Etude d'un algorithme d'IDCT pour la conception d'une architecture reconfigurable implantant une DCT/IDCT 2-D 8x8

Thomas BOLLAERT, Mohamed AKIL

Département informatique, Groupe ESIEE
2 Bd Blaise Pascal, BP 99, 93162 Noisy-le-Grand Cedex (France)

RESUME

Un des problèmes des chaînes de traitement temps réel des images provient des algorithmes de compression (DCT) et de décompression (IDCT) souvent trop lents. Ce que nous proposons ici est notre étude d'un algorithme d'IDCT 2-D 8x8. Nous présentons une architecture très rapide basée sur l'utilisation de convolutions cycliques. Après réordonnement, les échantillons pairs et impairs sont traités en parallèle, permettant le traitement d'un bloc 8x8 en 64 cycles.

ABSTRACT

One of the major problems in real time video treatment comes from compression (DCT) and decompression (IDCT) algorithms which are often too slow. What we present here is our solution for a fast 2-D 8x8 IDCT algorithm. Our architecture is mainly based on the use of cyclic convolutions. The odd and even samples are reorganized and processed in two parallel parts. This solution allows the computation of an 8x8 block in 64 cycles.

1. INTRODUCTION

De part ses propriétés dans le plan fréquentiel, la DCT (Discrete Cosinus Transform) est devenue un opérateur indispensable dans les chaînes de compression d'images, l'opérateur de décompression associé étant la DCT Inverse (IDCT). Malheureusement ces algorithmes présentent l'inconvénient d'êtres lourds et peu adaptés à des implantations matérielles et peuvent devenir par la même occasion les goulots d'étranglement des chaînes de traitement trop rapides. Dès lors des solutions plus rapides pour ses opérateurs sont indispensables.

Les récents travaux de H. de Perthuis [1] ont déjà montré la pertinence de ses idées sur les convolutions et la DCT. Le but de cette communication est de prouver l'intérêt de l'utilisation de ces convolutions cycliques dans l'implantation de l'algorithme d'IDCT.

Nous commencerons par présenter les bases de l'algorithme classique d'IDCT avant d'aborder la présentation de notre méthode fondée sur les convolutions et ses performances.

2. L'ALGORITHME

L'IDCT 2-D 8x8 est définie par :

$$S(k,l) = \frac{1}{4}\sum_{i=0}^{7}\sum_{j=0}^{7} c_i c_j x_{i,j} \cos\frac{(2k+1)i\pi}{16}\cos\frac{(2l+1)j\pi}{16}$$

avec : $c_k = 1/(\sqrt{2})$ si k=0; 1 sinon

Etant linéaire, l'IDCT est séparable et devient donc équivalente à 16 IDCT 1-D effectuées sur 8 points. La formule de l'IDCT 1-D 8 points est donnée par :

$$S(k) = \frac{1}{2}\sum_{i=0}^{7} c_i x_i \cos\frac{(2k+1)i\pi}{16}$$

Une telle équation revient au calcul d'une matrice 8x8, solution peu souhaitable pour une implantation ASIC à

cause du nombre d'opérateurs nécessaires. Nous avons donc essayé d'appliquer l'idée proposée par H. de Perthuis dans [1] pour optimiser l'algorithme de DCT : transformer l'équation 1-D à l'aide d'une convolution cyclique.

3. LA CONVOLUTION ET L'IDCT

Pour cela nous avons défini la suite :

$$y_k = S(k) + S(k+1)$$
$$y_0 = 2S(0)$$

Sachant que cos(a+b)=2cos(a)cos(b)-cos(a-b), on trouve pour y_k :

$$y_k = \frac{x_0}{\sqrt{2}} + \sum_{i=1}^{7} x_i \cos\left(\frac{ik\pi}{8}\right)\cos\left(\frac{i\pi}{16}\right)$$

Notre calcul d'IDCT 1-D se ramène donc à un produit matrice vecteur rendu simple par la régularité de la matrice et le petit nombre de coefficients différents :

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} A & A & A & A & A & A & A & A \\ A & B & 1 & C & 0 & -C & -1 & -B \\ A & 1 & 0 & -1 & -A & -1 & 0 & 1 \\ A & C & -1 & -B & 0 & B & 1 & -C \\ A & 0 & -A & 0 & A & 0 & -A & 0 \\ A & -C & -1 & B & 0 & -B & 1 & C \\ A & -1 & 0 & 1 & -A & 1 & 0 & -1 \\ A & -B & 1 & -C & 0 & C & -1 & B \end{bmatrix} \begin{bmatrix} x_0c_0 \\ x_1c_1 \\ x_2c_2 \\ x_3c_3 \\ x_4c_4 \\ x_5c_5 \\ x_6c_6 \\ x_7c_7 \end{bmatrix}$$

où l'on a :

$$A = \sqrt{2}$$
$$B = \sqrt{2}\cos\left(\frac{\pi}{8}\right)$$
$$C = \sqrt{2}\cos\left(\frac{3\pi}{8}\right)$$
$$c_i = \frac{1}{\sqrt{2}}\cos\left(\frac{\pi}{16}\right)$$

Cette matrice sera considérée comme étant notre matrice de référence.

