Titre: Title:	Spécification d'un décodeur séquentiel rapide utilisant une queue prioritaire systolique
Auteurs: Authors:	Pierre Lavoie, David Haccoun, & Yvon Savaria
Date:	1988
Type:	Rapport / Report
Référence: Citation:	Lavoie, P., Haccoun, D., & Savaria, Y. (1988). Spécification d'un décodeur séquentiel rapide utilisant une queue prioritaire systolique. (Rapport technique n° EPM-RT-88-11). https://publications.polymtl.ca/10109/

Institution:	École Polytechnique de Montréal
Numéro de rapport: Report number:	EPM-RT-88-11
URL officiel: Official URL:	
Mention légale: Legal notice:	

RAPPORT TECHNIQUE
EPM/RT-88/11

SPÉCIFICATION D'UN DÉCODEUR SÉQUENTIEL RAPIDE UTILISANT UNE QUEUE PRIORITAIRE SYSTOLIQUE[1]

Pierre Lavoie
Etudiant au Ph.D.

David Haccoun
Prof. titulaire

Yvon Savaria
Prof. Adjoint

Département de Génie Electrique
Ecole Polytechnique de Montréal

Mars 1988

[1]Cette recherche a été subventionnée en partie par le Conseil de la Recherche en Sciences Naturelles et en Genie du Canada et par les Fonds FCAR de Québec

Dépôt légal, 1er trimestre 1988
Bibliothèque nationale du Québec
Bibliothèque nationale du Canada

Pour se procurer une copie de ce document, s'adresser au:

Éditions de l'École Polytechnique de Montréal
École Polytechnique de Montréal
Case postale 6079, Succursale A
Montréal (Québec) H3C 3 A7
(514) 340-4000

Compter 0,10$ par page (arrondir au dollar le plus près) et ajouter 3,00$ (Canada) pour la couverture, les frais de poste et la manutention. Régler en dollars canadiens par chèque ou mandat-poste au nom de l'École Polytechnique de Montréal. Nous n'honorerons que les commandes accompagnées d'un paiement, sauf s'il y a eu entente préalable dans le cas d'établissements d'enseignement, de sociétés ou d'organismes canadiens.

SPÉCIFICATION D'UN DÉCODEUR SÉQUENTIEL RAPIDE UTILISANT UNE QUEUE PRIORITAIRE SYSTOLIQUE

par

Pierre Lavoie, David Haccoun, et Yvon Savaria

RÉSUMÉ

Dans ce rapport un décodeur très puissant en cours de réalisation à l'Ecole Polytechnique est décrit en détail. Ce décodeur est destiné à être utilisé avec des codes convolutionnels, et utilise l'algorithme de décodage séquentiel de Zigangirov-Jelinek, aussi appelé algorithme à pile. Le décodeur sera fabriqué à l'aide de circuits intégrés spécialisés (VLSI) conçus à l'Ecole Polytechnique.

Parmi les principales caractéristiques du décodeur figure une queue prioritaire systolique destinée à ordonner les noeuds explorés en fonction de leurs métriques. Cette queue prioritaire peut fournir à tout instant le noeud ayant la plus grande métrique. Le principal attrait de cette queue prioritaire est qu'elle soit systolique et peut donc fonctionner à grande vitesse indépendamment du nombre de noeuds qu'elle contient. De plus le décodeur peut être utilisé avec les taux de codage élevés grâce à une technique de perforation des codes.

SPÉCIFICATION D'UN DÉCODEUR SÉQUENTIEL RAPIDE UTILISANT UNE QUEUE PRIORITAIRE SYSTOLIQUE

par

Pierre Lavoie, David Haccoun, et Yvon Savaria

1 Introduction

Aujourd'hui les techniques de correction d'erreur par codage de canal sont de plus en plus utilisées pour améliorer la fiabilité des transmissions numériques. D'un point de vue pratique l'implantation de ces techniques requiert la réalisation de codeurs et de décodeurs offrant les meilleures performances possibles pour un coût et une complexité donnés [1,2].

Dans ce rapport un décodeur très puissant en cours de réalisation à l'Ecole Polytechnique est décrit en détail. Ce décodeur est destiné à être utilisé avec des codes convolutionnels, et utilise l'algorithme de décodage séquentiel de Zigangirov-Jelinek, aussi appelé algorithme à pile. Comme plusieurs documents traitent en détail des codes convolutionnels et des différents algorithmes permettant de les décoder [1], dans ce rapport l'emphase est directement mise sur la caractérisation du décodeur.

Parmi les principales caractéristiques du décodeur figure une queue prioritaire systolique destinée à ordonner les noeuds explorés en fonction de leurs métriques. Cette queue prioritaire peut fournir à tout instant le noeud ayant la plus grande métrique. Le principal attrait de cette queue prioritaire est qu'elle soit systolique et peut donc fonctionner à grande vitesse indépendamment du nombre de noeuds qu'elle contient [3]. De plus le décodeur peut être utilisé avec les taux de codage élevés grâce à une technique de perforation des codes. Enfin le décodeur sera fabriqué à l'aide de circuits intégrés spécialisés (VLSI) conçus à l'Ecole Polytechnique par des étudiants de 2^e cycle.

Le rapport débute par une description détaillée des différentes parties du décodeur et de leur fonctionnement. Ensuite le contrôle du décodeur est introduit et chaque étape du décodage d'un bloc est expliquée en détail. Les

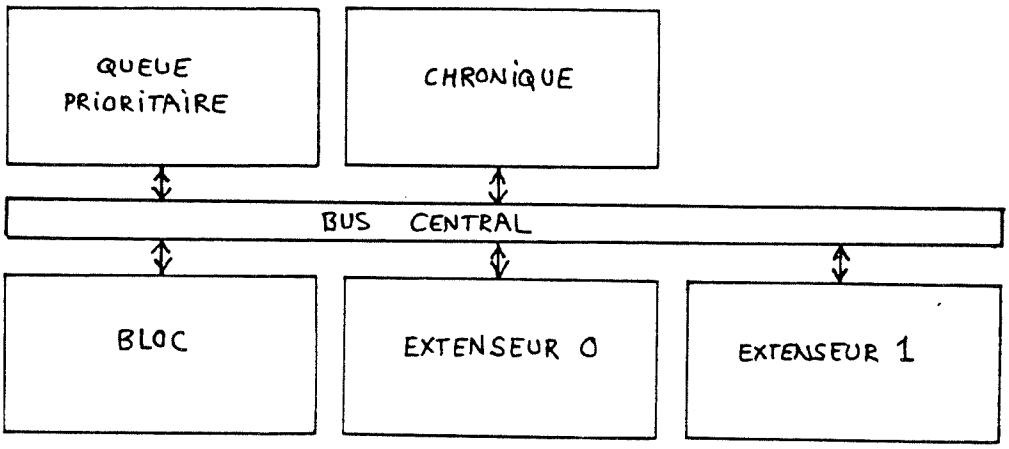

Figure 1: Structure générale du décodeur

sections suivantes du rapport traitent brièvement de la synchronisation et de la perforation des codes. Enfin trois Annexes complètent la description du décodeur. La première Annexe présente une analyse fonctionnelle de la queue prioritaire, la seconde donne les détails de la synchronisation, et la troisième donne une table de tous les signaux mentionnés dans ce rapport avec leurs abréviations.

2 Aperçu du Décodeur

Le décodeur séquentiel est constitué de cinq modules indépendants travaillant de concert au décodage d'un bloc de symboles reçus. La structure générale du décodeur apparaît à la Figure 1 où l'on aperçoit le module queue prioritaire, le module historique, le module bloc et deux modules extenseurs. Chaque module est responsable d'un aspect préçis du décodage. Sans perte de généralité, on suppose que le taux de codage est fixé à $R_N = 1/2$.

