

# ***Implémentation sur FPGA d'un turbo codeur-décodeur en blocs à haut débit avec une faible complexité.***

Thomas Quang Khoi TA

Équipe ETSN, Supélec, campus de Rennes

Mitsubishi Electric ITE-TCL, Rennes

08 décembre 2003

1

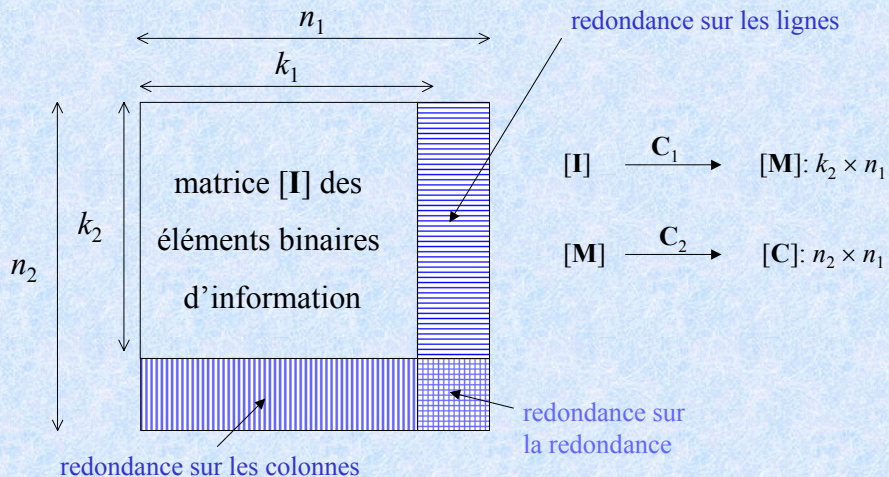
## **Plan de la présentation**

- 1- Codes produits,
- 2- Décodage itératif des codes produits : turbo codes en blocs
  - Décodage à entrée pondérée : décodage de Chase,
  - Décodage à sortie pondérée : calcul de l'information extrinsèque.
- 3- Implémentation du turbo décodeur des codes produits :
  - Étude et optimisation de divers paramètres,
  - Architectures du décodeur,
  - Architecture des mémoires.
- 4- Optimisation du turbo décodeur,
- 5- Conclusions et perspectives.

