Turbo décodage de codes produits très haut débit sur un circuit FPGA

C. LEROUX, C. JEGO, P. ADDE et M. JEZEQUEL

GET/ENST Bretagne, CNRS TAMCIC UMR 2872, Brest, France
{camille.leroux, christophe.jego, patrick.adde, michel.jezequel}@enst-bretagne.fr

Résumé – Dans cet article, nous proposons une étude de complexité permettant d'optimiser l'architecture d'un décodeur élémentaire BCH en minimisant la dégradation des performances. Ce décodeur simplifié est inclus dans une architecture innovante de turbo décodeur de codes produits sans mémoire d'entrelacement. Un prototype a été réalisé sur un circuit FPGA, le débit d'entrée du turbo décodeur obtenu est de 600Mb/s.

Abstract *– In this paper, we present a complexity analysis for optimising a BCH elementary decoder minimising performance degradation. This low-complexity decoder is duplicated in an innovative interleaving-memory-less block turbo decoder architecture. The resulting block turbo decoder was implemented on a FPGA device with a resulting input data rate is 600Mb/s.*

1. Introduction

Les turbocodes en blocs (TCB) sont une solution alternative aux turbo codes convolutifs introduits en 1993 par C. Berrou [1]. Les TCB ont été proposés par R. Pyndiah [2] en 1994. Le schéma de codage repose sur la concaténation série de codes en blocs linéaires systématiques (codes produits introduits par P. Elias [3]). Le décodage utilise des entrées et des sorties pondérées (décodage EPSP). Afin d'augmenter les débits de transmissions et/ou la qualité de service, les TCB sont l'une des familles de codes correcteurs d'erreurs sélectionnés dans la norme IEEE 802.16 des réseaux métropolitains sans fils (WMAN). De plus, certains systèmes tels que les transmissions optiques ou le stockage de données nécessitent des débits de codage/décodage atteignant 10Gbits/s. Ainsi, les premières générations de réseaux optiques utilisaient des codes Reed-Solomon (RS) comme code correcteur d'erreurs [4]. De tels codes permettent d'atteindre les débits de traitement nécessaires pour un gain de codage approchant les 6dB. Dans un tel contexte, les TCB sont des candidats potentiels. En effet, les codes produits présentent un fort taux de parallélisme et un gain de codage théorique pouvant atteindre 10dB [5][6]. Cependant, concevoir des décodeurs fonctionnant à plusieurs Gbits/s requiert des architectures massivement parallèles ayant une complexité réduite.

Dans les TCB, le décodage itératif d'une matrice de code produit implique un décodage successif des lignes et des colonnes. Les approches classiques nécessitent une grande quantité de mémoire (jusqu'à 75% du circuit) et limitent donc les possibilités d'implantation. De plus, ces architectures n'exploitent pas le parallélisme offert par la matrice de code produit car de nombreux accès mémoires simultanés sont alors nécessaires. Il est toutefois possible de décoder toutes les lignes (colonnes) en parallèle en remplacant la mémoire par une structure moins complexe. Dans [7], une architecture innovante permettant de s'affranchir de cette mémoire d'entrelacement est proposée. La complexité d'une telle architecture dépend principalement de la complexité des décodeurs élémentaires : le nombre de décodeurs EPSP nécessaires est $DEC_n = 2\,it\,n$, où it et n sont respectivement le nombre d'itérations et la taille du code. Dans une telle architecture, la maîtrise de la complexité du décodeur élémentaire devient donc primordiale.

