Implémentation matérielle du filtre anti-bloc pour la norme H.264/AVC

KTHIRI MOEZ[1], PATRICE KADIONIK[1], BEN ATITALLAH AHMED[2], BERTRAND LE GAL[1], HERVE LEVI[1]

[1] Laboratoire IMS - ENSEIRB-MATMECA, Université Bordeaux 1, CNRS UMR 5218
351, Cours de la Libération, 33 405 Talence Cedex, France
[2] Université de Sfax, Institut Supèrieur d'Electronique et Communication de Sfax,
BP 868, 3018 Sfax, TUNISIA

[1] kthiri@enseirb-matmeca.fr, kadionik@enseirb-matmeca.fr, bertrand.legal@ims-bordeaux.fr, herve.levi@ims-bordeaux.fr
[2] ahmed.benatitallah@isecs.rnu.tn

Résumé - La norme H.264/MPEG-4-part10 est un standard de compression vidéo qui permet de réduire de moitié le débit de transmission ou de stockage pour une qualité visuelle équivalente aux normes précédentes [1]. Cette norme intègre un filtre anti-bloc « *deblocking filter* » ou « *loopfilter* » pour améliorer la qualité visuelle des séquences vidéo en éliminant certains effets indésirables du codage comme les effets de blocs introduits par la segmentation des images et par la transformation entière. Ce filtre permet une meilleure prédiction apportant des gains dans le débit binaire, en général de 5% à 10% [1], tout en produisant la même qualité objective que la vidéo non filtrée. Cependant, le fonctionnement de ce filtre est la partie la plus complexe du décodeur H264/AVC. Il représente quasiment le tiers de la complexité de calcul [2]. En effet, cette caractéristique exige l'utilisation d'une implémentation matérielle pour un filtre anti-bloc qui est nécessaire pour les applications vidéo haute définition (HD). Plusieurs accélérateurs matériels ont été proposés dans la littérature au cours des dernières années pour la réalisation architecturale de ce filtre. La plupart de ces accélérateurs matériels utilisent une seule unité de filtrage pour mener à bien les opérations de filtrage dans les deux directions (horizontale et verticale). Ces approches nécessitent moins d'espace mais elles ne répondent pas aux exigences de débit pour le traitement en temps réel. Par conséquent, les solutions basées sur plusieurs corps de filtrage qui s'exécutent en parallèle nous permettent de fournir un meilleur débit sans une augmentation considérable de la surface d'intégration.

1 Algorithme de filtre anti-bloc

1.1 Profilage du décodeur H.264/AVC

L'étude de la complexité du décodeur est faite sur le processeur embarqué PowerPC. Le processeur PowerPC est le cœur de notre système embarqué. Il est connecté aux différents périphériques à travers le bus PLB « *Processor Local Bus* ». En effet, après le portage de linux sur le processeur PowerPC 440 du FPGA Xilinx virtex 5, il semble, par conséquent, nécessaire d'étudier la répartition du temps CPU suivant les différents traitements constituant l'algorithme de décodage. Cependant, on a utilisé l'appel system RDTSC « *Read Time Stamp Counter* » pour détecter les goulots d'étranglement dans le décodeur H.264/AVC. L'instruction RDTSC donne le nombre de cycles d'horloge écoulés et retourne la valeur courante du compteur de temps du microprocesseur dans le couple de registres ADX:EAX. Par conséquent, le profilage du décodeur H.264/AVC est résumé dans la figure 1.

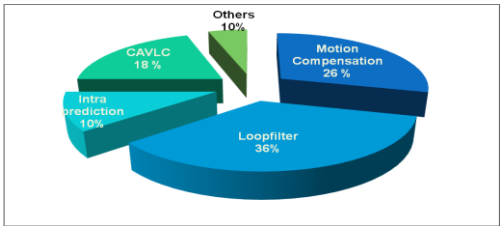


Figure 1 : Profilage du décodeur H.264/AVC

Le graphe de la figure 1 montre que la complexité est fortement localisée dans le module du filtre anti-bloc (36 % du temps CPU). En effet pour optimiser la complexité de ce bloc, nous proposons une architecture matérielle à fin de l'utiliser comme un bloc IP au sein d'un système embarqué.

