Conception d'un décodeur BCH (30,19,6) à entrées et sorties pondérées : application au turbo décodage

Patrick ADDE, Sylvie KEROUEDAN, Jean-René INISAN

ENST Bretagne, Technopôle Brest Iroise, BP 832,

29285 BREST Cedex , France

Patrick.Adde@enst-Bretagne.fr, Sylvie.Kerouedan@enst-Bretagne.fr, JR.Inisan@enst-Bretagne.fr

Résumé – Cet article présente la conception d'un décodeur BCH (32,19,6) à entrées et sorties pondérées corrigeant 2 erreurs. La cible technologique choisie, circuit intégré programmable (FPGA XILINX), ainsi que sa faible complexité (20000 portes), a permis son insertion dans une maquette de turbo décodage qui autorise des mesures de taux d'erreurs de l'ordre de 10^{-9}. Elle valide l'utilisation de turbo code produit obtenu à partir de codes BCH étendus de rendement proche de 0,5 et pour des blocs de la taille d'une cellule ATM.

Abstract – This paper presents the design of a BCH (32,19,6) decoder with soft inputs and soft outputs. The technological target, a programmable integrated circuit (FPGA XILINX) and its low complexity (20,000 gates) has permit its insertion in turbo decoding breadboard which allows BER (bit error rate) measurements down to 10^{-9}. It validates the use of product turbo codes obtained from extended BCH codes with code rates close to 0.5 and for small data blocks such as ATM cells.

1. Introduction

Le début des années 90 a vu la naissance d'une nouvelle technique de code correcteurs : les turbo codes. Ce concept a été introduit en 1993 par C. Berrou [1]. Le schéma de codage proposé repose sur la concaténation parallèle de deux codes convolutifs séparés par un entrelacement non uniforme. Le décodage utilise un processus itératif basé sur des décodeurs à entrées et sorties pondérées. Ces principes, connus sous le nom de turbo code convolutif TCC, ont des performances inégalées à ce jour et proches de la limite de Shannon : ils commencent à être utilisé dans les systèmes de communications numériques.

Une solution alternative aux TCC sont les turbo codes en blocs TCB proposés en 1994 par R. Pyndiah [2]. Le schéma de codage est basé sur la concaténation série de code en blocs (code produit, introduit en 1954 par P. Elias [3]) et le décodage reprend le processus itératif obtenu à partir de décodeurs élémentaires à entrées et sorties pondérées. La concaténation série garantit une grande distance de Hamming minimale (16, 24, 36 voire plus) et ce pour des blocs de données relativement petits. L'information extrinsèque qui joue un rôle important dans le turbo décodage est calculée non seulement pour les bits de données mais aussi pour les bits de redondance à chaque itération. Les TCB sont des solutions très attractives lorsque le rendement de codage est élevé (jusqu'à 0,95), lorsque les blocs de données sont petits (jusqu'à 100) et si le taux d'erreurs visé est faible.

Dans ce papier, nous présentons la conception d'un décodeur BCH étendu à entrées et sorties pondérées corrigeant deux erreurs. Son implantation dans un circuit ASIC FPGA 20000 portes montre sa faible complexité de réalisation. Il a été par ailleurs utilisé dans une maquette de turbo décodage permettant de mesurer des taux d'erreurs binaires TEB de l'ordre de 10^{-9} sur un canal gaussien pour un rendement de code variable (de 0,43 à 0,66) et pour des tailles de blocs correspondant à celle des cellules ATM.

2. Conception du décodeur BCH(30,19) à entrées et sorties pondérées

2.1 Décodage des codes BCH à entrées et sorties pondérées

Les décodeurs travaillent en décision douce (soft decision). L'entrée est constituée d'échantillons analogiques (appelés symboles) codés sur 5 bits : 4 bits de fiabilité et un bit de signe. Le bit de signe représente la valeur binaire du symbole ('0' ou '1'), la fiabilité le niveau de confiance avec lequel ce bit est reçu (une fiabilité forte

signifie que ce bit a peu de chance d'être erroné).

