

# Implémentation dans un ASIC et simulation mixte d'un cœur de microcontrôleur et de périphériques numériques et analogiques

Sylvain Garnier<sup>1</sup>, Mikaël Tual<sup>1</sup>, Richard Perdriau<sup>2</sup>, Mohamed Ramdani<sup>2</sup>

<sup>1</sup>ATMEL Nantes - La Chantrerie - Route de Gachet - 44036 Nantes Cedex - France

<sup>2</sup>ESEO - 4, rue Merlet-de-la-Boulaye - BP 30926 - 49009 Angers Cedex 01 - France (Pôle CNFM de Rennes - CCMO)

Présentation et contact : Richard Perdriau (richard.perdriau@eseo.fr)

## 1 Introduction

Dans le cadre de l'option Electronique Embarquée (EE) de l'ESEO, les étudiants suivent une formation complète à la microélectronique numérique et analogique. Cette formation fait suite à un enseignement des fondamentaux de l'électronique intégrée (25 heures) en tronc commun au semestre 6 (première année ingénieur - L3) et de conception VHDL (15 heures de cours magistral et 24 heures de mini-projets) en tronc commun au semestre 7 (deuxième année ingénieur - M1). L'enseignement d'option en microélectronique comprend au semestre 8 :

- un cours magistral de conception analogique intégrée (10 heures),
- un cours magistral de synthèse des amplificateurs CMOS intégrés (15 heures) suivi d'un mini-projet (37,5 heures) de conception d'un amplificateur Miller ou cascade,
- un cours magistral de technologie microélectronique (7,5 heures),
- un cours magistral (3,75 heures) suivi d'un TP (4 heures) de langage Verilog,
- un cours magistral (3,75 heures) et des travaux pratiques (12 heures) de logique programmable avancée en VHDL.

Au semestre 9, l'enseignement magistral comprend :

- un cours magistral (3,75 heures) et des travaux pratiques (12 heures) de langage VHDL-AMS,
- un cours magistral (3,75 heures) de langage SystemC,
- un cours magistral (7,5 heures) d'électronique faible consommation,
- une conférence (7,5 heures) sur les convertisseurs A/N et N/A,

En complément de cet enseignement, il nous a semblé primordial de proposer aux étudiants de synthétiser l'ensemble des savoirs de conception et modélisation microélectronique déjà vus au cours de leur cursus, au travers d'une activité de longue durée. De plus, il existait déjà une collaboration forte au niveau recherche entre l'ESEO et ATMEL Nantes. Il a donc semblé naturel de la compléter par une collaboration pédagogique très forte, s'appuyant sur l'ex-

périence des équipes de conception d'ATMEL. Ainsi, deux experts d'ATMEL Nantes, Sylvain Garnier et Mikaël Tual, ont accepté de bâtir, en collaboration avec les enseignants-chercheurs en microélectronique de l'ESEO, un projet de synthèse conséquent (67,5 heures). Ce projet a pour but de permettre aux étudiants d'appréhender l'ensemble des métiers de concepteur en microélectronique.

Il est proposé aux étudiants d'implémenter, à partir d'un cœur de microcontrôleur (en VHDL), un ASIC permettant de générer numériquement une sinusoïde puis de la mettre à disposition sur une sortie analogique du circuit. Cet ASIC devra en outre être robuste vis-à-vis d'une défaillance de l'alimentation, ce qui nécessite la mise en place d'un superviseur d'alimentation intégré. Le projet se décompose en trois parties :

- conception numérique et intégration (5 journées), animée par Sylvain Garnier (expert numérique, ATMEL),
- conception analogique (3 journées), animée par Mikaël Tual (expert analogique, ATMEL),
- flot back-end (1 journée), animée par Richard Perdriau et Mohamed Ramdani (enseignants-chercheurs, ESEO).

Les deux premières parties sont chronologiquement entrelacées, afin de conserver la cohérence du projet.

## 2 Conception numérique

### 2.1 Présentation

L'objectif de cette partie de réaliser un circuit capable de générer des formes d'onde dans le domaine numérique. L'application choisie est la génération d'une sinusoïde de fréquence 100 kHz avec une résolution de 8 bits.