- Le module *queue prioritaire* ordonne les données relatives aux noeuds visités en fonction de leurs métriques accumulées.

- Le module *historique* tient à jour une séquence chronologique de chaque pas de décodage et permet la récupération de la séquence décodée par le biais d'une liste chaînée.
- Le module *bloc* garde en mémoire la séquence reçue pour toute la durée du décodage. Chaque paire de symboles reçus correspondant à un bit d'information peut être consultée directement.
- Les modules *extenseurs* font simultanément l'extension des deux branches reliées à un noeud.

Les différents modules communiquent entre eux principalement par l'intermédiaire d'un *bus central* qui est aussi accessible au monde extérieur. Cette architecture ouverte permet une observation complète et détaillée du décodeur pendant son fonctionnement. D'autre part le décodeur peut facilement être modifié par l'ajout de modules additionnels.

Le traitement de chaque bloc de données se fait en trois étapes: tout d'abord le monde extérieur fournit au décodeur un bloc de symboles reçus; ensuite le décodeur effectue le décodage de ce bloc; enfin la séquence décodée est livrée au monde extérieur. Le décodeur proposé ne contient donc ni tampon d'entrée, ni tampon de sortie. On suppose que le monde extérieur effectue le tamponage nécessaire et accumule les statistiques voulues concernant le décodage par une observation adéquate du bus central du décodeur.

Le décodeur sera fabriqué à l'aide de circuits intégrés développés au laboratoire d'intégration à très grande échelle (VLSI) de l'Ecole Polytechnique. Les caractéristiques du décodeur sont données au Tableau 1. Il est à noter que le code, le patron de perforation, et les métriques de symboles sont programmables. La longueur d'un bloc est déterminée automatiquement lors de sa lecture. Le décodeur peur être catégorisé comme une machine à états finis synchronisée autour d'une horloge à trois phases [4].

3 Les Différentes Parties du Décodeur

3.1 Le Module Queue Prioritaire

Le module queue prioritaire accumule les données des noeuds candidats à l'extension et fournit à chaque pas de décodage le meilleur noeud, c'est-à-dire celui qui a la plus grande métrique accumulée. Le module contient une

Paramètre	*min*	*max*
métrique accumulée	$+0$	$+128K-1$
métrique de symbole	-256	$+255$
métrique de branche	-512	$+510$
longueur N d'un bloc en branches	2	2047
taille de l'historique	0	$32K-1$
longueur de contrainte L	0	24
nombre Q de niveaux de quant.	1	8
numérateur U du taux de codage R_N	1	16
dénominateur V du taux de codage	2	32

Tableau 1: Caractéristiques du décodeur

matrice systolique constituée de processeurs spécialisés organisés en *tranches*. Chaque tranche comprend trois rangées ayant chacune 77 éléments. En juxtaposant plusieurs tranches on obtient une matrice systolique [4] de profondeur voulue.

La matrice systolique répète systématiquement deux opérations: elle prend simultanément deux nouveaux noeuds, et livre ensuite le meilleur des noeuds qu'elle contient. Dans la matrice les noeuds sont déplacés selon un algorithme très simple qui garantit qu'à tout moment le meilleur noeud de la matrice se trouve dans la tranche du bas, prêt à sortir. Pour plus de détails sur cet algorithme et une preuve que seul le meilleur noeud peut sortir de la matrice, le lecteur peut consulter l'Annexe A de ce rapport. Il est à noter qu'un débordement de la matrice systolique n'est pas catastrophique et peut même s'avérer désirable s'il est correctement contrôlé. En effet seuls les noeuds dont la qualité relative est inférieure à un seuil donné peuvent déborder de la matrice. On peut donc déterminer quelle est la profondeur minimale de la queue requise pour que le décodeur fontionne de façon satisfaisante.

La matrice systolique a une propriété remarquable: sa vitesse de fonctionnement est indépendante de sa taille car les calculs sont distribués entre un grand nombre de processeurs élémentaires fonctionnant simultanément. On peut donc envisager l'utilisation d'une matrice systolique de grande dimensions pour réaliser la queue prioritaire du décodeur séquentiel sans conséquence sur la vitesse de décodage.

En plus de la métrique accumulée nécessaire pour comparer les noeuds entre eux, la queue prioritaire garde toutes les informations nécessaires pour l'extension d'un noeud. Ces informations sont décrites au Tableau 2.

Paramètre	*bit(s)*
métrique accumulée	17
symboles reçus	2×3
état du codeur	23
profondeur dans l'arbre	11
branche dans la queue du bloc	1
adresse du noeud-père	15
compteur pour taux de codage élevés	4
total	77

Tableau 2: Contenu de la queue prioritaire

Il faut noter cependant que la queue prioritaire ignore la nature des données qu'elle traite. Par exemple elle ne fait aucune distinction entre les bits d'état du codeur et les bits de profondeur dans l'arbre. De son point de vue chaque noeud est constitué d'un nombre positif de 17 bits (métrique) et des 60 bits d'information complémentaire qui lui sont attachés.

Lors du décodage il arrive régulièrement que les deux nouveaux noeuds fournis par l'extenseur aient la même métrique accumulée. Pour éviter un biais dans le décodage il faut tenter d'éviter que la matrice systolique favorise toujours le même des deux noeuds. Une première solution au problème consiste à inverser à chaque cycle l'ordre dans lequel les noeuds entrent dans la Queue. Une deuxième solution consiste à insérer un petit générateur de nombres aléatoires près de la tranche de la matrice systolique qui reçoit les noeuds de l'extérieur. Si cette tranche établit que les deux nouveaux noeuds ont la même métrique, un nombre aléatoire sert alors à donner une priorité à l'un des deux noeuds.

La queue prioritaire sera fabriquée à l'aide d'une matrice de circuits intégrés dédiés. Avec la technologie disponible à l'Ecole Polytechnique il est en effet impossible de réaliser ce module sur un seul circuit: le nombre maximal de broches par circuit (68) est insuffisant et le nombre de transistors requis est beaucoup trop grand—plusieurs centaines de milliers. L'approche

