Utilisation de la Reconfiguration Dynamique pour la rotation d'images en temps réel

E. Bourennane, S. Bouchoux, M. Paindavoine et C. Milan

Laboratoire LE2I, Université de Bourgogne

Aile des sciences de l'Ingénieur, B.P. 47870, 21078 Dijon Cedex

Tél./Fax : 03 80 39 63 33, E-mail : ebourenn@u-bourgogne.fr

Résumé - Les circuits FPGAs sont largement utilisés aujourd'hui pour effectuer divers traitements en temps réel. L'arrivée des FPGAs reconfigurables dynamiquement a permis de réduire le nombre de portes logiques nécessaires pour réaliser une application formée d'une chaîne d'algorithmes de traitement d'image. Nous présentons dans cet article une application de traitement d'image, la rotation d'images, qui exploite la Reconfiguration Dynamique (RD) des FPGAs. Une comparaison est effectuée entre la reconfiguration dynamique et statique via deux critères, un critère de coût et un critère de performances. Il apparaît alors, que selon le cas de figure, la reconfiguration dynamique peut être plus ou moins avantageuse. Afin de pouvoir tester la validité de notre approche en terme d'Adéquation-Algorithme-Architecture, nous avons réalisé une carte "ARDOISE" basée sur un AT40K40.

Abstract *– FPGA component are largely used today to perform various algorithms, digital filtering, in real time. The emergence of the Dynamically Reconfigurable FPGAs permits to reduce the number of the necessary resources to achieve an image processing application, tasks chain. We present in this article an application of image processing, image rotation, that exploits the FPGA's dynamic reconfiguration method. A comparison is undertaken between the dynamic and static reconfiguration by using two criteria, a cost criterion and a performance criterion. It appears then, that according to the case, the dynamic reconfiguration can be less or more advantageous. To verify the validity of our Algorithm-Architecture-Adequacy methodology, we realized an AT40K40 based board "ARDOISE ".*

1. Introduction

L'arrivée des FPGAs, en milieu des années 80 [1], a fait naître de nombreuses applications de Traitement du Signal et des Images basées essentiellement autour de ces composants [2], [3]. Depuis, les FPGAs sont devenus plus performants en terme de densité d'intégration et de fréquence de fonctionnement [4]. Pour certains traitements, la contrainte de 25 images par seconde est aujourd'hui, de loin, assurée par les nouvelles générations de FPGAs. Une fois reçue, l'image est traitée à une cadence très élevée comparée à la fréquence d'acquisition des pixels, puis le FPGA se met en attente de la nouvelle image. Le temps ainsi perdu vient du fait que jusqu'à présent, il était impossible de reconfigurer les FPGAs dans un délai raisonnable pour une nouvelle tâche. Traditionnellement, une fois la configuration est optimisée pour une application bien définie, celle-ci est chargée dans le FPGA et y demeure jusqu'à la fin de l'application. Ce type d'implantation est appelée la configuration statique. L'arrivée des FPGA reconfigurables dynamiquement en 1989 [5] a permis d'optimiser le rendement temporel des FPGAs. Quand une cellule (ou un bloc de cellules) a fini l'exécution d'une tâche, elle peut être reconfigurée pour accomplir une autre tâche. Un FPGA est dit reconfigurable dynamiquement lorsqu'il est possible de ne reconfigurer qu'une partie du composant pendant qu'une autre partie est en exécution. Nous avons exploité cette particularité des FPGAs à reconfiguration dynamiques pour réaliser la rotation d'images en temps réel. Le choix de cet algorithme a été motivé, d'une part, par la nécessité de reconfigurer le composant à chaque ligne d'image et d'autre part, par sa structure de filtres IIR et FIR. Pour cette application, nous avons réalisé, dans le cadre du projet ARDOISE en collaboration avec les laboratoires ETIS, LIEN, LPSI et SOSSO une carte à base du FPGA ATMEL AT40K40. La suite de cet article est structurée comme suit : La deuxième partie de cet article donne e les critères d'analyse des implantations à reconfiguration statique et à reconfiguration dynamique. La troisième partie expose la reconfiguration dynamique au cas par cas selon la nature de l'application. La quatrième partie décrit l'algorithme de rotation d'images utilisé. La cinquième partie analyse les deux approches d'implantation (statique et dynamique) de l'algorithme de rotation d'images.

