Architectures rapides dynamiquement reconfigurables des filtres numériques récursifs 1-D et 2-D

Fast dynamically reconfigurable architectures for 1-D and 2-D recursive digital filters

par Djamel CHIKOUCHE et Raïs El'hadi BEKKA

Institut d'Electronique
Université de Sétif
19000 Sétif, ALGERIE

résumé et mots clés

L'objectif de ce travail consiste à développer des architectures systoliques, aussi performantes que possible, pour des filtres numériques RII 1-D et 2-D nécessitant des calculs récursifs. La mise en œuvre directe des filtres RII sur les réseaux systoliques (type cylindrique) dynamiquement commutables est obtenue en les décrivant par des opérations matricielles dans l'espace d'état. Cependant, cette réalisation systolique engendre une latence proportionnelle à l'ordre du filtre. Pour améliorer d'une manière générale les performances en débit de données des réseaux de filtrage récursif, la solution proposée dans cet article repose sur la décomposition CTP de Porter qui transforme le produit d'une matrice par une colonne en un produit de trois matrices. Nous montrons que cette décomposition permet de réaliser des filtres RII par des structures cylindriques dynamiquement reconfigurables plus rapides. Néanmoins, le gain en débit de données est obtenu au détriment de la complexité de mise en œuvre. La version améliorée de la technique de décomposition CTP est appliquée aux filtres RII 1-D représentés par des matrices creuses du type tridiagonale dans l'espace d'état. Ce dernier algorithme permet une amélioration significative de la complexité matérielle.

Filtres numériques récursifs, processeurs, systolique, cylindrique, espace d'etat, matrices creuses, débit en données, latence.

abstract and key words

In this paper, we consider the array processors implementation of the infinite impulse response (IIR)1-D and 2-D digital filters that require recursive computations. We use the state space representation to obtain, in a straight forward manner, efficient implementation via dynamically switchable systolic arrays (cylindrical type) of 1-D direct realisation. This direct description leads to reduce the computation speed and the throughput rate. In order to improve, in a general way, the throughput rate performance of recursive filtering arrays, the solution proposed, in this paper, is based on the CTP decomposition technique of Porter which transforms the matrix-column product on a triple matrix product. It is shown in this work that this technique allows a realisation of IIR filters via dynamically reconfigurable cylindrical architectures that are much faster. However, this throughput improvement is obtained in the cost of a hardware complexity. The use of a sparse matrix of the tridiagonal type with the CTP decomposition permits a significant improvement of the hardware complexity of recursive filter arrays.

Recursive digital filters, array processors, systolic, cylindrical, state space, sparse matrix, throughput, latency.

1. introduction

Les réseaux de processeurs parallèles ont reçu une attention croissante dans le domaine du traitement numérique du signal ces dernières années [1]-[4]. Leurs caractéristiques de grande vitesse, de régularité et de modularité les rendent très adaptées pour la plupart des algorithmes de traitement numérique du signal. En particulier, les réseaux de processeurs pouvant exploiter effectivement les possibilités de la technologie VLSI constituent une solution aux exigences de traitement en temps réel.

Des progrès considérables ont été réalisés pour le développement de structures parallèles destinées aux calculs les plus complexes tels que : la convolution [5], la corrélation [6], la transformée de Fourier rapide [7], [8], la triangularisation [9], le filtrage en treillis [10] et le filtrage non linéaire [11]. Le produit matriciel, opérateur fondamental de la plupart des algorithmes de calculs linéaires, a été intensivement étudié dans les applications de réseaux de processeurs [12]. La majorité des algorithmes de traitement du signal utilise des opérations matrice/vecteur qui peuvent être effectuées à l'aide d'architectures parallèles. La vitesse de ces opérations matrice/vecteur est proportionnelle à la dimension des matrices[4], [13], [14]. Pour une matrice $N \times N$, une opération matrice/vecteur, utilisant N multiplieurs en parallèle, peut être réalisée en $O(N)$ unités de temps. Malheureusement, pour les grandes valeurs de N, cette vitesse de calcul n'est pas toujours adaptée aux applications en temps réel. Porter [15]-[17] a développé une classe spéciale d'algorithmes de calculs linéaires qui permettent un calcul en $O(\sqrt{N})$ unités de temps. Il a démontré que chaque algorithme de calculs linéaires peut être décomposé en une somme finie d'algorithmes (CTP : Concurrent Triple Product). L'auteur a trouvé une liaison directe entre ces algorithmes de forme rapide [15], [16] et les architectures systoliques dynamiquement commutables [13], [14].

L'objectif de ce travail consiste à développer des architectures systoliques, aussi performantes que possible, pour des filtres numériques RII 1-D et 2-D nécessitant des calculs récursifs. En effet, les calculs récursifs dans ce type de filtres introduisent une latence de N étapes de calcul dans le fonctionnement du réseau (N étant l'ordre du filtre). Les architectures systoliques ont été particulièrement appliquées aux filtres non récursifs RIF [18], [19]. Les premiers travaux relatifs aux filtres récursifs RII ont été introduits par Parhi et Messerschmitt [20], [21]. Cette approche utilisant la technique systolique à bit parallèle a permis d'obtenir un débit en données élevé, néanmoins elle augmente la complexité de mise en œuvre d'un facteur égal au niveau du pipeline. Pour réduire cette complexité de mise en œuvre tout en conservant un débit en données élevé, nous avons proposé dans un travail précédent [22] l'implémentation des filtres récursifs sur les réseaux cylindriques dynamiquement commutables [13] selon l'approche de Porter qui exige moins de matériel. Cette description offre l'avantage d'une mise en œuvre directe sur les réseaux systoliques (type cylindrique) dynamiquement commutables. Cependant, l'architecture obtenue engendre une réduction de la vitesse de calcul et du débit en données du réseau. Néanmoins, l'utilisation des matrices creuses du type tridiagonal pour représenter les filtres récursifs permet de réduire la complexité de mise en œuvre et d'augmenter le débit en données de ces structures. Toutefois, cette précédente approche est limitée par la forme des matrices tridiagonales qui est délicate à synthétiser.

Dans cet article, nous proposons une autre méthodologie de mise en œuvre des filtres récursifs 1-D et 2-D basée sur les résultats des algorithmes de Porter [16] qui permettent de transformer le produit d'une matrice par une colonne de l'algorithme de filtrage en un produit de trois matrices demandant moins de calculs (décomposition CTP). Celle-ci présente l'avantage d'une mise en œuvre directe du double produit matriciel sur une architecture cylindrique dynamiquement commutable nécessitant moins de processeurs élémentaires que la structure déjà proposée [22]. En outre, une simplication de la décomposition CTP en un simple produit de deux matrice [23] a été exploitée. Ce dernier algorithme conduit à une réalisation élémentaire. Ces nouvelles structures permettent une nette augmentation du débit en données. Dans la section 2, nous rappelons la conception des filtres récursifs 1-D et 2-D, leur représentation d'état tridiagonale et les notions de débit en données. La section 3 présente les réseaux cylindriques dynamiquement commutables des filtres RII proposés dans un de nos travaux précédents. Les avantages et les inconvénients de ce type de réalisations sont examinés. La section 4 traite le problème d'implémentation des filtres récursifs sur les architectures systoliques dynamiquement reconfigurables plus rapides que celles abordées dans la section 3. La rapidité de ces dernières est due essentiellement à l'application de la décomposition CTP à l'algorithme de filtrage. En plus, cette technique constitue une solution plus générale. La section 5 concerne le compactage des calculs dans les réseaux systoliques reconfigurables par l'usage des matrices creuses qui contribue à réduire la complexité des calculs. Une comparaison de l'architecture proposée avec celles de Parhi et Woods est faite dans la section 6. Enfin la section 7 conclut l'article.

