Architectures Reconfigurables : opportunités pour la faible consommation

Sébastien Pillement, Raphaël David, Olivier Sentieys

Equipe R2D2 - Université de Rennes I/IRISA
6 rue de Kerampont, BP 447, 22305 Lannion, France
Mél : pillemen@univ-rennes1.fr
http://lasti.enssat.fr/

résumé—**Les architectures reconfigurables représentent un nouveau choix dans le processus de conception et d'implémentation d'applications complexes. Ces architectures, de part leurs flexibilité, offrent des opportunités pour la prise en compte de la consommation dans le cycle de conception. Bien que récente dans le domaine du reconfigurable, cette contrainte est intégrée dans les nouvelles recherches à tout les niveaux de la conception (optimisations architecturales, logiques et technologiques). Nous présentons dans cet article les opportunités et les premiers résultats, du point de vue de la consommation, qu'offrent les architectures reconfigurables.**

mots clés—**Consommation, Architectures reconfigurables.**

1 Introduction

Depuis quelques années, les progrès technologiques réalisés ont permis l'émergence d'un nouveau type d'architectures : les architectures reconfigurables. L'idée de base de ces architectures est d'offrir aux concepteurs la flexibilité d'un microprocesseur et les performances temporelles d'un circuit dédié. Ces architectures ont donc été très largement utilisées dans des domaines très volatiles, où les normes font défauts et où les performances d'un microprocesseur seul ne sont pas suffisantes. Elles offraient cependant de très faibles capacités d'intégration et n'offraient pas des performances de calcul très élevées. Les recherches dans ce domaine se sont donc intéressées à l'amélioration de ces problèmes délaissant totalement les aspects consommation. Elles se sont traduites par l'introduction de nouvelles architectures [1] supportant les contraintes temporelles d'applications complexes et offrant de nouveaux paradigmes de calcul.

Dans le même temps, de nouvelles applications ont émergées telles que les ordinateurs personnels ou portables (PC, portable desktop, pagers, ...), les télécommunications sans fil (radiomobile, UMTS, ...), les assistants de personnes (PDA's), le multimédia, les réseaux ad'hoc, ... Ces applications intègrent des fonctionnalités complexes qui demandent des calculs performants.

En outre, du point de vue architectural, les systèmes de prochaines générations auront successivement à traiter des applications très différentes en terme de grain de calcul (du niveau bit au niveau arithmétique), de motif de calcul ou d'accès aux données. Lorsque par ailleurs, le système doit consommer très peu d'énergie (systèmes embarqués ou problème de refroidissement) le problème de sa conception devient insoluble si l'on se limite aux moyens actuels (ASIC, FPGA, processeurs programmables).

On peut prendre pour exemple les futures générations de télécommunications mobiles. En effet, au très haut niveau de performance (estimé à 12 GOPS) résultant de l'association de capacités multimédia et de techniques d'accès évoluées telles que le W-CDMA que devront supporter ces systèmes, s'ajoute la nécessité de supporter l'ensemble des algorithmes intégrés aux normes de générations actuelles ainsi que leurs évolutions. Ces systèmes étant embarqués leurs consommations devra être maîtrisée.

De part les contraintes associées aux nouvelles applications, les solutions totalement matérielles (ASIC) ou totalement logicielles (μP, DSP) ne sont plus adaptées. Dès lors, l'alternative proposée via les architectures reconfigurables est de plus en plus considérée. De nombreux travaux ont dés lors démarrés dans le but d'associer les trois principales contraintes inhérentes aux applications de prochaine génération que sont les hautes performances, la faible consommation et la flexibilité.

Dans ce papier, après un rappel des contraintes et des spécificités des architectures reconfigurables nous aborderons la problématique de la consommation pour cette cible technologique. Les opportunités de conception faible consommation au niveau physique, au niveau porte et au niveau architectural feront l'objet des

sections suivants.

2 Architectures reconfigurables : définitions et exemples

Il est facile d'associer architectures reconfigurables et FPGA[1], cependant ceci réduit considérablement l'espace de conception de ces architectures. L'étude sémantique du mot reconfigurable en donne une définition plus précise. En effet, **configuration**[2], signifie à l'origine *façonner à la ressemblance de* et a pris le sens de *disposition relative d'éléments*.

Une *architecture* est constituée d'une *disposition relative d'éléments* organisés selon un certain schéma. La reconfiguration, en permettant un choix des éléments d'une part, et de leur disposition relative d'autre part, autorise une variabilité des schémas et donc des architectures. Cette définition recouvre alors un large ensemble d'architectures dont les FPGAs font partis.

