

Rapport sur le mémoire de Mathieu LEONARDON

" Décodage de codes polaires sur des architectures programmables "

en vue de l'obtention du grade de Docteur de l'Université de Bordeaux,
et de Philosophie Doctor de l'Ecole Polytechnique de Montréal

Contexte/Problématique

Les recherches présentées par Mathieu Léonardon dans son mémoire de thèse concernent les codes correcteurs d'erreurs et plus précisément les codes polaires. L'auteur s'intéresse à l'implémentation logicielle des algorithmes de décodage de codes polaires et à la conception d'architectures programmables spécialisées pour leur exécution dans l'objectif d'une meilleure efficacité énergétique en comparaison d'une exécution sur un processeur généraliste.

Ces travaux ont été effectués dans le cadre d'une thèse en cotutelle entre le laboratoire IMS, Université de Bordeaux et le département Génie Electrique de l'Ecole Polytechnique de Montréal.

Contenu du mémoire

Le mémoire, rédigé en français, fait 110 pages et est composé d'une introduction générale, de 4 chapitres résumés ci-après et se termine par une partie conclusions et perspectives.

Chap.1 : Les codes polaires

Chap. 2 : Décodeur logiciel de codes polaires à liste

Chap. 3 : Conception d'un processeur par la spécialisation de son architecture

Chap. 4 : Conception d'un processeur par la description de son architecture

Chapitre 1

Ce premier chapitre introduit le contexte applicatif de la thèse : les codes polaires, utilisés en codage canal. Ceux-ci sont d'abord introduits ainsi que le principe de polarisation. Les principaux algorithmes de décodage de codes polaires sont ensuite abordés. Sont ainsi présentés l'algorithme originel, l'algorithme SC (*Successive Cancellation* pour annulation successive), l'algorithme SC-LIST (*Successive Cancellation List*) qui permet d'améliorer les performances de décodage pour des codes de petite taille, ainsi que des évolutions basées sur l'utilisation d'une concaténation CRC-code polaire (*Cyclic Redundancy Check*) en vue d'améliorer les performances de décodage. Le principe de deux algorithmes itératifs à sortie souple est rapidement évoqué. S'en suit un paragraphe qui permet d'apprécier les performances de décodage de ces algorithmes de manière comparative. La méthode de l'élagage de l'arbre de décodage, qui permet de simplifier les algorithmes de décodage et qui sera utilisée par la suite pour implémenter les décodeurs, est ensuite présentée. Le chapitre se termine par un tableau qui résume les principales caractéristiques des différents algorithmes. Ce chapitre introductif est bien écrit. Il permet de rentrer rapidement et efficacement dans le contexte de la thèse, sans avoir trop de détails qui perdraient le lecteur mais en ayant les informations pertinentes qui permettent de bien situer le travail dans son contexte (différents algorithmes existants, choix de l'algorithme de décodage) et d'appréhender la suite du mémoire.

Chapitre 2

Disposer d'un décodeur logiciel pour le décodage de codes polaires est primordial pour les nouvelles générations de réseaux mobiles (5G par exemple) où le principe de virtualisation semble incontournable. Aussi, ce chapitre est dédié à la réalisation d'un tel décodeur, basé sur l'algorithme de décodage à liste (SCL). Une première partie du chapitre dresse un état de l'art des décodeurs logiciels. Deux techniques sont classiquement utilisées pour améliorer les performances temporelles des décodeurs : la vectorisation, exploitable avec des processeurs SIMD, et le déroulage de code qui permet de réduire le contrôle. Le