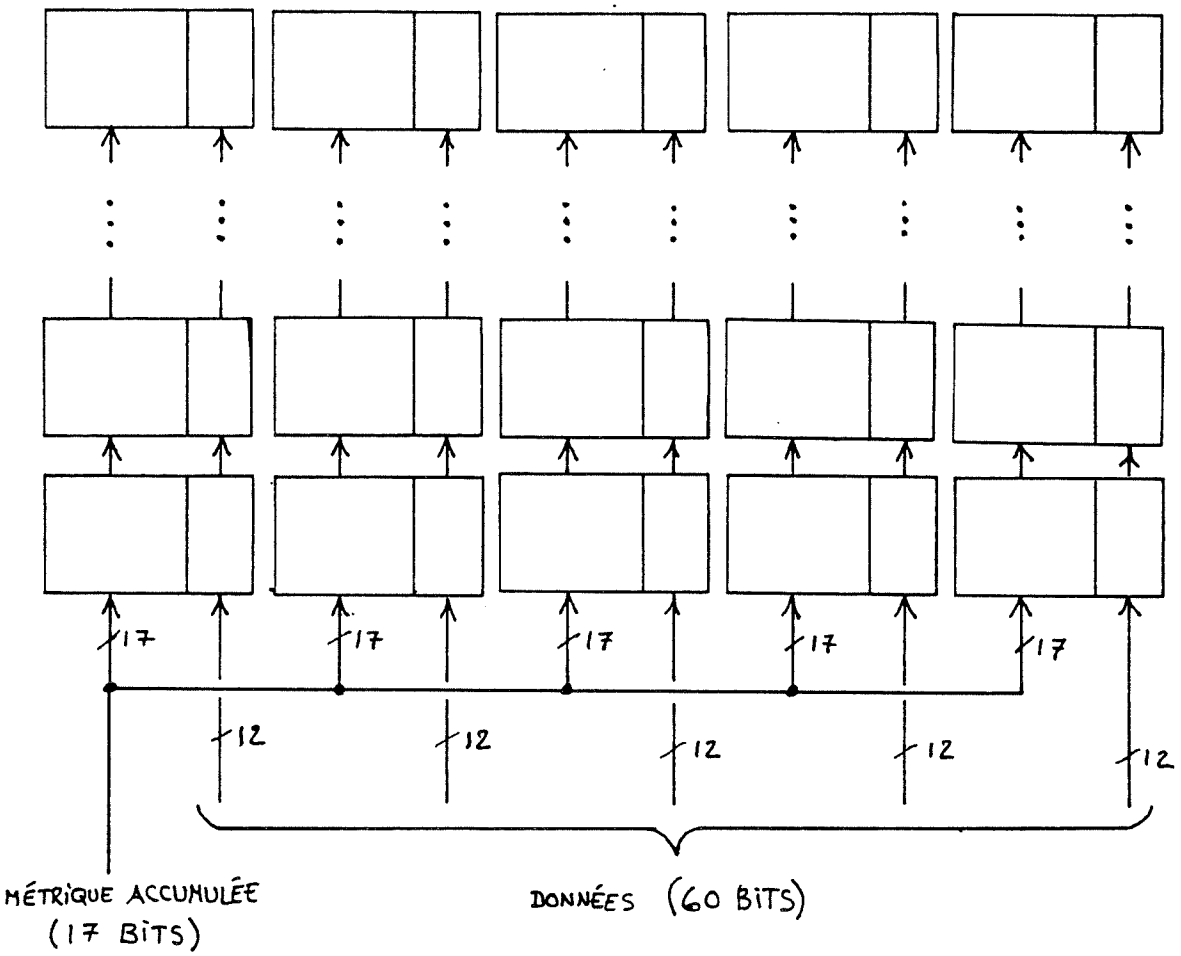

Figure 2: Matrice de circuits constituant la queue prioritaire

choisie consiste plutôt à sectionner symétriquement la pile pour n'obtenir que des sections identiques, chaque section étant suffisamment petite pour être contenue dans un seul circuit intégré. La profondeur de la queue prioritaire ainsi que le nombre de bits de données par noeud peuvent ainsi être ajustés en changeant respectivement le nombre de rangées ou de colonnes de la matrice de circuits intégrés.

Chaque circuit intégré requiert les métriques accumulées des noeuds (17 bits) et peut accepter jusqu'à 12 bits de données. Comme chaque noeud entré dans la queue prioritaire comprend 60 bits de données en plusdes 17 bits pour la métrique accumulée, 5 colonnes de circuits ($5 \times 12 = 60$) sont donc nécessaires pour réaliser la queue prioritaire. Le nombre de rangées de circuits sera ajusté en fonction de la profondeur de la queue prioritaire et de la complexité de chaque circuit intégré. La matrice de circuits resultante est montrée à la Figure 2.

3.2 Le Module Historique

Le module historique est utilisé pour récupérer à la fin d'un bloc la séquence décodée. En effet puisque les noeuds se déplacent continuellement dans la queue prioritaire, il est difficile de les lier les uns aux autres par des pointeurs. En complémentant la queue prioritaire par une mémoire conventionnelle liant tous les noeuds étendus à leurs noeuds-pères respectifs, on règle le problème sans réduire la vitesse du décodeur puisque les modules historique et queue prioritaire peuvent alors fonctionner en parallèle. L'ensemble des deux modules queue prioritaire et historique constitue ce qu'il convient d'appeler la *pile* du décodeur.

Le fonctionnement du module historique se résume comme suit:

- Le premier pas du décodage consiste à étendre le premier noeud de l'arbre appelé noeud-origine. Ce noeud-origine a une adresse égale à 0 dans l'historique. Les deux successeurs obtenus sont fournis à la queue prioritaire avec cette adresse (0) et un bit (0 ou 1) les distinguant.

- A chacun des pas suivants de décodage, un noeud sort de la pile pour être étendu à son tour. L'historique récupère le bit et l'adresse qui sort avec le noeud pour les insérer dans sa mémoire à l'adresse courante qui est ensuite incrémentée de 1. Les deux successeurs résultants de l'extension sont ensuite insérés dans la queue prioritaire avec l'adresse courante et le bit (0 ou 1) qui les distingue.

- Lorsque la fin du bloc à décoder est atteinte, l'adresse courante de l'historique permet la récupération de la séquence décodée.

Il est toujours possible que durant le décodage d'un bloc l'historique manque de mémoire, ce qui correspond à un débordement de la pile. Afin de limiter les conséquences facheuses de cette éventualité un mécanisme de protection a été prévu. Si l'historique détecte que le nombre de positions en mémoire qu'il lui reste est égal à la longueur du bloc à décoder, elle peut émettre un signal de "débordement d'historique" (HO) qui forcera le décodeur à entrer dans l'état de décodage "marche-avant", où les retours en arrière sont interdits. De cette façon la perte du bloc est évitée et la partie du bloc qui a été décodée correctement peut être récupérée. Cependant ce bloc sera déclaré non fiable.

Comme 15 bits ont étés réservés dans la pile pour les adresses des noeuds, chaque adresse est limitée à une valeur maximale égale à $2^{15}-1$ ou 32767. La mémoire de l'historique aura donc 32768 entrées de 16 bits chacune (15 bits d'adresse et un bit d'information). Pour augmenter le nombre d'entrées dans l'historique, et diminuer le risque de débordement, il serait possible d'utiliser des adresses plus longues. Par exemple des adresses de 20 bits permettraient plus d'un million d'entrées dans l'historique. Cependant il faudrait ajouter une colonne supplémentaire de circuits intégrés dans la queue prioritaire pour accomoder les bits d'adresse supplémentaires.

3.3 Le Module Bloc

Le module bloc garde en mémoire une séquence reçue du monde extérieur (le canal) pendant toute la durée de son décodage. A chaque pas de décodage le module lit du bus central la profondeur du noeud sorti de la pile et extrait de sa mémoire la branche qui devra être utilisée pour l'extension éventuelle d'un des successeurs de ce noeud. Le module bloc fontionne donc avec un pas de décodage en avance sur les modules extenseurs, et transmet cette branche à la queue prioritaire plutôt qu'aux extenseurs. De cette manière chaque extenseur reçoit simultanément toutes les informations nécessaires pour étendre un noeud et le décodeur peut fonctionner plus rapidement.

Comme le module bloc connaît le nombre de branches qu'il y a dans la séquence à décoder, il peut détecter la fin du décodage et transmettre un message "décodage terminé" (DF) au contrôleur si la profondeur du noeud sorti de la pile est égale à la longueur du bloc. L'entrée "profondeur du noeud" (ND) de la pile a une largeur de 11 bits, et la longueur d'un bloc est limitée à 2047 branches. Une branche reçue contient deux symboles de trois bits chacun—le décodeur fonctionne en décision douce. Pour chaque branche un bit supplémentaire indiquant si la branche est dans la queue du bloc (IQ) est aussi gardé en mémoire.

