# Contribution à la conception de systèmes numériques adaptatifs

Pascal Benoit

ACADEMIE DE MONTPELLIER

UNIVERSITÉ DE MONTPELLIER

# HABILITATION A DIRIGER DES RECHERCHES

# CONTRIBUTION A LA CONCEPTION DE SYSTEMES NUMERIQUES ADAPTATIFS

**Pascal BENOIT**

Maître de Conférences

Soutenue le 16 novembre 2015 devant le jury composé de :

| | | |
|---|---|---|
| M. Michel Auguin | Directeur de Recherche, CNRS | Rapporteur |
| M. Marc Belleville | Directeur de Recherche, CEA LETI | Examinateur |
| M. Ian O'Connor | Professeur, Ecole Centrale de Lyon | Rapporteur |
| M. Guy Gogniat | Professeur, Université de Bretagne Sud | Examinateur |
| M. Michel Paindavoine | Professeur, Université de Bourgogne | Rapporteur |
| M. Gilles Sassatelli | Directeur de Recherche, CNRS | Examinateur |
| M. Lionel Torres | Professeur, Université de Montpellier | Examinateur |

# AVANT PROPOS

Ce document présente la synthèse les travaux effectués depuis septembre 2001, date à laquelle j'ai débuté ma thèse de doctorat. Les travaux de recherche ont été effectués au LIRMM (Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier) au sein du département de Microélectronique. Les activités d'enseignement ont été menées à Polytech Montpellier / département EII (Electronique, Informatique Industrielle) de l'Université de Montpellier.

Ce document est composé de trois parties :

- Partie 1 : Bilan des activités et du parcours de recherche

Ce premier chapitre d'une quarantaine de pages présente mon curriculum vitae, le résumé des thèmes de recherche, le bilan de ma production scientifique, mes responsabilités scientifiques, mes responsabilités administratives, ainsi que les contrats de recherche et les activités d'enseignement dans lesquels je suis impliqué. Je termine cette partie par la liste de mes publications

- Partie 2 : Synthèse des travaux et projet de recherche

Cette deuxième partie a pour objectif de détailler mes thèmes de recherche et d'exposer mes perspectives pour les prochaines années.

- Partie 3 : Articles significatifs

Cette annexe regroupe 5 articles majeurs couvrant mes thématiques de recherche.

# Glossaire

| | |
|---|---|
| **AAA** | Adéquation Algorithme Architecture |
| **AES** | Advanced Encryption Standard |
| **ALU** | Arithmetic and Logic Unit |
| **API** | Application Programming Interface |
| **ASIC** | Application Specific Integrated Circuit |
| **ASIP** | Application Specific Instruction set Processor |
| **BIST** | Built-In Self Test |
| **BRAM** | Block RAM |
| **BTI** | Bias Temperature Instability |
| **CABA** | Cycle-Accurate Bit-Accurate |
| **CGRA** | Coarse Grain Reconfigurable Architecture |
| **CMOS** | Complementary Metal Oxyde Semi-conductor |
| **CMP** | Chip Multi Processing |
| **CPM** | Critical Path Monitor |
| **CPU** | Central Processing Unit |
| **DCT** | Discrete Cosine Transform |
| **DEMA** | Differential Electro Magnetic Analysis |
| **DES** | Data Encryption Standard |
| **DFG** | Data Flow Graph |
| **DHM** | Dynamic Hardware Multiplexing |
| **DRET** | Distributed Raw Event Table |
| **DSP** | Digital Signal Processor |
| **DVFS** | Dynamic Voltage and Frequency Scaling |
| **EM** | Electro-Magnetic |
| **FD-SOI** | Fully Depleted Silicon On Insulator |
| **FEAz** | Fault Effect Analyzer |
| **FFT** | Fast Fourier Transform |
| **FIFO** | First-In First-Out |
| **FPGA** | Field Programmable Gate Array |
| **FSM** | Finite State Machine |
| **GALS** | Globally Asynchronous Locally Synchronous |
| **GOPS** | Giga Operations Per Second |
| **GPP** | General Purpose Processor |
| **HCI** | Hot Carrier Injection |
| **HW** | Hardware |
| **IDCT** | Inverse DCT |
| **IoT** | Internet of Things |
| **IQ** | Inverse Quantification |
| **ISS** | Instruction Set Simulator |
| **IVLC** | Inverse Variable Length Coding |
| **KPN** | Kahn Process Network |
| **LNA** | Low Noise Amplifier |
| **LTE** | Long Term Evolution |
| **LUT** | Look Up Table |

| | |
|---|---|
| **MAC** | Multiply Accumulate |
| **MC** | Mode de Consommation |
| **MIMO** | Multiple-Input Multiple-Output |
| **MIPS** | Microprocessor without Interlocked Pipeline Stages |
| **MJPEG** | Motion JPEG |
| **MMU** | Memory Management Unit |
| **MPI** | Message Passing Interface |
| **MPSOC** | Multi-Processor SoC |
| **NOC** | Network on Chip |
| **NOP** | No Operation |
| **NPU** | Network Processing Unit |
| **OFDM** | Orthogonal Frequency-Division Multiplexing |
| **OS** | Operating System |
| **PCM** | Performance Counter Monitor |
| **PDS** | Path Delay Sensor |
| **PE** | Processing Element |
| **PID** | Proportionnel Intégral Dérivé |
| **PMIC** | Power Management Integrated Circuit |
| **PMU** | Performance Monitoring Unit |
| **PRN** | Pseudo Random Number |
| **PRNG** | PRN Generator |
| **PUF** | Physically Unclonable Function |
| **PVT** | Process Voltage Temperature |
| **PWCS** | Piece-Wise Convex Subset |
| **RISC** | Reduced Instruction Set Computer |
| **RO** | Ring Oscillator |
| **ROS** | RO Sensor |
| **RTL** | Register Transfer Level |
| **RTOS** | Real Time Operating System |
| **SA** | Signal Activity |
| **SME** | Smart Memory Engine |
| **SMT** | Simultaneous Multi Threading |
| | System on Chip |
| **SP** | Signal Probability |
| **SRAM** | Synchronous Random Access Memory |
| **SW** | Software |
| **TCP** | Transmission Control Protocol |
| **TDDB** | Time-Dependent Dielectric Breakdown |
| **TSI** | Traitement du Signal et des Images |
| **TTI** | Transmission Time Interval |
| **UART** | Universal Asynchronous Receiver Transceiver |
| **UDP** | User Datagram Protocol |
| **VCD** | Value Change Dump |
| **VFI** | Voltage Frequency Island |
| **VLIW** | Very Long Instruction Word |

# TABLE DES MATIERES

# PARTIE 1 :
# BILAN DES ACTIVITES ET DU PARCOURS DE RECHERCHE

Cette première partie présente mon curriculum vitae, le résumé des thèmes de recherche, le bilan de ma production scientifique, mes responsabilités scientifiques, mes responsabilités administratives, ainsi que les contrats de recherche et les activités d'enseignement dans lesquels je suis impliqué.

# 1 CURRICULUM VITAE

## 1.1 Etat civil


| | |
|---|---|
| Nom : | BENOIT |
| Prénom : | Pascal |
| Date de naissance : | 7 avril 1977 |
| Lieu de naissance : | Nîmes |
| Nationalité : | Française |
| Situation de famille : | Pacsé, 1 enfant |
| Adresse professionnelle : | Université Montpellier / LIRMM |
| | UMR 5506 - CC477 |
| | 161, rue Ada |
| | 34095 Montpellier Cedex 5 |
| Téléphone professionnel : | 04 67 41 86 96 |
| Fax : | 04 67 41 85 00 |
| Email : | Pascal.Benoit@lirmm.fr |


## 1.2 Cursus

| | |
|---|---|
| 2001 – 2004 | Thèse de doctorat en Systèmes Automatiques et Microélectroniques<br>Soutenue le 11 octobre 2004<br>LIRMM, Université Montpellier II |
| 2001 | DEA Systèmes Automatiques et Microélectroniques – Mention Bien<br>*LIRMM, Université Montpellier II* |
| 2000 | Maîtrise EEA – Mention Bien<br>*Université Montpellier II* |
| 1999 | Licence EEA<br>Université Montpellier II |
| 1998 | DEUG Sciences de la Matière<br>*Université Montpellier II* |
| 1995 | Baccalauréat série S (Option Physique-Chimie) – Mention ABien<br>*Lycée Montaury, Nîmes* |

## 1.3 Parcours professionnel

| | |
|---|---|
| Depuis 09/2006 | Maître de Conférences Titulaire – Section CNU 61<br>Enseignement : Polytech Montpellier/ EII, Université Montpellier II<br>Recherche : Département Microélectronique du LIRMM<br>*Titulaire de la PES depuis 2012* |
| 09/05 - 09/06 | Maître de Conférences Stagiaire – Section CNU 61<br>Enseignement : Polytech Montpellier/ EII, Université Montpellier II<br>Recherche : Département Microélectronique du LIRMM |
| 11/04 - 08/05 | Stage Post-Doctoral<br>Responsable: Prof. Juergen Becker<br>ITIV, Karlsruhe Institute of Technology, Allemagne |

# 2 ACTIVITES DE RECHERCHE

## 2.1 Contexte

Mes activités de recherche s'inscrivent dans les thèmes portés par le département Microélectronique du Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM). Le département Microélectronique du LIRMM est spécialisé dans la recherche de solutions innovantes pour modéliser, concevoir et tester les circuits et systèmes intégrés complexes. Ses champs de compétences lui permettent de mener des activités de recherche sur les aspects suivants : le développement de nouvelles générations d'architectures de processeurs pour des applications dans le domaine de l'embarqué ou du HPC (High Performance Computing) ; le test et la conception en vue du test de circuits intégrés et de microsystèmes (modélisation, détection, diagnostic de défaillances physiques, etc.) ; la sécurité numérique pour la confidentialité et l'intégrité au niveau matériel ; des applications autour de la santé, notamment la conception de circuits de neurostimulation pour les systèmes implantés dans le corps humain. Pour mener ces activités, le département s'appuie sur un effectif d'une centaine de personnes, parmi lesquelles des chercheurs, enseignants-chercheurs, doctorants, post-doctorants, ingénieurs et techniciens. Ses coopérations académiques, son implication dans de nombreux projets nationaux (ANR, FUI) et internationaux (FP7, ENIAC et CATRENE), sa participation et son aide à la création d'entreprises, et ses activités de transfert et de valorisation vers le secteur industriel, en font un acteur incontournable dans le paysage de la recherche scientifique française et européenne en Microélectronique.

J'ai intégré le département Microélectronique du LIRMM comme doctorant en septembre 2001. Mes recherches ont porté principalement sur le thème de la conception d'architectures reconfigurables dynamiquement. Dans ce cadre, j'ai plus particulièrement contribué à la mise en œuvre de méthodes permettant d'exploiter plus efficacement les ressources matérielles.

Après avoir soutenu mon doctorat en octobre 2004, j'ai effectué un stage post-doctoral dans l'équipe du Professeur Jürgen Becker à l'Université de Karlsruhe. J'ai ensuite réintégré le département Microélectronique du LIRMM en tant que Maître de Conférences en septembre 2005 et je suis depuis rattaché à un groupe constitué de 5 membres permanents : 3 enseignants chercheurs (Michel ROBERT - PR, Lionel TORRES – PR, Pascal BENOIT – MCF) et 2 chercheurs (Gilles SASSATELLI – DR CNRS, Abdoulaye GAMATIE – CR CNRS). Ce groupe a pour sujet principal le calcul adaptatif (ADAC - Adaptive Computing), abordé à travers différents thèmes : conception de systèmes adaptifs, sécurité matérielle, technologies émergentes.

## 2.2 Résumé des thèmes de recherche ADAC

Afin de donner un aperçu complet de l'environnement dans lequel j'évolue, les sous-sections suivantes présentent un résumé des travaux de recherche menés dans l'équipe ADAC. Mes contributions personnelles seront quant à elle développées dans le chapitre 2 de ce manuscrit. L'objectif de nos travaux est la conception de systèmes et l'élaboration de stratégies permettant une adaptation de la structure ou du fonctionnement du circuit, en vue d'optimiser les performances, minimiser la consommation, garantir une qualité de service, rendre le composant plus fiable, plus sécurisé, utiliser plus efficacement les nouvelles technologies, etc. Notre approche repose sur des schémas d'architectures fortement distribuées qui facilitent le passage à l'échelle dans un contexte technologique où la complexité croît de façon exponentielle. Les trois axes principaux de notre équipe concernent la conception d'architectures et le développement d'algorithmes pour l'adaptation, l'étude de technologies émergentes, et la sécurité matérielle.

### 2.2.1 Architectures et algorithmes pour l'adaptation

Même si l'évolution technologique permet d'accroître les fonctionnalités et les performances, elle engendre de réels problèmes pour les circuits et systèmes intégrés en termes de sensibilité à la variabilité des procédés de fabrication, à l'environnement applicatif ou au vieillissement. Notre objectif est de développer des méthodes et des architectures permettant aux circuits ou systèmes de

s'adapter. Nous avons développé une architecture MPSOC ouverte (*OpenScale*) à mémoire distribuée, qui permet de tester diverses solutions d'adaptation : les contributions couvrent notamment l'asservissement intelligent des modes de fonctionnement (couples fréquence-tension d'alimentation) ou des techniques de modélisation dynamique de la consommation en fonction de mesures de consommation locales à l'aide de capteurs, ou pour des considérations telles que la tolérance aux pannes. Toujours avec un approche haut niveau, nous avons élaboré des solutions dédiées aux FPGA permettant d'adapter les configurations en fonction des paramètres technologiques de fabrication et de vieillissement du circuit. Des solutions innovantes de gestion distribuée ont ainsi été développées dont certaines en coopération avec STMicroelectronics et le CEA dans le cadre du projet Européen MODERN. Des travaux débutés dans le cadre du projet européen Mont-Blanc s'intéressent à l'exploitation de ces propositions dans le cadre d'accélérateur de traitement pour l'informatique haute performance.

### 2.2.2 Technologies émergentes

Les technologies à base de mémoires magnétiques sont considérées comme une alternative réelle aux approches actuelles (SRAM, DRAM et FLASH). Notre équipe a largement contribué à l'étude de solutions architecturales tirant parti des caractéristiques de ces mémoires. Nous avons exploré leur exploitation dans le contexte des composants reprogrammables, qui a donné lieu à la réalisation d'un prototype. Ces travaux qui ont été menés dans le cadre de plusieurs projets ANR, ont débouché sur plusieurs innovations dont certaines ont été brevetées. Actuellement, nous étudions les apports relatifs à l'utilisation de cette technologie dans la hiérarchie mémoire d'architectures à base de processeurs généralistes. Ces travaux ont permis la collaboration avec la société CROCUS, leader en Europe sur ces technologies. Un projet FUI (*MultiSmart*), auquel je participe, vient de débuter sur cette thématique.

### 2.2.3 Sécurité matérielle

La sécurité requiert des méthodes et des moyens pour garantir la confidentialité, l'authenticité, l'intégrité et la disponibilité à la fois des données et des systèmes d'informations. Les systèmes intégrés étant le support matériel de l'information numérique, ils constituent un maillon essentiel de la chaîne de sécurité. Celle-ci repose sur la cryptographie, qui elle même s'appuie sur des outils mathématiques permettant notamment de chiffrer des messages afin de les rendre inintelligibles. Si les mathématiques assurent une certaine robustesse vis à vis des outils de cryptanalyse classique, le passage au monde physique pose de nouveaux problèmes. L'implémentation de ces algorithmes cryptographiques repose sur des *cryptoprocesseurs* qui peuvent laisser fuir des informations par des « canaux auxiliaires » de plusieurs types (temps de calcul, consommation, émissions électromagnétiques, etc.) : par l'observation de ces canaux cachés, il est possible de remonter jusqu'aux clés secrètes et mettre en péril la sécurité du système tout entier. L'objectif de nos travaux est d'étudier les vulnérabilités des implantations logicielles/matérielles des algorithmes cryptographiques face à ces attaques afin de concevoir des systèmes sécurisés de bout en bout. Il est nécessaire pour cela d'identifier les éléments pouvant être susceptibles de laisser fuir les informations secrètes afin de développer des stratégies efficaces privilégiant un équilibre entre performance et sécurité pour protéger de telles architectures. C'est dans le cadre de cette étude que fut développé le processeur *SecretBlaze*, qui est la brique élémentaire du MPSOC *OpenScale*. Le *SecretBlaze-SCR* est la version robuste développée dans le cadre de ces travaux, qui implémente un ensemble de contremesures visant à garantir une sécurité matérielle à moindre coût.

## 2.3 Plateformes technologiques

Nous sommes à l'initiative et impliqués dans la gestion de plusieurs plateformes utilisées dans le cadre de nos travaux de recherche, dans notre groupe et au delà tel que cela est précisé dans les paragraphes ci-dessous.

- ***SECNUM***

Cette plateforme financée à l'origine (2008) par la Région Languedoc Roussillon, le CNRS, l'UM2 et l'industrie dans le cadre des grands plateaux techniques, a pour objectif de proposer des services dans le domaine de la sécurité numérique pour les circuits intégrés. L'objectif de cette plateforme est d'analyser les potentialités, en termes sécuritaires, des plateformes matérielles et des systèmes embarqués, et de proposer une plateforme d'expérimentation académique ouverte (de l'échelle régionale à l'échelle internationale, ainsi qu'au monde industriel) unique pour l'analyse et la compréhension des mécanismes sécuritaires.

Ainsi, autour de cette plateforme, plusieurs projets de recherche ont vu le jour, comme l'ANR SECRESOC, ou le projet Européen MODERN. Par ailleurs, une opération de transfert a démarré en 2012, dans le cadre du projet d'initiative d'excellence IDEFI-FINMINA porté par le GIP CNFM. Il s'agit de créer une version « Education » de la plateforme qui est exploitée dans le cadre du Pôle CNFM de Montpellier. Au cours de l'année 2014 - 2015, plusieurs actions ont été réalisées sur les aspects équipement, formation et communication :

- *mise à jour de la Plateforme CNFM*
- *Réalisation d'un catalogue de formations et développement des supports pédagogiques correspondants*
- *Mise en place d'un partenariat avec le Rectorat de Montpellier*
- *Réalisation de conférences spécialisées*
- *Réalisation de stages technologiques*
- *Communication dans des conférences / journaux*

Au total ce sont plus de 500 personnes qui ont été touchées cette année à travers les actions que nous avons menées. Nous sommes également en train de faire évoluer les activités de recherche de cette plateforme vers les problématiques liées notamment à l'Internet des Objets.

- ***Plateformes libres***

Il s'agit là de deux systèmes open-source que nous avons souhaité ouvrir à la communauté : le *SecretBlaze* et *OpenScale*. Le premier, (microprocesseur sécurisé haute performance), est un processeur développé au sein de notre groupe du LIRMM. Ce processeur est actuellement utilisé dans plusieurs projets locaux (ANR, plateforme SECNUM, projet FP7 CLERECO etc.) afin de concevoir et prouver des mécanismes de protection contre les attaques par canaux cachés. Il a également mis à disposition de la communauté, et un système intégré en technologie 65nm de ST a été conçu et fabriqué par l'équipe DREAM[1] de l'institut Pascal à Clermont-Ferrand.

La plateforme multiprocesseur matérielle logicielle *OpenScale* comporte un support matériel multiprocesseur innovant de par ses caractéristiques et une architecture logicielle associée qui s'articule autour d'un système d'exploitation dédié et des API de programmation. Cette plateforme sert de support d'expérimentation pour des projets internes dédiés à l'informatique embarquée adaptative, et a donné lieu à des instances spécifiques constituant des accélérateurs de traitement pour le calcul intensif dans le cadre du projet Européen « MontBlanc ».

[1] http://cmp.imag.fr/aboutus/gallery/details.php?id_circ=498&y=2014

## 2.4 Collaborations scientifiques

Qu'il s'agisse de mes activités de chercheur ou d'enseignant, je suis impliqué dans de nombreuses collaborations académiques et industrielles, qui s'établissent dans le cadre de projets ou d'accompagnement telles que je le décris ci-dessous.

### 2.4.1 Collaborations académiques

- Laboratoire **ETIS** (Equipes de Traitement des Images et du Signal ) UMR CNRS 8051, équipe commune **ENSEA** /Université de Cergy Pontoise, France

C'est dans le cadre de ma thèse que j'ai eu l'opportunité de collaborer avec l'équipe de Didier Demigny, Professeur des Universités. Nous avons mis en place un certain nombre de métriques de caractérisation des architectures, ce qui a mené à la publication de plusieurs articles.

- Université de Karlsruhe, Institut für Technik der Informationsverarbeitung (**ITIV**), Karlsruhe Institue of Technology (**KIT**), Allemagne

J'ai travaillé dans l'équipe du Professeur Jürgen Becker au cours d'un séjour post-doctoral. Cette collaboration a donné lieu à plusieurs publications (cf. liste de publications) ainsi qu'un contrat de collaboration (MoKa).

- Chair of Dependable Nano Computing (**CDNC**), Karlsruhe Institue of Technology (**KIT**), Allemagne

Dans le cadre des travaux de recherche sur l'étude du vieillissement des composants, je travaille avec l'équipe du Professeur Mehdi Tahoori. Cette coopération récente et très active s'est concrétisée par une publication, l'invitation à un jury de thèse ainsi qu'à une proposition de projet ANR/DFG.

- Hardware Design Support Group (**GAPH**), Université pontificale catholique du Rio Grande do Sul (**PUCRS**), Porto Alegre, Brésil

Nous travaillons avec le Professeur Fernando Moraes et son équipe du groupe GAPH, sur la thématique de la conception des architectures MPSOC, ce qui a donné lieu à plusieurs publications, et contrats de collaboration (CAPES COFECUB).

- Département d'Informatique, Université fédérale du Rio Grande do Sul (**UFRGS**), Porto Alegre, Brésil

Nous travaillons avec Fernanda Lima Kastensmidt, Professeure dans le département d'informatique. Nous collaborons sur les aspects fiabilité des FPGA, dans le cadre de l'accord PHC (CAPES COFECUB) : nous avons déjà publié plusieurs articles sur nos résultats de recherche. Un nouveau contrat de collaboration est actuellement mis en place, qui vise notamment à intégrer le nouveau Centre Spatial Universitaire de l'Université de Montpellier.

- Laboratoire d'Electronique et de Technologie de l'Information, CEA Grenoble (**CEA LETI**) Grenoble, France

Dans le cadre des travaux sur l'optimisation des systèmes intégrés, j'ai eu l'occasion d'effectuer des recherches notamment avec Fabien Clermidy (thèses de Diego Puschini et Imen Mansouri), Didier Lattard (thèse de Camille Jalier), Suzanne Lesecq et Diego Puschini (thèse de Yeter Akgul). Nous avons déposé en copropriété des brevets sur plusieurs de ces contributions (cf. liste de publications).

- **Lab-STICC** CNRS UMR 6285, Université de Bretagne-Sud, France

Sur le thème de la sécurité matérielle, nous avons eu l'occasion de travailler avec l'équipe du Professeur Guy Gogniat, notamment dans le cadre du projet SecreSoC.

- Laboratoire d'Informatique de Paris 6 (**LIP6**), Université Pierre et Marie Curie, Paris, France

Dans le cadre du projet ANR ADAM, nous avons pu mettre en commun notre approche avec l'équipe de François Pêcheux, Professeur dans l'équipe ALSOC - Architecture et Logiciels pour Systèmes Embarqués sur Puce du LIP6.

### 2.4.2 Collaborations industrielles

- Société **INTEL**, Montpellier, France

dans l'accompagnement de plusieurs projets internes à la société (objets connectés, robots de test, téléphonie 4G, réalité augmentée)

- Société **NETHEOS**, Montpellier, France

dans le cadre de l'activité « sécurité matérielle ».

- Société **ST-Microelectronics**, Grenoble, France

dans le cadre de l'activité « architectures adaptatives ».

- Société **CROCUS** Technologie, Grenoble, France

dans le cadre de l'activité « technologies émergentes MRAM » (projet FUI MultiSmart)

- Société **CORTUS**, Montpellier, France

dans l'accompagnement de stages et projets industriels en lien avec Polytech Montpellier

- Société **SATIN**, Montpellier, France

dans l'accompagnement de l'entreprise, stages et projets industriels en lien avec Polytech Montpellier

- Société **ALGODONE**, Montpellier, France

dans l'accompagnement de la création de l'entreprise, en lien avec l'activité de recherche « sécurité matérielle »

- Société **OROSYS**, Montpellier, France

dans l'accompagnement de la création de l'entreprise, stages et projets industriels en lien avec Polytech Montpellier

- Société **SECLAB**, Montpellier, France

dans l'accompagnement de l'entreprise, en lien avec l'activité de recherche « sécurité matérielle », montage d'une convention CIFRE en cours

- Société **BULL-AMESYS** Groupe **ATOS**, Montpellier/Les Clayes-sous-Bois, France

Sur l'activité de recherche « sécurité matérielle », thèse CIFRE en cours (Rémy Druyer)

- Société **NanoExplore**, Montpellier, France

dans l'accompagnement de l'entreprise, stages et projets industriels en lien avec Polytech Montpellier

# 3 BILAN DE LA PRODUCTION SCIENTIFIQUE

Le **Tableau 1** fournit un bilan par année de mes différents types de publications ; revues scientifiques, brevets, ouvrages, contributions à ouvrage (chapitre), et communications dans des conférences.

**Tableau 1.** Récapitulatif chronologique des publications

| Année | 2001 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 2015 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Journal | | 1 | | | 1 | | | 1 | 2 | 2 | 1 | | 2 | 1 | 1 | **12** |
| Brevet | | | | | | | | | 1 | | | 1 | 3 | 1 | | **6** |
| Ouvrage | | | | | | | 1 | 1 | | | | | | | | **2** |
| Chapitre | | | | 1 | | | | | | | 4 | 1 | | | | **6** |
| Com. | 1 | 4 | 5 | 1 | 4 | 2 | 5 | 7 | 5 | 17 | 9 | 5 | 2 | 7 | 3 | **77** |
| Total | **1** | **5** | **5** | **2** | **5** | **2** | **6** | **9** | **8** | **19** | **14** | **7** | **7** | **9** | **4** | **103** |

La liste détaillée par catégorie des mes publications est donnée dans la section 9 de ce chapitre.

En plus de cette production écrite, on notera 8 interventions en tant que conférencier invité :

- « Sécurité des systèmes embarqués », séminaire « Logiciel et systèmes embarqués » organisé par le partenariat JESSICA France - TRANSFERTS LR novembre 2014, Montpellier, France
- « Self-Adaptive Systems: Trends and Challenges », 2014, FETCH, Ottawa, Canada
- « Distributed Approaches for Self-Adaptive Embedded Systems », WORLDCOMP juillet 2012, Las Vegas
- « SECNUM: An Open Environment for developing Side-Channel-Attacks and Countermeasures for Integrated », Sophia Antipolis MicroElectronics Forum, 12 octobre 2011, Sophia Antipolis, France
- « Métriques de caractérisation et de prise de décisions en ligne pour systèmes reconfigurables », Journée « Architectures Reconfigurables », GDR SOC SIP , 17 janvier 2009, Paris, France
- « Exploring Distributed and Scalable MP2SOC », UFRGS/PUCRS, 15 janvier 2008, Porto Alegre, Brésil
- « The Challenges of Hardware and Software Scalability », PUCRS/UFRGS, 8 mai 2007, Porto Alegre , Brésil
- « HS-SCALE - Towards MPSOC Self-Adaptability », Journée « Architectures Reconfigurables », GDR SOC SIP, 29-30 mars 2007, Paris, France,

Analyse de la production scientifique selon les critères de sélectivité du GDR SoC-SiP:

→ RICL[2]: 10, dont 6 revues très sélectives

→ CICL[3] : 51 articles, dont 15 de rang A+, 14 de rang A, et 22 de rang B

Pour information, les indicateurs Google Scholar donnent :

→ Citations : 627

→ Indice h : 12

→ Indice i10 : 19

[2] Revue Internationale avec Comité de Lecture, cf. section 9.2 pour le détail de ces publications
[3] Conférence Internationale avec Comité de Lecture, cf. section 9.6 pour le détail de ces articles

# 4 RESPONSABILITES SCIENTIFIQUES

Ces responsabilités concernent en tout premier lieu l'encadrement doctoral, pour lequel je fournis un bilan détaillé dans ce qui suit, mais aussi le suivi de projets et de stages de Master 2, et enfin un travail autour de l'animation scientifique, qui englobe les comités scientifiques et d'organisation de conférences, les expertises, les jurys de thèse.

## 4.1 Encadrement doctoral

### 4.1.1 Bilan des encadrements de thèses

Le **Tableau 2** fournit le bilan des thèses que j'ai co-encadrées [4]: 9 thèses soutenues et 4 thèses en cours.

**Tableau 2.** Récapitulatif des thèses

| Nom | Titre de la Thèse | Soutenance | Directeur de Thèse |
|---|---|---|---|
| N. Saint-Jean | Etude et conception de systèmes multiprocesseurs auto-adaptatifs pour les systèmes embarqués | Décembre 2008 | M. Robert |
| D. Puschini | Optimisation dynamique et distribuée des architectures MPSOC basée sur la théorie des jeux | Juillet 2009 | L. Torres |
| C. Jalier | Communication et contrôle dans les architectures homogènes de circuits pour les télécommunications | Juillet 2010 | L. Torres |
| N. Hébert | Stratégie de fiabilisation au niveau système des architectures MPSOC | Juillet 2011 | L. Torres |
| G.M. Almeida | Architectures Multi-Processeurs Adaptatives: Principes, Méthodes et Outils | Novembre 2011 | G. Sassatelli |
| I. Mansouri | Contrôle distribué pour les systèmes multi-cœurs auto-adaptatifs | Novembre 2011 | L. Torres |
| L. Barthe | Stratégies pour Sécuriser les Processeurs Embarqués contre les Attaques par Canaux Auxiliaires | Juillet 2012 | L. Torres |
| F. Bruguier | Méthodes de caractérisation des variations pour systèmes reconfigurables adaptatifs | Décembre 2012 | L. Torres |
| Y. Akgul | Gestion de la consommation basée sur l'adaptation dynamique sur les systèmes sur puce en technologie FD-SOI | Décembre 2014 | L. Torres |

[4] Ces thèses ont été encadrées à un taux supérieur ou égal à 30%, cf. partie 4.1.2

| | | | |
|---|---|---|---|
| M. Najem | Méthodes de monitoring de la puissance et de la température des systèmes intégrés | Soutenance prévue en 2015 | G. Sassatelli |
| R. Druyer | Sécurisation des réseaux intégrés sur silicium | Soutenance prévue en 2017 | L. Torres |
| M. El Ahmad | Systèmes intégrés adaptatifs ultra basse consommation pour l'Internet des Objets | Soutenance prévue en 2017 | G. Sassatelli |
| P. Rouget | Étude et conception de mécanismes de rupture et de filtrage de protocoles industriels | Soutenance prévue en 2018 | L. Torres |

Le **Tableau 3** montre la chronologie des principales thèses encadrées

**Tableau 3.** Chronologie de l'encadrement des doctorants

| Année | 2001 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 2015 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Thèse | | | | | | | | | | | | | | | |
| Post-doc | | | | | | | | | | | | | | | |
| NSJ | | | | | | | | | | | | | | | |
| DP | | | | | | | | | | | | | | | |
| CJ | | | | | | | | | | | | | | | |
| IM | | | | | | | | | | | | | | | |
| GMA | | | | | | | | | | | | | | | |
| NH | | | | | | | | | | | | | | | |
| LB | | | | | | | | | | | | | | | |
| FB | | | | | | | | | | | | | | | |
| YA | | | | | | | | | | | | | | | |
| MN | | | | | | | | | | | | | | | |
| RD | | | | | | | | | | | | | | | |
| MEA | | | | | | | | | | | | | | | |
| PR | | | | | | | | | | | | | | | |

J'ai aussi participé[5], dans une moindre mesure, au suivi d'autres thèses. Le Tableau 4 présente une synthèse de ces travaux.

**Tableau 4.** Thèses suivies

| Nom | Titre de la Thèse | Soutenance | Directeur de Thèse |
|---|---|---|---|
| F. Devic | Securing embedded systems based on FPGA technologies | Juillet 2012 | L. Torres |
| R. Busseuil | Exploration d'architecture d'accélérateurs à mémoire distribuée | Décembre 2012 | M. Robert |
| G. Perin | On the Resistance of RSA Countermeasures at Algorithmic, Arithmetic and Hardware Levels | Mai 2014 | P. Maurine |
| M. Kooli | Reliability Analysis of Microprocessor-based Architectures | Soutenance prévue en 2016 | G. Di Natale |

[5] Ces thèses on été ou sont encadrées à un taux inférieur à 20%

### 4.1.2 Présentation détaillée des sujets de thèse

Je présente dans cette section, dans l'ordre chronologique, les différentes thèses soutenues ou en préparation, les sujets, les résumés, ainsi que les publications associées à ces travaux.

#### *4.1.2.1 Thèses Soutenues*

Thèse de : Nicolas Saint-Jean (encadrement : 40%)
Titre : Etude et conception de systèmes multiprocesseurs auto-adaptatifs pour les systèmes embarqués
Soutenance : 16 décembre 2008
Jury :

| | |
|---|---|
| M. ROBERT Michel, | Directeur de Thèse |
| M. MORAES Fernando, | Rapporteur |
| M. GOGNIAT Guy, | Rapporteur |
| M. PAINDAVOINE Michel, | Examinateur |
| M. CLERMIDY Fabien, | Examinateur |
| M. SASSATELLI Gilles, | Examinateur |
| M. BENOIT Pascal, | Examinateur |

Publications associées : [J8][J9][C50] [C53] [C54] [C55] [C56] [C57] [C58] [C59] [C60]
Situation actuelle du diplômé : Ingénieur R&D, Fogale Nanotech
Résumé : Cette thèse se place volontairement dans un contexte futuriste où la complexité des systèmes sur puce a augmenté de façon exponentielle, où la technologie ne garantit plus la stabilité de ses paramètres, et où le nombre de transistors implantés oblige à repenser l'amélioration des performances architecturales en termes de multiplication des cœurs de calcul. L'architecture cible de cette thèse est une architecture massivement parallèle (plus de 100 éléments de calcul complexes). La maîtrise de ces architectures est un élément essentiel pour assurer la compétitivité des futurs systèmes embarqués. Cette thèse propose l'architecture HS-Scale composé un ensemble de briques de base permettant d'aller vers des architectures auto adaptatives, c'est à dire capables de réagir et de s'adapter à leur environnement extérieur et à leur état interne sans intervention extérieure.

---

Thèse de : Diego Puschini Pascual (encadrement : 50%)
Titre : Optimisation dynamique et distribuée des architectures MPSOC basée sur la théorie des jeux
Soutenance : 2 juillet 2009
Jury :

| | |
|---|---|
| M. Michel ROBERT, | Président |
| M. Lionel TORRES, | Directeur de thèse |
| M. Michel AUGUIN, | Rapporteur |
| M. Frédéric PÉTROT, | Rapporteur |
| M. Jürgen BECKER, | Examinateur |
| M. Christian BESSIÈRE, | Examinateur |
| M. Fabien CLERMIDY, | Examinateur |
| M. Pascal BENOIT, | Examinateur |
| M. Russell TESSIER, | Invité |

Publications associées : [J7][J10][B5] [B6] [CO5] [C49] [C51] [C52] [C44] [C45]
Situation actuelle du diplômé : Chercheur au CEA
Résumé : La complexité des Systèmes-sur-Puce () a exponentiellement augmenté, les technologies de pointe ne garantissent plus la stabilité des paramètres, et les contraintes des applications obligent à améliorer la performance architecturale. Nous considérons des  intégrant plusieurs éléments de traitement (MPSOC). Ces plates-formes sont conçues pour traiter des applications avec plusieurs contraintes, telles que télécom et multimédia. L'adaptabilité dynamique est alors obligatoire pour optimiser la puissance consommée face aux changements du système, par exemple l'évolution d'une application à une autre. L'optimisation distribuée est nécessaire pour assurer l'adaptabilité. L'absence

des techniques dynamiques et distribuées pour les MPSOC nous a conduits à proposer un modèle où chaque processeur peut prendre des décisions locales. Nous employons la théorie des jeux pour décrire l'optimisation distribuée. Les processeurs sont considérés comme joueurs essayant de trouver la meilleure configuration dans un jeu non coopératif, en optimisant plusieurs métriques (performance, puissance, latence, température). La théorie des jeux donne un ensemble de formalismes pour étudier la convergence d'un tel système vers une solution optimale ou quasi-optimale. Basé sur ces concepts, nous présentons une technique innovatrice pour optimiser les MPSOC d'une manière distribuée et dynamique. Nos analyses vérifient l'adéquation d'un tel modèle pour décrire les MPSOC. Une implémentation matérielle a été proposée et évaluée. Cette implémentation vise à proposer un bloc distribué et *scalable* avec une basse complexité capable d'être intégré dans de futures architectures MPSOC adaptatives.

---

Thèse de : Camille Jalier (encadrement : 30%)
Titre : Communication et contrôle dans les architectures homogènes de circuits pour les télécommunications
Soutenance : 5 Juillet 2010
Jury :

| | |
|---|---|
| M. Ahmed A. JERRAYA, | Président |
| M. Lionel TORRES , | Directeur de thèse |
| M. Guy GOGNIAT, | Rapporteur |
| M. Christophe MOY, | Rapporteur |
| M. Michel ROBERT , | Examinateur |
| M. Gilles SASSATELLI, | Examinateur |
| M. Pascal BENOIT, | Examinateur |
| M. Didier LATTARD, | Examinateur |
| M. Eric FLAMAND, | Invité |

Publications associées : [C42][C35][C34] [C33]
Situation actuelle du diplômé : Ingénieur R&D, Kalray
Résumé : Les travaux de thèse s'intéressent à la problématique de contrôle et de communication dans le domaine de la conception des systèmes numériques embarqués pour les applications de télécommunication de quatrième génération. La complexité des applications couplée aux besoins de productivité croissants impose de repenser les méthodologies de conception et les architectures sous jacentes. Afin de lever ces verrous, nous proposons plusieurs contributions originales. En effet, une méthodologie d'exploration d'un espace de conception ainsi qu'une architecture basée sur des nœuds de traitements homogènes et flexibles interconnectés à travers un réseau sur silicium sont proposées. Chaque nœud de traitement possède plusieurs blocs visant à exécuter efficacement et dynamiquement les applications de télécommunication. Pour répondre aux contraintes de faible consommation, nous proposons plusieurs solutions innovantes afin de minimiser cette métrique notamment au travers de techniques de migration de tâches.

---

Thèse de : Nicolas Hébert (encadrement 40%)
Titre : Stratégie de fiabilisation au niveau système des architectures MPSOC
Soutenance : 6 Juillet 2011
Jury :

| | |
|---|---|
| Pr. Daniel ETIEMBLE, | Président |
| Pr. Lionel TORRES, | Directeur de Thèse |
| Pr. Régis LEVEUGLE , | Rapporteur |
| D.R. Jean-Philippe DIGUET, | Rapporteur |
| M. Didier FUIN, | Examinateur |
| Dr. Pascal BENOIT, | Examinateur |
| Dr. Gilles SASSATELLI, | Examinateur |

Publications associées : [C27] [C28] [C29] [C47] [C48]
Situation actuelle du diplômé : Ingénieur R&D
Résumé : Cette thèse s'inscrit dans un contexte où chaque saut technologique, voit apparaitre des circuits intégrés produits de plus en plus tôt dans la phase de qualification et où la technologie de ces circuits intégrés se rapproche de plus en plus des limitations physiques de la matière. Malgré des contre-mesures technologiques, on se retrouve devant un taux de défaillance grandissant ce qui crée des conditions favorables au retour des techniques de tolérance aux fautes sur les circuits intégrés non critiques. La densité d'intégration atteinte aujourd'hui nous permet de considérer les réseaux reconfigurables de processeur comme des architectures d'avenir. En effet, l'homogénéité de ces architectures laisse entrevoir des reconfigurations possibles de la plateforme qui permettraient d'assurer une qualité de service et donc une fiabilité minimum en présence de défauts. Ainsi, de nouvelles solutions de protection doivent être proposées pour garantir le bon fonctionnement des circuits non plus uniquement au niveau de quelques sous-fonctionnalités critiques mais au niveau architecture système lui-même. En s'appuyant sur ces prérogatives, nous présentons une méthode de protection distribuée et dynamique innovatrice, D-Scale. La méthode consiste à détecter, isoler et recouvrir les systèmes en présence d'erreurs de type « crash ». La détection des erreurs qui ont pour conséquence un « crash » de la plateforme est basée sur un mécanisme de messages de diagnostique échangés entre les unités de traitement. La phase de recouvrement est quant à elle basée sur un mécanisme permettant la reconfiguration de la plateforme de manière autonome. Une implémentation de cette protection matérielle et logicielle est proposée. Le coût de protection est réduit afin d'être intégré dans de futures architectures multiprocesseurs. Finalement, un outil d'évaluation d'impacte des fautes sur la plateforme est aussi étudié afin de valider l'efficacité de la protection.

---

Thèse de : Gabriel Marchesan Almeida (encadrement : 30%)
Titre : Architectures Multi-Processeurs Adaptatives: Principes, Méthodes et Outils
Soutenance : 21 Novembre 2011
Jury :

| | |
|---|---|
| Gilles SASSATELLI, | Directeur de Thèse |
| Christophe JEGO, | Rapporteur |
| Michel PAINDAVOINE, | Rapporteur |
| Michael HÜBNER, | Examinateur |
| Fernando GEHM MORAES, | Examinateur |
| Pascal BENOIT, | Examinateur |
| Michel ROBERT, | Président |

Publications associées : [J4] [J5] [J8] [C18] [C22 [C23] [C25] [C28] [C29] [C32] [C36] [C38] [C41]
Situation actuelle du diplômé : Ingénieur R&D, LEICA, Allemagne
Résumé : Les systèmes multiprocesseurs sur puce (MPSOC) offrent des performances supérieures tout en conservant la flexibilité et la réutilisabilité grâce à la customisation du logiciel embarqué. Alors que la plupart de MPSOC sont aujourd'hui hétérogènes pour mieux répondre aux besoins des applications ciblées, les MPSOC homogènes pourraient devenir dans un proche avenir une alternative viable apportant d'autres avantages tels que l'équilibrage de charge de l'exécution, la migration des tâches et

l'ajustement de fréquence dynamique. Cette thèse s'appuie sur une plateforme MPSOC homogène, développée pour explorer techniques d'adaptation en ligne. Chaque processeur de ce système est compact et exécute un système d'exploitation préemptif qui surveille diverses métriques et est habilité à prendre des décisions de *remapping* grâce à des techniques de migration de code et du changement dynamique de la fréquence. Cette approche permet la mise en œuvre des capacités de raffinage d'application à l'exécution en fonction de différents critères.

---

Thèse de : Imen Mansouri (encadrement : 50%)
Titre : Contrôle distribué pour les systèmes multi-cœurs auto-adaptatifs
Soutenance : 30 Novembre 2011
Jury :

| | |
|---|---|
| M. Fabien Clermidy, | Examinateur |
| M. Frédéric Pétrot, | Rapporteur |
| M. Lionel Torres, | Directeur de Thèse |
| M. Michel Auguin, | Rapporteur |
| M. Olivier Sentieys, | Examinateur |
| M. Pascal Benoit, | Examinateur |

Publications associées : [J3][J7][B2] [B3] [B4] [C30] [C33]
Situation actuelle du diplômé : Ingénieur R&D, Bull
Résumé : Les architectures régulières intégrant plusieurs cœurs de traitement sont davantage utilisées dans les systèmes embarqués. Dans cette thèse, on s'intéresse aux mécanismes d'optimisation d'énergie dans des architectures avec une dimension étendue; pour faire face aux problèmes de variabilité technologique et aux changements du contexte applicatif, le processus d'optimisation se déroule en temps réel. Des capteurs in-situ détectent le degré de dégradation du circuit. Quant a la variabilité applicative, des moniteurs d'activité sont insérés sur un niveau architectural pour estimer la charge de travail engendrée par l'application en cours et la consommation qui en découle. Nous avons développé une méthode systématique pour l'intégration de ces capteurs avec un moindre coût en surface. Leurs sorties alimentent un processus d'optimisation basé sur la théorie de consensus et dupliqué dans chaque cœur. Ce contrôle vise à fixer la meilleure configuration locale à chaque cœur permettant d'optimiser la consommation globale du système tout en respectant les contraintes temps réel de l'application en cours. Ce schéma opère d'une manière complètement distribuée afin de garantir la scalabilité de notre solution, et donc sa faisabilité, compte tenu de la complexité des circuits actuels et futurs.

---

Thèse de : Lyonel Barthe (encadrement : 60%)
Titre : Stratégies pour Sécuriser les Processeurs Embarqués contre les Attaques par Canaux Auxiliaires
Soutenance : 10 Juillet 2012
Jury :

| | |
|---|---|
| Lionel Torres, | Directeur de Thèse |
| Pascal Benoit, | Examinateur |
| Jean-Luc Danger, | Rapporteur |
| François-Xavier Standaert, | Rapporteur |
| Jean-Claude Bajard, | Examinateur |
| Jacques Blanc-Talon, | Examinateur |

Publications associées : [J1] [CO2] [C18] [C19] [C24] [C37] [C43]
Situation actuelle du diplômé : Ingénieur R&D THALES, France
Résumé : Les attaques par canaux auxiliaires telles que l'analyse différentielle de la consommation de courant (DPA) et l'analyse différentielle des émissions électromagnétiques (DEMA) constituent une menace sérieuse pour la sécurité des systèmes embarqués. L'objet de cette thèse est d'étudier les

vulnérabilités des implantations logicielles des algorithmes cryptographiques face à ces attaques pour concevoir un processeur d'un nouveau type. Pour cela, nous commençons par identifier les différents éléments des processeurs embarqués qui peuvent être exploités pour obtenir des informations secrètes. Puis, nous introduisons des stratégies qui privilégient un équilibre entre performance et sécurité pour protéger de telles architectures au niveau transfert de registres (RTL). Nous présentons également la conception et l'implantation d'un processeur sécurisé, le SecretBlaze-SCR. Enfin, nous évaluons l'efficacité des solutions proposées contre les analyses électromagnétiques globales et locales à partir de résultats expérimentaux issus d'un prototype du SecretBlaze-SCR réalisé sur FPGA. A travers cette étude de cas, nous montrons qu'une combinaison appropriée de contre-mesures permet d'accroître significativement la résistance aux analyses par canaux auxiliaires des processeurs tout en préservant des performances satisfaisantes pour les systèmes embarqués.

---

Thèse de : Florent Bruguier (encadrement 70%)
Titre : Méthodes de caractérisation et de surveillance des variations technologiques et environnementales pour systèmes reconfigurables adaptatifs
Soutenance : 20 Décembre 2012
Jury :

| | |
|---|---|
| Lionel Torres, | Directeur de Thèse |
| Pascal Benoit, | Examinateur |
| Guy Gogniat, | Rapporteur |
| Marc Belleville, | Rapporteur |
| Laurent Dusseau, | Examinateur |
| Philippe Maurine, | Invité |

Publications associées : [J1][J2][J5] [C3] [C4] [C5] [C8] [C9] [C10] [C12] [C18] [C19] [C20] [C21] [C39] [C40]
Situation actuelle du diplômé : Ingénieur de Recherche, LIRMM
Résumé : Les circuits modernes sont de plus en plus sensibles aux variations technologiques et environnementales qui n'ont plus seulement un effet global sur les circuits mais aussi un effet local sur ceux-ci. Dans ce contexte, les composants reprogrammables que sont les FPGA représentent un support technologique intéressant. En effet, ces composants permettent d'adapter l'implantation physique du système grâce à une simple reconfiguration du circuit. C'est pourquoi, dans ce manuscrit, nous présentons un flot d'adaptation complet visant à compenser les variations des circuits reconfigurables. Pour cela, une étude de toutes les phases de conception des capteurs numériques est réalisée. Nous proposons ensuite une approche originale et unique de caractérisation basée sur l'analyse électromagnétique. Il est notamment montré que cette approche permet de se défaire des biais de mesure engendrés par les méthodes de mesure directe. L'utilisation conjointe des capteurs et de cette méthode d'analyse permet une caractérisation fine et précise des variations technologiques de n'importe quel type de circuit FPGA. Enfin, la cartographie issue de la phase de caractérisation permet ensuite de calibrer les capteurs pour une utilisation en ligne. Nous utilisons donc ensuite ces capteurs pour le monitoring dynamique d'un système MPSOC.

---

Thèse de : Yeter Akgul (encadrement 50%)
Titre : Gestion de la consommation basée sur l'adaptation dynamique de la tension, fréquence et body bias sur les systèmes sur puce en technologie FD-SOI
Soutenance : 9 décembre 2014
Jury :

| | |
|---|---|
| M. François Pêcheux, | Rapporteur |
| M. Jean-Philippe Diguet, | Rapporteur |
| M. Frédéric Pétrot, | Examinateur |
| M. Gilles Sassatelli, | Examinateur |
| Mme Suzanne Lesecq, | Examinateur |
| M. Sylvain Clerc, | Invité |
| M. Lionel Torres, | Directeur de Thèse |

M. Pascal Benoit, Examinateur
M. Diego Puschini, Examinateur

Publications associées : [B1] [C1] [C6] [C11] [C16]
Situation actuelle du diplômé : Post-doc au LAB-STICC
Résumé : Au-delà du noud technologique CMOS BULK 28nm, certaines limites ont été atteintes dans l'amélioration des performances en raison notamment d'une consommation énergétique devenant trop importante. C'est une des raisons pour lesquelles de nouvelles technologies ont été développées, notamment celles basées sur Silicium sur Isolant (SOI). Par ailleurs, la généralisation des architectures complexes de type multi-cours, accentue le problème de gestion de la consommation à grain-fin. Les technologies CMOS FD-SOI offrent de nouvelles opportunités pour la gestion de la consommation en permettant d'ajuster, outre les paramètres usuels que sont la tension d'alimentation et la fréquence d'horloge, la tension de *body bias*. C'est dans ce contexte que ce travail étudie les nouvelles possibilités offertes et explore des solutions innovantes de gestion dynamique de la tension d'alimentation, fréquence d'horloge et tension de *body bias* afin d'optimiser la consommation énergétique des systèmes sur puce. L'ensemble des paramètres tensions/fréquence permet une multitude de points de fonctionnement, qui doivent satisfaire des contraintes de fonctionnalité et de performance. Ce travail s'intéresse donc dans un premier temps à une problématique de conception, en proposant une méthode d'optimisation du placement de ces points de fonctionnement. Une solution analytique permettant de maximiser le gain en consommation apporté par l'utilisation de plusieurs points de fonctionnement est proposée. La deuxième contribution importante de cette thèse concerne la gestion dynamique de la tension d'alimentation, de la fréquence et de la tension de *body bias*, permettant d'optimiser l'efficacité énergétique en se basant sur le concept de convexité. La validation expérimentale des méthodes proposées s'appuie sur des échantillons de circuits réels, et montre des gains en consommation moyens allant jusqu'à 35%.

#### *4.1.2.2 Thèses en Cours*

Thèse de : Mohamad Najem (encadrement : 80%)
Titre : Méthodes de monitoring de la puissance et de la température des systèmes intégrés
Soutenance : prévue en novembre 2015
Directeur de Thèse : Gilles Sassatelli
Publications associées : [C5], 1 article de revue en révision, 1 article de conférence accepté – en cours de publication, 1 article de conférence soumis

Thèse de : Rémy Druyer (encadrement : 50%)
Titre : Sécurisation des réseaux intégrés sur silicium
Soutenance : prévue en mars 2017
Directeur de Thèse : Lionel Torres
Publications associées : 1 article de conférence accepté, en cours de publication

Thèse de : Mohamad El Ahmad (encadrement : 80%)
Titre : Systèmes intégrés adaptatifs ultra basse consommation pour l'Internet des Objets
Soutenance : prévue en octobre 2017
Directeur de Thèse : Gilles Sassatelli
Publications associées : 1 article de revue en révision, 1 article de conférence accepté, en cours de publication

Thèse de : Peter Rouget (encadrement : 90%)
Titre : Étude et conception de mécanismes de rupture et de filtrage de protocoles industriels
Soutenance : prévue en octobre 2018
Directeur de Thèse : Lionel Torres

## 4.2 Encadrement de Stages de Master 2

L'encadrement de projets et de stages et bien évidemment au cœur de mes activités, notamment à Polytech Montpellier, où j'accompagne chaque année en moyenne 4 à 5 projets de fin d'études (qui se déroulent au premier semestre de l'année universitaire) et environ 3 stages en entreprise (2ème semestre), où il s'agit principalement de suivre à distance le bon déroulement des choses. J'encadre aussi régulièrement des stages de Master 1 EEA depuis 2006. Ces projets sont souvent liées à mes activités de recherche, notamment sur des développements techniques, et font également partie de l'aspect « formation » de la fonction d'enseignant-chercheur. Les stages de Master 2 « recherche » sont quant à eux détaillés ci-dessous :

Stage de : C. Jalier
Sujet du Stage : Prototypage de système MPSOC sur plateforme FPGA
Période : février 2006 à juillet 2006

Stage de : F. Bruguier
Sujet du Stage : Intégration Silicium d'une Architecture Multi-processeur Tolérante aux Variations de Procédé
Période : mars 2009 à juillet 2009

Stage de : H. Karray
Sujet du Stage : Dynamic reconfiguration on MPSOC architecture
Période : mars 2009 à juillet 2009

Stage de : R. Hong
Sujet du Stage : Développement d'application en C pour processeur embarqué sur cible FPGA
Période : mars 2011 à juillet 2011

Stage de : K. Hacini
Sujet du Stage : Caractérisation du vieillissement de circuits reconfigurables
Période : mars 2014 à juillet 2014

Stage de : M. El Ahmad
Sujet du Stage : Application des techniques et des méthodes du Data Mining pour la prédiction de la consommation et de la température des systèmes intégrés
Période : mars 2014 à juillet 2014

## 4.3 Autres Responsabilités Scientifiques

Je termine cette quatrième partie par les différents jurys de thèses auxquels j'ai participé, les comités scientifiques de conférences et de revues, l'organisation de congrès et la présidence de sessions, ainsi que les expertises scientifiques réalisées.

### 4.3.1 Participation à des Jurys de Thèses

Thèse de Nicolas Saint-Jean– **Examinateur**
soutenue le 16 décembre 2008 au LIRMM, Université de Montpellier 2

Thèse de Diego Puschini – **Examinateur**
soutenue le 2 juillet 2009 au LIRMM, Université de Montpellier 2

Ashok Chandra Sekaran – **Rapporteur**,
« A Location Aware Wireless Sensor Network for Assisting Emergency Response to Disasters »
soutenue le 26 novembre 2009, Université de Karlsruhe, Allemagne

Thèse de Camille Jalier– **Examinateur**
soutenue le 5 juillet 2010 au CEA LETI, Grenoble

Thèse de Nicolas Hébert – **Examinateur**
soutenue le 6 juillet 2011 au LIRMM, Université de Montpellier 2

Thèse de Gabriel Marchesan Almeida – **Examinateur**
soutenue le 21 novembre 2011 au LIRMM, Université de Montpellier 2

Thèse de Imen Mansouri – **Examinateur**
soutenue le 30 novembre 2011 au LIRMM, Université de Montpellier 2

Thèse de Lyonel Barthe – **Examinateur**
soutenue le 10 juillet 2012 au LIRMM, Université de Montpellier 2

Thèse de Florent Bruguier – **Examinateur**
soutenue le 20 décembre 2012 au LIRMM, Université de Montpellier 2

Thèse de Yeter Akgul– **Examinateur**
soutenue le 9 décembre 2014 au CEA LETI, Grenoble

Thèse de Abdulazim Amouri – **Rapporteur**,
« Degradation in FPGAs: Monitoring, Modeling and Mitigation »
soutenue le 23 avril 2015, Université de Karlsruhe, Allemagne

### 4.3.2 Participation à des Comités de Programmes de Conférences

IEEE International Conference on Field Programmable Logic in Europe (FPL) :

*2005, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015*

International Symposium on Reconfigurable Communication-centric Systems-on-Chip

*2006, 2007, 2008, 2010, 2011, 2012, 2013, 2014, 2015*

IEEE IPDPS, Reconfigurable Architecture Workshop (RAW) :

*2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014*

IEEE International Conference on ReConFigurable Computing and FPGAs

*2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015*

IEEE International NEWCAS Conference

*2012, 2013, 2014, 2015*

IEEE International Symposium on VLSI (ISVLSI)

*2006, 2007, 2008, 2014*

IFIP/IEEE International Conference on Very Large Scale Integration

*2012, 2013, 2014*

Reconfigurable Computing (RC-Education)

*2009, 2010, 2011*

IEEE International Symposium on VLSI (ISVLSI) – PhD Forum

*2011, 2012, 2013*

Southern Programmable Logic Conference

*2012, 2014, 2016*

IEEE Design Automation and Test in Europe

*2006*

MAJECSTIC

*2003*

IEEE Symposium on Industrial Embedded Systems

*2008*

IEEE Rapid System Prototyping (RSP)

*2005*

Workshop on Intelligent Solutions in Embedded Systems (WISES)

*2005*

International Conference on Advances in Electronics and Micro-electronics (ENICS)

*2008*

International Embedded Systems Symposium (IESS)

*2005*

International Conference on Microelectronics Systems Education (MSE)

*2005*

IEEE International Conference on Application-specific Systems, Architectures and Processors

*2013*

IEEE International Symposium on Access Spaces

*2011*

### 4.3.3 Présidence de Sessions de Conférences

| | |
|---|---|
| 2006 | IEEE ISVLSI, Karlsruhe, Allemagne |
| 2006 | ReCoSoC, Montpellier, France |
| 2007 | ReCoSoC, Montpellier, France |
| 2008 | IEEE ISVLSI, Montpellier, France |
| 2008 | IEEE RECONFIG, Cancun, Mexique |
| 2010 | ReCoSoC, Karlsruhe, Allemagne |
| 2010 | VARI, Montpellier, France |
| 2011 | ReCoSoC Montpellier, France |
| 2011 | IPDPS, RAW, Anchorage, USA |
| 2014 | ReCoSoC Montpellier, France |

### 4.3.4 Participation à des Comités de Lecture de Revues

| | |
|---|---|
| JMM – Elsevier Microprocessors and Microsystems | depuis 2008 |
| TECS – ACM Transactions on Embedded Computing Systems | depuis 2008 |
| JOLPE – Journal of Low Power Electronics | depuis 2010 |
| TVLSI – IEEE Transactions on Very Large Scale Integration Systems | depuis 2009 |
| TC – IEEE Transations on Computers | depuis 2015 |

### 4.3.5 Participation à l'Organisation de Congrès

| | |
|---|---|
| 2002 | International Conference on Field Programmable Logic and Application (FPL)<br>La Grande Motte, France, 2-4 septembre 2002<br>Organisation locale |
| 2006 | RECOSOC<br>Montpellier, France, 3-5 juillet 2006<br>**Industrial Chair** |
| 2007 | RECOSOC<br>Montpellier, France, 18-20 juin 2007<br>**Program Co-Chair** |
| 2008 | IEEE International Symposium on VLSI (ISVLSI)<br>Montpellier, France, 7-9 avril 2008<br>**Local Chair / Finance Chair** |
| 2008 | IEEE International Symposium on Industrial Embedded Systems (SIES)<br>La Grande Motte, France, 11-13 juin 2008<br>**Finance Chair** |
| 2008 | IEEE International Conference on ReConFigurable Computing and FPGAs<br>Cancun, Mexico, 3-5 Décembre2008<br>**Track Chair « *Self-adaptive Computing* »** |

2010 Reconfigurable Computing Education (RC-Education 2010)
Karlsruhe, Allemagne, 19 mai 2010
**Program Chair**

2011 IEEE IPDS, Reconfigurable Architectures Workshop (RAW)
Anchorage, USA, 16-17 mai 2011
**Program Chair**

2011 Reconfigurable Computing Education (RC-Education 2011)
Karlsruhe, Allemagne, 22 juin 2011
**Program Chair**

2012 IEEE IPDS, Reconfigurable Architectures Workshop (RAW)
Shangai, Chine, 21-22 mai 2012
**Publicity Chair**

2013 IEEE IPDS, Reconfigurable Architectures Workshop (RAW)
Boston, USA, 20-21 mai 2013
**Publicity Chair**

### 4.3.6 Expertises

2006 ANR Architectures du Futur

2007 ANR Architectures du Futur

2011 Membre du pool d'experts de la section 61 de l'Université de Montpellier

2011 Membre du comité de sélection MCF61 Département Microélectronique / IUT de Montpellier, Université de Montpellier

2011 Membre du comité de sélection MCF61/63 de l'Université de Bretagne Sud

2012 Membre du comité de sélection MCF61 Département Microélectronique / IUT de Nîmes, Université de Montpellier

2012 Membre du comité de sélection MCF63 à l'IPB, Bordeaux

2013 Membre du Conseil d'Administration de l'association Media Cloud Cluster (regroupement de sociétés dans le domaine des objets connectés)

2014 ANR / Appel à projets générique (2$^{ème}$ phase)

2015 ANR / Appel à projets générique (2$^{ème}$ phase)

2015 Membre du Conseils d'Orientation Scientifiques, Techniques et Industriels (COSTI) du Languedoc-Roussillon

2015 Membre du comité de sélection MCF61 Département Microélectronique / IUT de Nîmes, Université de Montpellier

# 5 CONTRATS DE RECHERCHE

Cette section rassemble l'ensemble des contrats auxquels j'ai participé et précise pour chacun d'entre eux la nature, le titre et les partenaires.

## 5.1 Bilan Chronologique

Le Tableau 5 présente les différents contrats de recherche auxquels j'ai participé et je participe actuellement.

**Tableau 5.** Récapitulatif des contrats de recherche

| Année | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CEA | | | | | | | | | | | | | | | |
| ADAM | | | | | | | | | | | | | | | |
| STM | | | | | | | | | | | | | | | |
| MoKa | | | | | | | | | | | | | | | |
| Secresoc | | | | | | | | | | | | | | | |
| Modern | | | | | | | | | | | | | | | |
| DGA | | | | | | | | | | | | | | | |
| Netheos | | | | | | | | | | | | | | | |
| SECNUM | | | | | | | | | | | | | | | |
| FINMINA | | | | | | | | | | | | | | | |
| CLERECO | | | | | | | | | | | | | | | |
| Bull | | | | | | | | | | | | | | | |
| MultiS | | | | | | | | | | | | | | | |
| SECLAB | | | | | | | | | | | | | | | |

## 5.2 Présentation des Contrats

**CEA/LETI**
Nature du contrat : Contrat de collaboration / accompagnement des thèses
Titre du contrat : « Etude et validation d'architectures Multiprocesseurs tolérantes aux pannes »
Montant pour le LIRMM : 50 k€ (sur 24 mois)
Période du contrat : 2012-2014
Rôle : participant

**ADAM**
Nature du contrat : ANR, programme Architectures du Futur
Titre du contrat : « ADAPTIVE DYNAMIC ARCHITECTURE FOR MP$^2$SOC »
Partenaires : LIP6 (porteur), CEA/LETI, LIRMM
Montant pour le LIRMM : 215000 k€
Période du contrat : 2007-2010
Rôle : participant

**STM Grenoble**
Nature du contrat : Convention de collaboration industrielle + Bourse Cifre
Titre du contrat : « Stratégies de fiabilisation des MPSOC »
Partenaires : STM, LIRMM
Montant pour le LIRMM : 30 k€
Période du contrat : 2008-2011
Rôle : participant

**MoKa**
Nature du contrat : PHC PROCOPE
Titre du contrat : « Architectures multiprocesseurs reconfigurables »

Partenaires : DGA, LIRMM
Période du contrat : 2009-2012
Montant pour le LIRMM : 10 k€
Rôle : responsable du projet

**SECRESOC**
Nature du contrat : ANR, programme ARPEGE
Titre du contrat : « SECRESOC: Systèmes sur puce reconfigurables pour la sécurisation de données »
Partenaires : R. Fouquet/V. Fischer Univ. St Etienne, 2009-2013, Laboratoire LabSTICC, Paris'Tech, Laboratoire Hubert Curien, Société Netheos, LIRMM
Montant pour le LIRMM : 170 k€
Période du contrat : 2009-2013
Rôle : participant

**ENIAC MODERN**
Nature du contrat : Projet CEE ENIAC
Titre du contrat : « MOdeling and DEsign of Reliable, process variation-aware Nanoelectronic devices, circuits and systems »
Partenaires : 30 partennaires, 11 M€.
Montant pour le LIRMM : 500 k€
Période du contrat : 2008-2011
Rôle : participant

**DGA**
Nature du contrat : Allocation de recherche
Titre du contrat : « Architectures reconfigurables sécurisées »
Partenaires : DGA, LIRMM
Période du contrat : 2008-2011
Rôle : responsable du projet

**NETHEOS**
Nature du contrat : Convention de collaboration industrielle + Bourse Cifre
Titre du contrat : « Sécurisation des configurations des circuits FPGA »
Partenaires : Netheos, LIRMM
Montant pour le LIRMM : 20 k€
Période du contrat : 2009-2012
Rôle : participant

**SECNUM**
Nature du contrat : GEPETOS
Titre du contrat : « Sécurité Numérique »
Partenaires : LIRMM
Période du contrat : 2009-2012
Montant pour le LIRMM : 350 k€
Rôle : participant

**SECNUM**
Nature du contrat : ARPE
Titre du contrat : « Sécurité Numérique »
Partenaires : LIRMM
Période du contrat : 2012-2013
Montant pour le LIRMM : 30 k€
Rôle : participant

**FINMINA**
Nature du contrat : Projet IDEFI
Titre du contrat : « Formations Innovantes en Micro et Nanoélectronique »
Partenaires : GIP-CNFM
Période du contrat : 2012-2020
Montant pour le LIRMM/action SECNUM : 500 k€
Rôle : participant

**CLERECO**
Nature du contrat : Projet CEE FP7
Titre du contrat : « Cross-Layer Early Reliability Evaluation for the Computing cOntinuum »
Partenaires : POLITO, UoA, CNRS, UPC, THALES, INTEL, ABB, YOGITECH
Période du contrat : 2013-2016
Montant pour le LIRMM : 380 k€
Rôle : participant

**BULL-AMESYS**
Nature du contrat : Convention de collaboration industrielle + Bourse Cifre
Titre du contrat : « Sécurisation des réseaux intégrés sur Silicium »
Partenaires : STM, LIRMM
Montant pour le LIRMM : 35 k€
Période du contrat : 2013-2016
Rôle : participant

**MULTISMART**
Nature du contrat : FUI
Titre du contrat : « Plateforme sécurisée à mémoire magnétique Multi-bits pour Smart cards »
Partenaires : Porteur société CROCUS, partenaires LIRMM, IM2NP, GEMALTO, INVIA, CROCUS
Montant pour le LIRMM : 380 k€
Période du contrat : 2015-2018
Rôle : participant

**SECLAB**
Nature du contrat : Convention de collaboration industrielle + Bourse Cifre
Titre du contrat : « Etude et conception de mécanismes de rupture de protocoles industriels »
Partenaires : Seclab, LIRMM
Montant pour le LIRMM : en cours
Période du contrat : 2014-2017
Rôle : responsable du projet

# 6 ACTIVITES D'ENSEIGNEMENT

J'exerce mes activités d'enseignement dans le département d'Electronique et d'Informatique Industrielle (EII), de l'école Polytechnique Universitaire de Montpellier (Polytech Montpellier). J'interviens dans les deux spécialités du département : Microélectronique et Automatique (MEA) et Systèmes Embarqués (SE, formation par apprentissage), qui est une voie d'accès au diplôme qui existe depuis septembre 2012.