2. Critères d'analyse de l'implantation à configuration statique et à reconfiguration dynamique

L'analyse de l'implantation statique va porter sur l'évaluation de deux critères, l'un représente le coût de l'implantation et l'autre ses performances. Le coût sera désigné ici uniquement par le nombre de CLBs (Blocs Logiques Configurables) occupés par l'algorithme. N_{CLBs_Stat} est la somme des cellules de calcul et de contrôle. Le critère de performance (Perf.) peut être simplement donné par le temps d'exécution $T_{exéc\_Stat}$ nécessaire pour réaliser l'application. $T_{exéc\_Stat}$ sera exprimé seulement par le produit du nombre d'échantillons à traiter N (taille de l'image en pixels) multiplié par la période d'itération $T_{itération\_Stat}$ de

l'algorithme. $T_{itération\_Stat}$ est la période de l'horloge qui cadence les traitements dans FPGA. Ainsi, nous définissons l'efficacité temporelle d'utilisation du FPGA par le rapport de la fréquence d'utilisation du FPGA sur la fréquence maximale du FPGA : $E = f_{utilis}./f_{maxi}$. Nous définissons le coût dynamique par N_{CLBs_Dynam} et le critère de performance par $T_{exéc\_Dynam}$.

A performances égales, l'apport de la reconfiguration dynamique sera donné par le rapport :

$$\frac{N_{CLBs\_Stat} - N_{CLBs\_Dynam}}{N_{CLBs\_Stat}} X 100\ \% \qquad (1)$$

Nous utiliserons aussi le rendement spatio-temporel de chaque type d'implantation pour évaluer le degré d'optimisation de l'architecture mise en œuvre. Pour ceci, nous allons nous mettre dans l'hypothèse où le temps disponible pour effectuer tous les traitements sur une image est T_{dispo} et le nombre de ressources dont nous disposons est N_{CLBs_dispo}. En imagerie standard, nous disposons de 40 ms pour traiter une image.

3. Cas général

Comme nous l'avons signalé en introduction, la fréquences de fonctionnement des FPGAs a beaucoup progressé ces dernières années alors que certains besoins des utilisateurs sont restés les mêmes, en contrôle de qualité par exemple. Il s'en suit que les traitements d'une application utilisant la configuration statique des FPGAs soient exécutés en un temps très court et par conséquent, les FPGAs se mettent en attente de l'image suivante et restent inactifs. Même si l'application présente du pipeline, du fait de la technologie dont nous disposons, il est plus avantageux dans certains cas d'exploiter la reconfiguration dynamique que de continuer à utiliser une implantation à configuration statique. De ce fait, les applications se présentent plus souvent dans la situation tel que montrée sur figure 1. Nous allons voir qu'une utilisation de la configuration statique dans ce cas serait même une aberration. La figure 1 montre une application pouvant présenter ou non un pipeline et la durée de son exécution sur un FPGA classique en mode statique peut être inférieure à la durée d'acquisition de l'image : $M.N.T_{itération\_Stat} < T$.

Au vu de la figure 1, il s'agit d'une implantation statique à rendement spatial et à rendement temporel inférieurs à 1. Le rendement spatial est inférieur à 1 car toutes les tâches actives et inactives sont chargées dans le FPGA au démarrage de l'application bien qu'une seule soit active à la fois. Si de surcroît le FPGA statique en usage est trop rapide, par rapport aux besoins de l'application, il se trouvera sans activité pendant une certaine fraction de la durée d'acquisition de l'image. Nous obtenons ainsi un rendement temporel inférieur à 1. Le rendement spatio-temporel est donné par le relation (2).