3.4 Les Modules Extenseurs

Dans le décodeur deux modules extenseurs sont utilisés pour étendre le noeud-père sorti de la pile selon les branches 0 et 1 correspondantes aux bits d'information 0 et 1 respectivement. Les deux extenseurs fontionnent

simultanément et seront réalisés à l'aide de deux circuits intégrés physiquement identiques informés de leurs identités—extenseur 0 ou extenseur 1—lors de l'initialisation du décodeur.

Lors du décodage, chaque extenseur lit du bus central l'état (ST), la métrique du noeud-père (NM), les deux symboles reçus ($S1$ et $S2$), le bit indiquant si la branche est dans la queue du bloc (IQ), et la valeur du compteur de perforation (PE). L'extension débute par l'estimation des deux symboles transmis à l'aide de l'état et de deux arbres de ou-exclusifs. En comparant les deux estimés obtenus avec les symboles reçus, le module choisit deux métriques de symboles qui peuvent ensuite être perforées, ou forcées à zéro, selon la valeur du compteur de perforation—voir la Section 7 pour plus de détails. La somme des deux métriques de symboles donne une métrique de branche qui peut être ajoutée à la métrique du noeud-père pour donner la métrique du noeud successeur de l'extenseur. L'extenseur peut émettre un signal sur le bus central si la métrique du noeud successeur est devenue trop grande (signal MO). Une vue détaillée de l'extenseur apparaît à la Figure 3.

Lorsque le noeud sorti de la queue prioritaire est dans la queue du bloc ($IQ = 1$), la métrique du noeud successeur de l'extenseur 1 est forcée à zéro pour annuler la branche 1. Similairement, lorsque le décodeur est dans l'état décodage marche-avant, la métrique du noeud successeur de l'extenseur 1 est encore forcée à zéro, tandis que la métrique de branche de l'extenseur 0 est forcée à un pour augmenter systématiquement la métrique accumulée du noeud et empêcher les retours en arrière.

Différentes données doivent être stockées dans chacun des extenseurs lorsque le décodeur est dans l'état initialisation: les connections des deux arbres de ou-exclusifs, les métriques de symboles, le numérateur du taux de codage, l'identité de l'extenseur, les patrons de perforation et la métrique du premier noeud de l'arbre. Ces données sont identifiées au Tableau 3. Puisque les identités des deux extenseurs sont différentes, les deux modules doivent avoir des adresse de modules (MA) différentes pour pouvoir être programmés à tour de rôle—voir la section 4.2 pour plus de détails.

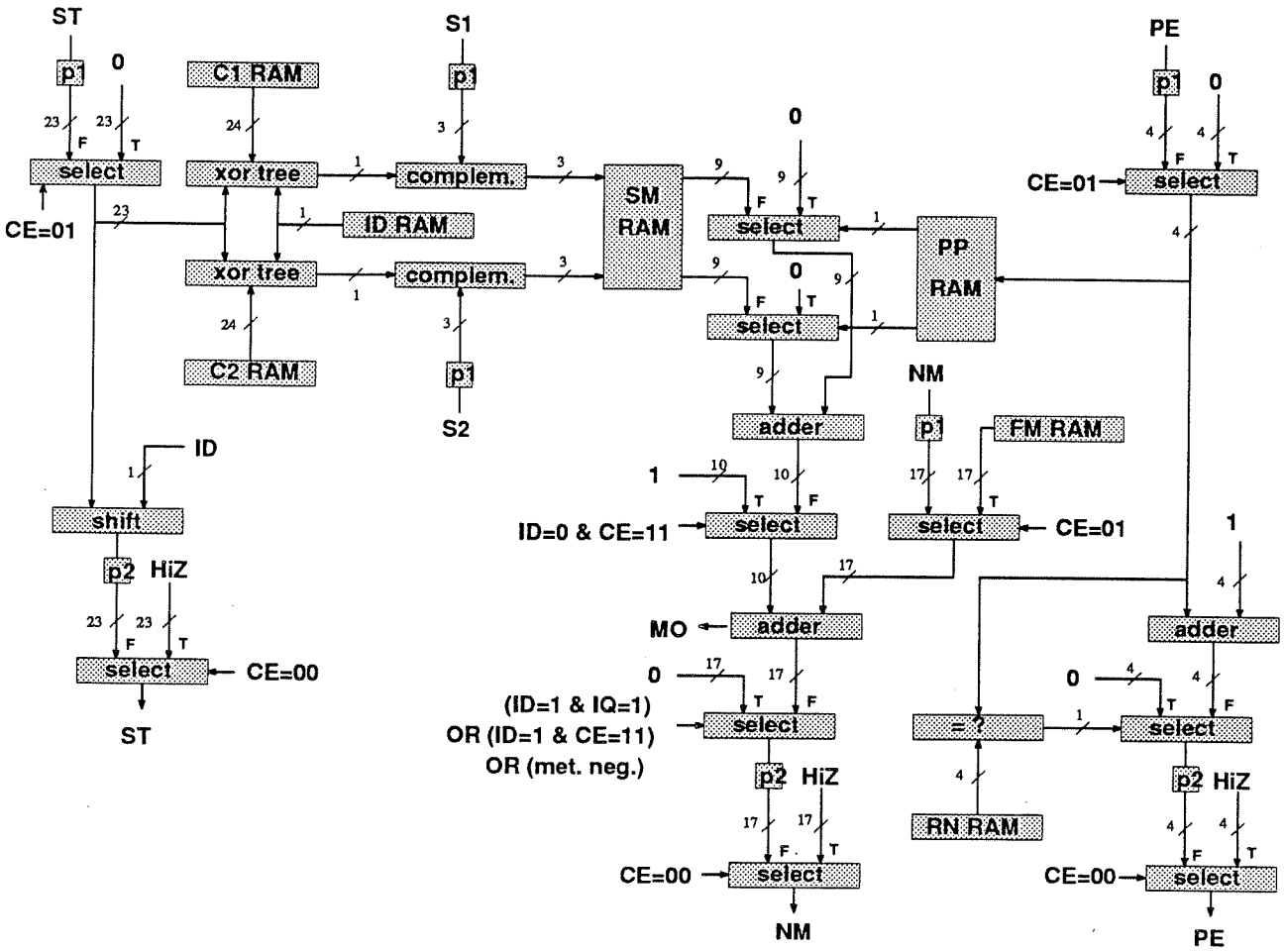

Figure 3: Schéma détaillé de l'extenseur

mémoire	*abrév.*	*bit(s)*
connections de l'arbre 1	$C1$	24
connections de l'arbre 2	$C2$	24
métriques de symboles	SM	8×9
numérateur du taux de codage	RN	4
identité de l'extenseur	ID	1
patrons de perforation	PP	16×2
métrique du premier noeud	FM	17

Tableau 3: Données mémorisées par un extenseur

3.5 Le Bus Central

Le bus central relie les modules du décodeur entre eux et leur permet de communiquer avec le monde extérieur. Il se décompose en 86 lignes, comme indiqué au Tableau 4. Il est à noter qu'il faut deux lignes MO pour permettre le signalement d'un débordement de métrique par chacun des deux extenseurs.