L'algorithme implanté sur le silicium utilise l'algorithme de décodage de type Chase [4] et l'algorithme de pondération de R. Pyndiah [2][5]. Pour ce dernier, c'est la version simplifiée qui a été choisie, celle qui prend en compte un seul concurrent dans le calcul de la pondération. Il délivre en sortie la séquence binaire la plus probablement émise. Il comporte 11 étapes [6][7] :

-réception du mot d'entrée R'_k,

-recherche des 4 symboles les moins fiables,

-construction des 8 vecteurs de test,

-calcul des syndromes S_1 et S_3 du premier vecteur de test,

-calcul des syndromes S_1' et S_3' des 7 autres vecteurs de test,

-calcul des métriques de chacun des vecteurs de test et choix du vecteur ″décidé″ et ″concurrent″,

-calcul de la nouvelle fiabilité pour chacun des symboles du mot décidé,

-calcul de l'information extrinsèque,

-addition avec le mot reçu du canal.

Les vecteurs de test constitués de 32 symboles sont au nombre de 8. Le premier vecteur C_0 est simplement constitué des bits de signe du mot reçu. Les 7 autres vecteurs (C_1 à C_7) sont construits à partir de celui-ci, en inversant les positions des symboles les moins fiables (PMF_i, i=1,..,4) :

-$C_{1\ à\ 4}$ où le symbole $PMF_{1\ à\ 4}$ est inversé,

-$C_{5\ à\ 7}$ où les 2 symboles PMF_1 et PMF_i (i=2 à 4) sont inversés.

Les syndromes de chacun de ces vecteurs sont calculés de la manière suivante :

$S_1'=S_1 \oplus (1^{ere}\text{ position inversée}) \oplus (1^{ere}\text{ position inversée})$

$S_3'=S_3 \oplus (1^{ere}\text{ position inversée})^3 \oplus (1^{ere}\text{ position inversée})^3$

où S_1 et S_3 sont les syndromes de C_0 et les positions inversées sont repérées dans le corps de Galois $GF(2^5)$.

S_1' et S_3' permettent le décodage algébrique de chaque vecteur de test et donnent les positions PC_1 et PC_2 à corriger : le calcul des syndromes après corrections vérifie que le mot obtenu est un mot de code.

Le calcul des distances euclidiennes entre chacun des vecteurs de test et le vecteur d'entrée ne prend en compte que les positions où les symboles différent :

$$M_L = \sum_K \left\| R'_K - C_L^K \right\|^2 = \sum_{K=0}^{N-1} \left| R'_K C_L^K \right|$$

pour les bits K tels que $\text{signe}\left(C_L^K\right) \neq \text{signe}\left(C_0^K\right)$

2.2 Structure fonctionnelle et séquencement

Après l'analyse des différentes fonctions apparaissant dans l'algorithme de décodage, le schéma fonctionnel du décodeur a été défini [8].

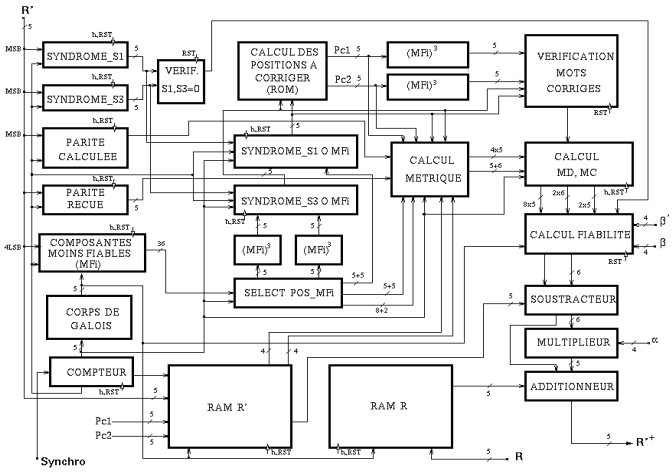

FIG.1 : schéma bloc du décodeur

Nous avons distingué, lors de la conception de ce schéma, différentes parties :