2. filtres numériques recursifs

2.1. filtrage numérique récursif 1-D forme tridiagonale

Un filtre récursif ou à réponse impulsionnelle infinie RII 1-D peut être décrit par une équation aux différences :

$$y(n) = \sum_{i=0}^{N} b_i e(n-i) - \sum_{i=1}^{N} a_i y(n-i) \qquad (1)$$

Au filtre RII 1-D d'ordre N est associé la représentation d'état d'ordre N de la forme issue de [22], [24]-[26] et composée de :

– l'équation d'état

$$\mathbf{x}(n+1) = \mathbf{A}\mathbf{x}(n) + \mathbf{B}e(n) \tag{2a}$$

– et l'équation de sortie

$$y(n) = \mathbf{C}\mathbf{x}(n) + \mathbf{D}e(n) \tag{2b}$$

où : $\mathbf{x}(x) \in R^N$, $e(n) \in R$ et $y(n) \in R$; les matrices du filtre $\mathbf{A}(N \times N)$, $\mathbf{B}(N \times 1)$, $\mathbf{C}(1 \times N)$, et $\mathbf{D}(1 \times 1)$, sont définies comme suit :

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & & \vdots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{bmatrix}$$

$$\mathbf{C} = [C_1\ C_2\ \dots\ C_N] \qquad \mathbf{D} = d$$

Les équations (2a) et (2b) montrent que la représentation d'état permet, contrairement à la description (1), de décrire l'état interne du filtre et d'offrir une formulation algorithmique utile à l'implémentation du système. A tout instant, les variables d'états $x(n)$ et l'entrée courante $e(n)$ sont utilisées pour calculer la sortie courante $y(n)$ et les variables d'état sont mises à jour. Ces équations facilitent la description du matériel utilisé et son organisation détaillée pour l'implémentation d'un filtre RII 1-D.

Il existe un certain nombre de structures pour l'implémentation des filtres récursifs. La complexité de calcul est une considération dans le choix entre ces différentes réalisations. Pour réduire le nombre de processeurs élémentaires requis par l'algorithme de l'équation (2) et augmenter le débit en données, des matrices tridiagonales pour représenter les filtres récursifs dans l'espace d'état sont retenues comme montré dans [22].

$$\mathbf{A} = \begin{bmatrix} \alpha_0 & \beta_1 & 0 & \dots & 0 & 0 & 0 \\ \gamma_1 & \alpha_1 & \beta_2 & 0 & \dots & 0 & 0 \\ 0 & \gamma_2 & \alpha_2 & \beta_3 & 0 & \dots & 0 \\ 0 & 0 & \gamma_3 & \alpha_3 & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \gamma_4 & \ddots & \beta_{N-2} & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \alpha_{N-2} & \beta_{N-1} \\ 0 & 0 & 0 & \dots & 0 & \gamma_{N-1} & \alpha_{N-1} \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ b \end{bmatrix}$$

$$\mathbf{C} = [0\ 0\ \dots\ c] \qquad \mathbf{D} = d$$

Ce type de représentation est obtenu en minimisant l'erreur quadratique moyenne dans le domaine fréquentiel. Cette procédure exige une spécification de la fonction de transfert désirée aux fréquences discrètes. L'erreur quadratique moyenne à ces fréquences est donnée par :

$$E(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}) = \sum_{i=1}^{M} \left(H(w_i) - H_d(w_i)\right)^2 \tag{3a}$$

La fonction de transfert $H(z)$ du filtre injectée dans le critère d'optimisation est déterminée à partir de la représentation d'état afin d'obtenir une solution directe dans l'espace d'état.

$$H(z) = \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} \tag{3b}$$

Cette procédure d'optimisation peut conduire à des paramètres conduisant à un filtre instable. La solution dépend entièrement des conditions et des algorithmes d'optimisation utilisés.

2.2. filtrage numérique récursif 2-D

La conception des filtres numériques récursifs 2-D est basée sur deux catégories de méthodes distinctes [27]-[29]. La première catégorie utilise la représentation d'état pour estimer une spécification donnée dans le domaine spatial [27] et [28]. La seconde catégorie se sert de la fonction de transfert rationnelle 2-D pour estimer une spécification donnée dans le domaine fréquentiel [29]. L'approximation est effectuée selon les mêmes procédures directes ou optimales que les filtres récursifs 1-D.

2.2.1. *filtres récursifs 2-D tout pôle et représentation d'état*

La fonction de transfert d'un filtre 2-D tout pôle obtenue par l'approximation des spécifications désirées se met sous la forme [29] :

$$H\left(z_1^{-1}, z_2^{-1}\right) = \frac{1}{\sum_{i=0}^{M}\sum_{j=0}^{N} h_{ij} z_1^{-i} z_2^{-j}} \tag{4}$$

Le modèle d'état le plus utilisé pour décrire les filtres 2-D est celui de Roesser [30] :

$$\mathbf{x}^*(i,j) = \mathbf{A}\mathbf{x}(i,j) + \mathbf{B}e(i,j) \tag{5a}$$

$$y(i,j) = \mathbf{C}\mathbf{x}(i,j) + \mathbf{D}e(i,j) \tag{5b}$$

avec

$$\mathbf{x}^*(i,j) = \begin{bmatrix} x_1^h(i+1,j) \\ \vdots \\ x_M^h(i+1,j) \\ \dots\dots\dots \\ x_1^v(i,j+1) \\ x_N^v(i,j+1) \end{bmatrix} \qquad \mathbf{x}(i,j) = \begin{bmatrix} x_1^h(i,j) \\ \vdots \\ x_M^h(i,j) \\ \dots\dots\dots \\ x_1^v(i,j) \\ x_N^v(i,j) \end{bmatrix}$$

où : $\mathbf{x}(i,j) \in R^{M+N}$, $e(i,j) \in R$ et $y(i,j) \in R$; les matrices du filtre récursif 2-D A, B, C et D sont décomposées comme suit :

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \vdots & \mathbf{A}_{12} \\ \dots & \dots & \dots \\ \mathbf{A}_{21} & \vdots & \mathbf{A}_{22} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} \mathbf{B}_1 \\ \dots \\ \mathbf{B}_2 \end{bmatrix} \quad \mathbf{C} = [\mathbf{C}_1 \vdots \mathbf{C}_2] \quad \mathbf{D} = d \tag{6}$$

Varoufakis et al [31] ont établi les formules de calcul des matrices d'état $\mathbf{A}_{11}(M \times M)$, $\mathbf{A}_{12}(M \times N)$, $\mathbf{A}_{21}(N \times M)$, $\mathbf{A}_{22}(N \times N)$, $\mathbf{B}_1(1 \times M)$, $\mathbf{B}_2(1 \times N)$, $\mathbf{C}_1(M \times 1)$, $\mathbf{C}_2(N \times 1)$ et $\mathbf{D}$ à partir de la fonction de transfert du système tout pôle dont la structure de réalisation est donnée en figure 1.

$$\mathbf{A}_{11} = \begin{bmatrix} -h_{10} & 1 & 0 & \dots & 0 \\ -h_{20} & 0 & 0 & \dots & 0 \\ -h_{30} & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ -h_{M0} & 0 & 0 & \dots & 0 \end{bmatrix}$$

$$\mathbf{A}_{12} = \begin{bmatrix} -h_{11} + h_{10}h_{01} & \dots & -h_{1N} + h_{10}h_{0N} \\ \vdots & & \vdots \\ -h_{M1} + h_{M0}h_{01} & \dots & -h_{MN} + h_{M0}h_{0N} \end{bmatrix}$$

$$\mathbf{A}_{21} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

$$\mathbf{A}_{22} = \begin{bmatrix} -h_{01} & -h_{02} & -h_{03} & \dots & -h_{0N} \\ 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & & \vdots \\ 0 & \dots & 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{B}_1 = \begin{bmatrix} -h_{10} \\ -h_{20} \\ \vdots \\ -h_{M0} \end{bmatrix} \quad \mathbf{B}_2 = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \mathbf{C}_1^{\mathrm{T}} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \mathbf{C}_2^{\mathrm{T}} = \begin{bmatrix} -h_{01} \\ -h_{02} \\ \vdots \\ -h_{0N} \end{bmatrix}$$

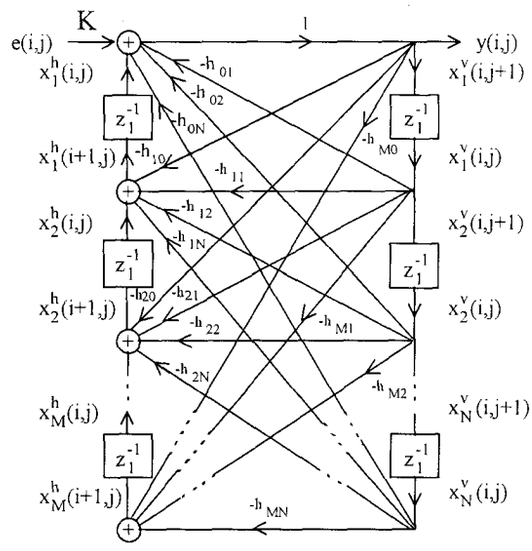


Figure 1. – Réalisation d'une fonction de transfert 2-D tout pôle $H(z_1^{-1}, z_2^{-1})$.

