RenPar'8,
Bordeaux, France, 20–24 mai 1996.

Tampons d'amorçage des microprocesseurs superscalaires à exécution non ordonnée : une analyse

Pascal Sainrat, Stéphan Jourdan, Daniel Litaize [a]

[a]Institut de Recherche en Informatique de Toulouse, 118 route de Narbonne, 31062 Toulouse cedex

1. Introduction

Dans un microprocesseur superscalaire à exécution non ordonnée, les instructions sont acquises depuis le cache, décodées et émises, dans l'ordre du programme, dans un tampon dit *tampon d'amorçage*. Les instructions attendent alors dans ce tampon la disponibilité d'une des unités d'exécution (ou *unités fonctionnelles*) et de leurs opérandes qui peuvent être en cours de calcul, ce qui permet une exécution non ordonnée.

La topologie de ce tampon d'amorçage peut être :

- unifiée comme dans le Pentium Pro : il y a alors un seul tampon qui contient toutes les instructions en attente quelque soit leur type ;
- distribuée comme dans le PowerPC620 : chaque unité fonctionnelle a alors une partition du tampon ;
- semi-unifiée comme dans le PA-8000, le R10000 ou le Sparc64 de HAL. Il y a alors plusieurs partitions, chacune traitant un ou plusieurs types d'instruction, un type d'instruction étant affecté à une seule partition. Dans le cas du PA-8000, il y a deux partitions, une pour les instructions d'accès à la mémoire et l'autre pour toutes les autres instructions.

Après la description de notre modèle d'exécution et des traces utilisées, nous présentons les performances respectives de ces topologies et discutons de leur mise en œuvre.

2. Modèle d'exécution

L'architecture modélisée met en œuvre une politique d'exécution des instructions non ordonnée et spéculative avec gestion précise des exceptions. Le nombre maximum d'instructions pouvant être émises à chaque cycle est appelé *degré* du processeur. Aujourd'hui, le degré est couramment quatre (six dans le cas du Power2) et sera bientôt huit. Chaque entrée du tampon d'amorçage est similaire à une station de réservation de l'IBM 360/91 [5] excepté le fait qu'elle peut être reliée à plusieurs unités fonctionnelles si la topologie est unifiée ou semi-unifiée. Une entrée du tampon d'amorçage maintient donc une instruction ainsi que ses opérandes s'ils sont calculés ou des identificateurs pour les retrouver sinon. Lorsque l'exécution d'une instruction s'achève, son résultat est transmis aux entrées du tampon (aux stations) qui le requièrent.

La politique d'amorçage consiste à choisir les instructions qui vont être amorcées parmi toutes celles qui peuvent l'être. Nous avons retenu celle qui sélectionne l'instruction exécutable qui a été insérée le plus tôt dans le tampon et dont les opérandes sont connus (Butler et Patt [1] ont montré que, si suffisament d'unités d'exécution sont présentes, les différentes politiques d'amorçage donnent à peu près la même performance).

Dans le cas où une instruction peut être émise vers plusieurs partitions (cas d'une topologie distribuée), c'est la moins remplie qui est choisie.