L'avantage de ces systèmes vient de la possibilité d'augmenter les performances de traitement en créant un chemin de données matériel adapté, tout en maintenant un niveau de flexibilité dévolu habituellement aux solutions logicielles. Cette flexibilité est obtenue par la mise en place de ressources redondantes offrant différentes possibilités de routage ou de calcul. Les architectures sont en général organisées selon un tableau régulier d'éléments de calcul (fig. 1) définis à différent niveau de granularité. Ces ressources de calcul sont interconnectées par un réseau plus ou moins flexible et plus ou moins performant [2]. Les connexions sont réalisées par des matrices d'interconnexions construites autour de portes de transmission permettant de créer des connexions entre les segments arrivant sur la matrice.

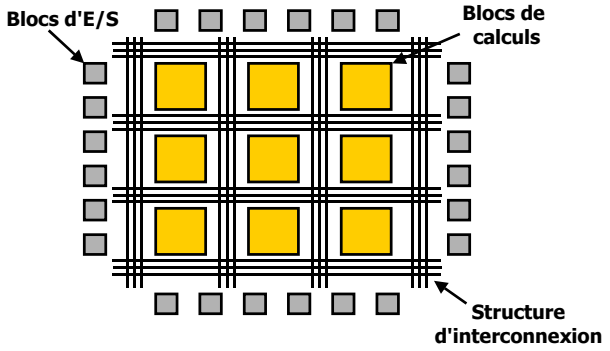

FIG. 1 – *Architecture reconfigurable générique*

La reconfiguration consiste donc à spécifier les opérations et les interconnexions de cet ensemble de ressources, que l'on nomme couche opératoire. Cette reconfiguration est effectuée dans une couche supérieure (dite couche de reconfiguration) construite autour de points de mémorisation statique.

2.1 Granularité de reconfiguration

Dans [3], Jan Rabaey défini quatre niveaux de reconfiguration. Le niveau porte qui correspond aux FPGA, le niveau opérateur, le niveau fonctionnel et le niveau système. Ce dernier niveau correspond en fait aux processeurs programmables. Les considérations de consommation pour ce niveau de reconfiguration sont connues et étudiées par ailleurs [4, 5, 6]. La classification proposée nous semble ambiguë et difficile à réaliser, dans le sens où la distinction fonctionnel/opérateur n'est pas claire. Nous regrouperons donc ces deux niveaux de reconfiguration. Nous qualifierons de *grain fin* les architectures de type FPGA qui permettent une optimisation au bit près. Les architectures de grain plus épais (de niveau fonctionnel ou opérateur) seront appelées *processeurs reconfigurables*.

Nous présentons ici les principales caractéristiques des architectures grain fin et des processeurs reconfigurables.

2.1.1 Reconfiguration au niveau porte

La reconfiguration au niveau porte est celle qui est réalisée sur les composants FPGAs où CPLD[3] [7]. Les architectures supportant ce type de reconfiguration sont qualifiées de grain fin en raison de la faible largeur des chemins de données.

Dans le cas (le plus répandu) des FPGAs ,la reconfiguration opère au niveau logique sur des LUTs (Look-Up Table) et sur leurs interconnexions. Etant donné la quantité de LUTs nécessaire à la définition de fonctions évoluées[4], et un réseau d'interconnexions totalement connecté, ces architectures nécessitent un très grand nombre de données de configuration. Pour ces architectures la couche de configuration est une mémoire de type SRAM.

En reconfigurant une architecture au niveau logique, on a la possibilité de définir n'importe quel type de chemin de données en *synthétisant* les opérateurs requis par l'application. Cette caractéristique est très intéressante pour les traitements logiques pour lesquels chaque bit est susceptible d'être traité indépen-

1. Field Programmable Gate Array
2. Le Robert, dictionnaire historique de la langue française sous la direction d'Alain Rey
3. Complex Programmable Logic Device
4. la complexité des équations booléennes implantées dans ces LUTs est limitée par le nombre d'entrées de ces équations et non par un nombre de portes logiques: typiquement 3/4 entrées

damment. Mais si cette flexibilité facilite la *"customisation"* des chemins de données disposant d'un fort parallélisme au niveau bit, elle est par contre défavorable aux traitements arithmétiques.

