Description haut niveau de formes d'ondes pour la radio logicielle sur architectures reconfigurables

Ganda Stéphane OUEDRAOGO, Matthieu GAUTIER, Olivier SENTIEYS

IRISA, INRIA, Université de Rennes 1, France,
IRISA-ENSSAT, 6 rue Kerampont, F-22305 Lannion Cedex
ganda-stephane.ouedraogo@irisa.fr, matthieu.gautier@irisa.fr,
olivier.sentieys@irisa.fr

Résumé – Les travaux s'inscrivent dans le contexte radio logicielle et proposent une alternative basée sur FPGA dans le but de tirer profit du compromis entre la consommation d'énergie et la flexibilité offert par cette technologie. Dans cet article, nous proposons un flot de conception innovant, alliant architecture matérielle et langage de description haut niveau. Ce flot comprend un langage de description spécifique aux formes d'ondes pour les systèmes de communication et la synthèse matérielle automatique utilisant des outils de synthèse de haut niveau. Cette démarche vise à faciliter l'exploration de l'espace d'architectures tout en bénéficiant de connaissances a priori des formes d'ondes cibles. La description haut niveau est validée sur une plateforme FPGA. Des résultats de synthèse d'une forme d'onde IEEE 802.15.4 sont donnés pour différents langages de description (C, VHDL).

Abstract – The work depicted in this paper deals with the outstanding context of Software Defined Radio (SDR) and aims at leveraging the FPGA technology to get to a trade-off between energy consumption and platform flexibility. In this paper, we propose a novel approach combining harware architecture and high level specification. This approach is comprised of a Domain-Specific Language meant for telecommunication systems waveforms and the High Level Synthesis (HLS) tools. It eases the Design Space Exploration and takes advantage of prior knowlegde of the target waveform. Design integration is performed on an FPGA-based platform. Synthesis results of the IEEE 802.15.4 waveform are given considering different description languages (C, VHDL).

1 Introduction

La technologie radio logicielle est un concept proposé par Joseph Mitola comme support matériel à la radio cognitive : une radio intelligente qui peut s'adapter à l'environnement et coopérer avec d'autres équipements de type radio logicielle [1]. Le principe de la radio logicielle est la reconfiguration dynamique de l'architecture en bande de base par le biais d'un ou plusieurs processeurs hôtes. Elle offre ainsi la possibilité de changer le schéma de codage, de modulation, de bande passante et de technique d'accès au canal sans modifier directement le maté- riel [2]. Si la radio logicielle offre une nouvelle infrastructure d'expérimentation des communications sans fil et mobile, les solutions basées sur des processeurs présentent des limites en termes de temps de reconfiguration ou de consommation en énergie [3].
Une alternative à cette radio uniquement logicielle est le développement d'une radio logicielle sur des architectures reconfigurables de type FPGA. L'utilisation de FPGA offre des temps de reconfiguration de l'ordre de la micro-seconde grâce aux techniques de reconfiguration partielle et réduit de façon significative la consommation d'énergie comparativement aux solutions purement logicielles mise en œuvre sur des cœurs de calcul de type processeur. Dans cet article, nous proposons une radio logicielle sur une architecture reconfigurable permettant d'atteindre l'objectif d'une couche physique flexible et faible consommation.
Les deux clés fondamentales pour atteindre cet objectif sont [4] une méthodologie de développement spécifique permettant la flexibilité des spécifications de haut niveau et une architecture modulaire permettant le déploiement de diverses formes d'onde. On se propose à travers nos travaux d'outiller la communauté radio logicielle d'un flot de conception basé sur un langage dédié combiné à la synthèse de haut niveau. Les détails de ce flot sont détaillés dans cet article.

Le suite de ce document est organisée de la façon suivante : la partie 2 détaille le langage proposé, une description du flot de conception est donné en partie 3. Une validation du concept à travers un exemple fonctionnel est proposée en partie 4 et la partie 5 conclut ce document.