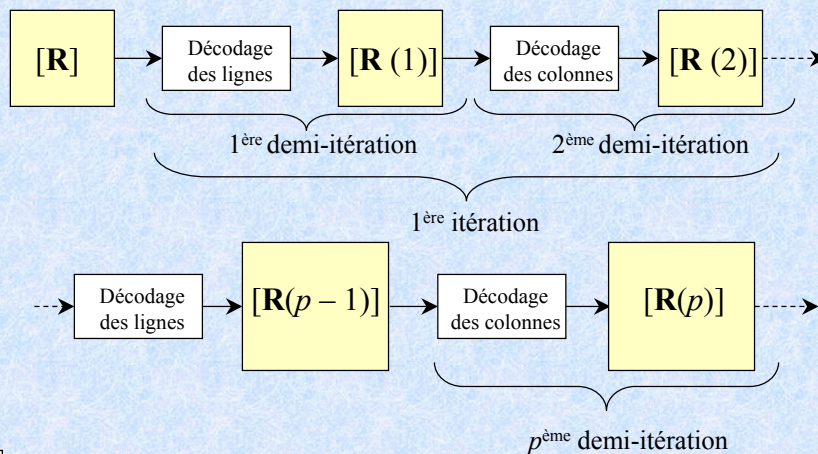
08 décembre 2003

2

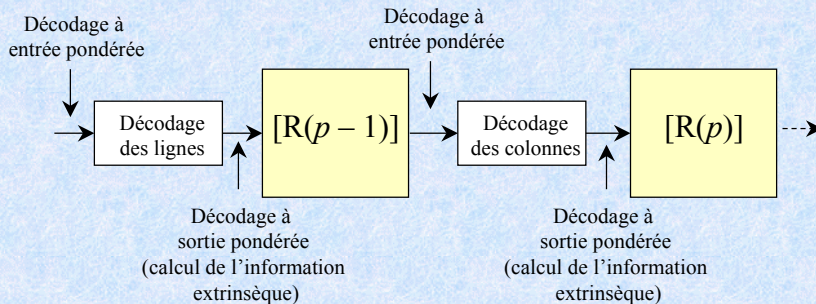
- Introduits en 1954 par P. Elias,
- Puissants codes correcteurs d'erreurs construits à partir de plusieurs codes en bloc linéaires de faibles pouvoirs de correction,
  - Deux codes en bloc linéaires :  
 $C_1(n_1, k_1, d_1)$  et  $C_2(n_2, k_2, d_2)$ ,
  - Code produit  $C(n, k, d) = C_1 \otimes C_2$ ,  
 $n = n_1 \times n_2$ ,  
 $k = k_1 \times k_2$ ,  
 $d = d_1 \times d_2$ ,
- Construction relativement simple.



- 1- Codes produits,
- 2- **Décodage itératif des codes produits : turbo codes en blocs,**
  - Décodage à entrée pondérée : décodage de Chase,
  - Décodage à sortie pondérée : calcul de l'information extrinsèque.
- 3- Implémentation du turbo décodeur des codes produits :
  - Étude et optimisation de divers paramètres,
  - Architectures du décodeur,
  - Architecture des mémoires.
- 4- Optimisation du turbo décodeur,
- 5- Conclusions et perspectives.



- Décodage à entrée pondérée exploite mieux les informations reçues (par rapport au décodage binaire),  
 bonnes performances.



- Le décodage itératif à entrée et à sortie pondérées permet d'avoir de meilleures performances grâce l'information extrinsèque calculée à chaque demi-itération.

08 décembre 2003

7

Algorithme Chase-Pyndiah ou turbo code en blocs :

- Inventé en 1994 par R. Pyndiah à l'ENST Bretagne,
- Performances proches de la limite théorique de Shannon,
- Décodage à entrée pondérée : décodage de Chase,
- Décodage à sortie pondérée : algorithme de Pyndiah  
 Calculer l'information extrinsèque.

08 décembre 2003

8

- 1- Codes produits,
- 2- Décodage itératif des codes produits : turbo codes en blocs,
  - **Décodage à entrée pondérée : décodage de Chase,**
  - Décodage à sortie pondérée : calcul de l'information extrinsèque.
- 3- Implémentation du turbo décodeur des codes produits :
  - Étude et optimisation de divers paramètres,
  - Architectures du décodeur,
  - Architecture des mémoires.
- 4- Optimisation du turbo décodeur,
- 5- Conclusions et perspectives.

- Décodage est optimal s'il cherche le mot de code selon le critère du maximum de vraisemblance *a posteriori* :

- ⇒ le mot de code à distance euclidienne minimale,
- ⇒  $2^k$  comparaisons de distances euclidiennes,

**➔** Complexité élevée pour  $k > 10$ .

*k* : dimension du code.

- Décodage de Chase :

- ⇒ le mot de code à distance euclidienne minimale,
- ⇒  $C_n^{\lfloor d/2 \rfloor}$  séquences de test maximum.

**➔** Complexité raisonnable pour  $d < 10$ .

*d* : distance minimale du code.

Algorithme de Chase se limite à une boule  $\mathbf{B}(\mathbf{Y}_0, d-1)$  :

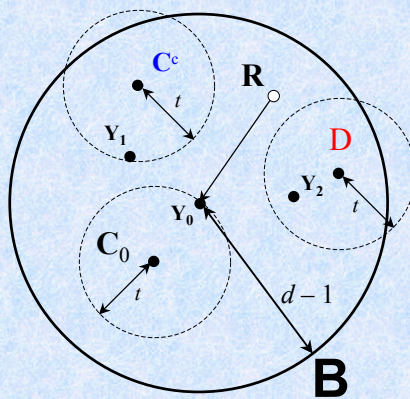
$$P \geq \Pr [\mathbf{X} \in \mathbf{B}(\mathbf{Y}_0, d-1)] = \sum_{i=0}^{d-1} C_n^i p^i (1-p)^{n-i}$$

$\mathbf{X}$  : mot de code émis,

$\mathbf{Y}_0$  : vecteur binaire obtenu par seuillage du mot reçu,

$p$  : probabilité d'erreur binaire sur le canal de transmission.