2.2.2. *conception directe des filtres récursifs 2-D stables dans l'espace d'état*

Le filtre récursif 2-D stable sera décrit de nouveau par les équations (5) et (6) d'état et de sortie. L'application de la transformée en z au modèle de Roesser (**A**, **B**, **C**, **d**) avec des conditions initiales permet d'obtenir la fonction de transfert du filtre 2-D :

$$H\left(z_1^{-1}, z_2^{-1}\right) = \mathbf{C}(\mathbf{S} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} = \frac{q\left(z_1^{-1}, z_2^{-1}\right)}{p\left(z_1^{-1}, z_2^{-1}\right)} \quad (7)$$

où

$$\mathrm{S} = z_1\mathbf{I}_M \oplus z_2\mathbf{I}_N = \begin{bmatrix} z_1\mathbf{I}_M & 0 \\ 0 & z_2\mathbf{I}_N \end{bmatrix} \quad (8)$$

et $\oplus$ représente la somme directe de deux matrices. $\mathbf{I}_M$ et $\mathbf{I}_N$ sont deux matrices identités de dimensions $(M \times M)$ et $(N \times N)$ respectivement.

Soit $\{X_d(m \times n)\}$ une spécification donnée (réponse impulsionnelle dans le domaine spatial ou réponse d'amplitude dans le domaine fréquentiel) et soit $\{X(m, n;\ \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})\}$ la réponse correspondant au modèle de Roesser (**A**, **B**, **C**, **D**). Le problème de conception optimale consiste à chercher un ensemble de coefficients des matrices (**A**, **B**, **C**, **D**) dont la réponse $\{X(m, n;\ (\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D})$ estime d'une façon optimale la spécification donnée $\{X_d(m \times n)\}$ selon le critère de l'erreur quadratique moyenne :

$$E(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}) = \sum_{(m,n) \in S} \sum w(m,n) \left| X_d(m,n) - X(m,n;\ \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}) \right|^2 \quad (9)$$

S_x^x est le support d'approximation de la spécification $\{x_d(m \times n)\}$ et $w(m, n)$ est une fonction de pondération. Puisque le modèle de Roesser (**A**, **B**, **C**, **D**) à concevoir doit être stable, on doit imposer une condition de stabilité sur la matrice **A**. Dans le cas où on désire obtenir une forme creuse des matrices (**A**, **B**, **C**, **D**), cette forme doit être imposée comme une contrainte d'optimisation.

2.3. mesure de la vitesse d'une structure de réalisation

Le débit en données et la latence sont deux grandeurs physiques indépendantes permettant de mesurer la vitesse d'exécution d'une structure de réalisation [32], [33].

Le débit en données d'un système est la cadence maximale de réception et de traitement des échantillons du signal d'entrée. L'inverse du débit en données est la période d'échantillonnage T_e qui est le temps minimal exigé entre l'arrivée d'échantillons successifs du signal d'entrée.

La latence T_l d'une sortie d'un système est le retard entre l'arrivée d'un échantillon d'entrée et sa production en sortie. La figure 2 illustre la période d'échantillonnage et la latence.

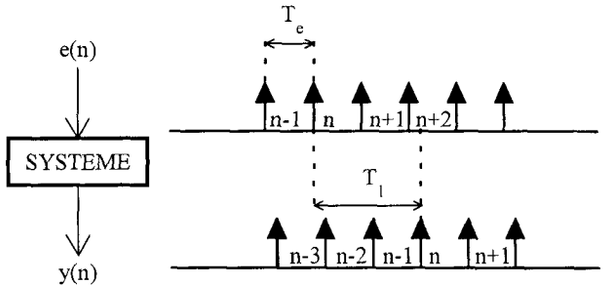


Figure 2. – Définition de la période d'échantillonnage et de la latence.

Pour avoir une interprétation de ces deux grandeurs dans la représentation d'état, nous considérons le cas simple d'un filtre récursif décrit par les équations (2) d'état et de sortie.

Si l'échantillon courant $e(n)$ et les N éléments du vecteur d'état courant $\mathbf{x}(n)$ sont disponibles en même temps, alors l'ensemble est utilisé pour calculer la sortie courante $y(n)$ et le vecteur d'état suivant $\mathbf{x}(n+1)$. La latence est le nombre d'étapes de calcul nécessaire pour générer un échantillon de sortie à partir de l'échantillon d'entrée correspondant. La latence est donc définie par le temps nécessaire au calcul de l'équation de sortie (2b). La période d'échantillonnage est la durée du plus long chemin de calcul mesuré en étapes de calcul entre une entrée courante ou un état courant et un état suivant. La période d'échantillonnage est donc déterminée par le temps d'exécution de l'équation d'état (2a).

Si chaque étape de calcul ou unité de temps comporte une multiplication et une addition, elle prend donc $(m+l)$ cycles d'horloge pour s'exécuter (m et l étant le nombre de cycles d'horloge pour exécuter une multiplication et une addition respectivement).

Les réseaux systoliques dynamiquement commutables développés dans cet article ont une période d'échantillonnage égale à la latence.

Ces notions de vitesse de calcul sont exploitées dans les sections suivantes pour estimer les performances des architectures systoliques utilisées pour implémenter les filtres récursifs.

3. réseau cylindrique dynamiquement commutable

Un réseau cylindrique est constitué de processeurs ou cellules élémentaires placés sur la surface d'un cylindre d'une manière régulière [13]. Les nœuds élémentaires sont formés essentiellement d'additionneurs et de multiplieurs. La figure 3 représente l'architecture d'un filtre RII 1-D du 3ième ordre décrit par les deux équations (2) d'état et de sortie qui peuvent se mettre sous la forme compacte suivante :

$$\begin{bmatrix} \mathbf{x}(n+1) \\ y(n) \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{x}(n) \\ e(n) \end{bmatrix} \tag{10}$$

A chaque instant n, le réseau de la figure 3 calcule la sortie $y(n)$ et les états suivants $x_1(n+1)$, $x_2(n+1)$, $x_3(n+1)$ du filtre. Les éléments des matrices **A**, **B**, **C** et **D** du filtre sont répartis dans les cellules, à raison d'une valeur par cellule selon la figure 3.

Les états courants $x_1(n)$, $x_2(n)$, $x_3(n)$ et l'entrée courante $e(n)$ sont transmis de haut en bas sur les chemins verticaux. Chaque nœud du réseau multiplie l'entrée verticale par le scalaire stocké dans son registre interne. Ce produit est ajouté à l'entrée arrivant le long du chemin transversal puis transmis transversalement. Chaque nœud transmet, sans modification, la séquence verticale à l'exception des nœuds situés en bas du réseau. Chacun de ces derniers nœuds multiplie l'entrée verticale par le scalaire stocké dans son registre interne, puis ajoute ce produit à l'entrée arrivant le long du chemin transversal et enfin transmet le résultat final sur les deux chemins vertical et transversal.

Le principe de fonctionnement des cellules du réseau est indiqué sur la figure 4. On suppose qu'à chaque instant n, ce réseau fonctionne en synchronisme. On peut vérifier intuitivement que les séquences disponibles sur les chemins transversaux en bas du réseau sont composées de la sortie $y(n)$ et des états suivants $x_1(n+1), x_2(n+1), x_3(n+1)$. La figure 5.a illustre la première étape de fonctionnement du réseau cylindrique. La sortie $y(n)$ du filtre est obtenue après $(N+1)$ étapes ($N+1=4$ dans notre exemple). A la fin de la $(N+1)^{\text{ième}}$ étape, on commute le réseau selon la figure 5.b. Les états suivants du filtre $x_1(n+1)$, $x_2(n+1)$, $x_3(n+1)$ sont transmis d'une façon cyclique sur les chemins transversaux des nœuds d'entrée du réseau de même que la prochaine entrée $e(n+1)$.

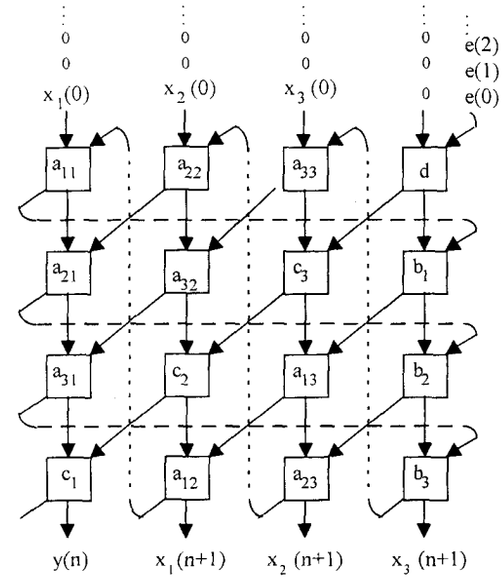


Figure 3. – Réseau cylindrique d'un filtre RII 1-D du 3ième ordre.