- la partie séquentielle en entrée, qui regroupent tous les blocs dont le calcul progresse au fur et à mesure de l'arrivée de chaque symbole du mot reçu (syndromes S_1 et S_3, parité et positions des symboles les moins fiables);

- la partie non séquentielle où les calculs sont effectués après réception complète du mot, et ne nécessitent donc aucune synchronisation vis-à-vis de l'arrivée des symboles (décodage binaire, calcul des métriques et choix des vecteurs décidé et concurrent). Ce bloc est relié à l'extérieur à une fonction combinatoire (ROM) qui donne les positions des symboles corrigés en fonction des syndromes;

- la partie séquentielle en sortie, qui regroupent les blocs travaillant au fur et à mesure de l'émission des nouveaux symboles (calculs de la fiabilité de chaque symbole et de son information extrinsèque, construction du mot suivant) ;

- la partie mémoire, constituée de deux RAM, insérée entre chaque demi-itération.

Le traitement de chaque mot est donc globalement divisé en trois parties : une partie séquentielle en réception, une partie non séquentielle de traitement du mot et une partie séquentielle en émission.

La partie "traitement du mot" consiste en un traitement de chaque vecteur de test : calcul des syndromes et de la métrique de chaque vecteur, calcul des positions à corriger sur chaque vecteur.

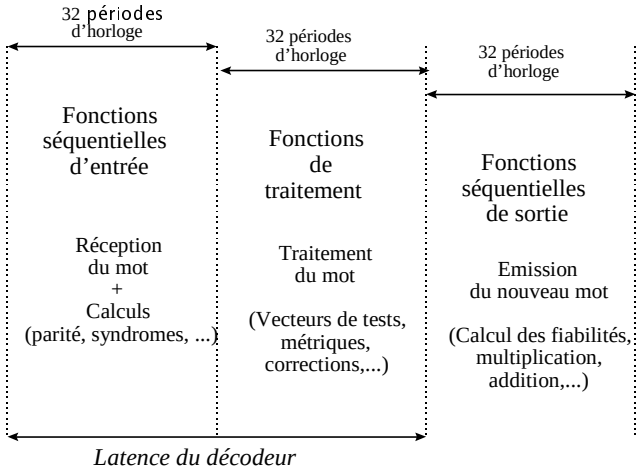

FIG. 2 : découpe temporelle du circuit

Nous avons décidé de réaliser ce traitement en 32 périodes d'horloge. Ainsi, le décodeur travaillant sur 8 vecteurs, 4 périodes d'horloge sont allouées à chaque vecteur.

A la fin de ce traitement, on peut sélectionner les deux vecteurs donnant les mots "Décidé" et "Concurrent", qui nous permettent de calculer la fiabilité à émettre en sortie de la demi-itération.

La latence du décodeur, c'est-à-dire le temps qui s'écoule entre la réception du premier symbole du mot R_k' et l'émission du premier symbole du mot R_k'$^+$, est de 64 périodes d'horloge.

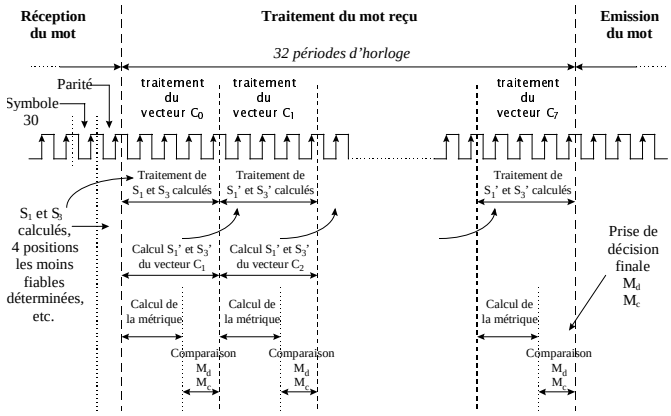

FIG. 3 : traitement de chaque mot.