Pour ce faire, il est donc mis à la disposition des étudiants le code source partiel en VHDL (sans le décodeur d'instructions) d'un cœur 8051 écrit par ATMEL, dans lequel les optimisations architecturales propriétaires ont été supprimées et remplacées par du code "standard" non optimisé (4 cycles d'horloge

par instruction) de manière à respecter la propriété intellectuelle de l'entreprise. La figure 1 résume l'architecture globale de la partie numérique du microcontrôleur avec ses mémoires associées.

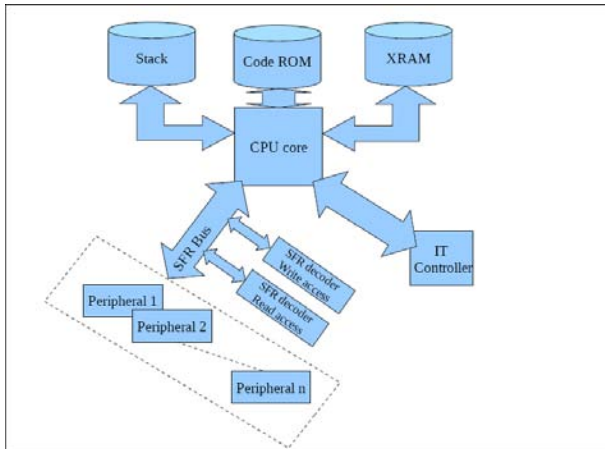


FIG. 1. Architecture de la partie numérique du microcontrôleur

## 2.2 Développement du décodeur d'instructions

Dans un premier temps, les étudiants ont à leur disposition un microcontrôleur "minimal" incluant seulement quelques instructions en exemple ainsi qu'un seul périphérique (un registre 8 bits) dans lequel le code viendra écrire les valeurs successives de la sinusoïde à générer. La figure 2 montre l'architecture du cœur.

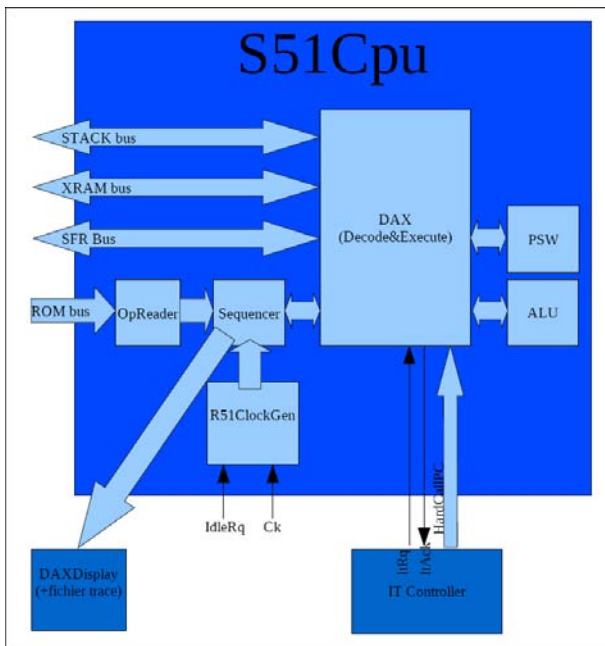


FIG. 2. Architecture du cœur

Les étudiants doivent alors compléter judicieusement le bloc DAX (Decode And Execute) qui est le décodeur d'instructions en logique câblée. Le bloc DAX-Display est un bloc de mise au point écrit spécifiquement par ATMEL, permettant de visualiser directement sous ModelSim® les instructions exécutées par leurs mnémoniques au lieu de leur code hexadécimal. La première étape du développement consiste à écrire le code C du générateur de sinusoïde (dans un premier temps, une simple boucle sans contraintes temporelles), puis à le compiler grâce à une chaîne de développement croisé 8051, ici le logiciel libre SDCC. A partir du code assembleur généré, les étudiants regardent ensuite quelles instructions assembleur sont effectivement utilisées, puis les implémentent une à une en VHDL dans le décodeur d'instructions. Une simulation RTL sous ModelSim® leur permet de valider le fonctionnement du circuit ; un modèle comportemental adéquat pour la ROM vient directement chercher les instructions machine dans le fichier généré par l'éditeur de liens de SDCC.