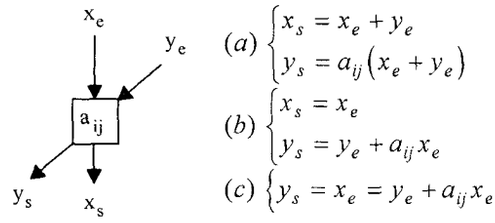
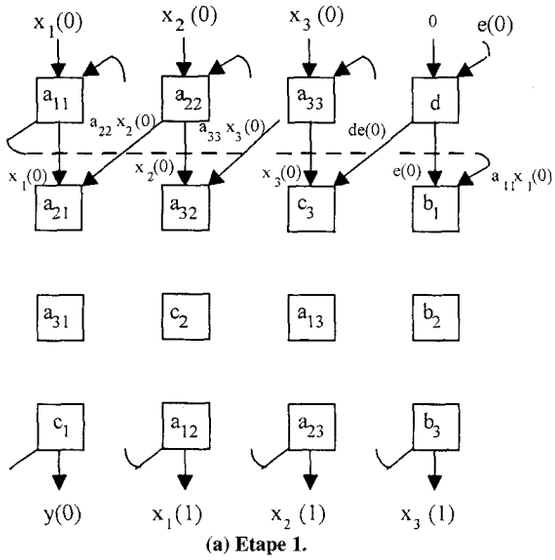
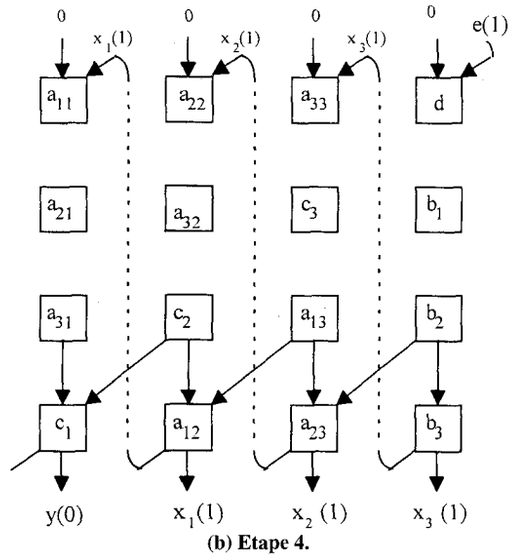


Figure 4. – Fonctionnement d'un processeur élémentaire. (a) cellule d'entrée du réseau, (b) cellule interne, (c) cellule de sortie.



(a) Etape 1.



(b) Etape 4.

$$y_1(0) = c_1x_1(0) + c_2x_2(0) + c_3x_3(0) + de_1(0)$$
$$x_1(0) = a_{12}x_2(0) + a_{13}x_3(0) + a_{11}x_1(0) + b_1e_1(0)$$
$$x_2(0) = a_{23}x_2(0) + a_{21}x_3(0) + a_{22}x_1(0) + b_2e_1(0)$$
$$x_3(0) = a_{31}x_2(0) + a_{32}x_3(0) + a_{33}x_1(0) + b_3e_1(0)$$

Figure 5. – Principe de fonctionnement du Réseau cylindrique de la figure 3.

Un nouveau front d'onde est alors généré à l'instant $(n+1)$ pour calculer la prochaine sortie $y(n+1)$.

Le réseau de la figure 3 ne délivre une sortie qu'après $(N+1)$ étapes de calcul. Ce qui implique une réduction de la vitesse de calcul de $y(n)$ et $x_i(n+1)$, $1 \leq i \leq N$ pour un filtre d'ordre N quelconque et une réduction du débit en données du réseau. Le débit en donnée de ce réseau est donné par : $\dfrac{1}{(N+1)(m+l)}$

Lorsque le filtre récursif possède une matrice tridiagonale, la latence du réseau qui l'implémente peut être réduite à trois étapes de calcul seulement. Malheureusement ce type de matrices est délicat à synthétiser en pratique.

Dans la suite de cet article, la technique de décomposition CTP sera adoptée comme un autre moyen plus général permettant d'améliorer la vitesse d'exécution des architectures cylindriques dynamiquement commutables des filtres récursifs 1-D et 2-D.

4. architectures systoliques rapides dynamiquement reconfigurables des filtres récursifs

4.1. technique de décomposition CTP

Considérons un filtre RII d'ordre $(N-1)$ décrit par l'équation (2). En posant :

$$\mathbf{H} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} \mathbf{x}(n+1) \\ y(n) \end{bmatrix} \qquad \mathbf{u} = \begin{bmatrix} \mathbf{x}(n) \\ e(n) \end{bmatrix}$$

l'équation (10) peut s'écrire sous la forme d'un produit d'une matrice par une colonne :

$$\mathbf{v} = \mathbf{H}\mathbf{u} \tag{11}$$

Nous allons résumer dans ce qui suit la technique de décomposition CTP et l'appliquer à l'algorithme (11) de filtrage RII pour obtenir une forme plus rapide.

Dans une étude publiée en 1989, Porter [16] a montré que chaque algorithme linéaire peut être décomposé en un produit CTP. Le principe d'un produit CTP est illustré par l'exemple suivant. Soit u un vecteur de dimension N avec $N = pq$. Soit $\mathbf{U}$ une matrice $p \times q$ formée par la transformation du vecteur $\mathbf{u}$ en une matrice en utilisant par exemple les segments de $\mathbf{u}$ comme des colonnes de la matrice $\mathbf{U}$.

Si $\mathbf{u} = (u_1, u_2, \ldots, u_q)$ où $u_i \in R^p$ sont des segments du vecteur $\mathbf{u}$ formés de p éléments, la matrice $\mathbf{U}$ associée au vecteur $\mathbf{u}$ est

donc définie par :

$$\mathbf{U} = (U_1, U_2, \ldots, U_q)$$

Par la même procédure, on obtient la matrice **V** à partir du vecteur **v**. On peut maintenant définir la forme rapide de l'algorithme (11) par la relation matricielle :

$$\mathbf{V} = \mathbf{LUR} \tag{12}$$

Il a été établi dans [16] que chaque matrice **H** est constituée d'un terme CTP unique, c'est à dire $\mathbf{H} = \mathbf{L} \cdot \mathbf{R}$, si et seulement si **H** s'écrit sous forme d'un produit tensoriel :

$$\mathbf{H} = \begin{bmatrix} r_{11}\mathbf{L} & \ldots & r_{1p}\mathbf{L} \\ \vdots & & \vdots \\ r_{p1}\mathbf{L} & \ldots & r_{pp}\mathbf{L} \end{bmatrix}$$

où : $\mathbf{R} = \{r_{ij}\}_{i,j=1,\ldots,p}$ et $\mathbf{L} = \{l_{ij}\}_{i,j=1,\ldots,p}$.

Cette décomposition CTP peut être déterminée par la minimisation d'une fonctionnelle :

$$J(\mathbf{L}, \mathbf{R}) = \sum_{ij} ||\mathbf{H}_{ij} - r_{ij}\mathbf{L}||^2 \tag{13}$$

par rapport à **L** et **R** [16]. Dans [15] et [16], la technique CTP est étudiée en détail et des méthodes de minimisation sont établies. Porter a montré que chaque matrice possède une forme CTP. Le développement CTP a été prévu pour un écoulement de données uniforme et des calculs ordonnés. Il existe aussi une relation directe avec les réseaux de processeurs qui peuvent réaliser le produit tensoriel avec une vitesse de calcul élevée. Dans les réseaux conventionnels, le produit d'une matrice par une colonne $\mathbf{v} = \mathbf{Hu}$ nécessite $O(pq)$ unités de temps. En utilisant ces mêmes réseaux, le produit de trois matrices $\mathbf{V} = \mathbf{LUR}$ peut être calculé en $O(p+q)$ unités de temps[16], [17].

La supériorité en vitesse de la dernière opération matricielle sur la première est évidente. Il apparaît clairement que la décomposition CTP est environ $O(\sqrt{N})$ unités de temps plus rapide que la multiplication d'une matrice par un vecteur.

