Architecture à base de FPGA reconfigurable dynamiquement dédiée au traitement d'image sur flot de données.

H.Guermoud, Y.Berviller, E.Tisserand, S.Weber

Laboratoire LIEN
Faculté des Sciences, Nancy I
BP239
54506 Vandoeuvre-lès- Nancy cedex
FRANCE
Tel:(33) 83 91 20 71, Fax: (33) 83 91 23 91, E-mail: guermoud@lien.u-nancy.fr

Résumé:
La reconfiguration dynamique (RTR: run time reconfiguration) des FPGA est devenue un procédé de plus en plus utilisé par les systèmes reconfigurables appliqués à des domaines très divers. Nous montrons dans cet article l'intérêt d'utiliser la RTR pour le traitement d'images. Nous proposons ainsi un système de vision temps réel mettant à contribution la reconfiguration dynamique des FPGA en vue d'implanter des algorithmes de traitements d'images bas niveaux. L'architecture du système a été validée par l'implantation d'un enchaînement de traitements (filtre médian séparable, gradient de Sobel et seuillage) permettant de montrer les nouvelles performances obtenues grâce à la RTR.

Abstract :
Run time reconfiguration (RTR) of FPGA's has became a method wich is used more and more by the reconfigurable systems. We show in this paper the interest of the RTR for image processing. We suggest a new real time vision system using run time reconfiguration of FPGA's with the aim of implement a low levels images processing algorithms. System's architecture has been checked by implementing piplined treatements, showing the performences enhancement obtained thanks to run time reconfiguration.

I. INTRODUCTION :

Depuis la commercialisation du premier prédiffusé programmable (FPGA) en 1985, l'utilisation de ces circuits ne cesse de s'étendre à des domaines et applications variés, parmi lesquels nous pouvons citer le traitement d'image [1][2], les réseaux de neurones [3], la comparaison de séquences génétiques [4], l'architecture des ordinateurs [5] etc...

L'intérêt suscité par les FPGA est dû essentiellement à leurs prix abordables, facilité de mise en oeuvre et flexibilité. En outre, les coûts fixes et délais de fabrications (NRE), en comparaison avec les circuits spécifiques (ASIC), sont totalement éliminés. Cependant, ils présentent une faible densité d'intégration de portes logiques et atteignent des fréquences de travail relativement faibles devant les ASIC.

Plusieurs systèmes à base de FPGA sont décrits dans la littérature. Nous pouvons les distinguer par leurs modes de configuration. Les premiers utilisent un mode de configuration statique, pour lequel nous avons une configuration du système par application (reconfiguration entre les applications), ex: Splash [6]. Les seconds utilisent un mode dynamique consistant à reconfigurer le système plusieurs fois durant l'exécution de la même application (ex:RRANN [3]).

Nous proposons dans cet article une architecture reconfigurable dynamiquement répondant aux contraintes de traitement d'images bas niveau à la cadence vidéo.

II. PRINCIPE:

Le principe de la reconfiguration dynamique consiste à reconfigurer un ensemble de FPGA-SRAM pendant l'exécution d'une même application. Cette dernière est au préalable décomposée en plusieurs étapes succesives. Chaque reconfiguration correspond alors à l'exécution d'une étape de l'application.

Le concept a déjà fait l'objet d'une application autre que le traitement d'images. La RRANN (run time reconfiguration artificial neural network) est l'une des applications les plus connues dans ce domaine. Elle exploite la RTR pour l'implantation de l'algorithme « backpropagation training » en utilisant les réseaux de neurones. Les résultats obtenus sont édifiants et se traduisent par une amélioration de la densité fonctionnelle du réseaux de près de 500% comparativement au même type d'architecture n'utilisant pas la RTR.

Le traitement d'images bas niveau est une autre application pouvant bénéficier des avantages de ce procédé. En effet, il peut naturellement être décomposé en un enchaînement d'opérateurs ou d'algorithmes s'exécutant en cascade (fig.1a).

Généralement, chaque algorithme est implanté dans un FPGA, l'utilisation de la RTR permettra d'implanter deux ou plusieurs algorithmes dans le même FPGA (fig 1b).