3.6 Le Contrôleur

Le rôle du contrôleur est de coordonner les opérations des différents modules pour que le décodeur puisse fonctionner en synchronisme et conformément aux commandes provenant du monde extérieur. Durant la phase $\phi 2$ le contrôleur lit les signaux de commande provenant du bus central du décodeur et du monde extérieur. Durant la phase $\phi 1$ il émet les signaux de commande spécifiques aux différents modules. Le contrôleur contient une boucle de contre-réaction qui permet de considérer le décodeur comme une machine à états finis. Le contrôleur est schématisé à la Figure 4.

nom	*abrév.*	*lignes*
métrique de noeud	NM	17
état	ST	23
adresse de noeud	NA	15
profondeur de noeud	ND	11
perforation	PE	4
symbole reçu 1	$S1$	3
symbole reçu 2	$S2$	3
dans la queue du bloc	IQ	1
décodage terminé	DF	1
écriture terminée	WF	1
historique débordée	HO	1
débordement de métrique	MO	2
adresse sur le bus	BA	3
donnée	DA	1

Tableau 4: Lignes du bus central

4 Le Contrôle

4.1 Les cinqs états

Le décodeur séquentiel est contrôlé comme une machine à états finis utilisant cinq états: initialisation, lecture, décodage libre, décodage marche-avant, et écriture. Le contrôleur commande le passage d'un état à un autre en utilisant les informations provenant du décodeur ou du monde extérieur.

Le monde extérieur peut transmettre au contrôleur deux signaux: RD et NS. Le premier signal force le décodeur à passer directement dans l'état initialisation et doit normalement être utilisé lorsque le décodeur est mis en marche. Le second signal fait passer le décodeur de l'état initialisation à l'état lecture, ou si le décodeur est déja dans l'état lecture, de l'état lecture à l'état décodage libre. Une fois dans l'état décodage libre le décodeur décide lui-même comment il retournera à l'état initialisation. Ce retour peut s'effectuer de plusieurs façons différentes tel qu'illustré à la Figure 5. Examinons maintenant de plus près ces cinq états.

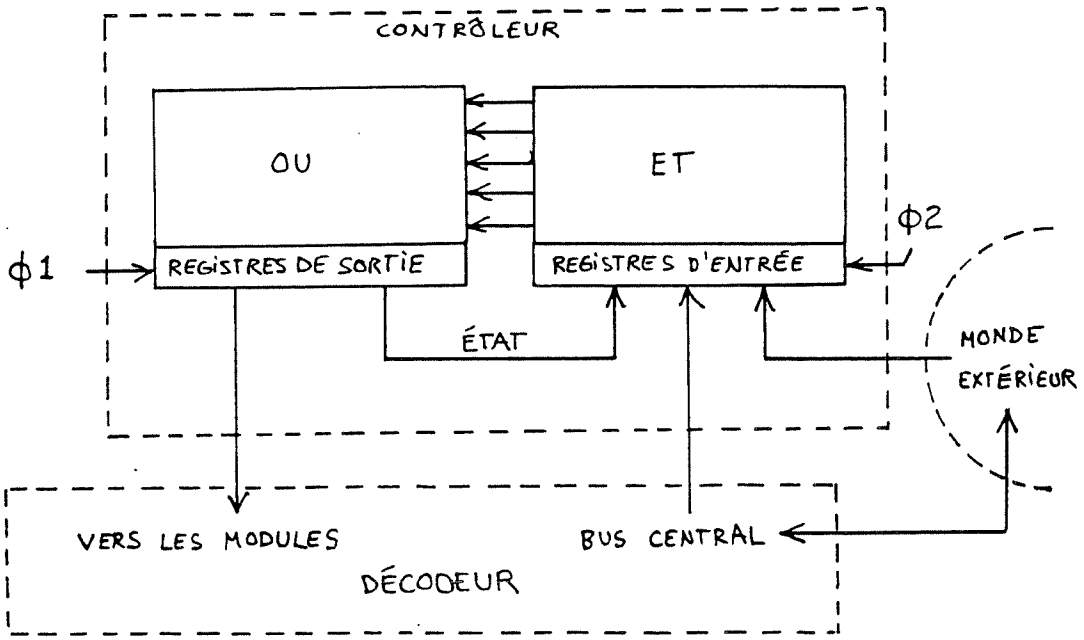

Figure 4: Le contrôleur

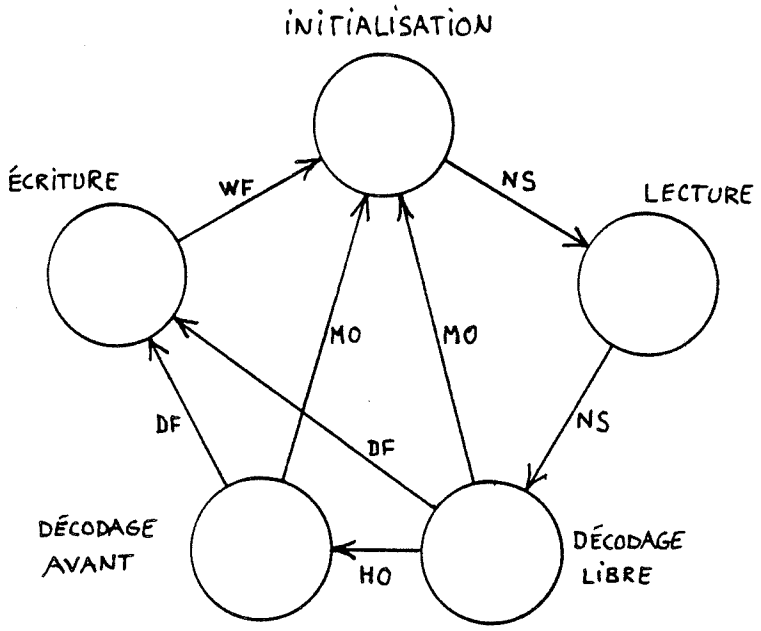

Figure 5: Diagramme d'états du décodeur

4.2 Etat Initialisation

Dans l'état initialisation, chaque module du décodeur peut être programmé individuellement par le monde extérieur. Une seule ligne sur le bus central, la ligne DA, suffit au transfert des données tandis que les trois lignes "adresse sur le bus" BA servent à activer leou les modules concernés par les données. Chaque module doit se conformer au protocole suivant s'il veut utiliser la ligne DA:

1. Comparer sa propre adresse MA avec l'adresse BA du bus central
2. Si les deux adresse sont identiques, ou si l'adresse BA du bus central égale 1, le module doit lire la donnée présente sur la ligne DA.
3. Par convention les adresse 0 et 1 sont spéciales et ne peuvent pas être utilisées comme adresse d'un module en particulier. Ainsi l'adresse 1 signifie "tous les modules" et l'adresse 0 "aucun module". Lorsque dans l'état initialisation l'adresse BA du bus égale 0, le décodeur reste donc en attente.

Normalement la programmation des modules n'est faite qu'une seule fois, avant le décodage du premier bloc. Par la suite l'état initialisation peut être considéré comme un état d'attente entre les blocs.

4.3 Etat Lecture

Dans l'état lecture, cycle après cycle le module bloc lit du bus central deux symboles reçus $S1$ et $S2$ provenant du monde extérieur , ainsi qu'un bit IQ indiquant si les symboles font partie de la queue du bloc. Le module garde ces informations dans une mémoire RAM tout en incrémentant un compteur d'adresse qui contient la longueur du bloc une fois la lecture terminée. Dans l'état lecture, tous les autres modules sont inactifs.

4.4 Passage de l'état lecture à l'état décodage libre

Lorsque le décodeur passe de l'état lecture à l'état décodage libre, les extenseurs étendent le noeud-origine de l'arbre et donnent les deux successeurs obtenus à la queue prioritaire. Ce cycle de décodage est donc particulier

puisque les extenseurs n'utilisent pas un noeud sorti de la queue prioritaire, mais utilisent plutôt les valeurs initiales prévues pour le premier noeud de l'arbre. Exceptionnellement les deux symboles reçus sont communiqués du module bloc aux extenseurs au début du cycle, et la métrique de départ du noeud-origine provient de la mémoire FM qui doit avoir été préalablement programmée dans l'état initialisation.