## 6.1 Bilan des heures d'enseignement

Le Tableau 6 fournit un bilan de la répartition des heures d'enseignements que j'ai effectuées depuis 2001. J'ai eu l'opportunité d'enseigner dès le début de ma thèse, en intervenant en DEUG STPI de 2001 à 2004 dans le cadre d'un monitorat.

**Tableau 6.** Récapitulatif chronologique des heures d'enseignements

| | **Monitorat** | **Maître de Conférences** | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 01 - 04 | 05 - 06 | 06 - 07 | 07 - 08 | 08 - 09 | 09 - 10 | 10 - 11 | 11 - 12 | 12 - 13 | 13 - 14 | 14 - 15 | **Total** |
| hETD | 198 | 198 | 264 | 219 | 130* | 238 | 314 | 320 | 273 | 303 | 332 | **2789** |

*décharge de service de 64h obtenue pour l'année 2008-2009

## 6.2 Détails des enseignements

Les enseignements que je dispense s'inscrivent dans les thématiques portées par la section CNU 61. Ils abordent l'informatique industrielle : programmation en C, compilation, logique combinatoire et séquentielle, langage VHDL, architecture des calculateurs, systèmes embarqués, FPGA et la conception des circuits intégrés digitaux.

J'ai eu l'occasion depuis le début de ma carrière de développer de nombreux cours, supports de cours, Travaux Pratiques, projets, etc. Je mets régulièrement à jour l'ensemble des supports et adapte mes méthodes pédagogiques, en essayant d'être innovant dans mon approche de l'enseignement (cours intégrés, utilisation de l'iPad[6] avec notamment mise en place de Quizz, apprentissage par problème, programme « un étudiant, une carte FPGA »[7], organisation de « design contest », etc.). Je suis intervenu dans de nombreux enseignements depuis 10 ans (langage C, C++, microcontrôleurs, synthèse et placement routage des circuits intégrés numériques). Je développe dans la suite les cours les plus récents mis en place.

### 6.2.1 Logique combinatoire et séquentielle

L'objectif de cet enseignement est de présenter les notions élémentaires de logique combinatoire (représentation, codage, minimisation) et séquentielle (bascules registres, compteurs) afin de pouvoir concevoir à l'aide de langages HDL des composants numériques de base. Les notions suivantes sont abordées : algèbre de Boole, représentation des fonctions logiques, circuits combinatoires et séquentiels, VHDL. La mise en oeuvre est faite sur composants FPGA.

Ce module, dispensé en 3ème année, aborde les points suivants :

- Rappels des fonctions logiques élémentaires.
- Notions de calcul, bases, compléments, opérateurs arithmétiques.

[6] Depuis 3 ans, Polytech met à disposition des iPad aux étudiants à leur entrée à l'école. Des actions pédagogiques sont menées pour développer de nouvelles pratiques intégrant cet outil

[7] Initiative développée à l'échelle locale en 2009, c'est maintenant un programme national que je porte avec un support du CNFM, de Xilinx et Altera

- Codage BCD / Gray
- Les bascules
- Les registres et registres spécifiques.
- Les compteurs et décompteurs.
- Machines à états finis

### 6.2.2 Conception VHDL

L'objectif de ce cours est de maîtriser les bases de la conception des circuits numériques en langage VHDL. Les éléments de syntaxe sont introduits sur des exemples concrets de circuits combinatoires et séquentiels, permettant dans un premier temps de simuler le fonctionnement de ces composants, puis de les implémenter sur une cible FPGA. Le VHDL est un langage informatique particulier, car il ne permet pas de programmer une machine comme on le fait en C ou en C++. Son rôle est de décrire du "matériel", des composants numériques. On trouve une application directe du VHDL dans le domaine de la microélectronique, pour la conception des circuits intégrés. C'est même un langage incontournable dans ce contexte car tous les blocs numériques sont décrits en HDL. Mais le VHDL ne se limite pas au seul domaine de la microélectronique. On trouve aujourd'hui de nombreuses applications dans le domaine des systèmes embarqués, notamment par l'utilisation de composants logiques programmables tels que les FPGA, qui permettent une flexibilité et des performances plus élevées que les microcontrôleurs.

Ce module, dispensé en 4ème année, aborde les points suivants du langage VHDL :

- Introduction au flot de conception VHDL
    - Logique combinatoire simple
    - Tutoriel ISE/Simulation
    - Composants à base de portes logiques
- Conception hiérarchique
    - Opérateurs arithmétiques
    - additionneur à base de portes logiques
    - additionneur comportemental
    - process
    - additionneur avec Carry Flag / Overflow Flag
    - généricité
- Variables et signaux
    - concept de variable dans les process
    - contexte d'utilisation des variables et des signaux
- Composants séquentiels
    - bascule D Flip Flop
    - bascule avec reset synchrone ou reset asynchrone
    - registre
    - compteurs et compteurs modulo N
- Machines à états finis synchrones
    - architecture des MEF
    - MEF simple pour le contrôle d'un feu tricolore
- Série de TP : Initiation à la synthèse, implémentation sur FPGA, Digicode, ...

### 6.2.3 Systèmes embarqués sur FPGA

Cet enseignement permet de mettre en œuvre et d'approfondir des notions du VHDL, d'architecture, et de programmation. Sur la base d'un système embarqué composé d'un processeur 32 bits décrit en VHDL, on réalise l'ajout de périphériques matériels ainsi que l'écriture des pilotes. Cela permet de comprendre le fonctionnement matériel et logiciel, ainsi que le lien entre ces deux couches. Ce cours trouve évidemment une application directe dans le domaine des Systèmes Embarqués, comme son nom l'indique. Même si la plateforme de validation est basée sur une carte FPGA, il couvre également un ensemble de notions et compétences du domaine de la microélectronique.

- Architecture des systèmes embarqués
- Programmation des systèmes embarqués
- Simulateur de jeu d'instructions (ISS)
- Ajout d'un périphérique: aspects matériels (mapping mémoire) et logiciels (pilote)
- Contrôleur d'écran LCD
- Contrôleur PS/2
- Contrôleur d'écran VGA
- Synthèse sonore
- Réalisation d'un projet

### 6.2.4 Conception des systèmes intégrés numériques

Ce module est dispensé en 5ème année. En 3ème et 4ème année, ce sont essentiellement les aspects techniques de la conception qui ont été étudiés. Ce cours propose de prendre du recul sur le domaine de la microélectronique, afin d'en comprendre les enjeux économiques et technologiques. Ainsi, en liaison avec le cycle de conférences spécialisées, ce cours fait une synthèse de l'état de l'art et des perspectives scientifiques et technologiques dans le domaine de la conception et de l'optimisation des performances électriques des circuits et systèmes intégrés microélectroniques, notamment définition et comparaison des différentes techniques de conception et de réalisation de circuits intégrés spécifiques, présentation de l'état de l'art dans le domaine de la conception des circuits intégrés spécifiques et des logiciels de CAO associés, étude de flots de conception. Architectures des systèmes sur puce (SOC) et conception de composants virtuels (IP), compléments sur les architectures de circuits VLSI, mais aussi :

- Evolution du marché de la microélectronique
- Problématiques de conception / fabrication
- Flot de conception ASIC
- Flot de conception FPGA
- Conception basse consommation
- La recherche en microélectronique

### 6.2.5 Faits marquants

- **Projet avec le navigateur Kito de Pavant : Made In Midi**

Depuis un peu plus d'un an, j'accompagne plusieurs projets d'étudiants en lien avec « Made In Midi », porté par le navigateur Kito de Pavant. L'objectif est de mettre au point l'électronique embarquée qui sera utilisée pour le prochain Vendée Globe en 2016. Le projet est déjà sur une bonne lancée puisque l'équipement sera intégré au voilier pour la prochaine Transat Jacques Vabre en octobre prochain, pour tester les dispositifs en conditions.

- **Projet pédagogique transversal « Robot Mobile »**

Je suis le porteur, avec mes collègues roboticiens du département EII, depuis 2011, d'un projet transversal qui met en œuvre un ensemble de compétences de notre formation (électronique embarquée, programmation, réseau, robotique, contrôle/commande, FPGA, etc.). Le prototype est aujourd'hui utilisé par de nombreux collègues, pour des Travaux Pratiques ou des projets.

- **Projet industriel « Test automatisé de dispositifs électroniques tactiles »**

En collaboration avec la société Intel, nous avons développé un robot parallèle dont le but est de tester des téléphones portables ou tablettes à écrans tactiles. Ce projet débuté il y a 3 ans maintenant intéresse une société de la région qui fait de la réparation de ces objets, et qui est donc fortement intéressée pour automatiser les tests à l'aide du robot. Une instance est dans nos locaux et fait également l'objet de projets pédagogiques.

- **Workshop du Media Cloud Cluster**

En tant qu'administrateur de l'association du Media Cloud Cluster, j'ai organisé dans les locaux de l'école un événement qui a permis de réunir des acteurs du milieu socio-économique local et nos étudiants dans un workshop sur les objets connectés. Ce sont 300 élèves des départements d'Informatique et Gestion et d'Electronique qui ont pu assister à des présentations de sociétés régionales particulièrement actives dans les nouvelles technologies (INTEL, AWOX, SILKAN, etc.).

- **Organisation et animation de tables rondes avec des industriels**

De manière générale, en tant que responsable des relations industrielles du département EII, je suis régulièrement amené à organiser des rencontres entre nos étudiants et les industriels, notamment au cours de tables rondes, de conseils de perfectionnement, d'échanges avec les anciens diplômés du département, etc.

# 7 RESPONSABILITES ADMINISTRATIVES

- **Responsable des relations industrielles du département EII de Polytech Montpellier (depuis 2008)**

Des réunions sont organisées tous les 3 mois pour faire le point sur les actualités relatives aux partenariats industriels (stages, taxe d'apprentissage, réglementation, organisation d'évènements, etc.). Au niveau du département EII, il s'agit de synchroniser les responsables des stages, faire l'interface avec le monde socio-économique, etc.

- **Responsable du département Microélectronique à la commission des utilisateurs des ressources informatiques du LIRMM (2008-2010, 2014)**

Cela implique des réunions tous les 3 mois environ avec les différents responsables, pour faire le point sur les dossiers relatifs à l'utilisation des ressources informatiques. Au niveau du département, c'est un travail de fond sur la synchronisation des besoins et du support outils qu'il faut chercher à optimiser.

- **Membre du pool d'experts 61 et participation à plusieurs comités de sélection (cf. expertises, section 4.3.6)**

La fréquence des réunions est variable en fonction des années, des postes et des invitations à des comités de sélection. Cela nécessite un travail d'expertise des dossiers, et d'audition des candidats.

- **Responsable de la 5ème année de la spécialité MEA du département EII (depuis 2013)**

Ce sont des promotions d'une trentaine d'élèves, pour lesquelles il faut veiller au bon déroulement de l'année, cela commence par la réunion de rentrée, l'organisation des jurys, la gestion des cas sensibles, les relations industrielles, pilotage de l'équipe pédagogique, le suivi des stages, les contrats pros, etc.

- **Directeur adjoint au pôle CNFM de Montpellier, Services Nationaux (depuis 2014)**

Le CNFM prend une part importante de mon temps dédié aux tâches administratives, je vais donc donner un certain nombre de détails sur le contexte, ainsi que l'évolution de mes responsabilités depuis 2006.

Bénéficiant des compétences des chercheurs et des enseignants-chercheurs du LIRMM et de l'Université Montpellier 2, le Pôle Montpelliérain de la Coordination Nationale pour la Formation en Microélectronique et en nanotechnologies (CNFM) héberge les Services Nationaux du GIP CNFM, outil de mutualisation au service de l'enseignement et de la recherche, qui se déclinent en quatre plateformes technologiques et une plateforme pour la formation continue.

La **Plateforme « Logiciels de CAO »** assure l'interface entre les fournisseurs et les sites de formation et de recherche du CNFM. Le pôle CNFM de Montpellier négocie les conditions de mise à disposition des logiciels, passe les marchés, diffuse les logiciels et les clés d'accès, apporte une aide à l'utilisation, organise la formation des formateurs. Quatre suites logicielles sont aujourd'hui disponibles sur la plateforme : Cadence, Synopsys, Coventor et Silvaco. Cette plateforme est utilisée par plus de 50 sites d'enseignement et de recherche. Je suis **responsable de l'outil Synopsys** depuis 2006 (mise en place d'une convention, support aux sites utilisateurs, développement d'un plan de formation). Depuis 2010, je suis également **responsable de la plateforme CAO du CNFM**.

La **Plateforme « Vérification et prototypage »** permet l'évaluation, la sélection et la distribution de cartes de vérification et de prototypage (à base de circuits programmables Altera et Xilinx) et des logiciels associés. A ce jour, ce sont plus de 400 sites répartis sur les niveaux LMD qui ont été équipés. Des cartes plus puissantes sont aussi disponibles pour des applications recherche. En 2012, c'est plus de 1200 cartes qui ont été mises à disposition. J'ai largement développé cette partie là, avec notamment la **mise en place d'un programme 1 étudiant-1 carte FPGA, des concours organisés chaque année, à l'échelle nationale et européenne, le partage de ressources**.

La Plateforme **« Test Industriel »** met à la disposition des utilisateurs un équipement permettant de caractériser les performances de circuits intégrés. Cette plateforme a été mise à niveau en 2007 avec le soutien de la région Languedoc-Roussillon, du GIP CNFM et de la société Verigy. Cette plateforme

est utilisée en enseignement (Niveau LMD) et en recherche (une thèse CIFRE soutenue en 2010). La plateforme dispose de son propre formateur mais l'accès à distance permet d'utiliser cet équipement depuis n'importe quel site d'enseignement en France ou à l'étranger. En 2012-2013, c'est ainsi 253 personnes qui ont été formées au test industriel pour un total supérieur à 3100 heures.

La **Plateforme « SECurité NUMérique »** est une plateforme (cf. section 2.3) qui concerne le transfert vers le CNFM de la plateforme recherche développée au LIRMM ces dernières années. Elle est financée à hauteur de 50% par l'ANR dans le cadre des initiatives d'excellence et de l'IDEFI FINMINA porté par le GIP CNFM en particulier. **Cette activité est en lien direct avec mes recherches sur la sécurité matérielle**.

La **Plateforme « Formation Continue »** a été mise en place au cours de l'année scolaire 2013-2014. Il s'agit de mettre en place un guichet national d'entrée à destination des chercheurs, des enseignants et des industriels du domaine afin de répondre aux demandes diverses en Formation Continue. Elle est financée à hauteur de 50% par l'ANR dans le cadre des initiatives d'excellence et de l'IDEFI FINMINA porté par le GIP CNFM en particulier.

# 8 EN RESUME

Afin de dresser une synthèse de mes différentes activités, le Tableau 7 regroupe :

- L'évolution de ma production scientifique (a)
- Les encadrements de thèse (b)
- Les contrats de recherche (c)
- Les prises de responsabilités administratives (d)

**Tableau 7.** Bilan chronologique des activités et de la production scientifique

| Année | 2001 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 2015 | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Journal | | 1 | | | 1 | | | 1 | 2 | 2 | 1 | | 2 | 1 | 1 | 12 |
| Brevet | | | | | | | | | 1 | | | 1 | 3 | 1 | | 6 |
| Ouvrage | | | | | | | 1 | 1 | | | | | | | | 2 |
| Chapitre | | | | 1 | | | | | | | 4 | 1 | | | | 6 |
| Com. | 1 | 4 | 5 | 1 | 4 | 2 | 5 | 7 | 5 | 17 | 9 | 5 | 2 | 7 | 3 | 77 |
| Total | 1 | 5 | 5 | 2 | 5 | 2 | 6 | 9 | 8 | 19 | 14 | 7 | 7 | 9 | 4 | 103 |

(a) production scientifique

| Année | 2001 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 2015 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Thèse | | | | | | | | | | | | | | | |
| Post-doc | | | | | | | | | | | | | | | |
| NSJ | | | | | | | | | | | | | | | |
| DP | | | | | | | | | | | | | | | |
| CJ | | | | | | | | | | | | | | | |
| IM | | | | | | | | | | | | | | | |
| GMA | | | | | | | | | | | | | | | |
| NH | | | | | | | | | | | | | | | |
| LB | | | | | | | | | | | | | | | |
| FB | | | | | | | | | | | | | | | |
| YA | | | | | | | | | | | | | | | |
| MN | | | | | | | | | | | | | | | |
| MK | | | | | | | | | | | | | | | |
| RD | | | | | | | | | | | | | | | |
| MEA | | | | | | | | | | | | | | | |
| PR | | | | | | | | | | | | | | | |

(b) encadrement doctoral

| Année | 2001 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 2015 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CEA | | | | | | | | | | | | | | | |
| ADAM | | | | | | | | | | | | | | | |
| ST | | | | | | | | | | | | | | | |
| MoKa | | | | | | | | | | | | | | | |
| Secresoc | | | | | | | | | | | | | | | |
| Modern | | | | | | | | | | | | | | | |
| DGA | | | | | | | | | | | | | | | |
| NETHEOS | | | | | | | | | | | | | | | |
| SECNUM | | | | | | | | | | | | | | | |
| FINMINA | | | | | | | | | | | | | | | |
| CLERECO | | | | | | | | | | | | | | | |
| Bull | | | | | | | | | | | | | | | |
| MultiS | | | | | | | | | | | | | | | |
| Seclab | | | | | | | | | | | | | | | |

(c) contrats de recherche

| Année | 2001 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | 11 | 12 | 13 | 14 | 2015 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Synopsys | | | | | | | | | | | | | | | |
| CAO | | | | | | | | | | | | | | | |
| SN | | | | | | | | | | | | | | | |
| Relindus | | | | | | | | | | | | | | | |
| MEA5 | | | | | | | | | | | | | | | |
| CURI | | | | | | | | | | | | | | | |

(d) responsabilités

# 9 LISTE DES PUBLICATIONS

## 9.1 Thèse

[T1] Pascal Benoit. « Architectures des Accélérateurs de Traitement Flexibles pour les Systèmes sur Puce ». Micro and nanotechnologies/Microelectronics. Université Montpellier II - Sciences et Techniques du Languedoc, 2004

## 9.2 Revues

[J1] F. Bruguier, P. Benoit, L. Torres, L. Barthe, M. Bourree, V. Lomne, "Cost-effective Design Strategies for Securing Embedded Processors", *IEEE Transactions on Emerging Topics in Computing*, doi:10.1109/TETC.2015.2407832, February 2015 – RICL [8]A+

[J2] Fernanda Lima Kastensmidt, Jorge Tonfat, Thiago Hanna Both, Paolo Rech, Gilson Wirth, Florent Bruguier, Pascal Benoit. Voltage scaling and aging effects on soft error rate in SRAM-based FPGAs. *Microelectronics Reliability*, Elsevier, 2014, 54 (9-10), pp.2344-2348 – RICL A+

[J3] Mansouri I., Benoit P., Torres L., Clermidy F., *"Fine-grain dynamic energy tracking for system-on-chip"*, IEEE Transactions on Circuits and Systems II: Express Briefs, Volume:60, Issue: 6, pp. 356 – 360 – RICL A+

[J4] Ost L., Mandelli M., Almeida G. M., Moller L., Indrusiak L. S., Sassatelli G., Benoit P., Glesner M., Robert M., Moraes F., *"Power-aware dynamic mapping heuristics for NoC-based MPSoCs using a unified model-based approach"*, journal ACM Transactions on Embedded Computing Systems (TECS), Volume 12 Issue 3, March 2013, Article No. 75 – RICL A+

[J5] G. Almeida, R. Busseuil, L. Ost, F. Bruguier, G. Sassatelli, P. Benoit, L. Torres, M. Robert, *"PI and PID Regulation Approaches for Performance-Constrained Adaptive Multiprocessor System-on-Chip"*, Embedded Systems Letters, IEEE, Volume: PP, Issue:99, ISSN: 1943-0663, DOI: 10.1109/LES.2011.2166373, September 2011, pp. 1-4 – RICL A+

[J6] P. Benoit, L. Torres, G. Sassatelli, M. Robert, N. Saint-Jean, *"Run Time Mapping for Dynamic Reconfiguration Management in Embedded Systems"*, International Journal of Embedded Systems (IJES), Inderscience, Volume 4 – Issue 3/4 – 2010, pp. 276-291– RICL

[J7] I. Mansouri, P. Benoit, D. Puschini, L. Torres, F. Clermidy, G. Sassatelli, *"Dynamic Energy Optimization in NoC-based System-on-Chips"*, Journal of Low Power Electronics JOLPE – Vol. 6, N° 4, December 2010, pp. 564-577 – RICL A

[J8] G. Almeida, Saint-Jean Nicolas, Varyani Sameer, G. Sassatelli, P. Benoit, L. Torres, *"An Adaptive Message Passing MPSoC Framework"*, Hindawi International Journal of Reconfigurable Computing, Volume 2009, 242981 (2009) 18, pp. 1-18 – RICL

[J9] Zipf P., Sassatelli G., Utlu N., Saint-Jean N., Benoit P., Glesner M., *"A Decentralized Task Mapping Approach for Homogeneous Multi-Processor Network-On-Chips"*, International Journal of Reconfigurable Computing, Volume 2009, 453970 (2009), pp. 1-10 – RICL

[J10] D. Puschini, P. Benoit, F. Clermidy, G. Sassatelli, *"A Game-Theoretic Approach for Run-Time Distributed Optimization on MP-SoC"*, International Journal of Reconfigurable Computing, Volume 2008, 403086 (2008), pp. 1-10 – RICL

[J11] Pascal Benoit, Gilles Sassatelli, Lionel Torres, Michel Robert, G. Cambon, et al.. Méthode de caractérisation des architectures d'accélérateurs flexibles pour systèmes sur puce. *TSI : Revue Technique et Science Informatiques*, 2005, 24 (6), pp.725-755

[8] RICL : Revue Internationale avec Comité de Lecture (A+ : excellent niveau, A : bon niveau)

[J12] Gilles Sassatelli, Pascal Benoit, Lionel Torres, G. Cambon, Jérôme Galy, et al.. Systolic Ring: Une Nouvelle Approche pour les Architectures Reconfigurables Dynamiquement. *Traitement du Signal (T.S.)*, GRETSI-CNRS, Saint Martin d'Hères (France), 2002, 19 (4), pp.293-313

## 9.3 Brevets

[B1] Y. Akgul, E. Beigné, S. Lesecq, P. Puschini and P. Benoit, "Procédé de commande d'un circuit électronique" (FR, 2013), N° E.N. 13 58541 (Date de dépôt : 06/09/2013)

[B2] Mansouri I., Clermidy F., Benoit P., "System and method for designing digital circuitry with an activity sensor", US 8782592 B2, US, 15/07/2014

[B3] Imen Mansouri, Fabien Clermidy, Pascal Benoit. Système et procédé de conception de circuit numérique à capteur d'activité. France, Patent n° : FR2982684 (A1). 2013, pp.N/A

[B4] Imen Mansouri, Fabien Clermidy, Pascal Benoit. Système et procédé de conception de circuit numérique à capteur d'activité, circuit numérique correspondant. France, Patent n° : FR2982685 (A1). 2013, pp.N/A

[B5] Diego Puschini, Pascal Benoit, Fabien Clermidy. Method for Optimising the Operation of a Multi-Processor Integrated Circuit, and Corresponding Integrated Circuit. France, Patent n° : EP2417506 A1. 2012, pp.30

[B6] Diego Puschini, Fabien Clermidy, Pascal Benoit. Procédé d'optimisation du fonctionnement d'un circuit intégré multiprocesseurs, et circuit intégré correspondant. France, Patent n° : PCT/FR2009/050581. 2009, pp.32

## 9.4 Ouvrages

[O1] Lionel Torres, Ian O'Connor, Pascal Benoit, Amar Mukherjee, Asim Smailagic. ISVLSI'08: Annual Symposium on VLSI. IEEE Computer Society, pp.N/A, 2008, 978-0-7695-3170-0

[O2] Gilles Sassatelli, Manfred Glesner, Christophe Bobda, Pascal Benoit. ReCoSoC'07: Reconfigurable Communication-Centric SoCs. CD-ROM, 2007

## 9.5 Chapitres d'ouvrage

[CO1] Pascal Benoit, Gilles Sassatelli, Philippe Maurine, Lionel Torres, Nadine Azemard, et al.. Towards autonomous scalable integrated systems. G. Nicolescu; Ian O Connor; C. Piguet. *Design Technology for Heterogeneous Embedded Systems*, Springer, pp.63-89, 2012.

[CO2] J. L. Danger, S. Guilley, L. Barthe, P. Benoit. Countermeasures Against Physical Attacks in FPGAs. *Security Trends for FPGAS From Secured to Secure Reconfigurable Systems*, Springer, pp.137-188, 2011.

[CO3] Eduardo Wanderley, Romain Vaslin, Jeremie Crenne, Pascal Cotret, Jean-Philippe Diguet, et al.. SecurityFPGA Analysis. *Security Trends for FPGAS - From Secured to Secure Reconfigurable Systems*, pp.7-46, 2011.

[CO4] Benoit Badrignans, Florian Devic, Lionel Torres, Gilles Sassatelli, Pascal Benoit. Embedded Systems Security for FPGA. *Security trends for FPGAs*, pp.137-187, 2011.

[CO5] Lionel Torres, Pascal Benoit, Gilles Sassatelli, Michel Robert, Fabien Clermidy, et al.. An Introduction to Multi-Core System on Chip - Trends and Challenges. Hübner, Michael and Becker, Jürgen. *Multiprocessor System-on-Chip - Hardware Design and Tool Integration*, Springer, pp.1-21, 2011, Chapter 1, 978-1-4419-6459-5

[CO6] Gilles Sassatelli, Lionel Torres, Pascal Benoit, Gaston Cambon, Michel Robert, et al.. Dynamically Reconfigurable Architectures for Digital Signal Processing Applications. *SoC Design Methodologies*, Kluwer, pp. 63-74, 2004, ISBN : 1-40-20-7148-5

## 9.6 Communications dans un congrès

**2015 (3)**

[C1] Yeter Akgul, Diego Puschini, Lionel Vincent, Pascal Benoit, Mauricio Altieri Scarpato, "Energy-efficient control through power mode placement with discrete DVFS and Body Bias", 13th IEEE International NEWCAS Conference 2015 – CICL B

[C2] Maha Kooli, Alberto Bosio, Pascal Benoit, "Software Testing and Software Fault Injection", 10th International Conference on Design & Technology of Integrated Systems in Nanoscale Era April 20-24, 2015, Napoli, Italy

[C3] Mario Barbareschi, Giorgio Di Natale, Florent Bruguier, Pascal Benoit, Lionel Torres,"Ring Oscillators Analysis for FPGA Security Purposes", 3rd TRUDEVICE Workshop, March 13, 2015, Grenoble (France)

**2014 (7)**

[C4] Abdulazim Amouri, Florent Bruguier, Saman Kiamehr, Pascal Benoit, Lionel Torres, et al.. Aging effects in FPGAs: an experimental analysis. *FPL'2014: 24th International Conference on Field Programmable Logic and Applications*, Sep 2014, Munich Germany. 2014 – CICL A+

[C5] Mohamad Najem, Pascal Benoit, Florent Bruguier, Gilles Sassatelli, Lionel Torres. Method for dynamic power monitoring on FPGAs. *24th International Conference on Field Programmable Logic and Applications (FPL)*, Sep 2014, Munich, Germany. 2014 – CICL A+

[C6] Yeter Akgul, Diego Puschini, Suzanne Lesecq, Edith Beigné, Ivan Miro Panades, Pascal Benoit, Lionel Torres, *"Power management through DVFS and dynamic body biasing in FD-SOI circuits",* 2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC), DAC 2014, 1-6 – CICL A+

[C7] Kooli Maha, Giorgio Di Natale, Pascal Benoit, Alberto Bosio, Lionel Torres, et al.. Fault injection tools based on Virtual Machines. *Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC), 2014 9th International Symposium on*, May 2014, Montpellier, France. 2014

[C8] F. L. Kastenschmidt, J. Tonfat, T. Both, P. Rech, G. Wirth, R. Reis, F. Bruguier, P. Benoit, L. Torres, C. Frost, *"Analyzing the Impact of Aging and Voltage Scaling under Neutron-induced Soft Error Rate in SRAM-based FPGAs"*, 19th IEEE European Test Symposium, May 26-30, 2014 – CICL A

[C9] F. Bruguier, P. Benoit, L. Torres, P. Nouet, « Formation en Sécurité Numérique : Théorie et Mise en pratique sous forme de Stage Technologique », Journées Pédagogiques du CNFM, St Malo, France, 19-21 novembre 2014, pp 1-6

[C10] F. L. Kastensmidt, J. Tonfat, T. Both, P. Rech, G. Wirth, R. Reis, F. Bruguier, P. Benoit, L. Torres, C. Frost, « Analysing Aging and Voltage Scaling Impacts under Neutron-induced Soft Error Rate in SRAM-based Field Programmable Gate Arrays », ESREF 2014 : 25th European Symposium on Reliability of Electron devices, Failure physics and analysis, Berlin, Allemagne, 29 septembre au 2 octobre 2014, pp 1-6 – CICL B

**2013 (2)**

[C11] Y. Akgul, D. Puschini, S. Lesecq, E. Beigne, P. Benoit, L. Torres, *"Methodology for Power Mode Selection in FD-SOI circuits with DVFS and Dynamic Body Biasing"*, The International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS) 2013, pp. 199-206, – CICL B

[C12] J. F. Tarrillo, J. Tonfat, R. Reis, F. Kastensmidt, F. Bruguier, M. Bourrée, P. Benoit and L. Torres, *"Using Electromagnetic Emanations for Variability Characterization in Flash-Based FPGAs"*, Symposium on VLSI, 2013. ISVLSI '13. IEEE Computer Society Annual, 2013, pp. 109-114, – CICL A

**2012 (5)**

[C13] Benoit P., "Distributed Approaches for Self-Adaptive Embedded Systems", WORLDCOMP 2012, ISBN #: 1-60132-233-X EDITOR: Toomas P. Plaks, *The 2012 International Conference on Engineering of Reconfigurable Systems and Algorithms,* ERSA'12, **Invited Talk**

[C14] Devic F., Torres L., Crenne J., Badrignans B., Benoit P., *"Secure DPR: Secure Update Preventing Replay Attacks for Dynamic Partial Reconfiguration"*, FPL'12: Field Programmable Logic and Applications (2012), Norway, pp. 57-62, – CICL A+

[C15] Bourrée M., Benoit P., Torres L., Maurine P., "*SECNUM: an Open Characterizing Platform for Integrated Circuits*", European Workshop on Microelectronics Education (EWME), EWME 2012, pp. 88-91

[C16] Akgul Y., Puschini D., Lesecq S., Miro-Panades I., Benoit P., Torres L., Beigne E., *"Power Mode Selection in Embedded Systems with Performance"*, FTFC 2012, Low Voltage Low Power Conference, 2012, France, pp. 1-4

[C17] Perin G., Torres L., Benoit P., Maurine P., *"Amplitude Demodulation-based EM Analysis of different RSA implementations"*, Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012, pp. 1167 - 1172, **Best Paper Award Candidate,** – CICL A+

**2011 (9)**

[C18] Busseuil R., Barthe L., Almeida G. M., Ost L., Bruguier F., Sassatelli G., Benoit P., Robert M., Torres L., *"Open-Scale: A Scalable, Open-Source NOC-based MPSoC for Design Space Exploration"*, IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig), 2011, pp. 357-362, – CICL B

[C19] Barthe L., Cargnini L. V., Benoit P., Torres L., *"Optimizing an Open-Source Processor for FPGAs: A Case Study"*, IEEE FPL'11: Field Programmable Logic and Applications (2011), Greece, pp. 551-556, – CICL A+

[C20] Bruguier F., Benoit P., Maurine P., Torres L., *"A New Process Characterization Method for FPGAs Based on Electromagnetic Analysis"*, IEEE FPL'11: Field Programmable Logic and Applications (2011), Greece, pp. 20-23 – CICL A+

[C21] Bruguier F., Benoit P., Maurine P., Torres L.,"A novel Process Characterisation Method for FPGAs based on Electromagnetic Analysis", VARI 2011, Grenoble, France**,** – CICL B

[C22] N. Hébert, G. M. Almeida, P. Benoit, G. Sassatelli, L. Torres*, "Evaluation of a Distributed Fault Handler Method for MPSoC"*, IEEE International Symposium on Circuits and Systems (ISCAS), May 15-18, 2011, Rio de Janeiro, Brazil, pp. 2329-2332, – CICL A

[C23] G. M. Almeida, R. Busseuil, E. A. Carara, N. Hébert, P. Benoit, G. Sassatelli, F.G. Moraes, L. Torres*, "Predictive Dynamic Frequency Scaling for Multi-Processor Systems-on-Chip"*, IEEE International Symposium on Circuits and Systems (ISCAS), May 15-18, 2011, Rio de Janeiro, Brazil, pp. 1500-1503, – CICL A

[C24] L. Barthe, L. V. Cargnini, P. Benoit and L. Torres, *"The SecretBlaze: A Configurable and Cost-Effective Open-Source Soft-Core Processor"*, 25th IEEE International Parallel & Distributed Processing Symposium, May 16-20, 2011, Anchorage (Alaska) USA, pp. 310-313, – CICL B

[C25] C. Roth, G. M. Almeida, O. Sander, L. Ost, N. Hebert, G. Sassatelli, P. Benoit, L. Torres and J. Becker, *"Modular Framework for Multi-Level Multi-Device MPSoC Simulation"*, 25th IEEE

International Parallel & Distributed Processing Symposium, May 16-20, 2011, Anchorage (Alaska) USA, pp. 146-142 – CICL B

[C26] Becker J., Benoit P., Cumplido R., *"RAW Introduction"*, IPDPS Workshops, 25th IEEE International Parallel & Distributed Processing Symposium, May 16-20, 2011, Anchorage (Alaska) USA, pp. 125-127

**2010 (17)**

[C27] N. Hébert, P. Benoit, G. Sassatelli, L. Torres, "D-Scale: A Scalable System-Level Dependable Method for MPSoCs," 19th IEEE Asian Test Symposium, 2010, pp.198-205 – CICL A

[C28] N. Hébert, P. Benoit, G.M. Almeida, G. Sassatelli, L. Torres, *"A Cost-Effective Solution to Increase System Reliability and Maintain Global Performance under Unreliable Silicon in MPSoC"*, IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig), 2010, pp. 346 – 351 – CICL B

[C29] G.M. Almeida, S. Varyani, R. Busseuil, N. Hebert, G. Sassatelli, P. Benoit, L. Torres, M. Robert, *"Providing Better Multi-processor Systems-on-Chip Resources Utilization by Means of Using a Control-Loop Feedback Mechanism"*, IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig), 2010, pp. 382 – 387 – CICL B

[C30] Mansouri I., Clermidy F., Benoit P., Torres L., *"A Run-time Distributed Cooperative Approach to Optimize Power Consumption in MPSoCs"*, SOCC'10: 23th IEEE International SOC Conference, United States, pp. 25-30 – CICL B

[C31] Barthe L., Benoit P., Torres L., *"Investigation of a Masking Countermeasure against Side-Channel Attacks for RISC-based Processor Architectures"*, IEEE FPL'10: Field Programmable Logic and Applications (2010), Italy, pp. 139-144 **Best Paper Award Candidate** – CICL A+

[C32] G.M. Almeida, S. Varyani, R. Busseuil, G. Sassatelli, P. Benoit, L. Torres, E.A. Carara, F.G. Moraes, *"Evaluating the Impact of Task Migration in Multi-Processor Systems-on-Chip"*, ACM 23rd Symposium on Integrated Circuits and Systems Design (SBCCI'2010). September, 2010. Sao Paulo, Brazil, pp. 73-78 – CICL B

[C33] I. Mansouri, C. Jalier, F. Clermidy, P. Benoit, L. Torres, *"Implementation Analysis of a Dynamic Energy Management Approach Inspired by Game-Theory"*, VLSI (ISVLSI), 2010 IEEE Computer Society Annual Symposium on, 2010 , pp. 422 – 427 – CICL A

[C34] C. Jalier, D. Lattard, G. Sassatelli, P. Benoit, L. Torres, *"A Homogeneous MPSoC with Dynamic Task Mapping for Software Defined Radio"*, VLSI (ISVLSI), 2010 IEEE Computer Society Annual Symposium on, 2010, pp. 345 - 350 – CICL A

[C35] C. Jalier, D. Lattard, A.A. Jerraya, G. Sassatelli, P. Benoit, L. Torres, *"Heterogeneous vs homogeneous MPSoC approaches for a Mobile LTE modem"*, Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010, pp. 184 - 189 – CICL A+

[C36] E. Faure, G.M. Almeida, M. Benabdenbi, P. Benoit, F. Clermidy, F. Pêcheux, G. Sassatelli, L. Torres, *"An in-memory monitoring database for self adaptive MP2SoCs"* , Design and Architectures for Signal and Image Processing (DASIP), 2010 Conference on, 2010 , pp. 97 – 104 – CICL B

[C37] L. Barthe, P. Benoit, L. Torres, *"Side-Channel Attacks on Embedded Systems"*, Colloque National du GDR SOC-SIP, Cergy, France, 9-11 juin 2010

[C38] S. Varyani, G. Marchesan Almeida, R. Busseuil, G. Sassatelli, P. Benoit, L. Torres, *"Exploration of Virtualization Techniques for Achieving Adaptability in Heterogeneous MPSoCs"*, Colloque National du GDR SOC-SIP, Cergy, France, 9-11 juin 2010

[C39] F. Bruguier, P. Benoit, L. Torres, *"A Variability Compensation Flow for FPGAs"*, Colloque National du GDR SOC-SIP, Cergy, France, 9-11 juin 2010

[C40] Bruguier F., Benoit P., Torres L., *"Investigation of Digital Sensors for Variability Characterization on FPGAs"*, ReCoSoC'10: 5th International Workshop on Reconfigurable Communication-Centric Systems on Chip, France, pp. 95-100

[C41] Busseuil R., Marchesan Almeida G., Varyani S., Sassatelli G., Benoit P., *"A Self-Adaptive Communication Protocol Allowing Fine Tuning Between Flexibility and Performance in Homogeneous MPSoC Systems"*, Reconfigurable Communication-centric Systems on Chip, Allemagne, pp. 1-6

[C42] C. Jalier, D. Lattard, G. Sassatelli, P. Benoit, L. Torres, *"Flexible and distributed real-time control on a 4G telecom MPSoC"*, Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on, 2010, pp. 3961 – 3964 – CICL A

[C43] Poucheret F., Barthe L., Benoit P., Torres L., Maurine P., Robert M., *"Spatial EM Jamming: a Countermeasure Against EM Analysis ?"*, VLSI-SoC'10: 18th IEEE/IFIP International Conference on VLSI and System-on-Chip, Spain, pp. 105-110 – CICL B

**2009 (5)**

[C44] D. Puschini, F. Clermidy, P. Benoit, G. Sassatelli, and L. Torres, *"Adaptive energy-aware latency-constrained DVFS policy for MPSoC"*, 2009 IEEE International SOC Conference (SOCC), IEEE, 2009, pp. 89-92 – CICL B

[C45] D. Puschini, F. Clermidy, P. Benoit, G. Sassatelli, and L. Torres, *"Dynamic and distributed frequency assignment for energy and latency constrained MP-SoC"*, Design, Automation & Test in Europe Conference & Exhibition, 2009. DATE '09, 2009, pp. 1564-1567 – CICL A+

[C46] Almeida G., Varyani S., Sassatelli G., Busseuil R., Benoit P., Torres L., *"Self-adaptability in Multi-processor Embedded Systems"*, Colloque GDR SoC SiP 2009, France

[C47] Nicolas Hebert, Lionel Torres, Gilles Sassatelli, Pascal Benoit, *"Une approche pour la fiabilité des plateformes Multiprocesseurs"*, 22ème édition du colloque GRETSI, Dijon, 2009

[C48] Nicolas Hebert, Lionel Torres, Gilles Sassatelli, Pascal Benoit, *"Etude de la fiabilité sur la plateforme MP-SoC HS-Scale"*, JNRDM - 12ème édition Journées Nationales du Réseau Doctoral de Microélectronique, Lyon, 2009

**2008 (7)**

[C49] D. Puschini, F. Clermidy, P. Benoit, G. Sassatelli, and L. Torres, *"Game-Theoretic Approach for Temperature-Aware Frequency Assignment with Task Synchronization on MP-SoC"*, 2008 International Conference on Reconfigurable Computing and FPGAs, IEEE, 2008, pp. 235-240 – CICL B

[C50] N. Saint-Jean, P. Benoit, G. Sassatelli, L. Torres, and M. Robert, *"MPI-Based Adaptive Task Migration Support on the HS-Scale System"*, Symposium on VLSI, 2008. ISVLSI '08. IEEE Computer Society Annual, 2008, pp. 105-110 – CICL A

[C51] D. Puschini, F. Clermidy, P. Benoit, G. Sassatelli, and L. Torres, *"Temperature-Aware Distributed Run-Time Optimization on MP-SoC Using Game Theory"*, Symposium on VLSI, 2008. ISVLSI '08. IEEE Computer Society Annual, 2008, pp. 375-380, (**nominated for the best paper award**) – CICL A

[C52] D. Puschini, F. Clermidy, P. Benoit, G. Sassatelli, and L. Torres, *"Convergence analysis of run-time distributed optimization on adaptive systems using game theory"*, 2008 International Conference on Field Programmable Logic and Applications, IEEE, 2008, pp. 555-558 – CICL A+

[C53] N. Saint-Jean, G. Sassatelli, P. Benoit, L. Torres, and M. Robert, *"Bio-inspiration helps computers: A new machine"*, 2008 International Conference on Field Programmable Logic and Applications, IEEE, 2008, pp. 697-698 – CICL B

[C54] Zipf P., Sassatelli G., Utlu N., Saint-Jean N., Benoit P., Glesner M., *"A Novel Task Allocation Approach for Homogeneous MPSOC"*, ReCoSoC'08: Reconfigurable Communication−Centric SoCs 2008, ISBN 978-84-691-3603-4, 9-11 juillet 2008, Barcelone, Espagne, pp. 40−47

[C55] G. Almeida, Saint-Jean Nicolas, Varyani Sameer, G. Sassatelli, P. Benoit, L. Torres, *"Exploration of Task Migration Policies on the HS-Scale System"*, ReCoSoC'08: Reconfigurable Communication−Centric SoCs 2008, ISBN 978-84-691-3603-4, 9-11 juillet 2008, Barcelone, Espagne, pp. 48−54

**2007 (5)**

[C56] N. Saint-Jean, P. Benoit, G. Sassatelli, L. Torres, and M. Robert, *"Application Case Studies on HS-Scale, a MP-SOC for Embbeded Systems"*, 2007 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation, IEEE, 2007, pp. 88-95 – CICL B

[C57] N. Saint-Jean, G. Sassatelli, P. Benoit, L. Torres, and M. Robert, *"HS-Scale: a Hardware-Software Scalable MP-SOC Architecture for embedded Systems"*, IEEE Computer Society Annual Symposium on VLSI (ISVLSI '07), IEEE, 2007, pp. 21-28 – CICL A

[C58] G. Sassatelli, N. Saint-Jean, P. Benoit, L. Torres, M. Robert, C. Woszezenki, I. Grehs, and F. Moraes, *"Run-time mapping and communication strategies for Homogeneous NoC-Based MPSoCs"*, 15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM 2007), IEEE, 2007, pp. 295-296 – CICL A

[C59] N. Saint-Jean, Jalier C., G. Sassatelli, P. Benoit, L. Torres, and M. Robert, *"HS Scale: A Run-Time Adaptable MP-SoC Architecture"*, ReCoSoC'07: Reconfigurable Communication-Centric SoCs 2007, Montpellier, France, pp. 39-46

[C60] Gilles Sassatelli, Nicolas Saint-Jean, Pascal Benoit, Lionel Torres, Michel Robert. An Adaptative MP-SoC Architecture for Embedded Systems. *CAR'07: 2nd National Workshop on Control Architectures of Robots: From Models to Execution on Distributed Control Architectures*, Jun 2007, Paris, France. pp.101-107, 2007

**2006 (2)**

[C61] Lionel Torres, Pascal Benoit, Laurent Latorre, Gaston Cambon. Synthèse, Simulation, Placement et Routage d'n Circuit. O. Bonnaud, H. Lhermite. *CNFM'06: Neuvièmes Journées Pédagogiques*, Nov 2006, Saint-Malo (France), Centre Commun de Microélectronique de l'Ouest, pp.23-31, 2006

[C62] Pascal Benoit, Lionel Torres, Gilles Sassatelli, Michel Robert, G. Cambon, et al.. Dynamic Hardware Multiplexing: Improving Adaptability with a Run Time Reconfiguration Manager. *ISVLSI'06: IEEE Computer Society Annual Symposium on VLSI*, Mar 2006, Karlsruhe (Germany), IEEE, pp.251-256, 2006 – CICL A

**2005 (4)**

[C63] Pascal Benoit, Gilles Sassatelli, Lionel Torres, G. Cambon, Michel Robert, et al., Run-Time Scheduling for Random Multi Tasking in Reconfigurable Coprocessors. Tero Rissa, Steve Wilton. *FPL'05 IEEE : 15th International Workshop on Field-Programmable Logic and Applications*, Aug 2005, Tampere (Finlande), pp.703-706, 2005 – CICL A+

[C64] Pascal Benoit, Lionel Torres, Gilles Sassatelli, Michel Robert, G. Cambon. Automatic Task Scheduling/Loop Unrolling Using Dedicated RTR Controllers in Coarse Grain Recongigurable Architectures. *IPDPS'05: International Parallel and Distributed Processing Symposium, RAW'05: Reconfigurable Architectures Workshop*, Apr 2005, Denver, Colorado (USA), IEEE, pp.148, 2005 – CICL B

[C65] Pascal Benoit, Gilles Sassatelli, Lionel Torres, Michel Robert, G. Cambon. Gestion Matérielle du Parallélisme pour les Architectures Reconfigurables à Grain Epais. *JFAAA'05: Journées Francophones sur l'Adéquation Algorithme Architecture*, Jan 2005, Dijon, France. pp.15-20, 2005

[C66] Pascal Benoit, Gilles Sassatelli, Lionel Torres, Michel Robert, G. Cambon. *« Dynamic Hardware Multiplexing for Coarse Grain Reconfigurable Architectures »*, FPGA'05: ACM/SIGDA 9th International Symposium on Field Programmable Gate Arrays, ISBN 1-59593-029-9, 20-22 février 2005, Monterey, Californie, pp. 270 – CICL A+

**2004 (1)**

[C67] Pascal Benoit, Gilles Sassatelli, Lionel Torres, D. Demigny, Michel Robert, et al., Metrics for Digital Signal Processing Architectures Characterization: Remanance and Scability. S. Vassiliadis. *Computer Systems: Architectures Modelling and Simulation*, Jul 2004, Samos (Greece), Kluwer, pp.128-137, 2004 – CICL B

**2003 (5)**

[C68] Pascal Benoit, Gilles Sassatelli, Lionel Torres, Michel Robert, G. Cambon, et al.. Comparaison d'Architectures Dédiées au Traitement Numériques des Données : du Processeur aux Architectures Reconfigurables. *RenPar'15 : Rencontres Francophones en Parallélisme, CFSE'3 : Conférence Française sur les Systèmes d'Exploitation, SympAAA'03 : Symposium en Architecture et Adéquation Algorithme Architecture*, Oct 2003, La Colle sur Loup (France), France. pp.322-326, 2003

[C69] Pascal Benoit, Gilles Sassatelli, Lionel Torres, Michel Robert, G. Cambon. Comparaison des Architectures Dédiées au Traitement des Données Numériques. *JNRDM'03 : 6ièmes Journées Nationales du Réseau Doctoral de Microélectronique*, May 2003, Toulouse (France), France. pp.115-117, 2003

[C70] Pascal Benoit, Gilles Sassatelli, Lionel Torres, D. Demigny, Michel Robert, et al.. Metrics for Reconfigurable Architectures Characterization: Remanence and Scalability. *SAMOS'03: Systems, Architecture, Modeling and Simulation*, Samos (Grèce), pp. 128-137, 2003 – CICL B

[C71] Pascal Benoit, Gilles Sassatelli, Lionel Torres, D. Demigny, Michel Robert. « Metrics for Reconfigurable Architectures Characterization: Remanence and Scalability ». *IEEE International Parallel and Distributed Processing Symposium*, Nice (France), pp. 176-180, 2003 – CICL B

[C72] Pascal Benoit, Gilles Sassatelli, Lionel Torres, Michel Robert, G. Cambon, et al.. A Novel Approach for Architectural Models Characterization. An Exemple Through the Systolic Ring. Springer. FPL'03: 12th International Workshop on Field-Programmable Logic and Applications, Lisbon (Portugal), pp. 722-732, 2003, Lecture Notes in Computer Science – CICL A+

**2002 (4)**

[C73] Pascal Benoit, Gilles Sassatelli, Michel Robert, Lionel Torres, G. Cambon, et al.. Architectures Reconfigurables Dynamiquement pour Applications TSI. *Colloque du Groupe de Recherche Conception Assistée par Ordinateur (GDR CAO) de Circuits et Systèmes Intégrés*, May 2002, Paris, France. pp.67-70, 2002

[C74] Pascal Benoit, Gilles Sassatelli, Michel Robert, Lionel Torres, G. Cambon, et al.. The Systolic Ring: A Scalable Dynamically Reconfigurable Core for Embedded Systems. *SAME'02: Sophia-Antipolis Forum on MicroElectronics*, Oct 2002, Sophia-Antipolis (France), France. pp.85-90, 2002

[C75] Pascal Benoit, Gilles Sassatelli, Lionel Torres, T. Gil, G. Cambon, et al.. Caractérisation et Comparaison d'Architectures Reconfigurables Dynamiquement, Un Exemple : Le Systolic Ring. *JFAAA'02: Journées Francophones sur l'Adéquation Algorithme Architecture*, Dec 2002, Monastir (Tunisie), France. pp.30-34, 2002

[C76] G. Sassatelli, L. Torres, P. Benoit, T. Gil, C. Diou, G. Cambon, J. Galy, "Highly Scalable Dynamically Reconfigurable Systolic Ring-Architecture for DSP Applications", *DATE*, 2002, Design, Automation & Test in Europe Conference & Exhibition, Design, Automation & Test in Europe Conference & Exhibition 2002, pp. 0553 – CICL A+

**2001 (1)**

[C77] Gilles Sassatelli, Lionel Torres, Pascal Benoit, G. Cambon, Michel Robert, et al.. Dynamically Reconfigurable Architectures for Digital Signal Processing Applications. *SoC Design Methodologies, Selection des meilleures contributions de l'IFIP 12th International Conference on Very Large Scale Intergration (VLSI'01)*, Dec 2001, Montpellier (France) – CICL B

# PARTIE 2 :
# SYNTHESE DES TRAVAUX ET PROJET DE RECHERCHE

Cette partie a pour objectif la synthèse de mes travaux de recherche, que j'ai regroupés en 3 thèmes principaux :

- conception d'architectures reconfigurables et adaptatives
- optimisation dynamique des systèmes adaptatifs
- systèmes adaptifs de confiance

Ce chapitre retrace presque quinze années de réflexions, de formalisations et de développements autour des systèmes numériques adaptatifs. Pour chaque thème, et chaque contribution, je me suis efforcé de rappeler le contexte, la problématique, ainsi que les principaux résultats obtenus. Tous les sujets sont bien évidemment liés, et plutôt que de faire une synthèse chronologique, j'ai préféré établir une synthèse thématique en explicitant au mieux les motivations et les résultats de chaque thèse, en essayant par ailleurs d'en formuler une analyse critique. Aussi, pour compléter ce manuscrit, certaines contributions pourront être consultées en annexe dans des articles publiés dans des revues scientifiques. Pour terminer cette partie, une présentation succincte des travaux en cours est proposée, qui fait le lien avec mes perspectives à court terme, ainsi qu'un projet de recherche à plus long terme.

# 1 CONCEPTION D'ARCHITECTURES RECONFIGURABLES ET ADAPTATIVES

Nous retraçons dans cette section les travaux de conception d'architectures reconfigurables, jusqu'à des architectures MPSOC adaptatives, ce qui couvre une période allant du début de ma thèse en octobre 2001, jusqu'à 2012.

## 1.1 Conception d'architectures reconfigurables dynamiquement

Ces travaux qui se sont déroulés au LIRMM faisaient suite à une série de projets dans le domaine de la conception des architectures dédiées aux applications du Traitement du Signal et des Images (TSI). Avec l'apparition à la fin des années 1990 des systèmes intégrés sur silicium, ou systèmes sur puce (), de nouvelles réflexions avaient été amorcées quant à l'opportunité de définir de nouvelles architectures, en lieu et place de processeurs dédiés ou programmables, au sein de ces systèmes, notamment pour la prise en charge des calculs nécessitant de hautes performances. Les impératifs de vitesse de traitement, de consommation et de flexibilité relatifs aux nouvelles applications envisagées (multimédia, réseaux de téléphonie mobile, sécurité etc.) imposaient la définition de nouveaux concepts et paradigmes architecturaux. Au classique modèle d'exécution séquentielle de *Von Neumann* animant les processeurs devait se substituer une référence d'exécution autorisant un parallélisme massif et hétérogène, au sens de la granularité.

### 1.1.1 Contexte et positionnement des travaux

Cette thèse fut tout d'abord l'occasion de réaliser un état de l'art relativement large des architectures de traitement flexibles dans le contexte SOC. Nous avions considéré à cette occasion deux familles d'architectures : les processeurs et les architectures reconfigurables. Pour chacune, nous avions insisté sur les attributs architecturaux qui permettaient d'améliorer le compromis de performances notamment les moyens matériels mis en œuvre pour exploiter le parallélisme, élément déterminant pour l'accélération des applications. Cette étude avait permis de dresser un bilan qualitatif suivant un ensemble de critères, notamment sur la prise en charge de différentes granularités de parallélisme en vue d'optimiser les performances (Tableau 8).

**Tableau 8.** Prise en charge des niveaux de parallélismes par différents types d'architectures programmables

| | | *Flot (tâche)* | *Boucle* | *Spatial* | *Temporel* |
|---|---|---|---|---|---|
| Processeurs Programmables | Superscalaire SMT | Oui (multi-thread) | Oui | Oui (<8) | Oui (faible) |
| | DSP VLIW | Non | Oui (automatique et/ou manuel) | Oui (<8) | Oui (faible) |
| Processeurs Reconfigurables | Grain Fin mixte | Oui | Oui (manuel) | Oui (variable) | Oui (variable) |
| | Grain épais | Théorique | Oui (manuel) | Oui (variable) | Oui (variable) |

C'est dans ce contexte que fut développé le *Systolic Ring*. Après avoir contribué à sa conception **[1]**, nous avions dû aborder une problématique de caractérisation, qui s'inscrivait d'ailleurs dans une réflexion plus large au niveau du RTP SOC (équipe projet POMARD) : comment positionner ce type d'architecture par rapport aux approches plus traditionnelles telles que les processeurs programmables ou les DSP ? Par rapport au domaine d'application, comment déterminer l'architecture d'accélérateur la plus adéquate (problématique AAA) ? Les réponses à ces questions étaient loin d'être triviales, si l'on tenait compte de la nature et de la pluralité des solutions : cœurs dédiés, cœurs de DSP, cœurs reconfigurables à grain fin, à grain épais ou à granularité mixte. Le choix n'était en fait jamais basé sur une approche formalisée, mais sur l'expérience et le pragmatisme du concepteur.

### 1.1.2 Métriques d'évaluation des performances

Il était donc nécessaire de définir une approche de caractérisation systématique, basée sur un ensemble d'indicateurs objectifs pour choisir plus simplement les architectures existantes susceptibles de

répondre aux contraintes imposées par les spécifications de l'application. Pour un architecte cherchant à évaluer et comparer plusieurs solutions au stade des spécifications, il fallait trouver un moyen de comparaison qui permettait de faire abstraction de l'implantation physique du circuit, mais qui aurait une signification réelle des performances, ce qui justifierait l'intérêt de cette approche. L'objectif de ces travaux était donc de proposer des modèles, des méthodes et des outils de mesures simples et formalisés pour comparer différentes architectures, et intervenir très tôt dans le flot de conception sur les solutions pertinentes à retenir. En nous basant sur des modèles génériques de l'architecture et des applicatifs, nous avons défini un ensemble de cinq métriques : la densité opérative qui caractérise le compromis puissance de traitement/surface, la rémanence qui caractérise le dynamisme de l'architecture, le taux d'interconnexion qui représente la flexibilité d'associativité des opérateurs, le taux d'accélération qui illustre le gain en vitesse par rapport à une implantation purement séquentielle et plusieurs fonctions de coûts permettant l'évaluation de l'efficacité énergétique des architectures de processeurs flexibles. Nous avons ensuite procédé à une expérimentation sur plusieurs architectures d'accélérateurs (DSP et architectures reconfigurables à grain fin ou à grain épais), travaux dont le détail des résultats est fourni dans **[2]**.

Nous donnons dans la suite un rapide aperçu de quelques métriques et résultats illustrant leur utilisation.

- **Densité opérative**

Pour caractériser le compromis fait entre le potentiel opératif ($N_{OP}$ étant le nombre d'opérateurs) et la surface relative ($A(M\lambda^2)$ calculée à partir de la surface absolue et la longueur caractéristique de grille dans une technologie donnée), nous utilisons la « densité opérative », $D_{OP}$, qui se calcule de la manière suivante :

$$D_{OP} = \frac{N_{OP}}{A_{COEUR}} \text{ [Nombre d'opérateurs.}\lambda^{-2}\text{]}$$

avec

$$A(M\lambda^2) = \frac{A(\mu m^2)}{(L/2)^2}$$

Dans le but d'évaluer le passage à l'échelle d'une solution architecturale, il est par exemple très intéressant de tracer l'évolution de $D_{OP}$ en fonction de $N_{OP}$. La courbe obtenue offre ainsi un aperçu de la *scalabilité*. Ainsi, un cœur générique intégré à une plate-forme de conception peut-être personnalisé suivant les besoins opératifs et les contraintes de surface. Ce tracé peut être également utilisé comme une référence permettant d'apporter des modifications à un modèle d'architecture lors de sa conception. Lorsque la valeur de $D_{OP}$ devient trop faible, le concepteur peut alors décider d'introduire un niveau de hiérarchie supplémentaire.

- **Rémanence**

Le dynamisme d'une architecture flexible peut être caractérisé par la mesure liée au temps nécessaire à la reconfiguration du composant. A cet effet, nous avons proposé la rémanence qui se calcule de la manière suivante :

$$R = \frac{N_{OP}.F_e}{N_c.F_c}$$

où R est égal au nombre de cycles nécessaires pour reconfigurer la totalité des opérateurs. L'inverse de R caractérise le dynamisme de l'architecture (plus R tend vers 1, plus l'architecture est dynamique). Puisqu'il s'agit d'un rapport, la rémanence est à la fois indépendant de la granularité de l'architecture et de la technologie silicium.

- **Facteur d'accélération**

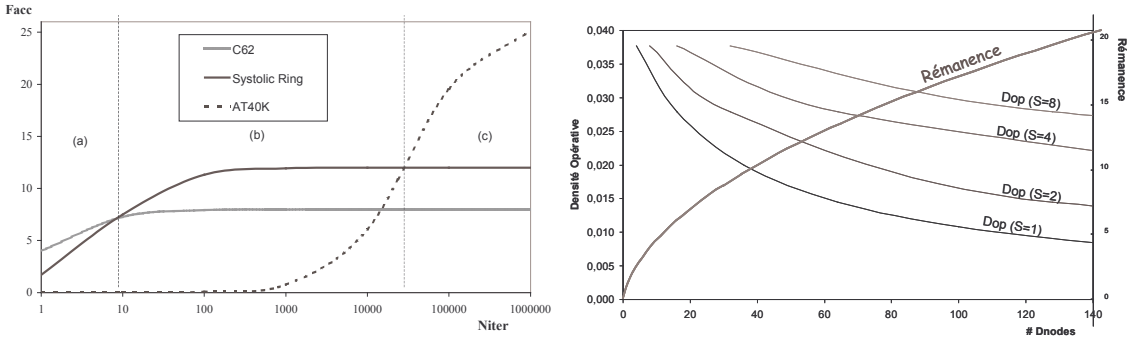
Dans la perspective d'ajout d'un accélérateur, il faut pourvoir estimer le gain potentiel sur le temps de traitement par rapport à une approche CPU. Il est possible d'évaluer ce gain au niveau système par la loi de *Amdahl* **[3]** par exemple. Au niveau du coprocesseur, on propose de mesurer le gain par rapport à une implémentation purement séquentielle. D'après le modèle de calcul, le nombre de cycles minimum requis pour un traitement purement séquentiel est $N*N_{ITER}$ cycles (N étant le nombre d'opérations d'une cœur de boucle, $N_{ITER}$ le nombre d'itérations de cette boucle), en considérant que le nombre de cycles par opération est de 1. Cette valeur est utilisée comme référence pour calculer $F_{ACC}$, le facteur d'accélération potentiel d'une architecture, calculé de la manière suivante :

$$F_{ACC}=\frac{N.N_{ITER}}{(C_{COMP}+C_{CONF})}$$

où $C_{COMP}$ est le nombre de cycles requis pour le traitement des $N.N_{ITER}$ opérations et $C_{CONF}$ le nombre de cycles nécessaire pour configurer l'architecture à ce traitement. Pour une architecture donnée, la borne supérieure du facteur d'accélération est son nombre d'opérateurs $N_{OP}$ (qui suppose donc un parallélisme maximal). Autrement dit, pour une boucle donnée, l'accélération sera d'autant meilleure que le nombre d'opérateurs utilisés sera grand. D'un point de vue système, cette métrique peut être intégrée dans la loi de *Amdahl* en tant qu'accélération de la partie améliorée.

- **Résultats**

Les **Figures 1 (a)** et **(b)** illustrent quelques uns des résultats obtenus. Tout d'abord le facteur d'accélération formalise clairement la zone d'intérêt de l'utilisation d'un type architecture (ici étaient comparés un DSP, le C62, et deux architectures reconfigurables, une à grain épais, le *Systolic Ring* et une autre à grain fin, le FPGA AT40K). On constate que chacune de ces approches peut présenter un intérêt qui dépend en fait du nombre de quantités de données à traiter sur un motif de calcul, permettant d'amortir le coût de la configuration et de déployer le parallélisme. La **Figure 1** montre des évaluations de la densité opérative et de la rémanence pour une architecture reconfigurable à grain épais, suivant plusieurs paramètres de l'architecture (le nombre d'éléments de calculs *Dnode*, et le nombre de clusters hiérarchiques utilisés S pour un nombre d'éléments de calculs). On peut lire la figure ainsi par exemple : pour un nombre de *Dnode* souhaité, et donc un certain niveau de parallélisme, on compare les choix possibles à faire entre le dynamisme, et la flexibilité de l'architecture. Cela illustre parfaitement la *scalabilité* d'une architecture, et donc la nécessité de faire des compromis entre la flexibilité, la performance et le dynamisme, par exemple en introduisant des niveaux de hiérarchie telle que cela a été formalisé et mis en œuvre dans cet exemple du *Systolic Ring*.




(a) (b)

**Figure 1.** Facteur d'accélération (a), Densité opérative et rémanence (b)

La méthode fut donc exploitée dans l'environnement de conception de l'architecture générique du Systolic Ring. Cette dernière dispose d'un outil conçu durant la thèse, permettant de générer le code VHDL de l'architecture suivant un ensemble de paramètres. Pour définir ces paramètres, nous avons développé un outil intégrant les différentes métriques proposées. Ainsi, en spécifiant des valeurs correspondant aux besoins des applications et du système (caractéristiques des applicatifs à accélérer,

surface silicium disponible, accélérations souhaitées), on peut extraire automatiquement un sous-ensemble de solutions et ainsi évaluer rapidement les différents compromis.

### 1.1.3 Multiplexage matériel

Les architectures reconfigurables, notamment à grain épais, et les microprocesseurs possèdent des caractéristiques très proches, avec une différence fondamentale qui réside dans le procédé de reconfiguration, systématique à chaque cycle pour ces derniers, alors que pour une architecture reconfigurable celle-ci ne l'est pas nécessairement. Ces dernières ont également la capacité de pouvoir exploiter un parallélisme en deux dimensions : lorsque celui-ci est possible, on peut alors figer la fonctionnalité afin de limiter le contrôle de la structure, donc sa complexité, la surface et la consommation qui lui sont associés.

Mais le potentiel d'accélération théorique de chaque architecture n'étant pas toujours simple à exploiter dans la pratique, ceci nous poussa à explorer d'autres voies, notamment par l'étude de la prise en charge de différentes sources de parallélisme. Le potentiel dynamique des architectures reconfigurables à grain épais nous a amené à considérer deux stratégies pour l'allocation des ressources : l'approche VLIW, où l'allocation et l'ordonnancement sont réalisés dès la phase de compilation, ou la solution superscalaire / multithread, qui permet une optimisation de l'utilisation des ressources de manière dynamique.

Dans un contexte d'intégration SOC à base d'OS multitâches, nous avons donc cherché à définir une méthode pour tirer parti du parallélisme à un niveau plus élevé. En permettant un placement dynamique des tâches, l'accélérateur matériel devient une plate-forme plus performante et plus flexible. C'est en partant de ce constat que nous avons entrepris une extension des propriétés de reconfiguration dynamique pour l'exploitation automatisée de différents niveaux de parallélisme. Cette méthode dite de **multiplexage matériel** permet d'exploiter le parallélisme dynamique, et ce de manière transparente du point de vue du système, du concepteur, et du programmeur. Autrement dit, c'est le matériel qui prend en charge cette tâche, avec bien entendu un effort à porter sur le coût d'intégration d'une telle solution.