1.2 Filtre anti-bloc

D'après la norme H.264/AVC, les bords verticaux sont filtrés de gauche à droite suivis par le filtrage des bords horizontaux de haut en bas. La seule restriction, en ce qui concerne l'ordre de traitement, précise que le processus de filtrage des bords verticaux doit avoir lieu avant le filtrage des bords horizontaux. La figure 2 présente les index des blocs 4*4 pour la luminance et la chrominance avec la partie supérieure (T) et les blocs voisins à gauche (L).

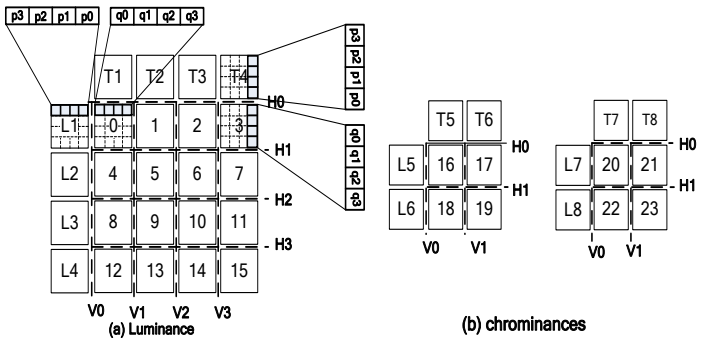


Figure 2 : Ordre de filtrage horizontal et vertical des blocs 4*4 pour la luminance et la chrominance

La figure 3 illustre un groupe de 4*4 pixels présentés par des zones ombres. Le groupe précédent de pixels est appelé P (bloc précédent) et celui en cours est appelé Q

(bloc courant). En effet, dans la première étape, le groupe cible est un bloc Q et le bloc P est le groupe précédent. Dans la seconde étape, le groupe cible représente le bloc P et le bloc Q est le groupe postérieur. Des considérations analogues sont valables pour les deux dernières étapes.

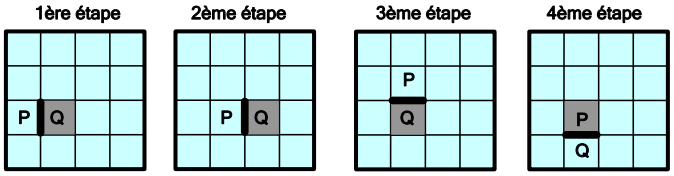


Figure 3 : Quatre étapes de filtrage d'un groupe 4*4 de pixels

Le filtre anti-bloc utilise un paramètre appelé BS « *Boundary Strength* ». Ce paramètre détermine l'intensité de filtrage. Il dépend de la localisation du bloc et d'autres informations (modes de prédiction, nombres d'images de référence…). La valeur du BS varie entre 0 (pas de filtrage) et 4 (maximum de filtrage).

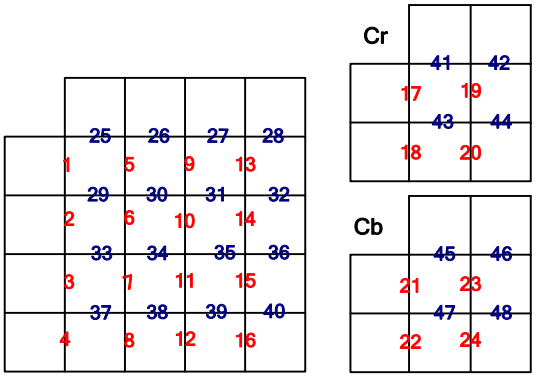
Par conséquent, pour BS=4, jusqu'à trois pixels de chaque côté du bord peuvent être modifiés, tandis que pour ($1 \leq BS \leq 3$), un maximum de deux pixels de part et d'autre du bord peut être affecté. En effet, une fois que la valeur de BS a été déterminée pour un bord, une analyse de type gradient est effectuée afin de déterminer si le bord doit être préservé ou filtré pour atténuer le phénomène d'artefact.

Dans ce sens, le filtre est désactivé quelle que soit la valeur de BS lorsque $|p0-q0| \geq \alpha$, $|p1-p0| \geq \beta$, ou $|q1-q0| \geq \beta$, où α et β reposent essentiellement sur le paramètre de quantification.