décodeur logiciel visé dans ce travail se veut adaptable à différents contextes. En ce sens, deux propriétés sont définies : généricité et flexibilité. La généricité du décodeur est sa capacité à supporter une très grande variété de codes polaires tandis que la flexibilité du décodeur est sa capacité à configurer l'algorithme de décodage. On notera à ce stade que le déroulage complet de code n'est pas compatible avec un décodeur générique et flexible. Par contre, la vectorisation ne pose pas de problème. Quelques expérimentations sont présentées pour montrer l'intérêt de travailler par exemple en format virgule fixe 8 bits plutôt qu'en format flottant. Trois améliorations algorithmiques ou d'implémentation sont appliquées au décodeur générique et flexible proposé pour améliorer le débit (l'augmenter) et la latence (la diminuer). L'auteur mentionne à plusieurs endroits qu'il s'agit de contributions originales. Il serait bon de préciser exactement ce qui est original, car deux des améliorations font au moins partiellement référence à des travaux antérieurs. Enfin, le décodeur proposé est confronté aux décodeurs logiciels de l'état de l'art. Si les latences obtenues sont plus élevées, les performances en terme de débit sont proches de celles des décodeurs de l'état de l'art, ce qui est particulièrement intéressant sachant que la solution proposée présente l'avantage d'être générique et flexible à l'inverse des autres décodeurs. Par ailleurs, en utilisant l'algorithme complètement adaptatif de décodage à liste aidé d'un CRC (FASCL) avec élagage de l'arbre, les débits obtenus sont supérieurs à ceux des décodeurs logiciels existants. Des résultats d'exécution sur des processeurs embarqués Cortex-A sont également présentés avec notamment l'énergie consommée par bit d'information. On notera que le décodeur logiciel proposé est intégré au sein d'une suite logicielle libre et que le code source est ainsi disponible pour la communauté.

Chapitre 3

Les décodeurs logiciels étant typiquement destinés à être exécutés sur des processeurs généralistes, si tel est le cas, la consommation énergétique du décodeur est alors relativement importante. Les chapitres 3 et 4 adressent donc l'étude d'architectures spécifiques programmables de type ASIP (processeurs à jeu d'instruction spécifique à l'application) visant une moindre consommation tout en ayant des débits de décodage raisonnables (comparativement à une exécution sur processeur généraliste). Dans ce chapitre 3, l'approche considérée est la spécialisation d'un processeur «de base», à savoir l'utilisation d'une architecture de processeur «de base» (au sens processeur classique minimaliste) à laquelle sont ajoutées des unités matérielles spécifiques aux besoins des applications à exécuter. Le jeu d'instruction du processeur est alors étendu afin de tirer parti de ces unités spécifiques. Dans le cadre de ce travail, les processeurs ciblés sont des Xtensa de Tensilica (processeurs de type RISC). Un environnement de conception très complet est disponible pour développer ce type de processeurs. Une première partie du chapitre 3 présente le concept des ASIP. La manière dont est configuré le processeur de base est ensuite présentée. Le travail qui a été effectué s'inscrit dans une démarche de conception de type adéquation algorithme-architecture avec des choix et des compromis liés soit à l'architecture Xtensa, soit à l'algorithme de décodage. A ce propos, quelques choix de valeurs mériteraient d'être justifiés. Une comparaison des latences, débits, consommation énergétique entre le processeur proposé et un processeur embarqué et un processeur x86 est faite qui montre l'intérêt de la solution proposée, en particulier en terme de consommation. On peut regretter que le choix des processeurs utilisés pour établir des comparaisons ne soit pas vraiment argumenté.