2. Principe de codage et de décodage des TCB

Le concept de code produit est une méthode de construction simple et efficace qui permet de construire des codes puissants à grande distance de Hamming minimale δ en utilisant des codes en blocs linéaires systématiques. Considérons deux codes en blocs linéaires C identiques de paramètres (n,k,δ), où n, k et δ sont respectivement la longueur du code, le nombre de symboles d'informations et la distance de Hamming manimale. Le rendement du code R correspond au rapport entre k et n. On obtient le code produit $P=C\times C$ en plaçant k^2 symboles d'informations dans une matrice de k lignes et de k colonnes, puis en codant chaque ligne et chaque colonne avec le code C comme décrit sur la figure 1. Les n lignes sont des mots de code de C tout comme le sont les n colonnes. Les paramètres du code produit P résultant sont alors $n_p=n^2$, $k_p=k^2$, $\delta_p=\delta^2$ et le rendement devient $R_p=R^2$. La distance du code obtenu (δ_p) est grande devant la distance des codes élémentaires (δ). Des codes puissants sont ainsi obtenus à partir de simples codes en blocs linéaires.

Décoder les codes produits nécessite l'utilisation de décodeurs EPSP qui traitent séquentiellement les lignes et les

colonnes. Le taux d'erreur binaire (TEB) diminue au cours des itérations de décodage.

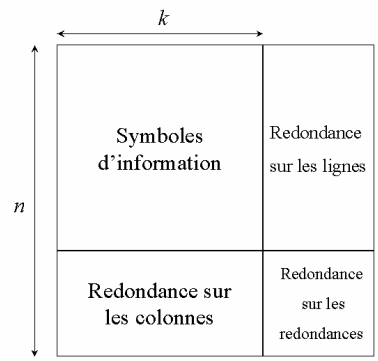

FIG. 1 : Structure d'une matrice de code produit

Ce processus itératif constitue le turbo décodage des codes produits. Chaque décodeur doit calculer l'information souple $[R']_{it+1}$ à partir de l'information souple issue du canal $[R]$ et de l'information souple de la demi-itération précedente $[R']_{it}$. L'algorithme de décodage Chase-Pyndiah [8][2] d'un code BCH est résumé ci-dessous :

1- Recherche des Lr bits les moins fiables et calcul du syndrome S_0 de $[R']_{it}$,
2- Génération des Tv vecteurs de test obtenus en inversant certains des bits les moins fiables,
3- Décodage binaire de chaque vecteur de test en utilisant S_0 et les bits les moins fiables,
4- Pour chaque vecteur de test, calcul du carré de la distance Euclidienne (métrique) $M_i(i=0,\ldots,Tv_n-1)$ entre $[R']_{it}$ et les vecteurs de test considérés.
5- Sélection du mot décidé (DW) ayant la plus petite distance avec $[R']_{it}$ et choix des Cw mots concurrents les plus proches de $[R']_{it}$.
6- Calcul de la fiabilité $[F]_{it}$ pour chaque symbole de DW,
7- Calcul de l'information extrinsèque $[W]_{it}=[F]_{it}-[R']_{it}$ pour chaque symbole de DW.
8- Ajout de l'information extrinsèque (pondérée par α_{it}) au mot reçu du canal, $[R']_{it+1}=[R]_{it}+\alpha_{it}[W]_{it}$

Le coefficient α_{it} permet de pondérer les décisions de décodage lors des premières itérations. Les paramètres Lr, Tv, and Cw ont une influence notable sur les performances de décodage et sur la complexité.

3. Analyse de la complexité et des performances d'un décodeur EPSP BCH

Des estimations de surface ont été réalisées avec l'outil de synthèse logique Design Compiler de Synopsys pour la technologie ASIC CMOS 0.09µm de chez ST Microelectronics. La fréquence de fonctionnement est fixée à $f=500MHz$. Les performances de décodage ont été simulées grâce à un modèle algorithmique de turbo décodeur pour les codes BCH(16,11)² et BCH(32,26)² après 6 itérations. Bien que la cible finale soit un circuit FPGA, les estimations de complexité ont été faites sur une cible ASIC. C'est pourquoi, les estimations de complexité sont exprimées en équivalent porte (surface équivalente à une porte NAND à deux entrées). En effet, ce type d'estimations, pour un circuit FPGA dépendent trop fortement de la cible considérée et il est, de plus, difficile de trouver un équivalent porte pour un circuit FPGA.