Le contrôleur dédié que nous avons conçu, implémente un algorithme d'allocation et d'ordonnancement, adaptant la configuration initiale suivant les contraintes matérielles d'implantation au moment de l'appel de procédure. Il est également capable de dérouler automatiquement des boucles de calcul, poussant au maximum l'exploitation du potentiel de ressources. Ainsi, c'est sur 3 niveaux de granularité de parallélisme que l'accélérateur peut améliorer les performances globales : opérateur, boucle et tâche. Ce contrôleur (*Saturne*, en référence à son positionnement dans la gestion des anneaux de calcul du *Systolic Ring*) a été conçu puis synthétisé. Nous avons alors substitué le processeur RISC initialement présent dans l'architecture *Systolic Ring* par *Saturne*, et validé notre concept sur plusieurs applications TSI (Transformée en ondelettes, DCT, estimation de mouvement et détection de contours, **[4]**). Ce cas d'étude a permis de mettre en évidence une amélioration significative de l'exploitation des ressources de l'accélérateur, avec un taux de remplissage de 30% plus élevé, obtenu grâce à l'exécution simultanée de plusieurs tâches. Cette meilleure utilisation des ressources permet d'augmenter le débit de traitement quasiment d'un facteur 2. L'algorithme d'allocation et d'ordonnancement des tâches basé sur des transformations géométriques exploitant efficacement les caractéristiques topologiques de l'accélérateur, implique un surcoût inférieur à 10% (sans optimisation particulière), et très largement inférieur à la solution initialement disponible. A titre d'exemple, la **Figure 2** illustre un scénario de prise en charge de différentes tâches par l'accélérateur (une tâche peut-être déroulée, ou enroulée suivant les besoins et les ressources disponibles).

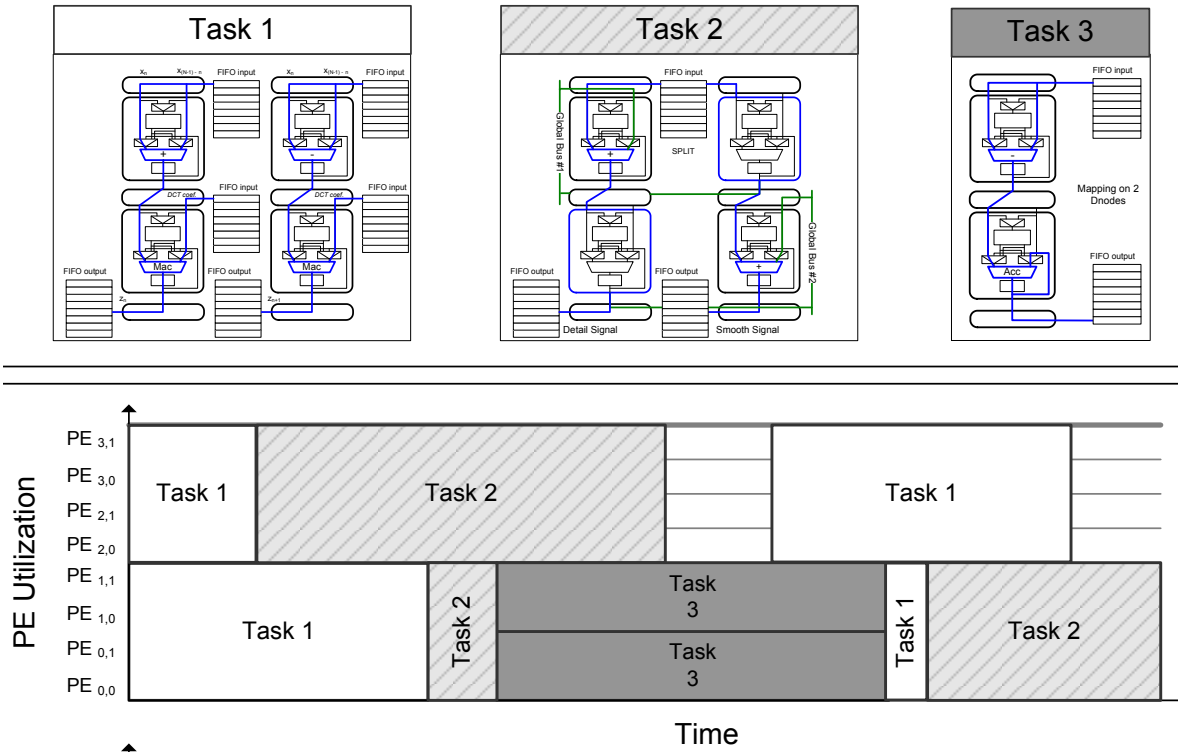



**Figure 2.** Scénario applicatif de l'accélérateur avec multiplexage matériel dynamique

### 1.1.4 Intégration système et évaluation

Mes travaux de thèse se sont essentiellement focalisés sur les aspects architectures des accélérateurs de traitement pour les systèmes sur puce. Afin d'étendre les concepts étudiés au niveau système, j'ai eu l'opportunité de poursuivre une partie de ces travaux dans le cadre d'un Post-doctorat en Allemagne en collaboration avec l'Université de Karlsruhe. La « *programmabilité* » de la solution avec en entrée un langage « haut niveau » (langage C typiquement) était un des objectifs de ce travail, en intégrant la méthodologie dans un contexte d'utilisation d'un micronoyau pour une plus grande réactivité du système face à différents contextes applicatifs (adaptation).

Le flot de conception général est illustré dans la figure 3. Durant la première phase du flot de conception, la phase de compilation, l'application est partitionnée en plusieurs sous-tâches (les tâches relatives au contrôle ne ciblent que le CPU « SW » et celles qui présentent potentiellement un certain degré de parallélisme « HW » sont compilées pour la cible configurable et la cible CPU, générant ainsi 2 codes objets distincts). La deuxième étape du flot se situe lors de la phase de fonctionnement. En s'appuyant sur une évaluation en ligne des contraintes (état des ressources, performances, charge de la batterie), un système d'exploitation temps réel sélectionne la configuration exécutable la mieux appropriée à l'état du système. Le DHM interagit directement avec le micronoyau et devient ainsi un service matériel de gestion des ressources.

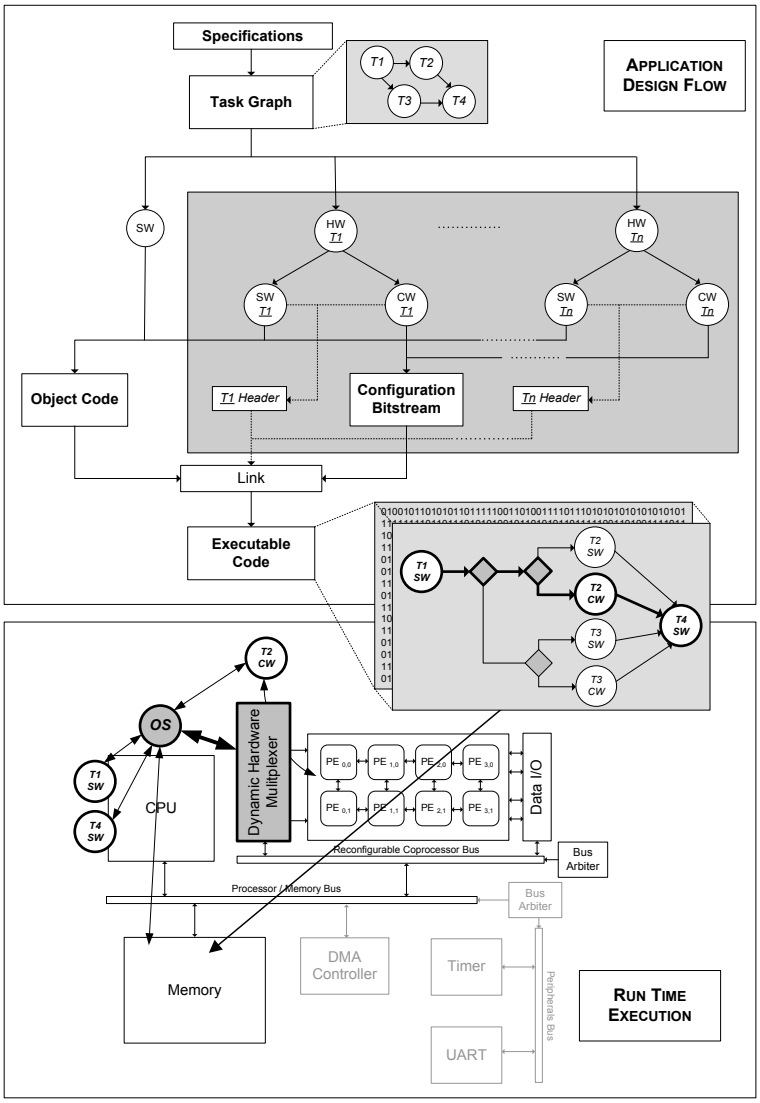



**Figure 3.** Intégration système du multiplexage matériel

Un nouveau travail de définition de métriques adaptées à la caractérisation de la prise en charge d'un parallélisme de plus haut niveau fut également engagé. Quatre métriques, $MT_{eff}$ qui représente l'efficacité du multitâche, le *WL* pour la charge de travail, *R* pour le ratio d'utilisation de l'accélérateur dans le temps, et enfin $P_{eff}$ pour l'efficacité de calcul furent définies. Le développement d'un environnement d'exploration et de simulation haut-niveau a été également mis en œuvre : dans le même esprit que le travail entrepris au cours de la thèse, l'objectif était ici de pouvoir fournir un outil d'évaluation afin de comparer plusieurs possibilités dans la définition des paramètres d'une architecture d'accélérateur, qui soient le plus en adéquation avec les contraintes des applications à prendre en charge, et ce pour réduire l'espace de conception. La **Figure 4** illustre les différentes étapes de cette exploration, et la **Figure 5** montre le résultat de l'évaluation suivant les métriques proposées pour différents paramétrages de l'accélérateur. L'ensemble des résultats est disponible dans **[5]**.

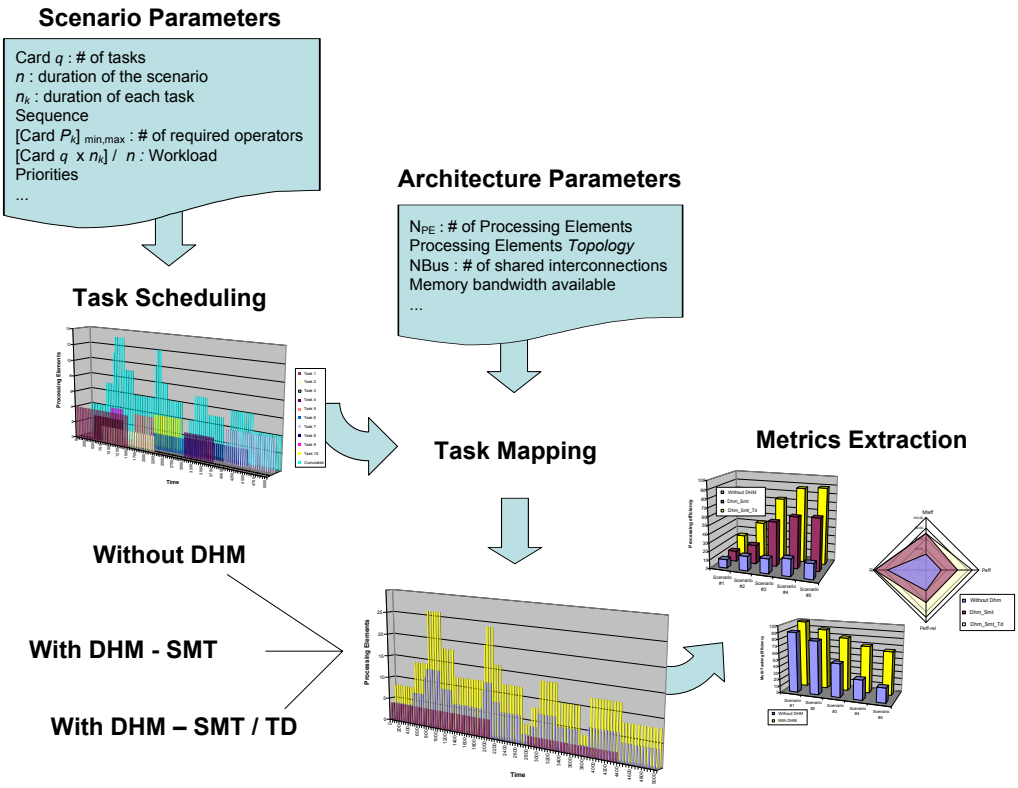



**Figure 4.** Outil d'exploration et de simulation haut niveau

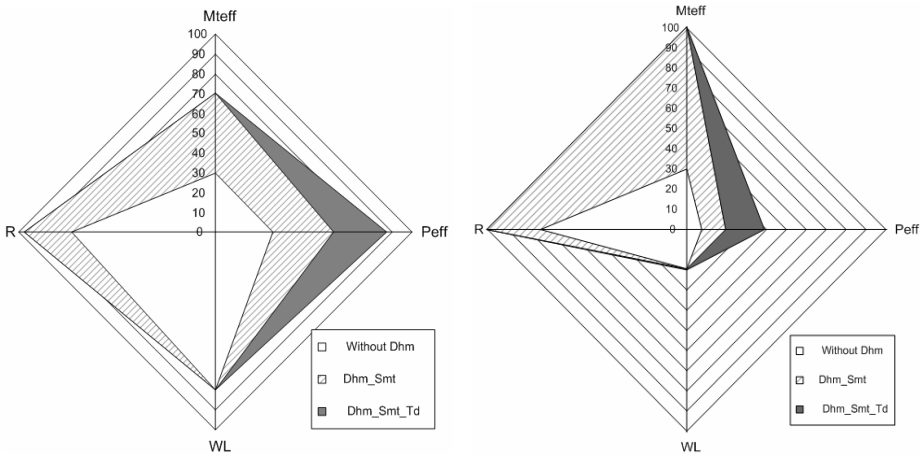



**Figure 5.** Exemple de résultats obtenus à l'issue de l'exploration pour 2 jeux de paramètres de l'accélérateur, et exploitant les caractéristiques ou pas de multiplexage matériel. Une surface plus grande indique une meilleure adéquation entre l'architecture et les applications.

## 1.2 Conception d'architectures MPSOC adaptatives

C'est par l'augmentation exponentielle des densités d'intégration suivant la loi de Moore depuis 50 ans, que les processus de conception, de vérification, de fabrication et de test tendent à se complexifier. Pour améliorer les performances de calcul des systèmes intégrés, le doublement des fréquences à chaque génération technologique fut pendant plus de 30 ans une réalité, qui commençait à s'essouffler dès le début des années 2000 : en cause notamment, la consommation d'énergie qui au traditionnel compromis surface / délai venait s'ajouter aux critères d'optimisation à considérer. Les circuits reconfigurables se positionnaient comme une solution intermédiaire entre le tout spécifique (ASIC) et le tout programmable (microprocesseur). A cela s'ajoutait un autre compromis : de la granularité fine pour plus de flexibilité (FPGA) ou de la granularité épaisse pour plus de performance (CGRA). Il est intéressant de noter que c'est à ce moment là que l'intégration de plusieurs processeurs sur un même substrat de silicium est devenue possible, et allait permettre d'exploiter le parallélisme intrinsèque de ces nouvelles architectures pour repousser les limites de la performance. Les architectures reconfigurables gros grains allaient dans le même temps se complexifier pour devenir plus facilement programmables, en reposant sur une brique de base type processeur, ce qui a d'ailleurs amené certains scientifiques de notre communauté à considérer que le processeur était le nouveau transistor[9]. On assistait donc à une convergence vers ce qu'on a appelé des MPSOC.

Derrière ce terme générique MPSOC, se trouve tout un ensemble de déclinaisons de systèmes intégrés dont le point commun est la présence d'au moins 2 processeurs. Pour le reste, il existe de nombreuses spécificités architecturales permettant de distinguer les différentes approches. En tout premier lieu, c'est la nature des éléments de calcul (PE : Processing Element) de la structure que l'on va comparer. Pour un système composé de différents types de PE, on parle de MPSOC hétérogène, ou de système multi-cœur (on trouve également le terme CMP, pour Chip Multi Processing ou encore les « *many-core systems* »). Généralement, ces circuits disposent de processeurs généralistes et d'accélérateurs matériels (programmables type DSP ou dédiés). Ces architectures multi-cœurs hétérogènes sont devenues aujourd'hui la norme, raison pour laquelle lorsqu'on parle de , il s'agit bien souvent de systèmes comportant plusieurs processeurs généralistes et des accélérateurs (par exemple le A7 de Apple est un double cœur ARMv8 64 bits avec un accélérateur graphique intégré).

Bien que ces architectures hétérogènes présentent de nombreuses qualités, notamment par rapport au compromis performance/ énergie, elles souffrent toutefois d'une *scalabilité* limitée, et d'une certaine rigidité. La nature même des composants utilisés impose une irrégularité de la structure, qui pose des problèmes à tous les étages : de la conception à la vérification, mais aussi au niveau de la programmation et du *debug*. Plutôt que de prendre des PE adaptés à certaines tâches (pour caricaturer, le contrôle pour les processeurs généralistes, le calcul pour le processeurs dédiés), une approche alternative consiste à ne prendre qu'un seul type de PE, et c'est en quelque sorte aux tâches de s'adapter en fonction des ressources disponibles. C'est ce qu'on appelle l'approche homogène, qui présente un avantage de poids : le même élément est répliqué autant de fois que nécessaire, pour former une grille régulière de PE. Cela signifie que n'importe quelle tâche peut s'exécuter sur n'importe quel PE du système, ce qui offre une grande flexibilité potentielle dans le *mapping* des applications. La régularité de l'architecture déplace la problématique de conception/vérification au niveau du PE, du moins en grande partie, puisqu'il ne s'agit ensuite que de rajouter plus ou moins d'éléments. Donc le changement d'échelle pour chaque génération technologique se trouve lui aussi largement simplifié. On s'attend ainsi à ce que la performance émane de l'exécution parallèle d'un ensemble de tâches, ce qui s'approche de la thématique des architectures parallèles. Cependant, à l'échelle d'un système intégré, cette problématique présente bon nombre de spécificités et contraintes qu'il faut considérer afin d'y apporter des solutions adéquates.

L'approche homogène basée sur un ensemble de PE exécutant des tâches en parallèle peut permettre une accélération « efficace » des traitements, en ajustant dynamiquement les paramètres de tension / fréquence de chaque processeur pour faire en sorte que le calcul respecte des contraintes de

[9] "The SOC **Processor** is **the new Transistor**", D. **Patterson**, UC Berkeley

performance avec une consommation énergétique minimale. Cette solution est naturellement plus flexible et permet un passage à l'échelle facilité par rapport à une structure hétérogène. Mais au delà des principes, il y a évidemment de nombreux points à considérer pour atteindre cet objectif, tels que l'organisation de la mémoire, l'infrastructure de communication entre processeurs, le modèle de programmation, etc. Une analyse étendue de la littérature aussi bien en ce qui concerne les architectures parallèles, que les architectures MPSOC hétérogènes et homogènes, nous a permis de dresser un ensemble de caractéristiques architecturales qui nous permettraient de distribuer les ressources de calcul, de mémorisation, de communication, mais aussi tous les paramètres permettant d'adapter en ligne le système. Le premier point concerne l'interconnexion des PE, et notre choix s'est porté sur les réseaux intégrés de type NOC, qui permettent une meilleure *scalabilité* de la bande passante avec l'accroissement du nombre de processeurs. L'utilisation d'une mémoire partagée avec des mémoires caches distribuées nécessite la mise en œuvre de mécanismes de cohérence de cache complexes, raison pour laquelle nous nous sommes orientés vers une architecture mémoire distribuée qui présente une meilleure *scalabilité*, mais implique un modèle de programmation adapté. Enfin, toujours pour des raisons de passage à l'échelle et de capacités d'adaptation, notre choix s'est porté sur une gestion asymétrique du système par l'exécution locale d'un micronoyau par processeur.

En résumé, les caractéristiques ciblées pour la conception d'un système MPSOC homogène sont:

1- l'utilisation d'un PE compact et simple

2- l'exécution d'un micronoyau par PE

3- des services d'adaptation pris en charge au niveau du micronoyau

4- des mémoires distribuées pour la prise en charge du code du micronoyau et des applications

5- une architecture de communication type NOC GALS

6- un modèle de programmation par passage de messages

7- des ressources pour l'adaptation locale du PE (ilot V/F ou VFI) avec des actionneurs de type DVFS et des capteurs pour le monitoring

### 1.2.1 HS-Scale

L'architecture simplifiée de ce MPSOC est représentée sur la **Figure 6**. Notre première contribution dans ce domaine se nomme *HS-Scale* ; c'est ce que nous allons détailler dans les prochains paragraphes.

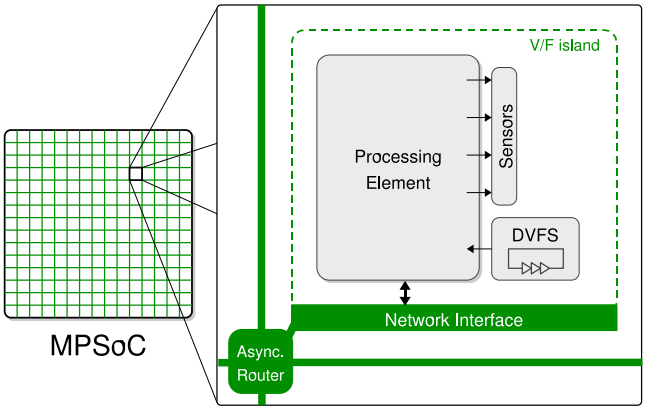



**Figure 6.** Architecture générique du MPSOC homogène

- **H-Scale**

*H-Scale* est la partie matérielle de ce MPSOC. C'est une matrice régulière d'éléments identiques appelés NPU (Network Processing Unit) représenté sur la **Figure 7**. Ce bloc de base comporte un processeur MIPS 32 bits (PE sur la figure), une mémoire (dans laquelle est stockée le micronoyau et les tâches à exécuter), un routeur qui peut transmettre des données à d'autres NPU du réseau sans interrompre le processeur local, ainsi que quelques périphériques (UART et *Timer*). Une interface réseau locale basée sur des FIFO matérielles asynchrones permet d'envoyer (recevoir) des données vers le (du) réseau, et donc à destination (provenance) de NPU distants. Le contrôleur UART permet un accès à l'environnement externe de *H-Scale*, et autorise par exemple le chargement de code en mémoire à partir d'un *bootloader*. Un contrôleur d'interruptions est également disponible, ainsi qu'un *timer* et un arbitre de bus (décodeur) qui permet d'interconnecter l'ensemble de ces composants.

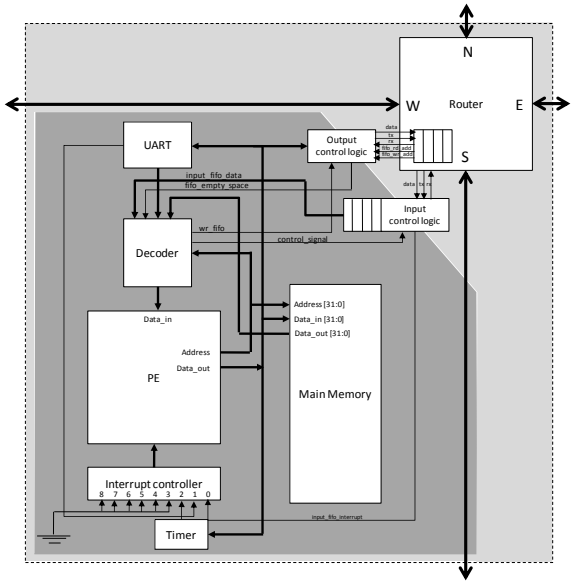



**Figure 7.** H-Scale, le NPU

Le système d'interconnexion entre les NPU est basé sur un réseau maillé 2D à commutation de paquets (HERMES **[6]**). Les communications entre processeurs sont basées sur un protocole « *toggle* » qui utilise 2 signaux pour la synchronisation, ce qui permet d'avoir un système localement synchrone

et globalement asynchrone (GALS) : il est ainsi possible de faire fonctionner les NPU à des fréquences différentes et d'implémenter des VFI pour mettre en œuvre des mécanismes locaux de DVFS pour une gestion adaptative de la consommation.

- **Gestion de HS-Scale : le micronoyau**

Le système fonctionne suivant un principe de gestion asymétrique qui nécessite donc l'exécution d'un OS par processeur. Il faut nécessairement que l'empreinte mémoire de celui-ci soit limitée au maximum, tout en offrant des services de base tels que l'ordonnancement de tâches, la gestion de périphériques, mais aussi des services avancés pour le chargement dynamique en mémoire (sachant que le NPU ne dispose pas de MMU) et l'adaptation en ligne.

Après avoir envisagé de recourir à des OS existants, nous n'avons pu que constater qu'aucun d'entre eux ne répondait à nos spécifications, raison pour laquelle nous avons développé notre propre solution. Le micronoyau *HS-Scale* (**Figure 8**) permet un ordonnancement des tâches préemptif, gestion des interruptions, des exceptions et des tâches, management de la mémoire sans MMU. La mise en œuvre de "Position Independent Code" en utilisant une Global Offset Table intégrée directement dans le fichier exécutable, grâce à un outil de post-compilation qui génère un nouvel entête, permet à l'OS de faire un *build* à chaque fois qu'il reçoit une nouvelle tâche à exécuter.

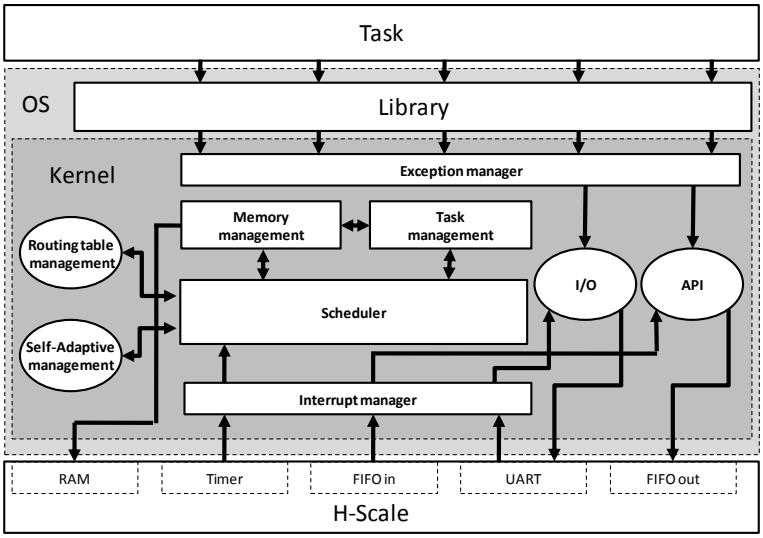



**Figure 8.** Vue d'ensemble du micronoyau

- **S-Scale**

L'utilisation d'une architecture à mémoire distribuée implique naturellement l'utilisation pour les communications entre tâches d'une API de type « passage de messages ». Les modèles de programmation efficaces pour cela reposent généralement sur des graphes de tâches dans lesquels les nœuds du graphe sont des tâches, et les arcs représentent les canaux de communications. Parmi les modèles existants, nous avons adopté un modèle de calcul KPN (Kahn Process Networks), avec une base de programmation utilisant un langage procédural séquentiel (langage C), un support pour le multitâche et des primitives de communication entre les tâches. C'est donc un parallélisme de niveau tâche qui est exploité ici. Le modèle KPN exige un partitionnement de l'application en plusieurs tâches qui sont reliées par le biais des primitives de communication, *Read_Socket()* et *Write_Socket()*. Celles-ci sont mises en œuvre au moyen de FIFO logicielles directement gérées par le micronoyau (**Figure 9**). Ces fonctions permettent aux tâches de communiquer entre-elles, quel que soit le *mapping* ; l'API permet de faire abstraction du placement, ce sont des tables de routage au niveau de l'OS qui font le lien entre les tâches locales et distantes.

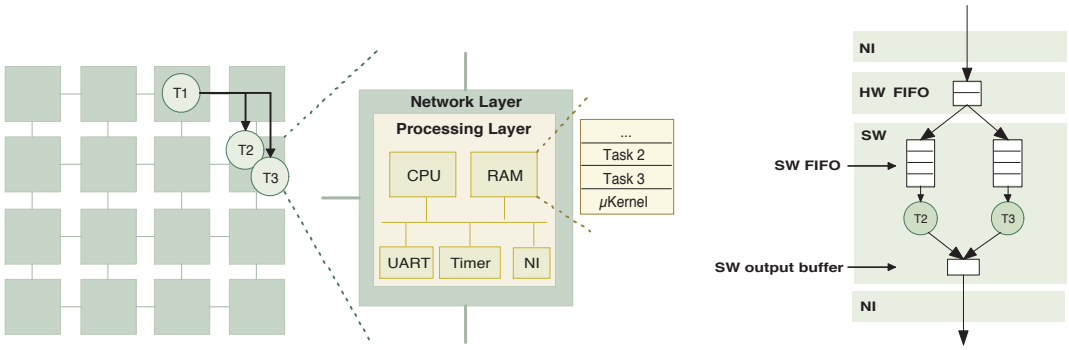



**Figure 9.** Exécution d'une application multitâche sur HS-Scale

- **Mécanismes d'adaptation**

La motivation sous-jacente au développement d'une architecture MPSOC homogène est la mise en œuvre de mécanismes d'adaptation permettant au système d'ajuster sa configuration en fonction de son environnement et des besoins applicatifs. C'est par le biais de capteurs et plus généralement d'un dispositif de monitoring que l'OS récupère des informations sur l'état du système. C'est à partir de cela qu'il va établir un diagnostique et décider des actions à effectuer, en particulier ici des migrations de tâches. Dans cette première version, ce sont des services logiciels de surveillance de la charge CPU, de la charge des FIFO et du placement des tâches (à partir des tables de routage) qui sont mis en œuvre. Ces services font partie intégrante du micronoyau et sont donc par conséquent déployés sur l'ensemble des unités. Le premier et deuxième algorithme mesurent la charge processeur ou la charge des FIFO et déclenchent une migration de tâche lorsque un seuil est dépassé. Le troisième algorithme cherche à rapprocher des tâches en fonction de leur *mapping*.

L'actionneur mis en œuvre dans ce travail est la migration de tâches, dont le but initial est d'équilibrer la charge de calcul sur l'ensemble des ressources. Cette migration implique un déplacement du code et des données de la tâche. Notre système ne disposant pas de MMU, il faut définir une procédure pour reconstruire le contexte de la pile sur un autre NPU. Il faut pour cela que la migration intervienne en un point spécifique du code (*checkpoint*) : celle-ci n'est autorisée qu'au niveau des fonctions de communication (*Read & Write_Socket()*).

- **Résultats**

Nous avons réalisé plusieurs implémentations de *HS-Scale*. La première utilise un design kit de ST Microelectronics 90nm. Le NPU a été synthétisé, placé et routé : il peut fonctionner à une fréquence maximale de 300 MHz. L'utilisation d'une mémoire locale de 64 Ko représente près de 90% de la surface totale du NPU. Le processeur occupe à lui seul une surface de 1,2mm2, le routeur 0,85mm2. Une seconde implémentation a été réalisée sous la forme d'un prototype utilisant un ensemble de cartes Digilent avec des FPGA Xilinx Spartan3-1000 et des mémoires SRAM de 1Mo. Ces réalisations nous ont permis de valider notre approche et d'évaluer un certain nombre de critères de performance. Un point important concerne le coût de l'utilisation d'un micronoyau par NPU, qui impacte finalement que très peu le débit des applications que nous avons testées (autour de 1% de chute de performance) et introduit également une légère variabilité du temps d'exécution des tâches.

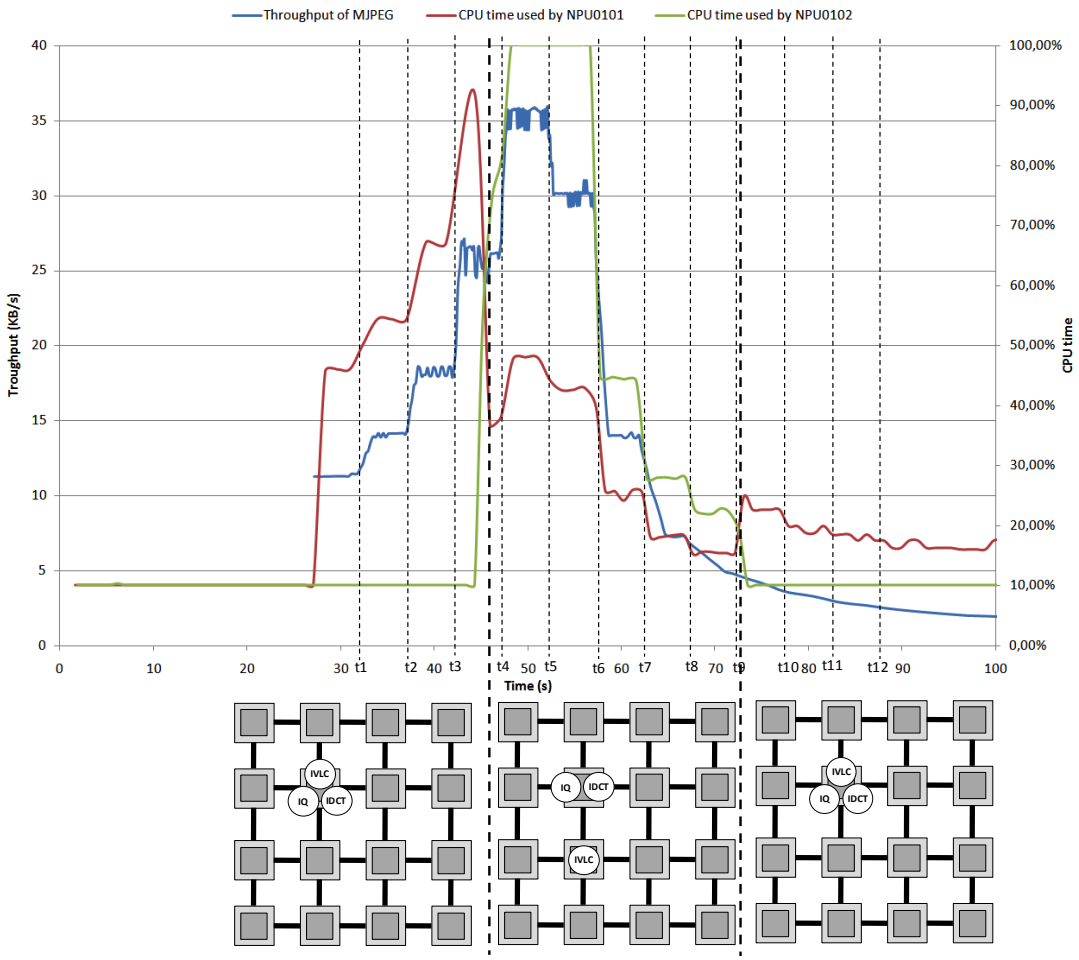



**Figure 10.** Scénario d'exécution d'une application MJPEG sur HS-Scale

L'exemple de la **Figure 10** montre le résultat de l'utilisation d'une politique de migration de tâches basée sur la charge CPU **[7]**. Nous avons dans cette expérience fait varier le débit des données entrantes afin d'observer comment le système réagit lors de l'exécution d'une application de décompression MJPEG. Au commencement, toutes les tâches (IVLC, IQ et IDCT) sont prises en charge par le NPU(1,1) avec un débit entrant relativement faible, qui implique un temps CPU de l'ordre de 47%. Ce débit est augmenté progressivement (t1, t2 and t3) et se caractérise par un accroissement de la charge CPU à chacune de ces étapes. Lorsque cette dernière dépasse le seuil fixé de 80%, l'OS du NPU(1,1) déclenche une migration de la tâche qui nécessite le plus de ressources, sur un NPU voisin. Dans cet exemple, c'est la tâche IVLC qui est déplacée vers le NPU(1,2), qui implique donc une diminution de la charge du NPU(1,1) de 35% et une augmentation de celle du NPU(1,2). A t4, le débit d'entrée augmente encore, et avec, celui du MJPEG, qui atteint 35Ko/s ce qui surcharge totalement le NPU(1,2) ; ce temps CPU de 100% n'entraine cependant pas de nouvelle migration puisqu'il n'y a qu'une seule tâche sur ce NPU. De t5 à t12, le débit d'entrée est diminué peu à peu : lorsque la charge du NPU(1,2) passe en dessous de 20%, l'OS décide de déplacer la tâche sur le NPU d'origine NPU(1,1). Après cela, nous observons la diminution du temps CPU du NPU(1,2) et en parallèle une augmentation de celui du NPU(1,1) mais sans saturation. Cet exemple très simple permet de voir comment avec un service de monitoring logiciel et un actionneur de type migration de tâche, le système est capable de s'adapter à des conditions de fonctionnement changeantes.

### 1.2.2 GENEPY

Ces travaux en collaboration avec le CEA LETI se situent dans le contexte des applications Télécom, en particulier sur la conception du Modem pour les réseaux cellulaires de 4ème génération (3GPP-LTE). La norme définit un ensemble de spécifications telles un débit de données élevé, une faible latence, en s'appuyant sur des techniques OFDM/MIMO avec modulation adaptative. L'architecture bande de base requiert de fait des capacités de reconfiguration dynamique, des ressources pour des calculs spécifiques devant être réalisés en temps réel, mais aussi une faible consommation électrique. Si l'on se focalise sur l'étage de démodulation, c'est une charge de 10 GOPS pour un budget de 200 mW.

Pour atteindre ce niveau d'exigences, on avait recourt par le passé à des solutions dédiées, mais l'évolution des densités d'intégration a ouvert de nouvelles voies vers des approches plus flexibles telles que les MPSOC, par exemple Picochip **[8]**, MuSIC **[9]** ou Sandbridge SB3011 **[10]** ; ces solutions bien que flexibles avaient des consommations bien trop élevées incompatibles avec une utilisation sur terminal mobile. MAGALI **[11]**, que l'on peut qualifier de MPSOC hétérogène s'appuie sur des cœurs de DSP programmables et des accélérateurs matériels spécifiques et configurables. Ce composant est également basé sur un processeur centralisé pour le contrôle des différentes opérations, ce qui limite sa *scalabilité*.

L'objectif de ce travail de thèse était de concevoir une architecture homogène avec des unités de calcul et de contrôles distribuées, afin d'améliorer la *scalabilité* des solutions existantes. L'enjeu était d'atteindre ce but tout en garantissant les contraintes fixées par le LTE.

- **Architecture matérielle**

Pour cela, l'architecture de GENEPY utilise les mêmes concepts généraux que HS-Scale : il s'agit d'un réseau MPSOC homogène et entièrement distribué, basé sur un réseau d'interconnexion NOC GALS. La spécificité des applications ciblées impose cependant une attention toute particulière à l'élément de calcul de base. Si celui-ci reprend certains principes de HS-Scale, il se distingue néanmoins par plusieurs aspects : les éléments de calcul et la manipulation des données.

L'unité de base se nomme SMEP : elle intègre un Smart Memory Engine (SME) et deux DSP. Les SME gèrent à travers des *process* de lecture/écriture (*RP* et *WP*) des buffers circulaires de données configurables sur une mémoire locale de 32Ko. Un bus de type *crossbar* permet d'interconnecter l'ensemble de ces éléments. Dans une première version de GENEPY, le SMEP version 0 était piloté par un processeur hôte centralisé. Bien que l'utilisation d'un PE unique répliqué N fois soit une première étape de régularisation de l'architecture, il n'en résulte cependant pas une parfaite homogénéité telle que souhaitée.

C'est la raison pour laquelle une deuxième version de GENEPY a été proposée. Celle-ci innove par l'adjonction d'un contrôleur local qui prend la forme d'un processeur MIPS 32 bits, autorisant ainsi une gestion parfaitement décentralisée du système et donc la suppression du processeur hôte. Le rôle du processeur local consiste à gérer l'ordonnancement requis par les procédés de modulation dynamique des applications Télécom ciblées. Il est également chargé de régir le flot de contrôle entre l'ensemble des unités, garantir les contraintes de temps réel, etc. Ce qui distingue cette approche de la version 0 du SMEP, c'est qu'elle autorise un contrôle logiciel, conférant une flexibilité plus grande dans le pilotage de la structure. La version 1 du SMEP est illustrée dans la **Figure 11**. Une mémoire locale de 8Ko permet le stockage du programme du MIPS, et un module se charge, suivant les instructions du processeur, des configurations du SMEP (**Figure 12**). Cette unité dispose également d'une unité de « power management » qui comme son nom l'indique peut gérer les actionneurs locaux tels que la fréquence et la tension d'alimentation du SMEP.

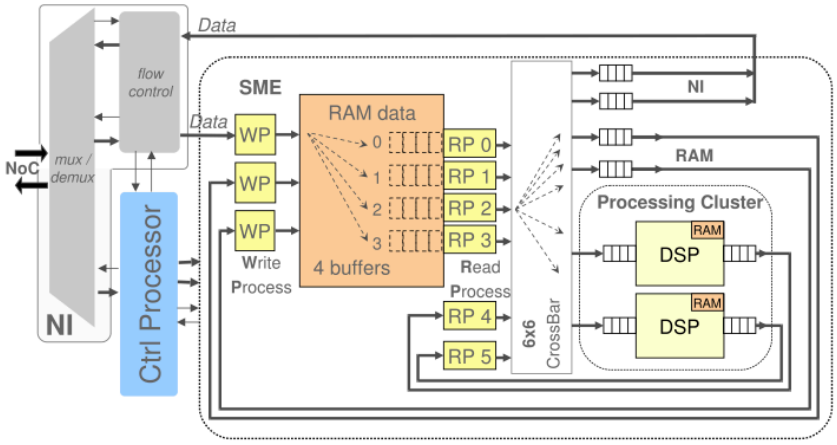



**Figure 11.** Architecture du SMEP

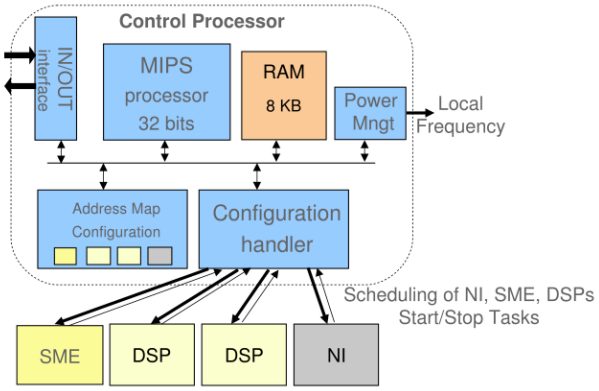



**Figure 12.** Le contrôleur local basé sur le processeur MIPS

- **Programmation de GENEPY**

D'un point de vue logiciel, ce MPSOC a été pensé pour des applications de traitement du signal orientées « flots de données ». A partir d'un DFG (Data Flow Graph), et suivant la nature des tâches à effectuer, c'est à la charge du programmeur à travers une chaîne d'outils dédiés, de répartir les nœuds et les arcs de ce DFG aux différents blocs matériels, chacune des tâches étant décrite et compilée séparément à partir d'un code C, assembleur, ou de fichiers de configurations. C'est finalement à travers le code du contrôleur MIPS que s'ordonnance toutes les opérations, puisque c'est ce dernier qui devra en fonction des évènements déclencher de nouvelles configurations des unités du SMEP.

- **Résultats**

La plateforme a été implémentée dans une technologie ST65nm Low-Power, pour une fréquence de fonctionnement à 400MHz (**Figure 13**).

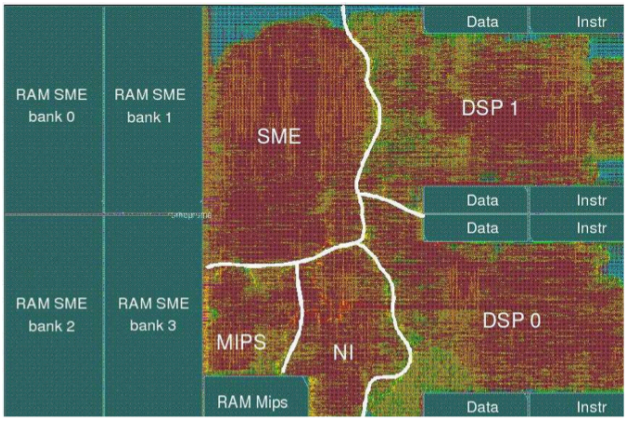



**Figure 13.** Layout du SMEP en technologie ST65nm

Le SMEP occupe une surface totale de 3,1 mm2 et consomme en moyenne 110,3 mW. Il est à noter que la partie contrôle ne représente qu'un faible pourcentage des ressources (~5% de la surface) et 1% de la consommation totale. Il est intéressant de constater que par rapport au circuit de référence (MAGALI), pour des fonctionnalités et une puissance de calcul équivalentes, la surface est moindre d'environ 15% (ce qui s'explique par le regroupement des ressources au sein d'une même unité, qui implique une économie de ressources au niveau du NOC). On soulignera également le coût très faible qu'implique l'adjonction du contrôleur MIPS par rapport au contrôleur dédié de la version 0 du SMEP (3% de surface supplémentaire uniquement pour le SMEP).

Afin d'évaluer les performances sur un cas d'étude réel, la partie démodulation du récepteur OFDM/MIMO a été implémentée dans GENEPY. Compte-tenu des contraintes de temps réel fixées par le standard, les différentes tâches impliquées requièrent de grosses puissances de calcul, qui de plus varient au gré des modes de configuration (Mode 0 étant le plus simple, jusqu'au mode 5 pour de la haute qualité de réception). Les 3 tâches considérées (Channel Estimation, Interpolation et MIMO) doivent traiter les différentes trames TTI avec pour support matériel 2 SMEP et 3 types de gestion des ressources : basic, pipeline et *mapping* dynamique. Ces trois scénarios sont illustrés sur la **Figure 14**.

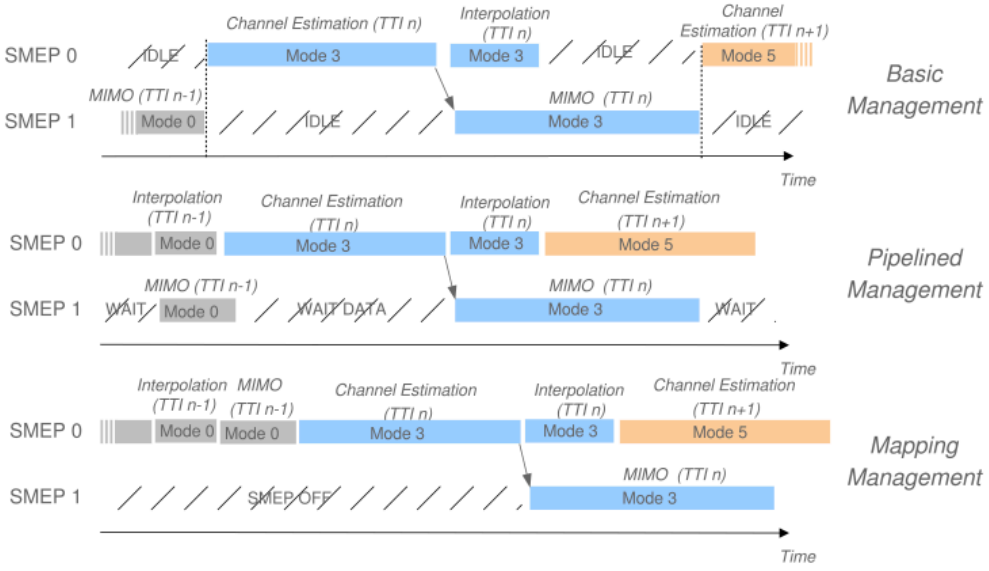



**Figure 14.** Les 3 scénarios considérés dans ce cas d'étude

Dans un cas de gestion basique des ressources, chaque trame TTI est traitée l'une après l'autre. C'est un cas typique d'ordonnancement tel qu'on le trouve dans des architectures disposant d'un contrôle centralisé. La technique de pipeline va permettre de déclencher des calculs sur un processeur dès que cela est possible, grâce à la prise en compte d'évènements tels que la disponibilité de données, la fin d'un processus, etc. Cette approche permet donc d'optimiser l'utilisation des ressources. On peut

également tirer avantageusement parti des mécanismes de DVFS, en étirant le temps (donc en réduisant la fréquence) lorsque cela est possible, ce qui permet par ailleurs de diminuer la tension d'alimentation et donc l'énergie consommée. Le dernier cas correspond à une allocation dynamique. En mode 0, la partie MIMO est beaucoup moins gourmande que dans les autres modes : la chaîne de traitement complète d'un TTI peut être entièrement prise en charge par un seul SMEP en mode 0. Ainsi il est possible de tirer avantageusement parti de techniques de *clock-gating* qui vont permettre une économie d'énergie conséquente. Les résultats obtenus **[12]** montrent que sur l'ensemble des unités considérées, le mode pipeline permet une économie d'énergie jusqu'à 30% par rapport à l'approche basique, et pour l'allocation dynamique, celle-ci atteint 37%.

### 1.2.3 Open-Scale

Les travaux de thèse de Nicolas Saint Jean avaient permis de poser les premiers jalons d'une architecture MPSOC homogène avec des capacités d'adaptation, notamment par migration de code en fonction de critères tels que la charge des processeurs. Ces travaux ont pris une orientation applicative à travers la thèse de Camille Jalier, où l'architecture globale reprenait les principes généraux de *HS-Scale*, en lui adjoignant des éléments de calculs et de gestion des données spécifiques au domaine ciblé (Telecom). L'adaptation était également au cœur des préoccupations de cette thèse puisque l'utilisation d'un contrôleur local programmable étendait les possibilités de gestion dynamique des ressources.

La thèse de Gabriel Marchesan Almeida se situe dans un contexte plus général que GENEPY, et en ce sens elle constitue une suite logique aux travaux sur *HS-Scale*. Les contributions se situent sur 2 plans que nous traiterons dans 2 sections différentes dans ce manuscrit : la première concerne le sujet, à savoir la conception de MPSOC homogène et c'est celle-ci que nous aborderons dans cette partie. La deuxième adresse une problématique d'optimisation que nous présenterons plus tard. A noter que ces travaux font également référence aux travaux de thèse de Lyonel Barthe, principal artisan de la conception du *SecretBlaze* qui sera le cœur de l'architecture d'*Open-Scale*.

- **Système d'exploitation temps réel ADEPT**

Une contribution de ce travail concerne tout d'abord le système d'exploitation entièrement refondé à partir d'un code open-source fourni par l'auteur du processeur plasma utilisé par le NPU de *HS-Scale*. Basé sur une architecture modulaire, plusieurs services furent développés et inclus afin de satisfaire nos objectifs de compacité et d'adaptabilité. On retrouve ainsi des fonctionnalités déjà disponibles dans la version précédente (chargement dynamique, ordonnancement préemptif, mécanismes de communications entre tâches locales, etc.) et des nouvelles, notamment :

- *une interface de communication haut niveau par passage de message (MPI)*

Ce sont deux fonctions, *MPI_Send()* et *MPI_Receive()* accessibles au niveau applicatif pour implémenter des applications flot de données sous la forme de réseaux KPN. La **Figure 15** suivante illustre une utilisation possible de ces fonctions.

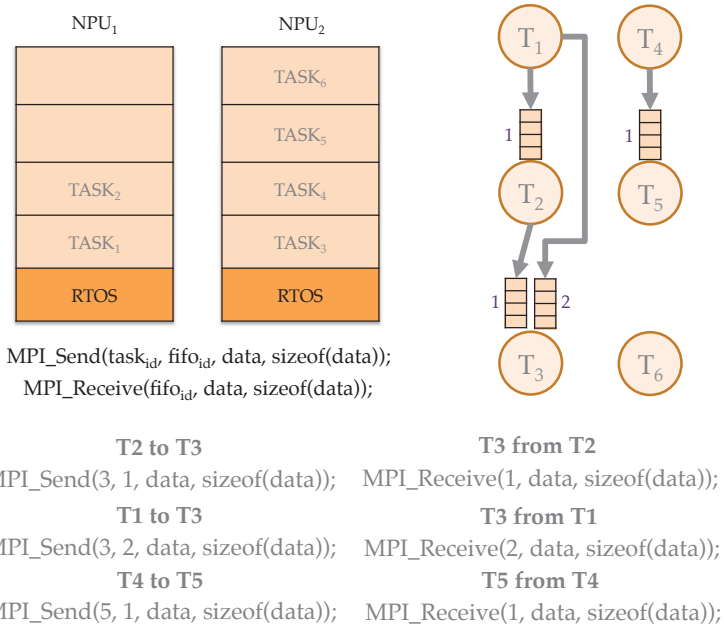



**Figure 15.** Fonctions MPI

- *des fonctions de communication inter NPU s'appuyant sur différents protocoles : RAW, TCP et UDP*

En fonction de la qualité de service requise ou de la performance, il est possible de configurer différents protocoles de communications.

- *une API légère (DRET) pour gérer les bases de données de monitoring stockées en mémoire*

La DRET (Distributed Raw Event Table) est une base de données associée localement à chaque NPU **[13]**. Elle est conçue en vue du stockage des données issues de différents types de capteurs (logiciels et matériels). L'API offre toutes les fonctions pour manipuler ces bases de données (création de table, d'attributs, d'insertion de données, d'association de données, de recherche, de tri, etc.). Compte-tenu de sa finalité et de son intégration au sein d'un système embarqué, cette API fut développée avec pour contrainte majeure la compacité du code, et la performance. On peut voir ainsi le nombre de cycles requis pour effectuer un certain nombre de fonctions de cette API (**Figure 16**).

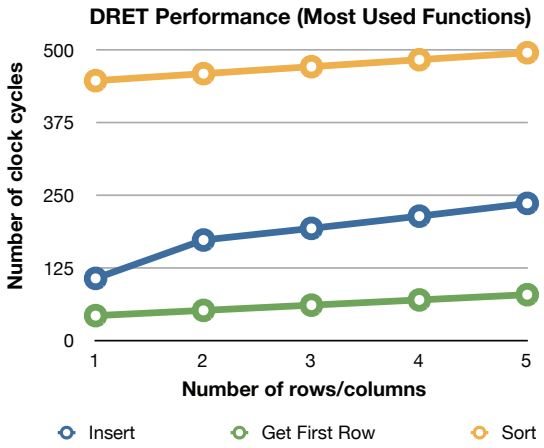



**Figure 16.** Evaluation des performances des fonctions de la DRET

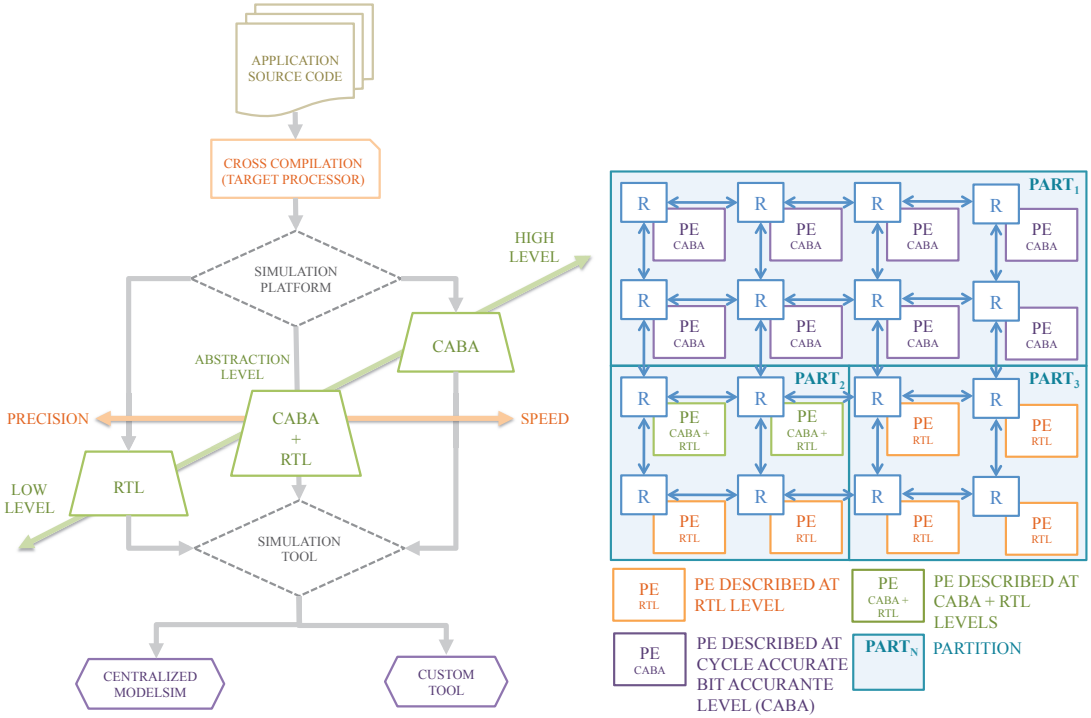
- *un mécanisme de monitoring supplémentaire mesurant le débit de l'application*

- **Environnement de conception et de simulation multi-niveaux**

Une problématique récurrente de conception concerne l'exploration et l'évaluation des différentes solutions potentielles. Les systèmes MPSOC intègrent les 3 axes possibles d'étude qui sont les aspects matériels, les aspects logiciels et le *mapping* (on retrouve les 3 axes AAA de l'adaptation algorithme architecture). Mais compte-tenu de la complexité croissante de ces systèmes, il est nécessaire de mettre en œuvre des méthodologies permettant de passer facilement d'un niveau d'abstraction à un autre, afin de jouer sur le compromis précision / vitesse de simulation. C'est dans cette optique qu'a été mis en œuvre un nouvel outil de simulation permettant cette flexibilité avec en plus la possibilité de mélanger les différents modèles et autoriser ainsi des raffinements ciblés en fonction des évaluations que l'on cherche à faire **[14]**.

A partir du modèle initial de *HS-Scale*, les modèles suivants ont été intégrés à cet environnement de simulation décrit dans la **Figure 17 (a)** :

- *modèle RTL : il s'agit de la description du système MPSOC en VHDL synthétisable*
- *modèle CABA : il utilise un ISS pour le processeur, et une description SystemC du NOC*
- *modèle mixte : il se base sur un modèle CABA pour pour le processeur et la mémoire via un ISS, le NOC étant lui décrit en VHDL*




(a) (b)

**Figure 17.** Flot de simulation multi-niveaux (a) et exemple d'utilisation (b)

Un exemple de simulation hétérogène est également illustré dans la **Figure 17 (b)**. Cette plateforme de simulation a permis notamment d'étudier et de mettre en œuvre différentes heuristiques de *mapping* de tâches [**15**].

- **Implémentation matérielle**

Le processeur utilisé dans la version initiale du NPU présentait plusieurs limitations (étages de pipeline, bus non standardisé, faible modularité du design, etc.). Dans le cadre de travaux sur la sécurité qui seront développés plus tard dans ce manuscrit, nous avons été amenés à concevoir un processeur disposant d'un jeu d'instruction compatible avec celui du MicroBlaze **[16]**. Il s'agit d'une architecture HARVARD RISC 32 bits, avec 5 étages de pipeline (*Fetch*, *Decode*, *Execute*, *Memory Access* et *Write-Back*) telle qu'elle est décrite sur la **Figure 18** avec son architecture mémoire **Figure 19**. Il dispose d'un certain nombre de configurations matérielles qui permettent de jouer sur le compromis surface/performance (*barrel-shifter* et multiplieur câblé 1 cycle, diviseur câblé, prédiction de branchement dynamique, cache optionnel *direct-map* avec politiques *write-back* et *write-through*, gestionnaire d'interruption, contremesures matérielles, etc.). Un ensemble d'optimisations ont été mises en œuvre au niveau RTL, lui permettant d'atteindre des performances similaires au MicroBlaze.

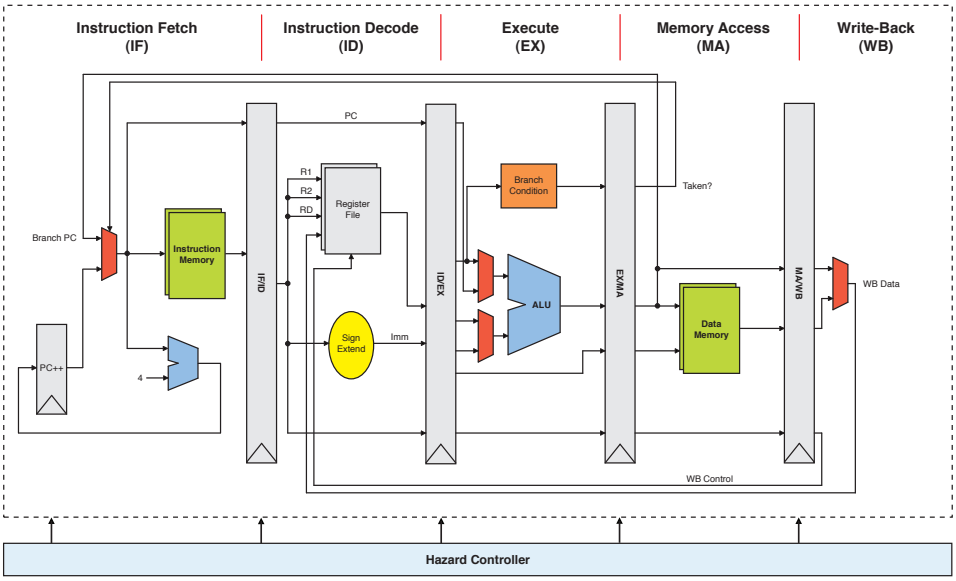



**Figure 18.** Chemin de données du *SecretBlaze*

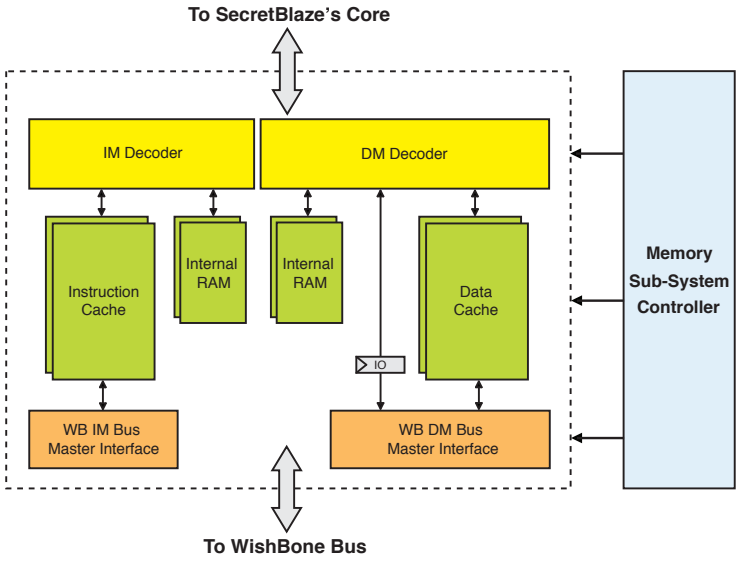



**Figure 19.** Sous-système mémoire du *SecretBlaze*

Il a été implémenté pour différentes cibles technologiques, en tout premier lieu pour FPGA où il atteint une fréquence de fonctionnement de 90MHz sur Spartan3 (technologie 90nm) et 240MHz sur Virtex5 (technologie 65nm). Ses performances ont été comparées avec d'autres architectures de processeurs, notamment l'OpenRISC 1200, le MB-lite qui est un clone du MicroBlaze, qui ont été implémentées dans des configurations équivalentes sur Spartan3, avec mémoires distribuées. Les performances sont celles obtenues après placement/routage des processeurs (**Figure 20**). Nous avons également établi une comparaison plus approfondie avec le LEON et le MicroBlaze **[17]**.

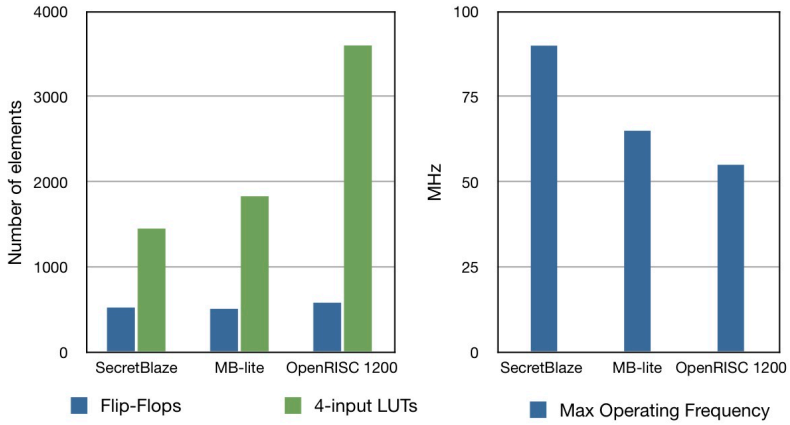



**Figure 20.** Comparaison du SecretBlaze, MB-lite et OpenRISC 1200

Une version de notre système MPSOC homogène basé sur le *SecretBlaze* a donc été développée. Une nouvelle architecture du NPU a dans un premier temps été mise en place : elle est illustrée dans la **Figure 21**. Le hardware correspond à une structure régulière de processeurs *SecretBlaze* connectés à travers le réseau NOC HERMES. La **Figure 22** montre le taux d'occupation d'un FPGA Xilinx Virtex 5 LX 110T en fonction du nombre de NPU, pour une version du *SecretBlaze* utilisant le multiplieur câblé, le *barrel-shifter*, 8Ko pour les caches de données et d'instructions, et 64 Ko de mémoire RAM interne. Les FIFO du NOC sont configurées en 256 mots de 32 bits. La mémoire limite l'intégration d'un nombre supérieur d'unités. Le RTOS ADEPT occupe suivant les configurations et options choisies entre 47Ko et 57Ko, ce qui représente une part importante de la mémoire embarquée. A ce titre on note qu'une approche pragmatique consisterait à introduire une hiérarchie dans la méthode de gestion, et les capacités individuelles de chaque nœud (on pourrait tout à fait envisager de n'allouer le RTOS qu'à un ensemble limité de NPU qui auraient la charge de répartir les tâches sur l'ensemble des processeurs). Les performances temporelles du système ont également été évaluées afin d'estimer le coût de l'utilisation des couches logicielles intermédiaires (RTOS et API de communication). Ces résultats sont analysés dans [**18**].