Chapitre 4

L'approche considérée dans ce chapitre vise à obtenir un autre compromis consommation / débits de décodage en utilisant un processeur toujours programmable mais avec un degré de spécialisation plus fin qu'au chapitre précédent. Un des constats fait au chapitre 3 était qu'une partie non négligeable du temps d'exécution était imputable aux échanges de données, nombreux lors d'un décodage de codes polaires, et ce malgré l'ajout d'unités spécifiques permettant de limiter les pénalités liées à ces échanges. Dans ce chapitre, la structure et les unités élémentaires du processeur sont entièrement définies par l'utilisateur. L'architecture du processeur repose sur l'architecture TTA (*Transport Triggered Architecture*) qui présente l'avantage de pouvoir exploiter le parallélisme d'instruction de manière particulièrement efficace et affiner la structure aux transferts à mettre en œuvre. Le principe des processeurs TTA et l'environnement de conception associé sont d'abord présentés. La démarche de conception du processeur spécialisé pour le décodage de codes polaires est ensuite expliquée. Deux architectures ont été conçues. La première correspond à un décodeur qui implémente l'algorithme de décodage par annulation successive (SC). La seconde implémente l'algorithme SCAN à sortie souple et correspond à une évolution de la première architecture en y ayant ajouté des unités fonctionnelles adéquates. A nouveau, une démarche de conception de type adéquation algorithme-architecture a été mise en œuvre, mêlant analyse de l'algorithme et exploitation des propriétés de l'architecture TTA, et nécessitant de nombreuses expérimentations et

analyses du résultat avant d'aboutir aux deux solutions proposées. Enfin, la dernière partie du chapitre présente les résultats d'implémentation et tente de les positionner par rapport à d'autres circuits. Cette partie du mémoire est sans doute la plus questionnable. Les cibles passent de circuits de type ASIC à des circuits de type FPGA puis à nouveau ASIC sans grandes transitions ni explications. Par ailleurs, on sait bien qu'il n'est jamais facile de comparer ses propres résultats d'implémentation à ceux de la littérature : les paramètres ne sont jamais exactement les mêmes et les cibles technologiques sont souvent différentes. L'environnement de conception utilisé dans ce chapitre pour concevoir le processeur présente l'avantage de permettre de passer relativement facilement d'une cible technologique à une autre. On comprend donc mal pourquoi lors de la comparaison des résultats de l'architecture basée sur le processeur TTA avec ceux de l'architecture basée sur le processeur Xtena (chapitre 3), la cible ASIC n'est pas la même. Dans le même registre, lors des comparaisons avec pour cible des circuits de type FPGA, des FPGA bien différents sont considérés. Ce n'est pas se mettre dans le meilleur contexte de comparaison que de faire ainsi. Il n'en demeure pas moins que, sur cible ASIC, les résultats obtenus sont particulièrement intéressants : les débits obtenus sont très élevés tout en ayant une consommation énergétique très faible.

Dernier chapitre

Ce chapitre conclut le mémoire et dresse des perspectives.

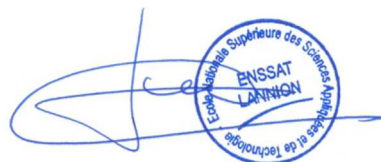
Conclusion

Dans le cadre de cette thèse, différentes solutions de décodage de codes polaires sur des architectures programmables sont proposées. Une première version de décodeur logiciel est proposée. Cette version présente l'avantage d'être générique, flexible, et portable sur des processeurs SIMD. Deux autres solutions de décodeur sont proposées, l'une ciblant un processeur RISC dont le jeu d'instructions peut être étendu, l'autre un processeur plus spécifique, de type TTA, pouvant être configuré plus finement. Dans les deux cas, l'objectif est d'atteindre des compromis débit/consommation plus intéressants qu'avec une exécution sur des processeurs d'usage général. Force est de constater que les résultats obtenus montrent la pertinence de ces solutions. La conception de ces décodeurs résulte d'une démarche de type adéquation algorithme-architecture.

D'une manière générale, le manuscrit est clair et bien organisé. Quelques erreurs sont présentes dans le texte (essentiellement des phrases où il manque un mot), en particulier dans les chapitres 1 et 2. Une ultime relecture devrait permettre d'y remédier rapidement.

Je donne un avis favorable à la présentation orale de la thèse de Mathieu Léonardon.

Fait à Lannion le 26 novembre 2018



Emmanuel Casseau
Professeur des Universités
Université de Rennes 1