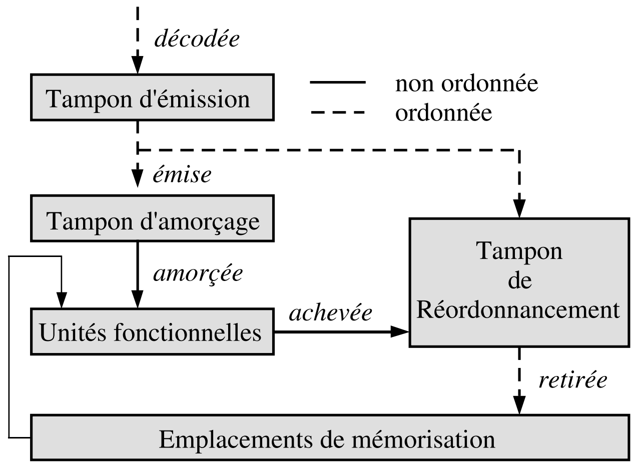

FIG. 1 - . Etats suivis par une instruction

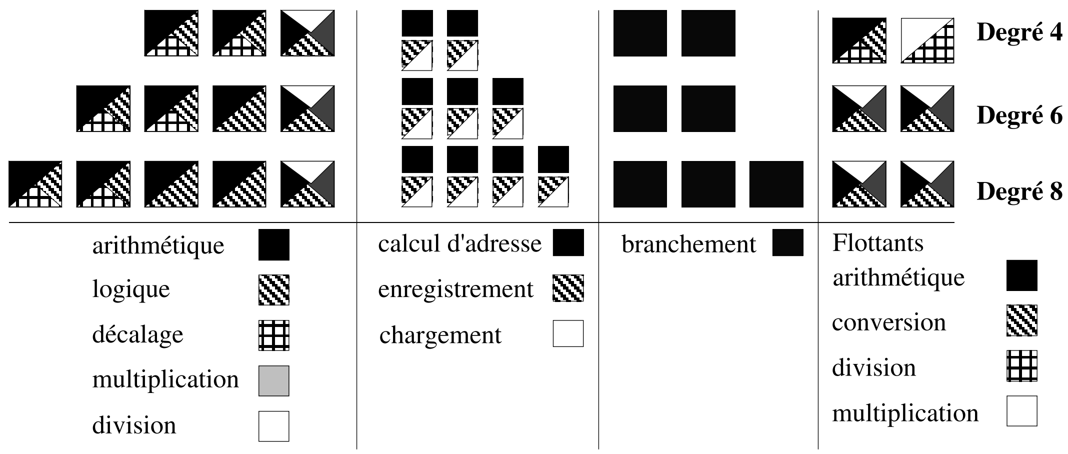

FIG. 2 - . Configurations des unités fonctionnelles

Les différents états suivis par une instruction sont *acquise, décodée, émise, amorcée, achevée et retirée* (cf. figure 1). Le tampon de réordonnancement est une structure qui ordonne la mise à jour des résultats afin de récupérer un état correct en cas d'exception.

La *fenêtre d'anticipation* est une représentation abstraite des instructions en cours. Une instruction entre dans la fenêtre d'anticipation lorsqu'elle est émise et elle en sort lorsqu'elle est retirée [4]. La taille de la fenêtre d'anticipation est celle du tampon de réordonnancement dans notre modèle. Dans [3], nous avons déterminé des tailles optimales d'un point de vue coût/performance de cette fenêtre (degré 4 : 32, degré 6 : 64, degré 8 : 96) et le nombre optimal d'unités fonctionnelles traitant chaque type d'instructions. Ces configurations sont représentées dans la figure 2. Chaque carré représente une unité fonctionnelle et les grisés les types d'instructions qu'elle peut exécuter.

3. Méthodologie de simulation

Comme nous voulons déterminer l'impact de la topologie du tampon d'amorçage, nous supposons que le mécanisme d'acquisition des instructions est idéal c'est à dire que:

- les instructions sont toujours présentes dans le cache d'instructions,
- les branchements sont toujours correctement prédits,
- l'émission est limitée seulement par le degré du processeur et l'absence d'entrées libres dans le tampon d'amorçage.

Nous supposons également que les données sont toujours présentes dans le cache de données.

Le simulateur est dirigé par des traces générées par pixie à partir d'une exécution réelle des programmes d'évaluation. Le simulateur lit ces traces et précise le nombre moyen d'instructions terminées par cycle (IPC). Afin de respecter les hypothèses ci-dessus, le simulateur ne prend en compte que le degré du processeur, la taille de la fenêtre d'anticipation, les dépendances de données, la configuration des unités fonctionnelles et enfin la topologie et la taille du tampon d'amorçage.

Les latences d'exécution des instructions sont celles du PowerPC604. Elles sont rappelées dans [3]. Toutes les unités sont indépendantes et pipelines sauf l'unité de division.