2.1.2 Reconfiguration au niveau fonctionnel / opérateur

L'inadéquation des FPGAs avec certaines applications à amené les concepteurs à proposer des architectures alternatives. Dans ces architectures les interconnexions et les opérateurs sont configurables mais travaillent au niveau arithmétique (par exemple 8 ou 16 bits). Ceci à pour effet d'améliorer les performances sur les traitements arithmétiques, au détriment des optimisations au niveau bit. Cette approche, provient du fait que les applications sont constituées de cœurs de boucles très réguliers [8]. Ces architectures réduisent par ailleurs la taille des configurations, permettant d'envisager des stratégies de reconfigurations dynamiques.
La plupart de ces architectures utilise un réseau d'interconnexion [2] à deux dimensions, éventuellement hiérarchique. On peut citer, par exemple, les architectures DReAM [9], Morphosys [10], FPFA [11], RaPiD [12] ou le KressArray [13]. Cependant certaines architectures présente une topologie linéaire comme le Piperench [14].
La taille des reconfigurations étant limitée, des mécanismes évolués peuvent ainsi être mis en œuvre pour distribuer les informations de configuration et réduire le coût (en consommation) du contrôle de ces architectures. Ces considérations ont été mises en œuvre dans les architectures Pleiades [15] et DART [16], spécifiquement conçues pour la faible consommation.

3 Répartition d'énergie dans les architectures reconfigurables

Les solutions dédiées (ASIC) peuvent êtres considérées comme optimales du point de vue de la consommation. Dans ces circuits les opérateurs sont optimaux en terme de largeur de donnée et de sophistication car ils sont conçus pour une application. Il n'y a donc pas de flexibilité et aucune ressource supplémentaire n'est nécessaire. L'exécution de l'application est déterministe et gérée par une simple machine d'états.
Le parallélisme (de donnée, d'instruction ou de tâches) peut par ailleurs être complètement exploité. En choisissant le bon niveau de parallélisme, il est alors possible de réduire la fréquence de fonctionnement. De ce fait, la tension d'alimentation peut être optimisée. Dans ces circuits la fréquence et la surface étant optimaux, l'arbre de distribution de l'horloge peut être réduit et optimisé. Les pénalités de distribution de l'horloge peuvent également être minimisées en utilisant des communications globalement asynchrone - localement synchrone [17].
Finalement, le fait de construire complètement l'architecture permet d'optimiser les échanges de données. La hiérarchie mémoire (très consommatrice d'énergie) est donc optimale et les accès aux données sont maîtrisés [18].
Clairement, les ASICs représentent la limite basse à atteindre pour la consommation, cependant ces circuits n'offrent aucune flexibilité. Cette problématique à des impacts sur le coût des circuits (car il y a très peu de réutilisation) et sur les nouvelles applications (qui nécessitent de la flexibilité). Cette constatation à des échos dans le monde industriel puisque Infineon, Motorola, et Texas Instruments annoncent l'utilisation d'architectures reconfigurables dans leurs futurs système pour l'infrastructure des télécommunications mobiles.

3.1 Distribution de l'énergie dans les architectures reconfigurables grain fin

Les FPGAs offrent une reconfiguration au niveau bit et utilisent généralement un réseau d'interconnexion de type mesh. De nombreux travaux ont démontrés le potentiel de ces architectures du point de vue de la performance [19], cependant ces architectures introduisent des éléments faisant augmenter la consommation d'énergie.
En fait, construire des opérateurs arithmétique implique l'utilisation d'un grand nombre de cellules 1 bit qui doivent être interconnectées. Le réseau étant très flexible, les signaux doivent alors traverser un grand nombre de matrices d'interconnexions (switch box) qui induisent un gaspillage d'énergie non négligeable.
Les études menées pour estimer la consommation d'énergie des FPGAs [20, 21, 22, 23] montrent toutes que 60-70 % de l'énergie consommée est dû à l'interconnexion, 20 % est dû à l'arbre d'horloge et que le reste se répartie entre les blocs de calcul et les entrées-sorties. Cette distribution inhabituelle de la consommation vient du fait que sur des opérations arithmétiques il n'est pas nécessaire de contrôler les bits individuellement [15], le parallélisme de niveau bit étant très limité.
Un autre inconvénient de ces architectures provient du volume de données de configuration nécessaire. Cette configuration (qui concerne les opérateurs au

niveau bit et les interconnexions) est chargé en série dans une mémoire de type SRAM. Lorsque ces configurations atteignent le million de bits [24, 25], l'énergie de la reconfiguration doit être pris en compte. Il n'existe cependant, pour l'instant, aucune estimation (sinon celle appliquée aux mémoires) sur le coût de la reconfiguration d'un FPGA.