4.5 Etat décodage libre

Tous les cycles suivants de décodage libre sont identiques. D'abord la queue prioritaire met le meilleur noeud sur le bus central, puis le module bloc,le module historique, et les modules extenseurs calculent toutes les caractéristiques des noeuds successeur. Ces caractéristiques sont écrites sur le bus central et lues par la queue prioritaire avant la fin du cycle. Le contrôleur utilise ensuite les signaux HO, MO, et DF, pour déterminer quel état sera le décodeur au cycle suivant.

- S'il y a un débordement d'historique ($HO = 1$), alors le décodeur passe à l'état décodage marche-avant.
- S'il y a un débordement de métrique ($MO = 1$), alors le décodeur retourne dans l'état initialisation.
- Si le décodage est terminé ($DF = 1$), alors le décodeur passe à l'état écriture.

4.6 Etat décodage marche-avant

Dans l'état décodage marche-avant, le décodeur fonctionne comme en décodage libre à la seule exception que les extenseurs choisissent les métriques des noeuds successeurs d'une façon telle que seul le dernier noeud obtenu par l'extension d'une branche 0 peut ressortir de la queue prioritaire. Pour ce faire l'extenseur 0 donne systématiquement à son noeud successeur la métrique du noeud-père incrémentée de un, tandis que l'extenseur 1 force toujours à zéro la métrique de son propre noeud successeur.

4.7 Etat écriture

Dans l'état écriture le module historique écrit sur le bus central la séquence décodée bit par bit. L'ordre d'écriture des bits est inversé par rapport à l'ordre de lecture des symboles reçus. Pendant ce temps tous les autres modules sont inactifs. Lorsque l'écriture est terminée, le module historique émet sur le bus central un signal WF indiquant au contrôleur que le décodeur peut retourner dans l'état initialisation.

4.8 Signaux de contrôle

Alors que la communication allant du décodeur jusqu'au contrôleur se fait par le bus central—via les signaux DF, WF, HO, et MO—, la communication dans la direction inverse se fait par des lignes séparées reliant le contrôleur aux différents modules. Les Tableaux 5 et 6 décrivent ces lignes de commande.

nom	*abrév.*	*lignes*
commande du module queue prioritaire	CQ	1
commande du module historique	CH	2
commande du module bloc	CB	2
commande des modules extenseurs	CE	2

Tableau 5: Signaux de contrôle des modules

Comme indiqué précédemment, le monde extérieur contrôle le décodeur par les signaux "prochain état" (NS) et "initialise décodeur" (RD). Maintenant que les entrées et les sorties du décodeur sont clairement identifiées, une table de transition d'états du décodeur peut être définie [4]. La Table 7 décrit toutes les transitions d'états possibles.

5 La Synchronisation Interne du Décodeur

L'architecture du décodeur a été conçue pour être synchronisée par une horloge à deux phases sans recouvrement $\phi 1$ et $\phi 2$. Cependant, en pratique les circuits intégrés constituant la queue prioritaire n'ont pas assez de broches

code	*description*
$CQ = 0$	lecture des noeuds successeurs
$CQ = 1$	écriture du meilleur noeud puis lecture des noeuds successeurs
$CH = 00$	programmation
$CH = 01$	mise à zéro puis ériture de l'adresse courante
$CH = 10$	lecture de l'adresse du père puis écriture de l'adresse courante
$CH = 11$	écriture d'un bit décodé
$CB = 00$	initialisation du compteur de branches
$CB = 01$	lecture d'une branche
$CB = 10$	écriture de la première branche puis de la deuxième
$CB = 11$	lecture de la profondeur puis écriture de la branche correspondante
$CE = 00$	programmation
$CE = 01$	extension du premier noeud de l'arbre
$CE = 10$	extension du meilleur noeud en décodage libre
$CE = 11$	extension du meilleur noeud en décodage marche-avant

Tableau 6: Description des signaux de contrôle

pour permettre le transfert de toutes les données requises en deux phases seulement, et une troisième phase $\phi 3$ a donc due être ajoutée. Ainsi à l'intérieur d'un cycle de décodage les circuits de la queue prioritaire écrivent sur le bus central le meilleur noeud durant $\phi 1$, et lisent les deux noeud successeurs durant $\phi 2$ et $\phi 3$.

L'Annexe B du rapport donne le détail de l'utilisation du bus central par les différents modules pour les trois phases $\phi 1$, $\phi 2$, et $\phi 3$. L'utilisation du bus central change selon l'état dans lequel se trouve le décodeur, et chaque transfert de données doit être rigoureusement synchronisé pour éviter toute "collision" sur le bus. Rappellons que le contrôleur lit les signaux du bus durant $\phi 2$ et propage les signaux de commande des modules durant $\phi 1$.

De façon interne tous les modules du décodeur effectuent leurs calculs en utilisant deux phases seulement. Conséquemment, ormis le nombre insuffisant de broches de quelques circuits intégrés, le décodeur pourait très bien fonctionner avec deux phases. L'architecture naturellement parallèle du décodeur n'est donc pas pleinement exploitée, et une réalisation éventuelle du décodeur entier sur un seul circuit intégré permettrait d'augmenter con-

valeurs lues durant $\phi 2$							*valeurs écrites durant $\phi 1$*				
entrées						*état*	*état*	*sorties*			
NS	RD	DF	WF	HO	MO	*présent*	*suivant*	CQ	CH	CB	CE
0	0	x	x	x	x	initial.	initial.	0	00	00	00
1	0	x	x	x	x	initial.	lecture	0	00	00	00
0	0	x	x	x	x	lecture	lecture	0	00	01	00
1	0	x	x	x	x	lecture	déc. lib.	0	01	10	01
x	0	0	x	0	0	déc. lib.	déc. lib.	1	10	11	10
x	0	0	x	1	0	déc. lib.	déc. av.	1	10	11	11
x	0	x	x	x	1	déc. lib.	initial.	0	00	00	00
x	0	1	x	x	0	déc. lib.	écriture	0	11	00	00
x	0	0	x	x	0	déc. av.	déc. av.	1	10	11	11
x	0	x	x	x	1	déc. av.	initial.	0	00	00	00
x	0	1	x	x	0	déc. av.	écriture	0	11	00	00
x	0	x	0	x	x	écriture	écriture	0	11	00	00
x	0	x	1	x	x	écriture	initial.	0	00	00	00
x	1	x	x	x	x	tous	initial.	0	00	00	00

Tableau 7: Transitions d'états du décodeur

sidérablement sa vitesse de fonctionnement.

6 Codes Perforés à Taux Elevés

On obtient un code perforé de taux $R_P = U/V$ si on omet systématiquement de transmettre certains symboles provenant de la sortie d'un codeur de taux R_N, $(R_P > R_N)$ [5,6]. Ces codes sont particulièrement attrayants d'un point de vue pratique puisqu'il est possible de décoder un code perforé de taux R_P avec un décodeur de taux R_N ayant une complexité réduite. Par exemple un code perforé de taux $R_P = 16/17$ peut être décodé avec un simple décodeur de taux $R_N = 1/2$, alors que le décodage d'un code non-perforé nécessiterait l'utilisation d'un décodeur de taux 16/17 considérablement plus complexe. Ainsi le décodeur de taux $R_N = 1/2$ présenté dans ce rapport peut être utilisé avec des codes de taux $R_P = U/V$, avec $U = 1, 2, \ldots, 16$ et $V = 2, 3, \ldots, 32$

$(U < V)$.