Lorsque  $p < 10^{-2} \Rightarrow P$  est assez grande.



- À partir de  $\mathbf{R}$ , on génère le mot binaire  $\mathbf{Y}_0$ ,
- Les séquences de test  $\mathbf{Y}_i$  sont obtenues par inversion d'un ou plusieurs bits de  $\mathbf{Y}_0$  dont les positions correspondent aux symboles les moins fiables de  $\mathbf{R}$
- Décodage algébrique de  $\mathbf{Y}_0$  et des séquences de test  $\mathbf{Y}_i$  donne les mots de code  $\mathbf{C}_i$ .
- Parmi  $\mathbf{C}_0$  et les  $\mathbf{C}_i$ , on sélectionne deux mots de code à distance euclidienne minimale de  $\mathbf{R}$  :  
le mot décidé  $\mathbf{D}$  et le concurrent  $\mathbf{C}^c$ .

- 1- Codes produits,
- 2- Décodage itératif des codes produits : turbo codes en blocs,
  - Décodage à entrée pondérée : décodage de Chase,
  - **Décodage à sortie pondérée : calcul de l'information extrinsèque.**
- 3- Implémentation du turbo décodeur des codes produits :
  - Étude et optimisation de divers paramètres,
  - Architectures du décodeur,
  - Architecture des mémoires.
- 4- Optimisation du turbo décodeur,
- 5- Conclusions et perspectives.

- Calcul de l'information extrinsèque  $w_j$  :

- Si le concurrent est trouvé :

$$w_j = (M^c - M^d)d_j - r_j$$

- Sinon :

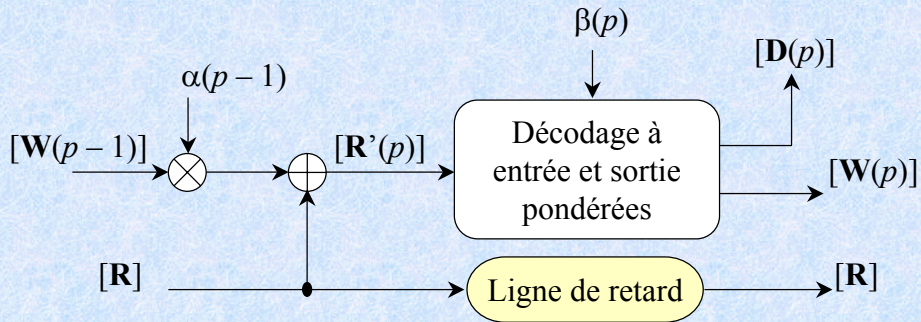
$$w_j = \beta d_j - r_j$$

- Calcul de fiabilité  $r'_j$  de  $d_j$  :

$$r'_j = w_j + r_j$$

$M^d$  : métrique du mot  $\mathbf{D}$ ,  
 $M^c$  : métrique du concurrent,  
 $d_j$  :  $j^{\text{ème}}$  élément binaire de décision  
 $\beta$  : constante positive,

À la  $p^{\text{ème}}$  demi-itération



$[R']$  : matrice des données pondérées

$\beta$  : constante positive

$[D]$  : matrice des mots décidés

$\alpha$  : coefficient de confiance

$[W]$  : matrice des informations extrinsèques

08 décembre 2003

15

1- Codes produits,

2- Décodage itératif des codes produits : turbo codes en blocs,

- Décodage à entrée pondérée : décodage de Chase,
- Décodage à sortie pondérée : calcul de l'information extrinsèque.

**3- Implémentation du turbo décodeur des codes produits :**

- Étude et optimisation de divers paramètres,
- Architectures du décodeur,
- Architecture des mémoires.

4- Optimisation du turbo décodeur,

5- Conclusions et perspectives.