2 Un langage de description spécifique (DSL) pour la SDR

L'approche de conception proposée dans ces travaux est illustrée par la Figure 1. L'idée de base est de s'appuyer sur un langage de description spécifique DSL (*Domain Specific Language*) afin de générer automatiquement des formes d'ondes variées (*Waveform Compiler*) et de les intégrer (*Platform in-*

tegration) sur une plateforme matérielle à base de FPGA. Le langage repose sur une bibliothèque d'IP *Intellectual Properties* de traitement du signal et permet de générer différentes architectures relatives à des standard de télécommunications. Un DSL est un langage qui à travers son expressivité ou sa syntaxe demeure spécifique à un domaine d'application [5]. C'est un outil qui simplifie la description d'application et qui est peu utilisé dans le domaine des télécommunications. Par ailleurs, certains travaux de recherches [6] utilisent cette approche DSL afin de modéliser le domaine d'application considéré. On distingue 2 types [5] de DSL : interne ou externe. Les DSL internes réfèrent à des langages fortement dépendant d'un langage hôte. Ils sont en effet imbriqués dans ce dernier et permettent de spécialiser une partie du traitement à effectuer à partir d'instructions spécifiques. Ce type de DSL bénéficie en générale de l'infrastructure de compilation du langage hôte mais ne s'avère pas être le meilleur candidat lorsqu'il s'agit de modéliser entièrement un domaine d'application. A l'opposé, les DSL externes se définissent comme des langages entièrement autonomes et dédiés à la modélisation d'un domaine d'application. Ils proposent une forme de spécialisation du modèle à travers une syntaxe et des outils de compilation adéquats. Cette approche permet aux concepteurs du langage de prendre différentes initiatives selon la finalité désirée sans se soucier d'éventuelles incompatibilités avec un langage hôte. Pour finir, un aspect notable d'une conception à base de DSL est qu'il facilite la communication entre expert du domaine et de ce fait favorise une prise en main assez rapide de l'environnement de programmation. Ces différents arguments nous ont conduit à proposer un DSL dédié au prototypage d'application radio logicielle sur cible FPGA.

Dans le contexte de la radio logicielle, la définition d'un DSL peut se montrer utile à la description d'une application pour une cible FPGA en ce sens qu'il permettra d'insérer à plus haut niveau des informations relatives à celle-ci. Ces informations couvrent un panel assez large allant de l'ensemble des fréquences de fonctionnement dans le cas des systèmes multi-cadences aux informations propres à la plateforme d'exécution (ressources disponibles). Les différents scénarios de reconfiguration peuvent aussi être gérés à ce niveau de description, ce

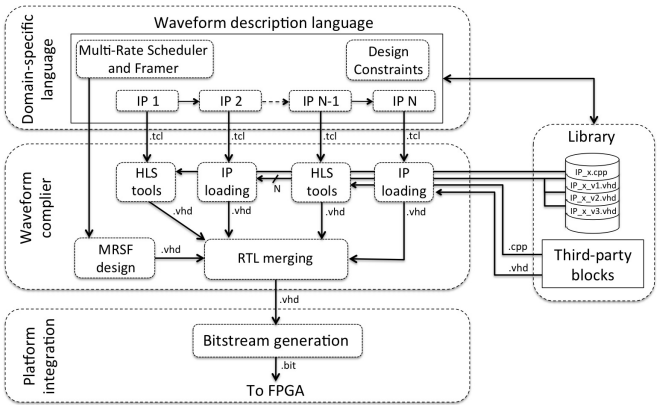

FIGURE 1 – Flot de conception de la radio logicielle sur FPGA.

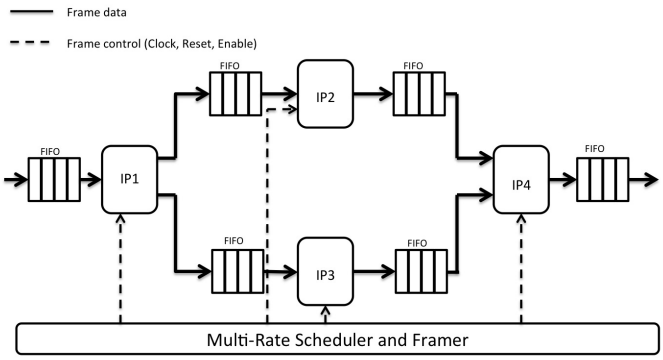

FIGURE 2 – Structure d'interconnections et de controle des IP.

qui représente une indéniable plus value en terme de temps de prototypage. Une caractéristique supplémentaire du DSL proposé est la possible interaction avec les outils de synthèse de haut niveau qui permettent de générer l'architecture de certains blocs de traitement à partir de leur spécification. Cette association vise à conférer un caractère hétérogène au flot de conception tout en garantissant d'éventuelles extensions de ce dernier. Le langage consiste à définir un ensemble de mots clés qui seront ensuite interprétés par le compilateur. Ainsi la spécifica- tion d'une forme d'onde à partir du langage que nous proposons se fait selon les étapes suivantes :

→ **Bibliothèques et fréquences de fonctionnement :** Cette étape est d'une part relative à l'inclusion des bibliothèques d'IP utiles à la synthèse de la forme d'onde et d'autre part à la déclaration des fréquences de fonctionnement des différentes parties du système. Les mots clés prévus à cet effet sont *#include* pour l'inclusion d'une bibliothèque et *frequency* pour la spécification d'une fréquence. Deux types de fréquence peuvent être définies à savoir des fréquences de référence ou des fréquences dérivées.