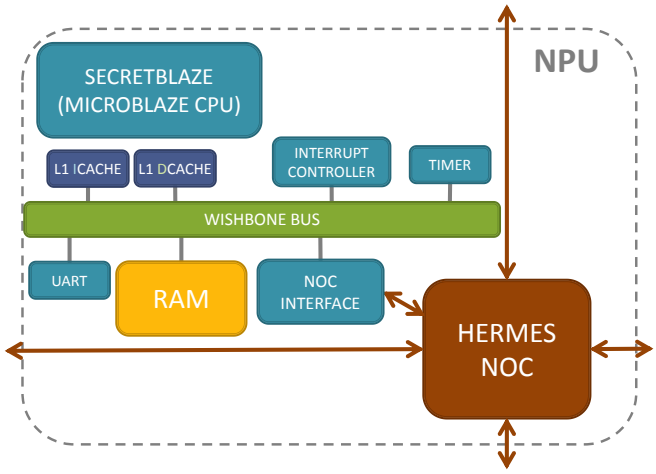



**Figure 21.** Architecture du NPU

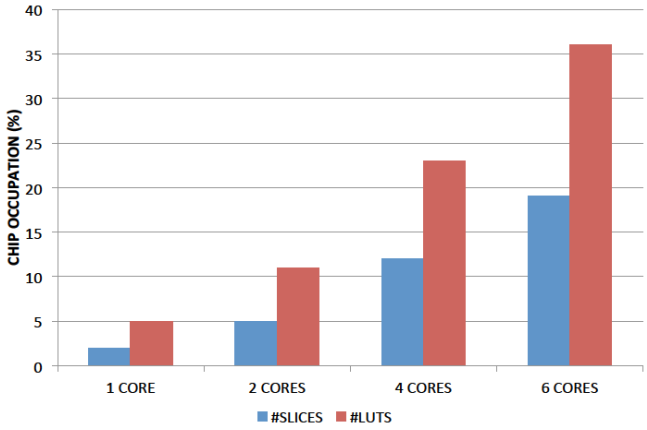



**Figure 22.** Taux d'occupation de Open-Scale en fonction du nombre de NPU sur Virtex 5 LX 110T

## 1.3 Bilan

Dans cette partie, nous avons présenté une synthèse des travaux relatifs à la conception d'architectures reconfigurables et adaptatives. Après avoir introduit le contexte et le positionnement des mes travaux de thèse, une première contribution concernant les métriques a été exposée. Nous avons ensuite montré notre approche concernant l'exploitation dynamique de différents niveaux de parallélisme par le biais d'une méthode de multiplexage matériel, qui a mené à une intégration système et une évaluation des gains de performance obtenus. La prise en compte d'un parallélisme au niveau tâche nous a naturellement amené vers des architectures de type MPSOC : nous avons développé une approche reposant sur une structuration régulière des ressources de traitement et de communication afin de faciliter le passage à l'échelle. Nous avons ainsi présenté plusieurs contributions dans ce domaine, l'architecture HS-Scale, premier prototype mettant à disposition à la fois le matériel et l'interface logicielle de programmation d'une architecture entièrement distribuée. Ce modèle de ressources régulières distribuées a été adapté dans le cadre d'une collaboration avec le CEA LETI afin de concevoir une architecture basée sur des DSP ciblant des applications de télécommunication. Enfin, notre architecture générique a été optimisée, notamment le processeur SecretBlaze et le système d'exploitation ADEPT, qui forment ensemble le système Open-Scale que nous avons mis à disposition de la communauté. Le travail présenté dans cette section a été valorisé à travers plusieurs publications[10] :

- 5 revues, dont 3 RICL: [J6][J8][J9][J11][J12] ([J8] joint en annexe de ce document (cité 43 fois))
- 3 chapitres d'ouvrage : [CO1] [CO5] [CO6]
- 25 CICL : [C18] [C19] [C24] [C25] [C32] [C34] [C35] [C36] [C41] [C42] [C50] [C53] [C56] [C57] [C58] [C62] [C63] [C64] [C66] [C67] [C70] [C71] [C72] [C76] [C77]

[10] cf. Partie 1, section 9 pour le détail des publications

# 2 OPTIMISATION DYNAMIQUE DES SYSTEMES ADAPTATIFS

La première partie des travaux exposés a mis en avant plusieurs contributions dans le domaine de la conception d'architectures. Ce chapitre qui couvre une période de 10 ans est caractéristique de la loi de Moore. D'un parallélisme de niveau bit (au niveau porte logique pourrait-on dire), nous avons exploré des solutions au niveau mot (niveau opérateur arithmétique), jusqu'à des motifs de calcul plus complexes (niveau boucle et cœur de boucle). Ce travail sur les architectures reconfigurables intégrait déjà des notions d'adaptabilité, en vue notamment d'améliorer l'utilisation des ressources et la performance. Le niveau de complexité de la gestion du parallélisme est ensuite passé d'un niveau matériel à un niveau logiciel, avec l'apparition d'architectures MPSOC, pour lesquelles nous avons œuvré dans le sens de l'homogénéisation des ressources dans un souci de *scalabilité*, déjà au cœur de nos préoccupations au cours de ma thèse. Le processeur devenait le nouveau transistor et l'agencement régulier de ces unités devait faciliter le passage à l'échelle. Cette philosophie générale devait être le fondement d'une approche permettant l'adaptation du système à plusieurs niveaux, et ce dans le but d'en optimiser ce qui devenait l'objectif premier des concepteurs : la consommation.

C'est donc naturellement que nous nous sommes intéressés à des approches qui pourraient permettre de garantir un certain niveau de performance et cela avec une consommation minimale. Notre apport se situe essentiellement dans la mise en œuvre de méthodes permettant de contrôler et d'optimiser le système en cours de fonctionnement. Pour cela, on considère que ce dernier fonctionne en boucle fermée : un ensemble de moniteurs observent le circuit, et permettent d'établir un diagnostique de l'état du système qui, suivant un objectif prédéfini (par exemple garantir une contrainte temps-réel en optimisant la consommation), va mener à une prise de décision quant à la mise en œuvre d'un certain nombre d'actions (telles que la migration de tâche, ou encore l'affectation d'un couple tension-fréquence). Ce principe est illustré par la **Figure 23**.

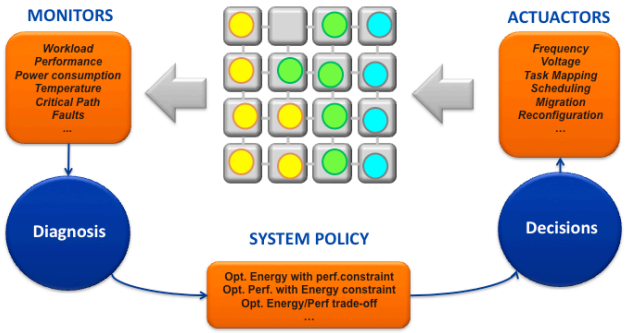



**Figure 23.** Adaptation dynamique d'un système MPSOC

La plupart des travaux antérieurs se basent généralement sur des approches centralisées de l'optimisation (**Figure 24**), tout simplement parce-que ce processus nécessite une vision complète de l'état des ressources à optimiser. Il est donc assez logique d'adopter une méthode qui pourra garantir une solution optimale. On trouve dans la littérature des contributions qui adressent l'ajustement de paramètres tels que la fréquence, la tension, le *mapping* et l'ordonnancement de tâches sur des systèmes MPSOC à partir d'algorithmes centralisés. C'est le cas par exemple de **[19]** qui présente une méthode pour sélectionner le couple V et F à partir d'une optimisation non linéaire de Lagrange. Dans **[20]**, il s'agit d'une technique inspirée des lois de Kirchhoff pour appliquer des configurations de DVFS. Ces techniques en ligne deviennent vite problématiques lorsque le nombre de cœurs augmente : il s'agit tout d'abord d'une question de complexité algorithmique mais aussi de centralisation de l'information, qui va nécessiter des transferts de données conséquents, qui vont introduire une certaine latence, et une pénalité de performance pour le système que l'on cherche à optimiser.

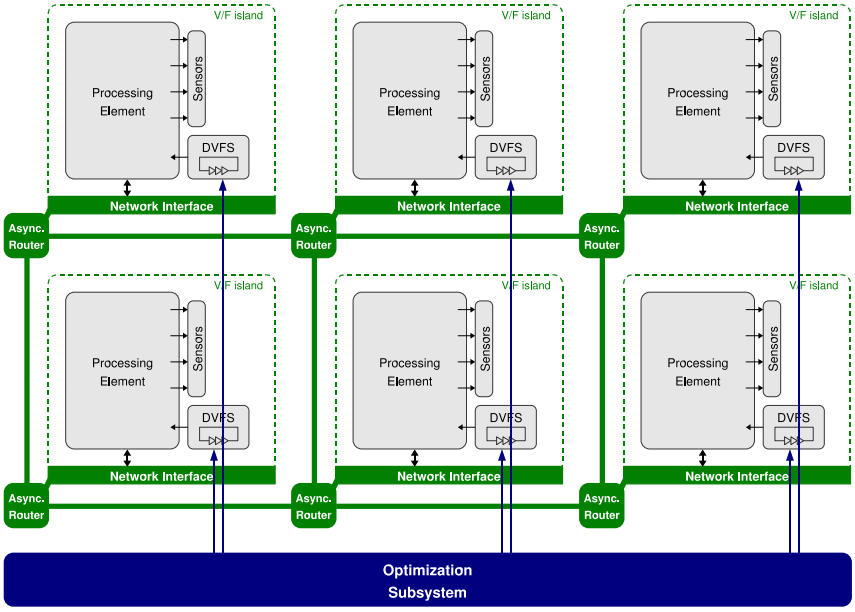



**Figure 24.** Optimisation en ligne centralisée

Par essence, un MPSOC est un système qui dispose de plusieurs unités de calcul. Se pose alors la question de savoir comment tirer parti de ce bénéfice dans un but d'optimisation ? La littérature nous donne quelques pistes, comme la résolution distribuée d'un problème d'optimisation convexe pour le contrôle thermique d'un circuit **[21]**, ou bien encore des méthodes stochastiques basées sur des chaînes de Markov **[22]**. Mais ces solutions qui fonctionnent bien sur un faible nombre d'unités de calcul, montrent quelques limitations dans leur passage à l'échelle. C'est la raison pour laquelle nous avons cherché à proposer de nouvelles méthodes distribuées (**Figure 25**) qui s'inspirent par exemple de l'automatique ou des mathématiques appliquées, avec pour motivation sous-jacente de maîtriser le coût de mise en œuvre. Ces différentes approches seront présentées dans cette partie, avec les principaux résultats obtenus. L'optimisation étant un problème multi-échelle, nos solutions adressent l'affectation globale des paramètres, mais aussi l'ajustement local tel que nous le verrons dans la mise en œuvre d'un mécanisme de contrôle pour des technologies FD-SOI.

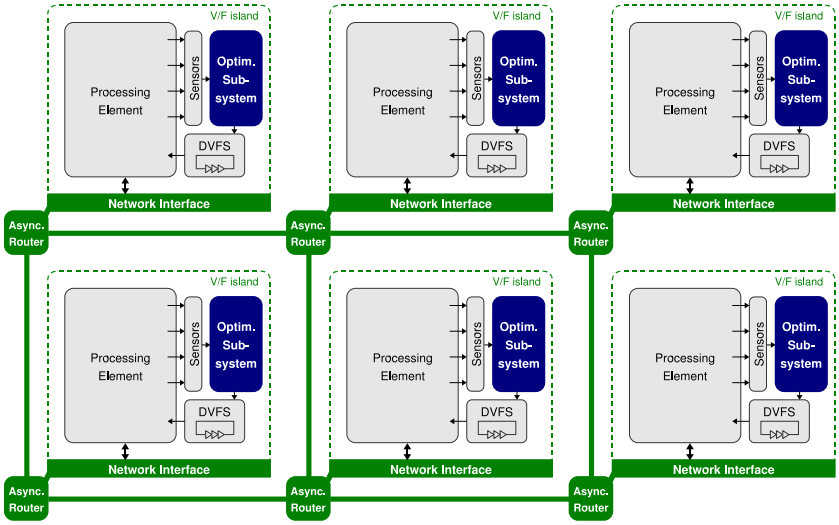



**Figure 25.** Optimisation en ligne distribuée

L'objectif d'efficacité énergétique requiert la garantie d'un niveau de performance minimal ajusté aux besoins de l'application. On s'appuie pour cela sur un modèle de contrôle dynamique qui nécessite une connaissance ou une mesure de l'énergie consommée globalement et par unité de calcul, pour

guider la prise de décision locale en vue d'adapter la vitesse du circuit et ce en fonction du contexte applicatif.

La prise de décision doit permettre de trouver la configuration optimale du système qui minimise sa dissipation sans nuire au déroulement de l'application, elle se ramène donc à un problème d'optimisation sous contrainte. La configuration du système concerne sa fréquence de fonctionnement. Avec l'intégration de plusieurs cœurs dans une même architecture, cet aspect multidimensionnel accentue les difficultés liées à la mise en place de ce contrôle, notamment le processus de prise de décision.

## 2.1 Contrôleur PID

Le système considéré fonctionne en boucle fermée, puisque un ensemble de moniteurs permettent d'évaluer l'état (mesure) et de le comparer à la consigne issue de la politique d'optimisation, avant de réguler et d'appliquer en conséquence une commande afin de rapprocher la mesure au plus prêt de la consigne. Ce mode de fonctionnement fait directement penser à une technique élémentaire de l'automatique, à savoir le contrôle en boucle fermée. C'est la raison pour laquelle on trouve dans l'état de l'art une application des principes de régulation au problème d'ajustement des paramètres du DVFS par exemple **[23][24]** ou encore au problème de la gestion du trafic dans les NOC **[25][26]**. Dans un système intégré, on peut imaginer de nombreuses sources de perturbation lorsque l'on cherche à garantir un certain nombre de critères. On peut imaginer par exemple qu'il s'agisse d'un critère de consommation maximale à ne pas dépasser, dans ce cas une augmentation de la température environnante entrainerait un accroissement de la consommation statique (donc totale et donc potentiellement supérieure à la limite fixée) qu'il faudrait compenser afin de garantir la contrainte. Une première approche utilisant un régulateur PID apparaît donc comme un choix judicieux dans l'optique d'adaptation du système.

La **Figure 26** illustre le schéma de principe de la méthode. Le SETPOINT correspond à une consigne de débit applicatif, c'est à dire le nombre de données qu'une application est censée produire par unité de temps. La commande correspond à la valeur de fréquence d'horloge à assigner à l'élément de calcul. PROCESS est l'exécution d'une tâche sur un NPU qui produit un certain débit. Celui-ci est mesuré par un service dédié de l'OS ADEPT puis comparé à la consigne. La différence entre les deux est ensuite traitée par un régulateur PID qui vient modifier en conséquence la fréquence du DVFS. Dans cette approche, on considère qu'à chaque fréquence (ou chaque sous-ensemble de fréquences) est associé une gamme de tension, ce qui permet de réduire la puissance consommée et l'énergie.

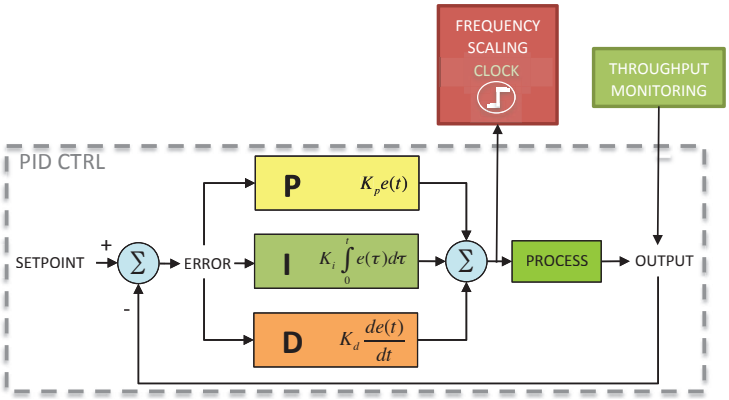



**Figure 26.** Principe du contrôleur PID utilisé pour la commande de *Frequency Scaling*

La **Figure 27** donne une illustration de l'approche distribuée du contrôle PID. Dans cet exemple, une tâche est affectée par NPU. A chaque tâche est associé un contrôleur PID (ce qui suppose qu'une exécution multitâche requerrait plusieurs contrôleurs PID par NPU) qui prend la forme d'un service de l'OS dont l'empreinte mémoire représente moins de 1%. Pour synthétiser le régulateur, il est nécessaire d'effectuer une simulation du MPSOC (SystemC) correspondant à l'exécution de la tâche par un NPU. La réponse à un échelon de fréquence permet de mesurer la réponse en débit de la tâche et de fixer la fonction de transfert du système « tâche – NPU », et à partir de là, un ajustement des paramètres du PID (les coefficients $K_p$, $K_i$ et $K_d$) est réalisé par un modèle sous Simulink.

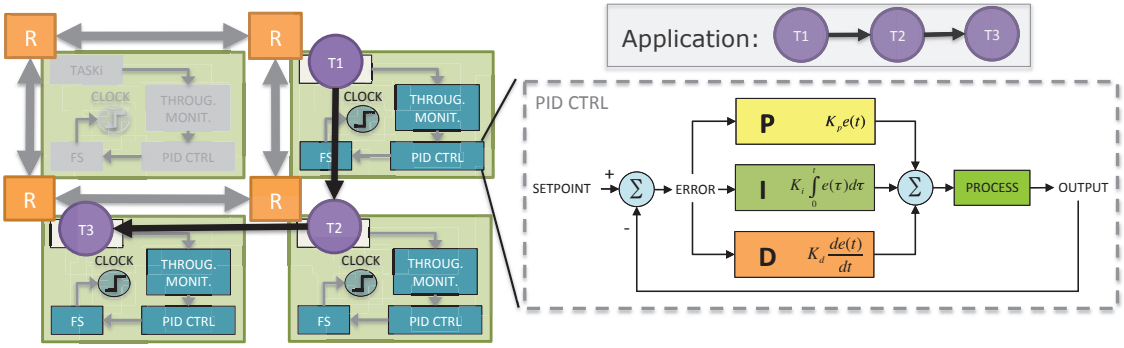



**Figure 27.** Intégration distribuée du PID à *OpenScale*

Un cas d'étude permet de comprendre l'intérêt de l'approche. L'idée est d'observer la robustesse du régulateur face à des perturbations. Comme l'un des actionneurs d'*OpenScale* est la migration de tâches, celle-ci est susceptible d'entrainer une diminution soudaine des performances lorsqu'une tâche se retrouve à partager les ressources processeurs suite à une migration. Afin de maintenir la consigne de débit, l'action va consister à augmenter la fréquence jusqu'à ce que la performance soit rétablie. Dans l'exemple proposé, il s'agit d'une application MJPEG dont le débit va être perturbé par l'arrivée d'une tâche sur le NPU hôte. Suivant la charge CPU requise par cette nouvelle tâche, la perturbation va être plus ou moins forte. La **Figure 28** montre l'effet d'une perturbation de ce type sur le débit de l'application MJPEG, avec et sans adaptation de fréquence utilisant le correcteur PI.

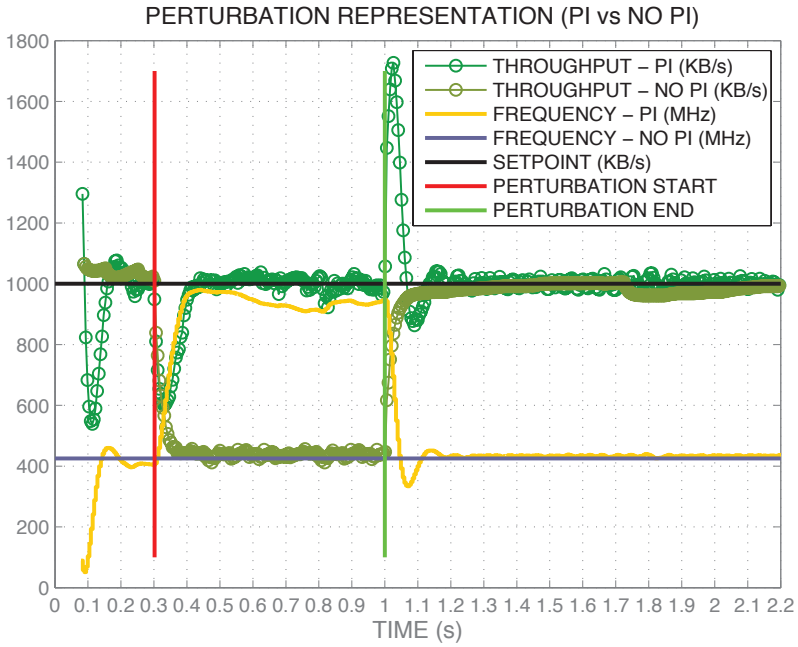



**Figure 28.** Illustration de la réponse en débit à une perturbation avec et sans adaptation

Il est intéressant de noter que sans régulation, il est nécessaire de configurer le système afin de lui permettre de répondre aux situations « pire cas », ce qui revient par exemple ici à affecter une fréquence plus élevée garantissant, quelle que soit la perturbation, que la consigne sera respectée. Par rapport au scénario le plus pessimiste, nous avons des estimations en technologie ST65nm qui montrent que l'utilisation de contrôleurs PID permettent de réduire de 30% à 40% la consommation par rapport à une approche nominale. L'ensemble des résultats correspondants est disponible dans **[27]**.

## 2.2 Optimisation inspirée de la théorie des jeux

L'approche PID présente un potentiel vraiment intéressant dans la capacité de régulation d'un élément de calcul vis à vis de perturbations, tel que cela a été présenté dans la section précédente. En revanche, cette méthode n'apporte pas de solution au problème de l'optimisation distribuée, où un ensemble de processus composés de multiples tâches s'exécutent sur de nombreuses unités de calcul avec des contraintes de performances et de consommation variables. Rappelons que l'objectif est de fournir une méthode distribuée qui puisse s'affranchir facilement des facteurs d'échelle, et que celle-ci doive garantir l'optimalité de la solution. Un autre point essentiel concerne la stabilité d'un système dont la prise de décision s'effectue de manière distribuée. Comment garantir par exemple, si l'on considère comme actionneur la migration de tâches, que les tâches ne se déplacent pas indéfiniment dans la grille ? Dans cette optique, il était nécessaire de formaliser les choses à travers un cadre mathématique, et notre premier choix s'est porté sur la théorie des jeux.

La théorie des jeux est une branche des mathématiques appliquées qui fournit un ensemble d'outils décrivant les interactions entre agents rationnels. Ce cadre suppose que les joueurs cherchent à atteindre un objectif bien défini en tenant compte de leur connaissance et des comportements des autres agents du système afin d'effectuer leurs choix. C'est une compétition dans laquelle chaque joueur est considéré comme rationnel, dans le sens où il va toujours chercher à améliorer son score ou son gain en effectuant l'action lui permettant d'atteindre ce but. Les jeux sont considérés comme « distribués » puisque la prise de décision relève de chaque joueur. Cette théorie a été largement étudiée dans différents domaines, tels que la biologie, la sociologie, l'économie ou l'informatique, particulièrement dans le cadre de problèmes d'optimisation multi-objectifs. Nous avons été les premiers à proposer l'application de ce formalisme au domaine des MPSOC. L'objectif de ce travail est illustré sur la **Figure 29**.

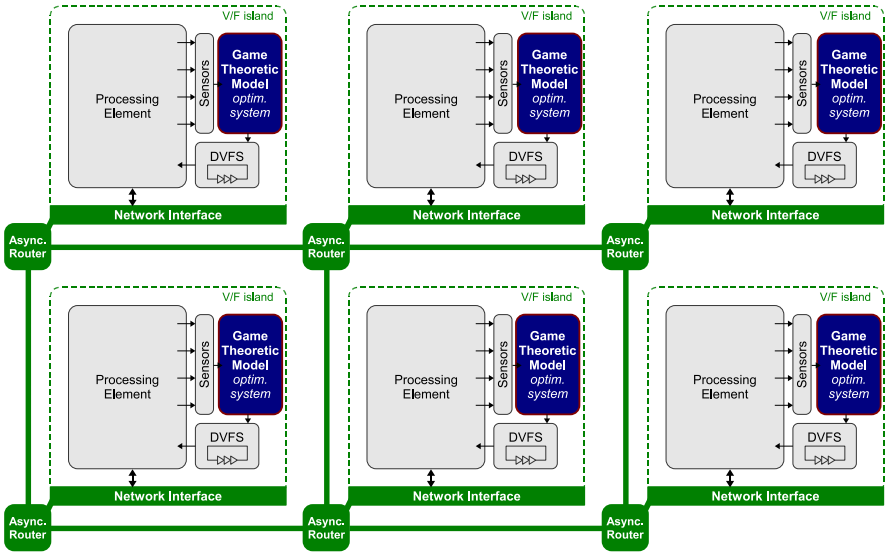
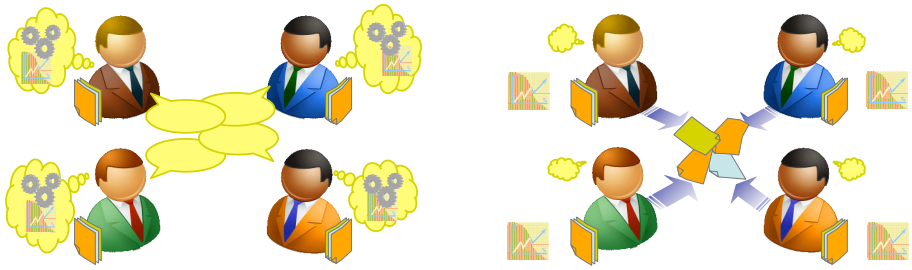



**Figure 29.** La théorie des jeux appliquée à l'optimisation en ligne d'un MPSOC

### 2.2.1 Un modèle de jeu non-coopératif et simultané

Dans [**28**], nous définissons les bases d'un jeu stratégique non-coopératif Γ, composé d'un ensemble N de *n* joueurs, d'un ensemble S d'actions possibles par joueur, et les résultats $u_i$, $\forall i \in N$. Dans un tel jeu, les *n* joueurs interagissent et jouent à travers l'ensemble des actions possibles de manière non-coopérative, dans le but de maximiser $u_i$. On considère par exemple le jeu non coopératif de la **Figure 30**, composée de $n = 4$ *joueurs*. Dans un premier temps, chaque joueur analyse la situation de jeu, c'est à dire qu'il récupère toutes les informations possibles et fait une estimation des gains personnels potentiels en fonction des différentes stratégies possibles. Le joueur choisit l'option lui permettant de maximiser ses gains. Ensuite, les agents réalisent simultanément leur action de jeu et recalculent le gain consécutif à leur choix. Compte-tenu des différentes interactions, il peut bien sûr y avoir des

différences entre le résultat attendu et celui obtenu. Mais en prenant connaissance des choix des autres joueurs, chaque joueur peut analyser une nouvelle fois la situation, estimer les différentes possibilités de jeu, rejouer, ainsi de suite de façon répétitive. Après un certain nombre de cycles, les joueurs atteignent une situation d'équilibre si cette solution existe : on parle alors d'équilibre de Nash. Cela signifie que dans cette situation, quelle que soit l'action possible, aucun des joueurs ne peut plus améliorer son gain.



(a) Les joueurs analysent le scénario afin de choisir la meilleure action (b) les joueurs font leur choix et jouent

**Figure 30.** Illustration d'un jeu simultané non-coopératif

### 2.2.2 Application du modèle aux MPSOC

Considérons maintenant que le but du jeu est de déterminer les couples tension/fréquence de chaque élément du système tel que cela est représenté sur la **Figure 31**. Il s'agit d'un modèle MPSOC similaire à *OpenScale* avec 4 NPU. Le but de la sélection d'une fréquence (comme dans le cas du PID, on considère qu'à chaque fréquence ou chaque intervalle de fréquence correspond une valeur ou une gamme de tensions) est d'optimiser une fonction d'utilité, composée par exemple de la puissance consommée et de la performance du système. Compte-tenu des interactions applicatives et physiques, ces métriques ne dépendent pas que de la configuration locale, mais bien de l'ensemble du système. Dans ce scénario, chaque NPU incarne un joueur du jeu, dans ce cas il y a 4 joueurs. L'ensemble des actions possibles correspond au nombre de fréquences assignables localement par un mécanisme de DVFS. L'interaction entre joueurs, *i.e.* le medium de communication, est incarné par le NOC, support des communications physiques entre NPU. Un module dédié (logiciel, matériel ou mixte) permet d'estimer la fonction d'utilité $u_i$, et enfin chaque NPU assigne la fréquence qui maximise la fonction d'utilité.

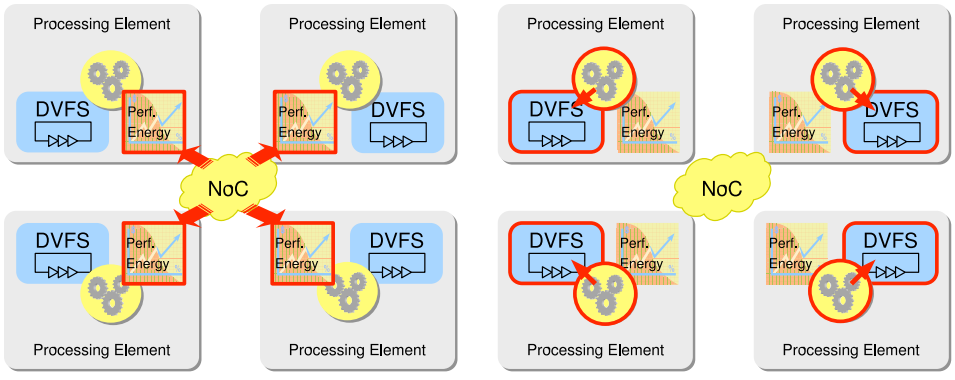



**Figure 31.** Jeu simultané non-coopératif appliqué à un MPSOC dans le but d'optimiser performance / énergie

Le modèle de MPSOC que nous avons développé dispose de ressources entièrement distribuées, qu'il s'agisse du calcul, de la mémoire, des moyens de communication, du monitoring ou des actionneurs. Il se prête donc parfaitement à ce type de formalisme inspiré de la théorie des jeux. Cette hypothèse de répartition des actionneurs à l'échelle d'éléments de calculs est largement vérifiée aujourd'hui car de

nombreux systèmes intégrés offrent cette possibilité de contrôle, à grain fin, d'actionneurs type DVFS. Le caractère non-coopératif peut se discuter : intuitivement, on pourrait penser qu'une solution collaborative serait plus efficace pour atteindre une situation d'équilibre. Or c'est là que cela devient réellement intéressant : le caractère non-coopératif est très simple à mettre en œuvre (pas de compromis entre joueurs) c'est la règle du jeu, à travers la fonction d'utilité, qui garantit par construction l'accessibilité de cet équilibre de Nash. Par rapport à une approche centralisée, cela réduit aussi l'espace d'observation requis pour prendre des décisions. L'information globale est bien présente, mais elle reste distribuée aux points stratégiques du système, ce qui évite le déplacement inutile de données de monitoring, ainsi que la diffusion des paramètres des actionneurs. Enfin, nous nous intéressons ici à des systèmes globalement asynchrones (les NPU sont connectés à travers un NOC de type GALS), nous ne définissons pas d'ordre de jeu pour les joueurs : on s'appuie sur un modèle qui considère des actions simultanées, où la référence commune entre agents est le cycle de jeu. Un exemple de la séquence d'ajustement des fréquences est donné dans la **Figure 32**.

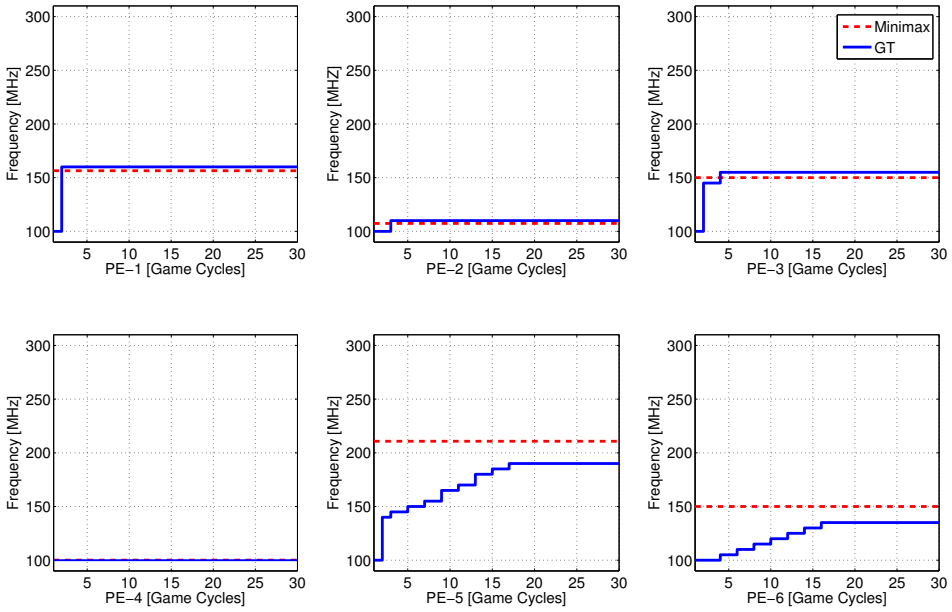



**Figure 32.** Exemple de déroulement d'une séquence de jeu pour 6 processeurs d'un MPSOC

### 2.2.3 Résultats

Un premier résultat auquel on va s'intéresser est la *scalabilité*. Il faut avoir à l'esprit que chaque composante du système est identique en termes de ressources matérielles et logicielles, afin de faciliter notamment le changement de facteur d'échelle d'une génération technologique à une autre. Il est donc nécessaire de s'intéresser à l'évolution des performances du modèle inspiré de la théorie des jeux en fonction du nombre de joueurs, ici donc les processeurs. Pour cela, nous avons réalisé une étude statistique sous Matlab couvrant quelques dizaines de milliers de scénarios applicatifs sur des réseaux allant jusqu'à 100 processeurs, pour lesquels nous avons relevé la vitesse de convergence, c'est à dire le nombre de cycles de jeux nécessaires pour atteindre l'équilibre de Nash. La **Figure 33** montre les résultats obtenus : on constate que pour une centaine de processeurs, dans un scénario pessimiste, 40 cycles de jeux seront nécessaires pour que l'algorithme distribué converge.

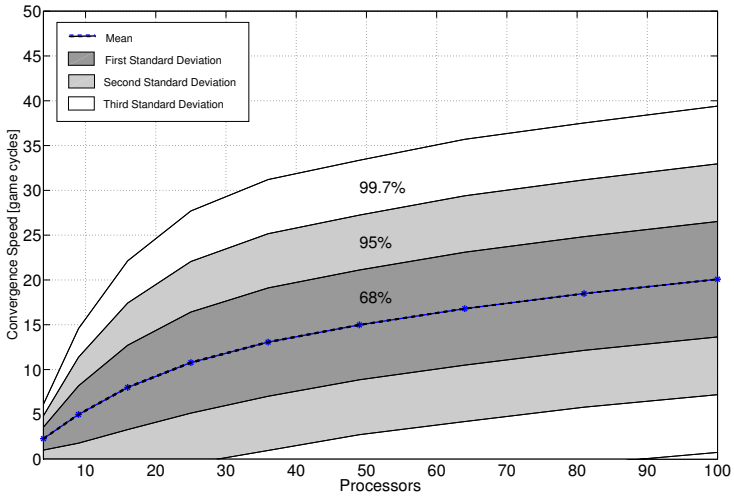



**Figure 33.** Etude de la vitesse de convergence en fonction du nombre de processeurs

Une deuxième étude consiste à comparer la qualité de la solution obtenue à l'équilibre, par rapport à une solution exacte (Minimax). Le résultat obtenu sur une dizaine de milliers de scénarios est illustré **Figure 34** : cette distribution, comprise entre 58 et 98%, montre une qualité d'optimisation moyenne de 89% avec un pic à 93%. Il y a certes une dégradation mais n'oublions pas que l'équilibre est atteint pour une centaine de processeurs en moins de 40 cycles de jeu, par rapport à cette approche hors-ligne centralisée qui ne serait pas envisageable d'intégrer.

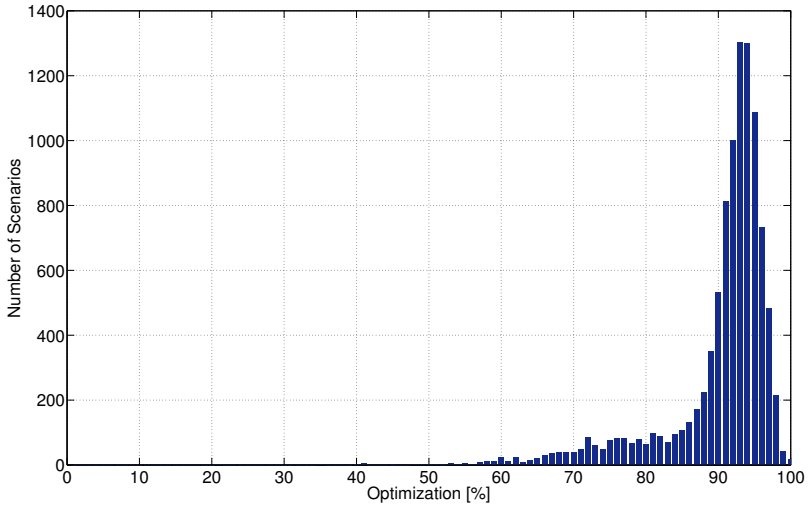



**Figure 34.** Distribution de la qualité d'optimisation

Nous avons eu l'opportunité de travailler sur plusieurs métriques (telles que le synchronisme entre tâches, la température, la consommation, l'énergie, la latence) pour composer les fonctions d'utilité, tout en considérant comme actionneur la possibilité de changer de fréquence. Il faut noter ici deux choses. Premièrement, c'est la convexité de la fonction d'utilité qui va garantir l'existence d'un équilibre atteignable. Cette propriété a une grande importance du point de vue système, mais aussi localement, car cela indique qu'il est possible par un algorithme glouton reposant sur un calcul de gradient d'évaluer les gains de chaque joueur. Le passage d'un espace continu à un espace discret, qu'il s'agisse du calcul de la fonction d'utilité ou des fréquences, peut introduire des oscillations autour de l'état d'équilibre, problème pouvant être résolu par un tirage aléatoire de fréquence lorsque cette situation est identifiée, ou encore par une construction de la fonction d'utilité qui assure la propriété de convexité.

Un autre exemple illustre un cas concret d'utilisation de ce modèle sur l'application de démodulation 3GPP-LTE déjà évoquée dans le travail de Camille Jalier. La **Figure 35** montre un scénario au cours

duquel le système change plusieurs fois de modes de réception, sous contrainte de latence fixe. La charge de calcul variant suivant le mode, il est possible d'adapter la fréquence des processeurs de façon à réduire l'énergie consommée. Cette séquence compare le résultat obtenu par application de la théorie des jeux, et celui obtenu dans une approche « pire cas » à fréquence fixe (nominale). Dans ce cas précis, c'est une réduction qui atteint 25% dans le mode basse-qualité et nécessite peu de cycles pour atteindre ce résultat. Ce qui est remarquable, c'est que l'ajustement de fréquences s'effectue de manière dynamique et distribuée.

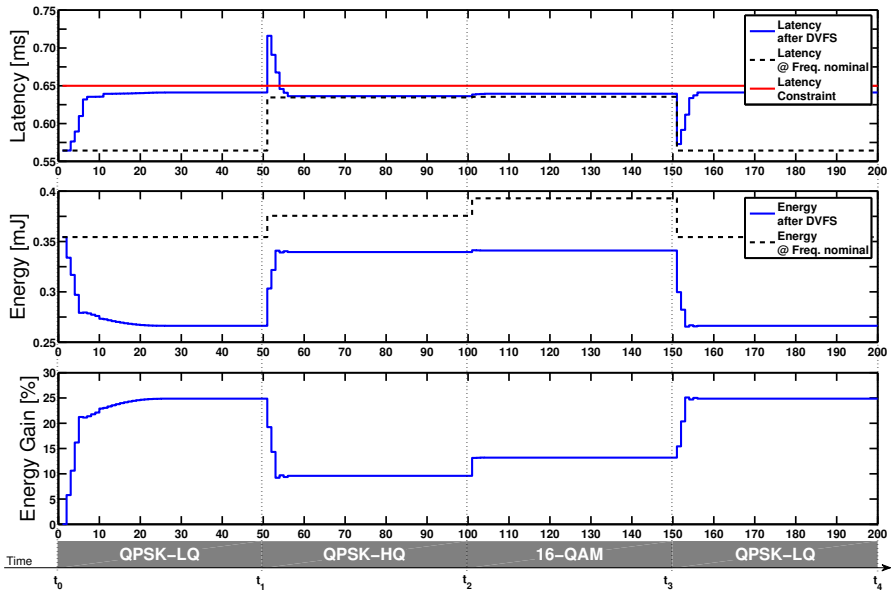



**Figure 35.** Scénario applicatif de la théorie des jeux, et comparaison des latences, énergie et gain énergétique par rapport à une approche purement nominale

Ce gain évalué dans un démonstrateur complet implanté dans GENEPY (techno ST 65nm Low Power) atteint jusqu'à 50% **[29]**, ce qui prouve le réel intérêt de cette approche. Il faut noter là encore, qu'une résolution exacte du problème hors-ligne, a permis de montrer que sur l'ensemble des configurations possibles, la méthode inspirée de la théorie des jeux s'approche de la solution optimale (moins de 5% d'énergie consommée en plus). Deux implémentations du contrôleur inspiré de la théorie des jeux ont été réalisées. En ajoutant le nombre de cycles nécessaire pour calculer la fonction d'utilité pour trois valeurs de fréquence, nous pouvons estimer le temps de réactivité du contrôle pour chaque schéma. Dans les deux cas, la durée d'un cycle de jeu reste raisonnable ne dépassant pas les 2420 cycles d'horloge pour le modèle software et 752 cycles pour le module matériel. Par ailleurs, nous avons enregistré une perte moyenne de 0.57mW par unité lors de l'exécution d'un cycle de jeu par HW-optimisé, cette consommation remonte à 3mW si le contrôle est assuré par le MIPS. Cela conduit à privilégier l'implémentation matérielle, mais pour des raisons de flexibilité la version logicielle reste une solution séduisante surtout si l'on dispose déjà d'un contrôleur programmable dans l'architecture ce qui évite d'encombrer davantage les unités de traitement.

## 2.3 Optimisation inspirée de la théorie du consensus

Les résultats issus de l'approche de contrôle inspiré de la théorie des jeux nous ont encouragé à poursuivre l'étude de solutions distribuées permettant une optimisation en ligne de systèmes MPSOC. Dans la continuité de ces travaux, l'implémentation du contrôleur en question dans un contexte d'évaluation concret a permis de mettre en lumière quelques limitations. Notre modèle s'appuie en effet sur une hypothèse simplificatrice qui minimise l'impact des latences de communication. Or la séquence de jeu nécessite de rassembler périodiquement des informations globales, ce qui constitue une charge sur le réseau et implique une contrainte de latence supplémentaire. Celle-ci dépend directement du nombre d'unités : pour une architecture maillée de dimension $n \times n$, la latence est proportionnelle à $n^2$. Fort de ce constat, nous nous sommes tournés vers d'autres classes d'algorithmes qui permettent de compenser cette surcharge de communication par le biais d'échanges privilégiés entre processeurs voisins. Contrairement à une des hypothèses du modèle inspiré de la théorie des jeux, cette nouvelle approche utilise la coopération pour optimiser le fonctionnement du système dans son intégralité sans pour autant en avoir une connaissance globale. Ce processus d'optimisation est régi par la « théorie du consensus » : un formalisme d'échanges complètement distribué entre un ensemble d'agents. Par le biais de cette nouvelle approche, les contrôleurs proposés émulent un comportement coopératif visant l'établissement d'une configuration optimale consensuelle du système.

### 2.3.1 Méthode du consensus

Le consensus est un protocole qui vise à mettre un ensemble d'agents en accord sur une valeur commune. Plusieurs classes de protocoles ont été dérivées de cette théorie selon la nature de la valeur mise en question. Nous avons cherché le formalisme du consensus le plus adapté à la problématique de cette thèse : un mécanisme de contrôle optimal, stable et distribué.

La fonction *objectif* globale est répartie entre les différents agents du système. Chacun d'entre eux essaie d'optimiser une fonction locale, et coopère avec ses voisins immédiats en leur transmettant le résultat de sa progression. Le modèle de consensus spécifie les règles qui gouvernent ces échanges d'informations par le biais d'un graphe d'échange G, et comment les informations reçues du voisinage sont traitées par rapport à celle estimée en local. Ce sont ces règles qui garantissent la convergence du processus vers un optimum de la fonction objectif globale.

Le graphe d'échange doit garantir l'existence d'un chemin direct ou indirect reliant chaque paire de sommets. Construit de la sorte, chaque agent a la capacité d'influencer l'ensemble des éléments ; si ce n'est pas immédiatement au voisinage direct, c'est dans la durée à travers une séquence d'échanges. Un lien reliant deux voisins est bilatéral, ce qui reflète bien le caractère coopératif de l'approche, où le système cherche un compromis entre tous ses membres. Tel que cela est formalisé dans **[30]** , il est nécessaire de mettre en place un protocole d'échange pour aboutir à un équilibre consensuel. Les échanges doivent être déclenchés de manière synchrone (par le biais d'un transfert de jetons par exemple) et les données échangées doivent être pondérées par une « matrice d'adjacence » construite à partir du graphe G. L'utilisation de la matrice *Metropolis-Hasting* garantit une convergence vers un consensus de manière asymptotique **[31]**.

Chaque entité est chargée localement de calculer la valeur qui optimise sa fonction locale par la méthode du sous-gradient. Le consensus est utilisé pour ajuster ces valeurs en fonction du gain total du système, tout en conservant les contraintes du problème. Les agents instaurent eux-mêmes un accord sur la valeur de consensus de façon itérative, sous certaines hypothèses liées notamment à la convexité des fonctions locales. Si l'ensemble de ces hypothèses est vérifié, il est démontré dans **[30]** que ce point de convergence correspond à l'optimum de la fonction globale.

### 2.3.2 Application du modèle coopératif aux MPSOC

Le modèle de MPSOC considéré ici est similaire à celui utilisé dans les travaux de Camille Jalier et Diego Puschini, à savoir une application de télécommunication 4G pour la réception de trames en 3GPP-LTE suivant plusieurs modes. L'application est de type flot de données temps réel: composée de plusieurs tâches nécessitant chacune de grandes capacités de calcul, elle fixe une échéance de

terminaison stricte sur l'exécution de la chaine des tâches (représentée par T0, qui fixe également la nature périodique de la chaîne de traitement). Les tâches composant l'application sont réparties sur les différents opérateurs de l'architecture MPSOC. Comme précédemment, on suppose que chaque tâche est exécutée par une unité de traitement.

Le modèle de dissipation explore la relation entre l'énergie dissipée par une unité en fonction de sa charge de travail sous forme d'une expression analytique faisant intervenir sa fréquence d'horloge locale. L'ajustement des performances d'une unité sera déduit par référence à ce modèle. Etant donné que le contrôle s'applique à un niveau système, le modèle doit être capable d'estimer la consommation sur une période de temps correspondant à celle de ce contrôle. On note que durant la période T0, on considère une période d'activité sur une certaine durée (NxT) correspondant à une dissipation de puissance dynamique maximale, et une durée au cours de laquelle l'élément de calcul est dans un mode *low-power* (correspondant donc à une énergie dynamique minimale). Le principe de cette hypothèse est représenté sur la **Figure 36**.

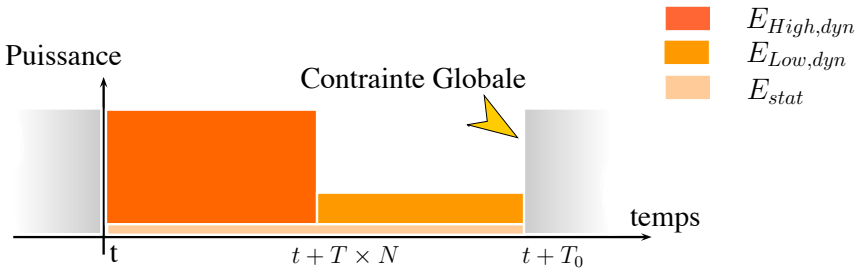



**Figure 36.** Représentation des différents niveaux de dissipation d'énergie au regard de la contrainte de la latence à l'échelle d'un élément de calcul

L'application du consensus est une innovation apportée du travail de thèse d'Imen Mansouri. C'est une technique qui répond bien à la problématique du contrôle dynamique dans les architectures multi-cœurs. Tout d'abord, il s'agit d'un mécanisme qui opère de manière complètement distribuée en faisant converger le système vers un optimum global. La convergence est parmi les facteurs qui favorisent la stabilité du contrôle et son efficacité, ce qui motive davantage le choix de cette technique pour assurer ce contrôle. En terme de faisabilité, les architectures multi-cœurs articulées autour d'un réseau sur puce présentent un support favorable à l'application du précédent formalisme. Sur un niveau structurel, chaque unité peut gérer ses performances en interne indépendamment des autres. Elle dispose entre autres de ressources de calcul suffisantes lui permettant de participer au processus d'optimisation. Au niveau des communications, le réseau offre un moyen d'échange fiable avec des délais de transmission bornés. Ces deux propriétés, qui paraissent assez triviales à première vue, sont nécessaires lors de l'application d'un protocole de consensus synchrone tel que présenté précédemment.

La mise en œuvre de la méthode (**Figure 37**) nécessite de fixer tout d'abord les fréquences de chaque unité de manière à assurer la contrainte de latence imposée par le graphe de tâches. Cette étape d'initialisation ne suppose aucun effort d'optimisation, son but étant simplement de définir des conditions initiales satisfaisant les contraintes applicatives. Une fois le processus d'optimisation déclenché, l'énergie dissipée pour exécuter les tâches durant la période T0 diminue d'une itération à une autre jusqu'à atteindre le minimum.

Deux cas de figures sont envisagés pour la fonction locale : il s'agit dans le premier cas d'étude de l'énergie dynamique de l'unité de calcul, et dans le deuxième de la température de ce bloc. Compte tenu de l'expression de la contrainte, on considère comme variable locale le terme Ni × Ti correspondant au temps de calcul de la tâche pris en charge par cette unité. Les différents éléments collaborent pour trouver la configuration optimale de leur fréquence locale de manière à optimiser la consommation ou la température globale sous contrainte de performance. Lorsqu'une nouvelle application est mappée, un élément initiateur (le premier sur la chaîne de traitement) débute ce processus d'adaptation et sollicite ses voisins pour chercher une nouvelle configuration du système qui soit plus adéquate à l'application en cours. Ce processus est déclenché ensuite au sein de chaque unité du système par effet de propagation. On retrouve ici l'aspect distribué de l'approche.

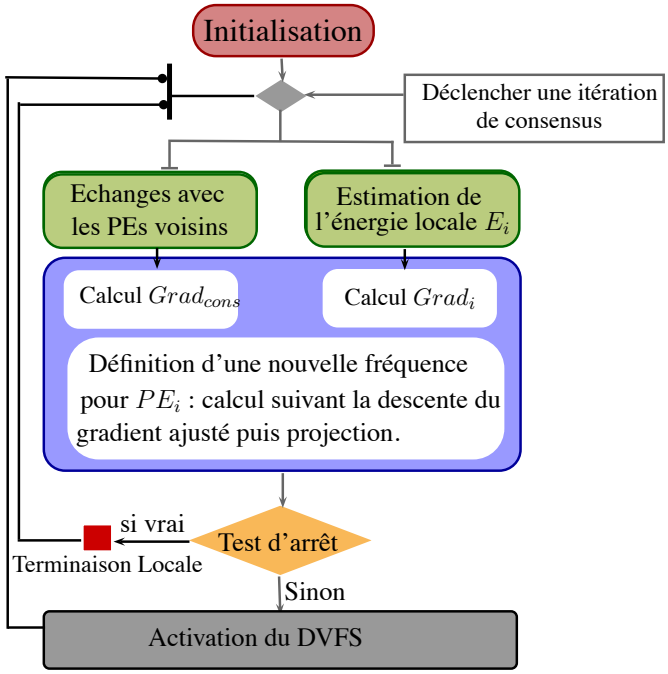



**Figure 37.** Les différentes étapes intervenant dans la méthode du consensus appliquée aux MPSOC

Le schéma de communication selon lequel ces échanges s'effectuent est mappé dynamiquement en fonction de plusieurs critères. Evidemment, il est formé par les unités actives qui ont été sollicitées pour le traitement de l'application en cours. Chaque PE décide du voisinage avec lequel il peut communiquer et calcule par la suite les coefficients d'échange correspondants selon *Metropolis-Hasting*. Le choix du voisinage est arbitraire, il peut être conduit à partir des règles de proximités physiques entre les unités ou à partir du graphe des tâches pour exploiter les flux de communications déjà existant.

### 2.3.3 Résultats

Nous avons validé sur un certain nombre de cas d'études l'utilisation du modèle de consensus appliqué au contrôle dynamique des fréquences d'un système MPSOC **[11]**. Dans l'expérience choisie ici, on reprend l'exemple de la chaîne de réception pour lequel le débit de transmission des données varie pour un mode de réception donné. On fait varier l'amplitude de la contrainte temporelle après chaque centaine d'itérations de l'algorithme. Cette contrainte est fixée de manière arbitraire entre 700us et 900us. L'aspect adaptatif est clairement démontré dans la **Figure 38**. A chaque nouvelle contrainte, le système s'initialise de manière à garantir la latence globale, puis commence à chercher la configuration optimale adéquate. On relâche par exemple la contrainte de 700us à 900us entre les itérations 200 et 300. En augmentant le temps de décodage d'un TTI, la charge de travail du circuit est diminuée, et donc le système réduit sa vitesse pour réaliser une économie d'énergie de 46% par rapport à une configuration nominale. Durant cette expérience, nous avons introduit le test d'arrêt sur les valeurs de fréquence calculées durant chaque cycle qui se base sur la résolution du DVFS. Nous observons clairement que le système atteint un équilibre très proche de l'optimum absolu de la fonction d'énergie calculée par MATLAB et qui ne tient pas compte de cet aspect de discrétisation. Les gains varient selon l'amplitude de la contrainte : nous pouvons observer de nettes réductions dès les premières itérations, ceci montre un comportement de convergence asymptotique lié à l'utilisation du consensus avec le protocole *Metropolis-Hasting*.

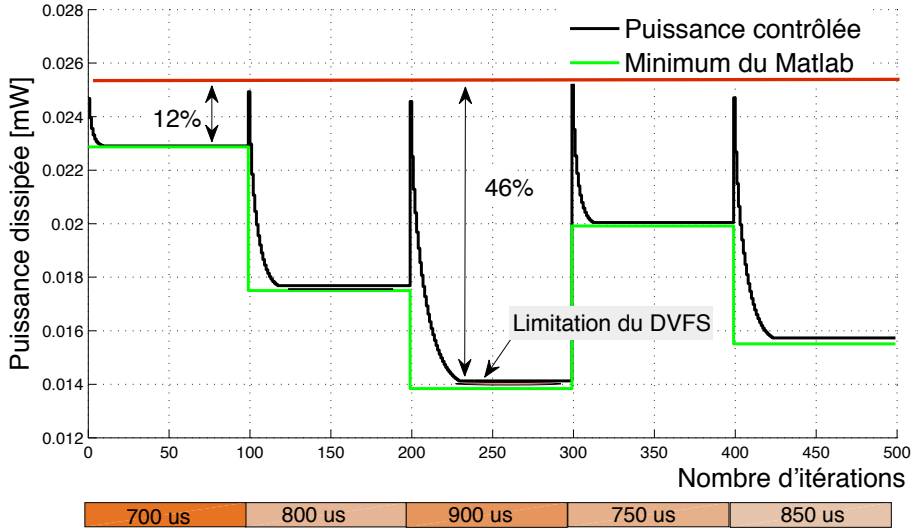
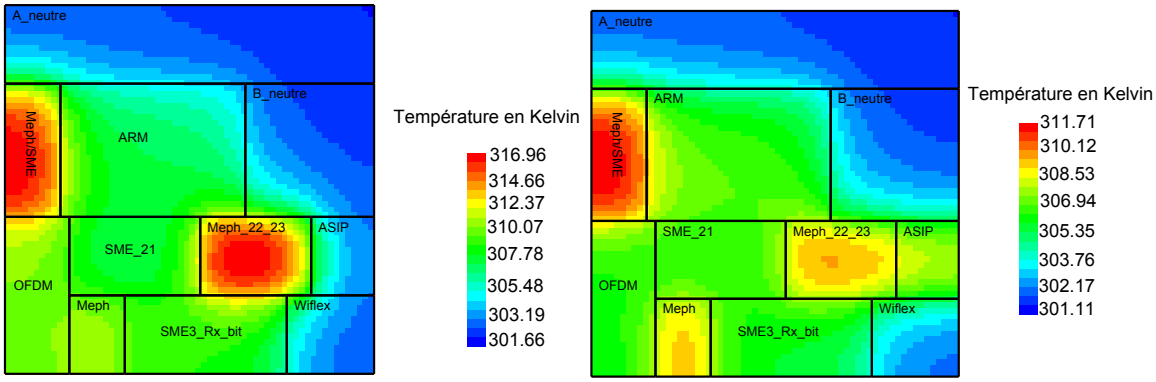



**Figure 38.** Exemple d'adaptation dynamique du système grâce au Consensus

Un deuxième résultat remarquable concerne l'utilisation du Consensus pour l'optimisation de la température sous contrainte de performance. C'est ce qu'illustre les figures suivantes obtenues par simulation avec l'outil Hotspot. On observe tout d'abord à fréquence nominale, avant optimisation une amplitude thermique de l'ordre de 15K, avec 2 points chauds (**Figure 39 (a)**). Après application du consensus, un point chaud a été fortement réduit, et l'amplitude n'est plus que de 10K avec une température maximale inférieure (**Figure 39 (b)**). Lorsqu'on compare le résultat de cette optimisation thermique à celle ciblant l'efficacité énergétique, il apparaît clairement des différences entre les 2 cartographies : en particulier, un point chaud se crée au niveau de l'ASIP avec une amplitude supérieure à celle initialement observée de l'ordre de 16K (**Figure 40**).




(a) (b)

**Figure 39.** Carthographie thermique initiale (a) et après optimisation thermique par consensus (b)

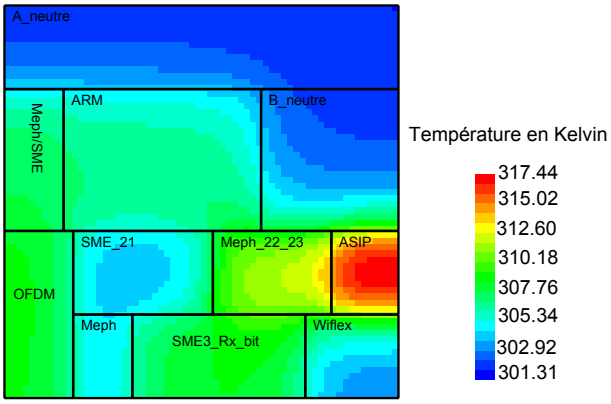



**Figure 40.** Carthographie thermique après optimisation énergétique par consensus

Une dernière expérience concerne l'étude de la *scalabilité* de la méthode en fonction du nombre d'éléments de calculs, comparée à une méthode centralisée et au modèle inspiré de la théorie des jeux. La **Figure 41** nous révèle clairement que l'approche du consensus permet une vitesse de convergence plus rapide que les autres méthodes.

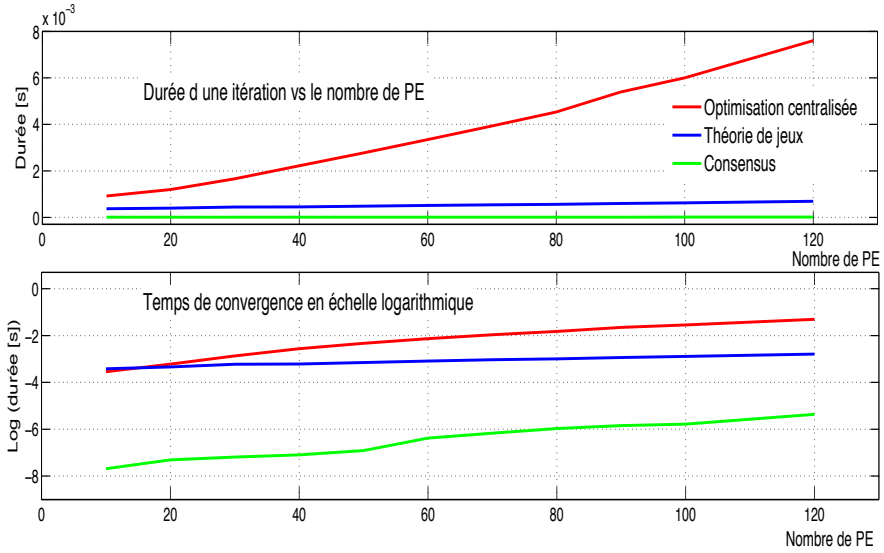



**Figure 41.** Etude de la *scalabilité* des différentes méthodes d'optimisation

L'implémentation logicielle du contrôleur « consensus » au sein du processeur MIPS de GENEPY, nous a permis également d'analyser les performances de l'algorithme d'optimisation d'énergie, qui nécessite 183 cycles pour une itération de consensus qui comprend l'estimation de la valeur de gradient et l'attribution d'une nouvelle consigne d'horloge. Une séquence d'échange de gradients entre 2 unités nécessite quant à elle 472 cycles. Cette durée tient compte de l'encapsulation du gradient dans un paquet et sa transmission, sa réception au niveau du MIPS voisin, l'exécution d'une itération de consensus au niveau de ce dernier, et l'acheminement du nouveau gradient vers le MIPS qui a initié ces échanges. Ces résultats confirment la simplicité de notre approche, et rendent son application dans un contexte concret très envisageable.

## 2.4 Optimisation de la consommation en technologie FD-SOI

L'ensemble des méthodes de contrôle et d'optimisation que nous avons étudiées jusque là considèrent comme variable d'ajustement la fréquence locale des éléments de calcul dans un contexte où l'on suppose des ilots de tension/fréquence. Cela signifie qu'à chaque fréquence ou plage de fréquence, est associée une tension d'alimentation, permettant ainsi de jouer sur le compromis performance / énergie. Dans tous les cas, nous n'avons pas réellement pris en compte le coût et la résolution de ces actionneurs. Nous avons d'ailleurs fait l'hypothèse d'un pas de fréquence de l'ordre de la dizaine de MHz dans les exemples et simulations que nous avons effectués. Il était donc nécessaire de revenir à l'échelle du nœud de calcul (ou VFI) pour prendre en compte ces critères dans la mise en œuvre d'un contrôle adaptatif cohérent de bout en bout, de l'échelle système à l'échelle locale.

Ce travail coïncide avec l'émergence de nouvelles technologies, et d'une nouvelle variable d'ajustement à considérer dans la gestion du compromis performance / énergie. En raison notamment d'une consommation énergétique devenant trop importante au delà du nœud CMOS BULK de 28nm, l'industrie des semi-conducteurs s'est tournée vers de nouvelles approches, notamment celles basées sur des substrats FD-SOI. Outre les paramètres usuels que sont la tension d'alimentation et la fréquence d'horloge, il devenait alors possible de contrôler la tension de substrat (*body bias*). C'est dans ce contexte que s'est déroulée la thèse de Yeter Akgul, qui a étudié les nouvelles possibilités offertes et exploré des solutions innovantes de gestion dynamique des paramètres de tensions/fréquence afin d'optimiser la consommation énergétique des systèmes sur puce (**Figure 42**).

L'ensemble des paramètres de tension/fréquence permet théoriquement une multitude de points de fonctionnement, qui doivent satisfaire des contraintes de fonctionnalité, de performance, et évidemment de coût : c'est pour cette raison d'ailleurs que l'on doit limiter leur nombre à l'échelle du PE. Ce travail s'intéresse donc dans un premier temps à une problématique de conception, en proposant une méthode d'optimisation du placement de ces quelques modes opérationnels. Une solution analytique permettant de maximiser le gain en consommation apporté par l'utilisation de plusieurs points de fonctionnement est proposée. La deuxième contribution importante concerne la gestion dynamique des paramètres de tension/fréquence, permettant d'améliorer l'efficacité énergétique en s'appuyant sur une méthode d'optimisation convexe. La validation expérimentale des méthodes proposées s'appuie sur des échantillons de circuits réels, et montre des gains significatifs comme nous le verrons dans la suite de cette partie.

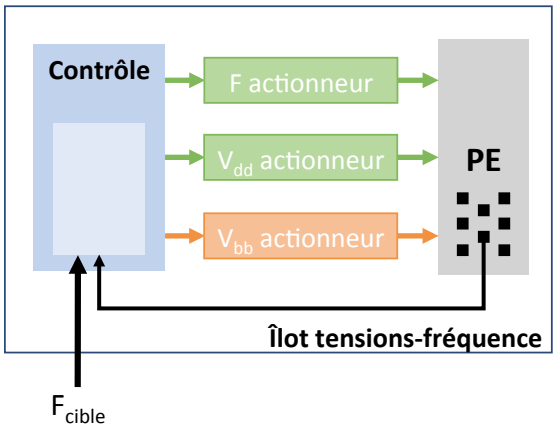



**Figure 42.** Contrôle local du DVFS en technologie FDSOI

### 2.4.1 Actionneurs DVFS

Le principe d'un DVFS est de modifier en cours de fonctionnement, dynamiquement, la fréquence et la tension d'alimentation du circuit. Une action sur la fréquence va permettre d'ajuster les performances temporelles du circuit en fonction des besoins applicatifs, et permet de réduire la puissance dynamique. Pour gérer efficacement l'énergie consommée, il faut également agir sur la tension d'alimentation, dont dépend également la consommation statique. Cette double action autorisée par la technique de DVFS a un impact très important sur la consommation, raison pour

laquelle la plupart des circuits en disposent. La gestion du DVFS passe souvent par un composant externe nommé PMIC (Power Management Integrated Circuit) qui interagit avec le circuit sous contrôle par le biais de commandes qui peuvent être initiées par des services dédiés d'un micro noyau (ou des gouverneurs Linux). La gestion des paramètres F et V nécessite une attention particulière quant à l'application de la séquence d'ajustement tel que cela est illustré sur la **Figure 43**. Pour passer en mode *low power*, il faut d'abord diminuer la fréquence, puis la tension d'alimentation ; dans l'autre sens, il faut augmenter premièrement la tension d'alimentation puis la fréquence, ceci afin d'éviter les erreurs temporelles (zone non-fonctionnelle). On peut remarquer ici que cette zone non-fonctionnelle ($F_{max}$) dépend du chemin critique dont les caractéristiques peuvent évoluer au cours du temps (vieillissement), ou en fonction des conditions d'utilisation du composant (Température ambiante par exemple). Pour garantir un mode de fonctionnement optimal du point de vue de la performance, on peut adjoindre un ensemble de capteurs (de type CPM et PVT) qui vont permettre au circuit de définir la limite réelle de la zone non-fonctionnelle, en fonction du procédé de fabrication (variabilité), des conditions environnementales ou du vieillissement [**32**]. De la même manière, l'ajustement de la tension du substrat en fonction des caractéristiques réelles du circuit va avoir un effet à la fois sur la performance et sur la consommation (statique), puisqu'elle impacte indirectement la tension de seuil des transistors [**33**].