3.2 Distribution d'énergie dans les processeurs reconfigurables

La distribution d'énergie de ces architectures peut être rapprochées de celle des processeurs programmables, dans lesquelles l'ajout de ressources pour obtenir de la flexibilité se paye par une forte consommation d'énergie (distribution du contrôle, accès à de large bancs mémoires, fréquence élevée, ...).
Cependant la méthode de reconfiguration des processeurs reconfigurable, permet la réduction de la consommation d'énergie. En limitant la taille de la configuration, et en tenant compte de la règle du 80/20 [26, 27] ces reconfigurations arrive très occasionnellement et réduisent ainsi la consommation dûe à la distribution du contrôle de l'architecture.
Un autre avantage de ces architectures vient de la possibilité de prendre en compte le parallélisme de l'application. Il est alors possible de réduire la fréquence de fonctionnement et la tension d'alimentation en adaptant le parallélisme de l'architecture à celui de l'application.

3.3 Synthèse

La problématique de la faible consommation dans les architectures reconfigurables doit donc prendre en compte ces spécificités. Cette contrainte doit être optimisée à tous les niveau de la conception d'un système à base d'architectures reconfigurables.
La table 1 résume les avantages et inconvénients des différentes classes d'architecture et montre l'incidence du mode de reconfiguration sur l'efficacité énergétique obtenue. La première colonne donne une estimation du nombre de données de reconfiguration nécessaire. La deuxième colonne présente alors la fréquence de reconfiguration envisageable et spécifie si la reconfiguration doit être effectuée à chaque cycle, à chaque appel de fonction (ou exécution d'une boucle) ou lorsqu'une nouvelle tâche (application) arrive. La troisième colonne résume les deux premières en approximant l'efficacité énergétique caractérisée par le nombre de MIPS[5] pour chaque mW consommé. Le critère de flexibilité est finalement présenté.

5. Millions of Instructions Per Seconds

Nous allons dans la suite du document présenter les optimisations possibles et les axes de recherche pour réduire la consommation des architectures reconfigurables.

4 Optimisation au niveau technologique

Au niveau transistor les initiatives d'optimisations pour les architectures grain fin sont faibles, le marché étant couvert par les deux principaux fabricants de FPGA (Xilinx et Altera). Cette situation laisse donc peux de place pour les innovations à ce niveau. Cependant Xilinx offre maintenant des architectures spécifiques pour le "low-power" avec son architecture CoolRunner©[28]. Cette architecture de CPLD possède un mode standby et supporte différents domaine d'horloge et un fonctionnement asynchrone. Cette architecture supporte des fréquences de 333 MHz et consomme 14 μA en mode standby. Le mode "standby" de cette architecture à par ailleurs était repris dans d'autres architectures de Xilinx, comme le Spartan et le XC4000 (serie XL).
Les fabricants intègrent désormais des mécanismes permettant de réduire la consommation. Ces blocs spécifiques concernent essentiellement la distribution de l'horloge [29]. Les FPGAs actuels intègrent plusieurs DLL[6] ou PLL[7] permettant de définir plusieurs domaines d'horloge indépendant [30, 25] et proposent des mécanismes de re-synchronisation.
Des recherches ont néanmoins était menée à ce niveau pour proposer notamment de nouvelles architectures de portes de transmission [23]. Les auteurs ont montrés qu'avec leur architecture la tension d'alimentation peut être ramenée à 2xV_T. Ce type de structure permet d'utiliser des tensions d'alimentation de l'ordre de 1,5 V pour les nouvelles technologies de FPGAs.
Une architecture grain fin et low-power a été proposée par V. George [31]. Cette architecture implémente une interconnexion de type "low-swing" et apporte une attention particulière aux matrices d'interconnexions par l'extraction de la taille optimale des transistors constituant les portes de transmission. Le problème principal provenant des connexions non activées introduisant de fortes capacités de charge, lls ont aussi travailler sur un réseau d'interconnexions hiérarchique permettant la réduction de la consommation. Enfin, la modification de la hiérarchie de calcul à été revue pour obtenir une consommation plus

6. Delay Locked Loop
7. Phase Locked Loop

Type d'Architecture	Taille de la configuration	Fréquence de configuration	Efficacité énergétique	Flexibilité
ASIC	0 bit	0	$\approx$ 100 MIPS/mW	- -
FPGA	$\approx$ 1 M bits	Tâche	$\approx$ 1 MIPS/mW	+/-
niveau opérateur	$\approx$ 10 k bits	Fonction	$\approx$ 1 MIPS/mW	+
niveau fonction	$\approx$ 100 bits	Fonction	$\approx$ 10 MIPS/mW	+
Processeur	$\approx$ 10 bits	Cycle	$\approx$ 1 MIPS/mW	++

TAB. 1 – *Efficacité énergétique des différentes classes d'architecture*

faible tout en maintenant un niveau de flexibilité acceptable.