4.2. application de la technique CTP à l'algorithme de filtrage RII 1-D

Considérons l'exemple du filtre RII 1-D du $8^{\text{ième}}$ ordre décrit par l'équation (11) dans l'espace d'état. Avec $N = 9 = 3 \times 3$, $p = q = 3$.

$$\mathbf{H} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{18} & b_1 \\ a_{21} & a_{22} & \ldots & a_{28} & b_2 \\ \vdots & \vdots & & \vdots & \vdots \\ a_{81} & a_{82} & \ldots & a_{88} & b_8 \\ c_1 & c_2 & & c_8 & d \end{bmatrix} \quad \mathbf{v} = \begin{bmatrix} x_1(n+1) \\ x_2(n+1) \\ \vdots \\ x_8(n+1) \\ y(n) \end{bmatrix} \quad \mathbf{u} = \begin{bmatrix} x_1(n) \\ x_2(n) \\ \vdots \\ x_8(n) \\ e(n) \end{bmatrix}$$

Une décomposition CTP à terme unique de **H** est déterminée par les méthodes [15], [16]. Cette décomposition est définie par les matrices (3×3) **L**, **R**, **U**, et **V** suivantes :

$$\mathbf{L} = \begin{bmatrix} l_{11} & l_{12} & l_{13} \\ l_{21} & l_{22} & l_{23} \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \qquad \mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

$$\mathbf{U} = \begin{bmatrix} x_1(n) & x_4(n) & x_7(n) \\ x_2(n) & x_5(n) & x_8(n) \\ x_3(n) & x_6(n) & x_9(n) \end{bmatrix} \tag{14}$$

$$\mathbf{V} = \begin{bmatrix} x_1(n+1) & x_4(n+1) & x_7(n+1) \\ x_2(n+1) & x_5(n+1) & x_8(n+1) \\ x_3(n+1) & x_6(n+1) & x_9(n+1) \end{bmatrix} \tag{15}$$

Le produit CTP associé à l'équation (11) prend la forme de l'équation (12).

Les réseaux cylindriques de la section précédente et des références [13], [14] sont compatibles avec le développement CTP. La figure 6 illustre un réseau cylindrique effectuant le produit **LU** de deux matrices (3×3). Les triangles représentent les mémoires locales de stockage des éléments de la matrice **L** selon les positions indiquées sur la figure 6.a. Les lignes de la matrice **U** sont transmises de haut en bas sur les chemins transversaux. En chaque nœud, l'entrée verticale est multipliée par le scalaire stocké dans son registre interne. Le produit ainsi obtenu est additionné à l'entrée arrivant le long du chemin transversal. Ce résultat est transmis transversalement. La séquence verticale est retransmise sans changement. La figure 6.a montre le déroulement du calcul au début de la seconde étape. La figure 6.b présente le calcul au cours de la seconde étape.

Nous supposons que notre réseau fonctionne en synchronisme. On peut vérifier intuitivement que les séquences disponibles sur les chemins transversaux, en bas du réseau, sont les colonnes du produit matriciel **LU**. Les nœuds du sommet terminent leurs calculs en même temps que le calcul de la première colonne **LU** par les nœuds du bas. Pendant l'étape p (dans notre cas $p = q = 3$), le réseau est commuté selon la figure 6.c. Les colonnes de la matrice **LU** sont donc rebouclées sur les chemins transversaux.

Les lignes de la matrice **R** suivent les colonnes de la matrice **U** sur les chemins verticaux. Le nœud change de fonction lorsque le nouveau calcul commence vers le bas du réseau. En conséquence, le nœud retransmet toutes les séquences d'entrée sans modification tout en calculant itérativement le produit élémentaire de ces séquences. Ce produit est stocké dans la mémoire du nœud selon la figure 6.a. La commutation dans le fonctionnement des nœuds se propage de haut en bas du réseau avec l'arrivée des nouvelles données des matrices **LU** et **R**. La figure 6.d illustre la situation lorsque le front d'onde de calcul atteint la deuxième ligne du réseau. On peut facilement vérifier que toutes les composantes de $\mathbf{V} = \mathbf{LUR}$ sont localisées dans les mémoires des nœuds à la $(p+q+1)^{ième}$ étape de cette séquence. Les indices i, j sur les nœuds de la figure 6.g représentent la position finale des éléments $\mathbf{V}_{ij}$.

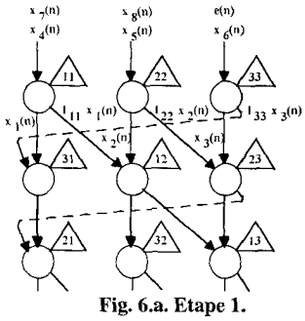


Fig. 6.a. Etape 1.

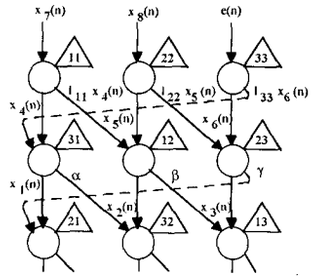


Fig. 6.b. Etape 2.

$\alpha = l_{31}x_1(n) + l_{33}x_3(n)$
$\beta = l_{12}x_2(n) + l_{11}x_1(n)$
$\gamma = l_{23}x_3(n) + l_{22}x_2(n)$

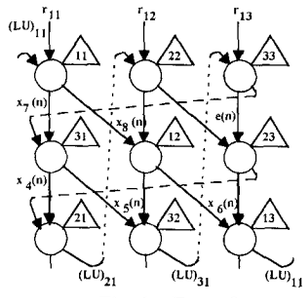


Fig. 6.c. Etape 3.

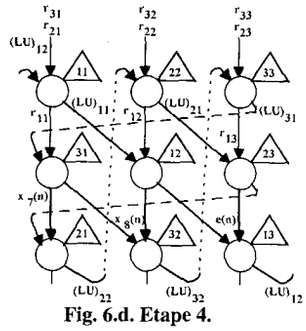


Fig. 6.d. Etape 4.

$(LU)_{11} = l_{13}x_3(n) + l_{12}x_2(n) + l_{11}x_1(n)$
$(LU)_{21} = l_{21}x_1(n) + l_{23}x_3(n) + l_{22}x_2(n)$
$(LU)_{31} = l_{32}x_2(n) + l_{31}x_1(n) + l_{33}x_3(n)$

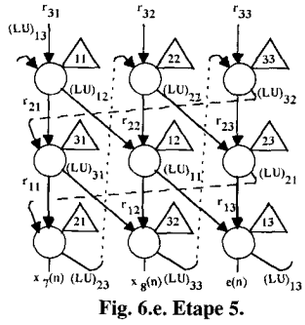


Fig. 6.e. Etape 5.

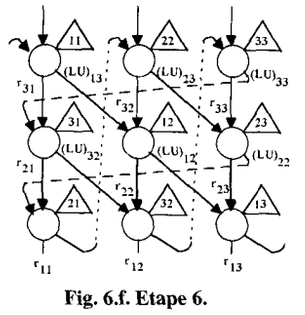
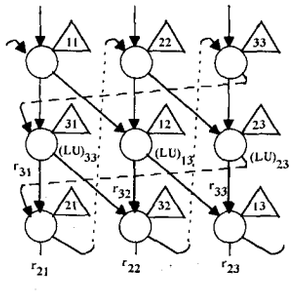


Fig. 6.f. Etape 6.

$V_{11} = (LU)_{11}r_{11} + (LU)_{12}r_{21} + (LU)_{13}r_{31}$
$V_{22} = (LU)_{21}r_{12} + (LU)_{22}r_{22} + (LU)_{23}r_{32}$
$V_{33} = (LU)_{31}r_{13} + (LU)_{32}r_{23} + (LU)_{33}r_{33}$
$V_{31} = (LU)_{31}r_{11} + (LU)_{32}r_{21} + (LU)_{33}r_{31}$
$V_{12} = (LU)_{11}r_{12} + (LU)_{12}r_{22} + (LU)_{13}r_{32}$
$V_{23} = (LU)_{21}r_{13} + (LU)_{22}r_{23} + (LU)_{23}r_{33}$



$V_{31} = (LU)_{31}r_{11} + (LU)_{32}r_{21} + (LU)_{33}r_{31}$
$V_{12} = (LU)_{11}r_{12} + (LU)_{12}r_{22} + (LU)_{13}r_{32}$
$V_{23} = (LU)_{21}r_{13} + (LU)_{22}r_{23} + (LU)_{23}r_{33}$

Fig. 6.g. Etape 7.

Figure 6. – Principe de fonctionnement du réseau cylindrique dynamiquement reconfigurable d'un filtre RII 1-D du 3ième ordre.