— *Discussion de l'implantation statique :* Le coût et la performance sont donnés par l'équation (3). On peut noter la présence de $T_{inactivité}$ dans la relation $Perf_{Stat}$. Ce temps est dû à la puissance de calcul du FPGA employé.

$$\rho_{spatio\_tempo\_Stat} = \sum_{i=1}^{M} \left(\rho_{spatial\_Tâche\_i} \cdot \rho_{temporel\_Tâche\_i} \right) < 1$$

$$\rho_{spatial\_Tâche\_i} = \frac{N_{CLBs\_actifs\_Tâche\_i}}{N_{CLBs\_dispo}} < 100\ \% \quad \forall\ i \qquad (2)$$

$$\rho_{temporel\_Tâche\_i} = \frac{Temps\ d'exécution\ de\ la\ tâche\ i}{Temps\ disponible} . E_i$$

$$E_i = \frac{f_{utilis.\_i}}{f_{\max i.}}$$

$$Coût_{Stat} = N_{CLB\_Stat} = \sum_{i=1}^{M} N_{CLB\_Tâche_i} + N_{CLB\_Contrôle}$$

$$Perf_{Stat} = T_{exéc\_Stat} = M\ N\ T_{itération\_Stat} + T_{inactivité} = T \qquad (3)$$

— *Discussion de l'implantation dynamique :* Dans ce cas de figure, nous obtenons, grâce à la RD et sous une certaine condition, un gain en surface et en fréquence d'exécution tout en ayant les mêmes performances. La fréquence d'itération dynamique $f_{itération\_Dynam}$ peut même être inférieure à la fréquence d'itération statique.

$$Coût_{Dynam} = N_{CLB\_Dynam} = \underset{i=1}{\overset{i=M}{MAX}} \left(N_{CLB\_Tâche_i} + N_{CLB\_Contrôle} \right)$$

$$Perf_{Dynam} = MN T_{itération\_Dynam} + MT_{config} = T \qquad (4)$$

La période d'itération dynamique nécessaire est telle que :

$$T_{itération\_Dynam} = T_{itération\_Stat} + \frac{T_{inactivité}}{MN} - \frac{T_{config}}{N} \qquad (5)$$

On peut parvenir ainsi à obtenir une réduction de la surface et une diminution de la fréquence des traitements grâce à la reconfiguration dynamique.

4. Algorithme de rotation d'images

On rencontre souvent le problème d'interpolation d'images lorsqu'on veut restituer l'image analogique I(x) à partir de ses échantillons I(k). La forme analogique de l'image nous permet par la suite d'effectuer des opérations de sur-échantillonnage et de translation par des valeurs non-entières. Nous avons choisi comme application la rotation d'images dont le principe, selon l'algorithme de M. Unser, repose sur trois translations : translations suivant les lignes, puis selon les colonnes et à nouveau suivant les lignes [6], [7]. Les translations sont à valeurs réelles. Un point de l'image décalée provient d'un point situé entre deux pixels de l'image initiale. Comme ces points n'existent qu'à des positions entières, il est nécessaire d'interpoler l'image initiale pour calculer les valeurs de l'image à des positions intermédiaires. L'interpolation choisie est à base de la B-spline d'ordre 3. Le problème de l'interpolation d'image repose sur la détermination des coefficients C(j). Connaissant la séquence de pixels I(k) et les coefficients $\beta_n(k-j)$, il est possible de déduire C(j) de l'équation (6).

L'image translatée de Δ s'obtient par l'équation 7.