L'ensemble de l'opération doit s'effectuer sans perdre la notion du temps réel pour laquelle tous les traitements doivent s'exécuter au minimum à la cadence vidéo (25images/s). L'exploitation du principe de la RTR n'est rendu possible qu'après

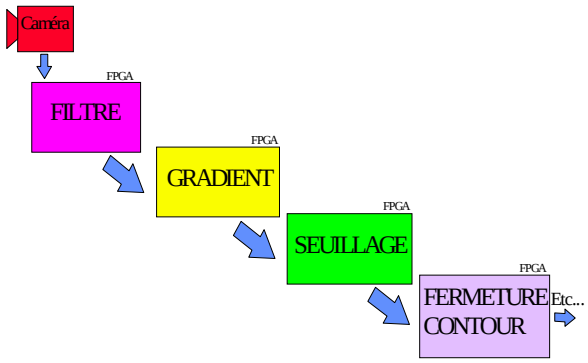

Fig 1a. Chaîne de traitement d'image classique.

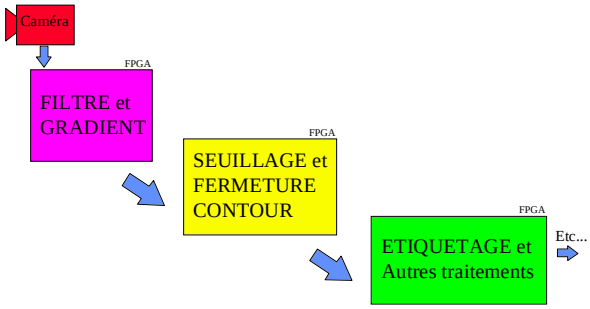

Fig 1b Chaine de traitement d'image utilisant la RTR.

avoir considérer un certains nombre de contraintes, ceci fait l'objet d'une étude détaillée présentée dans le paragraphe suivant.

III. METHODE PROPOSEE:

Afin de répondre aux contraintes imposées par les traitements sur flot vidéo, la reconfiguration des FPGA doit s'exécuter suivant les séquencements illustrés par la fig 2. L'exemple est donné sur deux étapes de traitement bas niveau.

Les deux traitements ainsi que les deux reconfigurations doivent s'effectuer pendant la durée d'une image soit 40ms. L'image considérée est de nature entrelacée. Pour ne pas gêner le déroulement des traitements nous pouvons déclencher les phases de reconfigurations pendant les périodes de suppression trame. Elles représentent chacune l'équivalent de 25 lignes vidéo soit 1,6ms, or cette durée n'est pas suffisante pour un FPGA de complexité moyenne dont le temps de reconfiguration est estimé à 6ms.

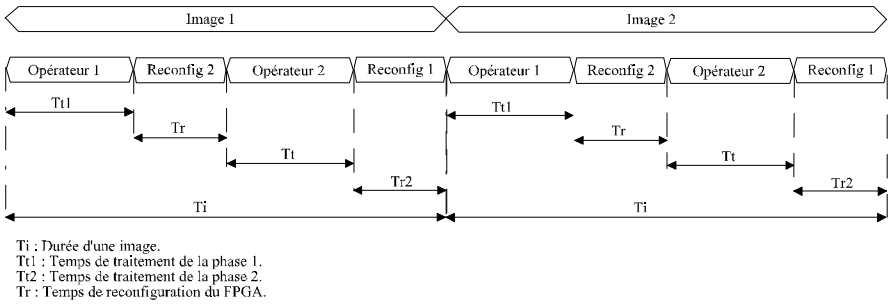

Fig 2 Séquencement des traitements et des phases de reconfigurations du FPGA.

Cependant pour une image de taille 512*512 pixels balayée à la fréquence de 10Mhz au standard CCIR, la durée du signal vidéo utile est de 26,2ms soit 13,8ms de temps mort (tops de synchronisation lignes et trames), lesquelles peuvent être exploitées pour la reconfiguration des FPGA. Afin de disposer continuellement de cette durée, nous proposons de désentrelacer l'image. La méthode fréquemment employée, utilise deux plans mémoires en lecture-écriture alternée. La phase d'écriture se fait à la fréquence de 10 Mhz, tandis que la phase de lecture s'exécute à une fréquence plus élevée. Nous imposons ainsi des traitements plus rapides qui nous permettent de disposer du temps nécessaire à la reconfiguration des *FPGA-SRAM*.