A l'intérieur du décodeur la perforation d'un symbole consiste à forcer à zéro la métrique de symbole correspondante. Afin de simplifier cette opération chaque extenseur contient un tableau des patrons de perforation (PP) indiquant pour chacune des U branches comment perforer les deux symboles. Puisque U est inférieur ou égal à 16 et que 2 bits sont nécessaires pour commander la perforation, le tableau est gardé dans une mémoire comprenant 16 registres de 2 bits. Cette mémoire peut être programmée lorsque le décodeur est dans l'état initialisation.

Pour savoir lequel des U patrons appliquer à une branche donnée, les extenseurs doivent trouver où se situe le noeud qu'ils étendent à l'intérieur de la branche du code perforé de taux R_P. Pour éviter un calcul et accélérer le décodage un nombre de 4 bits (PE) est associé à chaque noeud exploré. Ce nombre est incrémenté par les extenseurs et remis à zéro s'il atteint une valeur limite RN égale au numérateur U du taux de codage R_P. Cette valeur limite peut elle aussi être programmée lorsque le décodeur est dans l'état initialisation.

7 Conclusions

Un décodeur séquentiel à pile utilisant une architecture originale a été présenté. Des étudiants de 2^e cycle de l'Ecole Polytechnique ont commencé la conception des premiers circuits intégrés (VLSI) qui constitueront le décodeur. Le principal attrait du décodeur s'avère être sa queue prioritaire systolique qui peut ordonner les noeuds de la pile très rapidement.

Il a été démontré qu'à l'intérieur d'un cycle cette queue prioritaire peut lire deux noeuds différents puis livrer sans délai le meilleur noeud qu'elle contient. Le fonctionnement du décodeur est aussi remarquable par sa simplicité. Enfin une approche consistant à sectionner le décodeur en différents modules a été utilisée pour en faciliter l'étude.

On peut affirmer que la vitesse du décodeur sera limitée seulement par la durée des calculs requis pour l'extension d'une branche—encodage et addition sur 17 bits—et par l'insertion d'une troisième phase d'horloge permettant à la queue prioritaire de lire le deuxième noeud successeur. Le décodeur devrait donc pouvoir fonctionner à une vitesse de plusieurs millions de cycles par seconde.

Il est surprenant de constater que l'opération qui consiste à ordonner les noeuds ne fait pas partie du chemin critique et ne ralentit donc pas le décodeur. Rappellons que cette opération nécessite un temps de calcul très important lorsque l'algorithme à pile est implanté de façon traditionnelle, c'est à dire sans une queue systolique prioritaire.

Remerciements

Cette recherche est subventionnée par une subvention thématique du Conseil de la Recherche en Sciences Naturelles et en Génie du Canada, ainsi que par les Fonds FCAR de Québec. Les auteurs voudraient remercier la Société Canadienne de Microélectronique pour son support technique, Northern Telecom du Canada pour la fabrication des circuits intégrés, et tous les étudiants de 2^e cycle impliqués à ce jour dans le projet: André Beaulieu, Normand Bélanger, Jean Belzile, Michel Herzig, Denis Lachapelle, David Stannard, et Mylène Toulgoat.

References

[1] V. K. Bhargava, D. Haccoun, R. Matyas, and P. Nuspl, *Digital communications by satellite*, Wiley, New York, 1981.

[2] D. Haccoun, P. Lavoie, and Y. Savaria, "New architectures for fast convolutional encoders and threshold decoders", *IEEE Journ. Select. Areas in Comm.*, avril 1988.

[3] C. Y. Chang, and K. Yao, "Systolic array architecture for the sequential stack decoding algorithm", SPIE conference, San Diego, 1986.

[4] C. Mead, and L. Conway, *Introduction to VLSI systems*, Addison-Wesley, Reading, Mass., 1980.

[5] D. Haccoun, and Guy Bégin, "High-rate punctured convolutional codes", soumis en janvier 1988 pour publication dans le *IEEE Trans. on Comm.*.

[6] G. Bégin, and D. Haccoun, "High-rate punctured convolutional codes: properties and construction technique", soumis en mai 1987 pour publication dans le *IEEE Trans. on Comm.*.

Annexe A

De façon générale chaque cycle de fonctionnement de la matrice systolique se décompose en quatre étapes. On suppose que les noeuds entrent et sortent par le bas de la matrice:

1. Entrée de $(M-1)$ nouveaux noeuds dans la matrice et décalage de tous les autres noeuds de deux positions vers le haut.
2. Simultanément, à l'intérieur de chaque tranche de M noeuds: mise en ordre de sorte que le meilleur noeud se retrouve dans la position la plus basse de la tranche. Les positions des $(M-1)$ autres noeuds sont sans importance.
3. Sortie du meilleur noeud de la matrice et décalage de tous les autres noeuds de une position vers le bas.
4. Mise en ordre identique à l'étape 2, et retour à l'étape 1.

Le Tableau 8 donne un exemple pour M égal à trois. Les lettres e, o, et s représentent respectivement une entrée, une mise en ordre, et une sortie.

e o	*s o*	*e o*	*s o*	*e o*	*s o*	*e o*	*s o*	*e o*	*s o*
								1 1	
						2 1		2 2	1 1
				5 5		1 2	1 1	2 2	2 2
		5 5		7 1	5 2	5 3	2 2	2 2	2 0
6 5		8 7	5 5	2 2	1 1	3 3	3 2	9 0	2 2
5 6	5 5	7 8	7 7	1 7	2 5	3 5	3 2	0 9	0 2
	6		8		7		5		9

Tableau 8: Exemple de fonctionnement de la queue prioritaire

Pour démontrer que seul le meilleur noeud peut sortir de la matrice une preuve par induction sera utilisée. Associons à chaque noeud de la matrice une hauteur positive et non-nulle, et faisons l'hypothèse que la borne qui suit est correcte après m cycles de fonctionnement.

Borne 1 *Le i^{eme} meilleur noeud peut seulement se retrouver à une hauteur k telle que $1 \leq k \leq M(i-1)+1$.*

Vérifions qu'après $(m+1)$ cycles cette borne est toujours valide en étudiant les quatre parties d'un cycle.

1. Entrée de $(M-1)$ noeuds dans la matrice: dans le pire des cas les nouveaux noeuds sont infiniment mauvais, et la borne devient:

   Borne 2 *Le i^{eme} meilleur noeud peut seulement se retrouver à une hauteur k telle que $1 \leq k \leq Mi$.*

2. Mise en ordre: si un noeud n se trouve à une hauteur k telle que $M(i-1)+1 \leq k \leq Mi$, alors les $1^{er} \ldots (i-1)^{eme}$ noeuds meilleurs que le noeud n ne peuvent pas être dans sa tranche, et le noeud n descend alors jusqu'à l'hauteur $k = M(i-1)+1$ correspondante à la borne 1. D'autre part si le noeud n se trouve à une hauteur k telle que $k < M(i-1)+1$, alors la mise en ordre peut faire monter le noeud—seulement à l'intérieur de la même tranche—sans violer la borne 1.

3. Sortie d'un noeud de la matrice: comme après l'étape 2 la borne 1 est valide, c'est le meilleur noeud $(i = 1)$ qui sort, et tous les autres noeuds de la pile se retrouvent décalés d'une position vers le bas. A la fin de l'étape 3 la borne 1 n'est plus valide, mais la borne 2 redevient correcte.

4. Mise en ordre: même raisonnement qu'à l'étape 2, ce qui réétablit la borne 1 comme la borne valide, vérifie notre hypothèse, et termine la preuve par induction.