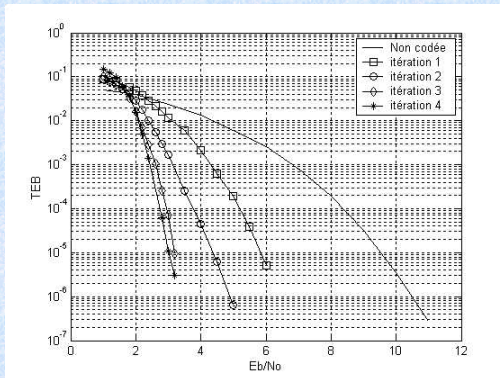
08 décembre 2003

16

Objectif : atteindre le haut débit ( $> 25$  Mbit/s) tout en ayant :

- bonnes performances en terme de TEB,
- faible complexité.

- 1- Codes produits,
- 2- Décodage itératif des codes produits : turbo codes en blocs,
  - Décodage à entrée pondérée : décodage de Chase,
  - Décodage à sortie pondérée : calcul de l'information extrinsèque.
- 3- Implémentation du turbo décodeur des codes produits :
  - **Étude et optimisation de divers paramètres,**
  - Architectures du décodeur,
  - Architecture des mémoires.
- 4- Optimisation du turbo décodeur,
- 5- Conclusions et perspectives.



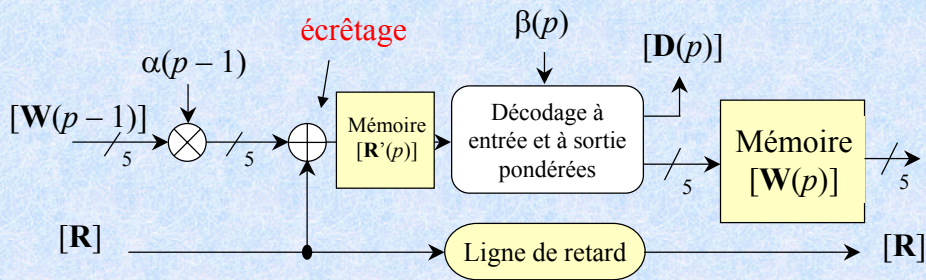
- Plus le nombre d'itérations est élevé :
  - meilleures sont les performances,
  - plus la complexité est grande.
- Lorsque  $NbI \geq 5$ , gain de codage est peu important.



4 itérations de décodage est donc un bon compromis performances/complexité.

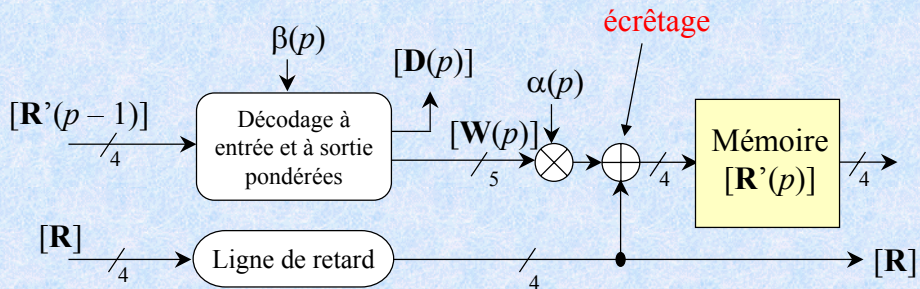
Bon compromis entre les performances (en terme TEB) et la complexité du turbo décodeur est obtenu pour :

- 4 itérations de décodage,
- 4 bits de quantification,
- 3 composantes les moins fiables,
- 8 séquences de test,
- 1 seul concurrent.



Décodage itératif nécessite de sauvegarder  $[W(p)]$  :

➔ Taille de mémoire de sauvegarde  $(n \times n \times 5)$  bits

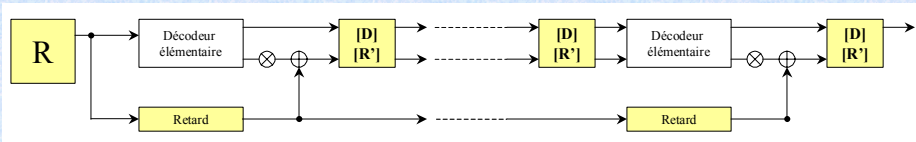


- Principe de décodage reste le même,
- Taille de mémoire est moins importante :  $(n \times n \times 4)$  bits,
- L'écrtage est retardé.