4. REORGANISATION MATRICIELLE

En excluant la ligne 0, qui pourra être traitée de façon très simple, nous pouvons faire les observations suivantes sur notre matrice :
Les coefficients des colonnes paires sont ±A, ±1, 0
Les coefficients des colonnes impaires sont ±B, ±C, ±1, 0.

Si l'on définit les couples de lignes 1 et 7, 2 et 6, 3 et 5, on remarque en plus que :
Dans les colonnes paires, les coefficients de chacun des membres d'un couple de lignes sont égaux.
Dans les colonnes impaires, les coefficients de chacun des membres d'un couple de lignes sont opposés.

En effectuant des permutations sur les lignes et les colonnes, nous pouvons donc réorganiser la matrice de la façon suivante :

partie	I	II			III			
kVi	0	2	4	6	1	3	5	7
1	A	1	0	-1	B	C	-C	-B
7	A	1	0	-1	-B	-C	C	B
2	A	0	-A	0	1	-1	-1	1
6	A	0	-A	0	-1	1	1	-1
3	A	-1	0	1	C	-B	B	-C
5	A	-1	0	1	-C	B	-B	C
4	A	A	A	A	0	0	0	0

On constate donc que si l'on traite en parallèle les parties II et III, l'une prenant en entrée les échantillons d'indice pair, l'autre les échantillons d'indice impair, on pourra obtenir en sortie simultanément deux composantes en additionnant ou en soustrayant les résultats de ces deux parties II et III. Ces résultats une fois déconvolués donneront les valeurs de l'IDCT 1-D 8 points.

Par exemple, si l'on appelle $P_{1,7}$ la valeur calculée par la partie II pour les lignes 1 et 7, et si l'on appelle $I_{1,7}$ la valeur calculée par la partie III pour ces mêmes lignes, on a alors :

$$P_{1,7} = Ax_0 + x_2 - x_6$$
$$I_{1,7} = Bx_1 + Cx_3 - Cx_5 - Bx_7$$

et donc :
$$y_1 = P_{1,7} + I_{1,7}$$
$$y_7 = P_{1,7} - I_{1,7}$$

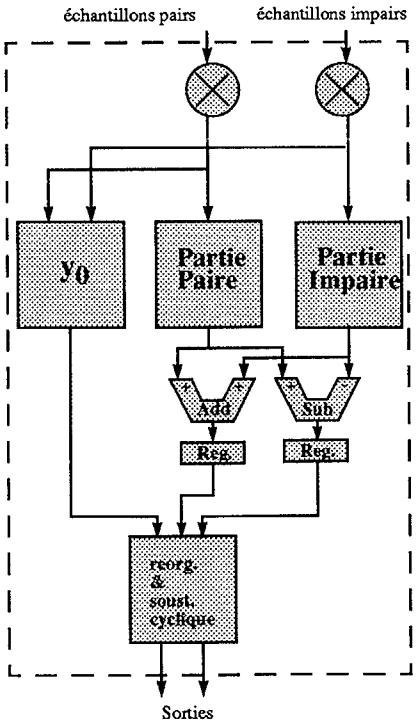
Les y sont ensuite déconvolués pour donner les S(k) valeurs de l'IDCT 1-D.

Cet exemple est significatif de la solution matérielle que nous proposons et que nous allons développer ci-après.

5. ARCHITECTURE

En entrée (*cf fig.1*) le circuit prend simultanément deux échantillons (un d'indice pair, un d'indice impair) et les multiplie par leur coefficient c_i associé. Les résultats sont ensuite envoyés sur les blocs chargés de calculer en parallèle y_0, la partie 'paire ' (parties I et II du tableau, *cf fig.2*) et la partie 'impaire' (partie III du tableau, *cf fig.3*). Les résultats des parties 'paire' et 'impaire' sont, une fois calculés, recombinés en somme et différence pour donner des couples de valeurs de y. Ces valeurs sont ensuite réorganisées pour être envoyées deux par deux sur le soustracteur cyclique établissant la convolution redonnant les S(i), résultats finaux de l'IDCT (*cf fig.4*).

fig.1 : architecture globale



Un des avantages de cette solution se situe en terme de nombre d'opérateurs. En ce qui concerne les multiplieurs, cette implantation n'en nécessite en effet que 5 dont 3 sont fixes (coefficients A, B et C).

fig.2 : partie 'paire'

