	631	200	2930
SR	193	233	1986
Serial	91	93	752
Total	590	596	4587

Tableau 3: Consommation en surface du NOC

SR et Serial sont des modules supplémentaires nécessaires à la connexion d'un switch avec un cœur (wrapper du cœur).

7. Conclusion

Ce travail présente une comparaison entre trois types d'interconnexion sur puce : un système classique avec un bus et deux réseaux sur puce. Pour de très petites charges, le système utilisant un bus est le plus performant au regard de la latence. Nous avons vu que les bus consomment peu de surface et sont très flexibles tant que le système ne dépasse pas une vingtaine de cœur. Au-delà, le système doit router plus de données et les solutions à bus classiques ne sont plus satisfaisantes. Les réseaux sur puce tels le SPIN32 ou Hermes sont plus appropriés aux systèmes supportant un grand nombre de cœurs. La topologie maillée, les algorithmes de routage XY et d'arbitrage dynamique permettent au réseau Hermes de ne pas saturer. Des études complémentaires sont en cours afin d'évaluer d'autres types d'algorithme de routage et d'arbitration dans le but d'implémenter la qualité de service de ce type de réseau.

REFERENCES

[1] International Technology Roadmap for Semiconductors *http://public.itrs.net/*
[2] Benini L; et al. "Powering Network on Chip", ISSS'2001, pp.33-38.
[3] Benini L; et al. "Network on Chips: A New SOC Paradigm". IEEE Computer, 35, Jan. 2002, pp70-78.
[4] Dally, W.J.; Towles, B. "Route Packets, Not Wires: On-Chip Interconnection Networks" DAC'2001, pp 684-689.
[5] Guerrier P., Greiner A., "A Generic Architecture for on-chip Packet Switched Interconnections" Date'2000, pp 250-256.
[6] Lauwereins R. "Creating a world of Smart Re-configurable Devices", Field Programmable Logic FPL'2002, pp790-794. [7] Lemieux G., et al "Generating Highly-Routable Sparse Crossbars for PLDs", FPGA'2000, pp 155-164.
[8] Lemieux G., et al "Analytical Framework for Switch Block Design " ; FPL'2002, pp 122-131.
[9] Rabaey J., "Busses and Networking", Computer Science 252, Spring 2000.
[10] http://www.sonicsinc.com/
[11] Leiserson C., "Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing", IEEE Transactions on Computers, vol. C-34, no. 10, pp 892-901, October 1985.
[12] Greiner A., Andriahantenaina A. "Micro-réseau pour systèmes intégrés: réalisation d'un réseau SPIN à 32 ports", GDR CAO'2002, pp71-74.
[13] Andriahantenaina A., et al. "SPIN : a Scalable, Packet Switched, On-chip Micro-network", DATE'2003, pp.1128-1129
[14] Charlery H., "SPIN, un micro-réseau d'interconnexiotn à commutation de paquets respectant la norme VCI. Concepts généraux et validation ", SympAAA'2003, pp.337-344
[15] Maresceaux, T.; Bartic, A.; et al. "Interconnection Network Enable Fine-Grain Dynamic Multi-tasking on FPGAs." FPL'2002. Sept. 2002; pp795-805.
[16] Sigüenza-Tortosa D., Nurmi J. "Proteo : A New Approach to Network-On-Chip" IASTED'2002, pp.355-357.
[17] Alho M., Nurmi J. "Implementation of Interface Router IP for Proteo Network-On-Chip ", DDECS'2003.
[18] Christopher, J. Glass and Lionel M. Ni, "The Turn for Adaptative Routing". Association for Computing Machinery ACM 1994, Vol 41, No 5, pp 874-902.
[19] Moraes, F. G.; Mello, A. V. de; Möller, L. H.; Ost, L.; Calazans, N. L. V.. "A Low Area Overhead Packet-switched Network on Chip: Architecture and Prototyping.", IFIP VLSI SOC 2003, 2003, Darmstadt. International Conference on Very Large Scale Integration. 2003.
[20] Karim,F.; Nguyen A.; Dey S. "An interconnect architecture for network systems on chips", IEEE MicroV22(5), sept-oct 2002, pp36-45.
[21] Karim,F.; Nguyen A.; Dey S.; Rao R ."On chip communication architecture for OC-768 network processors" 38th Design Automation Conference (DAC'01), Jun 2001, pp 678-683.