- 1- Codes produits,
- 2- Décodage itératif des codes produits : turbo codes en blocs,
  - Décodage à entrée pondérée : décodage de Chase,
  - Décodage à sortie pondérée : calcul de l'information extrinsèque.
- 3- Implémentation du turbo décodeur des codes produits :
  - Étude et optimisation de divers paramètres,
  - **Architectures du décodeur,**
  - Architecture des mémoires.
- 4- Optimisation du turbo décodeur,
- 5- Conclusions et perspectives.

Plusieurs structures peuvent être utilisées pour implémenter le turbo décodeur :

- Structure pipeline,
- Structure itérative :
  - Structure itérative à traitement par symbole,
  - Structure itérative à traitement par bloc.



- Adaptée au traitement à haut débit et souple à implémenter,
- Encombrement dépendant du nombre d'itérations,
- Latence dépendante du nombre d'itérations :  

$$2 \times NbI \times (n^2 + 2n) \text{ symboles}^*$$

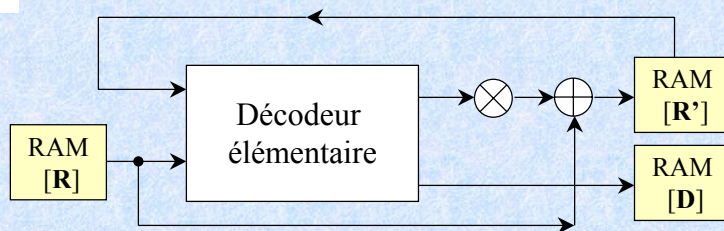
Latence : la durée entre la réception et l'émission d'un symbole,

$NbI$  : nombre d'itérations de décodage utilisé,

\* On suppose qu'un symbole est traité pendant une période d'horloge.

08 décembre 2003

25



➤ Un seul module de décodage pour plusieurs itérations

- ➔ L'encombrement indépendant du nombre d'itération,
- ➔ L'encombrement réduit (d'un facteur de 4 pour 4 itérations de décodage par rapport à la structure pipeline),
- ➔ La latence limitée à  $2n^2$  symboles quelque soit le nombre d'itérations.

08 décembre 2003

26

- Faible encombrement et simple à implémenter,
- Temps de traitement d'un symbole :

$$t_s = 2 \times NbI \times t_{rs}$$

➔ débit des données divisé par un facteur égal au nombre de demi-itérations utilisé.

$t_s$  : temps de calcul d'un symbole,  
 $t_{rs}$  : temps de remplissage d'un symbole,  
 $NbI$  : nombre d'itérations de décodage.

- Temps de traitement d'un vecteur :

$$t_v = \frac{n}{2 \times NbI} \times t_{rs}$$

$n$  : longueur  
du vecteur.

➔ Débit plus élevé car  $2 \times NbI < n$ .

- Encombrement plus élevé, mais reste indépendant du nombre d'itérations.
- Requiert une mémoire spécifique : capables d'être lue et écrite **par bloc** aussi bien en ligne qu'en colonne **en une seule période d'horloge**.

	Structure pipeline	Structure itérative par symbole	Structure itérative par bloc
Complexité	++	-	+
Débit	$d = \frac{1}{t_{rs}}$	$\frac{1}{2 \times NbI} \times d$	$\frac{n}{2 \times NbI} \times d$
Latence	$2NbI(n^2 + 2n)$	$2n^2$	$2n^2$
Mémoire	Simple	Simple	Complexe

Latence : la durée entre la réception et l'émission d'un symbole.



Structure itérative à traitement par bloc offre un bon compromis performances/complexité.

- Codes produits BCH(32,26,4)<sup>2</sup>  $\Rightarrow n = 32$ ,
- 4 itérations de décodage (8 demi-itérations).

	Structure pipeline	Structure itérative par symbole	Structure itérative par bloc
Débit	$d = 8$ Mbit/s	1 Mbit/s	32 Mbit/s
Complexité	12.000 EL (144.000 portes)	1.500 EL (18000 portes)	$12.000 > ? > 1.500$
Latence	1 ms	0,25 ms	0,25 ms

EL : éléments logiques (unité d'encombrement d'ALTERA),

Latence : la durée entre la réception et l'émission d'un symbole.