L'information souple utilisée au sein du décodeur est quantifiée sur Q bits (1 bit de signe et $Q-1$ bits de fiabilité). Le décodeur EPSP, décrit sur la figure 2, est constitué de trois étages fonctionnant en mode pipeline (étage de *réception*, *traitement* et *émission)*. Chaque étage traite n symboles en n périodes d'horloges. La latence résultante est alors égale à $2n$ cycles d'horloges.

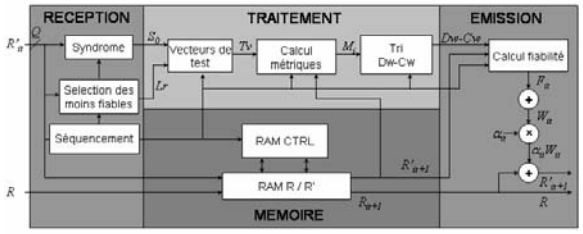

FIG. 2 : Architecture d'un décodeur EPSP BCH

Le décodeur étudié est composé de 12 blocs distincts. Des synthèses préliminaires ont montré que quatre de ces douze blocs nécessitent à eux seuls 75% de la surface du décodeur. Nous avons donc focalisé notre étude sur ces 4 blocs critiques. Ainsi, il apparaît que 5 paramètres ont une influence directe sur les complexités. Il s'agit des paramètres algorithmiques Lr, Tv, Cw ainsi que de la taille du code n et du nombre de bits de quantification des données Q.

Pour une taille de code fixée n, nous définissons un ensemble p_i de paramètres :

$$p_i = \{ Q_i, Cw_i, Tv_i, Lr_i \}. \qquad (1)$$

L'objectif de notre analyse est d'obtenir une estimation fiable de la complexité du décodeur EPSP pour n'importe quel ensemble p_i en extrapolant la complexité mesurée sur les 4 blocs critiques à l'ensemble du décodeur.

$C'_{pi}(n)$ est la complexité mesurée et exprimée en équivalent porte des 4 blocs critiques et $C_{pi}(n)$ la complexité estimée du décodeur EPSP pour un jeu de paramètre p_i. Puisque les 4 blocs critiques représentent 75% de la surface, nous avons,

$$C'_{p0}(n) = 0.75\, C_{p0}(n) \qquad (2)$$

De plus, nous supposons que,

$$C_{p0}(n) - C_{pi}(n) \approx C'_{p0}(n) - C'_{pi}(n) \qquad (3)$$

Cette hypothèse a été vérifiée lors des synthèses logiques a posteriori. En combinant (2) et (3), nous obtenons :

$$C_{pi}(n) = 1/3\, (C'_{p0}(n)) + C'_{pi}(n), \qquad (4)$$

tel que $C'_{pi}(n) = 462(Cw_i-1) + 261(\log(n)-4) + 183(Q_i-4) + 55.(Tv_i-4) + 46(Lr_i-2) + 1019$ (5)

et, $p_0=\{Cw_0=3,Q_0=5,Tv_0=16,Lr_0=5\}$.

$C'_{pi}(n)$ a été obtenue en synthétisant des descriptions génériques pour les 4 blocs critiques. L'utilisation de régressions linéaires multiples a ensuite permis d'extraire cette expression empirique. La précision de ce modèle a été testée a posteriori. L'erreur maximale et l'erreur moyenne entre le modèle et à l'issue de la synthèse logique, sont respectivement de 8% et 2.5%. La tableau I donne quatre exemples d'estimations de complexité pour des codes de taille $n=16$ et 32. Ce modèle permet d'obtenir rapidement une estimation de la surface du décodeur EPSP BCH pour n'importe quel jeu de paramètres sur une cible ASIC. En fonction des contraintes de surface et de performance de l'application, l'exploration des solutions architecturales est facilitée.