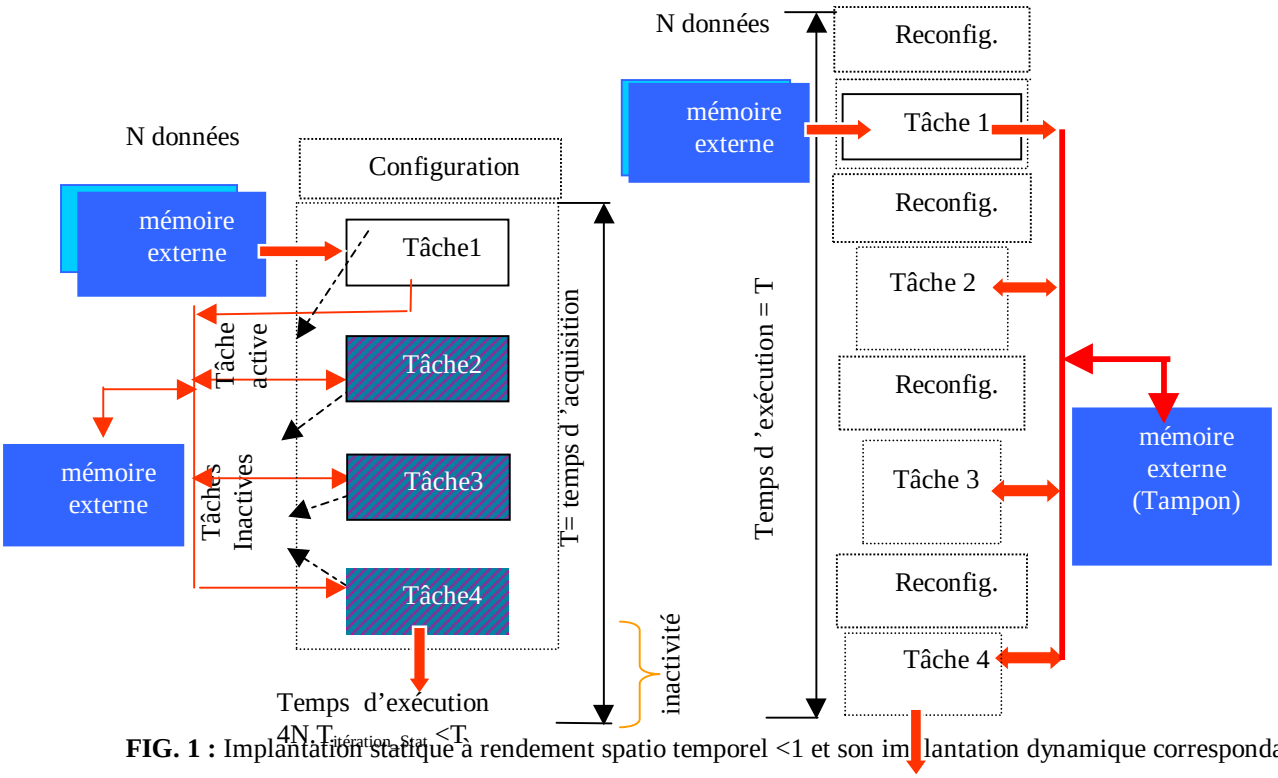

FIG. 1 : Implantation statique à rendement spatio temporel <1 et son implantation dynamique correspondante