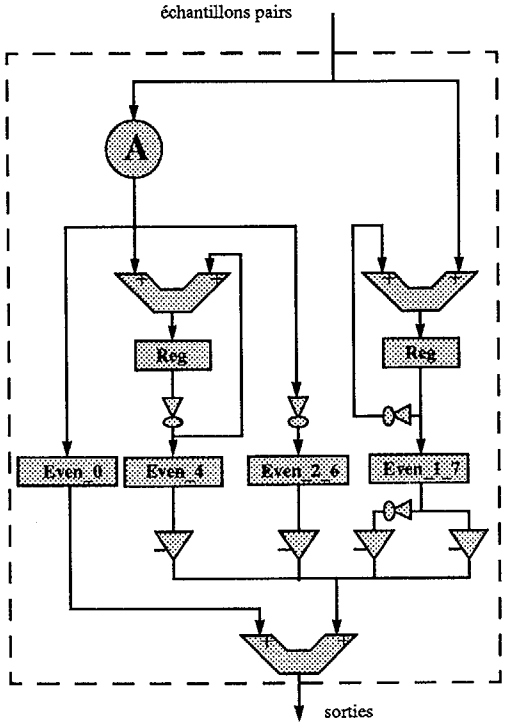
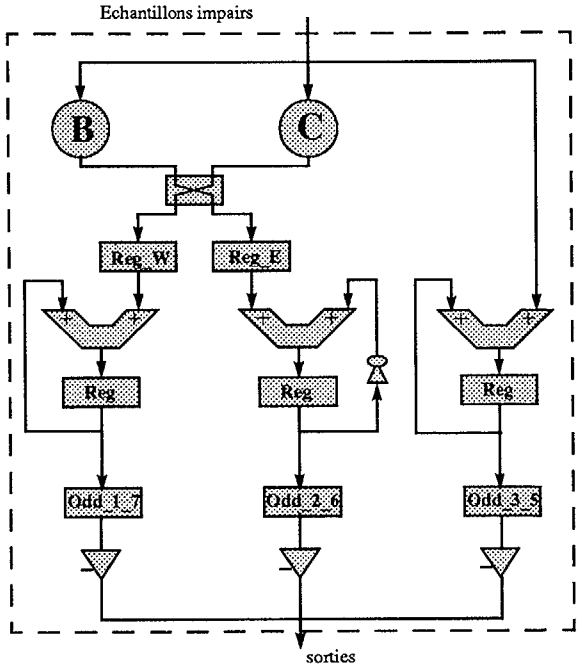
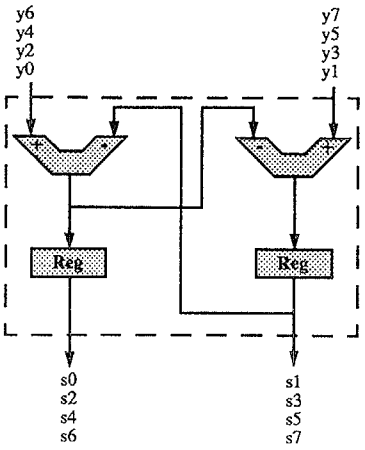


fig.3 : partie 'impaire'



Remarque : sur les figures 2 et 3, les symboles 'inverseur' représentent des opérateurs de complément à deux.

fig.4 : Soustracteur cyclique



rappelons que la convolution est définie par:

$$y_k = S(k) + S(k+1)$$
$$y_0 = 2S(0)$$

6. PERFORMANCES

Afin de tester l'efficacité de notre architecture, nous avons écrit un modèle VHDL que nous avons compilé, synthétisé sous Compass. Les premiers résultats font part d'environ 7000 portes (technologie 0.8µm) pour un fonctionnement à 33 MHz dans le cadre d'une approche ASIC. En ce qui concerne l'approche FPGA, but initial de notre étude, elle nécessiterait au moins deux Xilinx 4010 à cause du nombre de CLB (Control Logic Bloc) requis, un Xilinx 4010 contenant 400 CLB. Si les résultats semblent facilement améliorables dans le cas de l'ASIC, il n'en va pas de même pour le FPGA où la technologie des composants représente une limite.

7. CONCLUSION

Ce que nous proposons dans le cadre de cet article s'avère donc être une solution très rapide pour le calcul de la DCT Inverse. De part la structure même de notre circuit, une fois le pipeline chargé, il suffit de 4 cycles pour calculer une IDCT 1-D 8 points. Cela implique donc 32 cycles pour une IDCT 1-D 8x8 et 64 cycles pour une IDCT 2-D 8x8. Cadencé à 32MHz, notre circuit permettrait donc de calculer jusqu'à 500000 IDCT 2-D 8x8 à la seconde, ce qui suggère son utilité pour une chaîne de décodage MPEG2 par exemple. Enfin, si notre architecture semble mieux se prêter à une implantation ASIC, une approche FPGA permet, en tirant parti de la reconfigurabilité, de proposer un circuit effectuant au choix une DCT ou une IDCT pour un même nombre de composants.

8. REFERENCES

[1] H.de Perthuis, E.Bercovici, A.de Grandmaison, M.Akil : *"A fast VLSI architecture for 8x8 2-D DCT"* IS&T/SPIE Symposium on Electronic Imaging : Science and technology.
5-10 February 1995, San Jose, California, USA.