

Initiation à la conception d'un réseau de communication sur puce (*Network on Chip*) tolérant aux fautes

Cédric Killian, Camel Tanougast, Fabrice Monteiro, Camille Diou et Abbas. Dandache

CNFM – Pôle Grand Est : MIGREST

Laboratoire des Interfaces Capteurs et Microélectronique (L.I.C.M.)

Université Paul Verlaine de Metz, LICM-ISEA, 7 rue Marconi, 57070 Metz Technopole

Email : {cedric.killian, camel.tanougast}@univ-metz.fr

1. INTRODUCTION ET CONTEXTE

Cet article présente un projet pédagogique d'initiation à la détection d'erreurs dans un réseau de communication sur puce (*NoC - Network on Chip*) pour la conception de *NoC* tolérant aux fautes, mené avec les étudiants de master 2 **GEII** (*Génie Electrique et Informatique Industriel*) - parcours **RSEE** (*Radiocommunication et Systèmes Electronique Embarqués*) de l'Université Paul Verlaine de Metz. Etant donné l'évolution rapide et de plus en plus complexe des systèmes sur puce multi-processeurs (*MPSoC - Multiprocessors SoC*), l'interconnexion de communication des modules (*IP - Intellectual Property*) constituant ces systèmes constitue une partie fondamentale lors de la conception de tels systèmes. En effet, elle doit répondre à des contraintes de performance et de coût liés à la complexité et l'augmentation croissante de modules ou d'IPs interconnectés. Actuellement un tel réseau de communication sur puce met en œuvre des transmissions de données par paquets vers les nœuds interconnectés au réseau correspondant aux modules ou IPs intégrés au système (processeurs, mémoires, contrôleurs de périphériques reliés, etc.). Cette transmission est réalisée à travers des routeurs (constituant le réseau) en mettant en œuvre des règles d'aiguillage et de routage des paquets de données dans le réseau. Généralement, les performances d'un *NoC* sont exprimées en termes de bande passante, de latence, de dissipation de puissance et de fiabilité. Cette dernière permet de mettre en avant les aspects et les contraintes liés à la sûreté de fonctionnement d'un *NoC* (détection d'erreurs des données, utilisation d'algorithmes adaptatifs d'acheminement des paquets) et le surcoût lié aux solutions de mise en œuvre d'une tolérance aux fautes. En effet, les fautes permanentes, transitoires ou temporaires affectent la fiabilité des interconnexions d'un *MPSoC* [1], entraînant une altération du comportement du *NoC* et donc une dégradation de ses caractéristiques et performances de qualité de service. Ces types de faute sont donc critiques pour le fonctionnement de systèmes sur puce à base de *NoC*. Traditionnellement, les mécanismes de détection et de correction d'erreurs sont utilisés pour protéger une structure de communication contre les effets transitoires de dysfonctionnement. Les concepteurs doivent précautionneusement peser le coût d'implantation de ce type de mécanisme pour les infrastructures de communication de données sur puce par rapport aux réels bénéfices qu'ils peuvent apporter. C'est dans ce contexte que nous proposons aux étudiants du parcours **RSEE** un projet d'initiation à l'intégration de concepts de tolérance aux fautes dans un réseau *NoC* à l'issue d'un projet de développement, de modélisation et de simulation d'un Réseau *NoC* [3].

2 CONCEPTION DE ROUTEURS *NoC* TOLERANTS AUX FAUTES

2.1. ETUDE DE CAS : détection et correction d'erreurs dans une structure **MESH 4x4**

A partir d'un fichier contenant la description comportementale *VHDL* d'une structure *NoC* de topologie maillée (*Mesh 4x4*) et d'algorithme de routage *X-Y*, nous proposons aux étudiants de modifier l'architecture interne des routeurs afin d'y intégrer des modules de détection et de correction d'erreurs de données. Les solutions apportées par les étudiants sont globalement libres mais doivent valider en

termes de qualité les détections de fautes par simulation et évaluer le coût en termes de ressources logiques et de performance induites par les modifications architecturales des routeurs du réseau.

A. Description du Model VHDL comportemental de routeurs interconnectés selon une structure MESH 4x4

Le projet repose initialement sur une description comportementale VHDL d'un réseau NoC et de ses routeurs fournis aux étudiants. Il s'agit d'une description d'un réseau de topologie maillée de taille 4x4 dont l'architecture structurelle d'un routeur est détaillée en Figure 1. Chaque nœud de routage dispose de quatre directions (*North*, *South*, *East* et *West*) et des interconnexions unidirectionnelles permettant l'envoi et la réception simultanée de paquets de données issus du réseau. Une latence de transmission du routeur proposé est de 2 cycles d'horloge de simulation. La structure des paquets de données à transmettre dans le réseau, les règles de contrôle et de transmission des paquets à travers le model du réseau proposé sont spécifiés aux étudiants et décrits ci-dessous.