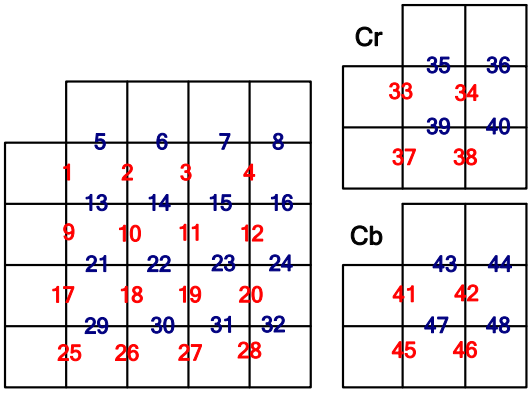
2 Travaux connexes

La figure 4(a) présente l'ordre de filtrage proposé par le standard H.264/AVC [1]. Les résultats du filtrage des bords verticaux sont utilisés dans les opérations du filtrage des bords horizontaux, d'où la nécessité de stocker les résultats intermédiaires.

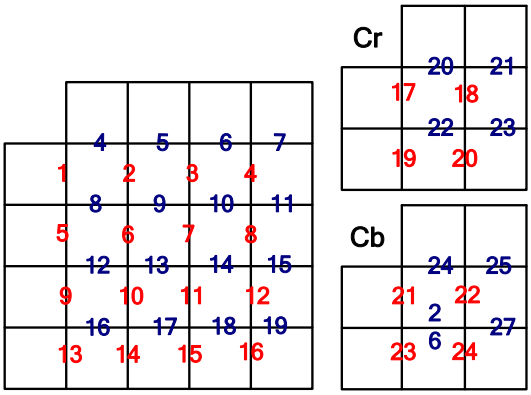
Par conséquent, cet ordre de traitement est très coûteux en termes d'utilisation mémoire et de temps d'exécution car on aura besoin de stocker 24 blocs 4*4 (16 blocs de luminance, 4 blocs de chrominance rouge et 4 blocs de chrominance bleu).



(a) Ordre de filtrage proposé par le standard [1]



(b) Ordre de filtrage proposé par G. Khurana [3]



(c) Ordre de filtrage proposé par He jing [4]

Figure 4 : ordre de filtrage au sein d'un Macrobloc

L'ordre de filtrage proposé par G. Khurana [3], présenté figure 4(b), est basé sur une alternance entre le filtrage horizontal et le filtrage vertical des blocs 4*4 au sein du Macrobloc. Cette solution conduit à une diminution de la taille de la mémoire locale utilisée car une seule ligne des blocs 4*4 doit être conservée pour être utilisée par les prochaines étapes de filtrage. De la même manière, la proposition d'He Jing [4] (figure 4(c)) est basée sur le principe de réutilisation des données. Pour cela, deux filtres sont nécessaires, un pour chaque direction de filtrage. Une autre architecture concurrente qui utilise quatre noyaux de filtrage est présentée par E.Ernst [5]. Le nombre de filtres utilisés est très limité en raison de la dépendance entre les données.

Les ordres de filtrages présentés ultérieurement sont tous réalisés au niveau des blocs, à savoir, le filtrage d'un bord du bloc 4*4 est réalisé en série par le même filtre notant que le bord d'un bloc peut être filtré seulement après le filtrage de quatre lignes de pixels (4 LOP) « *LOP : line of pixels* » du bloc précédent (situé à gauche).

L'architecture que nous proposons est basée sur un nouvel ordre de filtrage et sur une nouvelle organisation de la mémoire.

La figure 5 illustre l'ordre de filtrage proposé qui produit des résultats conformes à ceux de la norme H.264/AVC. Les chiffres répétés correspondent aux filtres qui s'exécutent en parallèle.

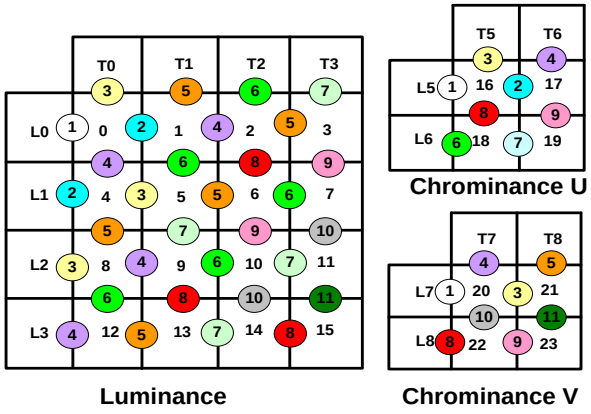


Figure 5 : Ordre de filtrage utilisé dans l'architecture proposée

En effet, l'idée est de paralléliser le traitement sans augmenter drastiquement l'utilisation d'espace mémoire.