Une autre approche se retrouve dans les architectures FPOP [32] ou le RNS Reconfigurable Processor [33], l'idée ici est de tirer partie des propriétés des codes redondants, ou de l'arithmétique des résidus (RNS [34]) , afin de réduire la surface des opérateurs arithmétiques. Le passage dans une représentation autre que le binaire naturel permet de réduire la consommation de ces architectures en simplifiant certaines opérations.

D'une manière générale, en ce qui concerne les processeurs reconfigurables les techniques de conception d'ASIC pour la faible consommation doivent êtres appliquées.

5 Optimisation au niveau porte

L'objectif prioritaire est ici de réduire au maximum les interconnexions et le taux d'activité des signaux.

La première optimisation vient de la décomposition logique des circuits. Cette optimisation concerne essentiellement les architectures grain fin, les processeur reconfigurable travaillant au niveau arithmétique. En effet dans un FPGA, la logique est implémentée sous forme de LUT généralement de 4 entrées. Ces LUT, pouvant être assimilées à de la mémoire SRAM, sont souvent sous-utilisées gaspillant alors de la logique et des interconnexions. Ce problème NP-complet n'est pas beaucoup abordé dans la littérature (du point de vue de la consommation), les architectures étant extrêmement protégée par leur constructeur. Cependant des optimisations peuvent êtres apportées à partir d'une solution initiale [35], l'objectif et de remanier le "mapping" technologique afin de réduire (à fonction constante) le taux d'activité des signaux internes d'une fonction complexe. Les résultats obtenus amènent une amélioration de l'ordre de 10 % sur la consommation globale du circuit. Il est possible ici, étant donner la flexibilité des architectures d'envisager des techniques d'encodage de bus ou de FSM [36] permettant encore une fois de réduire le taux d'activité.

Le deuxième aspect de ce niveau vient du placement des blocs de logiques dans la couche opératoire. Ce placement a un impact fort sur le routage. Les algorithmes de placement - routage minimisent le temps d'exécution ou la surface. Pour une conception efficace en consommation le placement - routage doit souvent être effectué à la main.

Les architectures reconfigurable intègrent aujourd'hui beaucoup de fonctionnalités [30, 37]. Le choix du bloc approprié a alors un impact majeur sur les performances en consommation. Par exemple, dans le VIRTEX II l'implémentation de ressources de stockage peut se faire de trois manières : par l'utilisation de registres, l'utilisation de mémoire locale qui est un premier niveau de hiérarchie mémoire, ou par l'utilisation du deuxième niveau de hiérarchie que sont les BRAM. Le choix d'un de ces éléments dépend du nombre de données à stocker, si ce nombre excède 48 le choix des BRAM est alors le meilleur. Un autre exemple vient de l'utilisation efficace des blocs intégrés dédiés, par exemple les multiplieurs des nouvelles génération de FPGAs sont plus efficace, en terme d'énergie, qu'une implémentation sous forme de LUT. Dans les processeurs reconfigurables ce choix est primordial, les opérateurs étant optimisés pour certain types d'opérations.

Finalement le "clock gating" qui est une technique très largement utilisée peut être appliquée au architectures reconfigurables. En utilisant les ressources de gestion d'horloge intégrées, il est possible de gérer dynamiquement l'arbre d'horloge afin de ne faire commuter que les parties utiles du circuit [29]. Une autre possibilité est de réduire dynamiquement la fréquence de certaines parties du circuit. Ces techniques sont aussi employées dans des processeurs reconfigurables comme DART par exemple.

Le dernier point consiste à utiliser des arithmétiques différentes s'adaptant au mieux aux architectures reconfigurables "classiques". L'arithmétique RNS permet d'envisager des résultats intéressant [38]. Cette arithmétique permet de représenter les systèmes sous la forme de petit datapath indépendant, permettant d'optimiser l'utilisation des ressources de routages. Le parallélisme des architectures grain fin est particulièrement bien adaptée à ce type d'approche.

6 Optimisation au niveau système

Les optimisations à ce niveau sont très classiques dans la conception en vue de la faible consommation. Cependant la flexibilité et les ressources intégrées dans une architecture reconfigurable permettent de tirer partie de ces optimisations pour différentes classes d'applications, au lieu d'une.