2.3 Implantation du décodeur

Après la description de chaque partie en langage VHDL et synthèse, le décodeur a été implanté dans un circuit FPGA Xilinx 20000 portes. Après optimisation en vitesse, les résultats de l'implantation ont donné :

- 598 blocs logiques programmables CLB (contient deux générateurs de fonctions logiques à 4 entrées F et G, un générateur de fonction permettant de combiner les sorties des blocs F et G, deux bascules D) sur 784 de disponibles (76%),
- période d'horloge minimale 172,5ns ce qui correspond à une fréquence maximale de 5,8Mhz (le débit des symboles est alors de 2,9Mbits/s).

Parmi les blocs demandant beaucoup de ressources, on peut citer : celui qui recherche les symboles les moins fiables (110 CLB), celui qui calcule les métriques (145 CLB) et celui qui sélectionne le mot décidé e le concurrent (121 CLB).

Des simulations après placement routage ont permis de valider complètement la conception par rapport aux résultats obtenus à l'aide d'un modèle en langage C.

Le décodage binaire des vecteurs de test, obtention des positions à corriger à partir des syndromes S_1' et S_3', est réalisé avec une ROM extérieure de 10kbits.

3. Maquette de prototypage et premières mesures

Un prototype avait été développé à l'ENST de Bretagne en 1997 [6][7] avec comme principales caractéristiques :

- un code produit avec comme codes élémentaires deux codes BCH étendus (32,26,4),
- un décodage séquentiel avec 4 itérations,
- l'algorithme Chase-Pyndiah avec 8 vecteurs de test et un seul vecteur concurrent.

Il avait permis de valider les nouveaux concepts et nouveaux algorithmes mis en œuvre dans les turbo décodeurs de code produit, construits avec des codes BCH étendus.

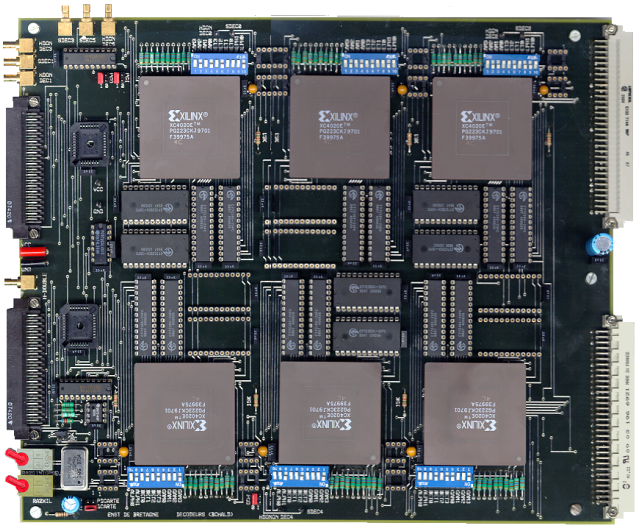

FIG. 4 : Maquette contenant 3 itérations de décodage.

Un second prototype a été construit en 1998. La taille des blocs de données peut varier entre 441 et 676 avec un rendement de code proche de 0,5 (de 0,43 à 0,66). Le codeur est programmable : il est possible d'utiliser en lignes

ou colonnes soit le code BCH étendu (32,26,4) ou le code BCH étendu (32,21,6) et de réduire le nombre de lignes et de colonnes dans la matrice (de 0 à 7) pour les raccourcir. A la sortie du codeur, un système de modulation/ démodulation à 4 états de phase sur canal Gaussien est modélisé. Les symboles bruités sont générés par un programme en C, stockés dans des RAM de 2x2Mbits et lus aléatoirement [9].

Des mesures ont été faites avec un code produit BCH(30,24,4)⊗BCH(30,19,6) obtenu par raccourcissement des codes BCH(32,26,4) et BCH(32,21,6). La taille des blocs de données est donc 456 (57 octets) pour un rendement de 0,51. Un débit de 2,5Mbits/s nous a permis de mesurer un TEB de 2.10^{-9} à Eb/N_0=4dB, après six itérations de turbo décodage. Cette courbe est comparée ci-après avec la courbe théorique obtenue avec le gain asymptotique.