TAB. 1 : Complexité (nombre de portes logiques) pour différents jeux de paramètres

p_i	n	Q	Lr	Tv	Cw	$C'_{pi}(n)$	$C_{pi}(n)$
p_0	**16**	**6**	**5**	**16**	**3**	**3107**	**4143**
p_1	16	4	2	4	1	1019	2055
p_0	**32**	**6**	**5**	**16**	**3**	**3368**	**4491**
p_1	32	4	2	4	1	1742	2865

Afin de valider le bon fonctionnement de l'architecture ainsi que les performances du turbo décodeur, nous avons implanté le turbo décodeur sur un circuit FPGA. Dans le cas de notre prototype, les contraintes sont principalement liées aux ressources disponibles sur le FPGA (30000 éléments logiques). Une taille de code n=32 nécessite 32 décodeurs par demi-itération. Même pour des décodeurs élémentaires très simplifiés, une itérations de décodage requiert une quantité de ressources supérieures à celles disponibles sur la cible FPGA choisie. C'est pourquoi, nous avons retenu une taille de code $n=16$ (code BCH(16,11)), et un ensemble de paramètres $p_1=\{Cw=1, Q=4, Tv=4, Lr=2\}$. Le gain en complexité, pour le jeu de paramètres retenu, est alors de 50% (par rapport à p_0) pour une dégradation de performance de seulement 0.3dB à un taux d'erreur binaire de 10^{-4}.

4. Implantation d'un turbo décodeur de codes produits très haut débit

Le décodeur utilisant le jeu de paramètres p_1 a été dupliqué pour réaliser un turbo décodeur entièrement parallèle. Un premier prototype de turbo décodeur inclus dans une chaîne de communications numériques très haut débit a été réalisé sur une plate-forme de développement XUP de base [9]. Deux FPGA (Xilinx Virtex II-Pro XC2VP30) communiquent via un bus SATA à très haut débit. Sur le FPGA émetteur, des données pseudo-aléatoires sont codées puis bruitées avant d'être envoyées à un débit de 2.4Gbits/s sur le FPGA récepteur qui contient le turbo décodeur. La chaîne de communication utilise deux horloges de fréquences $f_0=37.5MHz$ et $f_1=75MHz$. L'architecture de la chaîne de communication est détaillée sur la Figure 3.

Des données pseudo-aléatoires sont tout d'abord envoyées de manière parallèle à chaque cycle d'horloge (f_0). Ce générateur est composé de 11 LFSR ayant des polynômes générateurs différents assurant ainsi une bonne décorrélation entre les données générées. Un codeur BCH(16,11)² traite les 11 données en parallèle. L'architecture innovante de ce codeur parallèle évite l'utilisation d'un plan mémoire entre le codage des lignes et le codage des colonnes [8] : n codeurs séquentiels classiques sont mis en séries avec un codeur parallèle. 256 données (équivalent à une matrice de code produit 16x16) sont ainsi générées en 16 cycles d'horloges (f_0). L'émulation d'un canal Gaussien est réalisée dans le bloc canal AWGN dans lequel 16 échantillons décorrélés de bruit blanc Gaussien sont générés et additionnés aux données codées [11]. Chaque échantillon de sortie correspond à un vecteur de 4 bits, ainsi, 64 bits sont envoyés au cours d'une période d'horloge (f_0). Le rapport signal à bruit (SNR) est fixé pour des valeurs comprises entre 0 et 15.75dB avec un pas de 0.75dB. Le module protocole de transmission Aurora gère les signaux de contrôle pour la communication avec la carte réceptrice. Ce module reçoit 64 données en 2 cycles d'horloges et envoie 32 données à chaque cycle d'horloge (f_1). Le débit sur le bus de communication est donc de 2.4Gbits/s.