Annexe B

L'utilisation du bus central par les différents modules du décodeur varie selon la phase d'horloge qui est active ($\phi 1$, $\phi 2$, ou $\phi 3$), et l'état dans lequel se trouve le décodeur. Dans les Tableaux 9 à 23 l'écriture et la lecture d'une donnée sur le bus sont représenteés par les lettres "e" et "l" respectivement. Il est à noter que le module historique lit et écrit chaque bit d'information en utilisant une des lignes "état" (ST) du bus.

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	-	-	-	-	-	-	-	-	-	-	-	-	l	l
hist.	-	-	-	-	-	-	-	-	-	-	-	-	l	l
bloc	-	-	-	-	-	-	-	-	-	-	-	-	l	l
ext.0	-	-	-	-	-	-	-	-	-	-	-	-	l	l
ext.1	-	-	-	-	-	-	-	-	-	-	-	-	l	l
monde	-	-	-	-	-	-	-	-	-	-	-	-	e	e

Tableau 9: Etat initialisation durant $\phi 1$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	-	-	-	-	-	-	-	-	-	-	-	-	-	-
hist.	-	-	-	-	-	-	-	-	-	-	-	-	-	-
bloc	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
monde	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Tableau 10: Etat initialisation durant $\phi 2$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	-	-	-	-	-	-	-	-	-	-	-	-	-	-
hist.	-	-	-	-	-	-	-	-	-	-	-	-	-	-
bloc	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
monde	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Tableau 11: Etat initialisation durant $\phi 3$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	-	-	-	-	-	-	-	-	-	-	-	-	-	-
hist.	-	-	-	-	-	-	-	-	-	-	-	-	-	-
bloc	-	-	-	-	-	l	l	l	-	-	-	-	-	-
ext.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
monde	-	-	-	-	-	e	e	e	-	-	-	-	-	-

Tableau 12: Etat lecture durant $\phi 1$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	-	-	-	-	-	-	-	-	-	-	-	-	-	-
hist.	-	-	-	-	-	-	-	-	-	-	-	-	-	-
bloc	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
monde	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Tableau 13: Etat lecture durant $\phi 2$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	-	-	-	-	-	-	-	-	-	-	-	-	-	-
hist.	-	-	-	-	-	-	-	-	-	-	-	-	-	-
bloc	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
monde	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Tableau 14: Etat lecture durant $\phi 3$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	-	-	-	-	-	-	-	-	-	-	-	-	-	-
hist.	-	-	-	-	-	-	-	-	-	-	-	-	-	-
bloc	-	-	-	-	-	e	e	e	-	-	-	-	-	-
ext.0	-	-	-	-	-	l	l	l	-	-	-	-	-	-
ext.1	-	-	-	-	-	l	l	l	-	-	-	-	-	-
monde	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Tableau 15: Etat décodage libre (de lecture) durant $\phi 1$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	l	l	l	l	l	l	l	l	-	-	-	-	-	-
hist.	-	-	e	-	-	-	-	-	-	-	e	-	-	-
bloc	-	-	-	e	-	e	e	e	e	-	-	-	-	-
ext.0	e	e	-	-	e	-	-	-	-	-	-	e	-	-
ext.1	-	-	-	-	-	-	-	-	-	-	-	e	-	-
monde	-	-	-	-	-	-	-	-	l	-	l	l	-	-

Tableau 16: Etat décodage libre (de lecture) durant $\phi 2$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	l	l	l	l	l	l	l	l	-	-	-	-	-	-
hist.	-	-	e	-	-	-	-	-	-	-	e	-	-	-
bloc	-	-	-	e	-	e	e	e	e	-	-	-	-	-
ext.0	-	-	-	-	-	-	-	-	-	-	-	e	-	-
ext.1	e	e	-	-	e	-	-	-	-	-	-	e	-	-
monde	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Tableau 17: Etat décodage libre (de lecture) durant $\phi 3$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	e	e	e	e	e	e	e	e	-	-	-	-	-	-
hist.	-	l	l	-	-	-	-	-	-	-	-	-	-	-
bloc	-	-	-	l	-	-	-	-	-	-	-	-	-	-
ext.0	l	l	-	-	l	l	l	l	-	-	-	-	-	-
ext.1	l	l	-	-	l	l	l	l	-	-	-	-	-	-
monde	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Tableau 18: Etat décodage libre ou marche-avant durant $\phi 1$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	l	l	l	l	l	l	l	l	-	-	-	-	-	-
hist.	-	-	e	-	-	-	-	-	-	-	e	-	-	-
bloc	-	-	-	e	-	e	e	e	e	-	-	-	-	-
ext.0	e	e	-	-	e	-	-	-	-	-	-	e	-	-
ext.1	-	-	-	-	-	-	-	-	-	-	-	e	-	-
monde	-	-	-	-	-	-	-	-	l	-	l	l	-	-

Tableau 19: Etat décodage libre ou marche-avant durant $\phi 2$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	l	l	l	l	l	l	l	l	-	-	-	-	-	-
hist.	-	-	e	-	-	-	-	-	-	-	e	-	-	-
bloc	-	-	-	e	-	e	e	e	e	-	-	-	-	-
ext.0	-	-	-	-	-	-	-	-	-	-	-	e	-	-
ext.1	e	e	-	-	e	-	-	-	-	-	-	e	-	-
monde	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Tableau 20: Etat décodage libre ou marche-avant durant $\phi 3$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	-	-	-	-	-	-	-	-	-	-	-	-	-	-
hist.	-	e	-	-	-	-	-	-	-	-	-	-	-	-
bloc	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
monde	-	l	-	-	-	-	-	-	-	-	-	-	-	-

Tableau 21: Etat écriture durant $\phi 1$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	-	-	-	-	-	-	-	-	-	-	-	-	-	-
hist.	-	-	-	-	-	-	-	-	-	e	-	-	-	-
bloc	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
monde	-	-	-	-	-	-	-	-	-	l	-	-	-	-

Tableau 22: Etat écriture durant $\phi 2$

	NM	*ST*	*NA*	*ND*	*PE*	*S1*	*S2*	*IQ*	*DF*	*WF*	*HO*	*MO*	*BA*	*DA*
queue	-	-	-	-	-	-	-	-	-	-	-	-	-	-
hist.	-	-	-	-	-	-	-	-	-	e	-	-	-	-
bloc	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ext.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
monde	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Tableau 23: Etat écriture durant $\phi 3$

Annexe C

Le Tableau 24 donne l'abréviation et la description de tous les signaux utilisés dans ce rapport.

abrév.	*description*	*traduction anglaise*
BA	adresse sur le bus central	bus address
$C1$	connections de l'arbre d'encodage 1	connections 1
$C2$	connections de l'arbre d'encodage 2	connections 2
CB	commande du module bloc	control bloc
CE	commande des modules extenseurs	control extenders
CH	commande du module historique	control history
CQ	commande du module queue prioritaire	control queue
DA	donnée pour l'initialisation	data
DF	décodage terminé	decoding finished
FM	métrique initiale du noeud-origine	first metric
HO	débordement d'historique	history overflow
ID	identification de l'extenseur	identity
IQ	branche dans la queue du bloc	inside queue
MA	adresse de module	module address
MO	débordement de métrique de noeud	metric overflow
NA	adresse du noeud dans l'historique	node address
ND	profondeur dans le bloc	node depth
NM	métrique accumulée	node metric
NS	passe au prochain état	next state
PE	valeur du compteur de perforation	perforate
PP	patrons de perforation	perforation patterns
RD	initialise décodeur	reset decoder
RN	numérateur du taux de codage	rate numerator
SM	table des métriques de symboles	symbol metrics
$S1$	symbole reçu 1	symbol 1
$S2$	symbole reçu 2	symbol 2
ST	état du codeur	state
WF	écriture terminée	writing finished

Tableau 24: Résumé de tous les signaux