## 2.3 Développement d'un périphérique : timer

Afin d'obtenir la fréquence précise de 100 kHz pour la sinusoïde numérique, les étudiants sont ensuite amenés à se demander quels moyens peuvent être mis en œuvre à cet effet. Ils sont aiguillés vers l'utilisation d'un timer permettant, par scrutation de son registre d'état dans le code C, de spécifier l'intervalle de temps entre deux échantillons successifs. La prochaine étape consiste à spécifier l'architecture de ce timer (figure 3) puis à écrire son code VHDL. Simultanément, les étudiants sont initiés à la démarche de conception de blocs de propriété intellectuelle (IP), qui nécessite de bien séparer le cœur du périphérique (qui peut être générique) de l'interface spécifique au microcontrôleur dans lequel il sera intégré.

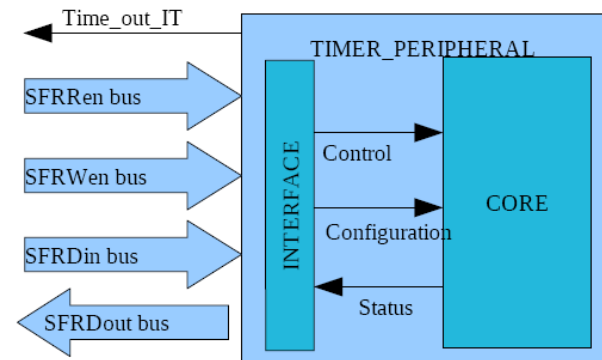


FIG. 3. Architecture du timer

Par la suite, les étudiants vont jouer le rôle d'intégrateurs après celui de concepteurs : à partir de la description générale de l'espace adressable du 8051, ils devront spécifier l'adresse de base du timer, puis com-

pléter en VHDL les décodeurs correspondants ("SFR Decoder" sur la figure 1). Après modification du code C et compilation, une nouvelle simulation RTL permet de vérifier que la fréquence générée est conforme au cahier des charges.

## 2.4 Développement du contrôleur d'interruption

Très vite, les étudiants s'aperçoivent de la faible efficacité de la technique de scrutation. Ils en viennent donc à implémenter la génération de la sinusoïde par une routine d'interruption déclenchée par le timer.

Cette étape consiste donc à écrire en VHDL le contrôleur d'interruption, connecter la sortie "overflow" du timer sur ce contrôleur, puis modifier le code C afin d'y insérer la routine d'interruption. A ce titre, le décodeur d'instructions du microcontrôleur doit de nouveau être modifié afin d'implémenter l'instruction machine de retour d'interruption.

Par la même occasion, les étudiants se rendent compte de la faible efficacité du séquenceur, qui utilise 4 cycles d'horloge par instruction ; ils sont initiés à l'optimisation en réduisant eux-mêmes ce temps à 3 cycles par instruction par modification du code VHDL du séquenceur. Une simulation fonctionnelle finale permet de valider l'ensemble du fonctionnement.

L'ensemble du projet, avant synthèse, aura occupé 2 journées.

## 2.5 Utilisation du synthétiseur logique

Les 2 journées suivantes sont consacrées à la synthèse logique. Le logiciel utilisé est BuildGates<sup>®</sup> Extreme de Cadence, fourni par le CRCC. La technologie cible est une technologie CMOS 0.35  $\mu\text{m}$ , qui a été choisie afin de permettre une conception assez simple de la partie analogique associée au projet.

Les étudiants commencent par effectuer une première synthèse sans contraintes du timer puis de l'ensemble de la partie numérique, suivie d'une simulation fonctionnelle au niveau structural sous ModelSim<sup>®</sup>, afin de vérifier que la synthèse n'a pas modifié la fonctionnalité du système. Ensuite, une simulation temporelle rétroannotée au moyen des fichiers SDF (Standard Delay File) leur permet d'appréhender l'influence des temps de propagation sur le fonctionnement du système ; entre autres, ils peuvent constater le bruit numérique généré sur la sinusoïde en raison des commutations décalées des différents bits du registre de sortie (figure 4).