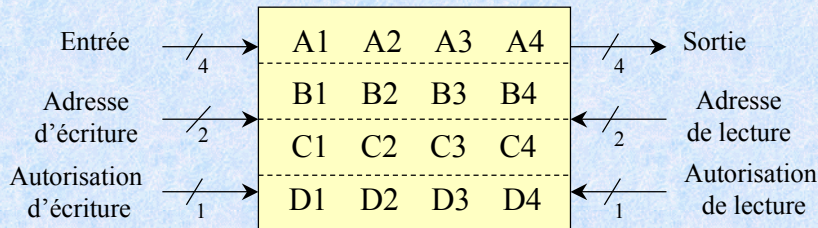
- 1- Codes produits,
- 2- Décodage itératif des codes produits : turbo codes en blocs,
  - Décodage à entrée pondérée : décodage de Chase,
  - Décodage à sortie pondérée : calcul de l'information extrinsèque.
- 3- Implémentation du turbo décodeur des codes produits :
  - Étude et optimisation de divers paramètres,
  - Architectures du décodeur,
  - **Architecture des mémoires.**
- 4- Optimisation du turbo décodeur,
- 5- Conclusions et perspectives.

Créer des mémoires à l'aide des RAM :

- Mise en œuvre simplifiée,
- Économique en consommation électrique,
- Possibilité de lire et d'écrire les données symbole par symbole ou vecteur par vecteur (en ligne ou en colonne),
- Possibilité de lire et d'écrire des données de différentes adresses en même temps (RAM double port),
- Possibilité de configuration en  $2048 \times 1$  bits ou  $1024 \times 2$  bits ... ou  $128 \times 16$  bits (ALTERA APEX 20K).

### Exemple de mémoire de capacité 4×4 bits

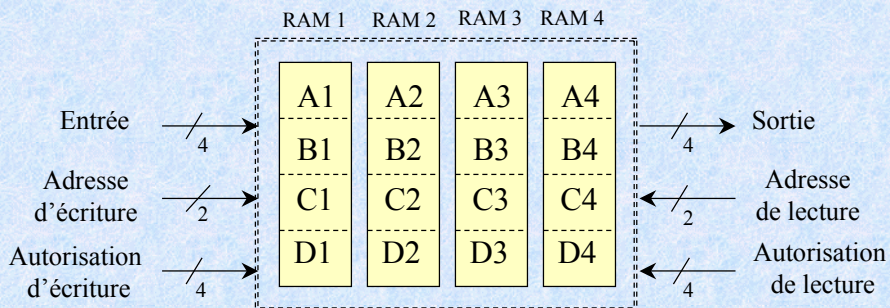
Conception classique à partir d'une RAM double port configurée en 4 mots de 4 bits.



➔ Écriture et lecture d'un vecteur en ligne en une période d'horloge,

➔ Écriture et lecture d'un vecteur en colonne en 4 périodes d'horloge.

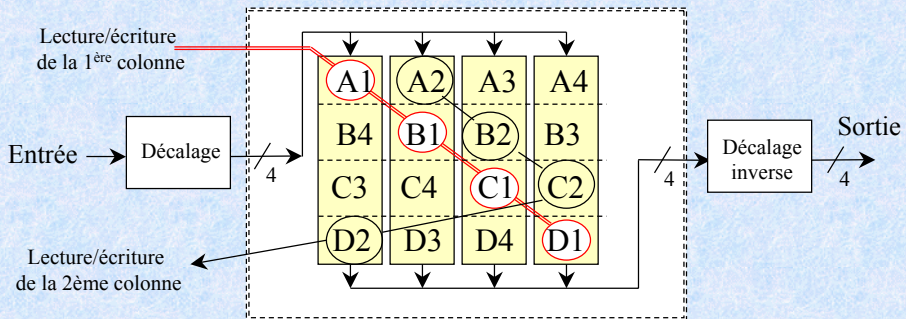
Conception à partir de 4 RAM double port configurée en 4 mots de 1 bit.