Pendant la dernière étape, un système de collection séparé peut être utilisé pour grouper les résultats $\mathbf{V}_{ij}$ à l'extérieur. Les états $x_i(n+1)$, $1 \leq i \leq N$, du filtre obtenus avec l'entrée suivante seront donc utilisés comme les entrées du réseau pour le prochain front d'onde. La sortie courante $y(n)$ est produite à partir de l'entrée courante $e(n)$ après $(p+q+1)$ étapes nécessitant chacune une multiplication et une addition. Le débit en donnée de ce réseau reconfigurable est estimé à $\dfrac{1}{(p+q+1)(m+l)}$ alors qu'il faut un débit en données de $\dfrac{1}{pq(m+l)}$ sur un simple réseau cylindrique de dimension $(N \times N)$ pour la mise en œuvre du même filtre récursif d'ordre $(N-1) = pq$. Un gain important en débit de données est obtenu à l'aide des architectures cylindriques dynamiquement commutables associées à la décomposition CTP de Porter.

Dans le développement précédent, des facultés de calculs flexibles ont été associées aux nœuds de calcul du réseau. Ce fonctionnement flexible des cellules est obtenu à l'aide d'une complexité de mise en œuvre supplémentaire. La capacité de commuter dynamiquement le réseau nécessite alors l'utilisation de bits de contrôle et d'un circuit logique associé au nœud. Toutes ces conséquences matérielles (conception des nœuds, sa logique de contrôle, etc.) doivent donc être évaluées avec soin.

4.3. architectures systoliques dynamiquement reconfigurables pour les filtres RII 2-D

Considérons un filtre récursif 2-D tout pôle d'ordre (M, N) décrit par le modèle de Roesser dans l'espace d'état (5). Ces équations d'état et de sortie sont regroupées en une seule équation matricielle pour arriver à l'algorithme (11) du filtre à implémenter sur les structures systoliques.

Où :

$$\mathbf{H} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \quad \mathbf{v} = \begin{bmatrix} \mathbf{x}^*(i,j) \\ y(i,j) \end{bmatrix} \quad \mathbf{u} = \begin{bmatrix} \mathbf{x}(i,j) \\ e(i,j) \end{bmatrix}$$

L'application de la technique de décomposition CTP permet d'aboutir à l'algorithme rapide (12). Les matrices **V**, **L**, **U** et **R** sont déterminées en utilisant la procédure des filtres récursifs 1-D.

Pour illustrer la méthode, nous considérons un filtre recursif 2-D tout pôle d'ordre $(2,1)$. Ce filtre est décrit par sa fonction de transfert [31] :

$$H\left(z_1^{-1}, z_2^{-1}\right) = \frac{1}{z_1^{-2}z_2^{-1} + 2z_1^{-1}z_2^{-1} + 2z_1^{-1} + 1} \qquad (16)$$

Les matrices d'état sont données par :

$$\mathbf{A} = \left[\begin{array}{cc:c} -h_{10} & 1 & -h_{11} + h_{10}h_{01} \\ -h_{20} & 0 & -h_{21} + h_{20}h_{01} \\ \hdots & \hdots & \hdots \\ 1 & 0 & -h_{01} \end{array}\right] = \begin{bmatrix} -2 & 1 & -2 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} -h_{10} \\ -h_{20} \\ \cdots \\ 1 \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \\ 1 \end{bmatrix} \quad \mathbf{C} = [1\,0 \vdots -h_{10}] = [1\,0\,0] \quad \mathbf{D} = 1$$

les vecteurs d'état sont donnés par :

$$\mathbf{x}^*(i,j) = \begin{bmatrix} x_1^h(i+1,j) \\ x_2^h(i+1,j) \\ x_1^v(i,j+1) \end{bmatrix} \qquad \mathbf{x}(i,j) = \begin{bmatrix} x_1^h(i,j) \\ x_2^h(i,j) \\ x_1^v(i,j) \end{bmatrix}$$

Ainsi l'algorithme de ce filtre peut être représenté par l'équation (11). Avec :

$$\mathbf{H} = \begin{bmatrix} -2 & 1 & -2 & -2 \\ 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{v} = \begin{bmatrix} x_1^h(i+1,j) \\ x_2^h(i+1,j) \\ x_1^v(i,j+1) \\ y(i,j) \end{bmatrix} \quad \mathbf{u} = \begin{bmatrix} x_1^h(i,j) \\ x_2^h(i,j) \\ x_1^v(i,j) \\ e(i,j) \end{bmatrix}$$

Une décomposition CTP à terme unique de **H** est définie par les matrices (2×2) **L**, **R**, **U**, et **V** suivantes :

$$\mathbf{L} = \begin{bmatrix} l_{11} & l_{12} \\ l_{21} & l_{22} \end{bmatrix} \qquad \mathbf{R} = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} x_1^h(i,j) & x_1^v(i,j) \\ x_2^h(i,j) & e(i,j) \end{bmatrix} \qquad \mathbf{V} = \begin{bmatrix} x_1^h(i+1,j) & x_1^v(i,j+1) \\ x_2^h(i+1,j) & y(i,j) \end{bmatrix}$$

Cette décomposition CTP permet d'obtenir l'algorithme rapide (12). Le réseau cylindrique de la figure 7 permet de calculer ce produit CTP selon le même principe que celui de la figure 6. Ce réseau effectue d'abord le produit **LU** des deux matrices de dimension (2×2). Les nœuds du sommet terminent leurs calculs en même temps que le calcul de la première colonne **LU** par les nœuds du bas. Pendant la deuxième étape, le réseau est commuté selon la figure 7.b. Les colonnes de la matrice sont alors rebouclées sur les chemins transversaux. Les lignes de la matrice **R** suivent les colonnes de la matrice **U** sur les chemins verticaux. Le nœud change de fonction au commencement du nouveau calcul. Un produit élémentaire est calculé avec accumulation et stocké sur place dans la mémoire du nœud. La commutation dans le fonctionnement des cellules se propage de haut en bas du réseau au fur et à mesure que les nouvelles données **LU** et **R** arrivent. La figure 7.c montre la troisième étape de calcul de l'algorithme lorsque le front d'onde de calcul atteint la deuxième ligne du réseau. Les éléments V_{ij} de la matrice **V** sont obtenus à la quatrième étape. Ces éléments sont localisés dans les nœuds (i,j) du réseau.

Un système de collection séparé peut être utilisé pour pomper les résultats V_{ij} vers l'extérieur.

Le débit en données est estimé à $1/4(m+l)$.

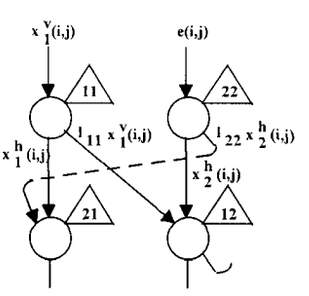


Fig. 7.a. Etape 1.

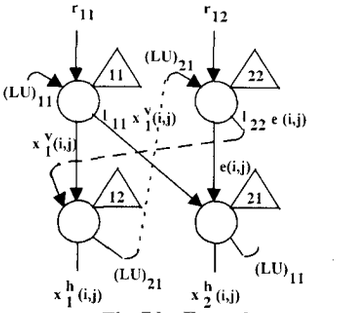


Fig. 7.b . Etape 2.

$$(LU)_{21} = l_{21}x_1^h(i,j) + l_{22}x_2^h(i,j)$$
$$(LU)_{11} = l_{12}x_2^h(i,j) + l_{11}x_1^h(i,j)$$

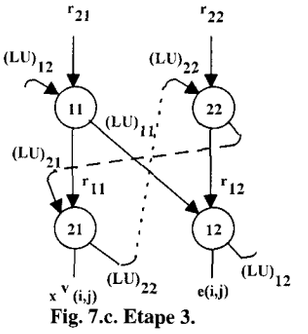


Fig. 7.c. Etape 3.

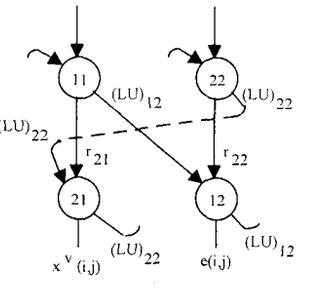


Fig. 7.d. Etape 4.

$$(LU)_{22} = l_{21}x_1^h(i,j) + l_{22}e(i,j)$$
$$(LU)_{12} = l_{12}e(i,j) + l_{11}x_1^h(i,j)$$

$$V_{11} = (LU)_{11}r_{11} + (LU)_{12}r_{21}$$
$$V_{12} = (LU)_{11}r_{12} + (LU)_{12}r_{22}$$
$$V_{22} = (LU)_{21}r_{12} + (LU)_{22}r_{22}$$
$$V_{21} = (LU)_{21}r_{11} + (LU)_{22}r_{21}$$

Figure 7. – Principe de fonctionnement du réseau cylindrique dynamiquement reconfigurable d'un filtre RII 2-D d'ordre (2,1).