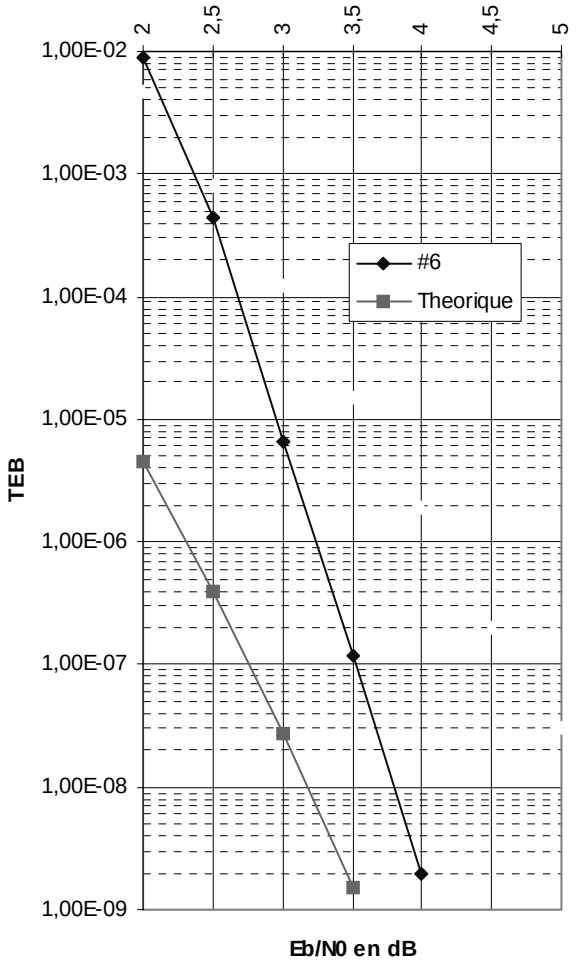

FIG. 5: Performance du code produit BCH(30,24,4)⊗BCH(30,19,6) et comparaison avec la courbe théorique obtenue avec le gain asymptotique

On peut noter un écart de 0,5 dB entre ces 2 courbes à faible taux d'erreurs.

4. CONCLUSION

Il est montré dans ce papier que l'intégration d'un décodeur de code BCH étendu corrigeant 2 erreurs est possible dans un circuit de faible complexité et de faible coût. On peut alors envisager l'intégration de turbo décodeur (4, 6 ou 8 itérations) dans des ASICs dont la surface silicium n'est pas prohibitive. Les résultats obtenus avec des tailles de blocs de l'ordre de celle des cellules ATM et des rendements proches de 0,5 montrent que l'utilisation des TCB est adapté pour ce genre d'application, surtout quand les TEB requis sont faibles

Références

[1] C. Berrou, A. Glavieux and P. Thitimajshima, "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, A. Glavieux, A. Picart and S. Jacq, "Near optimum decoding of product codes, "in proc. of *IEEE GLOBECOM '* 94 *Conference*, vol. 1/3, Nov.- Dec. 1994, San Francisco, pp. 339-343 .

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

[4] 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.

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

[6] P. Adde, R. Pyndiah, O. Raoul and J.R. Inisan, "Block turbo decoder design," *Int. Symposium on turbo codes and related topics*, Brest, Sept. 1997, pp.166-169.

[7] P. Adde, R. Pyndiah, J. R. Inisan et Y. Sichez, "Conception d'un turbo décodeur de code produit", *GRETSI'97*, Grenoble, pp. 1169-1172, septembre 1997.

[8] P. Adde, R. Pyndiah, O. Raoul, "Performance and complexity of block turbo decoder circuits," *Third International Conference on Electronics, Circuits and System ICECS'96*, pp. 172-175, 13-16 October 1996 - Rodos, Greece.

[9] M. Jézéquel, C. Berrou, J. R. Inisan and Y. Sichez, "Test of a Turbo-encoder/Decoder", *Turbo Coding Seminar*, Lund, Sweden, pp. 35-41, 28-29 August 1996.