Dans [39], les auteurs montre une réduction de la consommation d'énergie de l'ordre de 33% par l'utilisation d'un pipeline "en vague" (Wave Pipeline). Ce pipeline, ne découpe pas le circuit en fonctionnalité, mais crée des chemins de longueur égale, tirant partie de la topologie des architectures reconfigurables il permet de ne pas limiter la fréquence de fonctionnement par des chemins critiques trop long et disparate. Les auteurs ont aussi démontré un gain de 45 % par un partitionnement judicieux de l'application. Cette technique de pipeline augmente la puissance dissipée mais peut permettre la diminution de l'énergie consommée. Cette technique de pipeline est utilisée aussi dans le Piperench. Les auteurs utilise la reconfiguration pour définir la notion de pipeline virtuel [40] permettant l'implémentation d'une application dont le nombre d'étages de pipeline est supérieur à celui que peut supporter l'architecture.

La sélection des algorithmes et des architectures les implémentant est un facteur de réduction de la consommation. Dans [41], les auteurs obtiennent des gains de consommation de l'ordre de 70 % par rapport à une implémentation classique sur un Xilinx VIRTEX II et un DSP TI TMS320C6415. Ces gains sont obtenus par un choix judicieux des algorithmes, par exemple une multiplication de matrice consommera moins si elle est considérée comme étant un tableau 1D, car elle nécessitera moins d'interconnexion et par l'exploitation du parallélisme de l'application.

Le choix de l'architecture joue un rôle très important sur la consommation en déterminant le nombre de ressources d'interconnexion et de calcul nécessaires à l'application [42]. Si toutes les optimisations des architectures grain fin peuvent êtres transposées pour les processeurs reconfigurables, sur ce point ces dernières amènent des optimisations radicales par rapport au niveau bit.

Une approche spécifique vient des possibilités de reconfiguration partielle et dynamique des nouvelles architectures. Dans [43, 44], les auteurs utilisent la reconfiguration partielle pour réduire la taille du circuit utilisé, un circuit plus petit consommant moins d'énergie. L'objectif principal et d'éviter d'avoir des ressources non utilisées. Ils ont appliquées cette méthode au traitement des images et s'adaptent aux besoins en monitorant les entrées du système. Des blocs fonctionnel sont alors répliqués ou supprimés en utilisant la reconfiguration partielle.

La dernière optimisation au niveau système spécifique aux processeurs reconfigurables vient de la réduction de la taille de reconfiguration, amenant une réduction de la mémoire de stockage nécessaire. De plus ces architectures offrent (dans la majorité des cas) plusieurs niveau de parallélisme, permettant alors de s'adapter à l'application courante.

7 Conclusion

Il n'existe pas à l'heure actuelle d'architecture reconfigurable au niveau bit, associant des considérations de faible-consommation et une forte densité d'intégration. Des initiatives existent cependant à des niveaux de reconfiguration supérieurs. L'architecture Pléiades est une architecture reconfigurable hétérogène définit pour le "low-power". L'intégration de blocs de calcul dédiée et la définition d'un réseau d'interconnexions hiérarchique lui permettent d'obtenir une efficacité énergétique de 30 MOPS/mw. L'architecture DART est quand à elle une architecture reconfigurable au niveau fonctionnel. La prise en compte, lors de sa conception, de la contrainte de faible consommation permet d'obtenir une efficacité énergétique de 40 MOPS/mW.

La consommation des circuits reconfigurables devient une contrainte de conception importante. Ormis des effets de bord dûs aux spécificités de ces architectures permettant d'obtenir de la flexibilité, toutes les techniques classiques de conception en vue de la faible consommation peuvent êtres utilisées. Les fabricants de circuits reconfigurables apportent par ailleurs un intérêt de plus en plus grand à ces questions, et offrent des possibilités d'optimisation de la consommation dans leurs architectures.

Cependant de nombreux verrous reste à lever. En effet les architectures reconfigurables faible grain souffrent d'une sous-utilisation de leurs ressources de calcul amenant un gaspillage d'énergie. De plus, les modèles de consommation entre architecture reconfigurable et dédié sont différents. La puissance étant généralement dû à l'horloge dans les ASIC, alors que ce sont les interconnexions qui sont dominantes dans les architectures reconfigurables. Cette pression sur l'interconnexion risque de s'accentuer avec l'avènement des technologies sub-micronique profondes.

Enfin, les méthodologies de conception nécessitent des estimations précises et de haut niveau afin de guider la synthèse des applications. Si l'estimation de

consommation du plan opératoire commence à être maîtrisée, il en est autrement de la consommation du plan de configuration pour lequel aucun travail n'est mené à ce jour. Ces estimations deviendront prépondérante si l'on souhaite tirer partie des possibilités de reconfiguration partielle des futures architectures.