Les programmes d'évaluation utilisés sont les SPEC92 avec leurs jeux d'entrée réduits. En tout, plus de 600 millions d'instructions sont simulées. Le lecteur trouvera plus d'informations à ce sujet dans [2].

4. Résultats et conclusions

Dans un premier temps, nous avons déterminé la taille la plus intéressante en matière de rapport coût/performance d'un tampon d'amorçage unifié. Par manque de place, nous ne pouvons pas donner les courbes mais les tailles obtenues sont données dans la table 1.

	Degré 4	Degré 6	Degré 8
Tampon unifié	28	48	72
IPC (CINT92)	3,5	5	6,3
IPC (CFP92)	2,1	2,8	3,3

TAB. 1 -
Nombres d'entrées et performance d'un tampon unifié.

Type du tampon	Degré 4	Degré 6	Degré 8
Accès mémoire	12	24	36
Autres opérations	24	36	52
Total	36	60	88
IPC (CINT92)	3,5	5	6,3
IPC (CFP92)	2,1	2,8	3,3

TAB. 2 -
Nombres d'entrées et performance d'un tampon semi-unifié.

Dans le cas d'un tampon semi-unifié type PA-8000, nous avons évalué la taille de la partition pour les accès mémoire puis celle de la partition dédiée aux autres instructions. Les meilleures configurations sont résumées dans la table 2 avec leurs performances.

Enfin, la table 3 indique les performances obtenues pour une topologie distribuée toujours en considérant les configurations qui nous semblent les meilleures pour chaque partition.

Type du tampon	Degré 4	Degré 6	Degré 8
Arithmétique et logique		8	10
Idem et multiplication et division	4	8	10
Arithmétique et logique et décalage	6	8	10
Calculs d'adresses	4	8	8
Accès mémoire	8	10	12
Branchements	4	8	8
Flottants	12	16	22
Total	72	128	190
IPC (CINT92)	3,4	4,8	5,9
IPC (CFP92)	2	2,7	3,1

TAB. 3 -
Nombres d'entrées et performances d'un tampon distribué.

Les trois tables ci-dessus montrent que plus le tampon est distribué, plus le nombre d'entrées total est important. Un rapport proche de 1 à 3 existe entre les deux topologies extrêmes. De plus, la performance est sensiblement inférieure dans le cas d'un tampon distribué car, dans ce cas, la politique d'émission joue un rôle non négligeable. En effet, une instruction peut être émise vers plusieurs partitions capables de la traiter et le choix de la partition la moins remplie n'est peut-être pas le meilleur. Des études sont en cours à ce sujet. Toutefois, l'arbitrage est d'autant plus simple que le tampon est distribué. En effet, d'une part, le nombre d'entrées par partition diminue, et, d'autre part, le nombre d'arbitres associés à chaque partition diminue pour arriver à 1 dans le cas d'une topologie distribuée. Une topologie semi-unifiée semble donc être un bon compromis puisque la performance est très proche de celle d'un tampon unifié pour un côut légèrement supérieur mais compensé par des arbitrages plus simples.

Bibliographie

1. M. Butler et Y.N. Patt, *An Investigation of the Performance of Various Dynamic Scheduling Techniques*, Actes du 25ème Annual International Symposium on Microarchitecture, décembre 1992.
2. S. Jourdan et P. Sainrat, *Génération de trace avec pixie sous IRIX 5.3, et Analyse de l'impact des dépendances de données à partir des SPEC92*, Rapport Interne 95-10-R.
3. S. Jourdan, P. Sainrat et D. Litaize, *Exploring Configurations of Functional Units in an Out-Of-Order Superscalar Processor*, Actes du 25ème Annual International Symposium on Computer Architecture, juin 1995.
4. J. E. Smith, G. S. Sohi, *The Microarchitecture of Superscalar Processors*, Proceedings of the IEEE, vol. 83, num. 12, December 1995.
5. R.M. Tomasulo, *An Efficient Algorithm for Exploiting Multiple Arithmetic Units*, IBM Journal, Vol. 11, janvier 1967.