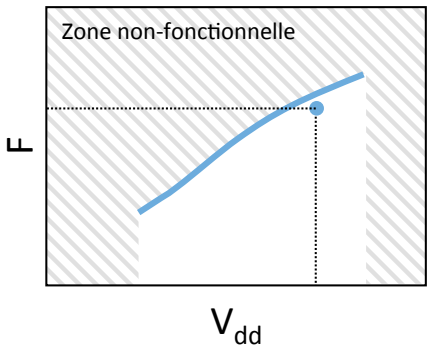
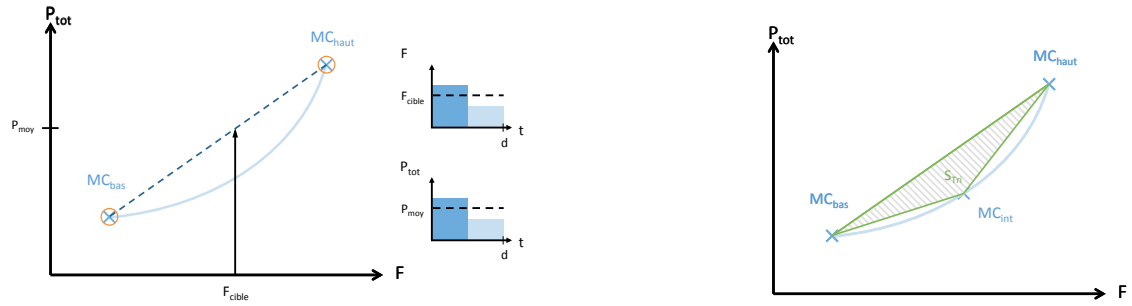



**Figure 43.** Zones fonctionnelles et non-fonctionnelles dans l'utilisation d'un DVFS

Cette étude s'intéresse à un contrôle grain fin de la consommation, non pas au niveau circuit, mais au niveau PE ou îlot de tension-fréquence. Cela exclut de fait la mise à disposition d'une plage de tensions continue qui nécessiterait des convertisseurs DC-DC, dont l'intégration serait bien trop coûteuse à l'échelle d'un cluster. C'est la raison pour laquelle la solution retenue se base sur une discrétisation des niveaux de tension : N convertisseurs DC-DC sont réglés sur leur rendement optimal pour délivrer N niveaux de tension. Chaque cluster a alors la possibilité de se connecter à l'un des N plans d'alimentation grâce à la technique de *Vdd-hopping*. Généralement, ce sont deux plans qui sont utilisés : se pose alors la question de savoir quel pourrait être l'apport d'un troisième niveau de tension et dans un contexte FD-SOI, le gain supplémentaire amené par le contrôle de la tension de *body bias* ?

### 2.4.2 Configuration optimale des modes de consommation

Nous nous sommes donc intéressés à une méthode qui permettrait de placer de façon optimale un mode de consommation intermédiaire, à partir d'un mode basse consommation ($MC_{bas}$) et un mode haute performance ($MC_{haut}$). Un mode de consommation MC correspond à une configuration du triplet $\{F,V_{dd},V_{bb}\}$, et la puissance totale consommée associée que l'on représente dans le plan $P_{tot}(F)$, tel que cela est illustré sur la **Figure 44 (a)**. Lorsque un PE reçoit une consigne correspondant à l'exécution d'une tâche à une fréquence cible donnée, située entre les 2 modes de consommation, c'est grâce à une exécution en mode *hopping* que la performance cible est atteinte, et permet d'atteindre la deadline fixée.




(a) (b)

**Figure 44.** Les différents modes de consommation représentés dans le plan $P_{tot}(F)$ (a) et positionnement de $MC_{int}$ (b)

Pour déterminer la position optimale du Mode de Consommation intermédiaire, plusieurs solutions ont été étudiées, proposées et comparées. Une approche consiste par exemple, à partir des valeurs de $P_{tot}(F)$ issues des simulations d'un circuit, de trouver la position de $MC_{int}$ de telle sorte que celle-ci maximise la surface du triangle $S_{tri}$ (**Figure 44 (b)**). Une méthode analytique peut également être mise en œuvre en approximant la courbe $P_{tot}(F)$ par un polynôme de degré 3. Les résultats obtenus grâce à des mesures issues d'un DSP[**34**] en technologie STM 28nm FDSOI montrent la réduction obtenue à partir de différentes positions intermédiaires. La **Figure 45** illustre les gains potentiels obtenus par rapport à 2 modes de consommation ($V_{dd}$ compris entre 0,7 et 1,3V ; $V_{bb}$ entre 0 et 1,5V ; F entre 0,7 et 2,5GHz) : en choisissant $F_{int}$ à 2GHz, $V_{bb}$ à 1V et $V_{dd}$ à 1,1V, on obtient un gain moyen en consommation sur toute la plage de 35%, sachant que le gain maximal est de 50% pour une fréquence cible à 2GHz. Il faut cependant préciser que l'ajustement dynamique du $V_{bb}$ a un certain coût qui pourrait s'avérer incompatible avec les fréquences de commutation en mode *hopping*. Nous avons également fait une évaluation des gains pour $V_{bb}$ fixé qui montre une réduction significative autour de 25% pour le mode optimal, avec une forte influence sur la plage des fréquences accessibles. Il est clair que cet actionneur ouvre de nombreuses pistes qu'il conviendrait d'examiner à une échelle système et ce dans différents contextes applicatifs.

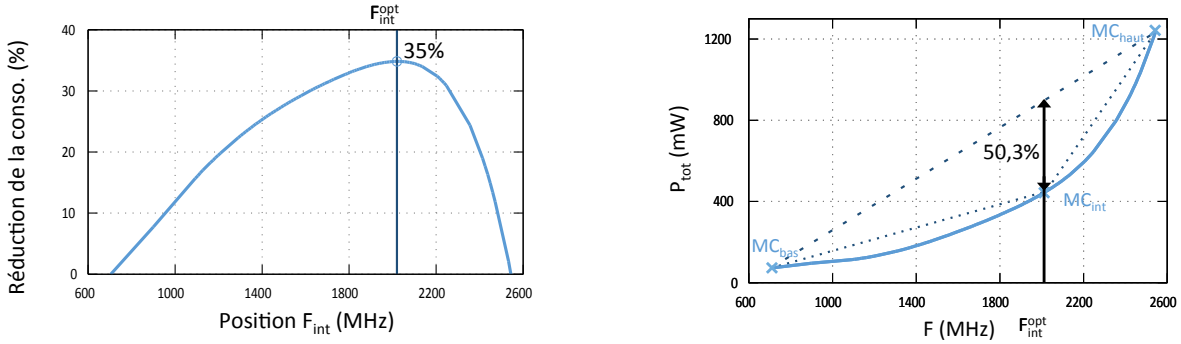


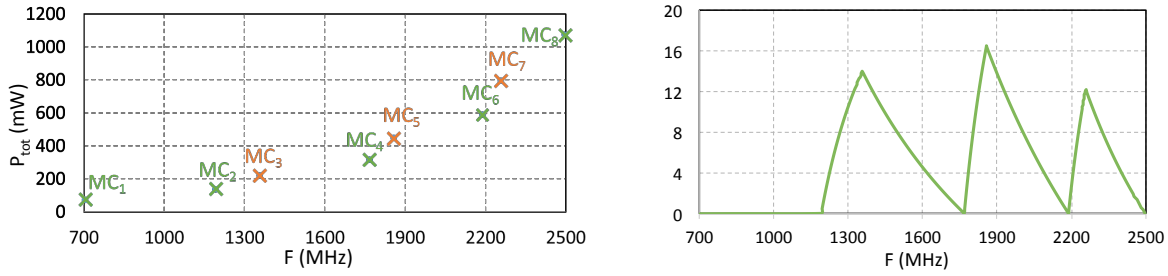

**Figure 45.** Résultats obtenus pour le positionnement du Mode de Consommation intermédiaire

### 2.4.3 Contrôle dynamique des actionneurs

On se place maintenant dans un contexte applicatif : le circuit dispose de plusieurs modes de consommation consécutifs à l'ajout de 2 niveaux de tension supplémentaires pour l'actionneur de $V_{dd}$ (4 tensions), et l'ajout de l'actionneur de $V_{bb}$. On considère dans un premier temps que ce dernier est discret (2 valeurs de $V_{bb}$), de même que la fréquence ($F_{max}$ pour $V_{dd}$ et $V_{bb}$ donnés). L'objectif est donc de déterminer la configuration $\{F,V_{dd},V_{bb}\}$ qui permet de minimiser la puissance totale sous contrainte de performance.

La **Figure 46 (a)** illustre les différents modes de consommation disponibles pour le contrôle dynamique local. La première étape dite d'identification, consiste à déterminer l'enveloppe convexe croissante de la courbe formée par ces points. Les modes de consommation n'appartenant pas à cette enveloppe sont de fait exclus des solutions potentielles, car l'application des configurations

correspondantes mène à une consommation plus élevée que l'utilisation en mode *hopping* des fréquences qui encadrent ce mode. Prenons l'exemple d'une fréquence cible qui correspondrait à l'utilisation du mode $MC_5$, l'utilisation en mode *hopping* de $MC_6$ et $MC_4$ permet de réduire de 17% la puissance totale consommée. La **Figure 46 (b)** montre les gains obtenus suivant les fréquences cibles à appliquer. Notons que ce gain ne tient pas compte de l'apport des modes intermédiaires par rapport à une approche statique nominale ou pire-cas (réduction maximale d'un facteur 10 dans l'exemple choisi).




(a) (b)

**Figure 46.** Les différents modes de consommation disponibles (a) et les gains obtenus suivant les fréquences cibles à appliquer (b)

Le dernier cas d'étude adressé ici considère que $V_{bb}$ et F sont des actionneurs continus (ou quasi-continus). Là encore vont se poser des situations où plusieurs modes de consommation sont possibles à une fréquence cible donnée, avec possibilité ou non d'exécuter la tâche en mode *hopping*. L'objectif est toujours de déterminer la configuration $\{F,V_{dd},V_{bb}\}$ qui permet de minimiser la puissance totale sous contrainte de performance. Comme précédemment, une phase d'identification est donc nécessaire dans un premier temps pour déterminer les morceaux de courbes appartenant à l'enveloppe convexe croissante dans le plan $P_{tot}(F)$, problème pour lequel une solution analytique est proposée. En phase de fonctionnement, 2 modes sont possibles suivant l'appartenance de la fréquence cible à l'enveloppe convexe (PWCS). Le mode 1 consiste à appliquer directement la fréquence avec les valeurs de $V_{dd}$ et $V_{bb}$ correspondantes (cas d'appartenance), alors que dans le mode 2 une exécution en mode *hopping* va permettre d'atteindre en moyenne la performance souhaitée tout en garantissant une consommation minimale (**Figure 47**). Une illustration de la méthode appliquée à un circuit DSP permet d'évaluer les gains sur la plage des fréquences disponibles, et montre encore une réduction maximale de 17% (**Figure 48**). Il faut remarquer ici que la courbe $P_{tot}(F)$ varie en fonction de l'activité du bloc, de la température, des paramètres technologiques et du vieillissement. Or la connaissance de cette courbe est fondamentale puisque des points limites dépendent les gains potentiels de la méthode.

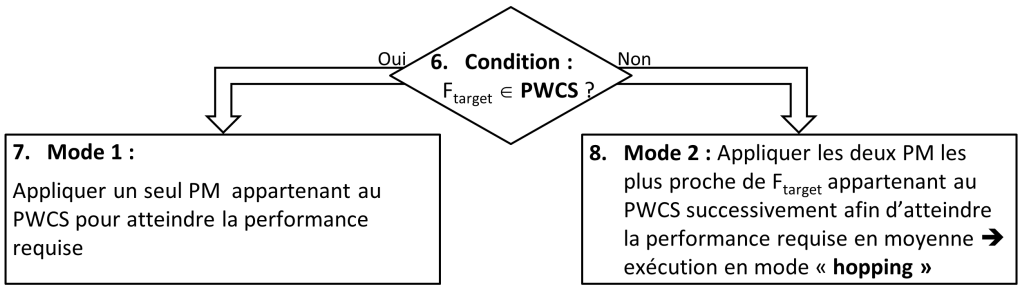



**Figure 47.** Application de la méthode en phase de fonctionnement

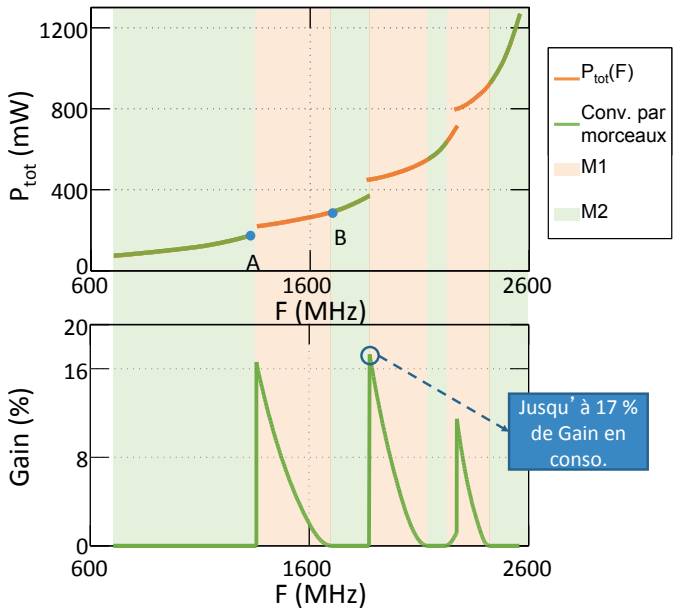



**Figure 48.** Illustration de $P_{tot}(F)$ et gains obtenus suivant les modes de fonctionnement

## 2.5 Bilan

Dans cette partie, nous avons présenté une synthèse des travaux concernant l'optimisation dynamique des architectures MPSOC. C'est tout d'abord par la mise en œuvre d'un contrôle local de type PID que nous avons montré une contribution permettant, à partir d'une consigne de performance, d'adapter un actionneur de type DVFS. Nous nous sommes ensuite intéressés à l'optimisation globale de ces systèmes distribués : en nous inspirant de formalismes issus de la théorie des jeux et du consensus, nous avons prouvé qu'il était possible d'assurer la convergence du processus vers une solution proche de celle obtenue par des algorithmes centralisés, et cela de manière entièrement distribuée et en cours de fonctionnement. Enfin, avons exploré les possibilités offertes par les technologies FDSOI, notamment en termes de contrôle de tension de Body Bias, conjointement aux actionneurs tension/fréquence disponibles dans la plupart des systèmes intégrés actuels. Nous avons développé une méthodologie d'optimisation locale permettant de garantir une consommation minimale pour une consigne de performance donnée. Le travail présenté dans cette section a été valorisé à travers plusieurs publications et brevets[11], dont :

- 5 revues, dont 3 RICL: [J4][J5][J7][J9][J10] ([J7] joint en annexe de ce document)
- 6 brevets : [B1] [B2] [B3] [B4] [B5] [B6]
- 2 chapitres d'ouvrage : [CO1] [CO5]
- 14 CICL : [C1] [C6] [C11] [C13] [C16] [C23] [C29] [C30] [C33] [C44] [C45] [C49] [C51] [C52]

[11] Cf. Partie 1, section 9, pour le détail des publications et brevets

# 3 SYSTEMES ADAPTATIFS DE CONFIANCE

Après avoir contribué à la conception d'architectures reconfigurables dynamiquement ainsi qu'à des méthodes de gestion efficaces des ressources, nous avons ciblé un niveau de parallélisme plus élevé par la mise en œuvre de structures matérielles et logicielles distribuées offrant un potentiel de *scalabilité* plus grand. Par la mise en œuvre de méthodes d'optimisation originales inspirées de l'automatique et des mathématiques appliquées, nous avons montré une maîtrise de l'efficacité énergétique allant du nœud de calcul jusqu'à l'échelle du système. Ces travaux montrent qu'un cap a été franchi dans la capacité des systèmes intégrés à s'adapter, notamment lorsqu'il s'agit de minimiser la consommation énergétique sous contrainte de latence.

L'adaptation repose sur 3 processus qui peuvent opérer à différentes échelles temporelles et spatiales. Par exemple, nous avons vu que le processus d'optimisation menant à la prise de décision est un processus multi-niveaux, réalisé à l'échelle du PE pour une consigne de performance locale d'exécution de tâche, et à l'échelle système pour satisfaire une contrainte de latence applicative. De la même manière, suivant la dynamique des contraintes, ce processus peut s'appliquer plus ou moins fréquemment aux différentes échelles. Si l'on tient compte par exemple de phénomènes plus lents tels que le **vieillissement**, une optimisation relative à sa prise en compte n'aurait de sens que sur de longues durées.

La qualité de l'adaptation dépend évidemment de ces optimisations, mais son bon fonctionnement repose fortement sur l'étape qui la précède, à savoir le **monitoring**. Celui-ci est censé produire une estimation de l'état du système, pouvant tenir compte d'un ensemble de caractéristiques telles que la puissance consommée, la température, la performance, pouvant elles-mêmes être échantillonnées à différents endroits du circuit et à différents moments. Le monitoring nécessite donc la mise en œuvre d'un ensemble de capteurs logiciels et matériels qui vont permettre d'établir le fameux diagnostique en ligne dont nous parlions dans la section précédente. Nous avons à ce moment là fait l'hypothèse que nous disposions d'une information digne de confiance sur l'état de ce système, mais produire une telle information se révèle être une tâche ardue et critique, compte-tenu de sa position dans le processus d'adaptation.

La capacité d'un système à réagir face à des situations nouvelles ou imprévues ouvre aussi des perspectives quant à la mise en œuvre de systèmes plus fiables, notamment par rapport à des problématiques de **tolérance aux fautes ou aux pannes**. Si l'on tient compte des caractéristiques de notre modèle de MPSOC, un point remarquable est la redondance naturelle des ressources, puisque le système est entièrement distribué. On peut dès lors s'interroger sur la mise en œuvre de mécanismes logiciels et matériels qui seraient capables de tirer avantageusement parti de ces propriétés.

Enfin, on ne peut ignorer aujourd'hui le problème de la **sécurité**, l'un des enjeux majeurs de la société numérique, en témoignent les actualités récentes sur la multiplication des menaces portées par des individus, des organisations ou même des états. La *cybersécurité* nécessite des méthodes et des moyens pour garantir la confidentialité, l'authenticité, l'intégrité et la disponibilité à la fois des données et des systèmes d'informations. Les circuits et systèmes intégrés sont le support matériel du numérique et constituent à ce titre un maillon tout aussi sensible que le logiciel. D'autant plus que ces dernières années, les attaques par « canaux cachés » se sont vulgarisées. Connues comme étant les attaques les plus dangereuses, elles permettent de découvrir assez facilement les clés de chiffrement des algorithmes de sécurité.

Cette dernière section adressant le chapitre de synthèse de mes travaux de recherche traitera l'ensemble de ces sujets qui ont un objectif commun : la **confiance numérique**. Confiance dans la capacité d'un système MPSOC à déployer des stratégies de tolérance aux pannes permettant une disponibilité des services, même en mode dégradé. Confiance dans la capacité du système à s'adapter par la mise en œuvre de méthodes de monitoring efficaces, délivrant une information précise à moindre coût. Confiance dans la capacité du système de préserver la confidentialité des données sensibles par le déploiement de contremesures adaptées au contexte de l'embarqué.

## 3.1 Architecture MPSOC tolérante aux pannes

Ce travail, en partenariat avec ST Microelectronics, s'intéresse à la problématique de fiabilisation des systèmes intégrés sur Silicium. Dans un contexte de complexité croissante de l'intégration sur silicium (quelques milliards de transistors), l'augmentation de la densité impacte la fiabilité des circuits dans la mesure où la moindre variation de paramètre, la moindre défaillance peut compromettre le bon fonctionnement du système tout entier. Qu'il s'agisse d'améliorer les rendements de production ou la durée de vie des composants, il est nécessaire de mettre en place des stratégies à différents niveaux. Malgré un état de l'art conséquent sur les outils et les méthodes de durcissement, on constate finalement que peu d'approches prennent en compte la tolérance aux fautes au niveau système. Avec l'émergence des architectures MPSOC, nous avons vu qu'il était possible d'adapter la plateforme, par ajustement de paramètres, par reconfiguration de l'application, ou bien encore par reprogrammation, ce qui permet potentiellement de maintenir une qualité de service et donc une certaine fiabilité, même en mode dégradé.

La thèse de Nicolas Hébert porte plus particulièrement sur les architectures régulières, basées sur des nœuds de calculs distribués identiques, telles que *OpenScale*. La redondance intrinsèque à ce type de structure permet d'envisager de nouvelles solutions de protection pour garantir le bon fonctionnement des circuits non plus uniquement au niveau de quelques sous-fonctionnalités critiques mais au niveau de l'architecture système elle-même. En s'appuyant sur ces prérogatives, nous présentons une méthode de protection distribuée et dynamique appelée *D-Scale*. Celle-ci consiste à détecter en ligne les dysfonctionnements des unités de calcul, isoler, diagnostiquer puis reconfigurer le système. Un ensemble de mécanismes logiciels et matériels sont mis en œuvre afin de garantir une couverture maximale tout en limitant l'impact sur la surface et les performances du système. Une expérimentation, basée sur un outil d'injection de fautes et un modèle SystemC de l'architecture MPSOC, nous permet de mesurer, analyser et comprendre l'impact des fautes en vue de concevoir un système réellement fiable.

### 3.1.1 Mécanismes de défaillance et solutions

On considère dans cette étude aussi bien les « défauts apparents » (consécutifs à la fabrication du composant) que les « défauts latents » (consécutifs au vieillissement). Une faute peut ne pas gêner le bon fonctionnement du système (faute dormante) ou le conduire dans un état erroné, non prévu (faute activée), puis potentiellement à une défaillance, et dans certains cas à une panne de la globalité du système. Dans ce travail, nous nous appuyons sur un modèle précis au niveau cycle d'horloge dans lequel nous considérons qu'un défaut peut engendrer deux types de fautes : un collage de bit à 1 ou à 0 à la sortie des bascules ou dans les mémoires. Pour atteindre l'objectif de fiabilité d'un système, il existe différents moyens tels que la tolérance aux fautes.

Afin de maintenir une qualité de service en présence de fautes actives, la tolérance aux fautes requiert un mécanisme de détection d'erreurs et de récupération du système. De manière générale le but est d'identifier la faute qui a conduit à l'état erroné et de stopper sa propagation. La détection d'erreurs peut s'effectuer de manière concurrente (par l'utilisation de *Watchdog* par exemple) ou préemptive (BIST par exemple, lorsque le système est temporairement suspendu). La récupération du système consiste à traiter la faute et ensuite, traiter l'erreur. Le premier va permettre de diagnostiquer, isoler, reconfigurer et réinitialiser l'élément protégé. Le rôle du second sera de remettre le système dans un état connu et sûr (en éliminant l'erreur, ou en passant dans un état antérieur ou postérieur sans erreur). On peut noter que suivant les conséquences d'une erreur, le mécanisme de traitement peut ne plus être capable de corriger l'erreur sans que le fonctionnement du système s'en retrouve impacté. Dans ce cas, on parle de tolérance aux pannes.

L'étude de travaux antérieurs à la thèse de Nicolas Hébert a permis de mettre en avant des éléments clés quant à l'élaboration d'une méthode efficace de tolérance aux pannes. Une première approche théorique de [**35**] montre des perspectives intéressantes concernant le diagnostic mutuel (réutilisant la redondance des nœuds de traitement) mais révèle certaines limitations : la détection d'erreur (qui s'effectue quand le système est en maintenance) et l'isolation des éléments (qui n'est pas traitée). Pour les méthodes TIRAN **[36]** et ASoC **[37]**, la fiabilité est gérée à travers différentes couches

logicielles/matérielles (système, service et bas niveau), de manière centralisée ou distribuée, permettant de contrôler de manière dynamique la dégradation du système en cours de vie en limitant le potentiel de défaillance brutale du MPSOC. Certaines spécificités montrent toutefois des limitations, comme la centralisation de certaines informations, l'utilisation de composants fiabilisés particuliers, ou l'absence d'un support simple et peu coûteux sur lequel viennent se greffer les protections.

### 3.1.2 Architecture D-Scale

Ces travaux font suite au développement de la plateforme *HS-Scale*, et reposent donc sur des hypothèses relatives aux caractéristiques de cette architecture, à savoir que l'on dispose d'un ensemble de processeurs et de mémoires associées, interconnectés via un réseau de communication intégré NOC de type GALS. Le modèle de calcul repose sur des processus de Kahn. Les tâches des différentes applications communiquent par le biais de services associés à un micronoyau exécuté par chacun des éléments de la grille. Ce système à topologie maillée régulière offre des services permettant d'adapter le *mapping* des tâches, notamment par le biais de mécanismes de migration de code. On suppose le MPSOC ainsi défini comme un accélérateur dans un système hôte disposant donc d'un CPU, de contrôleurs mémoires et d'un ensemble périphériques. Seul le NPU(0,0) est connecté au système hôte via un bridge, et dispose d'une mémoire sûre (par exemple Flash) contenant les codes des tâches (**Figure 49**) à recharger dans les mémoires locales de chaque NPU en cas de crash du système. L'adjonction d'une mémoire sûre à chaque NPU aurait un coût bien trop important, cette solution est donc un compromis intéressant dans un contexte d'applications non-critiques.

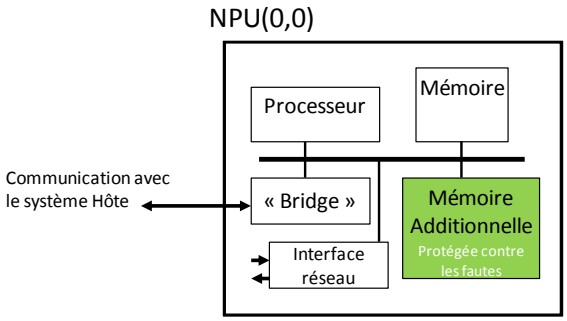



**Figure 49.** Le NPU bridge, interface avec le système hôte

La méthode *D-Scale* visant à fiabiliser le MPSOC repose sur la redondance inhérente à la structure, et sur le processus classique de détection, diagnostique, isolation et reconfiguration de l'application. L'originalité de l'approche réside notamment dans l'ordre des étapes, puisqu'au lieu de précéder la phase d'isolation, la phase de diagnostique arrive après. Ce choix est justifié pour des raisons de coût de mise en œuvre, de réactivité, et d'un point de vue fiabilité, elle permet de stopper au plus tôt la propagation d'une faute dès lors qu'elle a été détectée. Les quatre étapes sont représentées dans la **Figure 50**.

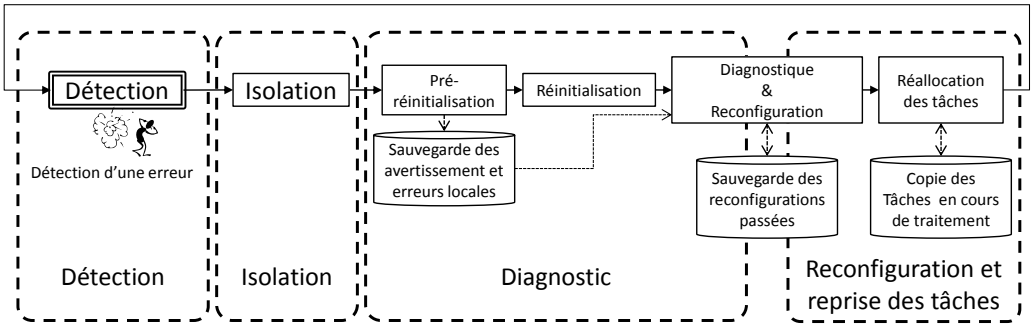



**Figure 50.** Principe D-Scale

Une unité de *watchdog* distribuée sur l'ensemble des PE assure la détection des erreurs par le biais d'un chainage permettant de tester l'unité et ses interfaces. Un PE va demander à un voisin de réinitialiser son *timer* par le biais d'un service dédié du micronoyau. Si ce dernier atteint 0, cela signifie qu'il n'a pas reçu l'ordre de réinitialisation attendu et qu'il y a potentiellement un problème. C'est à ce moment là que se déclenche la phase d'isolation, qui va s'appliquer à l'échelle du PE (à l'exclusion du NOC). L'exécution du diagnostique comprend tout d'abord une pré-initialisation dont le but est de sauvegarder un ensemble de données (avertissements et erreurs locales) avant la réinitialisation effective du PE. La phase de diagnostique s'étend de proche en proche, permettant au fur et à mesure de lever les isolations et de réallouer les tâches éventuellement avec un *remapping* s'il s'agit d'une faute permanente. *D-Scale* a nécessité des développements à la fois au niveau logiciel et matériel impliquant une attention particulière quant au compromis performance/coût : ces contributions sont illustrées sur la **Figure 51**. On retrouve bien les différentes étapes du processus décrit précédemment, ainsi que le lien entre les blocs fonctionnels.

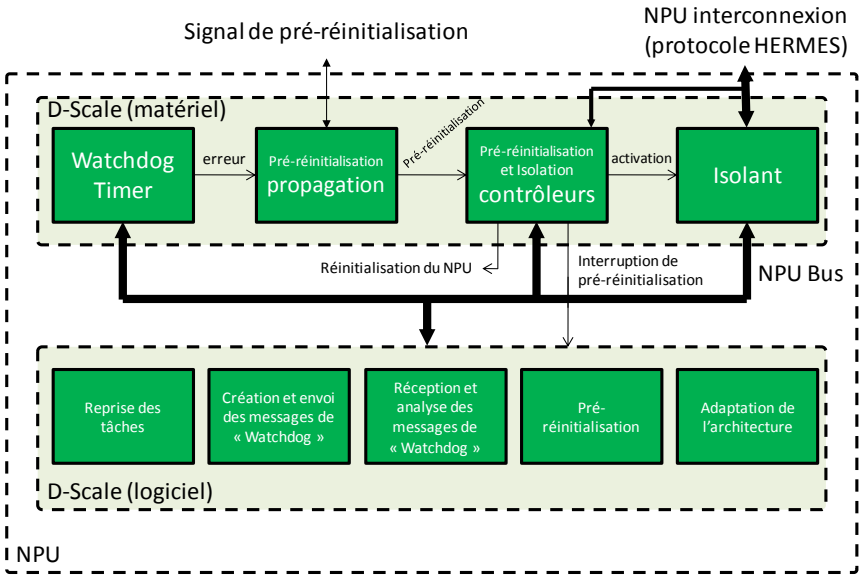



**Figure 51.** Implémentation de D-Scale (Logiciel et Matériel)

### 3.1.3 Résultats

Pour évaluer la solution développée, un environnement basé sur un outil d'analyse de fautes a été mis en œuvre autour du modèle SystemC de *HS-Scale* (**Figure 52**). L'objectif était de tester la fiabilité d'une application MJPEG mappée sur 8 NPU d'une grille 3x3 (9NPU) avec 1 NPU de secours. Le modèle de faute retenu tient compte des défauts apparents et latents et sont de type collage à 0 ou à 1. Les injections de fautes sont réalisées en créant un mutant du composant fautif. Une liste de fautes à tester est transmise à l'outil d'analyse (FEAz) : cette liste concerne les 32 bits de l'ensemble des registres du processeur où chaque bit peut être collé à 1 ou à 0. Les modèles qui en résultent sont simulés, en mode normal et en mode *D-Scale* qui inclut les mécanismes décrits plus haut. Les traces générées à différents niveaux (logique, transactionnel) sont analysées afin de déterminer l'effet de la faute injectée, à savoir si elle a été détectée, corrigée, ou si elle a entrainé un crash du système. Le protocole expérimental est représenté sur la **Figure 53**.

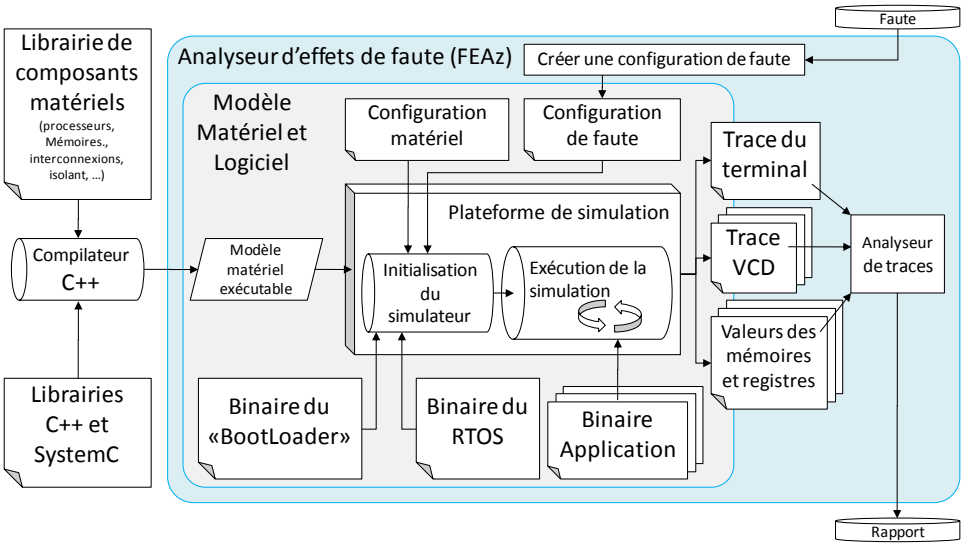



**Figure 52.** Analyseur d'effet de faute (FEAz)

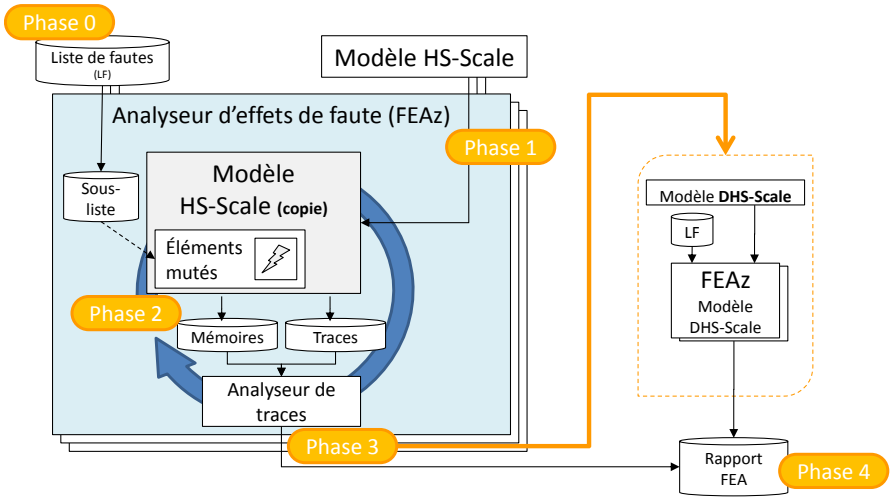



**Figure 53.** Protocole expérimental

L'analyse des résultats obtenus à l'issue de ces expérimentations permet de valider l'apport de la méthode *D-Scale* par rapport à l'architecture initiale. On note que seulement 2 à 7% (suivant le type de faute) des crash ne sont pas détectés. Ces résultats nous montrent aussi que plus de 90% des crashs ont été corrigés. L'outil développé permet également une analyse fine, à l'échelle des registres, et même au niveau bit, des points sensibles de l'architecture, qu'il conviendrait alors de rendre plus robustes par des méthodes de redondance au niveau logique par exemple. Un exemple de résultats obtenus sur les différents types de fautes injectées, ainsi que leur effet à l'échelle des différents registres du processeur est donné dans la **Figure 54**.

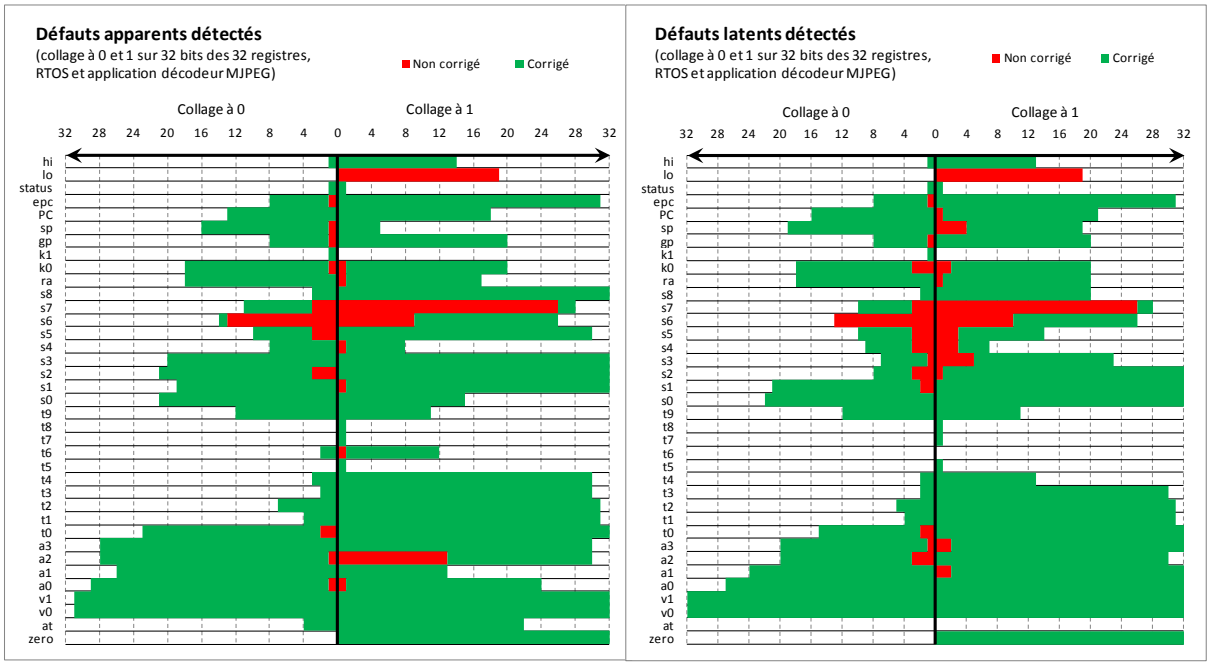



**Figure 54.** Conséquences de l'injection de fautes dans les registres du processeur avec D-Scale

Un autre exemple encore plus précis, puisqu'il se situe au niveau bit, compare dans le cas de fautes apparentes (collage à 0 dans cet exemple), pour chacun des registres, l'effet pour le MPSOC avec ou sans D-Scale (**Figure 55**). Cela révèle notamment que la méthode développée est très efficace dans la protection du RTOS, mais on observe aussi que lorsque des fautes affectent l'application, elles peuvent entrainer le crash de l'application sans que celui-ci ne soit détecté.

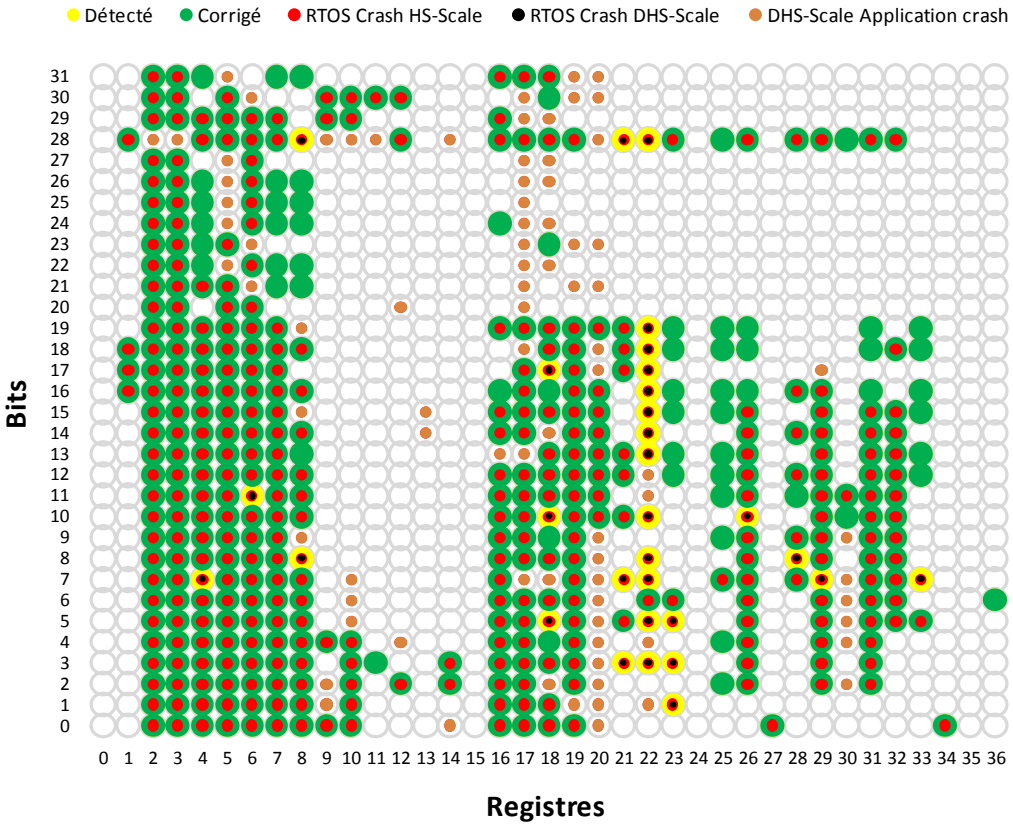



**Figure 55.** Crashs de HS-Scale et DHS-Scale dus aux défauts apparents (collage à 0).

Enfin en ce qui concerne le coût de la méthode D-Scale, les estimations montrent une pénalité fortement limitée puisque la surface additionnelle ne représente qu'un surplus de 13% à l'échelle du MPSOC, une consommation très faible (inférieure à 1% de celle du NPU), et en termes de performance, la pénalité de la routine du *Watchdog* reste négligeable si sa fréquence d'activation est choisie en adéquation avec la granularité des tâches. Le temps de reprise après réinitialisation pour un système à 9 NPU est de l'ordre de 200k cycles, ce qui représente à une fréquence de la centaine de MHz, une indisponibilité du système de quelques millisecondes, ce qui est tout à fait remarquable puisque quasi indécelable à l'échelle de l'utilisateur.

## 3.2 Monitoring des paramètres PVT dans les circuits FPGA

Nous avons vu dans les travaux de Nicolas Hébert une méthode originale tirant avantageusement parti des caractéristiques des MPSOC homogènes adaptifs, afin d'en améliorer la fiabilité. Comme nous l'avons énoncé, cette contribution se justifie par le fait que des défauts susceptibles de créer des fautes peuvent survenir lors de la fabrication, ou au cours du cycle de vie du composant. Ce besoin d'une plus grande fiabilité est motivé par une amélioration plus rapide des rendements de production, un accroissement de la durée de vie, et un service minimum garanti, même s'il peut s'agir de modes de fonctionnement dégradés. Nous nous intéressons ici, toujours dans un contexte de systèmes intégrés adaptatifs, plus particulièrement aux conséquences de ces défauts et aux moyens d'en mesurer les effets. De part la miniaturisation toujours plus poussée, les procédés technologiques sont de plus en plus complexes à maîtriser, ce qui implique notamment des incertitudes plus grandes sur les dimensions effectives des transistors, les concentrations de dopants, etc. A cela peuvent s'ajouter les effets des variations environnementales (tension d'alimentation, température, radiations, etc.) ou applicatives, qui sont susceptibles elles aussi d'impacter les performances des circuits. La prise en compte de ces contraintes se traduit par des marges de design autour des situations pire-cas toujours plus grandes, et donc des gains en performance réduits.

C'est pour répondre à ces contraintes difficilement prévisibles en temps de conception que des solutions sont mises en place notamment en s'orientant vers une adaptabilité des circuits. Celle-ci passe comme nous l'avons vu dans la partie précédente, par l'utilisation de méthodes de contrôle et d'optimisation en ligne. Or pour contrôler et optimiser le circuit, il faut avoir de lui une estimation précise de son état, condition nécessaire à une adaptation en adéquation avec les objectifs recherchés (performance, efficacité énergétique, fiabilité, etc.). On peut définir l'état du système à un instant donné par un ensemble de métriques : pour cela il va donc falloir mesurer des signaux caractéristiques, par le biais de capteurs, conditionner et traiter les valeurs obtenues afin de fournir au contrôleur les entrées nécessaires à la mise en œuvre du processus d'optimisation. L'ensemble des capteurs et de l'infrastructure logicielle/matérielle requise pour réaliser cette estimation en ligne constitue la partie monitoring (ou de surveillance) du système.

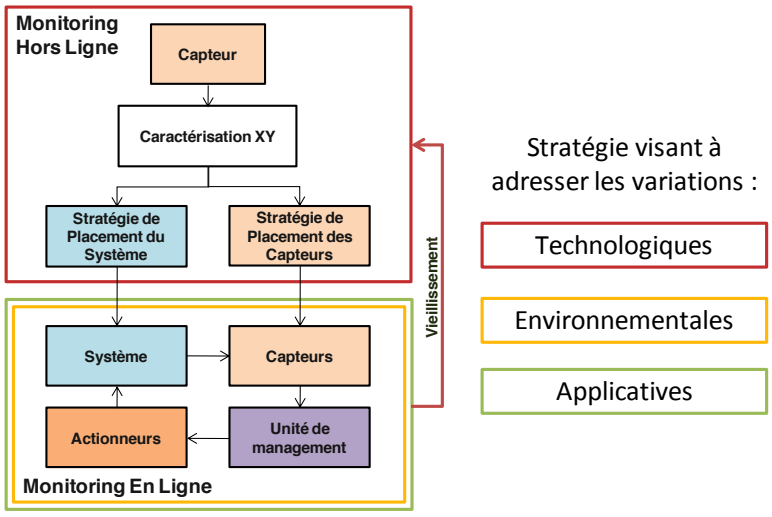



**Figure 56.** Les différentes étapes pour définir un monitoring intégré

L'objectif de ce travail est très clairement focalisé sur le monitoring des circuits FPGA. Cela inclut donc la définition de ce que l'on va chercher à mesurer, l'étude et la mise en œuvre de capteurs, le conditionnement et le traitement des signaux issus de ces capteurs, et l'utilisation des données correspondantes dans le but d'adapter un système reprogrammable doté de ce monitoring intégré. La démarche complète est illustrée par la **Figure 56**. Comme on peut le noter, la méthode repose sur une étape hors ligne, en phase de conception, et une autre en ligne, en cours de fonctionnement (qui met en œuvre l'adaptation dynamique). La première vise à caractériser le composant reconfigurable. Pour cela, un capteur est utilisé pour établir une cartographie en deux dimensions du FPGA. Celle-ci permet de déterminer les performances réelles en chaque point du circuit. Les outils de synthèse du FPGA

peuvent être contraints en fonction des mesures effectuées, et des capteurs peuvent être placés en des endroits stratégiques du système à implémenter. Ce dernier bénéficie de l'ensemble des éléments nécessaires à son adaptation ; un actionneur envisageable dans ce cas peut-être la reconfiguration dynamique de fonctions. Enfin, la démarche est complétée par un bouclage du processus permettant de refaire une caractérisation complète du FPGA, autorisant ainsi la prise en compte du vieillissement à travers une boucle d'adaptation plus lente. On remarque que cette méthode adresse aussi bien le monitoring technologique, qu'environnemental et applicatif.

C'est dans ce contexte que s'inscrivent les travaux de Florent Bruguier. Ils s'adressent en particulier aux composants reprogrammables type FPGA, car ils présentent un support technologique tout à fait intéressant dans une optique d'adaptation. Contrairement aux composants non reprogrammables, qui ont une implantation physique fixe par nature, ceux-ci autorisent par simple reconfiguration des adaptations du placement/routage des fonctions logiques qui composent le système. Ils présentent une très grande flexibilité, attribut déjà largement commenté dans la première partie de cette synthèse. Cette caractéristique intrinsèque est évidemment un atout de choix dans le but d'adaptation que nous nous sommes fixés.

### 3.2.1 Monitoring hors-ligne

L'étude des capteurs spécifiques actuellement disponibles dans les circuits reconfigurables a permis de révéler que leurs caractéristiques n'étaient pas en adéquation avec nos prérogatives : généralement localisés à un endroit bien particulier, ils ne permettent pas une résolution spatiale élevée. Une seconde approche consiste à concevoir des capteurs à partir des ressources logiques (LUT) pour évaluer la performance à partir de la mesure de délais de propagation ou pour estimer la température de circuit grâce à des mesures de fréquences d'oscillateurs en anneau. Cet état de l'art révèle un certain nombre d'approximations, notamment la prise en compte de l'erreur intervenant dans la chaîne, ou encore de l'effet des perturbations potentielles. Une première contribution de ce travail concerne donc la définition de deux capteurs : le PDS (Path Delay Sensor) et le ROS (Ring Oscillator Sensor). Un travail d'expérimentation complet a permis pour chacun de déterminer leur gamme de mesure, leur sensibilité, leur résolution et leur exactitude de mesure, et ce par la mise en œuvre d'un contrôle précis de l'environnement du circuit sous test (température et tension). Après cette première étude, notre choix s'est porté sur l'utilisation du ROS qui présente une plus grande résolution spatiale du fait de sa compacité.

La deuxième partie de ce travail concerne toute la phase de caractérisation nécessaire à l'établissement d'une cartographie complète du circuit (monitoring hors-ligne). L'état de l'art nous montre une approche généralement choisie dans cette optique, qui consiste à implémenter le capteur dans une position donnée, relever la mesure (en général une fréquence ou un délai), puis reconfigurer le circuit en plaçant le capteur dans une autre position, et ainsi de suite, dans chacune des localisations potentielles pouvant accueillir le capteur. On réalise ainsi une carte de performance ou de fréquences en 2 dimensions. Notons qu'en général cette caractérisation ne concerne que les ressources reconfigurables à grain fin. Comme la structure choisie (par exemple un oscillateur en anneau) est sensible aux conditions environnementales, il est nécessaire de contrôler parfaitement les paramètres pouvant influencer la mesure, c'est à dire la température et la tension (d'où l'utilisation requise d'une enceinte thermique et d'une alimentation stabilisée externe). Outre le capteur, la réalisation de la chaine de mesure requiert un conditionnement des signaux dont le rôle est d'amplifier et de transmettre les mesures numérisées que l'on peut par la suite analyser. La littérature nous révèle là encore une approche consistant à intégrer cette partie de conditionnement à l'intérieur même du composant reconfigurable, tel que cela est représenté sur la **Figure 57**.

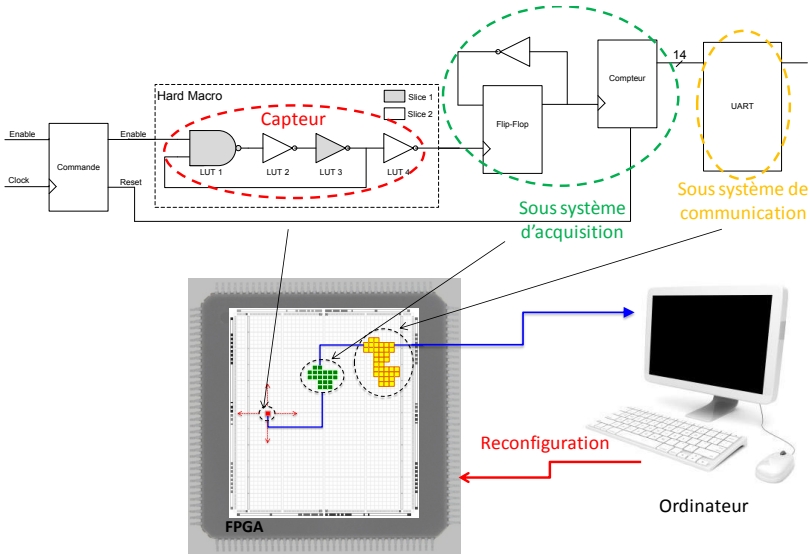



**Figure 57.** Chaîne de mesure proposée dans la littérature

Or cette méthode n'est pas idéale puisque l'implantation d'une logique de conditionnement (sous-système d'acquisition et de communication) peuvent impliquer une augmentation de la température du circuit ainsi que des appels en courant susceptibles d'introduire des erreurs de mesure. Suivant la position du capteur, il est clair que cette influence ne sera pas la même. Pour démontrer cela, nous avons réalisé une expérience très simple résumée dans la **Figure 58**. La cartographie du circuit a été réalisée avec une chaine de conditionnement du signal placée à deux endroits différents : les deux cartographies présentent évidemment une forte similitude, mais aussi des différences flagrantes. Cette méthode n'étant donc pas totalement satisfaisante, elle nous a amené sur de nouvelles pistes à explorer permettant de s'affranchir de cette partie logique d'acquisition et de communication.

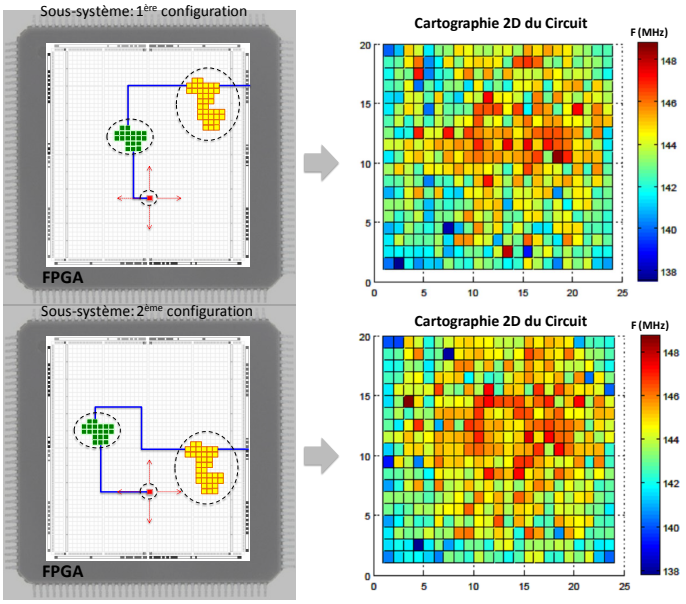



**Figure 58.** Mise en évidence de l'impact de la chaine de conditionnement interne

Nous avons pour cela proposé une approche tout à fait originale se basant sur la capture des émanations électromagnétiques du circuit. Comme dans la littérature, le capteur est positionné en différents endroits du circuit, successivement. Mais au lieu de faire intervenir une logique supplémentaire intrusive, nous utilisons une sonde (spire) positionnée au dessus du circuit qui capte en

champs proche le signal électromagnétique induit par l'oscillateur en anneau. Ce signal est amplifié par un LNA, transmis à un oscilloscope numérique qui réalise une FFT permettant de récupérer la fréquence d'oscillation. Le processus de mesure est illustré sur la **Figure 59**. La mise en œuvre de notre protocole expérimental impliquant une enceinte thermique, l'alimentation stabilisée et la méthode EM, a permis d'établir une estimation de l'erreur introduite par cette chaine de mesure, qui est d l'ordre de la centaine de kHz, alors que la méthode de l'état de l'art présente au mieux une précision de l'ordre du MHz. Cette nouvelle approche est donc 10 fois plus précise que la précédente.

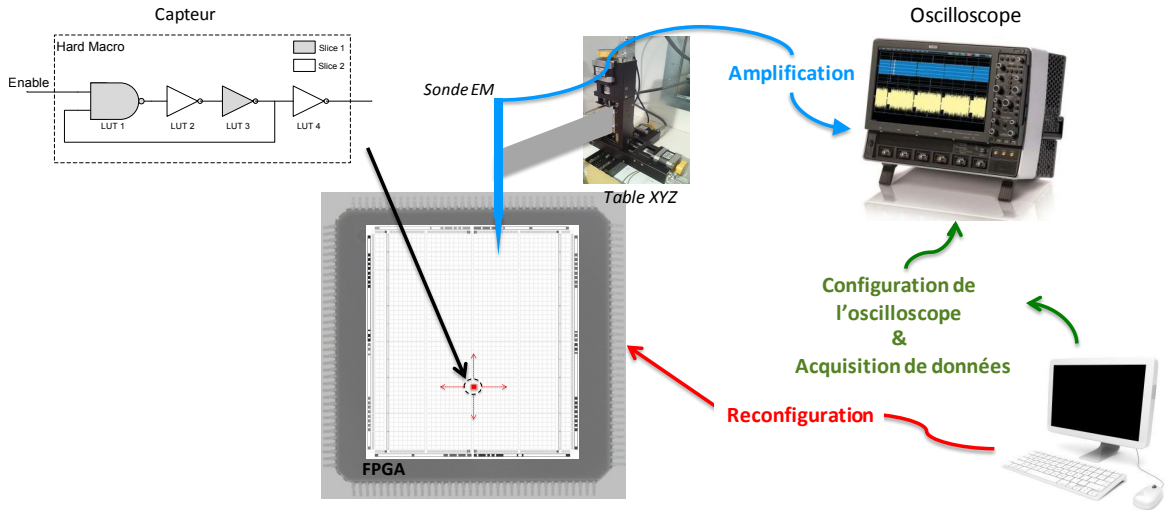



**Figure 59.** Méthode de caractérisation basée sur la mesure des émanations électromagnétiques (EM)

La mise en œuvre de notre méthode a permis de réaliser un ensemble de cartographies de différents circuits FPGA XILINX (technologies 90 à 45nm). Une illustration des résultats obtenus est donnée dans la **Figure 60** qui montre 2 cas assez significatifs. Le premier à gauche représente une cartographie typique d'un composant neuf, le deuxième à droite nous montre un composant vieilli : généralement les ressources les plus utilisées se situent plutôt au centre du circuit, et on constate ici que ce sont celles les plus impactées. Cette observation nous indique que cette méthode peut être envisagée pour évaluer les effets du vieillissement.

Cette méthode nous a permis de développer des collaborations avec plusieurs laboratoires. Elle a notamment été mise en œuvre sur des technologies de FPGA Flash [**38**], puis utilisée pour étudier l'impact du vieillissement sur la sensibilité des FPGA aux radiations [**39**]. Le vieillissement est également un thème sur lequel nous collaborons avec le KIT, afin d'établir des modèles permettant à partir d'un ensemble de paramètres tels que la fréquence, la température, la tension d'alimentation, l'activité, etc., de prédire les performances du circuit, ou encore de mettre en place des mécanismes de mitigation afin de réduire l'impact de la dégradation des transistors [**40**].

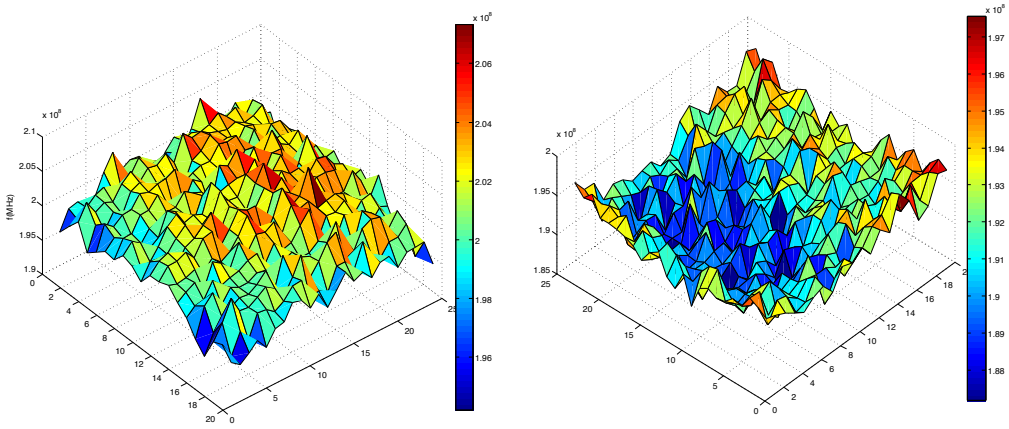


**Figure 60.** Exemples de cartographies obtenues par la méthode EM

### 3.2.2 Monitoring en ligne

Comme nous l'avons vu précédemment, ces mêmes capteurs (utilisant le principe du ROS) sont placés en des endroits stratégiques du système à surveiller. La détermination formalisée du nombre et du placement n'est pas un problème directement abordé dans le cadre de ces travaux (nous verrons que ce sujet a été adressé dans la thèse de Imen Mansouri, et également dans celle de Mohamad Najem), mais on peut intuitivement déterminer ces positions, comme par exemple à proximité d'une unité de calcul dont l'activité est élevée. L'objectif ici était de démontrer la possibilité d'utiliser des capteurs à très faible empreinte silicium (seulement quelques LUT) pour *monitorer* le système, en vue de son optimisation.

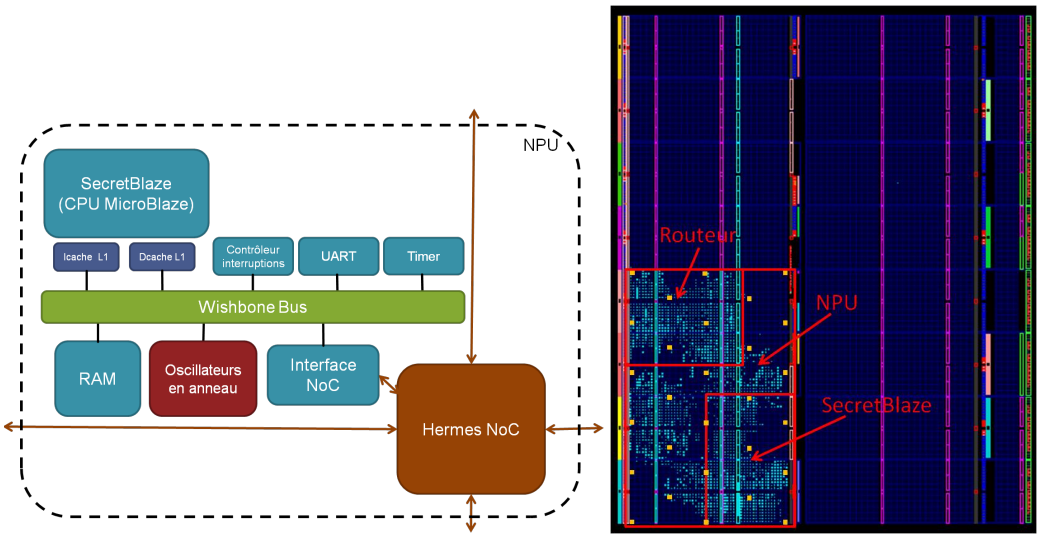



**Figure 61.** Open-Scale augmenté d'un module de monitoring distribué et implémentation sur Virtex 5

Nous avons donc réalisé une expérience dans laquelle nous avons placé 28 capteurs composés chacun de 3 ROS différents (présentant une sensibilité différente à la tension et la température) au sein d'un NPU de Open-Scale. Chacun de ces capteurs est mappé en mémoire et connecté au bus Wishbone afin de pouvoir venir lire les valeurs enregistrées en cours de fonctionnement. La **Figure 61** montre le schéma de l'architecture d'Open-Scale avec ses capteurs intégrés, ainsi qu'une vue de l'outil Plan Ahead de Xilinx permettant d'observer le placement contraint du NPU et de ces différents capteurs.

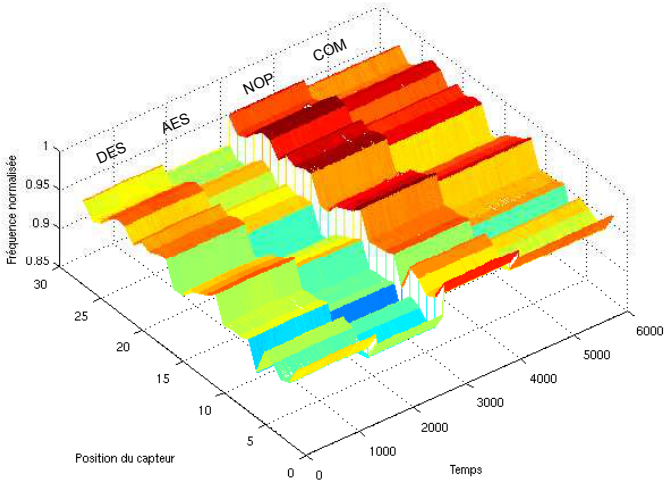



**Figure 62.** Résultats de mesures obtenus en ligne

Nous avons généré un binaire d'une application qui enchaine en série 4 tâches différentes (2 applications de chiffrement AES et DES, une application qui ne fait que des NOP et une application qui fait usage de l'UART, COM). Les données collectées lors de l'exécution de cette application sont représentées sur la **Figure 62**. Ce graphique en 3 dimensions nous donne en fonction du temps et de la position du capteur (ici son numéro), la fréquence mesurée au cours de l'expérience. Il se dessine en chaque point des paliers caractéristiques pour chacune des applications, ce qui montre la sensibilité du capteur dans le temps en fonction de l'activité (qui a un impact direct sur V et T). Ensuite, on observe des différences spatiales qui nous confirment que le positionnement du capteur a une influence sur la mesure, et que cela peut être d'un intérêt tout particulier dans la stratégie de monitoring que l'on cherche à mettre en place. A partir d'un modèle différentiel, cette expérience a permis de reconstituer une carte (V, T) du circuit en cours de fonctionnement. Notons que plus récemment nous avons fait l'acquisition d'une caméra thermique, et que celle-ci nous a permis de valider les données issues du monitoring, ce qui prouve la pertinence de cette approche.

## 3.3 Monitoring de la consommation

Un moniteur peut être vu comme un dispositif intégré au système chargé de faire remonter des informations sur son état. Son rôle est de détecter à différents niveaux, du matériel au logiciel, l'apparition d'un phénomène, d'un évènement, ou bien de mesurer des métriques de performance. Nous pouvons mentionner par exemple la charge de calcul, déjà vue notamment pour les techniques de *load-balancing*, la température, la consommation, des alertes de dysfonctionnement telles que des erreurs de timing, ou défaillance du système, etc. Le monitoring est un processus qui assure tout d'abord la détection ou la mesure, puis la mémorisation des données et enfin leur signalisation/transmission. Dans une stratégie d'optimisation en ligne, il est bien sûr évident qu'en plus des contraintes sur sa surface, celui-ci doit être le moins intrusif possible, c'est-à-dire ne doit pas (ou peu) impacter le système. Nous avons vu dans la partie précédente la nécessité de mettre en œuvre des capteurs dans le but de pouvoir adapter de manière optimale le système. Notre étude s'est pour l'instant focalisée sur des capteurs PVT, essentiellement de type RO, avec une mise en application sur circuits FPGA. Nous développons ici notre approche de monitoring par des méthodes visant à établir une estimation de l'état avec un minimum de ressources et une précision maximale.