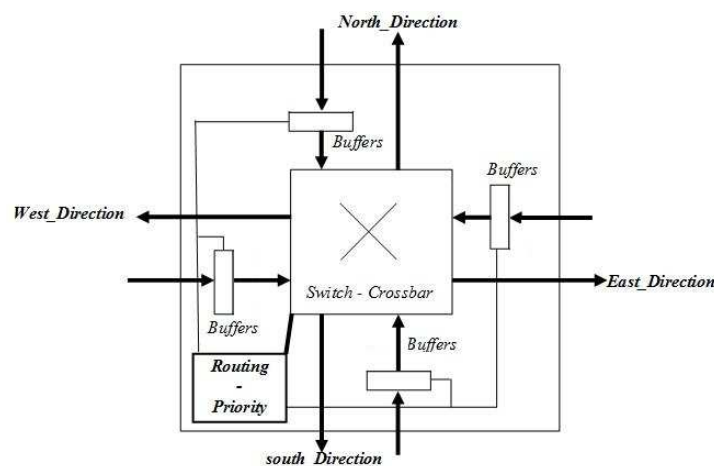


Fig. 1. Architecture d'un routeur NoC.

Structure des messages : Afin de faciliter les règles d'échanges des paquets de données entre les routeurs, les paquets sont composés d'un seul *flit* (mot de données de taille fixe paramétrable). Le réseau étant constitué de 16 nœuds selon un acheminement des paquets de sur les axes X et Y du réseau, un paquet de données est alors constitué de 4 bits d'adressages (2 bits pour la position du nœud selon l'axe X et 2 bits pour la position selon l'axe Y). Un paquet contient 4 bits de données. La figure 2 illustre la structure d'un paquet de données. La taille générale du message est paramétrable via une déclaration générique afin de permettre l'intégration de données de contrôle en vue de l'intégration de concepts de tolérance aux fautes sur les données circulant dans le réseau (voir §.2.2.).

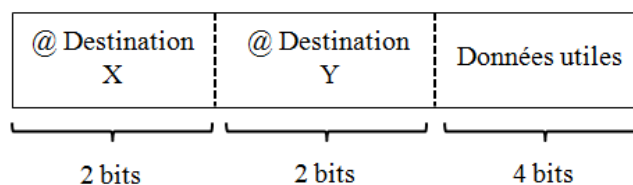


Fig. 2. Structuration d'un paquet de données du réseau NoC –Mesh 4x4.

Technique d'aiguillage : La technique d'aiguillage des messages entre les routeurs s'effectue par commutation de paquet (*packets switching*) de type *Store and Forward* [2]. Cela signifie qu'un paquet ne peut pas être transféré vers un autre routeur tant que ce dernier ne peut le recevoir dans son intégralité.

Contrôle de flux : Le contrôle de flux des données entre les routeurs est de type *ACK/NACK* [2]. Plus précisément, une copie de la donnée à transmettre est gardée dans le routeur émetteur (buffer local) jusqu'à ce que le routeur destinataire valide la réception de la donnée. Dans le cas d'une validation positive (*ACK*), la copie est supprimée. Dans le cas contraire (*NACK*) la donnée est retransmise. Cependant, dans la version fournie du routeur, il n'y a aucun contrôle de la validité de transmission des paquets. Le routeur est initialement configuré pour effectuer des acquittements permanents. C'est aux étudiants de compléter ce bloc de contrôle (les signaux de contrôle du flux sont déjà présents) afin de demander une retransmission dans le cas d'une réception de données erronées.

Algorithme de routage et technique d'arbitrage : L'algorithme de routage initialement implanté dans les nœuds du réseau est de type *XY* déterministe [3]. La technique d'arbitrage est celle d'une priorité à droite [3]. Lorsqu'un routeur reçoit simultanément plusieurs paquets, le paquet le plus à droite est routé en priorité comme illustré en Figure 3. Si quatre paquets arrivent simultanément dans un routeur, c'est une direction prioritaire préalablement définie par l'utilisateur qui est routé en premier. Dans notre cas d'étude, la direction *EAST* est définie comme prioritaire dans ce cas de figure.