Il faut également préciser que pour les traitements utilisant des opérateurs locaux, nous n'introduisons aucune ressource matérielle supplémentaire. En effet le désentrelacement de l'image est nécessaire car les traitements doivent s'appliquer sur plusieurs lignes consécutives. Nous préconisons, dans ce cas, une augmentation de la fréquence de lecture. Celle-ci correspond évidemment à la fréquence des traitements, elle est déterminée par l'équation (1). Elle dépend de la taille de l'image n*n, du nombre N de reconfiguration que l'on souhaite effectuer, du temps de reconfiguration T_{FPGA}, et enfin de la durée de l'image T_i.

$$F_{min} = \frac{N*n^2}{T_i - N*T_{FPGA}} \qquad (1)$$

La figure 3 montre le nombre d'étapes que l'on peut implanter par FPGA en fonction de différentes fréquences de travail et différents temps de reconfiguration. Les courbes de la fig 3 ont été calculées pour une image de taille 512*512. Pour une fréquence de travail donnée, le nombre N de reconfiguration est inversement proportionnel au temps de reconfiguration du FPGA. Pour chaque nombre N de reconfigurations correspondent plusieurs fréquences de travail, cela dépend éventuellement du temps de reconfiguration du FPGA utilisé. Ceci nous confère, pour un nombre N de reconfigurations fixé, la possibilité de choisir le circuit adapté aux algorithmes que l'on souhaite implanter. En effet, le temps de reconfiguration des FPGA est directement proportionnel au nombre de portes utiles qu'ils renferment.

Nous avons effectué la même étude, cette fois-ci, avec des images de taille différentes (256*256, 128*128 et 64*64 pixels). La fig 4 montre bien que pour une image de taille 64*64 pixels le nombre N de reconfiguration dépend exclusivement du temps de reconfiguration du FPGA. En effet, l'équation (2) montre que le temps imparti aux traitements devient négligeable devant le temps de reconfiguration du FPGA lorsque la taille de l'image diminue sensiblement.

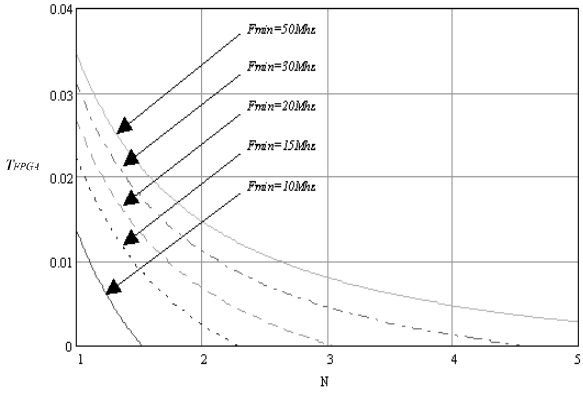

Fig. 3 Nombre d'étapes implantées par FPGA en fonction de la fréquence de travail et du temps de reconfiguration.

$$T_i = N\left(n^2 * T_{max} + T_{FPGA}\right) \qquad (2)$$

Grâce à cette analyse et à la connaissance à priori du nombre de reconfiguration que l'on veut effectuer par FPGA, nous sommes en mesure de déterminer le circuit FPGA-SRAM qui correspond le mieux aux algorithmes utilisés.

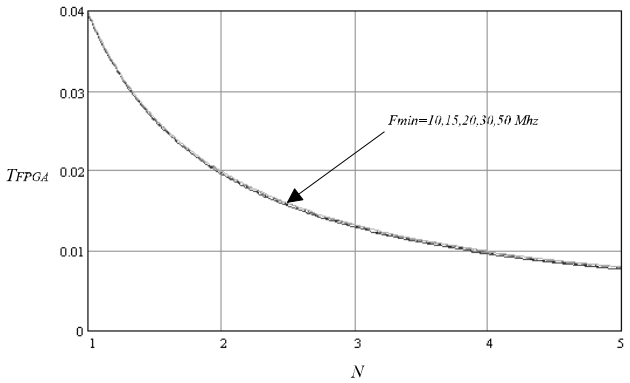

Fig 4 Le nombre d'étapes implantées est indépendant de la fréquence de travail.

IV. ARCHITECTURE DU SYSTEME DE VISION TEMPS REEL:

L'architecture du système de vision temps réel est decrite dans la Fig 5. Elle est composée de deux *FPGA XILINX (XC4004A)*, l'un faisant office d'unité de contrôle, l'autre d'unité de traitement, deux plans mémoires vidéo utilisés pour désentrelacer l'image et stocker les résultats intérmédiaires. Les deux Prom série (XC17128) contiennent chacune les fichiers correspondant respectivement au premier traitement (ex: filtre médian) et au deuxième traitement (ex: gradient Sobel).