Intéressons nous tout d'abord à l'ensemble des capteurs susceptibles d'apporter de l'information utile à l'élaboration d'une estimation précise de l'état du système, en partant de structures matérielles pour de l'observation bas niveau, jusqu'à des approches logicielles plus macroscopiques.

Au niveau circuit, nous pouvons faire référence aux moniteurs technologiques qui s'utilisent par exemple pour détecter des phénomènes liés aux variations du procédé (après fabrication ou suite au vieillissement). Les oscillateurs en anneau et moniteurs de chemin critiques (CPM pour Critical Path Monitor) font partie de cette catégorie (Architecture Razor **[41]**, réplication de chemin critique Replica **[42]**, etc.). Ce type de structure produit des mesures à une fréquence élevée (ou proche de celle du fonctionnement du système), et donc le débit d'information généré constitue une première contrainte dans leur mise en œuvre.

Si la vitesse de réactivité souhaitée est plus lente que la cadence des moniteurs, on peut envisager la mise en place d'un module intermédiaire pour rendre plus synthétiques les données brutes, tel que le comptage du nombre d'erreurs ou d'évènements, déclenchant une interruption au delà d'un certain seuil. Cette fonction est assurée par des capteurs d'activité spécifiques ou des structures de type BIST (*Built-In-Self-Test*) embarquées dans le circuit dans le but de vérifier son intégrité après fabrication, ou au cours de son cycle de vie. Cette procédure d'instrumentation peut être répétée à faible fréquence (deux fois par mois par exemple) lorsque le circuit est inactif. Les capteurs d'activité sont largement mentionnés dans la littérature et présents dans de nombreux systèmes intégrés. Ces dispositifs caractérisent l'activité engendrée au sein de l'architecture au cours de l'exécution des applications, et permettent notamment le débogage ou l'optimisation du code, d'où la dénomination de capteur de performance, au sens applicatif du terme. Ils permettent d'identifier des évènements tels que l'activation de composants particuliers, la charge des accès mémoire, des opérations de calcul spécifiques, etc. On retrouve ces dispositifs dans la plupart des processeurs modernes dans des unités de monitoring configurables (en général un ensemble de N compteurs pour M évènements observables, avec N<M) : par exemple *Performance Monitoring Unit* pour les processeurs ARM, *Performance Counter Monitor* pour INTEL. Dans le contexte des architectures NOC, ce sont des moniteurs de trafic qui ont été proposés afin de pouvoir identifier des points de congestion, et d'adapter le routage par exemple. Bien qu'elle contribue à une complexité accrue du réseau, cette démarche permet d'améliorer la qualité de service ou encore éviter la surcharge de certains liens de communication.

Au niveau logiciel, les systèmes d'exploitation peuvent offrir, une interface de visualisation par le biais de threads dédiés qui récupèrent l'identifiant de la tâche en cours ainsi que le contenu des registres d'activité. On trouve également des techniques de monitoring qui ciblent directement le profilage de l'application, par le biais de processus d'instrumentation du code permettant de mesurer la fréquence d'exécution des séquences de code, ou le nombre de transitions d'un processus à un autre, et plus globalement de calculer la charge des différents processeurs, etc.

Comme nous venons de le voir, il existe de nombreux moyens permettant d'observer le système, du support matériel jusqu'au logiciel, en passant par des couches intermédiaires. Rappelons que l'objectif est d'estimer l'état du système afin de l'optimiser en ligne : suivant la politique d'adaptation (performance, consommation, fiabilité, etc.), la réactivité requise ne sera pas la même. Il faut donc mettre en place une chaîne cohérente entre l'observation et l'action. Autrement dit, il faut une estimation de l'état qui soit précise et à jour au moment de la prise de décision. Il faut également tenir compte du fait que les phénomènes physiques en jeu sont liés, mais qu'ils n'ont pas tous la même dynamique : c'est le programme et les données qui fixent l'activité du circuit, qui impose une certaine consommation de puissance dynamique, qui implique à son tour une température d'équilibre du circuit dépendante de son environnement, qui a un impact sur la consommation statique, donc la consommation globale, mais aussi sur la vitesse de vieillissement donc la fiabilité du circuit… Et c'est en fonction de toutes ces informations que le contrôleur doit décider d'un placement de tâche, d'une isolation de processeur, d'une affectation de tension ou de fréquence à un cœur, etc. Il y a donc premièrement une dimension temporelle dans l'observation, dont la fréquence et la latence qui doivent être choisies en adéquation avec le phénomène sous surveillance, et l'actionneur. Il y a ensuite une dimension spatiale, dans le sens où ces phénomènes se produisent en différents endroits du circuit, avec une granularité plus ou moins fine.

### 3.3.1 Flot de modélisation

C'est dans l'optique de définir une stratégie de monitoring optimale de la consommation du circuit que nous avons défini une nouvelle approche générique permettant de déterminer l'emplacement optimal des capteurs d'activité dans le circuit, et ce sans avoir de connaissance a priori de son architecture. Il s'agit d'identifier systématiquement les signaux ayant l'activité la plus corrélée avec la consommation du circuit, de les associer ensuite à des compteurs qui comptabilisent l'activité sur une période donnée, à pas fixe ou variable. C'est ce que nous allons présenter dans les sections suivantes.

Cette stratégie repose sur un flot de modélisation. La première étape consiste à instrumenter le circuit. Par cela, nous entendons la mise en place d'un environnement permettant de récupérer toutes les informations nécessaires à l'estimation de la consommation. Cet environnement est décrit sur la **Figure 63** : le circuit est simulé d'une part au niveau RTL et d'autre part après synthèse/ placement-routage. Ceci permet de générer une base VCD utilisée par un outil qui fait une estimation précise de la puissance instantanée consommée (profil de consommation). Au niveau fonctionnel, les traces d'évènements sont récupérées dans des fichiers LST. Une fois les fichiers VCD et LST disponibles, on passe à la deuxième étape du flot consistant à établir un modèle linéaire de la consommation moyenne en fonction des évènements, sur une période de temps, de la forme suivante :

$$P_T = c + \alpha_1 N_{e_1} + \ldots + \alpha_i N_{e_i} + \ldots + \alpha_n N_{e_n}$$

Une telle relation repose sur la définition des éléments suivants :

- $\{e_i\}_{i=1:n}$ l'ensemble des évènements caractérisant l'activité du bloc sur un niveau matériel.
- $N_{ei}$ est le nombre d'occurrence de l'évènement $e_i$ au cours de la durée sur laquelle la moyenne est calculée T.
- $\alpha_i$ est un coefficient dégagé par régression linéaire. $\alpha_i$ permet de quantifier la consommation partielle due à l'évènement $e_i$ tout en tenant compte des sous-blocs non modélisés, *i.e.* dont l'activité ne figure pas l'équation ci dessus.
- la constante de régression *c* englobe la puissance statique du bloc et un terme correctif du modèle.
- *n* définit la complexité du modèle, c'est le nombre d'évènements qui y sont inclus. C'est aussi le nombre d'opérations de calcul à effectuer pour estimer $P_T$.

Nous avons proposé deux méthodes pour établir un modèle de la consommation. La première repose sur un échantillonnage des compteurs à pas constant, la deuxième à pas variable se base sur un signal qui permet de définir des modes de consommation. Dans les deux cas, des méthodes de filtrage sont

mises en œuvre pour extraire les signaux, au niveau fonctionnel, dont l'activité est fortement corrélée avec les variations de consommation. Puis par régression, le ou les modèles linéaires sont générés. Suivant la méthode retenue, les compteurs d'activité sont synthétisés et attachés aux signaux sous surveillance sélectionnés.

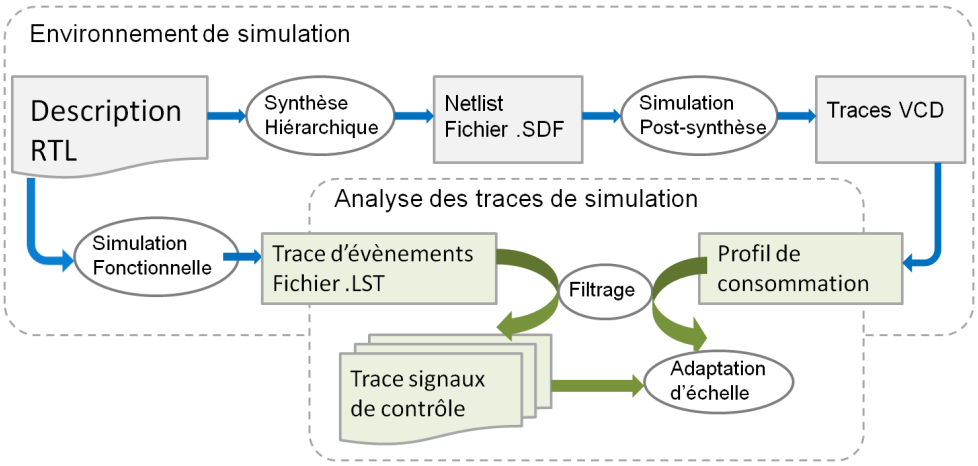



**Figure 63.** Flot de modélisation

### 3.3.2 Monitoring de la consommation à pas constant

Pour établir une estimation de la puissance du circuit, il faut tout d'abord passer par une sélection des évènements. Cette approche repose sur l'hypothèse que de nombreux signaux présentent une redondance statistique importante. Notre méthode utilise un algorithme nommé *stepwise*, permettant de réaliser des régressions multiples en présence de variables colinéaires. Le degré de pertinence des variables est quantifié grâce aux mesures de corrélation partielle : la significativité est testée et la variable proposant la valeur p la plus faible est choisie. Chaque nouvelle variable introduite entraine la remise en compte de celles ajoutées aux itérations précédentes, puisqu'une variable peut avoir perdu en signification par l'introduction conjointe de la nouvelle variable et d'une autre déjà dans le modèle : la significativité est donc ré-estimée et si elle est supérieure à un seuil donné, elle est rejetée. Si l'on cherche à avoir le moins de variables possibles dans le modèle on fixe les deux seuils de probabilités à de faibles valeurs. Le réexamen des composantes du modèle à chaque itération permet d'accroître la pertinence du processus de sélection des évènements dans notre contexte.

Pour nos expérimentations, nous avons choisi un processeur VLIW (Very Long Instruction Word) dédié aux applications télécom qui est l'une des briques de calcul de l'architecture GENEPY. Le contrôleur de ce processeur présente une grande complexité tant sur le plan fonctionnel que matériel. Il contribue à hauteur de 44% de la puissance totale du processeur. Synthétisé en technologie 65nm de ST, il occupe une surface de 0,27mm2. Nous avons appliqué notre approche pour évaluer la consommation du contrôleur avec différents niveaux de précision. La **Figure 64** présente les courbes des puissances estimées par les différents modèles (complet et réduits) superposées à celle de la puissance réelle mesurée avec l'outil *PrimePower*. Un deuxième graphique représente l'erreur en valeur absolue de ces modèles. D'après cette figure, nous pouvons constater que l'erreur des modèles établis en supprimant les évènements [$e_6$ $e_7$] augmente légèrement. Cette erreur augmente encore plus avec la suppression du signal $e_5$, en moyenne nous l'estimons à peu près 4%. Le modèle reste néanmoins assez fidèle aux mesures. Ensuite, il est clair que le retrait d'un signal critique remet en cause la fiabilité du modèle.

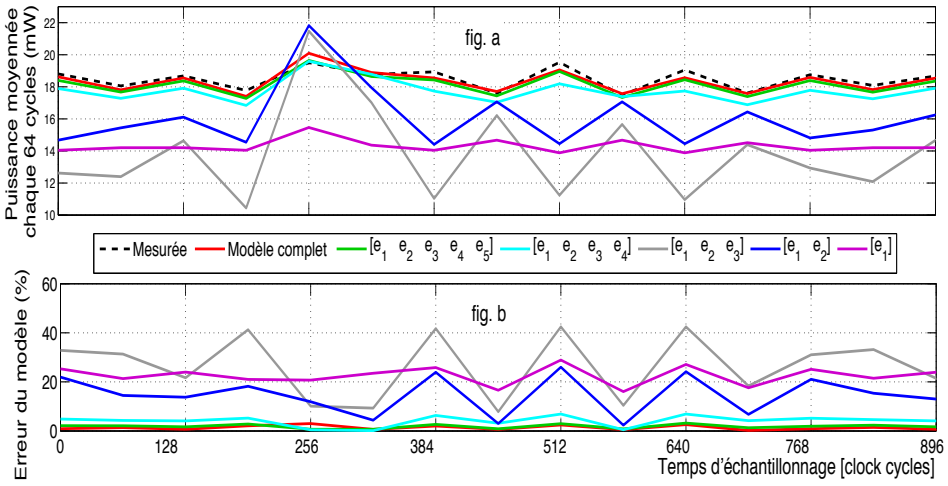



**Figure 64.** Différents modèles consommation construits à partir des 5 signaux sélectionnés

### 3.3.3 Monitoring de la consommation à pas variable

Comme nous l'avons vu dans les paragraphes précédents, il est tout à fait possible de modéliser à partir d'un sous ensemble de signaux la consommation totale d'un bloc complexe. A la mise en œuvre de la méthode, on observe cependant des phases de fonctionnement où la précision du modèle est moins bonne. L'objectif ici est d'identifier les évènements qui mènent à des disparités significatives entre les niveaux de consommation et de les relier aux phases de fonctionnement de l'application en cours : on parle de mode de consommation. Une fois ces évènements détectés, un modèle linéaire spécifique pour chacun des modes est mis en œuvre suivant la méthode vue précédemment, ce qui revient à reconfigurer dynamiquement le dispositif de monitoring. Les compteurs d'activité sont rattachés à certains évènements prédéfinis suivant les modes. Au lieu de se positionner en permanence sur un signal particulier, ce dernier est enregistré pour une durée déterminée, puis négligé si le bloc passe à autre mode. Le compteur associé peut être mis dans un état de *standby* où il consomme moins, ou être configuré pour traquer un autre évènement lié à un autre mode. Un seul évènement peut aussi être utilisé de façon permanente (à condition que ce signal soit actif quelque soit le mode) : le modèle associé change alors de paramètre dynamiquement pour s'adapter à chaque phase de fonctionnement et compenser l'absence d'informations fournies par les évènements non observés. La reconfiguration dynamique permet d'améliorer la précision et de réduire davantage le nombre d'évènements à observer simultanément, donc les ressources matérielles associées. Par ailleurs, l'estimation n'est plus réalisée à pas constant comme précédemment, mais dépend du mode de consommation (**Figure 65**).

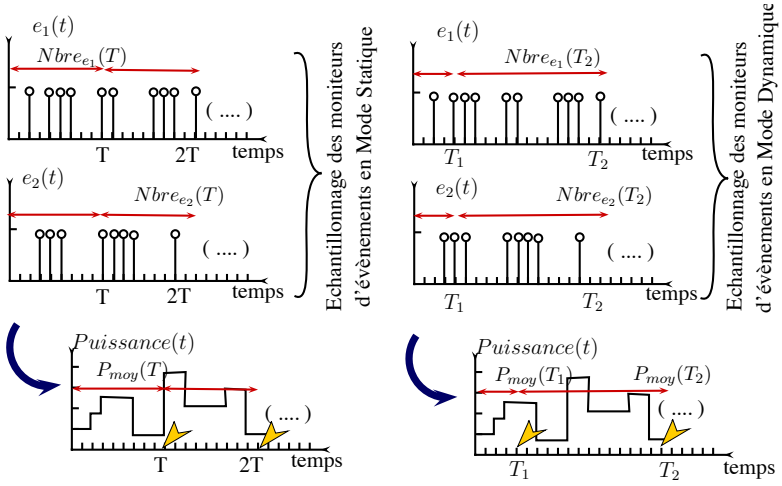



**Figure 65.** Comparaison des méthodes à pas constant et à pas variable

Pour capturer ces phases de fonctionnement caractéristiques d'un niveau de consommation, nous avons utilisé les chaînes de Markov cachées (Hidden Markov Model) **[43]**, et plus précisément le formalisme Markov Switching Model **[44]**. Le processus observé représente la consommation du circuit, la matrice des variables contient les évènements sélectionnés par le *stepwise* avec une période d'échantillonnage très fine (quelques cycles d'horloge). Au niveau du modèle markovien, l'état détecté à chaque phase représente un mode de fonctionnement et correspond à un modèle régressif particulier. En pratique, nous sélectionnons le signal ayant le plus fort impact à l'issue du *stepwise*. L'estimation markovienne permet d'identifier les états afin de les interpréter en fonction des autres signaux sélectionnés par le *stepwise*.

Pour valider cette approche, nous considérons le MAC du processeur spécialisé. Nous avons pris le signal qui active ce bloc pour chaque opération comme une entrée de base pour le modèle (signal *valid* dans la **Figure 66**). Les différents états tels que perçus par le modèle markovien sont au nombre de trois. Pour l'implémentation, le dispositif de monitoring implique une FSM, un compteur associé au signal *valid*, réinitialisé à chaque changement d'état et trois registres accumulateurs pour mémoriser le temps écoulé durant chaque état jusqu'à une lecture du contrôleur.

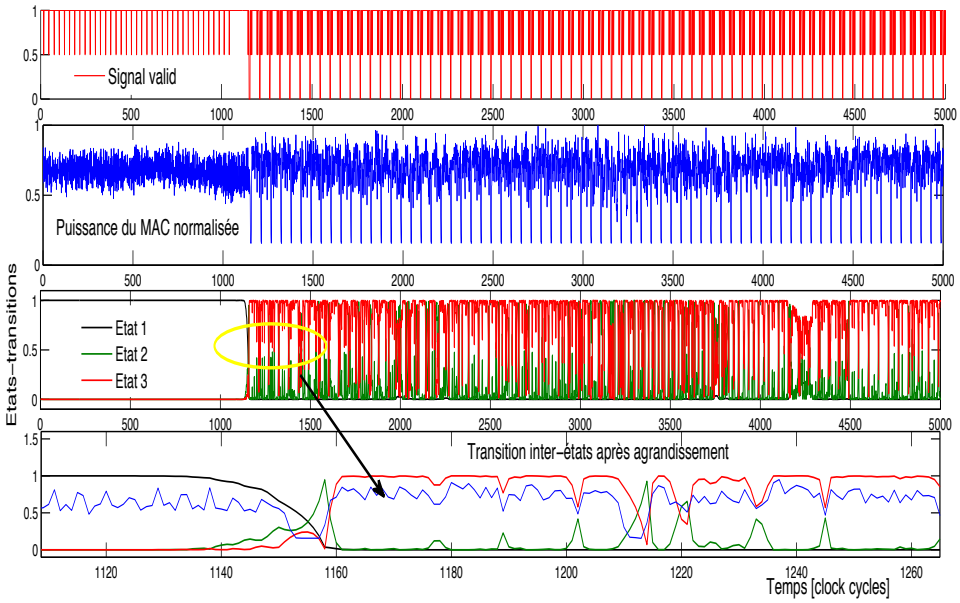



**Figure 66.** Illustration du modèle de monitoring du MAC avec 3 modes de consommation

Pour comparer les deux approches (à pas constant et pas variable), nous avons implémenté le modèle statique avec un seul compteur attaché au signal *valid*. La **Figure 67** montre les sorties des deux modèles superposés aux mesures. La précision du modèle à pas variable comparée à celle du modèle à pas constant est clairement visible, et ce malgré un seul compteur d'activité. Cette deuxième méthode assure plus de flexibilité grâce aux modes de consommation qui permettent d'adapter le modèle selon le contexte d'usage. On retient également un deuxième avantage lié à la réduction du surcoût de synchronisation nécessaire pour implémenter ces capteurs. La communication de ces dispositifs ne se fait plus périodiquement, elle peut-être préemptive à la demande du processus qui gère l'estimation, ou déclenchée sur interruption, ce qui diminue considérablement l'impact du monitoring vis-à-vis de l'application.

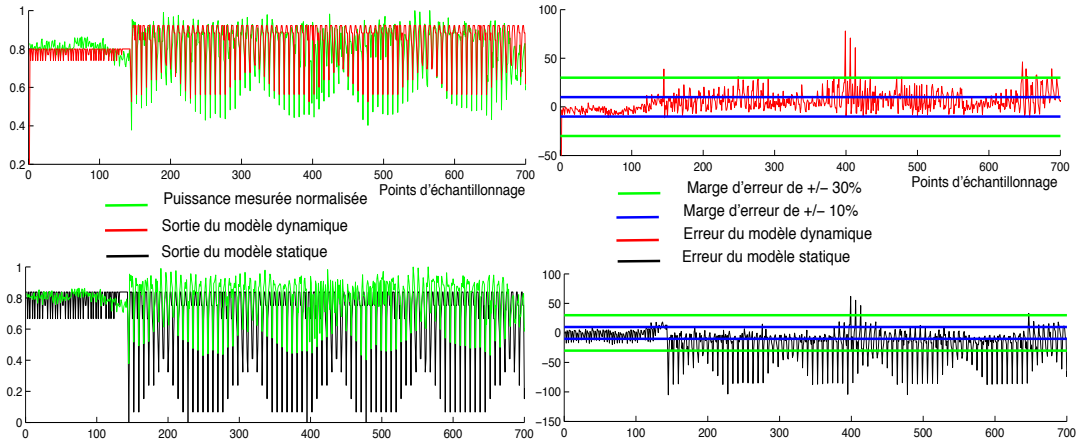



**Figure 67.** Comparaison des deux méthodes avec un seul compteur d'activité

## 3.4 Contremesures pour processeurs RISC

Dans cette troisième partie, nous avons adressé la problématique de la sûreté de fonctionnement, en proposant des mécanismes pour l'amélioration de la tolérance aux pannes, en lien avec nos recherches sur l'adaptation dynamique. Les pannes étant la conséquence de phénomènes physiques de bas niveau, nous nous sommes alors intéressés à la problématique de caractérisation technologique, en vue de développer des dispositifs de monitoring, permettant d'obtenir en ligne une estimation de l'état du système, en vue de son adaptation. Cette étude a été prolongée par une analyse plus générale visant à déterminer de façon systématique une stratégie d'observation précise et faiblement intrusive.

Pour être digne de confiance, un système numérique doit disposer d'un certain nombre d'attributs, comme par exemple la fiabilité et l'observabilité, et un aspect très important que nous n'avons pas abordé jusque là et qui concerne la sécurité. Celle-ci requiert des méthodes et des moyens pour garantir la confidentialité, l'authenticité, l'intégrité et la disponibilité à la fois des données et des systèmes d'informations. Mécanismes de sûreté et de sécurité diffèrent dans le sens où les premiers protègent contre des actions non-intentionnelles, et les seconds contre des actions intentionnelles

Les systèmes intégrés étant le support matériel de l'information numérique, ils constituent un maillon essentiel de la chaîne de sécurité. Celle-ci repose sur la cryptographie, qui elle même s'appuie sur des outils mathématiques permettant notamment de chiffrer des messages afin de les rendre inintelligibles. Si les mathématiques assurent une certaine robustesse vis à vis des outils de cryptanalyse classique, le passage au monde physique pose de nouveaux problèmes. L'implémentation de ces algorithmes cryptographiques repose sur un support matériel (circuit intégré), des *cryptoprocesseurs*, qui réalisent entre autres les opérations de chiffrement. Le problème, c'est que ces implémentations matérielles peuvent laisser fuir des informations par des « canaux auxiliaires » de plusieurs types (temps de calcul, consommation, émissions électromagnétiques, etc.) : par l'observation de ces canaux cachés, il est possible de remonter plus ou moins facilement jusqu'aux clés secrètes et mettre en péril la sécurité du système tout entier.

L'objectif de nos travaux est d'étudier les vulnérabilités des implantations logicielles des algorithmes cryptographiques face à ces attaques afin de concevoir un processeur générique sécurisé. Il est nécessaire pour cela d'identifier les éléments pouvant être susceptibles de laisser fuir les informations secrètes afin de développer des stratégies efficaces privilégiant un équilibre entre performance et sécurité pour protéger de telles architectures. Notons également que c'est dans le cadre de cette étude que fut développé le processeur *SecretBlaze*, qui est la brique élémentaire du MPSOC *OpenScale*. Le *SecretBlaze-SCR* est la version robuste développée dans le cadre de cette thèse, qui implémente un ensemble de contremesures visant à garantir une sécurité matérielle à moindre coût. Pour évaluer l'efficacité des solutions proposées contre les analyses électromagnétiques, nous avons réalisé une implémentation du processeur sur carte FPGA et avons pu valider de manière expérimentale nos résultats.

### 3.4.1 Analyse DEMA d'un processeur RISC non sécurisé

La mise en œuvre d'une analyse par canaux cachés implique un attaquant qui dispose du système de chiffrement, dans ce cas un processeur cryptographique, et d'un équipement permettant de mesurer les informations issues du canal auxiliaire, en l'occurrence il s'agit ici des émissions électromagnétiques. Les échantillons acquis sur un intervalle de temps sont appelés « traces ». Dans la pratique, pour réussir une attaque, il faut faire l'acquisition d'un grand nombre de traces : l'efficacité de l'attaque dépend fortement du niveau de bruit, résultant en partie de la chaîne d'acquisition. Le dispositif expérimental que nous avons utilisé est représenté sur la **Figure 68** (plate-forme de SECNUM). Nous distinguons trois éléments principaux : le dispositif sous analyse (ici une carte FPGA Digilent disposant d'un circuit Xilinx XC3S1000), une sonde en champ proche pour mesurer les émissions électromagnétiques du composant, et d'un oscilloscope numérique (3,5 GHz de bande passante, 40 G *samples* par seconde) pour échantillonner les signaux électromagnétique (il s'agit de l'équipement mis en œuvre dans le cadre des travaux de caractérisation de procédé des FPGA). Une table motorisée XYZ pilotée à distance par une liaison série est utilisée pour positionner avec précision la sonde sur la cible. Enfin, bien que la sonde soit placée au plus près de la surface du dispositif, le signal émis

dispose d'une amplitude très faible. Un amplificateur à faible bruit est donc utilisé pour augmenter la force du signal. Un ordinateur pilote l'ensemble des instruments par l'intermédiaire d'un programme MatLab. Il est relié à l'oscilloscope par un câble Ethernet et au dispositif attaqué par un protocole de communication série.

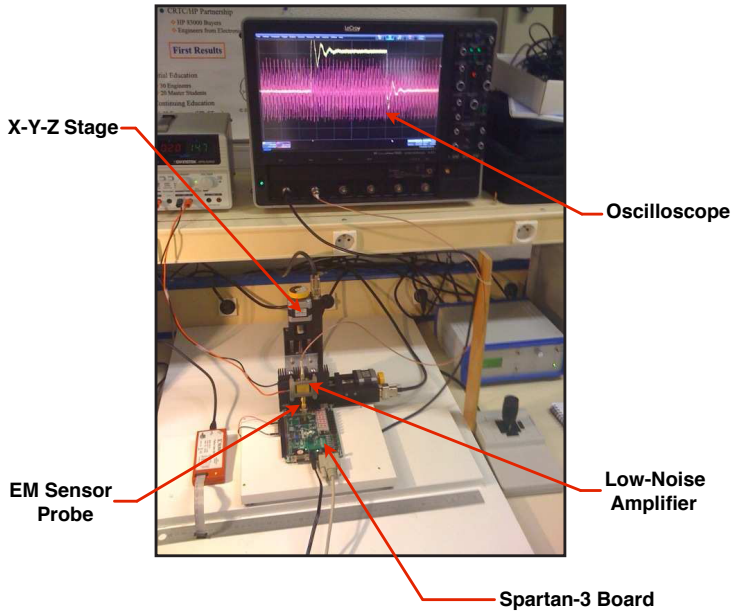



**Figure 68.** Equipement de la plateforme SECNUM utilisé pour réaliser les attaques

Nous avons donc réalisé une attaque DEMA sur le processeur *SecretBlaze*. Les traces DEMA obtenues à l'issue de l'attaque sont représentées sur la **Figure 69** au cours de l'exécution des instructions qui manipulent la clé de chiffrement lors d'un DES. On peut remarquer que la sous-clé correcte (trace noire) semble ressortir pendant plusieurs périodes de temps (plusieurs pics d'amplitude). En ne tenant compte du fait que seulement deux instructions sont sensibles, on pourrait penser ne voir apparaître que deux pics d'amplitude. Pour comprendre cette observation, nous devons considérer tous les détails de la mise en œuvre de *SecretBlaze*. Un élément clé de l'architecture du chemin de données est le pipeline. Les traces confirment donc la vulnérabilité du pipeline qui fait ressortir plusieurs pics, au niveau des éléments séquentiels et combinatoires.

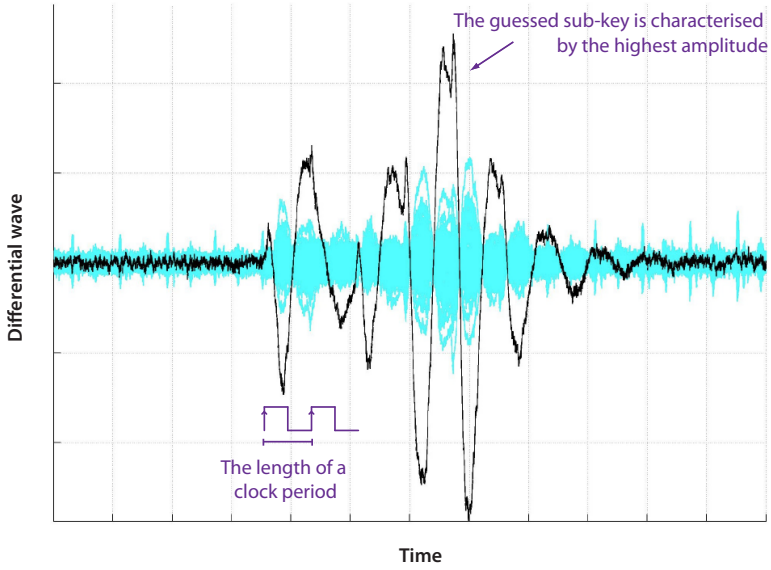



**Figure 69.** Traces DEMA obtenues pour la première sous-clé

### 3.4.2 Contremesures

L'observation faite précédemment montre la nécessité de mettre en place des contremesures dédiées à la protection des éléments sensibles des processeurs génériques. Avant d'entrer dans le détail de nos contributions, nous revenons sur les notions élémentaires de cet aspect. Classiquement, on classe les contre-mesures ciblant les analyses de consommation ou électromagnétiques en deux catégories principales: « *hiding* » et « *masking* » **[45]**. Les techniques de « *hiding* » visent à cacher l'information confidentielle en essayant de rompre le lien entre calculs et fuites, tandis que les celles de « *masking* » masquent les valeurs intermédiaires par des valeurs aléatoires afin de tromper l'attaquant.

La randomisation est sans doute l'approche la plus simple dans la mise en œuvre du principe « *hiding* ». Elle peut être réalisée dans le temps, par randomisation de l'exécution du calcul cryptographique, et en amplitude, par randomisation de la puissance consommée. L'objectif est de complexifier la tâche de l'attaquant qui cherche à retrouver des informations exploitables dans les mesures collectées. L'autre technique consiste à casser le lien entre les calculs et les fuites, en essayant par exemple de faire en sorte quels que soient les calculs et les données, la consommation reste toujours identique, ce qui peut être réalisé par le biais de logiques équilibrées. La combinaison de la logique double rail et de logique à précharge permet d'obtenir un nombre constant de transitions à chaque cycle d'horloge. Par conséquent, la fuite est censée être sensiblement la même quelles que soient les données en cours de traitement, ce qui améliore donc la robustesse du circuit contre ces attaques. La logique SABL (Sense Amplifier Based Logic) **[46]**, la WDDL (Wave Dynamic Differential Logic) **[47]**, la STTL (Secure Triple Track Logic) **[48]** et la BCDL (Balanced Cell-based Differential Logic) (BCDL) **[49]** sont différentes implémentations de ces techniques.

Une autre stratégie consiste à mélanger les valeurs intermédiaires qui dépendent du texte clair et de la clé secrète, à des données aléatoires. Les techniques de masquage booléen et arithmétique cherchent à atteindre cet objectif par l'ajout d'une valeur aléatoire appelée masque. L'opération de dissimulation est définie en fonction du type de l'algorithme cryptographique ; il s'agit souvent du masquage booléen (XOR) ou arithmétique (addition, multiplication). Puisque chaque donnée intermédiaire est masquée par une valeur aléatoire, la fuite mesurée n'est plus directement corrélée à la valeur intermédiaire, mais à la donnée intermédiaire masquée, supposée aléatoire. En conséquence, l'attaquant ne peut plus deviner les bits liés à la clé de chiffrement. Un avantage de cette méthode est son applicabilité à la fois au niveau matériel et logiciel.

**Tableau 9.** Contremesures et impact au niveau de la surface et de la performance

| **Countermeasure** | | **Overhead Factor** | |
|---|---|---|---|
| Type | Algo. | Size | Timing |
| **Hiding** | | | |
| SW randomisation | All | $> 1$ | $> 1$ |
| HW randomisation | DES | 3 to 24 | $> 1$ |
| WDDL | AES | 3 | 3.85 |
| STTL | DES | 3 to 5.5 | 3.25 to 5 |
| BCDL | AES | 2 to 4 | 1.4 |
| **Masking** | | | |
| SW Boolean | DES | 3.4 | 5 |
| SW Boolean/multiplicative | AES | 2.4 | 3.25 |
| HW Boolean | DES | N/C | 1.25 |
| HW Boolean | DES | 1.6 | 1.35 |
| **Hiding & Masking** | | | |
| SW randomisation/masking | AES | N/A | $> 3$ |
| MDPL | AES | 4.76 | 1.7 |

L'analyse d'un certain nombre de publications montre que le surcoût moyen de ces contremesures est relativement élevé (**Tableau 9**). Il en résulte que la plupart d'entre elles ne sont pas adaptées aux systèmes fortement contraints, pour lesquels la compacité et la faible consommation d'énergie sont des facteurs critiques. Il est donc nécessaire d'étudier des solutions en vue de concevoir un processeur respectant ces contraintes et l'objectif de sécurité.

### 3.4.3 Conception d'un processeur sécurisé

Notre objectif est de proposer une solution efficace en termes de sécurité, mais qui soit également peu coûteuse. La difficulté est de trouver le bon compromis, donc de tirer avantage des différents types de contremesures pour atteindre ce but. Notre approche repose sur une technique de « *masking* » dont le but est essentiellement de contenir les fuites du pipeline, et une autre de « *hiding* », dont l'objectif est de randomiser l'exécution des instructions, dans le temps, et en amplitude.

La première méthode repose sur le masquage, dont l'efficacité a été démontrée à plusieurs reprises. En revanche, aucune implémentation n'avait été proposée pour les processeurs génériques au moment de cette étude. Cette technique s'applique ici au niveau de la description RTL du processeur *SecretBlaze* et consiste à mettre en œuvre un double pipeline de traitement.

L'idée est d'introduire un chemin de données dédié au masque lui-même, couplé au chemin de données classique du processeur RISC. Au lieu de traiter directement les données brutes, le processeur fonctionne sur un double chemin de données avec des données masquées. Le rôle principal du deuxième chemin de données est de garder le masque correspondant pour chaque donnée masquée le long de la structure pipeline du processeur et d'assurer la correction de masque pour une exécution correcte des opérations. Des générateurs de nombres pseudo aléatoires (PRNG) sont également introduits pour générer des masques mis à jour à chaque étage. Le modèle simplifié de notre approche pour les instructions de registre à registre est illustré sur la **Figure 70**. Le deuxième chemin de données dispose d'une banque de registres pour le stockage des masques associés aux données masquées, ainsi que des registres du pipeline pour les masques.

La mise en œuvre du processus de correction du masque est un peu plus complexe suivant le type d'instruction. Pour les instructions arithmétiques et logiques, les données masquées sont d'abord transférées d'étage en étage sans perte d'intégrité des données et ce jusqu'à la phase exécution, où toutes les opérations mathématiques sont mises en œuvre. La technique de « *masking* » implique la mise en œuvre de tables de correspondances pré-calculées spécifiques pour chaque opération. Compte-tenu du nombre important d'instructions et de leur relative complexité, la mise en œuvre du *masking* au niveau de l'ALU serait trop coûteuse, raison pour laquelle les données sont démasquées avant, puis re-masquées après le calcul avec un nouveau nombre pseudo-aléatoire. Ce choix est également motivé par le fait que, même si les attaques fonctionnent sur la logique combinatoire, les résultats expérimentaux montrent que les fuites au niveau des registres sont plus importantes. Bien que cette solution ne soit pas parfaite, elle a l'avantage de protéger les parties les plus sensibles du chemin de données, tout en réalisant un compromis intéressant entre performance et sécurité.

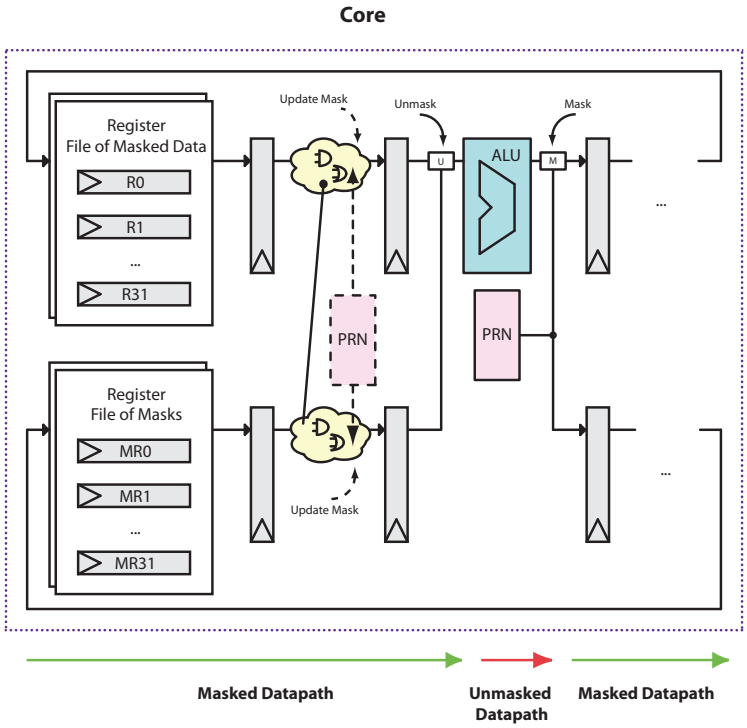



**Figure 70.** Masquage du pipeline

Pour la deuxième contremesure, notre idée repose essentiellement sur le concept de processeur non-déterministe **[50]**, dans lequel le logiciel est exécuté avec insertion aléatoire d'opérations supplémentaires, qui sont générées directement par l'architecture matérielle du processeur. Il s'agit d'une part de cycles factices qui changent le temps d'exécution, mais aussi d'une randomisation de l'utilisation des ressources matérielles disponibles, afin d'accroître le "bruit". La contre-mesure affecte donc à la fois les dimensions temporelles et spatiales des fuites physiques.

Pour sa mise en œuvre, une solution consiste à randomiser le contrôle du pipeline RISC par des signaux de commande et de données factices. Dans cette structure, un PRNG génère un nombre aléatoire utilisé pour prendre une décision pour chaque instruction en cours de traitement, qui consiste soit à conserver une exécution normale, soit effectuer un cycle factice. Des PRNG supplémentaires sont également mis en œuvre pour générer des nombres aléatoires à l'entrée des opérateurs de l'ALU au cours des cycles factices, ce qui augmente l'activité aléatoire du circuit. La plupart des architectures RISC de processeurs embarqués (tels que ARM, MIPS et PowerPC), disposent d'un mécanisme de verrouillage du pipeline, dont le rôle est de détecter et résoudre les aléas de contrôle et de données. Ce contrôleur d'aléas gère ainsi l'état de chaque étage du pipeline à travers l'ensemble de l'architecture.

Les techniques courantes pour résoudre ces aléas consistent à suspendre l'exécution *pipelinée* (*stall* et insertion de bulles) ou vider le pipeline (*flush*). Pour réaliser une exécution non-déterministe, nous pouvons profiter des fonctionnalités matérielles du contrôleur d'aléas (signaux de *stall* et *flush*) pour insérer les cycles factices le long du chemin de données. Ce choix de conception réduit considérablement les coûts de la contre-mesure (en termes de surface, de consommation et de performances). En ajoutant des aléas factices, le pipeline fonctionne alors de manière inattendue que l'on peut efficacement exploiter pour insérer des cycles de calculs supplémentaires. Nous l'avons appelé : génération d'aléas fantômes.

Pour que cette technique soit efficace, les aléas fantômes doivent être générés de façon aléatoire. En outre, plus le nombre de cycles factices est grand, plus la contre-mesure est efficace. Néanmoins, une utilisation excessive de cycles aléatoires peut sérieusement diminuer le débit de l'architecture. Dans la pratique, un compromis acceptable doit être trouvé. Pour répondre aux exigences de l'application de l'utilisateur, la probabilité des aléas fantômes doit être configurable. Notre solution consiste à mettre en œuvre un module qui compare une valeur de seuil définie dans un registre de commande avec un

PRN généré à chaque cycle d'horloge. Le résultat détermine s'il y a génération d'aléa fantôme ou pas. L'architecture de notre dispositif de commande des aléas fantômes est illustrée sur la **Figure 71**.

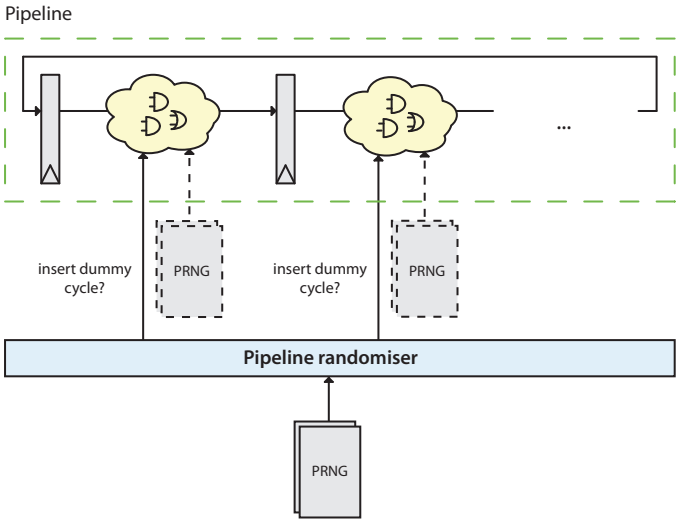



**Figure 71.** Randomisation du pipeline

### 3.4.4 Résultats

Comme nous pouvons l'observer sur la **Figure 72**, la stratégie de masquage réduit l'efficacité de l'attaque, puisqu'au lieu d'une propagation le long des étages du pipeline, seule la fuite de l'ALU reste exploitable. D'autre part, la contremesure de *hiding* configurée avec un faible tirage aléatoire fournit un certain niveau de protection, mais le grand nombre d'opérations critiques liées à l'architecture du pipeline permet à l'attaque de fonctionner néanmoins. Alors que 1000 traces sont suffisantes pour retrouver la clé de chiffrement pour la version non protégée du *SecretBlaze*, notre évaluation expérimentale montre que 200.000 traces ne permettent pas de mettre en péril le *SecretBlaze SCR* combinant les contremesures de *masking* et de *hiding*.

Les deux contremesures ont été implémentées au sein du processeur SecretBlaze et leur impact a été évalué afin de pouvoir comparer notamment le coût en surface et en performance par rapport aux techniques référencées dans la littérature. Les résultats correspondent à une technologie Spartan-3 de Xilinx. Comme le résume le **Tableau 10**, la technique de *masking* est celle dont le coût en ressources est le plus élevé et qui de plus diminue assez significativement la fréquence maximale (de manière équivalente aux travaux cités). Mais par rapport à l'état de l'art, la pénalité en surface est cependant beaucoup plus limitée, compte-tenu du fait que le masquage n'affecte pas l'ALU (ce qui aurait nécessitait beaucoup plus de ressources). En ce qui concerne la gestion des aléas fantômes, le coût est presque négligeable, tant au niveau des LUT et des Flip-flops que pour la fréquence maximale. Cela nous permet de conclure que cette combinaison de contremesures permet d'obtenir un niveau de sécurité démontré par nos expériences et ce qui est tout à fait remarquable, avec un très faible coût matériel.

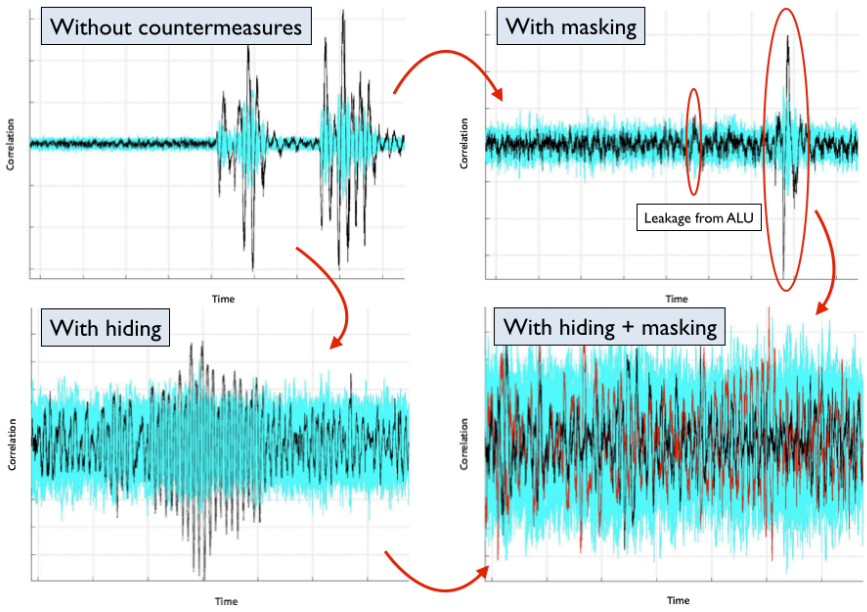



**Figure 72.** Traces DEMA avec et sans contremesures

**Tableau 10.** Evaluation du coût des contremesures

| | Masked Datapath | Ghost Hazard Mechanism | Both |
|---|---|---|---|
| **# Flip-Flops** | +37.6% | +1.4% | +39.0% |
| **# LUTs** | +39.9% | +9.5% | +48.9% |
| **# BRAMs** | +22.2% | 0.0% | +22.2% |
| **fMAX in MHz** | −18.8% | −2.9% | −18.9% |

## 3.5 Bilan

Dans cette dernière partie concernant la synthèse des travaux, nous avons exposé nos travaux relatifs à la conception de systèmes adaptatifs dits de confiance. Sur la base des travaux réalisés sur l'architecture HS-Scale, nous avons développé une approche originale permettant d'assurer la disponibilité des fonctionnalités du système en mode dégradé, garantissant ainsi une certaine tolérance aux pannes du système. Sur le thème de la fiabilité, nous nous sommes intéressés à des méthodes caractérisation des paramètres PVT dans les circuits FPGA. Nous avons pour cela développé des capteurs et une approche nouvelle assurant une extraction fiable des informations de performance, basée sur une analyse des émanations électromagnétiques : celle-ci a d'ailleurs été exploitée pour des expérimentations concernant l'étude du vieillissement de ces circuits. Afin de pouvoir gérer les mécanismes d'adaptation se basant sur une estimation réaliste de l'état du système, nous avons proposé plusieurs techniques de monitoring de la consommation. Enfin, le domaine de la sécurité a été adressé à travers la conception d'un processeur résistant aux attaques par canaux cachés. Le travail présenté dans cette section a été valorisé à travers plusieurs publications, dont :

- 3 revues, dont 3 RICL: [J1][J2][J3] (ces trois articles sont joints en annexe de ce document)
- 3 chapitres d'ouvrage : [CO2] [CO3] [CO4]
- 19 CICL : [C2] [C3] [C4] [C5] [C7] [C8] [C9] [C10] [C12] [C14] [C15] [C17] [C20] [C21] [C22] [C27] [C28] [C31] [C43]

# 4 BILAN ET TRAVAUX EN COURS

A travers ces différentes sections, nous avons fait la synthèse d'un ensemble de contributions autour de mes 3 axes principaux de recherche que sont la conception d'architectures, l'optimisation dynamique des systèmes adaptifs, et les systèmes intégrés de confiance. Un point important à souligner est la complémentarité de l'ensemble de ces travaux, soutenue par une vision de la conception des circuits intégrés élaborée dès le début de ma thèse. De manière générale, celle-ci repose sur une exploitation dynamique des ressources, que ce soit pour accroître les performances, réduire la consommation, ou encore améliorer la fiabilité. Les termes qualificatifs ont changé au fil du temps et de l'évolution de la complexité des approches : reconfiguration, reconfiguration dynamique, adaptation, auto-adaptation. Le terme générique ultime serait sans doute celui de « systèmes intégrés autonomiques », qui regrouperait les notions :

- d'auto-configuration
- d'auto-optimisation
- d'auto-réparation
- d'auto-protection

Ce manuscrit est une photographie d'un domaine en mutation constante et rapide, porté par une loi de Moore toujours d'actualité, et l'émergence de nouvelles technologies. Aussi j'ai décidé d'exposer une partie des travaux en cours et des perspectives à court terme, qui posera les fondements de mon projet de recherche.

## 4.1 Supervision optimale

L'autonomie (au sens des *systèmes autonomiques*) ne peut être atteinte sans une connaissance précise de l'état du système. Cette problématique a été adressée dans les travaux de Florent Bruguier par la mise en œuvre de capteurs bas niveau (oscillateurs en anneau) sensibles au Process, à la Tension et à la Température, puis prolongée dans les travaux d'Imen Mansouri à un niveau plus abstrait par utilisation de compteurs d'évènements. Nous avons vu comment nous pouvions exploiter efficacement des méthodes statistiques pour sélectionner un sous-ensemble de signaux à observer, ainsi que l'extraction de modes de consommation par une modélisation à base de Chaînes de Markov à états cachés pour optimiser encore plus loin le dispositif d'observation.

La supervision est un processus complexe qui nécessite la prise en compte de multiples variables d'état par une instrumentation intelligente du système. L'objectif de ce travail en cours est de pouvoir déterminer des modèles prédictifs précis et faible coût. Nous focalisons l'étude sur l'estimation de la consommation et de la température, qui sont les deux critères que l'on cherche le plus souvent à optimiser avec la performance. Cette estimation présente en plus une interdépendance, puisque il existe une relation exponentielle entre la puissance statique et la température.

Les modèles de consommation et de température s'obtiennent par un flot de simulation et/ou prototypage du système, qui permet de construire des bases de données dans lesquelles nous associons à chaque instance (à un temps donné) une valeur de la variable à expliquer (température ou consommation). Le flot de conception mis en œuvre pour construire ces bases de données est représenté sur la **Figure 73**. Celui-ci fait intervenir les outils de conception (synthèse, placement-routage et simulation) ainsi qu'un outil de simulation de la température du circuit à partir de la distribution de la puissance consommée (*HotSpot*).

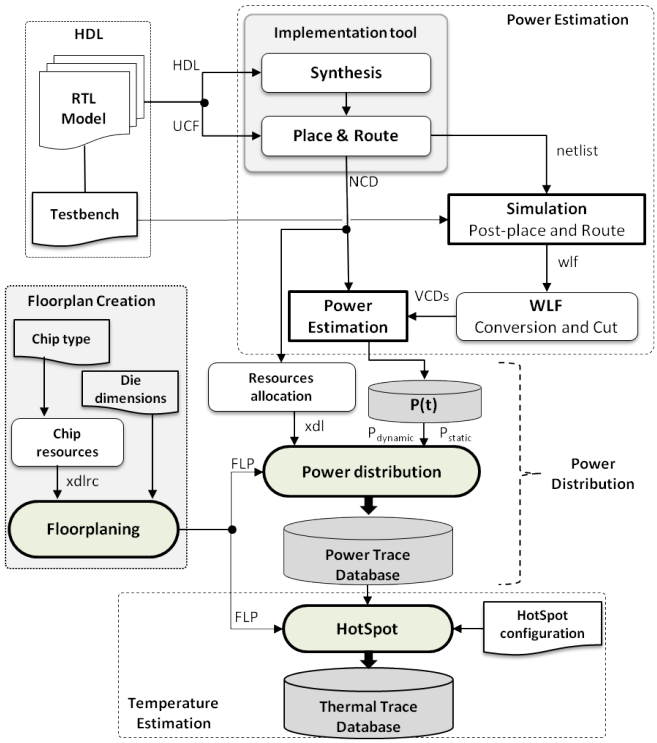



**Figure 73.** Flot de génération des bases de données de puissance et température

Nous expérimentons une méthode d'apprentissage non-supervisée basée sur une segmentation k-moyenne, par laquelle nous pouvons identifier des zones d'intérêt présentant des comportements thermiques similaires. Il est ainsi possible à partir d'une régression linéaire de créer un modèle de la température avec un nombre réduit de capteurs : le placement peut-être déduit précisément à partir de la technique de *clustering*. La **Figure 74** illustre le résultat de cette segmentation avec le placement des capteurs en conséquence.

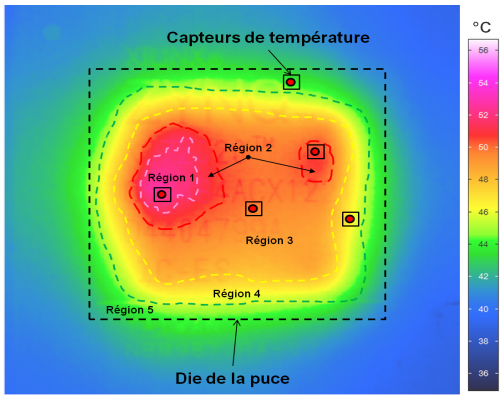



**Figure 74.** Image infra-rouge obtenue par une caméra thermique et placement des capteurs de température

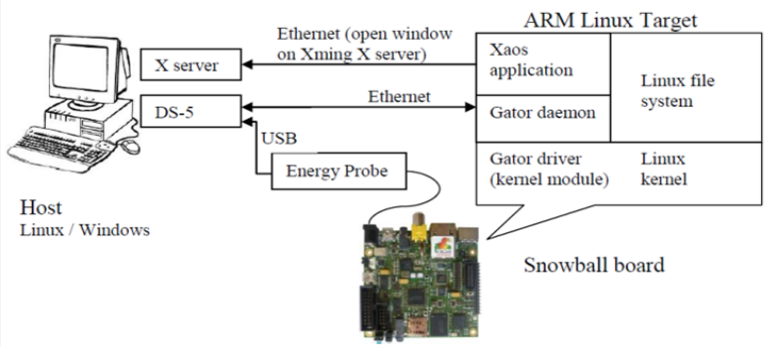



**Figure 75.** Méthode de prototypage avec un processeur ARM

Nous travaillons également sur une méthode permettant de construire des modèles de consommation et de température à partir de PMU (Performance Monitoring Unit), qui sont des dispositifs intégrés dans les processeurs ARM, permettant de relever des activités sur un ensemble de signaux configurables. En exécutant un certain nombre de benchmarks logiciels, nous relevons sur un PC externe (**Figure 75**) des données relatives à ces évènements, à la puissance consommée (via le Energy Probe) et la température (via une caméra thermique). Les bases de données sont ensuite analysées avec un outil de Data Mining (Weka). Nous développons des algorithmes permettant de sélectionner les signaux les plus pertinents dans la construction de modèles de consommation. Ensuite, c'est par une méthode d'apprentissage supervisé (réseaux de neurones) que nous estimons la consommation. Les résultats obtenus (**Figure 76**) montrent une grande précision de l'estimation (99%) avec une très faible pénalité en performance.

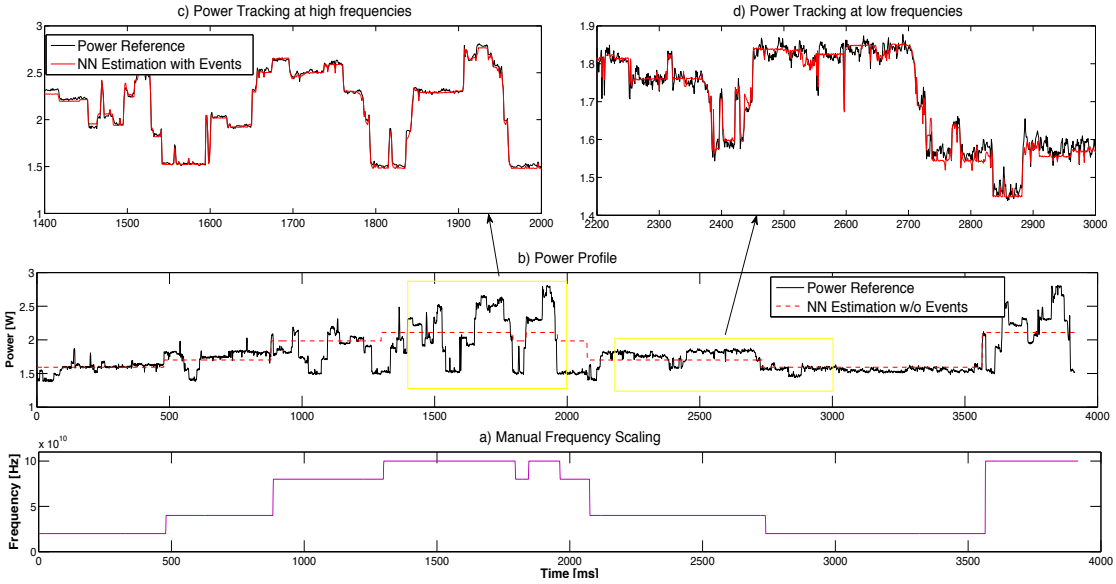
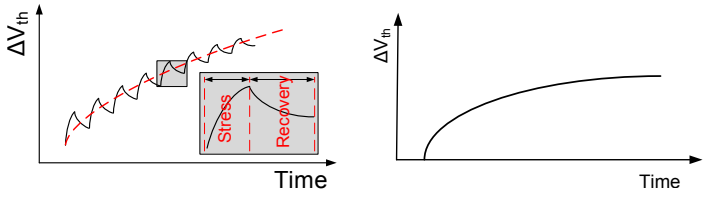



**Figure 76.** Estimation en ligne de la consommation du processeur ARM

A partir de ces informations, il est intéressant de voir dans quelle mesure nous pouvons inférer des modèles précis de la consommation et de la température (et notamment des *hotspots*) à partir d'un très petit nombre de signaux. Nous étudions également l'impact des perturbations environnementales sur la précision des estimations, afin de pouvoir intégrer cette dimension dans nos modèles et disposer ainsi d'une précision garantie dans toutes les circonstances.

## 4.2 Modélisation du vieillissement

Les performances d'un circuit évoluent au cours du temps en fonction d'un ensemble de phénomènes qui entrainent la dégradation des transistors et des interconnexions au sein des circuits intégrés. Dans la littérature, on recense un certain nombre de mécanismes tels que le BTI (Bias Temperature Instability) qui affecte les transistors de type N (PBTI) ou de type P (NBTI) lorsque ceux-ci sont actifs (rapport cyclique ON/OFF). Le HCI (Hot Carrier Injection) est lui aussi impliqué dans la dégradation des performances des transistors, et dépend du taux d'activité. Le BTI et le HCI ont un impact sur les tensions de seuil des transistors, qui augmentent progressivement et ralentissent donc le fonctionnement des transistors (augmentation des délais) jusqu'à provoquer des fautes temporelles. Le TDDB (Time Dependent Dielectric Breakdown) s'observe lui par la rupture de l'isolation entre la grille et le canal des transistors, et l'électro-migration (EM) s'applique aux interconnexions métalliques, qui peuvent au cours du temps se dégrader elles-aussi (défauts pouvant aller jusqu'à une rupture de piste ou création de court-circuit). Nous nous intéressons plus particulièrement au BTI et HCI (**Figure 77**), dont nous essayons de mesurer l'impact, afin de définir des modèles de vieillissement qui seraient exploitables en cours de fonctionnement, afin de mettre en place des stratégies (gestion des couples (triplets) tension(s)/fréquence, placement de tâches, etc.) permettant de mieux maîtriser la dégradation des transistors. Ces mécanismes de vieillissement dépendent de plusieurs facteurs répertoriés dans le tableau suivant (T est la température, V la tension d'alimentation, alpha le taux d'activité des transistors et Y le rapport cyclique des transistors). A noter que le BTI présente une phase de récupération relative aux instants où les transistors sont OFF tel que cela peut-être observé sur la figure 78 (a).




| Factor | $T$ | $Vdd$ | $\alpha$ | $Y$ |
|---|---|---|---|---|
| BTI | exponential | exponential | - | sub-linear |
| HCI | exponential | exponential | sub-linear | - |

**Figure 77.** Allure de l'impact du BTI et du HCI en fonction du temps, et relation de ces mécanismes avec différents facteurs

Le modèle du BTI au niveau transistor est représenté par l'équation suivante **[51]** :

$$\Delta d = A_{BTI2} \times Y^n \times t^n \times e^{\left(\frac{-E_a}{kT}\right)} \times d_0$$

Où $d_0$ est le délai à t=0, $E_a$ est un paramètre d'ajustement, $k$ la constante de Boltzmann, $T$ la température, $Y$ le rapport cyclique, $n$ un paramètre qui dépend de la technologie (compris entre 0 et 1), t le temps, $A_{BTI2}$ facteur qui dépend de paramètres technologiques et de la tension. Δd est l'augmentation du délai.

De la même manière, on peut exprimer le modèle HCI au niveau transistor par l'équation suivante :

$$\Delta d = A_{HCI} \times \alpha \times f \times t^{0.5} \times e^{\left(\frac{-E_b}{kT}\right)} \times d_0$$

Où $\alpha$ représente l'activité, $f$ la fréquence de fonctionnement, et $A_{HCI}$ est un facteur dépendant de la technologie et de la tension d'alimentation.

Pour évaluer les conséquences des différents phénomènes en jeu, il faut pouvoir fixer précisément les facteurs technologiques, mais aussi l'activité, le rapport cyclique pour chaque transistor, la température (qui dépend de l'environnement, mais aussi de la consommation donc de l'activité), ce qui

nécessite à l'échelle du système des simulations longues et complexes à mettre en œuvre. Une autre méthode consiste à faire une évaluation directement sur prototype, en effectuant un vieillissement accéléré du composant. C'est ce que nous expérimentons sur des circuits FPGA en collaboration avec le CDNC au KIT. Nous avons réalisé plusieurs études au cours desquelles nous avons placé le composant dans des conditions de stress (alimenté à une tension 50% plus élevée que la tension nominale, et une température à 80°C dans une enceinte thermique). Le procédé de caractérisation (**Figure 78**) développé dans le cadre de la thèse de Florent Bruguier est utilisé afin de mesurer les fréquences des oscillateurs utilisés pour générer de l'activité interne au circuit.

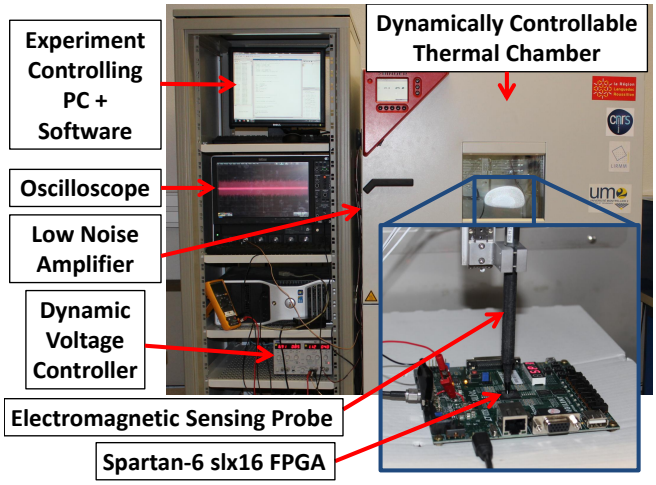



**Figure 78.** Equipement mis en œuvre dans le processus de vieillissement

Pour déterminer avec exactitude l'influence des paramètres tels que le taux activité et le rapport cyclique, il faut pouvoir comparer différentes durées de rapports ON/OFF des transistors et différentes fréquences. Notre première expérience utilise pour cela les quatre structures logiques représentées sur la **Figure 79**. Un signal de contrôle (*enable*) permet de jouer sur les rapports cycliques (5% ou 95%) et par le nombre d'étages des oscillateurs en anneau on obtient des circuits commutant à différentes fréquences (350MHz et 900MHz pour un FPGA Spartan 6 en technologie 45nm). La mise en œuvre des quatre combinaisons telles qu'elles sont représentées sur la **Figure 79** nous permet de comparer l'impact de l'activité (SA) et du rapport cyclique (SP). Afin de s'affranchir des variations dues à la position des structures (en termes d'auto-échauffement local ou de *process*), ces structures sont répliquées plusieurs fois (40 fois pour chaque oscillateur) et sont placées entrelacées (160 positions au total) sur l'ensemble du circuit.