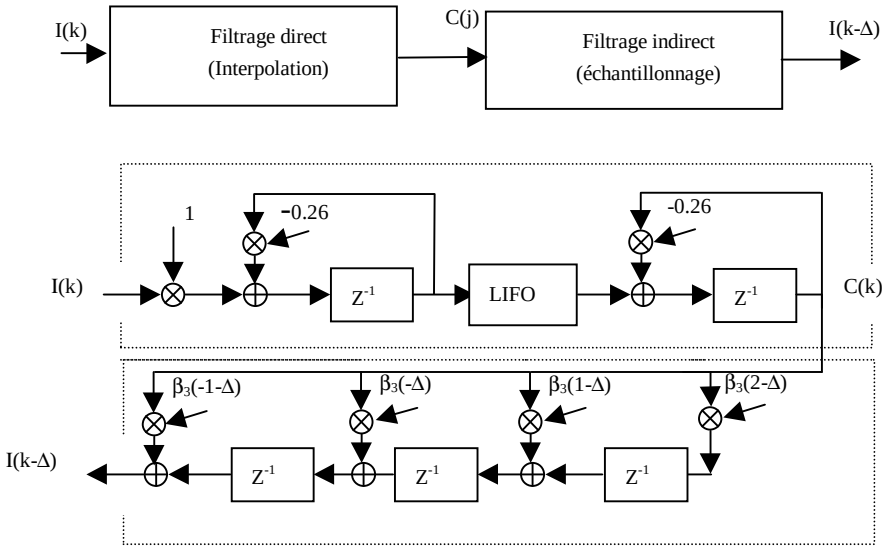

FIG. 2 : Structure du filtre de translation

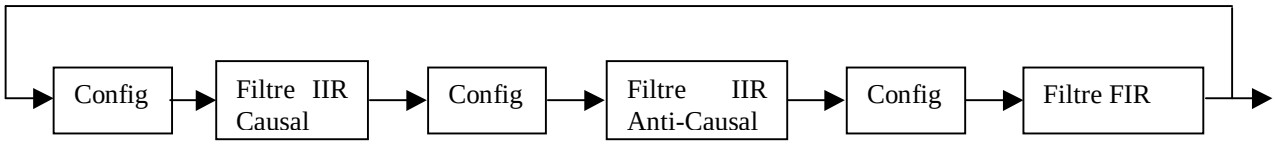

FIG. 3 : Partitionnement temporel de la translation d'image

$$I(k) = \frac{1}{6}\cdot C_{k-1} + \frac{4}{6}\cdot C_k + \frac{1}{6}\cdot C_{k+1} \Rightarrow \qquad (6)$$

$$C(z) = \frac{6}{z^{-1}+4+z}\cdot I(z) = \frac{1.6}{(1+0.26z^{-1})(1+0.26z)}\cdot I(z)$$

$$I(k-\Delta) = \sum_j C(j) B_3(k-\Delta-j) \qquad (7)$$

La figure 2 donne l'implantation des équations 6 et 7.

5. Analyse de l'implantation utilisant la reconfiguration statique/ dynamique globale

L'algorithme de rotation d'images est décomposable en une succession de trois translations : suivant les lignes, puis les colonnes et à nouveau les lignes. La figure 3 montre l'implantation d'une translation en la partitionnant en 3 tâches séquentielles.

Les calculs internes du filtre de la figure 2 sont effectués sur 13 bits de codage [8]. Le coefficient constant (1,6) est codé sur 7 bits alors que le coefficient (-0,26) est codé sur 6 bits. Les coefficients du filtre FIR (les B-splines) sont codés sur 9 bits. L'image translatée est codée sur 8 bits. Dans cette première version, nous avons utilisé des multiplieurs à coefficients variables crées à l'aide du générateur de Macros de l'outil IDS6.

— *Discussion de l'implantation statique :* Afin d'effectuer une comparaison sur la même base, l'évaluation du coût est effectuée en implantant l'algorithme de rotation d'images d'une manière statique sur la carte ARDOISE (AT40K40). Les trois translations qui composent notre application sont concurrentes. La fréquence d'exécution de ce composant est de 20MHz. Il en découle un temps d'inactivité du FPGA entre deux images successives. Ce temps d'inactivité sera exploité par l'implantation utilisant la reconfiguration dynamique pour réduire le nombre de ressources nécessaires à l'exécution de l'application. Pour cette application et vu le temps d'inactivité engendré par l'utilisation d'un AT40K40, une solution serait de n'utiliser qu'un FPGA et de le reconfigurer à chaque fin d'une translation. Il ne s'agirait plus d'une implantation à configuration statique mais d'une implantation à reconfiguration dynamique globale. Cette solution est analysée dans le point suivant.

Une translation est formée de deux filtres IIR et d'un filtre FIR et occupe 1600 cellules sur un total de 2024. Pour effectuer une rotation, il faut donc trois translations, soit 4800 cellules. Pratiquement, il faut 3 FPGAs AT40K40. Si les FPGAs étaient de taille adaptée aux besoins de la rotation, le coût statique serait :

$$Coût_{Stat} = N_{CLB\_Stat} = 4800\,CLBs$$

Cependant, dans la réalité, nous choisissons un FPGA avec l'ensemble des cellules qui le composent sachant que certaines ne seront pas utilisées. Dans notre cas, nous utilisons 3 FPGAs AT40K40 :

$$Coût_{Stat} = N_{CLB\_Stat} = 3x2024 = 6072\,CLBs$$

$$\begin{aligned} Perf_{Stat} &= T_{exéc\_Stat} = 3\,N\,T_{itération\_Stat} + T_{inactivité} \\ &= (20 + 20)\ ms = 40\ ms = T \end{aligned} \qquad (8)$$