Le module de réception Aurora reçoit les données à 2.4Gbits/s et envoie, au turbo-décodeur, 64 bits tous les 2 cycles d'horloges (f_1). Le turbo-décodeur est composé de deux blocs de 16 décodeurs EPSP (16 décodeurs par demi-itérations) connectés par deux réseaux Ω. Des détails sur la structure des réseaux Ω sont disponibles dans [7][12]. Chaque décodeur reçoit 4 bits à chaque cycle d'horloge (f_0). Le même générateur pseudo-aléatoire est implanté en réception. Il génère les même données qu'en émission afin de pouvoir comparer les données avant et après le décodage. Enfin, un module de calcul de TEB est implanté pour mesurer le taux d'erreur en sortie du décodeur. Un minimum de 1000 erreurs est garantie avant d'afficher le résultat sur un module LCD de la carte. La valeur minimale de TEB atteignable de manière fiable est 10^{-9}.

L'objectif de ce premier prototype est de montrer qu'un turbo décodeur de code produit peut effectivement fonctionner à haut débit sans mémoire d'entrelacement sur un circuit FPGA. Le turbo décodeur traite les données d'entrées à 600Mbits/s avec une fréquence de fonctionnement de 37.5MHz. En effet, le protocole de communication fonctionne uniquement à des fréquences multiples de 37.5MHz. Cette contrainte limite donc les choix de fréquences. Cependant, le turbo décodeur en lui-même, peut fonctionner jusqu'à 70MHz sur la cible considérée, ce qui correspond à un débit d'entrée de 1.12Gbits/s. Le débit de sortie $D_{out} = P.f.R$ tient compte du rendement du code. P représente le taux de parallèlisme ($max(P)=n$), et f la fréquence de fonctionnement du décodeur. Dans notre cas, $P=16$, $f=f_0=37.5MHz$ et $R=0.473$, le débit de sortie est alors de $284Mb/s$.

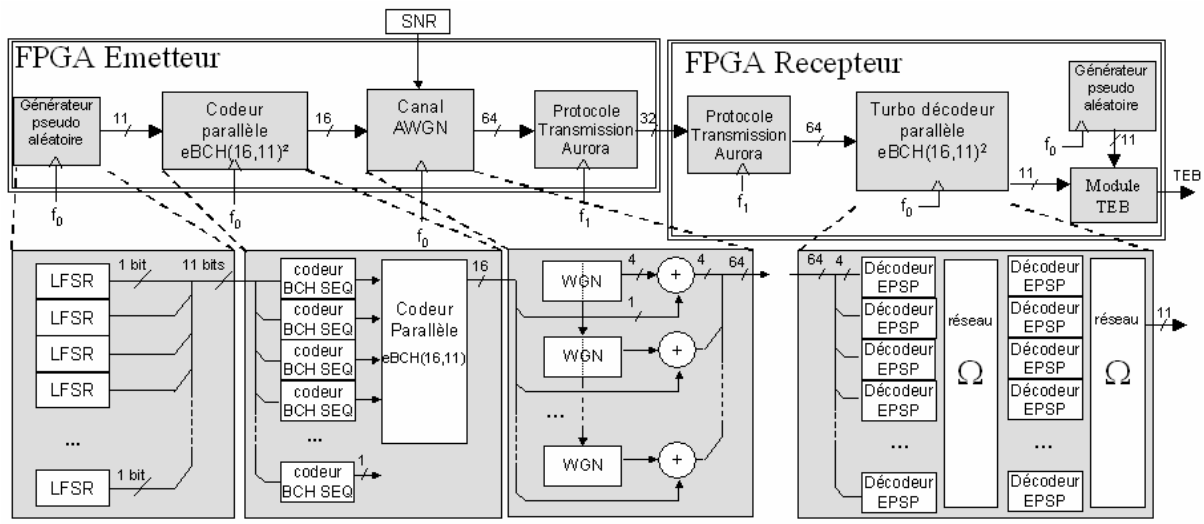

FIG. 3 : Implantation d'un turbo décodeur dans une chaîne de communications mumériques très haut-débit