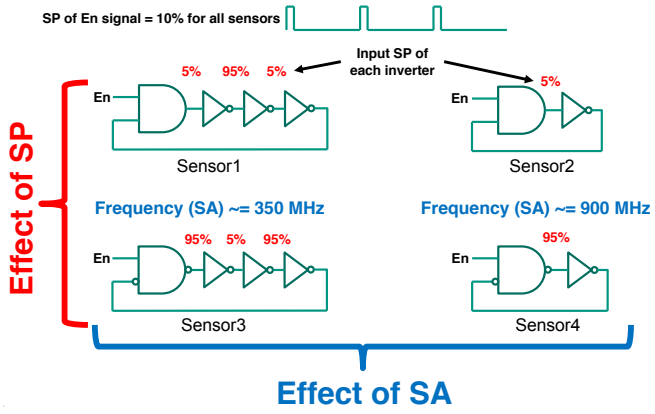
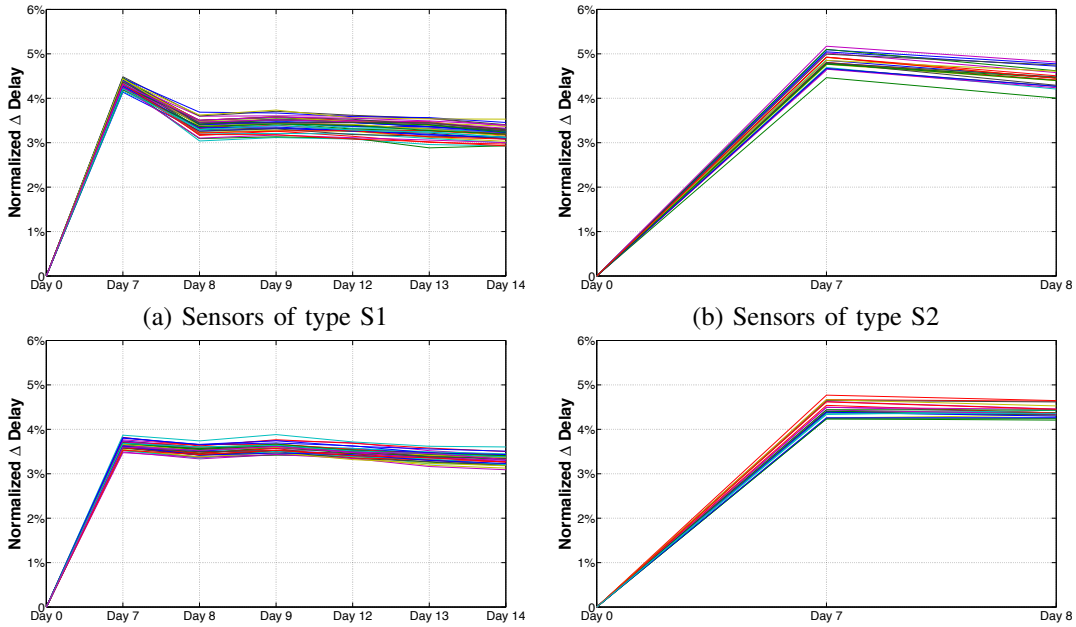



**Figure 79.** Structures logiques utilisées pour analyser l'impact du rapport cyclique (SP) et de l'activité (SA)

Une première analyse des délais est effectuée à température et tension nominale (T=25°C et V=1,2V) à t=0 : elle permet de relever les fréquences de fonctionnement de chaque structure d'oscillateur. Après 7 jours de vieillissement accéléré (80°C, 1,8V), et retour aux conditions nominales, le circuit est à nouveau caractérisé, et ce pendant 3 jours afin d'observer les éventuels récupérations de performance. Les différentes valeurs relevées pour les 4 structures de capteurs sont représentées sur la **Figure 80**. Ces résultats mettent clairement en évidence l'influence l'impact du BTI et HCI, ainsi que l'importance de chacun des paramètres puisque un vieillissement de l'ordre de 4 à 5% est relevé. La récupération supposée du BTI est aussi vérifiée nettement pour la structure S1.




(a) Sensors of type S1




(b) Sensors of type S2




(c) Sensors of type S3




(d) Sensors of type S4

**Figure 80.** Résultats (delta délai normalisé) obtenus pour les 4 structures de capteurs

Ce travail développé dans **[40]** ouvre des pistes intéressantes de travaux de recherche, notamment pour quantifier précisément l'impact de chacun des facteurs, étudier la sensibilité de certaines configurations logiques par rapport à d'autres, la mise en place de techniques adéquates de monitoring et de réduction des effets de vieillissement. Une perspective d'application de ces travaux concerne leur utilisation pour la caractérisation des fonctions physiques non-clonables (PUF) dans un contexte de sécurité.

# 5 PROJET DE RECHERCHE

## 5.1 Supervision prédictive

Mon objectif à moyen terme est de poursuivre l'effort de recherche que j'ai engagé dans la conception de systèmes intégrés autonomes, en me focalisant sur les processus de supervision et de contrôle. Nous sommes sur le point de pouvoir faire des estimations précises d'un ensemble de variables d'état, telles que la consommation et la température, et ce à partir d'un faible nombre de capteurs embarqués. Mais l'interdépendance des variables, à plusieurs échelles (spatiale et temporelle) reste à approfondir, puisque par exemple les politiques d'optimisation ne tiennent pas compte actuellement de la dégradation des performances, ce qui fait le lien avec notre étude sur le vieillissement. Au delà de ces dépendances multiples, le processus de supervision repose sur une analyse réactive, c'est à dire que lorsqu'une estimation est faite, elle représente l'état à l'instant présent. Or le mécanisme de prise de décision, pour être tout à fait pertinent, devrait pouvoir se baser sur une analyse prédictive qui lui permettrait d'évaluer les gains potentiels suivant différentes options de paramétrage de ses actionneurs.

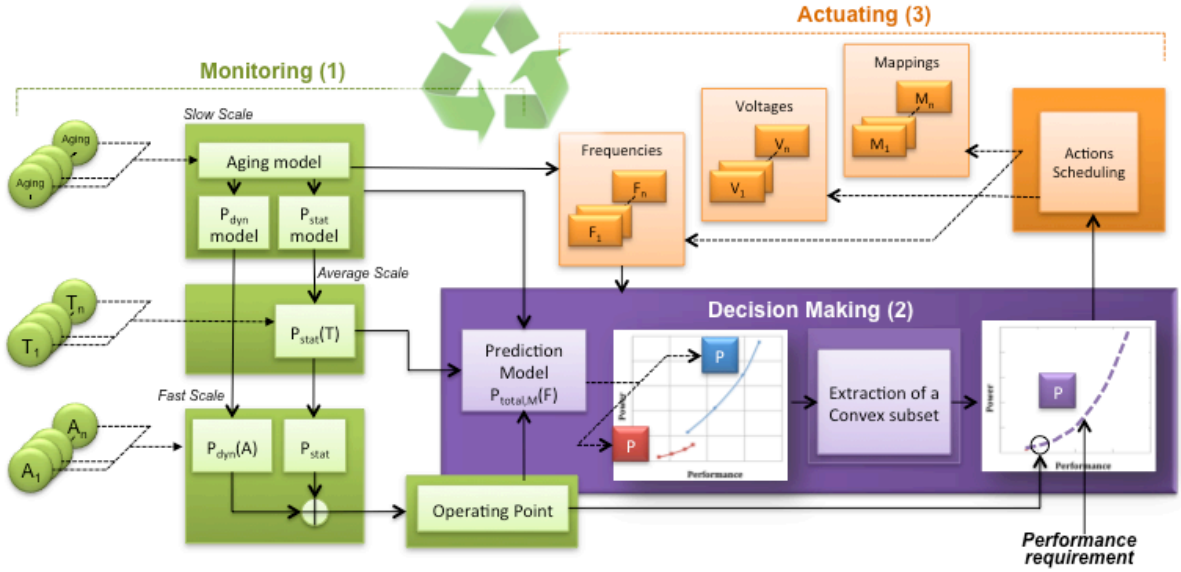



**Figure 81.** Infrastructure de supervision et contrôle d'un système autonome intégré

Le schéma de principe de la **Figure 81** illustre le processus complet incluant l'analyse prédictive. Cette illustration suppose un système intégré doté d'un ensemble de capteurs physiques lui permettant de mesurer des données relatives à l'activité, la température, les délais des chemins critiques, etc. Le processus de monitoring doit disposer d'un ensemble de modèles intégrés lui permettant de déduire de ces mesures des informations qui font sens au niveau système, par exemple la consommation totale (qui dépend de la consommation dynamique et de la consommation statique), des points chauds, du vieillissement cumulé de certaines zones du circuits, etc. Cette estimation de l'état présent est le point d'entrée d'un modèle prédictif, qui doit être en mesure de définir l'ensemble des points de fonctionnement accessibles à partir de l'état courant. Ce processus doit même permettre d'aller au delà, puisqu'il doit être en mesure de quantifier ces états futurs potentiels. Cela revient à calculer des modèles qui sont fonction des paramètres ajustables du système (*mapping* de tâches, tensions d'alimentation, de substrat, fréquences des PE). Après extraction de l'enveloppe convexe de ces fonctions (les autres points de fonctionnement sont exclus), le point de fonctionnement futur optimal peut être ainsi évalué et le système peut être configuré en conséquence. Un planificateur des actions traduit le résultat du processus de décision en un ensemble d'étapes cohérentes permettant de modifier les consignes des actionneurs.

L'implémentation de cette approche requiert un travail très important en temps de conception, qui doit permettre par des simulations multi-échelles d'extraire des figures de performance du système. L'analyse basée sur une fouille des données telles que nous l'avons élaborée dans nos travaux récents, sera la première étape de définition des modèles descriptifs de l'état du circuit. Nous étudierons ensuite l'application de méthodes d'apprentissage et les règles d'association pour évaluer les mécanismes les plus efficaces pour intégrer des modèles prédictifs. Enfin, c'est la mise en œuvre à travers une approche distribuée qui sera expérimentée afin de démontrer ce concept.

Ce projet réalise la synthèse des travaux sur l'optimisation et le monitoring, en y ajoutant une dimension nouvelle relative à l'objectif de supervision prédictive. Jusqu'à présent, nous avons essentiellement considéré le problème des systèmes intégrés autonomes à l'échelle du circuit : le processus de décision s'appuie sur un monitoring qui tient compte essentiellement des ressources internes, voire de l'environnement du circuit. Si l'on considère que dans un futur proche tous les objets seront capables de communiquer, cela pose des questions quant à la prise en compte de cette nouvelle donnée dans notre vision de la conception des circuits intégrés. Cela suppose des interactions complexes avec l'environnement, constitué du monde physique et du monde numérique à travers différents canaux de communication, ce qui ouvre de nouvelles perspectives à plus long terme où la notion de réseau doit être intégrée à la problématique.

## 5.2 Passerelles pour l'Internet des Objets

Mon projet à plus long terme s'inscrit dans le contexte de l'internet des objets (IoT). La vision développée ici s'appelle TrustNet, un réseau cognitif décentralisé, où les composants collaborent pour interpréter et sécuriser les flux de données qu'ils gèrent. Cette approche déporte une partie des traitements des données des *data centers* vers ces interfaces intelligentes, pour réduire les temps de réponse pour des applications temps-réel et améliorer l'efficacité énergétique des quelques 200 milliards d'objets qui produiront plus de 40000 milliards de Go en 2020.

Le projet a pour objectif l'étude et la conception d'un système numérique autonomique sécurisé comme technologie clé pour la mise en œuvre de passerelles interopérables pour l'IoT. Les objets dits « intelligents » ont la capacité d'interagir avec leur environnement, de communiquer avec d'autres objets. Au delà des applications de l'électronique grand-public, ces objets offrent potentiellement de nombreuses opportunités, comme la gestion plus efficace des ressources, de l'énergie, des transports, des catastrophes naturelles, ou encore pour des usages innovants autour de l'e-santé, de la culture et de l'éducation. Mais il ne faut pas sous-estimer les menaces sous-jacentes à une connectivité toujours plus grande : qu'il s'agisse de confidentialité, d'intégrité, de disponibilité, la sécurité numérique doit pouvoir être garantie de bout en bout.

L'écosystème structurel de ce qu'on appelle généralement l'internet des objets comprend trois éléments essentiels : les objets intelligents d'une part, Internet et le *cloud* d'autre part, et entre les deux, ces passerelles qui font le lien entre les données produites et les *data centers*, où sont stockées et analysées ces données. Ce schéma simplifié est en fait adapté au domaine d'application, en fonction notamment des contraintes de performance, de sûreté de fonctionnement ou de sécurité. Si les applications grand public reposent généralement sur des infrastructures génériques (par exemple le *smartphone* ou une box ADSL peuvent faire office de passerelle entre l'objet intelligent et le *cloud*), il n'en va pas de même pour des applications plus spécifiques qui nécessitent une qualité de service garantie par l'utilisation de passerelles dédiées. Au delà de la notion de sécurité, c'est l'impact environnemental de ces solutions numériques qui doit être intégré dès la phase de conception. Au delà même de la passerelle numérique, c'est tout l'écosystème qui doit être pensé de façon cohérente, notamment dans le stockage et le transport des données numériques dont le coût énergétique ne cessera de croître avec l'émergence de milliards d'objets connectés.

La conception d'une passerelle numérique intelligente doit répondre à un certain nombre d'exigences, en particulier :

- l'**interopérabilité** : le système doit fournir des services de connectivité locale, permettant aux objets intelligents de fonctionner ensemble, à travers différents supports et protocoles de communications,
- la **sécurité** : le système doit garantir l'intégrité et la confidentialité des données, de même que la disponibilité de l'équipement
- la **basse consommation** : la consommation d'énergie doit-être minimale pour une tâche donnée, de même que la gestion des données doit être pensée dans le but de réduire le coût associé au transport de ces données

Se pose ensuite le problème du **support matériel et logiciel** de cette passerelle. Une solution à base de plateforme programmable générique, type PC embarqué présente de grandes faiblesses d'un point de vue sécurité du fait de sa généricité. Une approche alternative à base de composants reprogrammables semble être l'alternative possible à une architecture spécifique, qui offrirait une sécurité accrueau détriment d'une évolutivité très limitée, d'une obsolescence rapide et d'un coût très élevé.

La **Figure 82** ci-dessous illustre la vision générale de TrustNet. Un ensemble d'objets, tels que des capteurs intelligents, échangent localement des données grâce à l'infrastructure de service offerte par des passerelles intelligentes. Ces Smart Gateway sont des plateformes collaboratives assurant l'interopérabilité, l'analyse, voire l'interprétation des flux de données, déportant ainsi une partie des traitements critiques des data center vers le réseau local. Ces nœuds assurent un lien fortement sécurisé vers le réseau Internet, permettant ainsi de rationaliser les échanges numériques : seule l'information utile est accessible à travers des canaux chiffrés, réduisant drastiquement la quantité de bits transmis sur le réseau global. TrustNet forme ainsi un réseau sémantique partitionné. Cette vision qui repose énormément sur des mécanismes de supervision prédictive, apporte une solution cohérente au problème de la sécurisation des espaces numériques sensibles et de l'amélioration de l'efficacité énergétique.

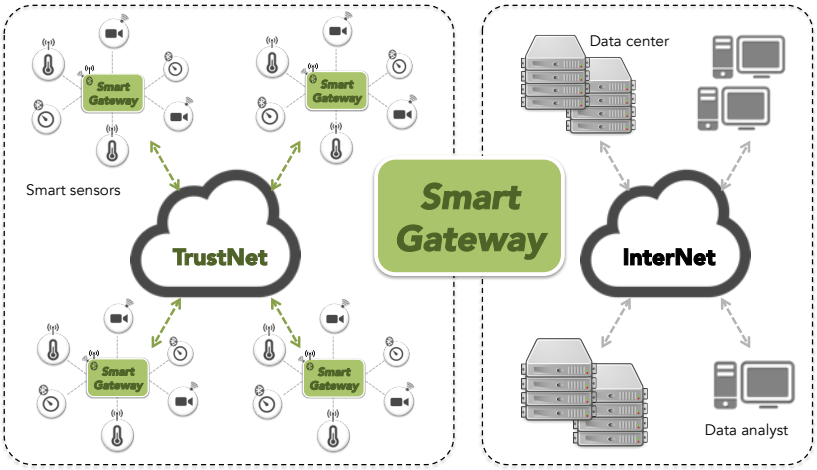



**Figure 82.** Positionnement du projet de recherche

L'architecture envisagée dans cette approche est décrite dans la **Figure 83**. Elle repose sur 3 piliers : l'interface « TrustNet » (qui assure un service sécurisé d'interopérabilité locale, ainsi qu'une analyse intégrée des données), l'interface « Internet » (qui garantit une connectivité sécurisée au réseau Internet), et le « system management » (qui gère l'ensemble des éléments de la passerelle, et permet une collaboration des Smart Gateway pour un transport des données optimal du point de vue de l'efficacité énergétique). La structuration de cette approche est une première originalité de ce projet, complétée par des contributions pour chacun des trois éléments clés, qui feront l'objet de recherches et de développements spécifiques. Nous étudierons l'applicabilité d'approches d'optimisation convexe à travers une approche collaborative consensuelle, permettant de garantir une convergence asymptotique du processus de minimisation de l'énergie entre plateformes. L'analyse des données sera au cœur des

deux interfaces réseau et constituera par son implémentation matérielle en une approche nouvelle par rapport à l'état de l'art. Ce sont des algorithmes de classification non-supervisée (segmentation k-moyennes entre autres) qui seront envisagés pour identifier des sous-espaces sémantiques dans les données « capteurs ». Des méthodes de filtrage adaptatif mettant en œuvre des techniques d'apprentissage (une solution à base de réseaux de neurones par exemple) seront étudiées en vue d'une intégration matérielle des mécanismes de sécurité pour détecter, dans les flux de données, les tentatives d'intrusions sur le réseau TrustNet.

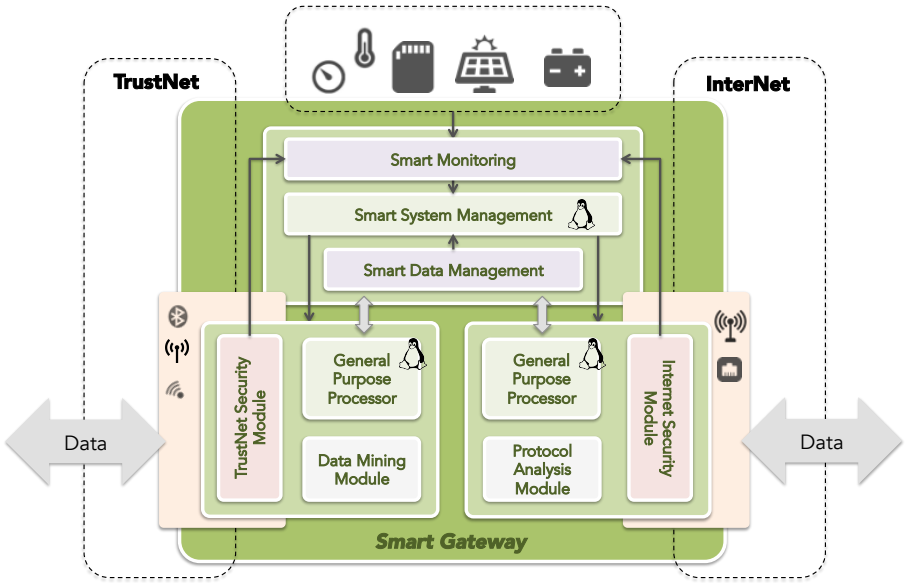



**Figure 83.** Architecture de la Smart Gateway

Ce projet s'inscrit dans la continuité de mes travaux appliqués à un domaine porteur. L'intégration de la notion de connectivité à l'échelle des composants implique un changement de paradigme qui justifie les efforts de recherche que je souhaite mener dans cette direction.

# 6 REFERENCES


[1] G. Sassatelli, L. Torres, P. Benoit, T. Gil, C. Diou, G. Cambon, J. Galy, "Highly Scalable Dynamically Reconfigurable Systolic Ring-Architecture for DSP Applications", *DATE*, 2002, Design, Automation & Test in Europe Conference & Exhibition, Design, Automation & Test in Europe Conference & Exhibition 2002, pp. 0553

[2] Pascal Benoit, Gilles Sassatelli, Lionel Torres, Michel Robert, G. Cambon, et al.. Méthode de caractérisation des architectures d'accélérateurs flexibles pour systèmes sur puce. *TSI : Revue Technique et Science Informatiques*, 2005, 24 (6), pp.725-755

[3] Mark D. Hill, Michael R. Marty, "Amdahl's Law in the Multicore Era," Computer, vol. 41, no. 7, pp. 33-38, July, 2008

[4] Pascal Benoit, Lionel Torres, Gilles Sassatelli, Michel Robert, G. Cambon. Automatic Task Scheduling/Loop Unrolling Using Dedicated RTR Controllers in Coarse Grain Recongigurable Architectures. *IPDPS'05: International Parallel and Distributed Processing Symposium, RAW'05: Reconfigurable Architectures Workshop*, Apr 2005, Denver, Colorado (USA), IEEE, pp.148, 2005

[5] P. Benoit, L. Torres, G. Sassatelli, M. Robert, N. Saint-Jean, *"Run Time Mapping for Dynamic Reconfiguration Management in Embedded Systems"*, International Journal of Embedded Systems (IJES), Inderscience, Volume 4 – Issue 3/4 – 2010, pp. 276-291

[6] Fernando Moraes, Ney Calazans, Aline Mello, Leandro Möller, Luciano Ost, HERMES: an infrastructure for low area overhead packet-switching networks on chip, Integration, the VLSI Journal, Volume 38, Issue 1, October 2004, Pages 69-93, ISSN 0167-9260

[7] G. Almeida, Saint-Jean Nicolas, Varyani Sameer, G. Sassatelli, P. Benoit, L. Torres, *"An Adaptive Message Passing MPSoC Framework"*, Hindawi International Journal of Reconfigurable Computing, Volume 2009, 242981 (2009) 18, pp. 1-18

[8] D. Pulley, "Multi-core DSP for Base stations: Large and Small," *Design Automation Conference (ASPDAC'08)*, pp. 389-391, March 2008

[9] U. Ramacher. Software-defined radio prospects for multistandard mobile phones. Computer, 40(10) :62 –69, oct. 2007

[10] Glossner *et al.*, "The sandbridge sb3011 platform," *EURASIP J. Embedded Syst.*, vol. 2007, no. 1, pp. 16-16, 2007

[11] F. Clermidy, R. Lemaire, X. Popon, D. Kténas and Y. Thonnart, "An Open and Reconfigurable Platform for 4G Telecommunication: Concepts and Application," *Euromicro Conference on Digital System Design (DSD)*, Aug. 2009

[12] JALIER, Camille, LATTARD, Didier, JERRAYA, Ahmed Amine, *et al.* Heterogeneous vs homogeneous MPSoC approaches for a mobile LTE modem. In : *Proceedings of the Conference on Design, Automation and Test in Europe*. European Design and Automation Association, 2010. p. 184-189.

[13] FAURE, Etienne, ALMEIDA, Gabriel Marchesan, BENABDENBI, Mounir, *et al.* An in-memory monitoring database for self adaptive MP 2 SoCs. In : *Design and Architectures for Signal and Image Processing (DASIP), 2010 Conference on*. IEEE, 2010. p. 97-104.

[14] C. Roth, G. M. Almeida, O. Sander, L. Ost, N. Hebert, G. Sassatelli, P. Benoit, L. Torres and J. Becker, *"Modular Framework for Multi-Level Multi-Device MPSoC Simulation"*, 25th IEEE International Parallel & Distributed Processing Symposium, May 16-20, 2011, Anchorage (Alaska) USA, pp. 146-142

[15] Ost L., Mandelli M., Almeida G. M., Moller L., Indrusiak L. S., Sassatelli G., Benoit P., Glesner M., Robert M., Moraes F., *"Power-aware dynamic mapping heuristics for NoC-based MPSoCs using a unified model-based approach"*, journal ACM Transactions on Embedded Computing Systems (TECS), Volume 12 Issue 3, March 2013, Article No. 75

[16] L. Barthe, L. V. Cargnini, P. Benoit and L. Torres, *"The SecretBlaze: A Configurable and Cost-Effective Open-Source Soft-Core Processor"*, 25th IEEE International Parallel & Distributed Processing Symposium, May 16-20, 2011, Anchorage (Alaska) USA, pp. 310-313

[17] Barthe L., Cargnini L. V., Benoit P., Torres L., *"Optimizing an Open-Source Processor for FPGAs: A Case Study"*, IEEE FPL'11: Field Programmable Logic and Applications (2011), Greece, pp. 551-556

[18] Busseuil R., Barthe L., Almeida G. M., Ost L., Bruguier F., Sassatelli G., Benoit P., Robert M., Torres L., *"Open-Scale: A Scalable, Open-Source NOC-based MPSoC for Design Space Exploration"*, IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig), 2011, pp. 357-362

[19] Niyogi, K. and Marculescu. "Speed and voltage selection for GALS systems based on voltage/frequency islands", ASP-DAC '05. ACM, New York, 292-297

[20] Deniz, Z.T.; Leblebici, Y.; Vittoz, E.A., *"On-Line Global Energy Optimization in Multi-Core Systems Using Principles of Analog Computation",* Solid-State Circuits, IEEE Journal of vol.42, no.7, pp.1593-1606, July 2007

[21] Mutapcic, A. et al., *"Processor speed control with thermal constraints".* Trans. Cir. Sys. Part I 56, 9 (Sep. 2009), 1994-2008.

[22] H. Jung and M. Pedram, *"Uncertainty-Aware Dynamic* Power Management in Partially Observable Domains", IEEE Trans. on VLSI Systems, 2009.

[23] Q.Wu,P.Juang, *et al.* , *"Formal online methods for* voltage/frequency control in multiple clock domain microprocessors", SIGARCH Comput. Archit. News, vol. 32, pp. 248–259, October 2004

[24] Y. Zhu and F. Mueller, "Feedback EDF scheduling exploiting hardware- assisted asynchronous dynamic voltage scaling", SIGPLAN Not., vol. 40, pp. 203–212, June 2005

[25] U. Y . Ogras, R. Marculescu, and et al., *"Variation- adaptive feedback control for networks-on-chip with multiple clock domains",* Proceedings of the 45th annual Design Automation Conference (DAC'08), pp. 614– 619, June 2008

[26] A. Sharifi, H. Zhao, and et al., *"Feedback control for providing qos in noc based multicores",* Proceedings of the Conference on Design, Automation and Test in Europe (DATE'10), pp. 1384–1389, March 2010

[27] G. Almeida, R. Busseuil, L. Ost, F. Bruguier, G. Sassatelli, P. Benoit, L. Torres, M. Robert, *"PI and PID Regulation Approaches for Performance-Constrained Adaptive Multiprocessor System-on-Chip"*, Embedded Systems Letters, IEEE, Volume: PP, Issue:99, ISSN: 1943-0663, DOI: 10.1109/LES.2011.2166373, September 2011, pp. 1-4

[28] Puschini D., Clermidy F., Benoit P., Sassatelli G., Torres L., A Game-Theoretic Approach for Run-Time Distributed Optimization on MP-SoC, International Journal of Reconfigurable Computing ID 403086 (2008) 11

[29] I. Mansouri, P. Benoit, D. Puschini, L. Torres, F. Clermidy, G. Sassatelli, *"Dynamic Energy Optimization in NoC-based System-on-Chips"*, Journal of Low Power Electronics JOLPE – Vol. 6, N° 4, December 2010, pp. 564-577

[30] Johansson, B.; Keviczky, T.; Johansson, M.; Johansson, K.H., "Subgradient methods and consensus algorithms for solving convex optimization problems," *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on* , vol., no., pp.4185,4190, 9-11 Dec. 2008

[31] Stephen Boyd, Persi Diaconis, and Lin Xiao. Fastest mixing markov chain on a graph. SIAM Rev., 46 :667–689, April 2004.

[32] Miro-Panades, I.; Beigne, E.; Thonnart, Y.; Alacoque, L.; Vivet, P.; Lesecq, S.; Puschini, D.; Molnos, A.; Thabet, F.; Tain, B.; Ben Chehida, K.; Engels, S.; Wilson, R.; Fuin, D., "A Fine-Grain Variation-Aware Dynamic Vdd-Hopping AVFS Architecture on a 32 nm GALS MPSoC," *Solid-State Circuits, IEEE Journal of* , vol.49, no.7, pp.1475,1486, July 2014.

[33] Alyssa Bonnoit, Sebastian Herbert, Diana Marculescu, and Lawrence Pileggi. 2009. Integrating dynamic voltage/frequency scaling and adaptive body biasing using test-time voltage selection. In *Proceedings of the 2009 ACM/IEEE international symposium on Low power electronics and design* (ISLPED '09).

[34] R. Wilson, E. Beigne, et al., ''A 460 MHz at 397mV, 2.6 GHz at 1.3V, 32b VLIW DSP, embedding Fmax tracking'', ISSCC 2014

[35] Piotr Zajac and Jacques Henri Collet, "Production Yield and Self-Configuration in the Future Massively Defective Nanochips," in Defect and Fault-Tolerance in VLSI Systems (DFT), Rome, September 2007, pp. 197-205.

[36] Vincenzo De Florio, Geert Deconinck, and Rudy Lauwereins, "An Algorithm for Tolerating Crash Failures in Distributed Systems," Engineering of Computer Based Systems, pp. 9-17, April 2000.

[37] Bernauer Andreas, Fritz Dirk, Sander Björn, Bringmann Oliver, and Rosenstiel Wolfgang, "Current state of ASoC design methodology," Organic Computing - Controlled Self-organization - Dagstuhl Seminar Proceedings, 2008.

[38] J. F. Tarrillo, J. Tonfat, R. Reis, F. Kastensmidt, F. Bruguier, M. Bourrée, P. Benoit and L. Torres, *"Using Electromagnetic Emanations for Variability Characterization in Flash-Based*

*FPGAs"*, Symposium on VLSI, 2013. ISVLSI '13. IEEE Computer Society Annual, 2013, pp. 109-114

[39] Fernanda Lima Kastensmidt, Jorge Tonfat, Thiago Hanna Both, Paolo Rech, Gilson Wirth, Florent Bruguier, Pascal Benoit. Voltage scaling and aging effects on soft error rate in SRAM-based FPGAs. *Microelectronics Reliability*, Elsevier, 2014, 54 (9-10), pp.2344-2348

[40] Abdulazim Amouri, Florent Bruguier, Saman Kiamehr, Pascal Benoit, Lionel Torres, et al.. Aging effects in FPGAs: an experimental analysis. *FPL'2014: 24th International Conference on Field Programmable Logic and Applications*, Sep 2014, Munich Germany. 2014

[41] D. Ernst, S. Das, S. Lee, D. Blaauw, T. Austin, T. Mudge, Nam Sung Kim, and K. Flautner. Razor : circuit-level correction of timing errors for low-power operation. Micro, IEEE, 24(6) :10 –20, nov.-dec. 2004.

[42] T. Kuroda, K. Suzuki, S. Mita, T. Fujita, F. Yamane, F. Sano, A. Chiba, Y. Watanabe, K. Matsuda, T. Maeda, T. Sakurai, and T. Furuyama. Variable supply-voltage scheme for low-power high-speed cmos digital design. Solid-State Circuits, IEEE Journal of, 33(3) :454 –462, mar 1998.

[43] R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Ap- plications in Speech Recognition," in *Readings in Speech Recognition*. San Francisco, CA, USA: Morgan Kaufmann, 1989, pp. 267–296

[44] SSRN M. Perlin, MS regress—A package for Markov regime switch- ing models in Matlab, Aug. 2007. [Online]. Available: http://ssrn.com/ abstract=1714016 SSRN

[45] S. Mangard, E. Oswald, and T. Popp. *Power Analysis Attacks - Revealing the Secrets of Smart Cards*, Springer, 2007

[46] K. Tiri and I. Verbauwhede. Securing Encryption Algorithms against DPA at the Logic Level: Next Generation Smart Card Technology. In *Cryptographic Hardware and Embedded Systems*, CHES, pages 125-136, 2003

[47] K. Tiri, D. Hwang, A. Hodjat, B.-C. Lai, S. Yang, P. Schaumont, and I. Ver- bauwhede. Prototype IC with WDDL and Differential Routing - DPA Re- sistance Assessment. In *Proceedings of the 2005 Cryptographic Hardware and Embedded Systems Workshop*, CHES, pages 354-365, 2005

[48] V. Lomné, P. Maurine, L. Torres, M. Robert, R. Soares, and N. Calazans. Evaluation on FPGA of Triple Rail Logic Robustness against DPA and DEMA. In *Design, Automation and Test in Europe Conference*, DATE, pages 634-639, 2009

[49] M. Nassar, S. Bhasin, J.-L. Danger, G. Duc, and S. Guilley. BCDL: A High Speed Balanced DPL for FPGA with Global Precharge and no Early Evalua- tion. In *Design, Automation and Test in Europe Conference*, DATE, pages 849- 854, 2010.

[50] D. May, H. L. Muller, and N. P. Smart. Non-Deterministic Processors. In *Australasian Conference on Information Security and Privacy*, ACISP, 2001

[51] Amouri, A.; Tahoori, M., "High-level aging estimation for FPGA-mapped designs," *Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on* , vol., no., pp.284,291, 29-31 Aug. 2012

# PARTIE 3 :
# ARTICLES SIGNIFICATIFS

Afin d'approfondir certaines des contributions de cette thèse, les articles suivants sont joints en annexe du manuscrit :

1) Thème « Conception d'architectures reconfigurables et adaptatives »

   G. Almeida, Saint-Jean Nicolas, Varyani Sameer, G. Sassatelli, P. Benoit, L. Torres, *"An Adaptive Message Passing MPSoC Framework"*, Hindawi International Journal of Reconfigurable Computing, Volume 2009, 242981 (2009) 18, pp. 1-18

2) Thème « Optimisation des systèmes adaptatifs »

   I. Mansouri, P. Benoit, D. Puschini, L. Torres, F. Clermidy, G. Sassatelli, *"Dynamic Energy Optimization in NoC-based System-on-Chips"*, Journal of Low Power Electronics JOLPE – Vol. 6, N° 4, December 2010, pp. 564-577

3) Thème « Systèmes adaptifs de confiance » / Fiabilité

   Fernanda Lima Kastensmidt, Jorge Tonfat, Thiago Hanna Both, Paolo Rech, Gilson Wirth, Florent Bruguier, Pascal Benoit. Voltage scaling and aging effects on soft error rate in SRAM-based FPGAs. *Microelectronics Reliability*, Elsevier, 2014, 54 (9-10), pp.2344-2348

4) Thème « Systèmes adaptifs de confiance » / Monitoring de la consommation

   Mansouri I., Benoit P., Torres L., Clermidy F., *"Fine-grain dynamic energy tracking for system-on-chip"*, IEEE Transactions on Circuits and Systems II: Express Briefs, Volume:60, Issue: 6, pp. 356 – 360

5) Thème « Systèmes adaptifs de confiance » / Contremesures pour processeurs RISC

   F. Bruguier, P. Benoit, L. Torres, L. Barthe, M. Bourree, V. Lomne, "Cost-effective Design Strategies for Securing Embedded Processors", *IEEE Transactions on Emerging Topics in Computing*, doi:10.1109/TETC.2015.2407832, February 2015

Hindawi Publishing Corporation
International Journal of Reconfigurable Computing
Volume 2009, Article ID 242981, 20 pages
doi:10.1155/2009/242981

*Research Article*

# An Adaptive Message Passing MPSoC Framework

**Gabriel Marchesan Almeida, Gilles Sassatelli, Pascal Benoit, Nicolas Saint-Jean, Sameer Varyani, Lionel Torres, and Michel Robert**

*Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM), Centre National de la Recherche Scientifique (CNRS), University of Montpellier 2, 161 rue Ada, 34392 Montpellier, France*

Correspondence should be addressed to Gabriel Marchesan Almeida, gabriel.marchesan@lirmm.fr

Received 19 December 2008; Accepted 14 April 2009

Recommended by J. Manuel Moreno

Multiprocessor Systems-on-Chips (MPSoCs) offer superior performance while maintaining flexibility and reusability thanks to software oriented personalization. While most MPSoCs are today heterogeneous for better meeting the targeted application requirements, homogeneous MPSoCs may become in a near future a viable alternative bringing other benefits such as run-time load balancing and task migration. The work presented in this paper relies on a homogeneous NoC-based MPSoC framework we developed for exploring scalable and adaptive on-line continuous mapping techniques. Each processor of this system is compact and runs a tiny preemptive operating system that monitors various metrics and is entitled to take remapping decisions through code migration techniques. This approach that endows the architecture with decisional capabilities permits refining application implementation at run-time according to various criteria. Experiments based on simple policies are presented on various applications that demonstrate the benefits of such an approach.



## 1. Introduction

The exponentially increasing number of transistors that can be placed on an integrated circuit is permitted by the dropping of technology feature sizes. This trend plays an important role at the economic level, although the price per transistor is rapidly dropping the NRE (Nonrecurring Engineering) costs, and fixed manufacturing costs increase significantly. This pushes the profitability threshold to higher production volumes opening a new market for flexible circuits which can be reused for several product lines or generations and scalable systems which can be designed more rapidly in order to decrease the Time-to-Market. Moreover, at a technological point of view, current variability issues could be compensated by more flexible and scalable designs. In this context, Multiprocessor Systems-on-Chips (MPSoCs) are becoming an increasingly popular solution that combines flexibility of software along with potentially significant speedups.

These complex systems usually integrate a few mid-range microprocessors for which an application is usually statically mapped at design-time. Those applications however tend to increase in complexity and often exhibit time-changing workload which makes mapping decisions suboptimal in a number of scenarios. Additionally, such systems are designed in very deep-submicron technologies that bring a number of hardly predictable physical effects that, associated also to the increasing process variability, demonstrate the intrinsic and unavoidable unreliability of future nanoscale integrated systems.

These facts challenge the design techniques and methods that have been used for decades and push the community to research new approaches for achieving system adaptability and reliability (out of unreliable technology components).

This paper presents a hardware/software framework (HS-Scale platform) that is based on a set of adaptive principles which endows the architecture with some decisional capabilities. This approach helps continuously refining application mapping for optimizing various criteria such as performance or power consumption and should eventually enable fault tolerance.

This hardware/software framework is intended to permit the exploration of scalable solutions for future MPSoCs

in the context of massive on-chip parallelism with several hundreds of processing elements (PEs). Therefore, the proposed architecture relies on principles that do not imply resource sharing among processors in the broad sense of the term. The system is made of a regular arrangement of PEs that runs applications in a distributed way, exchanging messages that relate to both platform management and application data. The used programming model is derived from a popular message passing interface (MPI) system that has been augmented for supporting adaptive mechanisms.

This paper is organized as follows.

Section 2 presents the related works in the field of multiprocessor systems, programming models, and task migration techniques. Section 3 introduces the HS-Scale framework which covers the hardware, software, and the programming model used in this approach. Section 4 shows the validations in terms of both developed hardware and area utilization figures in the context of an SoC realization. Section 5 presents the results of various applications mapped on the framework emphasizing on the cost induced by the used migration techniques and the corresponding observed benefits. Section 6 draws some conclusions on the presented work and puts this in perspective with other upcoming challenges of the area.

## 2. Related Works

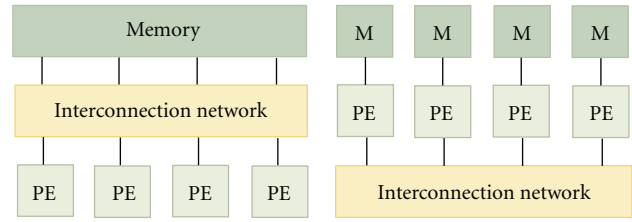
This section shortly introduces the different existing families of multiprocessor systems and puts focus on the relevant approaches that are found in the fields of scalable message passing architectures and task migration techniques.

*2.1. Preliminary Considerations.* Multiprocessor systems are increasely considered as an attractive solution for accelerating computation. Parallel architectures have been studied intensively during the past 40 years; there is consequently a huge amount of books [1, 2] related to this topic, and we will therefore only focus on general concepts. The most common type of multiprocessor systems falls into the Multiple Instruction Multiple Data (MIMD) family as that defined by the Flynns taxonomy [3]. There are two types of MIMD machine classified in accordance with their memory architecture:

(i) shared memory architectures in which all processors share the same memory resources; therefore all changes done by a processor to a given memory location become visible to all other processors of the system;

(ii) distributed memory architectures in which every processor has its own private memory; therefore one processor cannot read directly in the memory of another processor. Data transfers are implemented using message passing protocols.

From an architecture design point of view shared memory machines are poorly scalable because of the limited bandwidth of the memory. Existing realizations marginally use more than tenth of processors because of this reason.




(a) Shared memory (b) Distributed memory

Figure 1: Shared *x* distributed memory.

Distributed memory machines are more scalable since only the communication medium may be shared among processors. There exist two families of programming models, each of which exhibits a better adequacy to one of the previously presented architecture families.

(i) Shared memory systems require synchronization mechanisms such as semaphores, barriers, and locks since no explicit communication mechanism exists. POSIX threads [4] and OpenMP [5] are two popular implementations of the thread model on shared memory architectures.

(ii) Distributed memory systems require mechanisms for supporting explicit communications between processes (that may be hosted on the same or different processors). Usually a library of primitives that allow writing in communication channels is used. The Message Passing Interface (MPI) [6] is the most popular standard that is used in High-Performance Computing (HPC) computer clusters for instance.

In a shared memory architecture (Figure 1(a)), processes (executed by different processors) can easily exchange information through shared variables; however it requires handling carefully synchronization and memory protection. In a distributed memory architecture (Figure 1(b)), a communication infrastructure is required in order to connect processing elements and their memories and allow exchanging information.

Since this work targets massively parallel on-chip Multiprocessor systems, scalability is a major concern in the approach. For this reason, we put focus on distributed memory machines and therefore choose a message passing programming model for it provides a natural mapping to such machines.

*2.2. Message Passing for Embedded Systems.* The Message passing model is based on explicit communication between tasks. This model is often used for architectures that do not provide global address space; here communications among tasks take place through messages and are implemented with functions allowing reading and writing to communication channels. CORBA, DCOM, SOAP, and MPI are examples of message passing models.

Message Passing Interface (MPI) is the most popular implementation of the message passing model, and only for this model some embedded implementations exist. The

Message Passing Interface is a specification for an API that allows many PEs to communicate through a communication network.

MPI provides a comprehensive number of primitives that relate to general-purpose distributed computing; a number of work have devised lightweight implementations supporting only a subset of the mechanisms of MPI for embedded processors and systems. This makes sense since the nature of applications for these systems is well defined, often limited to data flow applications for which Kahn Process Networks formalism offer a sufficient support that requires only blocking read operations [7]. Some MPI implementations are layered, and advanced communication synchronization primitives (such as collective, etc.) found in the upper layers make use of the simple point-to-point primitives such as MPI_Send() and MPI_Receive(). This enables using these collective mechanisms in an application-specific basis in case they prove necessary.

The specific requirements of embedded systems have led programmers to develop lightweight MPI implementations, basically built upon a subset of the original MPI mechanisms. In [8] the authors present TMD-MPI which is a lightweight MPI implementation for multiple processors across multiple FPGAs. It relies on a layered implementation which provides only 11 primitives. As no operating system is used, task mapping is static and done at design-time. Similarly, authors in [9] have also selected 11 primitives among which only 2 relate to point-to-point communications. Finally, in [10] authors present eMPI which also uses the simplest low-level point-to-point communication primitives in a layered style.

*2.3. Task Migration Support.* Task migration techniques have been mainly studied in contexts that fall in one of the following areas.

(i) General purpose computing, involving usually a single computer made of several processors or processor cores. Such systems are usually built around shared memory architectures.

(ii) High-Performance Computing (HPC) computer clusters. Such systems are usually of distributed memory type and therefore generally use message passing programming style.

(iii) Multiprocessor embedded systems, which may make use of either shared or distributed memory architecture.

For shared memory systems such as today's multicore computers, task migration is facilitated by the fact that no data or code has to be moved across physical memories: since all processors are entitled to access any location in the shared memory, migrating a task comes down to electing a different processor for execution. There exist several efficient implementations on general purposes OS such as Windows or Linux [11].

In the case of multiprocessor-distributed memory/message passing architectures, both process code and state have to be migrated from a processor private memory to another, and synchronizations must be performed using exchanged messages such as in [12] which targets Linux computer clusters. Some other approaches aimed at augmenting MPI for providing a support for process migration, such as [13, 14]. In [15] users present similar features based on a JAVA MPI framework that provides hardware independence; they show that despite migrating tasks imply overheads, which are in the order of seconds, significant speedups can be achieved. All these approaches target computer clusters with the typical resources of general-purpose computers and are therefore hardly applicable to MPSoCs.

Task migration has also been explored for MPSoCs, notably based on locality considerations [12] for decreasing communication overhead or power consumption [16]. In [17], authors present a migration case study for MPSoCs that relies on the Clinux operating system and a check pointing mechanism. The system uses the MPARM framework [18], and although several memories are used, the whole system supports data coherency through a shared memory view of the system.

In [19] authors present an architecture aiming at supporting task migration for a distributed memory multiprocessor-embedded system. The developed system is based on a number of 32-bit RISC processors without memory management unit (MMU). The used solution relies on the so-called "task replicas" technique; tasks that may undergo a migration are present on every processor of the system. Whenever a migration is triggered, the corresponding task is respectively inhibited from the initial processor and activated in the target processor. This solution induces a significant memory overhead for every additional task and therefore falls beyond the scope of this paper. Finally, to the best of our knowledge, no other work combines the use of a message passing programming model, on-chip multiprocessor system, and transparent decentralized automated task migration.

## 3. HS-Scale

The key motivations of our approach are scalability and self-adaptability; the system presented in the rest of this paper is built around a distributed memory/message passing system that provides efficient support for task migration. The decision-making policy that controls migration processes is also fully distributed for scalability reasons. This system therefore aims at achieving continuous, transparent, and decentralized run-time task placement on an array of processors for optimizing application mapping according to various potentially time-changing criteria.

*3.1. System Overview.* The system is based on an array of compact general-purpose PEs interconnected through a packet switching Network-on-Chip. The HS-scale system is a purely distributed memory system which is programmed using a simple message passing protocol. Contrary to MPI, processes must not be mapped to a given processor but shall freely move in the system according to user-definable policies that may aim at optimizing a given property in the system, such as performance or power consumption. Both hardware and software resources are intended to be minimalist for

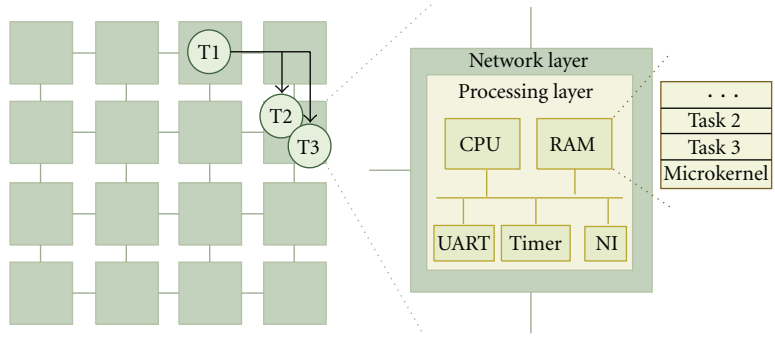



Figure 2: NPU structural description.

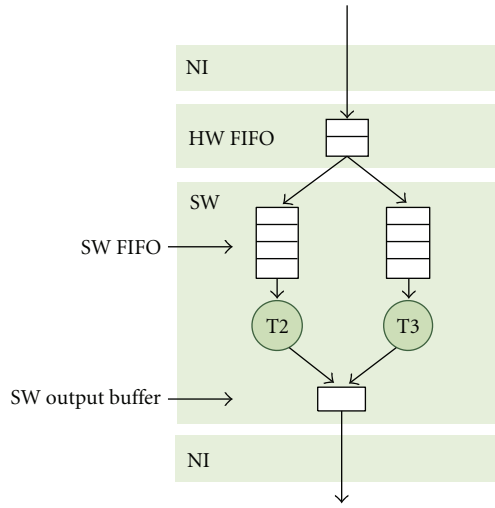



Figure 3: NPU functional description.

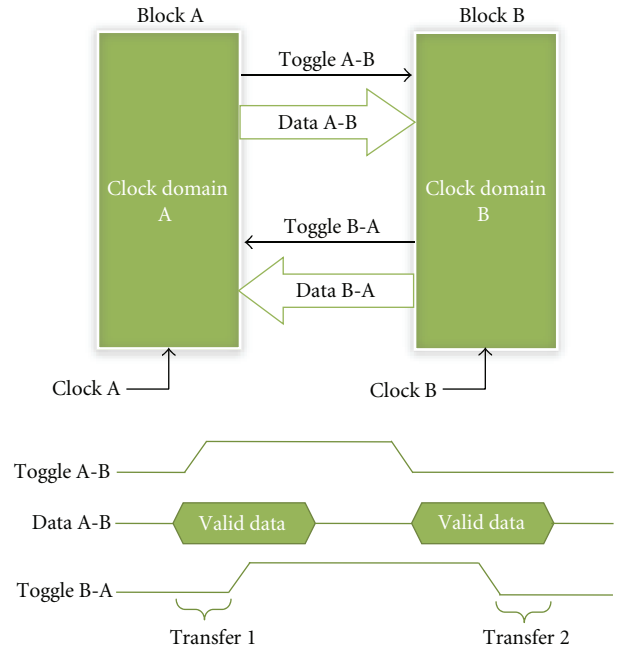



Figure 4: The asynchronous toggle protocol.

favoring compactness of processors and therefore encouraging massive parallelism. Also, for scalability reasons, there exists no master in the system unless a given application requires it.

### 3.2. Hardware Structure

*3.2.1. Network Processing Unit.* The architecture is made of a homogeneous array of Processing Elements (PEs) communicating through a packet-switching network. For this reason, the PE is called NPU, for Network Processing Unit. Each NPU, as detailed later, has multitasking capabilities which enable time-sliced execution of multiple tasks. This is implemented thanks to a tiny preemptive multitasking Operating System which runs on each NPU. The structural and functional views of the NPU are depicted in Figures 2 and 3, respectively.

The NPU is built around two main layers, the network layer and the processing layer. The Network layer is essentially a compact routing engine ($XY$ routing). Packets are read from incoming physical ports, then forwarded to either outgoing ports or the processing layer. Whenever a packet header specifies the current NPU address, the packet is forwarded to the network interface (NI in Figure 3). The network interface buffers incoming data in a small hardware FIFO and simultaneously triggers an interrupt to the processing layer. The interrupt then activates data demultiplexing from the single hardware FIFO to the appropriate software FIFO as illustrated in Figure 4.

The processing layer is based on a simple and compact RISC microprocessor, its static memory, and a few peripherals (one timer, one interrupt controller, one UART) as shown in Figure 2. A multitasking microkernel implements the support for time-multiplexed execution of multiple tasks.

The processor used has a compact instruction set comparable to an MIPS-1 [20]. It has 3 pipelines stages, no cache, no Memory Management Unit (MMU), and no memory protection support in order to keep it as small as possible.

*3.2.2. Communication Infrastructure.* For technology-related concerns, a regular arrangement of processing elements (PEs) with only neighboring connections is favored. This helps in (a) preventing using any long lines and their associated undesirable physical effects in deep submicron CMOS

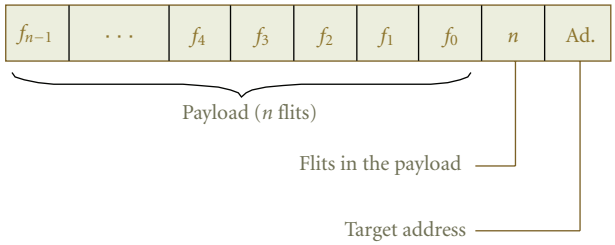



Figure 5: Packet format.

technologies and (b) synthesizing the clock distribution network since an asynchronous communication protocol between the PEs might be used. Also, from a communication point of view, the total aggregated bandwidth of the architecture should increase proportionally with the numbers of PEs it possesses, which is granted by the principle of abstracting the communications through routing data in space. The Network-on-Chip (NoC) paradigm enables that easily thanks to packet switching and adaptive routing.

The communication framework of HS-scale is derived from the Hermes Network-on-chip; refer to [21] for more details. The routing is of wormhole type, which means that a packet is made of an arbitrary number of flits which all follow the route taken by the first one which specifies the destination address. Figure 5 depicts the simple packet format used by the network framework constituted by the array of processing elements. Incoming flits are buffered in input buffers (one per port). Arbitration follows a round-robin policy giving alternatively priority to input ports. Once access to an output port is granted, the input buffer sends the buffered flits until the entire packet is transmitted (wormhole routing).

Inter-NPU communications are fully asynchronous and are based on the toggle-protocol. As depicted in Figure 4 this protocol uses two toggle signals for the synchronization, a given data being considered valid when a toggle is detected. When the data is latched, another toggle is sent back to the sender to notify the acceptance. This solution allows using completely unrelated clocks on each PE in the architecture.

*3.3. Operating System.* The lightweight operating system we use was designed for our specific needs. Despite being small (28 KB), this kernel does preemptive switching between tasks and also provides them with a set of communication primitives that are presented later. Figure 6 gives an overview of the operating system infrastructure and the services it provides.

Figure 7 presents the process state diagram each task follows depending on the events that may occur. This scheme answers to the general principles of operating systems in general although transition events have been specialized for this specific case.

The interrupts manager may receive interrupts from 3 hardware sources: UART, Timer, and network interface (NI). Whenever an interrupt occurs, other interrupts are disabled, and the processor context is saved in the system stack. Following the type of interrupt, it reads from the UART, schedules another task (timer), receives data from other NPUs, or use a communication primitive (interrupt from the network interface FIFO_in). Afterwards processor context is restored, and interrupts are re-enabled. The scheduler is the core of the microkernel. Each time a timer interrupt occurs, it checks if there is a new task to run. In the positive case, it executes this new task. Otherwise, it has two possibilities: either there is no task to schedule then it just runs an idle task or there is at least one task to schedule. Tasks are scheduled periodically following a round robin policy (there is no priority management between tasks) as depicted in Figure 8.

Figure 9 presents the memory layout of an NPU running this operating system along with two tasks. Each task is located in a memory region that embeds code, data, and stack segments.

*3.4. Dynamic Task Loading and Migration (PIC).* One of the objectives of this work is to enable dynamic load balancing which implies the capability to migrate running tasks from processor to processor. Migrating tasks usually implies the following.

(i) To dynamically load in memory and schedule a new process.

(ii) To restore the context of the task that has been migrated.

*3.4.1. Dynamic Process Loading.* Both points are challenging for such microprocessor targets since, for density reasons, no Memory Management Unit (MMU) is available. An MMU, among other tasks, usually performs the translation between virtual and physical addresses and therefore permits to load and run a code in an arbitrary region of the physical memory. The code then performs read, write, and jump operations to virtual memory locations that are being translated into physical locations matching the memory layout decided by the operating system upon loading.

A possible alternative to overcome this problem relies in resolving all references of a given code at load-time; such a feature is partly supported in the ELF format [22] which lists the dynamic symbols of the code and enables the operating system loader to link the code for the newly decided memory location. Such mechanisms are memory consuming and imply a significant memory overhead which clearly puts this solution out of the scope of the approach.

Another solution for enabling the loading of processes without such mechanisms relies on a feature that is partly supported by the GCC compiler that enables to emit relocatable code (PIC: Position Independent Code). This feature generally used for shared libraries generates only relative jumps and accesses data locations and functions using a Global Offset Table (GOT) that is embedded into the generated ELF file. A specific postprocessing tool which operates on this format was used for reconstructing a completely relocatable executable. Experiments show that both memory and performance overheads remain under 5% for this solution which is clearly acceptable.

Figure 10(a) shows an example of relative jump with PIC compilation with code compiled for an execution at the

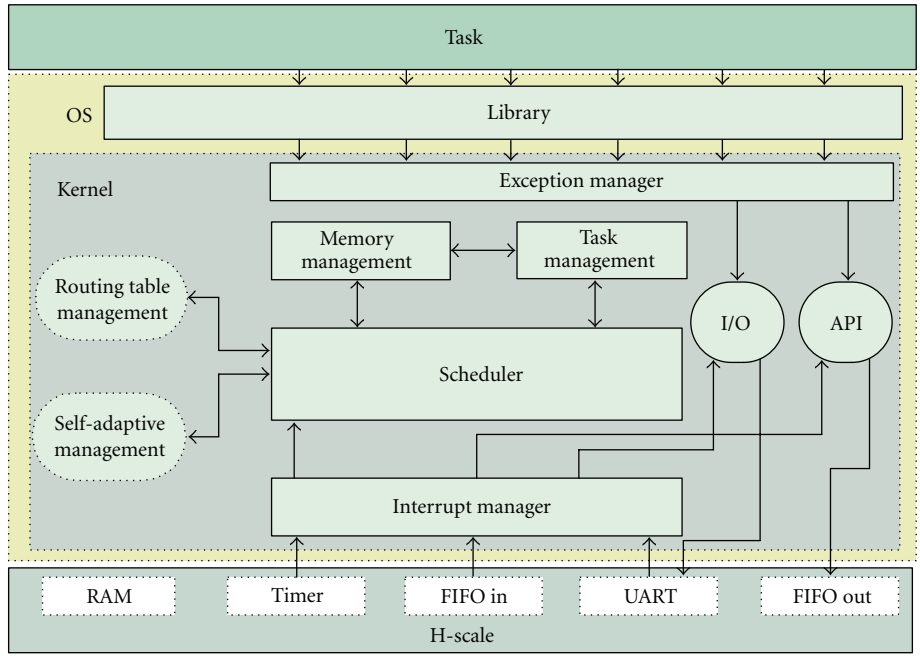



Figure 6: Operating system overview.

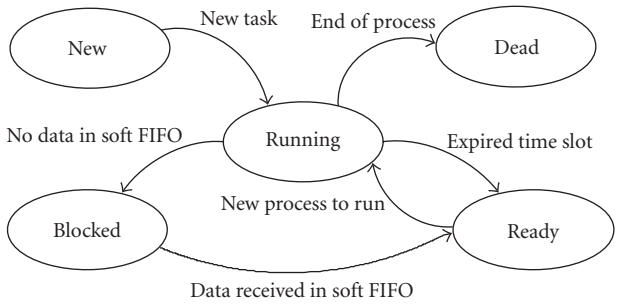



Figure 7: Task state diagram.

address 0x0000; the address for the jump is referenced by the sum of current address and GOT entry reference. So if the code needs to be executed to another location, for example, $0 \times 0100$ (Figure 10(b)), the code is copied to the new location, and the only modification to perform is to add the same offset of the code ($0 \times 0100$) to the GOT section entries.

*3.4.2. Task Context Migration.* Migrating a process implies not only instantiating a new executable into the memory but also restoring its context. Again, the lack of MMU makes this task difficult since the context of the process includes the stack which not only embeds data (such as return values of functions) but also returns addresses that are memory-location dependent. The solution we developed is based on defining migration points that are at specific locations in the code, namely, whenever a communication primitive is called. This method is restrictive since it assumes that the computation relies on a strict consumer/producer model where no internal state is kept from iteration to iteration. This translates in the fact that there cannot be any dependencies between two adjacent computed data chunks.

When a task migration order is issued by the operating system, the following sequence of action is initiated between NPU1 which is current host for the task and NPU2 which is the future.

(1) Task code is sent from NPU1 to NPU2. NPU2 then loads the task into memory, creates the necessary software FIFOs, and runs the task which is frozen when it reaches the first communication primitive call (MPI_Send() or MPI_Receive()).

(2) NPU1 modifies routing tables (that embeds task and FIFO placements) and broadcasts this information to the other NPUs. Future messages for the task will be buffered in the newly created FIFO on NPU2.

(3) Task execution on NPU1 continues until the next communication primitive call is reached which freezes application execution. Following this, the remaining task FIFO content on NPU1 is sent and reordered on NPU2.

(4) Task execution is resumed on NPU2, and concurrently the task is removed from memory on NPU1.

Table 1 presents an example of migration time of one task (20 KB), between two NPUs with a distance of one. These results show that time migration is mainly due to the time to send executable through the network (with a frequency $f = 25$ MHz). *TShutdown* refers to the time taken by the operating system for unregistering the task. *TSend* represents the time taken for the operating system to perform the necessary actions for formatting and sending the task to

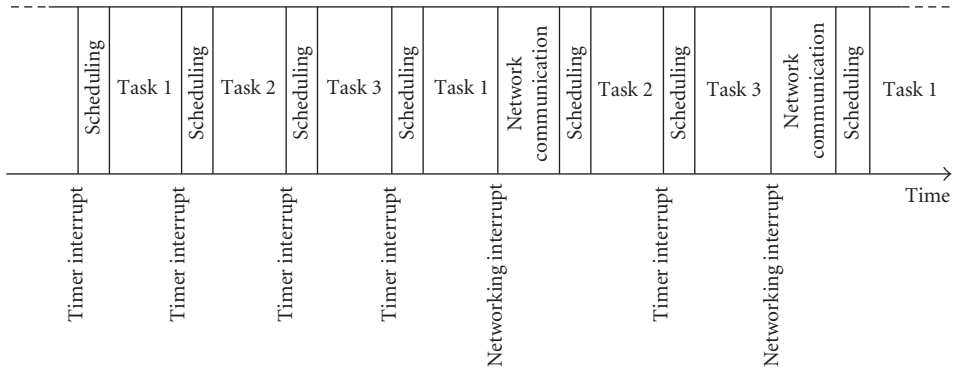



Figure 8: Scheduling diagram.

Table 1: Timeline of the migration mechanism.

| | *TShutdown* | *TSend* | *TReceive* | *TRelight* | *TReboot* | *TMigration* |
|---|---|---|---|---|---|---|
| Time (ms) | 3.067 | 13.970 | 13.968 | 3.110 | 0.107 | 34.222 |

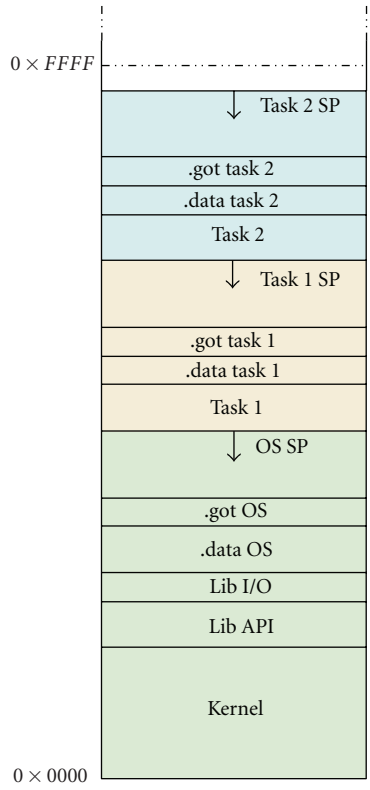



Figure 9: Memory layout.

the remote NPU, the time actually spent in sending the task through the network *TSend* being negligible. Similarly, the operating system of the remote NPU requires a significant amount of time *TReceive* for receiving and instantiating the task in memory. Finally, the last action taken before running the task is updating both local and remote routing tables; this is realized in a time *TRelight*. The task is then rapidly scheduled and executed (*TReboot*).

*3.5. Programming Model.* Programming takes place using a message passing interface. Hence, tasks are hosted on NPUs which provide through their operating system communication primitives that enable data exchanges between communicating tasks. The proposed model used only two communication primitives, MPI_Send() and MPI_Receive(). This communication primitive is based on the synchronous MPI communication primitive (MPI_Send and MPI_Receive). Figure 11 depicts the layered view of the communication protocol we use. MPI_Receive() blocks the task until the data is available while MPI_Send() is blocking until the buffer is available on the sender side. In our implementation each call exhibits this behavior and is translated into a sequence of low-level Send_Data()/Receive_Data() methods that set up a communication channel through a simple request/acknowledge protocol as depicted in Figure 12. This protocol ensures that the remote processor buffer has sufficient space before sending the message which helps lowering the contentions in the communication network and also prevents deadlocks.

Figure 12 depicts the communication stack that is used in our system. Although a hardware implementation could certainly help improving performance, for compactness reasons it is fully implemented in software down to packet assembling.

No explicit group synchronization primitives are provided; however this can be simply achieved in an ad hoc fashion through using MPI_Send() and MPI_Receive() for passing tokens. Broadcast, gather, and similar mechanisms can also be implemented in the same manner. Furthermore, although it could be easily implemented, no nonblocking receive method is provided since the targeted applications usually do not require it.

The prototypes of those functions are as in Algorithm 1.

The prototypes of these functions are self explanatory, a reference to the graph edge identifier, a constant void pointer, and data size expressed in bytes.

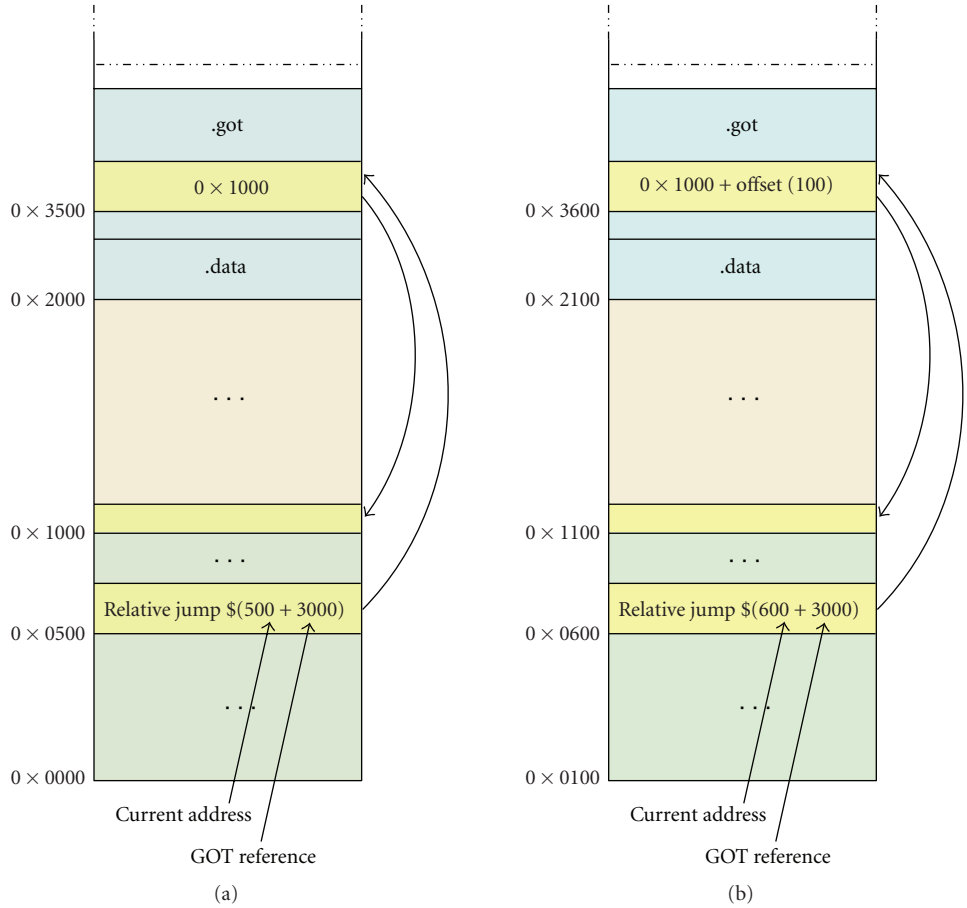



Figure 10: Relative jumps with GOT.

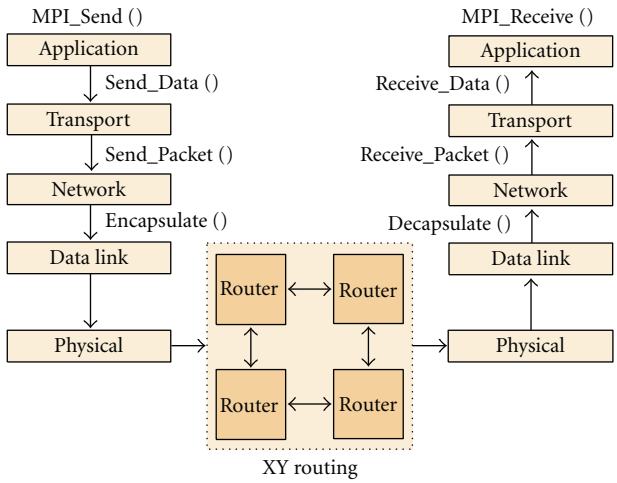



Figure 11: HS-Scale protocol stack.