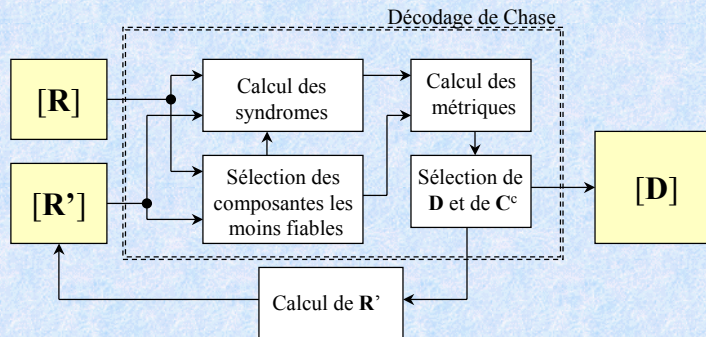
➔ Écriture et lecture d'un vecteur en ligne en une période d'horloge,

➔ Écriture et lecture d'un vecteur en colonne en 1 période d'horloge est impossible car les données en colonne se trouvent dans les mêmes plans mémoire.

Conception à partir de 4 RAM double port configurée en 4 mots de 1 bit.

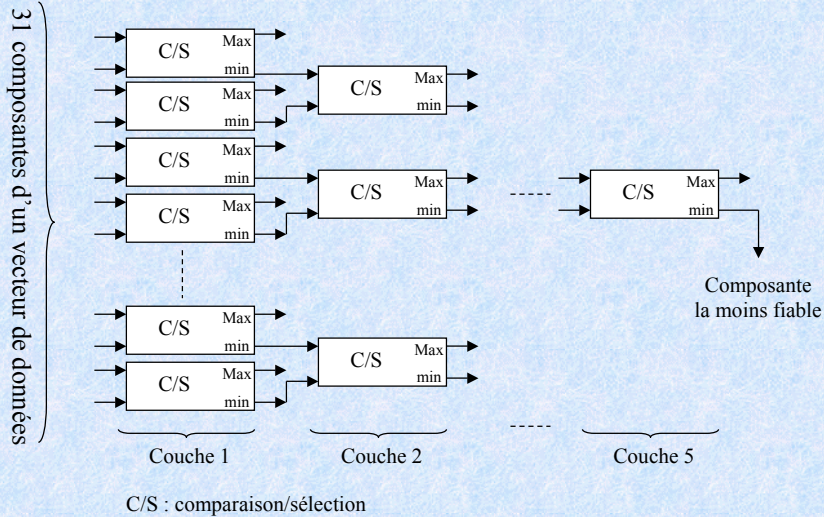


⇒ Les symboles d'un même vecteur (ligne ou colonne) doivent être mis sur des différents plans mémoire.



Complexité	Débit
9576 EL (≈115.000 portes)	12 Mbit/s

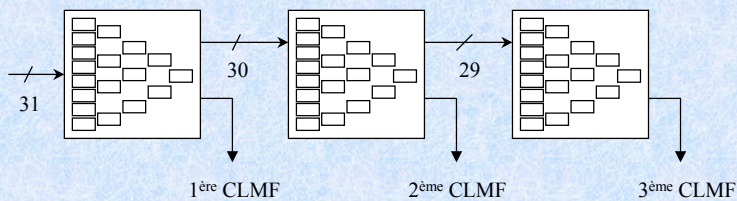
## Sélection d'une composante la moins fiable



08 décembre 2003

37

## Sélection de 3 composantes les moins fiables



- ➔ L'information doit traverser  $3 \times 5 = 15$  couches de « comparaison/sélection »,
- ➔ Temps de propagation est long ( $\approx 80$  ns),
- ➔ Le débit est limité à 12 Mbit/s.

CLMF : composante la moins fiable.

08 décembre 2003

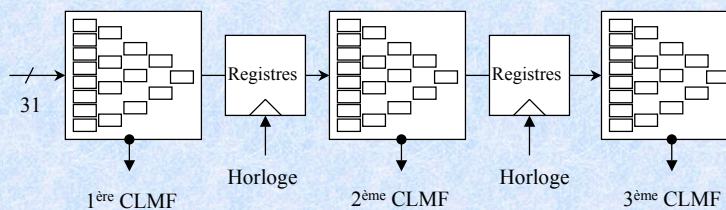
38

- 1- Codes produits,
- 2- Décodage itératif des codes produits : turbo codes en blocs,
  - Décodage à entrée pondérée : décodage de Chase,
  - Décodage à sortie pondérée : calcul de l'information extrinsèque.
- 3- Implémentation du turbo décodeur des codes produits :
  - Étude et optimisation de divers paramètres,
  - Architectures du décodeur,
  - Architecture des mémoires.
- 4- Optimisation du turbo décodeur,
- 5- Conclusions et perspectives.