La prochaine étape consiste à réaliser une synthèse sous contraintes temporelles, qui permet de montrer l'influence de ces contraintes au niveau porte (insertion de buffers d'horloge et augmentation de la surface) ainsi que la notion de "slack". Dans la technologie choisie, cette simple synthèse ne permet pas au microcontrôleur de fonctionner à la fréquence nominale (50 MHz) d'après la simulation. Ceci permet

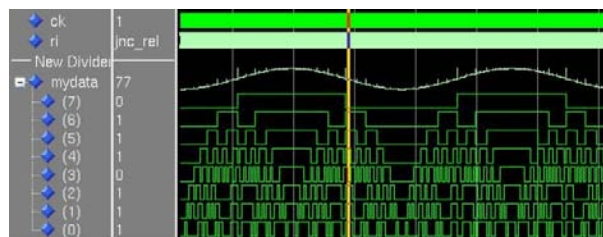


FIG. 4. Simulation numérique rétroannotée : bruit observé sur la sinusoïde

d'initier les étudiants à la notion de chemin critique et d'identifier ce dernier. Une analyse fine indiquant que ce chemin n'est en fait jamais emprunté en fonctionnement normal du processeur ("faux chemin"), ils peuvent modifier les contraintes afin de permettre au microcontrôleur d'être simulable à la fréquence maximale de fonctionnement prévue.

Au bout des 4 premiers jours, les étudiants sont prêts à passer à la simulation mixte, et c'est à ce moment que démarre la partie conception analogique.

## 3 Conception analogique

### 3.1 Présentation

Le sujet de la partie analogique consiste en la conception d'un superviseur d'alimentation pour le microcontrôleur développé dans la partie numérique. Ce superviseur a pour objet de fournir, à partir de la surveillance de la tension d'alimentation, un signal logique passant à l'état haut quand cette tension passe au-dessous d'un seuil déterminé ; ce signal logique peut ensuite être utilisé pour déclencher une interruption ou même un RESET du microcontrôleur. Ce superviseur d'alimentation, dont l'architecture globale est représentée figure 5, comprend :

- une référence de tension "bandgap" stable en température et indépendante de la tension d'alimentation,
- un pont diviseur fournissant une image de la tension d'alimentation,
- un comparateur fournissant le signal logique de sortie.

Comme indiqué au début de l'article, des cours de conception analogique intégrée ont été dispensés au semestre 8 ; cette partie du projet permet aux étudiants de mettre directement en pratique ces connaissances. De plus, un objectif connexe du projet est l'apprentissage de la syntaxe SPICE, jusqu'alors toujours "masquée" par l'utilisation à l'intérieur du cursus d'outils graphiques comme PSpice<sup>®</sup> ou LTSpice<sup>®</sup>. Les outils utilisés sont Virtuoso<sup>®</sup> de Cadence (fourni par le CRCC) pour la saisie de schéma, et Eldo<sup>®</sup> de Mentor Graphics pour la simulation.

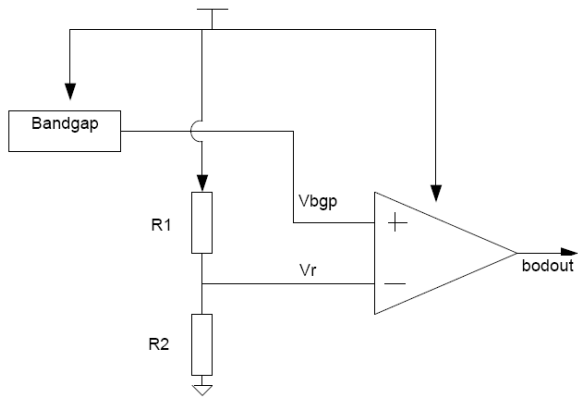


FIG. 5. Architecture globale du superviseur d'alimentation

### 3.2 Conception de la référence bandgap

L'architecture globale de la référence de tension bandgap est donnée figure 6 (sans le circuit de démarrage).

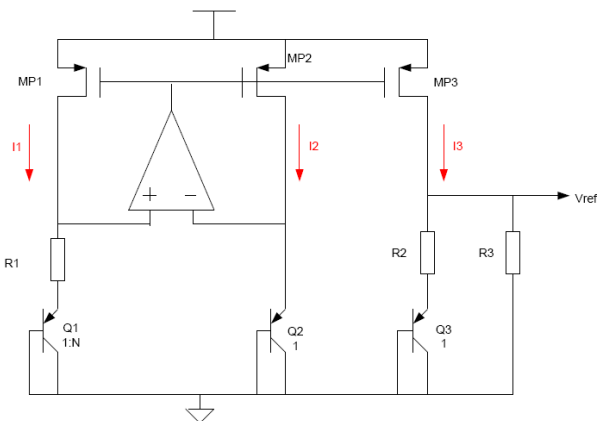


FIG. 6. Architecture du bandgap (sans le circuit de démarrage)