Les architectures reconfigurables offrent des opportunités pour la conception faible-consommation. Si des progrès sont encore à faire pour les architectures faible grain, les processeurs reconfigurables offrent des optimisations et des résultats très encourageant.

Références

[1] R. Hartenstein. A Decade of Reconfigurable Computing : A Visionary retrospective. In *Design Automation and Test in Europe*, Munich, Germany, March 2001.

[2] H. Zhang, M. WAN, V. George, and J. Rabaey. Interconnect Architecture Exploration for Low-Energy Reconfigurable Single-Chip DSPs. In *IEEE Workshop on VLSI*, April 1999.

[3] Jan M. Rabaey. Reconfigurable Processing : The solution to Low-Power Programmable DSP. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, April 1997.

[4] J. Fridman. Sub-Word Parallelism in Digital Signal Processing. *IEEE Signal Processing Magazine*, 17(2):27–35, March 2000.

[5] Paolo Faraboshi, Giuseppe Desoli, and Joseph A. Fisher. The Latest Word in Digital and Media Processing. *IEEE Signal Processing Magazine*, pages 59 – 85, March 1998.

[6] Venkata S. Krishnan. *Speculative Multithreading Architectures*. PhD thesis, University of Illinois, 1998.

[7] Christan Tavernier. *Circuits logiques programmables*. Microcontrôleur et environnement. Dunod, 1996.

[8] N. Zhang and R.W. Brodersen. Architectural Evaluation of Flexible Digital Signal Processing for Wireless Receiver. In *Asilomar Conference*, October 2000.

[9] Jürgen Becker, Thilo Pionteck, and Manfred Glesner. DReAM : A Dynamically Reconfigurable Architecture for Future Mobile Communication Applications. In *international Workshop on Field Programmable Logic and Applications*, pages 312–321, Villach, Austria, August 2000. Lecture notes in Computer Science 1896.

[10] H. Singh, G. Lu, M. Lee, E. Filho, and R. Maestre. MorphoSys : Case study of a Reconfigurable Computing System targeting Multimedia Applications. In *International Design Automation Conference*, pages 573–578, Los Angeles, California, June 2000.

[11] Gerard J.M. Smit, Paul J.M. Havinga, Lodewijk T. Smit, Paul M. Heysters, and Michel A.J. Rosien. Dynamic Reconfiguration in Mobile Systems. In *International Conference on Field Programmable Logic and Applications*, pages 171–181, Montpellier, France, September 2002. Lecture Notes in Computer Sciences 2438.

[12] D. C. Cronquist, P. Franklin, C. Fisher, M. Figueroa, and C. Ebeling. Architecture Design of Reconfigurable Pipelined Datapath. In *Advance Research in VLSI*, pages 23–40, Atlanta, USA, March 1999.

[13] R. Hartenstein and R. Kress. A Datapath Synthesis System for the Reconfigurable Computing. In *Asia and South Pacific Design Automation Conference*, Chiba, Japan, 1995.

[14] S. Goldstein, H. Schmit, M. Moe, M. Budiu, and S. Cadambi. PipeRench : A Coprocessor for Streaming Media Acceleration. In *International Symposium on Computer Architecture*, Atlanta, USA, May 1999.

[15] Arthur Abnous. *Low Power Domain Specific Processors for Digital Signal Processing*. PhD thesis, University of California, Berkeley, 2001.

[16] R. David, D. Chillet, S. Pillement, and O. Sentieys. *VLSI-SOC 2001 Post Conference Book*, chapter A Dynamically Reconfigurable Architecture for Low-Power Multimedia Terminals. Kluwer Academic Publishers, 2002.

[17] W.S. VanScheik and R.F. Tinder. High Speed Externally Asynchronous/ Internally Clocked Systems. *IEEE Transaction on Computers*, 46(7):824–829, July 1997.

[18] S. Wuytack, J.Ph. Diguet, F. Catthoor, and H. De Man. Formalized methodology for data reuse exploration for low-power hierarchical memory mappings. *IEEE Transactions on VLSI Systems*, 6(4):529–537, December 1998.

[19] G.R. Goslin. A Guide to Using Field Programmable Gate Arrays (FPGAs) for Application-Specific Digital Signal Processing Performance. In *High- Speed computing, Digital Signal Processing and Filtering Using reconfigurable Logic*, pages 321–331. SPIE, 1995.