A la mise sous tension, le séquencement de l'ensemble des opérations est assuré par l'unité de contrôle. La première étape consiste à autoriser le téléchargement du premier algorithme (filtre médian séparable) de la Prom1 vers l'unité de traitement. L'opération de filtrage peut ainsi commencer, elle s'effectue sur les données vidéo fournies par le module de désentrelacement. Il faut préciser que les données appartiennent à l'image précédente. Pendant ce temps, l'image courante s'enregistre dans le deuxième plan mémoire du module de désentrelacement. Le résultat du filtrage est stocké dans le plan à partir duquel les données vidéo ont été lues. Cela est possible car les mémoires utilisées sont à double port. Dès que l'opération de filtrage est terminée, l'unité de contrôle enclenche la reprogrammation de l'unité de traitement pour effectuer le gradient, contenu dans la Prom2.

Il est important de noter que le FPGA de l'unité de contrôle est configuré en maître tandis que l'autre, celui de l'unité de traitement, est configuré en esclave. En effet, c'est le mode de configuration des FPGA Xilinx qui permet la plus grande fréquence (10Mhz) de transfert des fichiers binaires entre les Proms et le FPGA (unité de traitement).

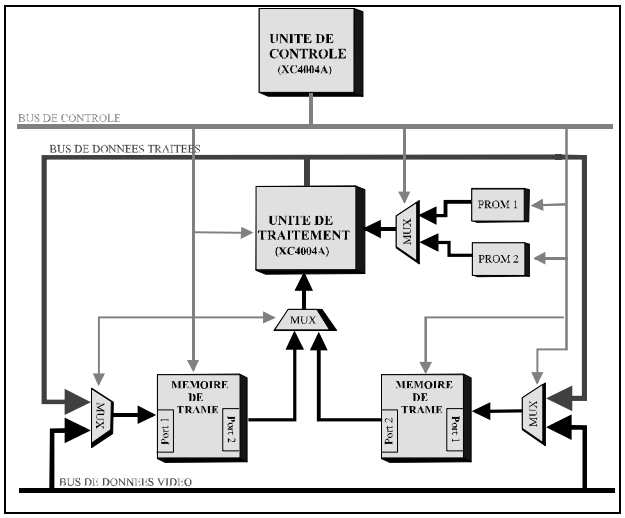

Fig 5 Architecture à reconfiguration dynamique.

V. CONCLUSION :

L'architecture de reconfiguration dynamique des *FPGA-SRAM* proposée est adaptée au traitement d'images bas niveau sur flot vidéo au standard *CCIR*.. Elle exploite le désentrelacement de l'image et permet, sans perte d'informations, de chaîner deux opérateurs (filtre Médian et gradient de Sobel) par la reconfiguration d'un circuit *FPGA*. La densité d'intégration fonctionnelle des circuits est ainsi accrue de 100%. Les résultats sur l'exemple décrit sont intéressants; nous travaillons sur la généralisation de la méthode, dans l'objectif de choisir le circuit le mieux adapté aux opérations à réaliser et de définir les contraintes sur les opérateurs.

VI. REFERENCES :

[1] S. C. Chan, H.O. Ngai and K.L. Ho "A programmable image processing system using FPGA" International journal of electronics, vol 75, N°4 pp 725-730, 1993.

[2] M. Alves de barros "Traitement bas niveau d'images en temps réel et circuits reconfigurables" Thèse de doctorat, Université de Paris-Sud, 1994.

[3] J.G. Eldrerge, B.L. Hutchings "Density enhancement of neural network using FPGAs and run-time reconfiguration" FCCM, 1994.

[4] E. Lemoine, J. Quinqueton and J. Salantin "High speed pattern matching in genetic data base with reconfigurable hardware" Proceeding of th 2nd INT. Conf. o, Intelligent systems for molecular biology, pp 269-276. AAAI, 1994.

[5] B.Heeb and C. Pfister "Chameleon, a workstation of a different colour" 2nd International Workshop on Field-Programmable Logic Applications, paper 5.6, Vienna, Austria, 1992.

[6] J. Arnold, D. Buell and E. Davis, "Splash II", in 4th ACM Symposium on parallel Algorithms and Architectures, San Diego, California, USA, 1992.