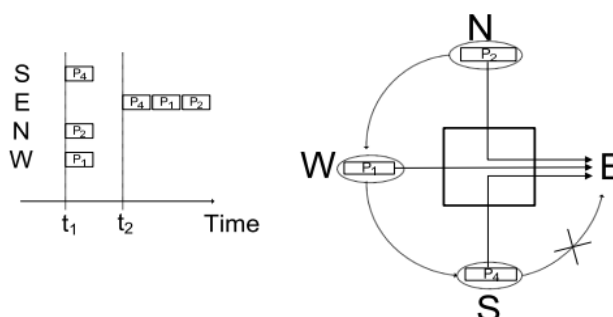


Fig. 3. Illustration de l'arbitrage par « priorité à droite ».

La logique de routage étant centralisée (au sein du routeur), elle ne peut router qu'un seul paquet à la fois. Dans le cas où plusieurs paquets arrivent simultanément, la logique de contrôle du routeur active des signaux indiquant à ses routeurs voisins qu'il est occupé et qu'il ne peut pas recevoir temporairement de paquets de données. Un routeur souhaitant transmettre un paquet doit donc attendre que le nœud destinataire soit disponible.

B. Structure fautive à erreurs intégrées du *NoC*

L'objectif de ce projet est de sensibiliser les étudiants aux différents types d'erreurs qui peuvent survenir pendant le fonctionnement d'un *NoC*, notamment les erreurs permanentes et transitoires. En effet, il est important de distinguer une erreur temporaire d'une erreur transitoire. Car après détection du type d'erreur, la stratégie de mise en œuvre d'une tolérance aux fautes sur l'élément fautif est différente. Par exemple, isolation définitive du nœud comportant une faute permanente, ou demande de retransmission dans le cas d'une faute transitoire.

Le réseau *NoC* fautif proposé aux étudiants contient deux blocs *IPs* effectuant des envois périodiques (tous les 10 cycles d'horloge) de paquets de données d'un *IP1* vers un *IP2*. Plusieurs erreurs ont été introduites dans la description comportementale du *NoC* à corriger :

- Une erreur permanente sur 1 bit entre l'interconnexion de l'*IP1* et le routeur (1,0).
- Une erreur permanente sur 2 bits entre l'interconnexion du routeur (1,0) et du routeur (2,0).
- Une erreur transitoire périodique sur 1 bit entre les interconnexions routeurs [(1,1) ; (1,2)] et routeurs [(2,1) ; (2,2)] tous les 10 paquets de données transmis.

La figure 4 présente la structure fautive de la description *VHDL* comportementale du réseau proposé aux étudiants.

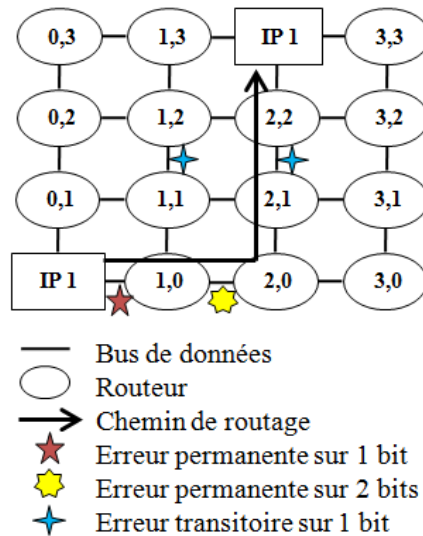


Fig 4. Réseau *NoC* fautif de topologie maillée 4x4.

Les blocs générant les erreurs permanentes ou transitoires dans le réseau proposé sont implantés sur les bus de données liant les différents nœuds du réseau. L'ensemble des routeurs du réseau ont une structure identique et font appel à un même module *VHDL* d'instanciation (*port map*) dans la conception du réseau. Les étudiants doivent uniquement modifier les routeurs pour pallier aux dysfonctionnements survenant dans le réseau dû aux erreurs d'interconnexions intégrées dans la structure du réseau.

2. 2 Conception et implantation d'un routeur tolérant aux fautes

A partir de la simulation du réseau initialement proposé et émettant des erreurs lors des échanges de paquets de données entre les deux IPs interconnectés, les étudiants modifient la structure des routeurs afin de les adapter pour une sûreté de fonctionnement. La fiabilité du réseau proposée par les étudiants consiste à chercher à corriger les erreurs de transmission à travers une nouvelle conception des routeurs mettant en œuvre des techniques de détection et de correction d'erreurs en temps réel des paquets de données échangés (cas de fautes transitoires) ainsi que des solutions algorithmiques de contournement des nœuds de défaillance permanente (proposition et implantation d'algorithme de routage adaptatif au sein des blocs de routage des nœuds du réseau).