→ **Champs et trame de données :** Chacun des champs d'une trame de donnée peut être spécifié avec différents arguments référant à sa taille ou encore à la présence éventuelle d'une redondance de donnée. La nature constante ou variable d'un champs est précisée en début de spécification à partir des mot clés *#fieldC* et *#fieldV* respectivement. Cette spécification nous permet entre autres de réaliser des optimisations matérielles consistant par exemple à mapper les données des champs constants en mémoire puis à construire la trame finale par un mécanisme de multiplexage temporel des données. La trame de donnée est ensuite construite en renseignant de façon chronologique les différents champs de cette trame.

→ **Le Start-Of-Frame :** Le *Start-Of-Frame* désigne un champ spécifique de la trame et est utilisé comme évènement de synchronisation en réception. En effet, l'un des objectifs du DSL proposé est de générer automatiquement la logique contrôle des architectures spécifiées. Pour ce faire, un point de synchronisation doit être introduit lors de la spécification de la trame.

→ **Déclaration d'IP :** Dans un but d'hétérogénéité de l'approche, la déclaration des blocs de traitement se fait précédée

d'annotation renvoyant à un outil de synthèse de haut niveau. Les blocs décrits en RTL (Register-Transfert-Level) sont aussi pris en considération ce qui permet de décrire des systèmes mixtes en termes de langage. Ces annotations sont entres autres *#CatapultC* or *#RTL* qui désignent l'outil HLS de Calypto ou la description VHDL d'un bloc.
→ **Déclaration d'interconnexion :** L'idée du DSL est de faciliter le prototypage de forme d'ondes radio logicielle sur FPGA. Ces formes d'ondes sont de types flots de données et requièrent une structuration particulière s'agissant de l'interfaçage des différents blocs qui les composent. Un exemple du flot de données d'une telle forme d'ondes est donné dans la Figure 2. On y voit des interfaces implémentées sous forme de FIFO, ce qui s'avère être une approche commune pour des applications flots de données. Ces FIFOs permettent de réduire les contraintes sur les blocs de calculs en leur permettant de fonctionner à un rythme spécifique. La primitive *framedata* génère une interconnexion de type FIFO et s'agissant des données véhiculés dans l'architecture sans contrainte de débit, la primitive *data* permet de les implémenter sous formes de bus de données. Les outils des synthèse de haut niveau permettent de dimensionner des FIFOs à la volée et nous tirons profit dans le DSL de telles caractéristiques.
→**Instanciation d'IP :** Après la déclaration d'interconnexion et d'IPs, ces IPs sont instanciés selon la forme d'ondes à définir et leurs ports sont connectés aux différentes interconnexions déclarées. Il est possible lors de cette instanciation de préciser quelles champs de la trame sera filtré par le bloc de traitement instancié. Cette astuce permet d'aiguiller la trame de façon optimisée à travers le chemin de donnée et est réalisée avec des signaux de contrôle de type *enable*.

Le langage ainsi décrit a donc pour tâche principale d'assembler différentes IP (ou *block*) tout en générant une logique de contrôle ainsi qu'une infrastructure de communication adéquate. Aussi, l'origine de ces blocs de base se veut diverse. Le développement récent ainsi que la mise sur le marché d'outils de synthèse de haut niveau (*HLS tools*) permet de considérer dans notre approche des IP décrites en *C/C++* ou *SystemC*, langages présentant une certaine abstraction en comparaison au *VHDL* ou au *Verilog* dédiés à l'architecture matérielle. Le caractère hétérogène ainsi conféré à la bibliothèque en cours de développement vise à élargir le spectre de couches physiques mais aussi à bénéficier de travaux antérieurs.