Les étudiants commencent tout d'abord par effectuer un calcul théorique de l'ensemble des résistances et des dimensions des transistors du montage en fonction du cahier des charges. Une simulation électrique est ensuite effectuée avec un macromodèle Eldo<sup>®</sup> parfait de l'amplificateur. Des ajustements sont effectués avant de passer à la conception de l'amplificateur lui-même.

L'étape suivante consiste en la conception complète de l'amplificateur du bandgap au niveau transistor. Il s'agit d'un amplificateur simple étage avec level shifters et circuit de mise en veille, dont l'architecture est présentée figure 7.

Le produit gain-bande étant sans objet dans le cadre de cet amplificateur destiné à fonctionner en statique,

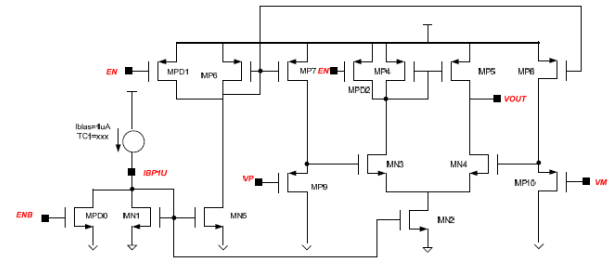


FIG. 7. Architecture de l'amplificateur

les étudiants se concentrent sur l'amélioration du gain en boucle ouverte ainsi que sur la plage de tension d'entrée utilisable.

La dernière étape consiste en l'inclusion d'un circuit de démarrage et de mise en veille, vu théoriquement pendant les cours du semestre 8 et rappelé dans ce projet. Les dimensions des transistors de ce montage sont fournies aux étudiants afin de gagner du temps. Ensuite, la simulation globale peut être effectuée avant passage à la partie intégration et simulation mixte. Le comparateur du bandgap est laissé au niveau macromodèle afin de simplifier le projet. La figure 8 présente un exemple.

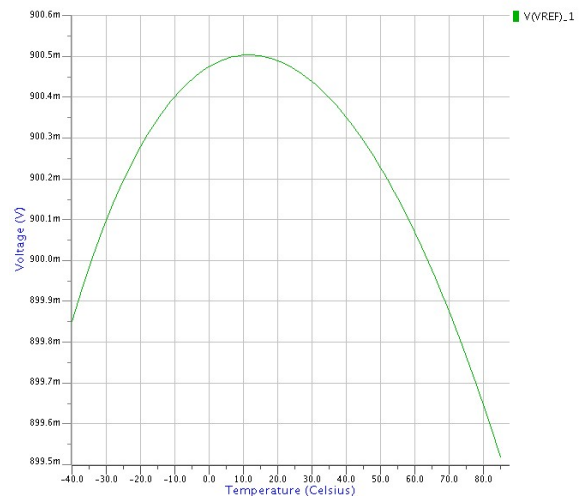


FIG. 8. Exemple de simulation du bandgap

## 4 Intégration et simulation mixte

Le dernier jour de conception du projet est consacré à la simulation mixte. Il s'agit d'ajouter à l'existant numérique le superviseur d'alimentation ainsi que le modèle d'un convertisseur numérique-analogique 8 bits tel qu'il serait réellement implémenté dans un microcontrôleur réel. Le synoptique global utilisé pour la simulation mixte est représenté

figure 9. L'outil utilisé est ADVance-MS<sup>®</sup> de Mentor Graphics, qui présente l'avantage de pouvoir à la fois importer directement la netlist du superviseur d'alimentation et simuler des modèles écrits en VHDL-AMS.