A. Conception *VHDL* et implantation *FPGA* du codeur d'*Hamming*

La première partie du travail consiste à détecter les erreurs présentes dans le NoC et de les corriger. Les solutions sont libres, mais l'étudiant se voit proposer une solution simple à mettre en œuvre basée sur l'implantation d'un code correcteur de *Hamming* et d'une parité. L'ajout de 4 bits de *Hamming* ainsi qu'un bit de parité, permet de détecter jusqu'à deux erreurs et de corriger une erreur. Ce bloc de détection et de correction est décrit en *VHDL* et intégré à l'ensemble des routeurs constituant le réseau. De même, des blocs de codage de *Hamming* sont également implantés dans les blocs *IPs* de transmission et de réception de paquets de données. Les spécifications de conception du principe du codage et décodage d'*Hamming + 1 bit de parité* sont décrites ci-dessous. Pour un message initial codé sur 8 bits $[D_1 D_2 D_3 D_4 D_5 D_6 D_7 D_8]$, 4 bits d'*Hamming* (P1, P2, P3, P4) et un bit de parité (P5) sont générés pour une transmission finale sur 13 bits dont la structure est donnée dans la table ci-dessous :

Position des bits	1	2	3	4	5	6	7	8	9	10	11	12	13
Message	P_1	P_2	D_1	P_3	D_2	D_3	D_4	P_4	D_5	D_6	D_7	D_8	P_5

Le calcul de ces parités repose sur l'hypothèse d'une parité paire entre les bits du message D_i et les bits de parité P_i telle que:

$$\begin{aligned}
 P_1 &= D_1 \oplus D_2 \oplus D_4 \oplus D_5 \oplus D_7; & P_2 &= D_1 \oplus D_3 \oplus D_4 \oplus D_6 \oplus D_{11}; \\
 P_3 &= D_2 \oplus D_3 \oplus D_4 \oplus D_8; & P_4 &= D_5 \oplus D_6 \oplus D_7 \oplus D_8; \\
 P_5 &= P_1 \oplus P_2 \oplus D_1 \oplus P_3 \oplus D_2 \oplus D_3 \oplus D_4 \oplus P_4 \oplus D_5 \oplus D_6 \oplus D_7 \oplus D_8
 \end{aligned}$$

Le principe du décodeur de *Hamming + 1 bit de parité* est réalisé à la réception du message. Les bits de vérification V_i sont calculés de la même manière que pour le codage des bits de parité P_i . Un bit de parité globale P est également calculé.

$$\begin{aligned}
 V_1 &= P_1 \oplus D_1 \oplus D_2 \oplus D_4 \oplus D_5 \oplus D_7, & V_2 &= P_2 \oplus D_1 \oplus D_3 \oplus D_4 \oplus D_6 \oplus D_{11} \\
 V_3 &= P_3 \oplus D_2 \oplus D_3 \oplus D_4 \oplus D_8, & V_4 &= P_4 \oplus D_5 \oplus D_6 \oplus D_7 \oplus D_8 \\
 P &= P_5 \oplus P_1 \oplus P_2 \oplus D_1 \oplus P_3 \oplus D_2 \oplus D_3 \oplus D_4 \oplus P_4 \oplus D_5 \oplus D_6 \oplus D_7 \oplus D_8
 \end{aligned}$$

A partir d'une analyse des bits de vérification et de la parité globale, des détections et corrections d'erreurs peuvent être réalisées en considérant le mot binaire $\mathbf{V} = V_4 V_3 V_2 V_1$. Quatre cas de figure se présentent alors :

- Si $\mathbf{V} = 0000$ et $\mathbf{P} = 0$: aucune erreur est détectée;
- Si $\mathbf{V} \neq 0$ et $\mathbf{P} = 1$: une seule erreur pouvant être corrigée est détectée. Le codage \mathbf{V} donne la position du bit erroné à inverser pour correction (Par exemple $\mathbf{V} = 0110$ signifie inversion du digit à la 6ème position) ;
- Si $\mathbf{V} \neq 0$ et $\mathbf{P} = 0$: deux erreurs sont détectées mais ne peuvent être corrigées ;
- Si $\mathbf{V} = 0000$ et $\mathbf{P} = 1$: une erreur est présente sur le bit de parité \mathbf{P} .