3 Architecture proposée

L'architecture a été décrite en langage de description matérielle VHDL et tous les modules ont été synthétisés avec l'outil « Xilinx ISE Design suite 11 ». Afin de garantir que tous les transferts puissent être effectués en un seul cycle d'horloge, nous proposons d'utiliser 14 modules de mémoire locale (figure 6). Les données intermédiaires sont stockées dans des mémoires 4*32 bits et dans des registres (4*4 pixels) afin d'être utilisées pour le filtrage des prochains bords. Notre architecture du filtre anti-bloc, conçu comme un accélérateur matériel, utilise six unités de filtrage identiques pour améliorer le temps de traitement (trois filtres pour les bords verticaux et trois filtres pour les bords horizontaux afin d'obtenir un traitement simultané).

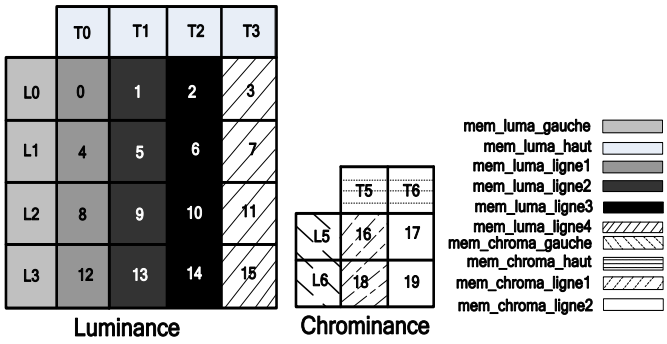


Figure 6 : Organisation de la mémoire locale.

L'architecture proposée contient aussi une unité de calcul du BS, un module de calcul du seuil, un module de calcul de C1 (critère de coupure), 12 modules de transposition, 6 registres 4*4 et 13 mémoires temporaires « *temporal buffer* ».
Ces derniers modules sont utilisés pour stocker les données intermédiaires à fin de les utiliser dans le filtrage des bords suivants.