3 Description du flot de conception

Dans la partie 2 nous avons détaillé les différentes étapes de description d'une forme d'ondes par le biais du DSL proposé. Cette spécification est ensuite interprétée selon le flot proposé en Figure 1 qui illustre le mécanisme sous-jacent au DSL. Ce flot se décline en trois étapes essentielles à savoir une première de spécification de la forme d'ondes à partir d'une syntaxe DSL. Ensuite, survient une étape de synthèse des blocs de bases via les outils de synthèse de haut niveau et d'assemblage ceux-ci. Au cours de cette étape, la description de la trame de donnée fournie dans la spécification DSL est utilisée afin générer automatiquement la logique de contrôle du chemin de donnée en construction. Pour finir, une étape d'intégration sur la plateforme cible est réalisée.

Ainsi, l'une des étapes essentielles de ce flot de conception est la phase d'assemblage des blocs de bases au cours de laquelle s'opère l'interconnexion des différents blocs issus de la bibliothèque dans le but de produire la forme d'onde souhaitée. Cette étape est illustrée dans la Figure 1 comme *VHDL merging*. Il y est question d'une part d'outiller les outils HLS afin de générer les descriptions RTL des blocs instanciés ou de directement charger ces blocs lorsque nativement décrits au niveau RTL. D'autre part, il est aussi question de s'assurer de la cohérence des types de données à l'entrée et à la sortie de chaque bloc et de fournir entre autres la structure d'interconnexion (cf. Figure 2) selon que certaines contraintes de débits et/ou de latences soient prises en considération. On peut, comme mentionné plus haut, dégager différents profils d'interconnexion qui vont du simple bus de données à la mémoire RAM en passant par des FIFO. Cette étape s'accompagne en plus de la génération d'une logique de contrôle bidirectionnelle dont la tâche est de fournir et recevoir un ensemble de signaux (*enable*, *clock*, etc) utile au séquencement de ces blocs. Il incombe à l'interpréteur prévu à cet effet d'exécuter correctement cet ensemble de tâches.

Ce flot, en rappel, s'adresse à des plateformes à base de carte FPGA. Cette technologie relativement récente et en perpétuelle évolution offre de nombreuses perspectives en termes de flexibilité et des capacités de calcul fortement intéressant pour des application de type radio logicielle. Les interfaces de programmation de telles cartes sont dans la plupart des cas éditées par les concepteurs, ce qui contraint à l'utilisation d'outils spécifiques dans le processus d'intégration. A ce jour, le flot que nous proposons ne réalise pas une intégration automatique des solutions radio logicielles sur les cartes FPGA, nous utilisons donc comme point d'entrée les différents outils de génération de bitstream de chez Xilinx (EDK, XPS, etc).

4 Validation sur la norme IEEE 802.15.4

Dans cette partie, nous validons différentes étapes de notre étude en implémentant la couche physique IEEE 802.15.4 d'une forme d'ondes Zigbee. La première phase est la description de la structure de la trame à partir du DSL, tandis que la deuxième servira à tester l'approche de synthèse de haut niveau sur une plateforme matérielle.

Le point d'entrée du flot de conception est la spécification de la structure d'une trame (*cf.* partie 2). La trame IEEE 802.15.4 est présentée sur la Figure 3. Dans cette trame, le champ SHR composé du préambule et du SFD est constant (de type *#fieldC*). Le préambule possède une structure répétitive (composée de huit symboles 0) ce qui n'est pas le cas du SFD. Les champs PHR et PSDU sont des champs variables (de type *#fieldV*) dont

FIGURE 3 – Structure de la trame 802.15.4 et type de champs associés.

les données seront issues de blocs de traitement du signal. Le DSL proposé exploite ces différentes caractéristiques afin d'optimiser la structuration de la trame en sauvegardant les données des champs constants dans des mémoires, données qui seront ensuite multiplexées en temps aux données des champs variables issues de blocs de traitement.

L'intégration a été faite sur une plateforme Nutaq Perseus 6010 équipée d'un FPGA Virtex-6 et d'une tête d'émission radio large bande et agile en fréquence. Dans ce contexte radio logicielle, les travaux que nous avons effectués sur cette plateforme ont fait l'objet d'une publication [7]. La forme d'onde IEEE 802.15.4 spécifié à partir du DSL a été généré automatiquement, cette spécification s'est faite selon les étapes mentionnées dans la partie 2. Les résultats de synthèse sur FPGA sont présentés dans la Table 1. On y compare deux approche d'implémentation l'une se basant sur une description RTL de la forme d'onde (HC-VHDL pour Hand-Coded VHDL) et l'autre décrite à partir du flot présenté (HLS-VHDL). La figure 4 montre les signaux Zigbee décodés par le biais de l'équipement *Vector Signal Analysis* (VSA) de chez Agilent. Elle donne le spectre reçu, les signaux bande de base temporels ainsi que la constellation après décodage.