5. compactage des calculs dans les réseaux systoliques reconfigurables par l'usage des matrices creuses

L'usage des matrices creuses dans la représentation d'état des filtres récursifs permet une réduction importante des calculs effectués dans les réseaux systoliques reconfigurables. Nous allons réutiliser les matrices creuses pour compacter les calculs effectués par les réseaux reconfigurables. Cette technique introduite par Porter [23] permet une simplification de l'algorithme (12). Pour introduire les idées de cette méthode, nous considérons, sans perte de généralité, la représentation d'état par la matrice

bidiagonale (obtenue à partir de la matrice tridiagonale en posant $\beta_i = 0$, $\alpha_i = a_1$ et $\gamma_i = a_2$).

$$\mathbf{H} = \begin{bmatrix} a_1 & 0 & \dots & \dots & \dots & \dots & \dots & \dots & 0 \\ a_2 & a_1 & 0 & & & & & & \vdots \\ 0 & a_2 & a_1 & 0 & & & & & \vdots \\ \vdots & 0 & \boldsymbol{a_2} & a_1 & 0 & & & & \vdots \\ \vdots & & 0 & a_2 & a_1 & 0 & & & \vdots \\ \vdots & & & 0 & a_2 & a_1 & 0 & & \vdots \\ \vdots & & & & 0 & \boldsymbol{a_2} & a_1 & 0 & \vdots \\ \vdots & & & & & 0 & a_2 & a_1 & 0 \\ 0 & \dots & \dots & \dots & \dots & \dots & 0 & a_2 & a_1 \end{bmatrix} = \{h_{ij}\}_{i,j=1,\dots 9} \tag{17}$$

La matrice **H** vérifie la somme de deux produits tensoriels :

$$\mathbf{H} = \mathbf{L}_1 \circ \mathbf{R}_1 + \mathbf{L}_2 \circ \mathbf{R}_2 \tag{18}$$

où

$$\mathbf{L}_1 = \begin{bmatrix} a_1 & 0 & 0 \\ a_2 & a_1 & 0 \\ 0 & a_2 & a_1 \end{bmatrix} \qquad \mathbf{R}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{L}_2 = \begin{bmatrix} 0 & 0 & a_2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \mathbf{R}_2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

L'équation (18) est une décomposition CTP à deux termes. L'exploitation de la symétrie de **H** et le déplacement des éléments $h_{43} = a_2$ et $h_{76} = a_2$ respectivement aux positions h'_{13} et h'_{46} de la matrice modifiée $\mathbf{H}'$, tout en annulant les éléments h'_{43} et h'_{76}, permettent une décomposition CTP à un terme.

$$\mathbf{H}' = \begin{bmatrix} a_1 & 0 & \boldsymbol{a_2} & 0 & \dots & \dots & \dots & \dots & 0 \\ a_2 & a_1 & 0 & 0 & \ddots & & & & \vdots \\ 0 & a_2 & a_1 & 0 & 0 & \ddots & & & \vdots \\ \vdots & 0 & \mathbf{0} & a_1 & 0 & \boldsymbol{a_2} & \ddots & & \vdots \\ \vdots & & 0 & a_2 & a_1 & 0 & 0 & \ddots & \vdots \\ \vdots & & & 0 & a_2 & a_1 & 0 & 0 & 0 \\ \vdots & & & & 0 & \mathbf{0} & a_1 & 0 & 0 \\ \vdots & & & & & 0 & a_2 & a_1 & 0 \\ 0 & \dots & \dots & \dots & \dots & \dots & 0 & a_2 & a_1 \end{bmatrix} = \{h'_{ij}\}_{i,j=1,\dots 9} \tag{19}$$

La matrice **H'** vérifie le produit tensoriel :

$$\mathbf{H} = \begin{bmatrix} 1 \times \mathbf{L} & 0 \times \mathbf{L} & 0 \times \mathbf{L} \\ 0 \times \mathbf{L} & 1 \times \mathbf{L} & 0 \times \mathbf{L} \\ 0 \times \mathbf{L} & 0 \times \mathbf{L} & 1 \times \mathbf{L} \end{bmatrix} = \mathbf{L} \circ \mathbf{R} \tag{20}$$

où

$$\mathbf{H} = \begin{bmatrix} a_1 & 0 & \phi \\ a_2 & a_1 & 0 \\ 0 & a_2 & a_1 \end{bmatrix} \qquad \mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

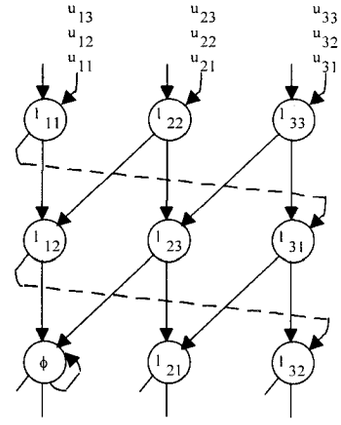


Figure 8. – Implémentation d'un filtre récursif représenté par la matrice creuse du type (17) en utilisant le compactage des calculs.

$\phi = a_2$ lorsque la matrice $\mathbf{H}'$ agit sur la première et la deuxième colonne la matrice **U** (équation (14)) (pour alléger la notation, on pose $\mathbf{U} = \{u_{ij}\}_{ij=1,2,3}$); et $\phi = 0$ lorsqu'elle agit sur sa troisième colonne. Ainsi, les calculs intervenant dans l'algorithme rapide (12) sont compactés.

$$\mathbf{V} = \mathbf{LUR} = \mathbf{LU} \tag{21}$$

Dans cet exemple, le terme $0 \times u_{31}$ faisant parti des calculs initiaux est remplacé par $a_2 \times u_{31}$ qui provient de $\mathbf{L}_2 \circ \mathbf{R}_2$. Le produit $a_2 \times u_{31}$ est alors dirigé vers la somme partielle adéquate pour reproduire l'intégralité des calculs. La même démarche est utilisée pour le produit élémentaire $a_2 \times u_{32}$.

Dans la figure 8, les éléments de la matrice **L** sont enregistrés dans les registres des nœuds. La fonction de compactage des calculs est entièrement affectée au nœud l_{13}. Chaque produit scalaire est retardé d'une unité de temps avant de l'additionner à la somme partielle. L'architecture cylindrique résultante du filtre récursif 1-D d'ordre $(N-1) = 8$ à matrice d'état bidiagonale se réduit au calcul du produit de deux matrices de dimension $(p \times p)$ (où $N = p \times p$). Le débit en données de la structure est estimé à $1/(2p-1)(m+l)$

6. discussion et comparaison de l'architecture proposée avec celles de Parhi et Woods

Les structures systoliques des références proposées dans [34] et [21] implémentent sur silicium des filtres RII du 4ième ordre et du premier ordre respectivement. Ils utilisent les techniques

systoliques à bit parallèle. Parhi et Hatamian [34] utilisent une technique récursive qui augmente le débit en données au détriment de la complexité matérielle. Par contre Woods et McCanny [21] évitent la récursivité et permettent de réduire la complexité de mise en œuvre tout en diminuant le débit en données.

Les différences dans l'ordre du filtre, la technologie utilisée et la méthode de conception ne permettent pas de faire une comparaison quantitative entre les architectures systoliques citées en références et les architectures présentées dans ce travail. Néanmoins, une comparaison qualitative de la complexité de réalisation montre que les architectures proposées demandent moins de cellules de calcul élémentaire, pour un même filtre, que les structures de Parhi et de Woods. Une comparaison du point de vue débit en données ne peut se faire que si une implantation à l'aide d'une technologie CMOS est envisageable.

7. conclusion

La représentation d'état des filtres RII 1-D et 2-D permet de les mettre en œuvre directement sur des réseaux systoliques de type cylindrique dynamiquement commutables. Cependant, cette conception directe réduit la vitesse de calcul et le débit en données du réseau. Une amélioration du débit en données de ces structures est réalisée en représentant les filtres par des matrices creuses qui permettent de réduire la complexité matérielle. En utilisant la technique de décomposition CTP, nous avons présenté une méthode générale d'augmentation de la vitesse de calcul des architectures cylindriques dynamiquement reconfigurables des filtres récursifs. Cependant, le gain en débit de données de ces structures est atteint au détriment d'une complexité de mise œuvre supplémentaire des cellules de calcul qui doit être considérée avec soin. L'application de la technique de décomposition CTP à des filtres RII, représentés par des matrices du type tridiagonale dans l'espace d'état, permet une amélioration significative de la complexité matérielle.