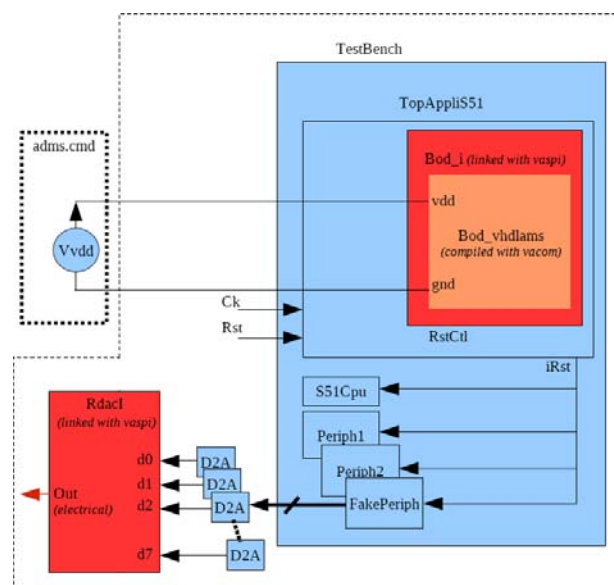


FIG. 9. Architecture complète du microcontrôleur

Dans un premier temps, les étudiants écrivent un modèle VHDL-AMS haut niveau du superviseur d'alimentation, ce qui permet en même temps d'effectuer des rappels sur le langage, de montrer l'intérêt d'une modélisation haut niveau et de diminuer le temps de simulation initial. Ensuite, ils remplacent le modèle par la netlist SPICE du superviseur et vérifient que les résultats de simulation restent corrects. La dernière étape consiste en l'écriture directe du modèle SPICE du convertisseur numérique-analogique (de type R-2R) et son importation dans le projet. La figure 10 présente la simulation globale du microcontrôleur avec son superviseur d'alimentation ; on peut y voir un RESET généré par une chute de la tension d'alimentation.

A ce point, les étudiants ont pu voir l'ensemble des métiers liés à la conception d'un circuit intégré mixte : concepteur d'IP, intégrateur, concepteur analogique. Il leur reste la partie back-end.

## 5 Back-end

La partie back-end dure une journée et est basée sur l'utilisation de First Encounter<sup>®</sup> de Cadence, fourni par le CRCC. Cette partie vise avant tout à présenter l'outil sur un exemple assez complexe. Les notions suivantes y sont abordées :

- l'utilisation de générateurs de blocs (RAM, ROM),

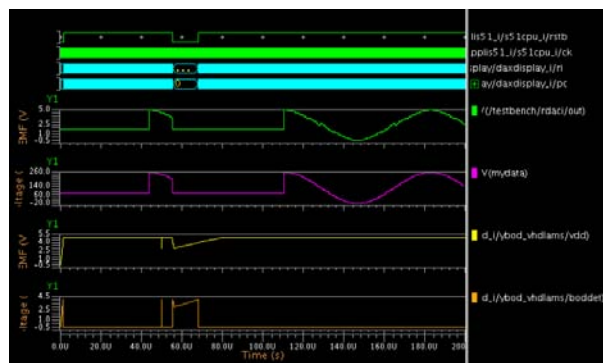


FIG. 10. Simulation mixte complète du microcontrôleur

- le plan d'ensemble (floorplanning),
- le placement des entrées/sorties,
- le routage des alimentations (capacités de découplage de la matrice, bandes et anneaux d'alimentation),
- l'extraction des parasites,
- les fichiers à fournir à un fondeur (GDS2).

A la fin, le fichier GDS2 est réimporté sous Virtuoso<sup>®</sup> et permet aux étudiants de visualiser l'ensemble du layout (sauf les blocs de RAM et ROM, propriétés du fondeur).

## 6 Conclusion

Ce projet de synthèse, très apprécié des étudiants comme des enseignants, regroupe de façon cohérente tous les métiers de la conception en microélectronique, au travers d'outils industriels de plusieurs éditeurs. Il permet aux étudiants de l'option Electronique Embarquée (EE) de l'ESEO d'être pleinement opérationnels dans une double compétence microélectronique analogique-numérique, qui devient de plus en plus rare aujourd'hui. Ceux-ci ont pu réviser ou apprendre les langages VHDL, VHDL-AMS, SPICE et C embarqué grâce à un exemple directement tiré de l'industrie, et appliquer une méthode de développement également industrielle.

Ce projet montre également de façon très claire les bénéfices liés à une collaboration étroite entre l'industrie et l'enseignement, au travers de l'engagement des experts d'ATMEL Nantes vis-à-vis de la pédagogie. Ce projet sera bien évidemment pérennisé au cours des années scolaires ultérieures.