Cette phase applicative a permis de consolider le travail réalisé à ce jour et permet de dégager différentes perspectives. Ce DSL permet comme nous l'avons signalé plus haut d'assembler des blocs hétérogènes afin d'obtenir des formes d'ondes variées. Une étape significative de ce projet consiste aussi en l'enrichissement de la bibliothèque. Ces deux efforts sont réalisés en parallèle, permettant ainsi de tenir compte dans le DSL de l'hétérogénéité de la bibliothèque.

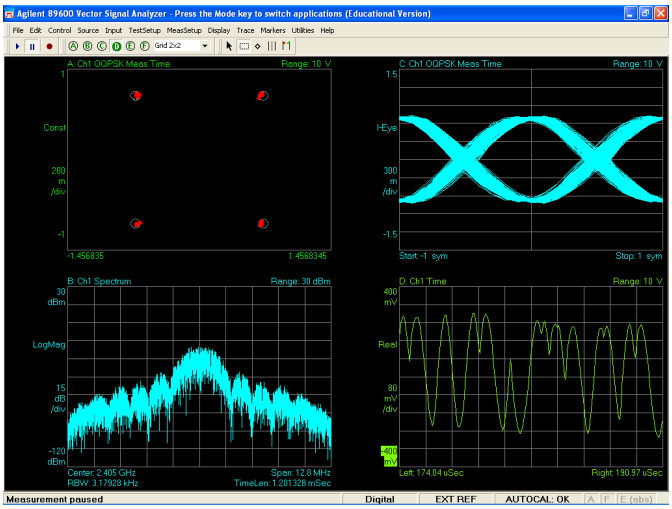

FIGURE 4 – Décodage de signaux IEEE 802.15.4 sur équipement VSA.

5 Conclusion

Dans cet article, nous avons présenté un flot de conception innovant relatif à l'implantation d'application radio logicielle sur des cibles de calcul de type FPGA. Ce flot repose essentiellement sur un langage dédié à la radio logicielle combiné aux outils de synthèse de haut niveau. Les différentes étapes de la spécification à l'aide du langage ont été détaillées et un exemple de réalisation a été discuté. Quelques résultats d'intégration sur une plateforme matérielle sont présentés et commentés.

Les travaux en perspective consistent en un raffinement du DSL afin d'insérer des caractéristiques propres à la radio logicielle à savoir les différents scénarios de reconfiguration dynamique une fois les formes d'ondes spécifiées.

Références

[1] J. Mitola III, *Software radios : Survey, critical evaluation and future directions*, Aerospace and Electronic systems Magazine, IEEE, vol. 8, 1993.

[2] A. Di Stefano, G. Fiscelli, and C. Gianconia, *An FPGA-Based Software Defined Radio Platform for the 2.4GHz ISM Band*, in IEEE Research in Microelectronics and Electronics Ph. D.,vol. 12, 2006, pp. 73-76.

[3] Sous la direction de J. Palicot, *De la radio logicielle à la radio intelligente*, Institut Télécom et LAVOISIER, Paris, 2010

[4] K. He, L. CrocKett, R. Stewart, *Dynamic Reconfiguration Technologies Bases on FPGA in Software Defined Radio System*, in Proceedings of SRD'11 WInnComm-Europe, 22-24 June 2011

[5] M. Fowler with R. Parsons, *Domain-Specific Languages*, The Addison-Wisley Signature Series, 2011.

[6] J. Schafer and D. Klein, *Implementing Situation Awareness fo Car-to-X Applications using Domain Specific Languages*, in IEEE International Conference on Vehicular Technology (VTC-Spring), June 2013.

[7] V. Bhatnagar, G. S. Ouedraogo, M. Gautier, A. Carer, O. Sentieys, *An FPGA Software Defined Radio Platform with High Level Synthesis Design Flow*, in IEEE International Conference on Vehicular Technology (VTC-Spring), June 2013

	Emetteur			Récepteur		
	Slices	FF	LUT	Slices	FF	LUT
HC-VHDL	35	134	86	3668	4292	10945
HLS-VHDL	76	188	202	543	1690	1058

TABLE 1 – Estimation des ressources de l'émetteur et du récepteur IEEE 802.15.4.