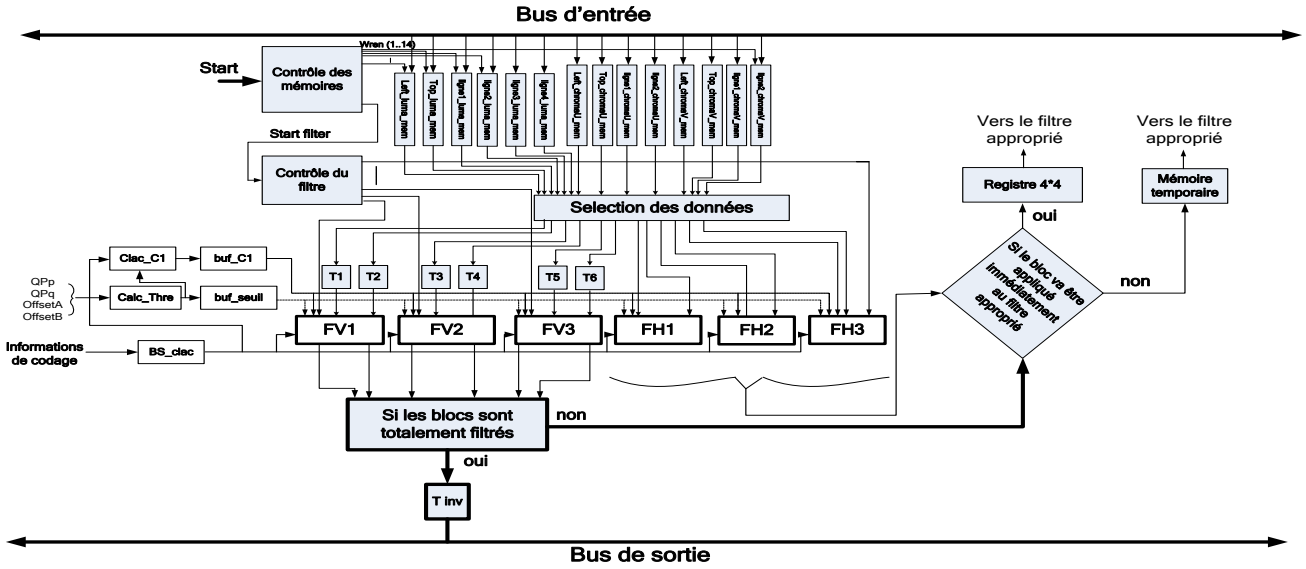


Figure 7 : Architecture du filtre anti-bloc proposée

Le stockage des données dans la mémoire appropriée est synchronisé par un module de contrôle via des signaux de contrôle. Ce module est conçu à l'aide d'une machine d'états. En effet, les mémoires utilisés par l'architecture proposée sont commandées par le signal *« wren »*. Par conséquent, si *« wren='1' »*, la mémoire reçoit les données et les stocke en fonction des adresses appropriées. Si non, si «*wren ='0' »*, on peut lire a travers les mémoires les pixels nécessaires pour le filtrage en fournissant juste les adresses générées par le module du contrôle.

En outre, les modules de transposition T et Tinv sont utilisés pour transposer les blocs 4*4 (ligne vers colonne et colonne vers ligne). L'unité T consomme un seul cycle d'horloge.

Tab 1. Les entrées et les sorties du module de trasnposition

Data input	Data output
a00,a01,a02,a03	a00,a10,a20,a30
a10,a11,a12,a13	a01,a11,a21,a31
a20,a21,a22,a23	a02,a12,a22,a32
a30,a31,a32,a33	a03,a13,a23,a33

En outre, le module « Contrôle du filtre » est responsable de la synchronisation de tous les transferts de données pour que les filtres récupèrent les nouvelles valeurs de pixels. L'architecture proposée effectue le

filtrage des bords horizontaux et des bords verticaux avec le même filtre en utilisant les modules Transposition.

La figure 8 résume le processus de filtrage au sein du Macrobloc en utilisant l'architecture proposée (les flux de données pour chaque quatre cycle d'horloge)

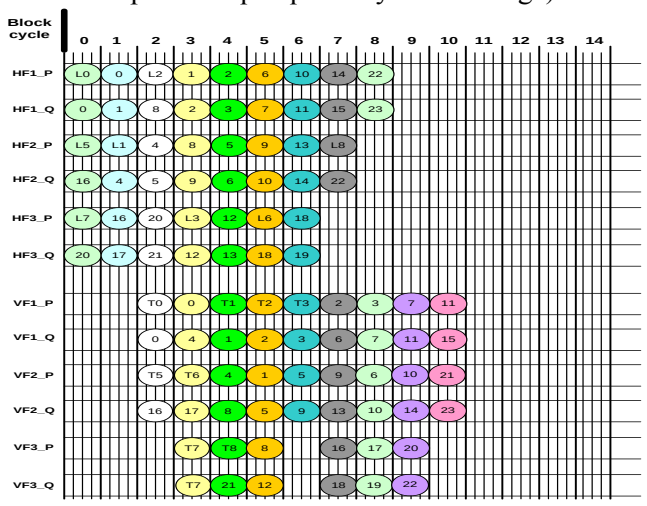


Figure 8 : Chronogramme de l'architecture proposée.

4 Résultats de simulation

A fin de valider l'accélérateur matériel conçu dans ce papier, plusieurs tests ont été effectués en faisant la comparaison des résultats obtenues avec les résultats fournis par le code source de la norme H.264/AVC décrit en langage C (la version JM 10.2).

La réduction du nombre de cycles de traitement du filtre anti-bloc est un point critique pour l'implémentation de la norme H.264/AVC partie décodeur. Cependant, avec l'ordre de filtrage proposé, nous pouvons filtrer un Macrobloc en seulement 49 cycles d'horloges (le filtrage d'un bord consomme 4 cycles d'horloge en ajoutant 1 cycle à cause du module de transposition utilisés dans le filtrage des bords horizontaux).