Différentes solutions existent pour augmenter ce débit; la plus simple serait de prendre un code de plus grande taille (pour augmenter *P*) et de plus grand rendement *R*. En considérant un turbo décodeur de code produit BCH(32,26)² fonctionnant à 70MHz, nous obtenons des débits d'entrée et de sortie respectivement de 2.24Gbits/s et de 1.48Gbits/s. Enfin, utiliser des décodeurs EPSP qui traitent plusieurs données pendant un cycle d'horloge et/ou étendre notre étude aux codes M-aires RS [10][11] permettent encore une fois d'augmenter le débit de traitement pour une complexité raisonnable. La cible utilisée limite l'implantation à deux demi-itérations. Un turbo décodeur comprenant 6 itérations est réalisable sur un circuit FPGA Virtex 4.

5. Conclusion

Dans cet article, nous avons montré comment implanter un turbo décodeur parallèle de code produit très haut débit sur un circuit programmable. Une étude fine de complexité a permis de réduire de 50% la complexité du turbo-décodeur en limitant la dégradation à 0.3dB. En utilisant une famille de circuits FPGA plus récente et un protocole de communication plus approprié, le turbo décodeur peut atteindre des débits supérieurs au Gbit/s. Plusieurs solutions pour augmenter le débit ont d'autre part été évoquées; parmi celles-ci, certaines sont actuellement à l'étude.

Références

[1] C. Berrou et al, *"Near Shannon limit error-correcting coding and decoding : Turbo-codes (1),"* IEEE Int. Conf. on Comm.ICC' 93, vol 2/3, May 1993, pp. 1064-1071.

[2] R. Pyndiah, *"Near optimum decoding of product codes: Block Turbo Codes"*, IEEE Trans. on Comm., vol 46, n° 8, August 1998, pp. 1003-1010.

[3] P. Elias, *"Error-free coding"*, IRE Trans. on Inf. Theory, vol. IT-4, pp. 29-37, Sept. 1954.

[4] K. Azadet et al, *"Equalization and FEC techniques for optical transceivers"*, Solid-State Circuits, IEEE Journal of Volume 37, Issue 3, March 2002, pp. 317-327.

[5] O. Ait Sab, O. V. Lemaire, *"Block turbo code performances for long-haul DWDM optical transmission systems"*, Optical Fiber Communication Conference,Volume 3, March 2000 pp. 280-282.

[6] T. Mizuochi, *"Recent Progress in Forward Error Correction for Optical Communication Systems"*, IEICE Transactions on Communications, Volume E88-B, Number 5, May 2005.

[7] C. Jego, P. Adde, C. Leroux, *"Full-parallel architecture for turbo decoding of product codes"*, Electronics Letters Volume 42, Issue 18, 31 August 2006 pp. 55 – 56.

[8] D. Chase, *"A class of algorithms for decoding block codes with channel measurement information"*, IEEE Trans. Inform. Theory, vol IT-18, Jan. 1972, pp 170-182

[9] http://www.xilinx.com/univ/XUPV2P/Documentation/XUPV2P_User_Guide.pdf

[10] J.Cuevas, P.Adde, S.Kerouedan, *"Very powerful block turbo codes for high data rates applications"*, 3rd International Symposium On Turbo Codes & Related Topics, Brest, France, 1-5 septembre, 2003. p 251-254.

[11] E.Piriou, C. Jego, P. Adde, R. Le Bidan, M. Jezequel, *"Efficient architecture for Reed Solomon block turbo code"*, ISCAS 2006 : International Symposium on Circuits and Systems, Kos, Greece, May 21-24, 2006. p. 3682-3685.

[12] D.H. Lawrie,: *"Access and alignment of data in an array processor"*, IEEE Trans. Comput., 1975, pp. 1145–1155 Electronics Letters 31st August 2006 Vol. 42 No. 18