Remerciements

Les auteurs remercient les experts pour leurs remarques constructives.

BIBLIOGRAPHIE

[1] H. T. Kung, «Why systolic architectures?», Computer, Vol. 15, 1982, pp. 37-46.

[2] L. Guo-Jie, B. Wash, «The design of optimal systolic arrays», *IEEE Trans. Comput.*, Vol. C-34, N°1, Jan.1985, pp. 66-77.

[3] P. R. cappello, K. steiglitz,«Unifying VLSI array designs with geometric transformations», *in Proc. Int. Conf. Parallel Processing*, 1983, pp. 448-457.

[4] S. Y. Kung, H. J. Whitehouse, T. Kailath, «VLSI and modern signal processing», Englewood Cliffs, N. J., Prentice-Hall, Inc. 1985.

[5] J.H. Cheng, O. H. Ibarra, T. C. Pong, S. M. Sohn, «Two- dimensional convolution on a pyramid computer», in *Proc. Int. Conf. Parallel Processing*, 1987, pp. 780-782.

[6] J. V. Mc Canny, J. G. Mc Whirter, «Some systolic array developments in the UK», *IEEE Comp.*, Vol. 20, N° 7, 1987, pp. 53-65.

[7] G. I. Kechriotics, M. Au, M. Bletsas, R. Tolimieri, E. S. Manolakos, «A new approach for computing multidimensional DFT's on parallel machines and its implementation on the iPSC/860 hypercube», *IEEE Trans. on Signal Processing*, Vol. 43, N° 1, Jan. 1995.

[8] I Gertner, M. Shamash, «VLSI architectures for multidimensional Fourier transform processing», *IEEE Trans. on Computers*, Vol. C-36, N°11, Nov. 1987.

[9] R. K. Montoye, D. H. Lawrie, «A practical algorithm for the solution of triangular systems in a parallel processing systems», *IEEE Trans. Comput.*, Vol. C-31, N°1, Nov.1982.

[10] J. G. Chung, K. K. Parhi, «Pipelining of lattice IIR digital filters«, *IEEE Trans. on Signal Processing*, Vol. 42, N°4, April 1994, pp. 751-761.

[11] L. E. Lucke, K. K. Parhi, «Parallel processing architecture for rank order and stack filters», *IEEE Trans. on Signal Processing*, Vol. 42, N°5, May 1994, pp. 1178- 1188.

[12] P. Comon, Y. Robert, «A systolic array for computing BA-1», *IEEE Trans. on Acoustics, Speech, and Signal Processing*, Vol. ASSP 35, N°6, June.1987.

[13] W. A. Porter, J. L. Aravena, «Cylindrical arrays for matrix multiplications», *in Proc. 24th Allerton Conf.*, Oct. 1986.

[14] W. A. Porter, and J. L. Aravena,«Orbital architectures with dynamic reconfiguration», *Proc.IEE*, part E, Vol. 134, N°6, Nov.1987, pp. 281-287.

[15] W. A. Porter, J. L. Aravena, «Array based design of digital filters», IEEE Trans. on Acoustics, Speech, and Signal Processing, Vol. 38, N°9, Sep.1990, pp. 1628-1632.

[16] W. A. Porter, «Concurrent forms of signal processing algorithms», *IEEE Trans. on Circuits and Systems*, Vol. 36,N°4, Apr.1989, pp. 553-560.

[17] W. A. Porter, «Fast forms of banded maps», *IEEE Trans. on Acoustics, Speech*, and Signal Processing, Vol. 38, N°7, July.1990, pp. 1192-1197.

[18] P. R. Cappello, «Towards an FIR filter tissue», *Proc. IEEE Int Conf. on Acoustics*, Speech, and Signal Processing, Tampa, Florida, USA, Mar.1985, pp. 276-279.

[19] M. M. Fahmy, Y. Wan, «New array processor architectures for two-dimensional FIR digital filters», *IEE Proc.* , Vol. 136, part E, N°4, July 1989, pp. 234-238.

[20] K K. Parhi, D G. Messerschmitt, «Block digital filtering via incremental block-state structure», *Proc. IEEE Int. Symp. on Circuits and Systems*, Philadelphia, PA, USA, May 1987, pp. 645-648.

[21] R. F. Woods, J. V. McCanny,«Design of a high-performance IIR digital filter chip», *IEE Proc.*, part E, Vol. 139, N°3, May 1992, pp. 195-202.

[22] D. Chikouche, R. E. Bekka, A. Khellaf, A. Boucenna, «Orbital architectures for recursive digital filters», *Proc. Maghrebean Conf. on Soft. Eng. and Art. Intel.*, Algiers, Apr.1996, pp. 273-277.

[23] W. A. Porter, «Fast forms of banded maps», *IEEE Trans. on Acoustics*, Speech, and Signal Processing, Vol. 38, N°7, Jul.1990.

[24] K. Ogata, « », Englewood Cliffs, N. J., Prentice-Hall, Inc. 1987.

[25] D. F. Delchamps, «State-space and imput-ouput linear systems», Springer-Verlag, N. Y. 1988.

[26] F. J. Taylor, «Digital filter design handbook», Marcel dekker, Inc., N. Y. 1983.

[27] T. Lin, M. Kawamata, T. Higuchi, «Design of 2-D digital filters with an arbitrary response and no overflow oscillations based on a new stability

condition», *IEEE Trans. on Circuits and Systems*, Vol. CAS-34, N°2, Feb. 1987, pp. 113-126.

[28] S. A. H. Aly, M. Fahmy, «Spatial-domain design of two-dimensional recursive digital filters», *IEEE Trans. on Circuits and Systems*, Vol. CAS-27, Oct. 1980, pp. 892- 900.

[29] J. W. Woods, J. H. Lee, I. Paul, «Two-dimensional IIR filter design with magnitude and phase error criteria», *IEEE Trans. on Acoustics*, Speech, and Signal Processing, Vol. ASSP-31, Aug. 1983, pp. 886-894.

[30] R. P. Roesser, «A discrete state space model for linear image processing», *IEEE Trans. Automat. Contr.*, Vol. AC-20, Feb. 1975, pp. 1-10.

[31] S. J. Varoufakis, P. N. Paraskevopoulos, G. E. Antoniou, «On the minimal state-space realizations of all- pole and all-zero 2-D systems», *IEEE Trans. on Circuits and Systems*, Vol. CAS-34, N° 3, Mar. 1987, pp. 289- 292.

[32] M. B. Srivastava, M. Potkonjak, «Optimum and heuristic transformation techniques for simultaneous optimisation of latency and throughput», *IEEE Trans. VLSI Systems*, Vol. 3, N° 1, March 1995, pp. 2-19.

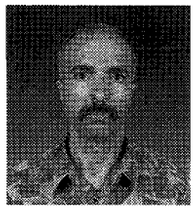
[33] P. R. Gelabert, T. P. Barnwell, «Optimal periodicmultiprocessor scheduler for fully specified flow», *IEEE Trans. Signal Process.*, Vol. 41, 1993, pp. 858-888.

[34] K K. Parhi, M. Hatamian, «A high sample rate recursive digital filter chip», in «VLSI Signal Processing III«, *IEEE Press*, N. Y., USA, 1988, pp. 3-14.

Manuscrit reçu le 3 octobre 1997.

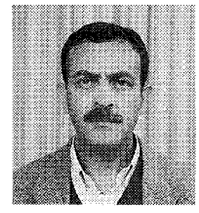
LES AUTEURS

Djamel CHIKOUCHE



Ingénieur diplômé de l'Université de Constantine (1981), Master of Science à Ohio State University de Columbus, U.S.A. (1984). Il prépare actuellement une thèse de doctorat d'Etat à l'Université de Sétif et occupe un poste d'enseignant. Domaine de recherche : traitement du signal et systèmes de communication.

Raïs El'hadi BEKKA



Ingénieur diplômé de L'Ecole National Polytechnique d'Alger (1980), Magister et Docteur d'Etat (1987 et 1994 , Université de Sétif). Maître de Conférences à l'Institut d'Electronique-Université de Sétif. Domaine de recherche : traitement du signal et instrumentation médicale.