Dans le cas d'une erreur détectée, le paquet est corrigé et la transmission est acquittée. Dans le cas d'une détection de deux d'erreurs, deux solutions sont alors envisagées :

- Le routeur est déclaré définitivement fautif. Il est isolé du reste du réseau en activant de façon permanente ses connexions d'indisponibilité aux routeurs voisins.
- Une mémorisation des syndromes de *Hamming* (valeurs V_i et \mathbf{P}) et une demande de retransmission (*NACK*) est mise en œuvre. Si lors de cette seconde transmission du message, les mêmes syndromes sont obtenus, alors une erreur permanente est considérée et le routeur est déclaré fautif permanent.

B. Conception VHDL d'un bloc de routage adaptatif : Modification de la logique de routage des routeurs

Lorsqu'un ou plusieurs nœuds ou routeurs (zone) sont déclarés fautifs, une solution consiste à mettre en œuvre un algorithme adaptatif de routage des paquets de données. L'objectif est le contournement de la zone fautive afin de maintenir la qualité de service du réseau tout en fiabilisant le réseau. Les étudiants modifient alors la logique de routage des routeurs afin d'obtenir des acheminements adaptatifs des paquets de données. Une solution simple proposée aux étudiants est l'ajout d'interconnexions supplémentaires entre les routeurs indiquant un état fautif ou non. Si un routeur a ses

indications *d'état fautif* activées à ses routeurs voisins, sa position est contournée selon des règles de routage à élaborer (voir exemple en figure 5). Les étudiants sont alors sensibilisés aux situations de bouclages (*livelock*) induit par l'utilisation et le développement d'algorithme de routage adaptatif dit « tolérant aux fautes ». Chacune de ces étapes sont validées par simulation.

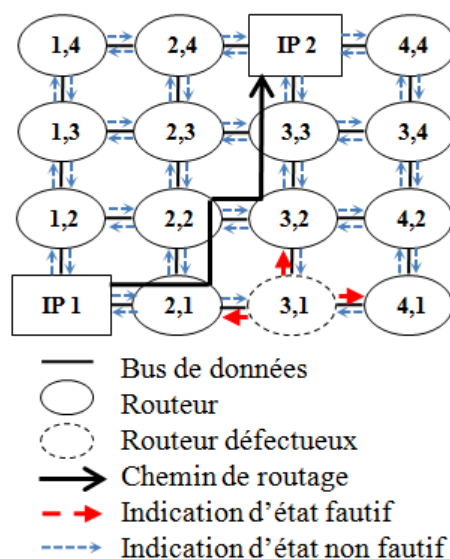


Fig 5. Illustration d'une solution d'algorithme adaptatif

2. 3. Implantation et analyse de performances

Dans chacune des étapes de la conception du *NoC* sûr de fonctionnement, des implantations sont réalisées dans une carte de développement *FPGA Nios II embedded Kit d'Altera*. Des analyses du surcoût en termes de ressources logiques, de latence et de fréquence de fonctionnement, induit à l'intégration des aspects de sûreté de fonctionnement à l'architecture des routeurs, sont alors menées.

3. CONCLUSION

Cet enseignement propose un projet de conception d'un *NoC* tolérant aux fautes permanentes et transitoires à travers la détection et la correction d'erreurs de transmission de paquets données circulant dans un tel réseau. L'objectif pédagogique est de sensibiliser les étudiants, au cours d'une conception architecturale de Systèmes sur puce, au rôle fondamental des concepts de tolérance aux fautes sur la fiabilité et les performances des interconnexions d'un *MPSoC*. La conception et l'intégration sur carte *FPGA* d'éléments assurant une sûreté de fonctionnement à un système de communication sur puce permet une prise de conscience par les étudiants des phases de développement et de conception microélectronique de systèmes embarqués fiables. Les étudiants développent ainsi leurs compétences conception fiable à travers l'élaboration d'une stratégie de sûreté (justification des choix de conception) et d'analyse de l'impact des solutions sur les performances du *NoC* développé.

REFERENCES

- [1] C. Constantinescu, "Trends and challenges in VLSI circuit reliability", *IEEE Micro*, Vol.23, Issue 4, July-Aug. 2003, pp. 14-19.
- [2] G. De Micheli et L. Benini, « Networks on Chips, Technology and tools », Morgan Kaufmann publishers, 2006.
- [3] C. Tanougast, S. Jovanovic, F. Monteiro, C. Diou et A. Dandache, "Initiation à la modélisation et co-simulation comportementale C-VHDL d'un Réseau de communication sur Puce (*Network on Chip*)", 10ème JPCNFM, Saint Malo, 26-28 novembre 2008, pp.27-32.