En effet, en se référant au Tableau 2, cette nouvelle architecture consomme un nombre de cycle d'horloge largement inferieurs aux autres architectures. En outre, on a réussi avec notre architecture proposée à atteindre une fréquence de fonctionnement amplement supérieure par rapport aux autres travaux ultérieurs.

Tab 2: Résultat de synthèse

	E. Ernst[5]	H.loukil[6]	Tobajas [7]	Jin-Woo [8]	M Nadeem[9]	**Notre architecture**
Technologies	Virtex5	Stratix II	0.18 µm	0.18 µm	0.18 µm	**Virtex5**
Fréquence (MHz)	43,40	150	36.45	200	166	**290**
Memoires utilisées	32 BRAMS	1792	64*32	0	64*32	**1728**
ALUTs ou portes	16,594	23874	12.600	22.300	12.060	**11,830**
Nombres de cycles/MB	70	105	110	132	112	**49**

Ce qui fait, la conception proposée est capable de nous aider à faire le traitement en temps réel et de réaliser des véritables applications vidéo 4096*2048 à 30fps « frame par seconde ».

5 Conclusion

Une nouvelle méthode de filtrage pour la norme H.264/AVC a été proposée. Elle permet d'atténuer le temps de traitement d'un Macrobloc de l'ordre de ~50% par rapport aux autres approches présentées dans la littérature grâce à l'exécution parallèle de six filtres.

En effet, avec l'utilisation de l'ordre de filtrage proposé qui favorise le parallélisme, les bords verticaux et les bords horizontaux peuvent être filtrés simultanément tout en tenant compte des contraintes proposées par le standard H.264/AVC. Cependant, le traitement d'un Macrobloc 4 :2 :0 prend seulement 49 cycles d'horloge. L'architecture a été mise en œuvre et validée sur un circuit FPGA Xilinx Virtex-5.

Les perspectives que nous pouvons envisager sont l'utilisation du bloc IP «*Intellectual Property*» développé pour l'implantation logicielle/matérielle de la norme de compression vidéo H.264/AVC à faible débit sur une plateforme comportant un processeur embarqué (processeur PowerPC du circuit FPGA Virtex-5 de Xilinx).

6 Bibliographie

[1] Joint Video Team of IT-T VEG and ISO/IEC MPEG, "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification", ITU-T Rec. H.264 and ISO/IEC 14496-10 AVC, 2003.

[2] M. Horowitz, A. Joch, F. Kossentini, and A. Hallapuro, "H.264/AVC baseline profile decoder complexity analysis", IEEE Trans. on Circuits and Systems for Video Technology, vol. 13, no. 7, pp.704-716, 2003.

[3] G. Khurana, T. Kassim, T. Chua, and M. Mi, "A pipelined Hardware Implementation of In-loop Deblocking Filter in H.264/AVC", IEEE Tran sactions on Consumer Electronics, 2006.

[4] He Jing, Huang Yan, Xu Xinyu, " An Efficient Architecture for Deblocking Filter in H.264/AVC", Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, 2009

[5] E. Ernst, "Architecture Design of a Scalable Adaptive Deblocking Filter for H.264/AVC", MSc Dissertation, Rochester, New York, 2007.

[6] H. loukil, A. Ben Atitallah, N. Masmoudi," Hardware architecture for H.264/AVC deblocking filter algorithm", IEEE International Conference on Systems, Signals and Devices , SSD 2009, pp 1-6.

[7] F. Tobajas, G. M. CalIic6, P. A. Perez, V. de Armas, and R. Sarmiento, "An Efficient Double-Filter Hardware Architecture for H.264 /AVC Deblocking Filtering", IEEE Transactions on Consumer Electronics, 2008.

[8] Jin-Woo Hwang and Jun-Dong Cho,"A Reconfigurable Pipelined Deblocking Filter for H.264/AVC", IEEE International Conference on Consumer Electronics, ICCE 2010, pp 403-404..

[9] M. Nadeem, S. Wong, G. Kuzmanov, A.Shabbir,"A High-throughput, Area-efficient Hardware Accelerator for Adaptive Deblocking Filter in H.264/AVC", IEEE/ACM/IFIP 7th Workshop on Embedded Systems for Real-Time Multimedia, ESTMED 2009, pp 18-19.