08 décembre 2003

39

Solution : pipeliner la sélection des composantes les moins fiables,



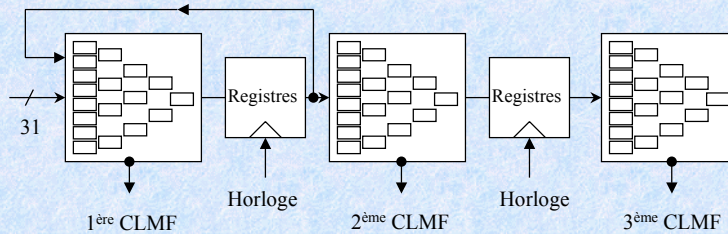
- ➔ Augmentation de la cadence de traitement (période d'horloge = 20 ns),
- ➔ Augmentation du débit (50 Mbit/s),
- ➔ Augmentation de complexité (ajout des registres).

CLMF : composante la moins fiable.

08 décembre 2003

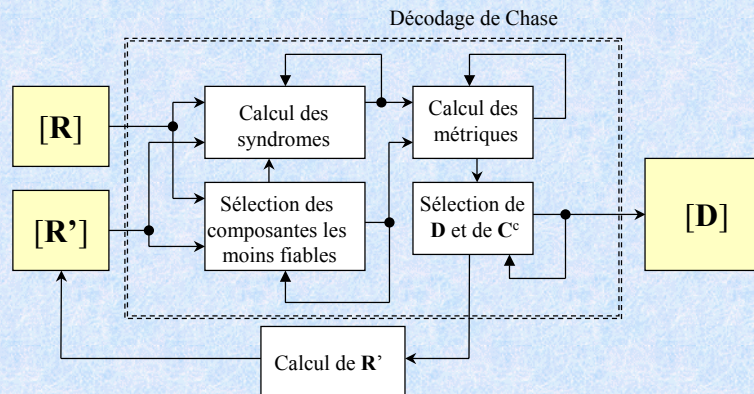
40

Solution : utilisation de la structure itérative.



➔ Réduction de la complexité sans augmenter le temps de propagation

CLMF : composante la moins fiable.



➔ Augmentation du débit à 50 Mbit/s,  
Diminution de complexité plus de la moitié.

	Débit maximal (Mbit/s)	Complexité (éléments logiques)	Nombre de bascules
Codeur	64	140 ( $\approx 1700$ portes)	74
Décodeur	12	9576 ( $\approx 115.000$ portes)	539
Décodeur optimisé	50	4251 ( $\approx 51.000$ portes)	1980

Composant APEX 20K (Altera)

- 1- Codes produits,
- 2- Décodage itératif des codes produits : turbo codes en blocs,
  - Décodage à entrée pondérée : décodage de Chase,
  - Décodage à sortie pondérée : calcul de l'information extrinsèque.
- 3- Implémentation du turbo décodeur des codes produits :
  - Étude et optimisation de divers paramètres,
  - Architectures du décodeur,
  - Architecture des mémoires.
- 4- Optimisation du turbo décodeur,
- 5- Conclusions et perspectives.

- Structure itérative à traitement par bloc :
  - Débit : 50 Mbit/s,
  - Encombrement : 4500 EL,
  - Paramétrable : nombre d'itérations.
  
- Mémoire à accès multiples :
  - lecture et écriture des vecteurs aussi bien en ligne qu'en colonne en une seule période d'horloge,

- Mémoires à accès multiples :
  - Mise en parallèle de plusieurs module de turbo décodeur
- ➔ Augmentation de débit (de 100 Mbit/s à 1,6 Gbit/s).
  
- FPGA = reprogrammable,
- Reconfiguration dynamique,
- ➔ Adapter le même turbo codeur/décodeur aux différentes dimensions de codes.