[20] A. D. Garcia. *Etude sur l'estimation et l'optimisation de la consommation de puissance des cir-*

cuits logiques programmables du type FPGA. PhD thesis, Ecole Nationale Supérieure des Télécommunications, Paris, 2000.

[21] K. K. W. Poon. Power Estimation For Field Programmable Gate Arrays. Master's thesis, University of British Columbia, 2002.

[22] L. Shang, A. S. Kaviani, and K. Bathala. Dynamic power consumption in Virtex-II FPGA family. In *International Symposium on Field-Programmable Gate Arrays*, pages 157–164, 2002.

[23] E. Kusse. Analysis and Circuit Design for Low Power Programmable Logic Modules. Master's thesis, University of California, Berkeley, 1997.

[24] Xilinx Inc. *FPGA Configuration Guidelines, XAPP090*, November 1997.

[25] Altera. *Configuring APEX20k, Flex10k and Flex6000 Devices, A.N. 116*, May 2000.

[26] Greg Stitt, Brian Grattan, Jason Villarreal, and Franck Vahid. Using On-Chip Configurable Logic to Reduce System Software Energy. In *IEEE Symposium on Field-Programmable Custom Computing Machines*, 2002.

[27] Jason Villarreal, Dinesh Suresh, Greg Stitt, Franck Vahid, and Walid Najjar. Improving Software performance with Configurable Logic. *Design Automation for Emebedded Systems*, 7(4):325–339, November 2002.

[28] Xilinx Inc. *CoolRunner-II CPLD Family*, March 2003.

[29] I. Brynjolfson and Z. Zilic. Fpga clock management for low power applications (poster abstract). In *International Symposium on Field programmable gate arrays*, page 219, 2000.

[30] Xilinx Inc. *VIRTEX2 1.5V Series Field Programmable Gate Arrays*, July 2001.

[31] V. George. *Low Energy Field-Programmable Gate Array*. PhD thesis, University of California, Berkeley, 2000.

[32] A. Tisserand, P. Marchal, and C. Piguet. An on-line arithmetic based FPGA for low-power custom computing. In *International Workshop on Field Programmable Logic and Applications*, pages 264–273. Lecture notes in Computer Science 1673, August/September 1999.

[33] G. C. Cardarilli, A. Del Re, A. Nannarelli, and M. Re. Residue Number System Reconfigurable Datapath. In *International Symposium on Circuits and Systems*, pages 756–759, May 2002.

[34] M. A. Soderstrand, W. K. Jenkins, G. A. Jullien, and F. D. Taylor. *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*. IEEE Press Reprint Series, 1986.

[35] C.-S. Chen, T. Hwang, and C. L. Liu. Low power FPGA design: A re-engineering approach. In *Conference on Design Automation (DAC-97)*, pages 656–661, NY, USA, June 9–13 1997.

[36] G. Sutter, E. Todorovich, S. Lopez-Buedo, and E. Boemo. Low-power FSMs in FPGA: Encoding alternatives. *Lecture Notes in Computer Science*, 2451:363–370, 2002.

[37] Altera. *Stratix FPGA Family*, December 2002.

[38] U. Meyer-Baese, J. Ramírez, and A. García. Low power high speed algebraic integer frequency sampling filters using FPLDs. In *International Workshop on Field Programmable Logic and Applications*, pages 897–904, Montpellier, France, September 2002. Lecture Notes in Computer Science 2348.

[39] E. I. Boemo, S. López-Buedo, and J. M. Meneses. Some experiments about wave pipelining on fpga's. *IEEE Trans on Very Large Scale Integration Systems*, 6(2):232–237, June 1998.

[40] S. C. Goldstein, H. Schmit, M. Budiu, S. Cadambi, M. Moe, and R. R. Taylor. PipeRench : A Reconfigurable Architecture and Compiler. *IEEE Computer*, 33(4), April 2000.

[41] S. Choi, R. Scrofano, V. Prasanna, and J.W. Jang. Energy-Efficient Signal Processing Using FPGAs. In *International Symposium on Field Programmable Gate Arrays*, pages 225–234, Monterey, USA, February 2003.

[42] P. Bertin, D. Roncin, and J. Vuillemein. Introduction to programmable active memories. In *Systolic Array Processor*, pages 301–309. Prentice Hall, 1989.

[43] S. Park and W. Burleson. Reconfiguration for power saving in real-time motion estimation. In *International Conference on Acoustics, Speech and Signal Processing*, pages 3037–3040, May 1997.

[44] S. Park and W. Burleson. Configuration cloning: Exploiting regularity in dynamic DSP architectures. In *International Symposium on Field Programmable Gate Arrays*, pages 81–89, February 1999.