— *Discussion de l'implantation dynamique :* L'implantation dynamique testée a été effectuée à la même cadence qu'en statique (20MHz) et sur le même composant AT40K40. Comme nous l'avons indiqué ci-dessus, neuf reconfigurations globales sont nécessaires pour effectuer les trois translations. Par translation, il faut deux reconfigurations pour les filtres IIR et une reconfiguration pour le filtre FIR. Le temps moyen de chaque reconfiguration globale est d'environ 0,5ms, soit un temps de reconfiguration par image de 4,5ms.

Pour effectuer la rotation d'une image, l'exécution de l'algorithme prend un temps de :

$$T_{IIR\_causal} + T_{IIR\_Anticausal} + T_{FIR} = 20ms$$

Le temps global de l'application, y compris les temps de reconfiguration, est de 24,5ms. La performance obtenue est identique à celle obtenue au moyen de l'implantation à configuration statique. Le coût dynamique est donné par l'équation 9.

$$\begin{aligned} Coût_{Dynam} &= N_{CLB\_Dynam} = \underset{i=1}{\overset{i=3}{MAX}} \left(N_{CLB\_Tâche_i} + N_{CLB\_Contrôle} \right) \\ &= 2NCLB_{FIR} = 1011CLBs \\ Perf_{Dynam} &= T_{exéc\_Dynam} + T_{config.\_total/image} + T_{inactivité} \\ &= (20 + 4.5 + 15.5)\ ms = 40\ ms = T \end{aligned} \qquad (9)$$

6. Conclusion

Cette première implantation de la rotation d'image en utilisant la reconfiguration dynamique nous a permis d'ores et déjà de cerner quelques problèmes propres à l'aspect de la reconfiguration dynamique. Un exemple de réalisation concrète, la rotation d'images, est donné pour illustrer notre étude. Sous la contrainte de la bande passante des données de la carte ARDOISE actuelle, nous étions amenés à proposer une solution sous optimale à notre problème de rotation d'images.

Références

[1] W. S. Carter, Khue Duong, Ross H. Freeman, Hung-Cheng Hsieh, Jason Y. Ja, John E. Mahoney, Luan T. Ngo, and Shelly L. Sze, *A User Programmable Reconfigurable Logic Array*. In IEEE 1986 Custom Integrated Circuits Conference, pp. 233-235. IEEE, May 1986.

[2] J. Waldemark, M.Millberg, T. Lindblad and K. Waldemark. *Image analysis for airborne reconnaissance and missile applications*. revue Elsevier, Pattern Recognition Letters, N°21, pp239-251, 2000.

[3] [PAR 00] S. W. Park, Y. Seo and K. S. Hong. *Real-Time Camera Calibration for Virtuel Studio*. Real-Time Imaging, N°6, pp 433-448, 2000.

[4] [ATM 99] Atmel Co.. *AT40K FPGAs with FreeRAM*. http://www.atmel.com/atmel/products/prod99.htm

[5] J.P. Gray and T.A. Kean. *Configuration hardware : New paradigm for computation*. In Decennial cal Tech conference on VLSI, pp. 277-293, Pasadena, CA, 1989

[6] M. UNSER, A. ALDROUBI , M. EDEN. *B-Spline Signal Processing : PartI-Theory*. IEEE Transaction on Signal Processing, Vol. 41. N°2. February 1993.

[7] M. UNSER, P. THEVENAZ , L. YAROSLAVSKY. *Convolution-Based Interpolation for fast, High-Quality Rotation Images*. IEEE Transaction on Signal Processing, Vol. 4. N°10. October 1995.

[8] C. Berthaud, E. Bourennane, M. Paindavoine et C. Milan. *Implementation of a real time image rotation using B-spline interpolation on FPGA's board*. SPIE, pp 512-519, San Diego, 1998.