Figure 13 shows an example of task graph where it can be seen that communication channels feature a (software) FIFO queue at the receiver side. Queues sizes can be parameterized, and their size can be tuned on-line as the operating system provides memory allocation and deallocation services.

```
MPI_Send(int edge, const void *data, int size)
MPI_Receive(int edge, void *data, int size)
```

Algorithm 1

*3.6. Self-Adaptive Mechanisms.* The platform is entitled to take decisions that relate to application implementation through task placement. These decisions are taken in a fully decentralized fashion as each NPU is endowed with equivalent decisional capabilities. Each NPU monitors a number of metrics that drive an application-specific mapping policy; based on these information an NPU may decide to push or attract tasks which results in, respectively, parallelizing or serializing the corresponding tasks execution, as several tasks running onto the same NPU are executed in a time-sliced manner.

Figure 14 shows an abstract example where it can be observed that upon application loading the entire task graph runs onto a single NPU; subsequent remapping decisions then tend to parallelize application implementation as the final step exhibits one task per NPU. Similarly, whenever a set of tasks become subcritical, the remapping could revert to situation 3 where T1, T2, and T3 are hosted on a single

NPU while the other supposedly more demanding do not share NPU processing resources with other tasks. These mechanisms help achieving continuous load-balancing in the architecture but can depending on the chosen mapping policy help refining placement for lowering contentions, latency, or power consumption.

Mapping decisions are specified on an application-specific basis in a dedicated operating system service. Although the policy may be focused on a single metric, composite policies are possible. Three metrics are available to the remapping policy for taking mapping decisions.

(i) *NPU load*. The NPU operating system has the capability of evaluating the processing workload resulting from task execution.

(ii) *FIFO queues filling level*. As depicted in Figure 13, every task has software input FIFO queues. Similarly to NPU load, the operating system can monitor the filling of each FIFO.

(iii) *Task distance*. The distance that separates tasks is also a factor that impacts performance, contentions in the network, and power consumption. Each NPU microkernel knows the placement of other tasks of the platform and can calculate the Manhattan distance with the other tasks it communicates with.

Algorithm 2 shows an implementation of the microkernel service responsible of triggering task migrations. The presented policy simply triggers task migration in case one of the FIFO queues of a task is used over 80%.

The request_task_migration() call then sequentially emits requests to NPUs in proximity order; the migration function will migrate the task to the first NPU which has accepted the request; the migration process is started according to the protocol described previously in Section 3.4.2. This function can naturally be tuned on an application/task specific basis and select the target NPU taking into account not only the distance but also other parameters such as available memory and current load.

We have implemented also a migration policy based on the CPU load. The idea is very similar to the first one, and it consists of triggering a migration of a given task when the CPU load is lower or greater than a given threshold. This approach may be subdivided in two subsets.

(1) Whenever the tasks time is greater than or equal MAX_THRESHOLD, it means that tasks are consuming more than or equal to the maximum acceptable usage of the CPU time.

(2) Whenever the tasks time is less than MIN_THRESHOLD, it means that the tasks are consuming less than the minimum acceptable usage of the CPU time.

For both subsets, the number of tasks inside one NPU must be verified. For the first subset, it is necessary to have, at least, two tasks running in the same NPU. For the second subset, whenever there are one or more tasks in the same NPU, the migration process may occur.

In the same way the migration process occurs whenever the CPU load is less than MIN_ THRESHOLD (20%). When

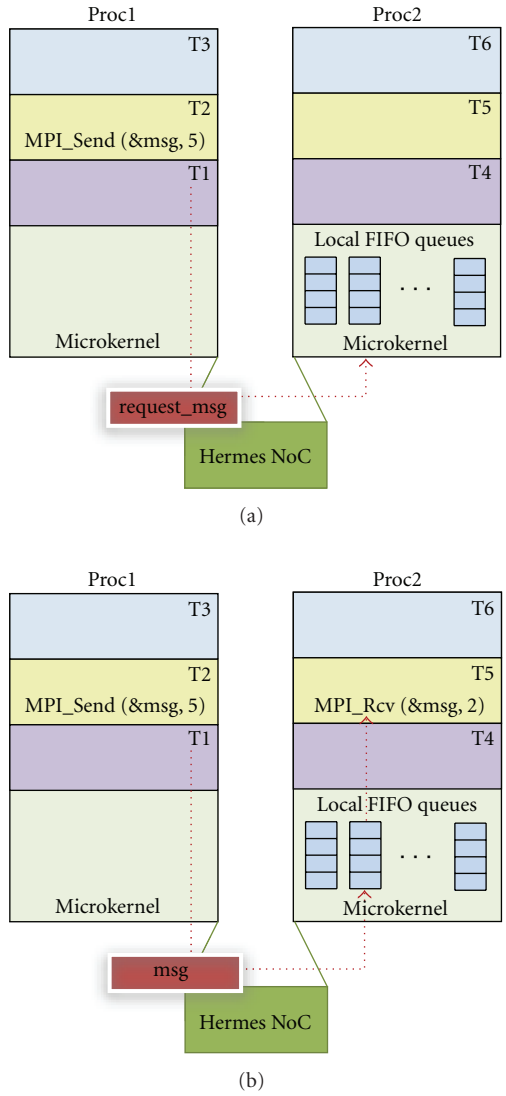



Figure 12: Proactive communication principle.

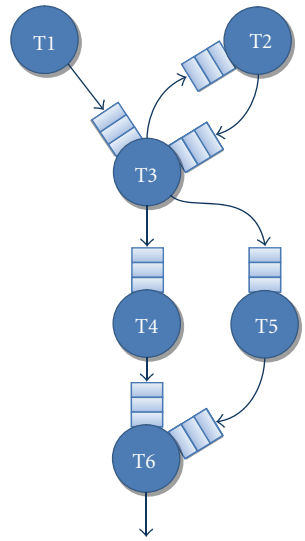



Figure 13: Example of task graph.

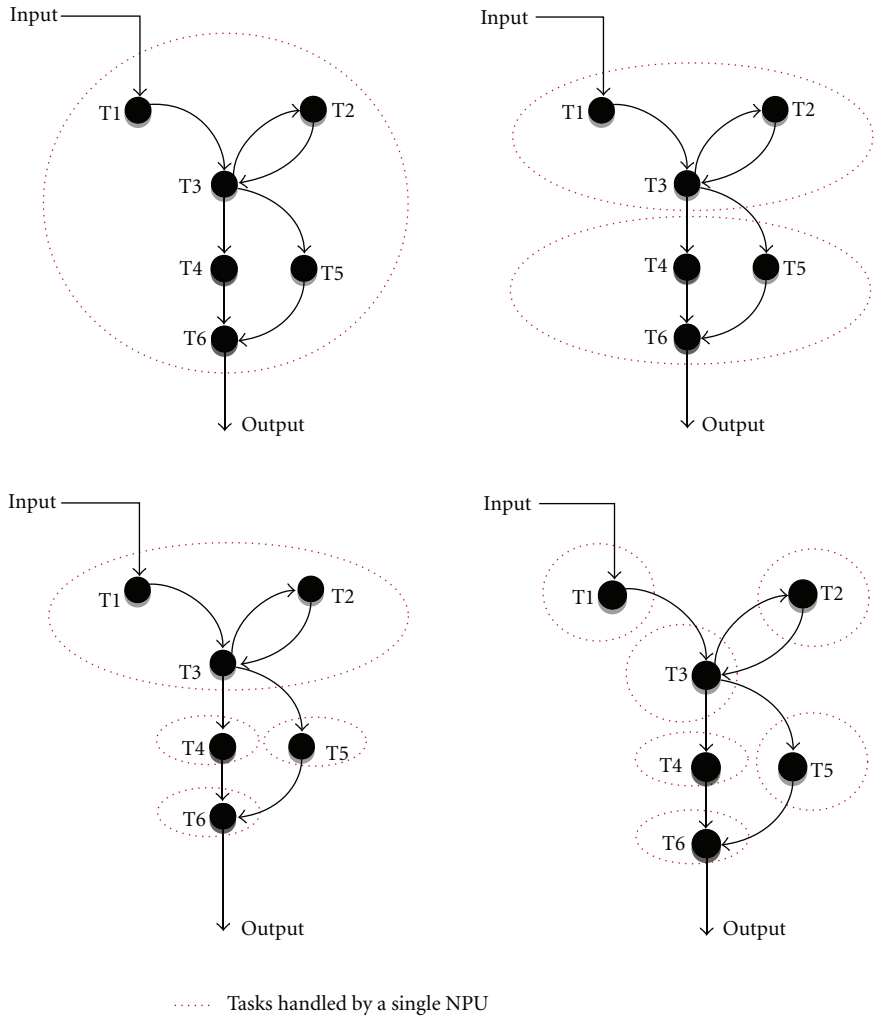



Figure 14: Task graph.

```
void improvement_service_routine(){
    int i, j;
    //Cycles through all NPU tasks
    for (i = 0; i < MAX_TASK; i++){
        //Deactivates policy for dead/newly instantiated tasks
        if (tcb[i].status != NEW && tcb[i].status != DEAD){
            //Cycles through all FIFOs
            for (j = 0; j < tcb[i].nb_socket; j++){
                //Verifies if FIFO usage > MAX_THRESHOLD
                if (tcb[i].fifo_in[j].average > MAX_THRESHOLD){
                    //Triggers migration procedure if task
                    //is not already alone on the NPU
                    if (num_task > 1)
                        request_task_migration(tcb[i].task_ID);
                }
            }
        }
    }
}
```

Algorithm 2

Table 2: Area scalability results.

| Number of NPU | 1 | 2 | 4 (2 × 2) | 9 (3 × 3) | 16 (4 × 4) |
|---|---|---|---|---|---|
| Area ($mm^2$) | 1.14 | 2.29 | 4.60 | 10.33 | 18.40 |

this occurs, the migration function must look for a NPU that is using at given threshold of CPU usage, in this case, 60% of usage. To avoid the task with less than MIN_THRESHOLD keep migrating every time, we have inserted a delay to reduce the number of migrations.

## 4. Validations

*4.1. Estimations of the Silicon Hardware Prototype.* A complete synthesizable RTL level description (about 6000 lines of VHDL) of the H-Scale system has been designed. It has allowed us to validate our approach, to estimate areas (post place and route, with ST Microelectronics 90 nm design kit), and to improve the design. Any instance of the H-Scale MP-SOC system may be easily generated with generic parameters and then evaluated with any standard CAD tool flow (Encounter Cadence was used).

Table 2 summarizes these evaluations. The clock of the NPU has been constrained to 3 nanoseconds allowing a 300 MHz frequency. Table 2 clearly shows the area scalability of H-Scale hardware system (the very low overhead is due to the wires needed to interconnect the NPUs). These results let us easily extrapolate that we could design an HS-Scale system with 32 processors and 2 MB of embedded memory with less than 50 $mm^2$ of silicon area.

*4.2. Multi-FPGA Prototype.* The first validations of the systems were performed thanks to VHDL simulation. Obviously, this was far too slow for realistic application scenarios (about 4 minutes for a 10 milliseconds simulation with a 1.6 GHz processor). Although a SystemC prototype is also available, we chose to develop a scalable multi-FPGA prototype.

*4.2.1. Platform Description.* It is essentially based on a Spartan3 S1000 FPGA which 1920 configurable logic blocks (CLB). The board features several general purpose I/Os (8 slide switches, 4 pushbuttons, 8 LEDs, and 4-digit seven-segment display), 1 MB of fast asynchronous SRAM, several ports for debugging/monitoring purposes (one serial port, a VGA port, and PS2 mouse/keyboard port), and three 40-pin expansion connectors for the interconnections of boards.

As mentioned, one NPU is synthesized on a single FPGA board. The maximum frequency of the synthesized design on Spartan3 S1000 FPGA is 25 MHz. Figure 15 depicts the board with two of the 40-pin expansion connectors used for North, South, East, and West connections. Communications are taking place in an asynchronous fashion as described previously (toggle protocol).

Table 3 gives the device utilization figures for a single NPU hosted on a single board. The complete prototype is then composed of several instances of the prototyping boards connected through the 40-pin expansion connectors.

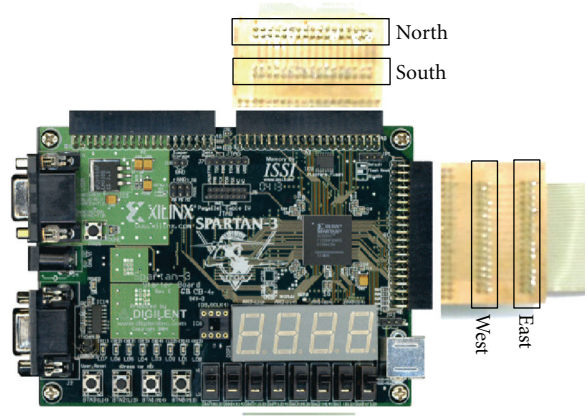



Figure 15: The prototyping board.

One board, that is, one UART of a single NPU, is directly connected to a PC as depicted in Figure 16. This PC is used as a human-machine interface for sending program data (i.e., task codes and microkernel code), the data to compute and to display debugging messages in the monitoring terminal.

Each NPU has originally a bootloader which performs upon power up the following operations in sequence.

(1) It checks whether a PC is connected to the UART port. If so, the NPU initializes its $XY$ coordinates to address $(0 : 0)$. It then acts as a Dynamic Host Configuration Protocol (DHCP) server and proactively sends packets to the East and South ports informing that it has taken address $(0 : 0)$.

(2) If no UART connection is detected, an incoming network request is expected. Once the corresponding packet is received (that the interface NPU has initiated as described above), an address is calculated: East neighbor will take address $(1 : 0)$ and South neighbor address $(0 : 1)$.

(3) This process is reiterated until the boundaries of the network are found. The $X$-axis and $Y$-axis boundaries are then broadcasted in the network in order to inform each NPU of the current network topology.

(4) After the topology information update, the interface NPU bootloader downloads the microkernel code from the PC through the UART interface and broadcasts it to the other NPUs in the network. Each NPU starts up the operating system as soon as received. The microkernel is common for each NPU. Depending on the address of the router, the microkernel knows its location. For application code, again NPU0 receives the code since it is the only one connected to the UART. This code is forwarded only to the processor, where it is supposed to execute. Only a single copy of application code exists inside the system.

This method allows to easily scale up the prototype to any size and shape (form factor may be different than 1). It has

Table 3: FPGA synthesis result.

| Device | #Slices | #FPGA resources used |
|---|---|---|
| NPU | 2496 | 32,50% |
| Router | 683 | 8,89% |
| MIPS R3000 | 1462 | 19,04% |
| Other | 351 | 4,57% |
| Total used | 4992 | 65% |

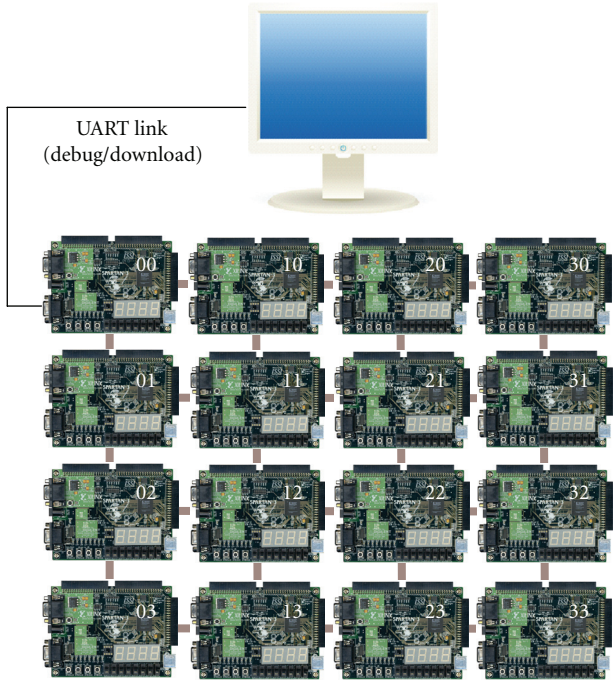



Figure 16: Array of 4 × 4 NPU multiboard.

Table 4: Operating system time and memory costs (KB) size.

| | |
|---|---|
| Microkernel | 10 |
| Com. primitive | 5,12 |
| I/O primitive | 3,12 |
| DATA section | 8,48 |
| STACK section | 0,56 |
| OS Size | 27,28 |

been designed for later exploring reliability issues for reconfiguring the system if an NPU becomes unreachable (defective hardware) for instance. In such a case, the faulty NPU can be removed from the table of available processing units.

*4.2.2. Kernel Characteristics.* Table 4 provides an overview of the memory footprint of our Operating System. In terms of time penalty, each time the OS is invoked (each time a timer interrupt happens), it requires 218 cycles to perform its job. In terms of memory overhead, it requires 27.28 KB. The communication primitives represent almost one fifth of the total memory required by our OS.

*4.3. Description of the Experiments.* The FPGA platform is the basis of our experiments. Those have been carried out on the HS-Scale system in order to study and characterize the strengths and the weaknesses of our approach.

*4.3.1. Set of Applications.* We have chosen 3 different applications: a 2-TAP Finite Impulse Response (FIR) filter, a Data Encryption Standard (DES) encoder, and an MJPEG decoder. The main motivation for using such applications was to cover a wide range of possible dataflow applications in terms of granularities and regularities of the tasks. The FIR filter is based on fine grain tasks with a task graph requiring multiple dependencies. Compared to FIR filter, the granularity of DES tasks and MJPEG tasks is coarser. DES tasks are regular and not data dependent, while MJPEG tasks are irregular and depend on the image characteristics. Some dummy applications have been created for better highlighting the capabilities of different policies.

*4.3.2. Experimental Protocol.* We have developed a set of self-adaptive features for the HS-Scale system: the purpose of this study is to evaluate and to measure the impact of the self-adaptability on application performance. The main metric presented in the next section is the Throughput (TP). It is computed as follows:

$$\mathrm{TP}(\mathrm{KB}\times \mathrm{s}^{-1}) = \frac{\text{Number of Computed Data (KB)}}{\text{Number of Cycles}} \times f(\mathrm{Hz}). \tag{1}$$

Our experiments were performed with each NPU running at $f = 50$ MHz. We have implemented different application scenarios as follows.

*(i) Monoprocessor Implementations.* Each application of our test set is programmed as a monolithic task (with or without the operating system) in order to calculate a reference throughput.

*(ii) Multiprocessor Implementations with Static Mappings.* Each application is described as a task graph application. Performing figures for various static mappings have been collected.

There is a certain degree of randomness in the execution of a scenario due to different reasons. Firstly, the communications between routers are asynchronous. Secondly, the execution of a task on a given NPU depends on several varying parameters such as the presence of data in its FIFO (may depend on other tasks placed on different NPUs communicating through the asynchronous network) and the timer interrupts regarding application start that can induce a different scheduling and a different timing in the decision making process. This is the reason why, for (1) and (2), the experiments were repeated 10 times in order to expose the average throughput and its standard deviation.

*(i) Multiprocessor Implementations with Dynamic Mappings (Migrations).* This case represents our main contribution with self-adaptability features. The studied scenario relies

Table 5: FIR, DES, and MJPEG monolithic implementations and OS cost.

| | | $-OS^{(a)}$ | $+OS^{(b)}$ |
|---|---|---|---|
| FIR | Average TP (KB/s) | 315.92 | 313.08 |
| | Standard deviation of TP (Kb/s) | 0 | 0.09 |
| DES | Average TP (KB/s) | 6.56 | 6.54 |
| | Standard deviation of TP (Kb/s) | 0 | 0.06 |
| MJPEG | Average TP (KB/s) | 35.39 | 34.98 |
| | Standard deviation of TP (Kb/s) | 0 | 0.14 |

[a]Without the operating system.
[b]Without the operating system.

on an application that is sequentially injected on a single NPU which triggers remapping decisions. These remapping decisions are all based on nearest-free neighbor policy where every time the FIFO utilization reached the 80% threshold, a migration was triggered. We have monitored dynamically the throughput and the FIFO utilization ratio in order to plot these metrics as temporal functions.

## 5. Applications and Results

This section is devoted to the analysis of the self-adaptive results obtained on the FPGA prototype. Three classes of results are exposed: (i) monoprocessor implementations used as reference, (ii) multiprocessor static mappings, and (iii) self-adaptive implementation where tasks freely migrate from NPU to NPU.

*5.1. Monoprocessor Study.* Each application (FIR, DES, and MJPEG) was programmed as a monolithic task with or without the Operating System. The aim of this study is to evaluate the impact of the use of our OS on performance and also to provide a reference performance for further implementations.

Table 5 summarizes the results. The important information from these results are (1) that the impact of the Operating System on the throughput performance is relatively low (less than 0.95% for each application of our test set) and (2) that it introduces a certain level of randomness (shown by the standard deviation). As a conclusion, the OS provides low-cost multitasking capabilities and implies a very little reduced quality-of-service as the throughput is not deterministic anymore.

*5.2. Static Placement Study.* Each application of our test set was partitioned into several tasks. Our objective was to distribute the computations of a given application onto several processors in order to evaluate the impact on the throughput. This distribution was hand-made (static placement), and no migration was allowed for discarding the influence of transient phenomena. All references to task placements made throughout this section rely on the addressing mode presented in Figure 16.

*5.2.1. Data Encryption Standard.* The Data Encryption Standard (DES) algorithm is composed of several computational steps as depicted in Figure 17. All 16 rounds are functionally equivalent but operate with different keys ($K_1 \cdots K_{16}$).

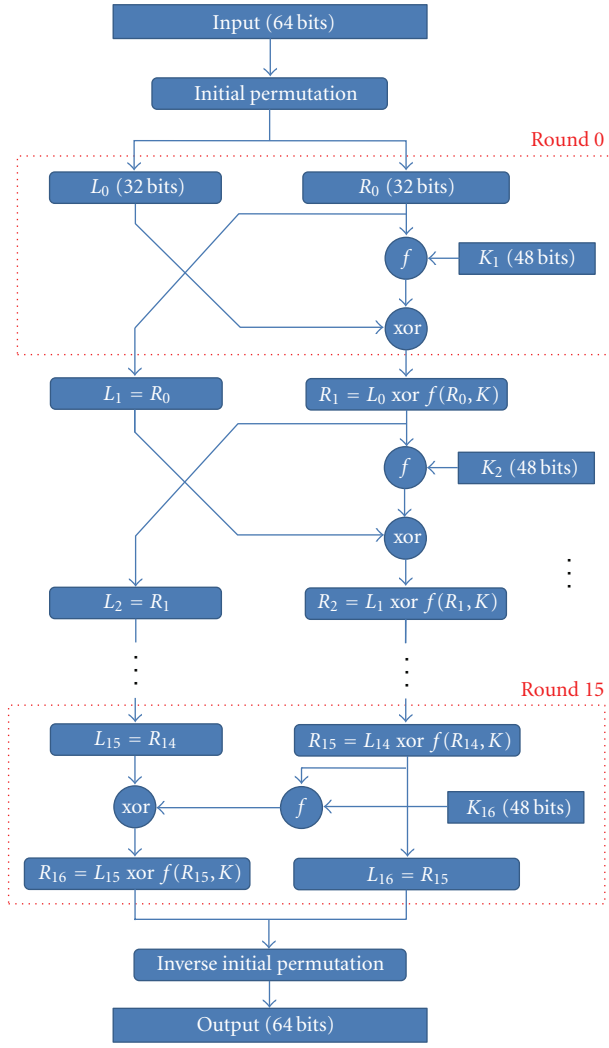



Figure 17: The DES algorithm.

We have chosen to implement it in a pipeline fashion, by decomposing the rounds (16 rounds). Figure 18 shows the DES performance results for different partitioning of the DES algorithm. We devised 4 different pipelines, with 2, 4, 6, and 8 tasks which therefore correspond to tasks embedding 8 to 2 rounds. Due the communication overhead introduced by the task partitioning, we observed that the performance is decreased when running all tasks on the same NPU. Then, when expanding the task graph to other NPUs, we observe that the throughput increases rapidly until it reaches its maximum value when $n$ tasks are mapped to $n$ NPUs. The OS overhead is generally hidden when the $n$ tasks can be mapped to $n/2$ NPU. Finally, the performance improvement of $n$ task partitioning corresponds to $n$ stage pipeline, that is, the reference throughput is at the maximum approximately multiplied by $n$.

*5.2.2. MJPEG Decoder.* Figure 19 shows the processing pipeline of a JPEG encoder operating on grey-coded images.

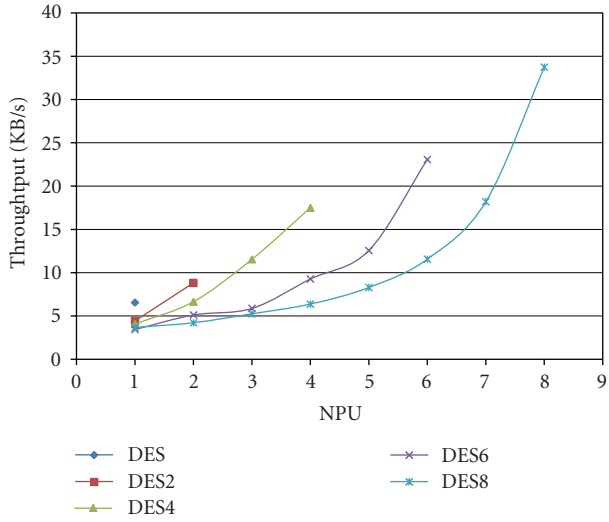



Figure 18: DES performances results with 1, 2, 4, 6, and 8 tasks.

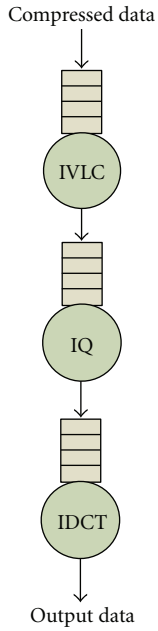



Figure 19: MJPEG data flow.

The first step of this application is the inverse variable length coding (IVLC) which relies on a Huffman decoder. This processing time for that task is data dependent. The two last tasks of the processing pipeline are, respectively, the inverse quantization (IQ) and the inverse discrete cosine transform (IDCT). The atomic data transmitted from task to task is an $8 \times 8$ pixel block which has a size of 256 bit. We naturally chose to use a traditional task partitioning as depicted in Figure 19 with both a task-level dataflow description.

Similarly to the FIR and DES applications, the operating system communication primitives induce a performance overhead when the decoder is splitted into 3 tasks (Table 6, column 1) compared to the performance shown in Table 6, column 2 (39.11 KB/second). Distributing the processing of

Table 6: Throughput evolution with task graph expansion.

| #NPU | 1 | 2 | 3 |
|---|---|---|---|
| Task placement | 3L[(a)] | 2L, 1R | 3R |
| Average TP (KB/s) | 29.21 | 39.11 | 39.05 |
| Standard deviation of TP (Kb/s) | 0.21 | 0.52 | 0.40 |

[(a)]The notations L and R refer to "Local" and "Remote" executions.

IVLC on a remote NPU (Table 6, column 3) immediately pays with a significant increase in the throughput. The fully distributed implementation exhibits a very small performance improvement when comparing to a local implementation, which is due, as we will show in the next section, to the fact that a critical task in the processing pipeline already fully employs the processing resources of a given NPU. The standard deviation, as previously observed for DES and FIR applications, increases with task partitioning and distribution.

*5.3. Dynamic Placement Study (with Migrations).* The aim of this section is to analyze the dynamic behavior of HS-Scale. We will study on the different remapping policies of the transient phenomenon in time, and the impact on performance will be measured.

*5.3.1. Diagnostic and Decision Based on Communication Load.* The first migration policy corresponds to a percentage of the FIFO utilization threshold (80% in our experiments). In this case, when the software monitor detects that the FIFO is filled over 80% for a given task, this task is automatically moved to another NPU according to a first neighbor policy.

Two scenarios DES4 and MJPEG applications are exposed in this section to prove the validity of this policy, starting from the neighbor on the east.

*5.3.2. DES4.* In this scenario, we used the DES application partitioned into 4 tasks: task 1 (T1) corresponds to the rounds 1 to 4, task 2 (T2) corresponds to the rounds 5 to 8, task 3 (T3) corresponds to the rounds 9 to 12, and task 4 (T4) to rounds 13 to 16. Figure 20 depicts the measured throughput of the application with the normal policy.

During the first seconds, the tasks (T1, T2, T3, T4) are manually sent from the external PC and placed on the NPU(1, 1). At t1 = 25.43 seconds, the DES starts the computation: the four tasks are running sequentially on the same NPU. Three migrations are performed by the policy in less than 200 milliseconds; the FIFO levels of T1, T3, and T4 are above the threshold at times $t = 26.21$ seconds, $t = 27.02$ seconds, and $t = 27.78$ seconds, respectively. After the last migration, we can see an increase of the throughput that also causes a decrease T3 FIFO filling. The throughput then stabilizes around 17 KB/s.

Comparing the performance of the last placement with the static mapping presented previously, we observe identical figures; this policy has rapidly found (1.59 seconds) one of the best placement. In order to better observe the evolution of performance in the different steps, results presented in

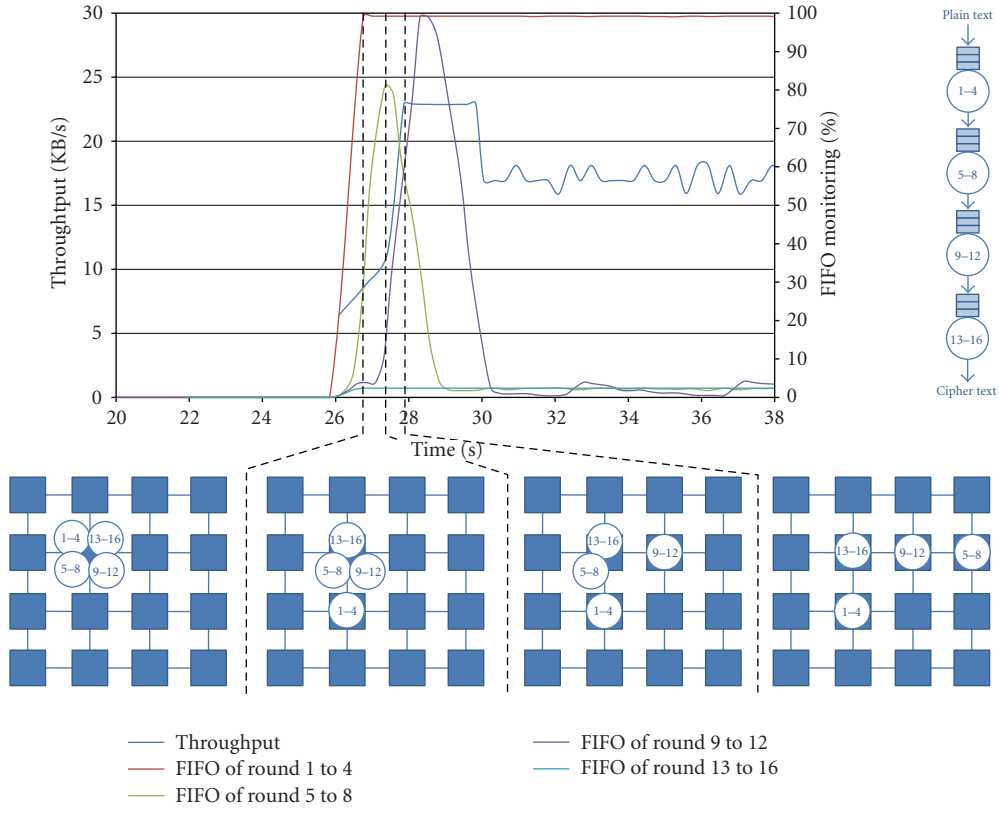



Figure 20: DES4 execution in time without delay.

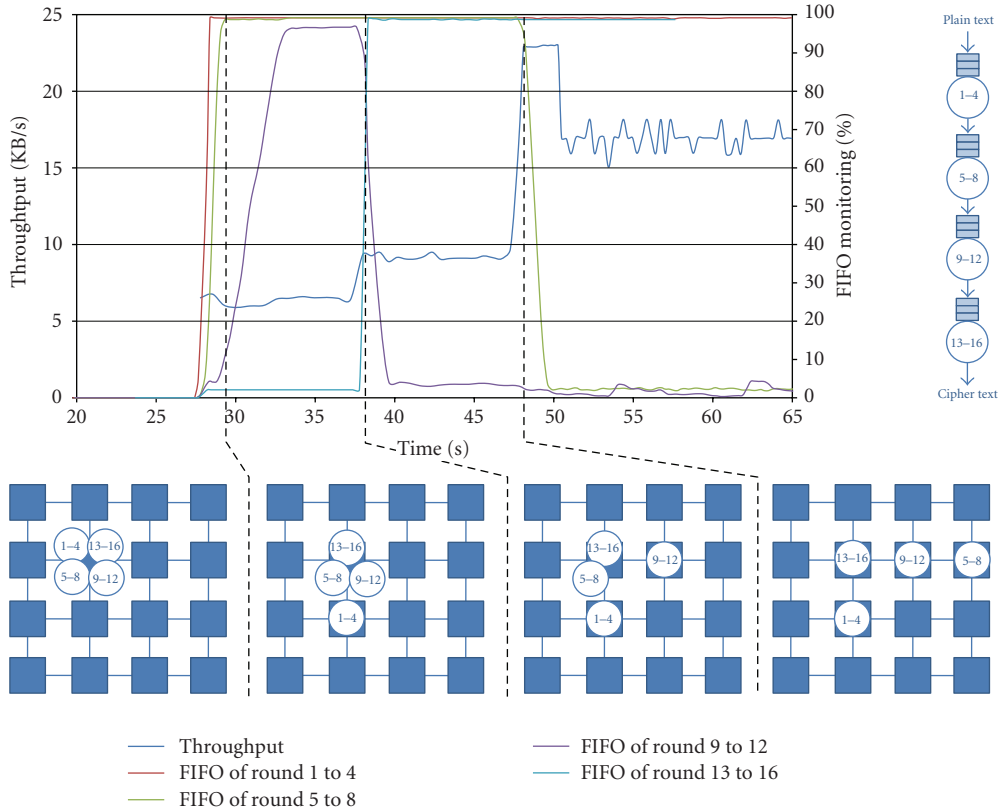



Figure 21: DES4 execution in time with delay.

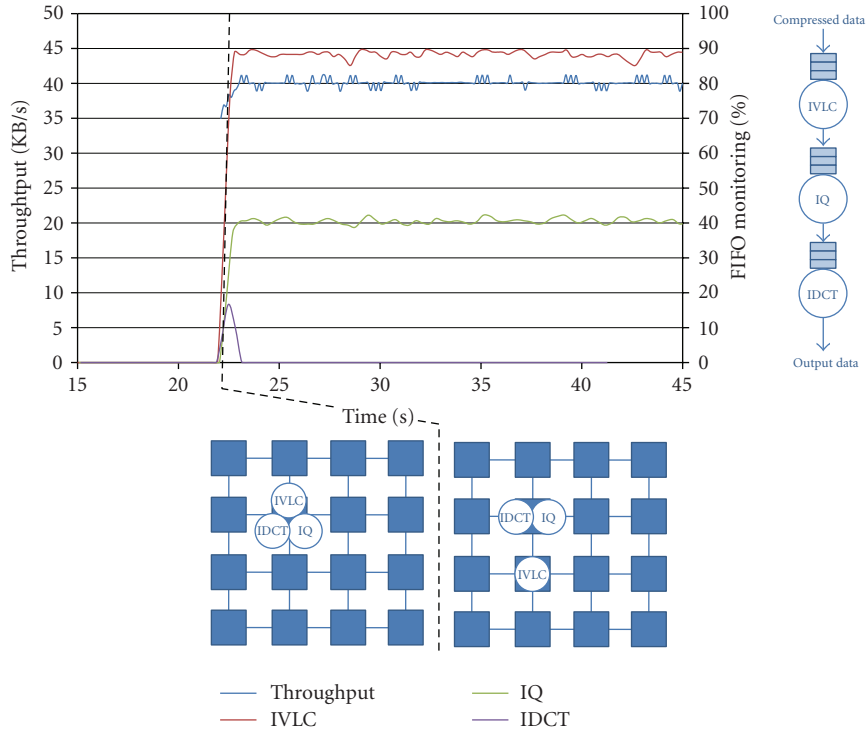



Figure 22: MJPEG execution in time.

Figure 21 were implemented using a policy that allows one migration every 10 seconds at most.

At the beginning after the start of the DES application at t1 = 28.71 seconds, all tasks are running in the same NPU, and the throughput of the application is around 4 KB/s. At t2 = 29.02 seconds, the FIFO monitor of T1 indicates a FIFO usage greater than 80%: it then begins automatically to move the task (i.e., (1) waiting a migration point, (2) looking for a free NPU, (3) migrating the task, and (4) restoring the context). Due to the migration policy, the task is placed on the NPU(1, 2). Then, the throughput stabilizes around 6.5 KB/s. After this migration, FIFO fillings of T2 and T3 increase quickly above 80% because T1 shows a higher throughput due to the fact it benefits from an entire NPU. As soon as the policy re-enables migrations (t3 = 39.03 seconds), T3 migrates on NPU(2, 1). FIFO usage of T4 then increases because of the time-multiplexing execution which results ten seconds later into another migration (t4 = 49.04 seconds).

The same final state is observed as previously with an average throughput of 17 KB/s. This clearly demonstrates for the DES application that such a simple policy is capable of rapidly converging to a best placement.

*5.3.3. MJPEG.* In this scenario, we use the MJPEG decoder application partitioned into three tasks: IVLC, IQ, and IDCT. Figure 22 depicts the application throughput and the FIFO usage of each task in the pipeline.

During the very first seconds, all tasks are instantiated manually on the NPU(1, 1). From t1 = 21.35 seconds to t2 = 22.26 seconds, these tasks are executed sequentially on the same NPU which provides an average throughput of 30 KB/s. At t2, the IVLC FIFO reaches a value greater than 80%: this leads to a migration process that involves the following steps.

(i) Freezing task execution and search for a free NPU (3.11 milliseconds).

(ii) Migrating the task (14.05 milliseconds).

(iii) Restoring the context (3.32 milliseconds).

Due to the migration policy, the task is moved from NPU(1, 1) to NPU(1, 2): it takes 20.48 milliseconds for the whole task migration process. During this time, the OS consumes CPU time for the migration process which decreases the application throughput. After the migration completion, the average throughput reaches 40 KB/s: it takes 153.6 milliseconds until a performance benefit is observed.

We then observe the following behavior of the system: on one hand, the IQ FIFO utilization remains stable meaning that it has just enough CPU time to process its data, and on the other hand the IDCT FIFO decreases meaning that it has enough CPU time to process all the data in its FIFO. In this situation, the mapping is stable from the migration policy point of view.

This clearly suggests that the policy is adequate with respect to the optimization of performance: as seen previously in Table 6 since IVLC proves the most time consuming task, therefore any further migration would not help improving performance.

*5.3.4. Diagnostic and Decision Based on CPU Workload.* The example on Figure 23 shows the results for this migration policy based on the CPU workload. The experimental

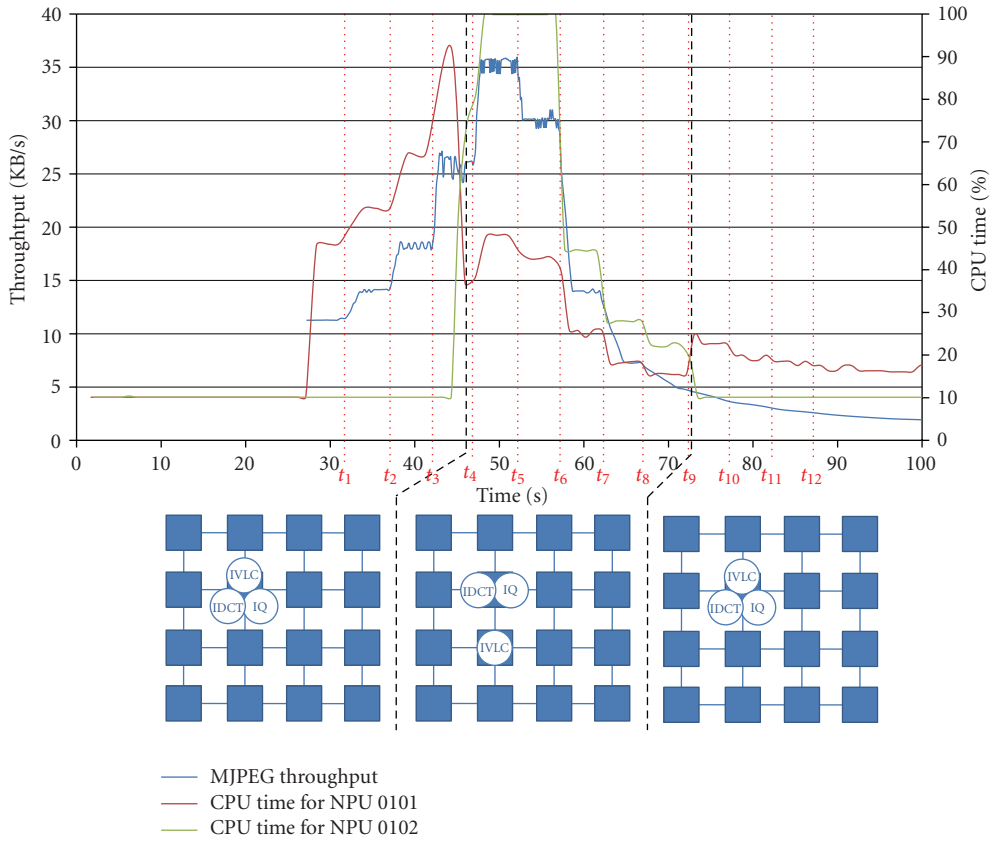



Figure 23: MJPEG decoder throughput based on CPU workload.

protocol used for these results relies on varying the input data rate for observing how the system adapts.

At the beginning all tasks (IVLC, IQ, and IDCT) are running on the same NPU(1, 1), but the input throughput on the MJPEG application is lower so the CPU time consume is around 47%. At each step t1, t2, and t3 the input throughput is increased, so we can see an increase of the CPU time consumed step by step. When the CPU time used exceed the threshold (i.e., 80%), the operating system detects that the NPU(1, 1) is overloaded (at 45 seconds) so it decided to migrate the task which uses the most of CPU time on a neighboring NPU. In this example IVLC task migrates on NPU(1, 2) which decreases the CPU time used by NPU(1, 1) around 35% and an increase of CPU used by NPU(1, 2) around 80%. At t4 the input throughput increases more which leads to an MJPEG throughput increase around 35 KB/s and overloaded the NPU(1, 2) at 100% but migration is not triggered because just one task is compute.

From t5 to t12 the input throughput of the MJPEG application is decreased step by step, and when the CPU time of NPU(1, 2) is less than 20% (at 72 seconds), the operating system decides to closer task on the same NPU (the NPU(1, 1)). We can see after this migration a decrease of CPU time used by the NPU(1, 2) and an increase of CPU time used by NPU(1, 1) but without saturate it.

We can observe that MJPEG application performances are lower than in the static mode; this is because the operating system uses more CPU time (around 10%) to monitor CPU time and average sample.

*5.3.5. Diagnostic and Decision Based on Locality.* The third migration policy corresponds to optimizing placement for decreasing Manhattan distance [23] (number of hops) between communication tasks: whenever the software monitor detects a closer placement of a given task, this task is automatically migrated to this NPU.

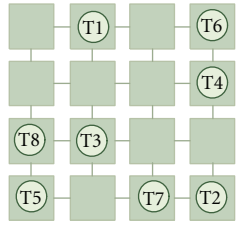
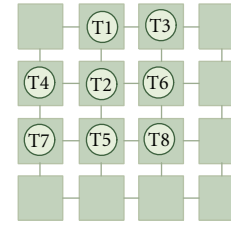
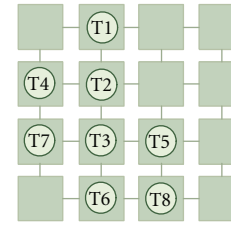
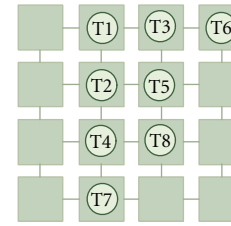
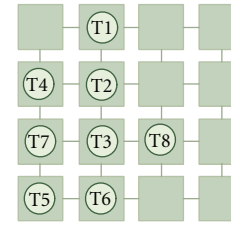
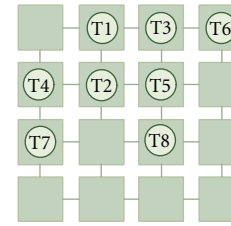
To prove the validity of this policy, we will expose results on one scenario with a dummy application that is made of numerous communicating tasks.

In this scenario, we use a dummy application which generates intensive traffic between communicating tasks. Each task consumes a different value of CPU time to compute this data. The graph task of this dummy application is presented on Figure 24.

Table 7 presents the results obtained with the policy based on the closest placement. For four initial placements, we have executed five simulations and observed the final placements obtained. This table summarizes these experiments and gives the initial placement, the final placements as well as the cumulated distance for these.

The best obtained placements exhibit 13 hops which is a satisfying result with respect to the best placement presented previously. These placements were furthermore obtained with a limited number of migrations (around 10) which suggests that deriving policies which would take into account

Table 7: Dummy application with closest placement policy.

| Initial placement | | Number of migrations | Final placement | |
|---|---|---|---|---|
| Number of hops between tasks | Graph | | Number of hops between tasks | Graph |
| 31 |  | 10 | 13 |  |
| | | 7 | 14 |  |
| | | 7 | 13 |  |
| | | 6 | 18 |  |
| | | 10 | 13 |  |

more parameters or even take sub-optimal decisions for avoiding local minima could help further improving these results.

## 6. Conclusions

Microelectronics currently undergo profound changes due to several factors such as the approaching limits of silicon CMOS technology as well as the inadequacy of the machine models that have been used until now. These challenges imply to devise new approaches to the design and programming of future integrated circuits. Hence, parallelism appears as the only solution for coping with the ever increasing demand in term of performance. Together with this, issues such as reliability and power consumption are yet to be addressed. The solutions that are suggested in literature often rely on the capability of the system to take online decisions for coping with these issues, such as scaling supply voltage and frequency for increasing energy efficiency, or testing the circuit for identifying faulty components and discarding them from the functionality.

MPSoCs are certainly the natural target for bringing these techniques into practice: provided they comply with

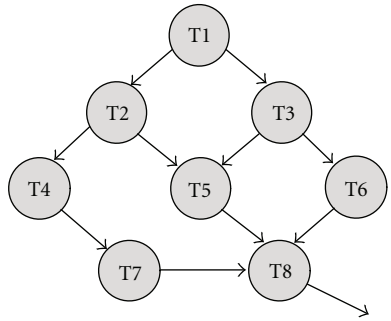



Figure 24: Dummy task graph.

some design rules they may prove scalable from a performance point of view, and since they are in essence distributed architectures, they are well suited to locally monitoring and controlling system parameters.

From the software point of view, the system presented in this paper relies on a tiny operating system that is used on every processing element. This decision certainly helps in improving system scalability as the adaptability policies that were proposed rely on a purely decentralized decisionmaking approach. The three presented policies suggest that although distributed and operating on either local or possibly non up-to-date system information, performance versus static scenarios are comparable. Finally we also demonstrated that such as solution could be viable even considering economic constraints such as silicon cost: the overhead incurred by this approach has been quantified in term of logic and memory thanks to the realization of the 16 processors prototype which is fully functional.

This paper describes an adaptive MPSoC framework that utilizes a message passing programming model. Based on information implemented in the form of distributed software monitors, the user can specify migration policies that enable the architecture to refine task placement. The conducted experiments show the following.

(i) Migration helps better balancing processor load at run-time for achieving better performance.

(ii) Task migration incurs a minimal performance overhead.

The important characteristics that have been considered are mostly flexibility, scalability, and adaptability. The remapping policies adopted show that it is possible to have a very flexible, scalable, and self-adaptable MPSoC by using monitoring systems not complex which does not affect the overhead of the system.

The proposed architecture relies on the following design decisions.

(i) Decentralized control.

(ii) Homogeneous array of processing elements.

(iii) Distributed memory.

(iv) Scalable NoC-style communication network.

Although such principles have so far demonstrated the interest in the sole context of performance, it suggests that other benefits could be achieved such as power management where highly communicating task could be mapped to neighbor processors whereas this constraint is relaxed for others. Future work aims at exploring these functionalities in connection with voltage and frequency scaling techniques since these approaches combined could reveal useful for better balancing performance and power consumption.

Another promising perspective of this work relies on task replication which helps further taking advantage of processing resources whenever needed. Although this is applicable for certain applications only, our first experiment on the MJPEG application shows that only critical task gets replicated with similar mapping/replication policies as the one presented in this paper. This should prove particularly useful for tasks which data dependant processing load such as the IVLC in our case.

Although this has not yet been demonstrated, the structure also intrinsically supports fault tolerance to a certain degree since each processor is aware of all others in the system. A faulty processor identified as such using mutual testing techniques could be discarded from the list of functional units, and further mapping decisions would therefore target the remaining processors only. Extending the control to some different system parameters such as processing elements frequency and supply voltage would certainly help further improving observed performance. Among the considered possible strategies, task replication is certainly of significant interest as it would help better matching the number of processing resources to the performance demand. This technique would prove applicable in some restricted domains (mostly for dataflow applications) but could probably be extended.

Similarly, for the growing concern of fault tolerance the developed techniques could here constitute a viable solution; since the system is entirely distributed, faulty units could be identified by others and therefore functionality would rely on the remaining processing units.

Finally, we think that adaptability is an approach that will in the near future be widely adopted in the area of MPSoC. Not only because of the here mentioned limitations such as technology shrinking, power consumption, and reliability but also because computing undoubtedly goes pervasive. Pervasive or ambient computing is a research area on its own and in essence implies using architectures that are capable of self-adapting to many time-changing execution scenarios. Examples of such applications range from ad-hoc networks of mobile terminals such as mobile phones to sensor networks systems aimed at monitoring geographical or seismic activity. In such application fields, power efficiency, interoperability, communication and scalability are primary concerns such systems have to cope with many limitations such as limited power budget, interoperability, communication issues, and finally, scalability.

## References

[1] D. E. Culler, *Parallel Computer Architecture: A Hardware/Software Approach*, Morgan Kaufmann, San Francisco, Calif, USA, 1st edition, 1998.

[2] W. M. Collier, *Reasoning about Parallel Architectures*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1992.
[3] M. J. Flynn, "Some computer organizations and their effectiveness," *IEEE Transactions on Computers*, vol. 21, no. 9, pp. 948–960, 1972.
[4] B. Nichols, D. Buttlar, and J. P. Farrell, *Pthreads Programming*, O'Reilly, Sebastopol, Calif, USA, 1996.
[5] "The OpenMP API specification for parallel programming," http://openmp.org/wp.
[6] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI: Portable Parallel Programming with the Message-Passing Interface*, Scientific and Engineering Computation Series, MIT Press, Cambridge, Mass, USA.
[7] G. Kahn, "The semantics of a simple language for parallel programming," in *IPIP Congress*, pp. 471–475, 1974.
[8] M. Saldana and P. Chow, "TDM-MPI: an MPI implementation for multiple processors across multiple FPGAs," in *Proceedings of the International Conference on Field Programmable Logic and Applications (FPL '06)*, pp. 1–6, Madrid, Spain, August 2006.
[9] J. Kohout and A. D. George, *A High-Performance Communication Service for Parallel Computing on Distributed DSP Systems*, Elsevier Science, Amsterdam, The Netherlands, 2003.
[10] J. J. Murillo, D. Castells-Rufas, and J. C. Bordoll, "HW-SW framework for distributed parallel computing on programmable chips," in *Proceedings of the Conference on Design of Circuits and Integrated Systems (DCIS '06)*, 2006.
[11] "MPCore Linux 2.6 SMP kernel and tools," ARM Limited, http://www.arm.com/products/os/linux_download.html.
[12] A. Barak, O. La'adan, and A. Shiloh, "Scalable cluster computing with MOSIX for Linux," in *Proceedings of the Linux Expo*, pp. 95–100, Raleigh, NC, USA, May 1999.
[13] J. Robinson, S. Russ, B. Heckel, and B. Flachs, "A task migration implementation of the message-passing interface," in *Proceedings of the IEEE International Symposium on High Performance Distributed Computing (HPDC '96)*, p. 61, 1996.
[14] A. R. Dantas and E. J. Zaluska, "Improving load balancing in an MPI environment with resource management," in *High-Performance Computing and Networking*, pp. 959–960, Springer, Berlin, Germany, 1996.
[15] L. Chen, C.-L. Wang, F. C. M. Lau, and K. K. Ricky, "A grid middleware for distributed Java computing with MPI binding and process migration supports," *Journal of Computer Science and Technology*, vol. 18, no. 4, pp. 505–514, 2003.
[16] S. Carta, M. Pittau, A. Acquaviva, et al., "Multi-processor operating system emulation framework with thermal feedback for systems-on-chip," in *Proceedings of the 17th Great Lakes Symposium on VLSI (GLSVLSI '07)*, pp. 311–316, Stresa-Lago Maggiore, Italy, March 2007.
[17] S. Bertozzi, A. Acquaviva, D. Bertozzi, and A. Poggiali, "Supporting task migration in multi-processor systems-on-chip: a feasibility study," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '06)*, vol. 1, pp. 1–6, Munich, Germany, March 2006.
[18] L. Benini, D. Bertozzi, A. Bogliolo, F. Menichelli, and M. Olivieri, "MPARM: exploring the multi-processor SoC design space with systemC," *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, vol. 41, no. 2, pp. 169–182, 2005.
[19] M. Pittau, A. Alimonda, S. Carta, and A. Acquaviva, "Impact of task migration on streaming multimedia for embedded multiprocessors: a quantitative evaluation," in *Proceedings of the IEEE/ACM/IFIP Workshop on Embedded Systems for Real-Time Multimedia (ESTIMedia '07)*, pp. 59–64, October 2007.
[20] MIPS corp., http://www.mips.com.
[21] F. Moraes, et al., "Hermes: an infrastructure for low area overhead packet-switching networks on chip integration," *VLSI Journal*, vol. 38, pp. 69–93, 2004.
[22] J. R. Levine, *Linkers and Loaders*, Morgan Kaufmann, San Francisco, Calif, USA, 1999.
[23] P. E. Black, "Manhattan distance," http://www.itl.nist.gov/div897/sqg/dads.

*Journal of*
*Low Power Electronics*
Vol. 6, 1–14, 2010

# Dynamic Energy Optimization in Network-on-Chip-Based System-on-Chips

Imen Mansouri[1, 2, *], Pascal Benoit[2], Diego Puschini[1], Lionel Torres[2], Fabien Clermidy[1], and Gilles Sassatelli[2]

[1] *CEA Leti - MINATEC, DRT/LETI/DACLE/LISAN 17, Martyrs, 38054, Grenoble, France*
[2] *University of Montpellier 2, LIRMM UMR CNRS, 34095, Montpellier, France*

(Received: 10 September 2010; Accepted: 4 October 2010)

Overall complexity of regular architectures is drastically increasing in nowadays System-on-Chip (SoC). Many cores are embedded in the same device and interconnected with a Network-on-Chip (NoC), providing more functionality with higher performance. The complexity of such systems introduces many challenges; one of them, probably the most important, is the power management. This paper focuses on the use of low-cost optimization mechanisms able to manage at run-time the SoC power consumption, when sequentially running applications with very different performance features, on chips with unequal characteristics due to technology variability. The optimization process is used to set the appropriate frequency of cores each time the application changes. This process has to meet some constraints. First of all, it operates intrusively on the fly to tackle changes in the system. Besides, fast response, low complexity and stability are required to have an efficient and reliable control scheme. The complexity of the optimization process has to scale with the number of cores. As a consequence, centralized optimization methods may present some drawbacks when the number of cores goes up to hundred. On the other side, distributed optimization methods can scale better, but may increase the total communications in the system. In this paper, we propose a distributed technique inspired by Game Theory (GT) to solve this optimization issue. In order to show the pertinence of the approach, we compare this solution to a centralized one based on state-of-the-art Lagrangian method. A telecom test-case application is used to compute the efficiency of the both techniques. Hardware/Software implementations of the game theoretic approach are proposed. We show that the optimization stage has an average latency of 5 ms and an area of 0.014 mm$^2$ in 65 nm technology for the hardware implementation, which is really encouraging when considering SoC constraints.

**Keywords:** MPSoC, Energy Management, Dynamic Voltage and Frequency Scaling (DVFS), Constrained Multidimensional Optimization, Game Theory.

## 1. INTRODUCTION

When scaling beyond the 32 nm node, parasitic phenomena are impacting circuits' behavior.[1] For instances, threshold voltage degradation due to Negative Bias Temperature Instability (NBTI) or electro-migration in interconnections can induce hotspots or out-of-scope features, and thus strongly reduce the component lifetime. In fact, a system can integrate cores with different performances and effects when facing the same perturbations or even a same core with different performances during its lifetime. What is at stake here is directly yield and chip area due to more margins taken, and thus the price of the chip. For such issues, adaptability, i.e., the capacity to change the functionality of the system from an external mechanism, is not sufficient. Self-adaptive schemes are required, i.e., the capacity for the circuit itself to automatically change its behavior. Self-adaptation is based on a three stages process: (i) monitoring based on sensors to detect changes in the chip; (ii) run-time algorithms to compute a new operating point and (iii) actuators to reconfigure the system.

In 2008, the International Technology Roadmap for Semiconductors (ITRS) predicted that the number of processing cores in SoC for consumer portable systems will increase reaching up to six hundred cores in the next 10 years. This projection underlines a main problem: the system scalability. Scalability is a property of a system which indicates its ability to either handle growing amounts

*Author to whom correspondence should be addressed.
Email: imen.mansouri@cea.fr

of work or to be enlarged while keeping its main figures of merit. For multi-core embedded systems, it refers to the capability of a system to increase the total computation power under an increasing load when resources are added, without exploding power consumption. An algorithm, design, networking protocol, program, or other system is said to scale if it is suitably efficient and practical when applied to large situations. We strongly believe that distributing the self-adaptation functions of multi-core architectures provides scalability.

In this paper, we address the energy management issue in multi-core architectures articulated around a Network-on-Chip (NoC). As illustrated in Figure 1, we assume distributed sensors to collect the power consumption of each core, and actuators to modify the voltage and frequency (DVFS) of each Processing Element (PE). This hypothesis of fine grain power management is made possible thanks to advanced voltage scaling techniques, such a VDD hopping.[2] To cope with the increasing scalability of NoC architectures and the unpredictable behavior of recent technologies, the energy management is based on a run-time distributed approach: a set of control loops, called "local decision makers," is attached to each PE, and operate in a coordinated manner to assign the needed frequency for each core when applications constraints vary.

In previous work,[3,4] a first approach based on Game Theory (GT) was proposed. In Ref. [3], the scalability of this solution was demonstrated on temperature profile and the synchronization between tasks on a MPSoC. Game theory was successfully used to adapt frequencies/voltages under latency and energy constraints in Ref. [4] on a real 3GPP-LTE test case simulated in Matlab. In this work, Game-Theory approach is compared to a centralized approach based on the "*Augmented Lagrangian*" method, in order to figure out the complexity of such approaches when dealing with large-scale systems. Furthermore, the distributed approach is discussed through two different implementations to prove the hardware/software feasibility of our optimization approach.

This paper is structured as follows. Section 2 provides an overview of related works; Section 3 describes the game-theoretic models and shows how this model is used for energy management in MPSoC. Section 4 gives an insight about a gradient based centralized algorithm, deployed for the same purposes, as well as some numerical comparison results. In Section 5, we propose two implementations, hardware and software, to integrate the GT technique in a concrete context. Finally, we present the efficiency of the proposed dynamic controller in terms of reactivity and area overhead.

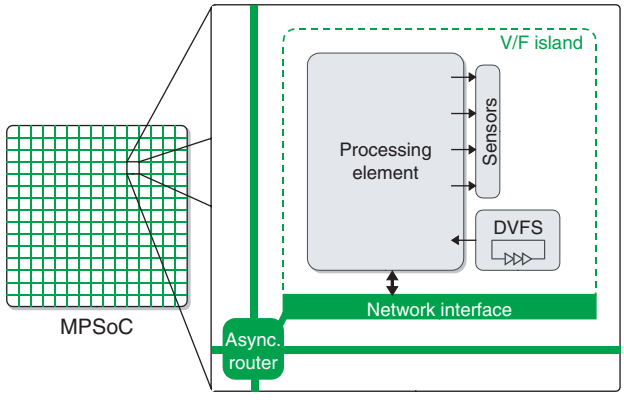



**Fig. 1.** Multi-Processor System-on-Chip.

## 2. RELATED WORKS

When optimizing embedded systems, two approaches are possible: static and dynamic schemes. In the context of this article, it is natural to focus on dynamic approaches which are the only ones providing flexible approaches and reliable designs.[5] Nevertheless, we consider that static explorations are necessary to make design-time decisions. An inspection of state-of-the-art proposals shows that most of dynamic techniques are based on centralized optimization frameworks. In the following, we survey some examples of dynamic optimization in order to understand why they do not fulfill our objectives. This analysis mainly looks at the models used and the distributed or centralized nature of possible implementations.

In Ref. [6], the frequency and voltage selection for Globally Asynchronous Locally Synchronous (GALS) systems based on Voltage Frequancy Island (VFI) is addressed. A centralized method based on non-linear Lagrange optimization is used to select the frequencies and voltages. Authors present static and dynamic algorithms. Moreover, they claim that ideally, in latency constrained systems, a global decision strategy is mandatory for an optimal voltage assignment.

In the same way, in Ref. [7], authors propose a centralized energy management engine using models inspired by Kirchoff's current law. Their approach is based on an on-line global energy managemeq2nt unit that controls the PEs through a power source and a clock generator. The authors exploit the analogy between the energy minimization problem under timing constraints in a general task graph and the power minimization problem under Kirchoff's current law constraints in an equivalent resistive network.

A convex optimization scheme has been used for temperature-aware assignment on MPSoC in Ref. [8]. Based on a complex temperature model, the problem is formulated for steady-state and dynamic-state: it consists in assigning a single frequency to each processor while maintaining the temperature and power consumption below user-defined thresholds. In steady-state, frequency and voltage are assigned once and remain constant, so without taking advantage of the dynamic nature of DVFS. In dynamic state, frequencies and voltages are changed over time allow to better optimize the system performance.

Authors formulate both scenarios as convex optimization problems. For the dynamic case, a 2-phase algorithm is used. Nevertheless, authors only present the mathematic formulation and a test-case scenario solved by a Matlab convex-optimization solver. The same authors propose in Ref. [9] to pre-calculate some valid solutions at design time by using the convex-optimization method, and to implement a control to choose at run-time the best solution for each case.

In Ref. [10], the DVFS controller allows a distributed implementation. Nevertheless, the design reacts to local queue occupancies variations without considering interactions with all the other queues in the system. This feature cannot be exploited as it clearly may lead the adaptation process to local unstable minima.

In Ref. [11], authors propose a design-time Pareto exploration and characterization combined with run-time management. They firstly make a multi-dimensional design-time exploration. The space includes costs (e.g., energy consumption), constraints (e.g., performance) and platform resources (e.g., memory usage, processors, clocks, communication bandwidth). Critical decisions are taken during a second phase by a low-complexity run-time manager implemented in the Operating System (OS). The optimization problem is modeled as a NP-hard algorithm: the multi-dimension multi-choice knapsack problem.[12] In order to find a near-optimal solution, they propose a heuristic-based OS implementation. As most of the decision is taken off-line, this solution cannot adapt to unpredictable events.

Concerning software-based optimization methodologies, an interesting survey of energy-efficient scheduling in real-time systems integrating DVFS is presented in Ref. [13]. It mainly targets scheduling techniques such as frame-based real-time task scheduling, periodic real-time scheduling, leakage-aware energy-efficient scheduling and slack reclamation scheduling. This review highlights the possibility of solving optimization problems based on software solutions. For instance, in Ref. [14] authors investigate dynamic OS-based schedulers for thermal management of MPSoCs. In Ref. [15], thermal gradients are also minimized. They focus on MPSoCs where workload is not known a priori and generally not easy to predict. The authors propose an OS-based task migration and scheduling policy that optimizes the thermal profile of the chip by balancing the system load. Authors claim to obtain significant reductions in temporal and spatial temperature variations. These software-based approaches have the advantage that they can be implemented in a distributed way. Nonetheless, these software solutions are quite heavy with many calculations, and turned out to have a large hardware overhead with limited reactivity.

Finally, in Ref. [3] game-theoretic principles are used as follows to optimize the temperature profile and the synchronization between tasks on a MP-SoC. Processing Elements (PE) are modeled as players and the temperature and synchronization are considered as metrics to build the preference functions of each PE. The convergence and optimization performances were discussed in Ref. [16]. This optimization scheme is natively distributed and scalable: the non-cooperative game concepts have been developed to describe the interaction between millions of agents making decisions in a distributed way. For these reasons, we consider it as promising model to inspire a natively distributed optimization approach.

Table I summarizes the optimization methods described previously. Several approaches have been considered, representing the directions of MPSoC optimizations. This table allows a qualitative comparison. The methods are compared regarding the off-line and dynamic phases, their complexities, and their implementations (centralized or distributed). Table I also compares the metrics used in each case as well as the actuator type.

**Table I.** MPSoC optimization synthesis.

| | EPFL | Stanford and EPFL | Carnegie melon | SUN and U. california | IMEC | LIRMM and CEA LETI |
|---|---|---|---|---|---|---|
| Reference | [7] | [8, 9] | [6] | [11, 15] | [12] | [3, 4] |
| Model | Kirchoff's current law analogy | Convex optimization | Non-linear lagrange optimization | Integer linear programming | Off-line exploration | Game theory |
| Offline phase | Yes | Yes | Yes | No | Yes | No |
| Complexity | Medium | High | High | — | Very High | — |
| Dynamic Phase | Yes | — | — | Yes | Yes | Yes |
| Complexity | Low | — | — | Medium | Medium | Low |
| Distributed Imp. | No | No | No | Possible | Possible | Yes |
| Distrib. model | No | No | No | No | No | Yes |
| Objective metrics | Energy | Performance | Energy, throughput | Hot-spot, Temp. gradient | Multiple | Mulitple |
| Constraints | Latency | Temperature | Latency | — | — | Latency, Energy |
| Actuator | Global power manager | DVFS DVFS | Vdd and freq. | OS-based task migration scheduling | OS-based | Local DVFS |

## 3. A GAME THEORETIC-APPROACH FOR MP-SoC ENERGY MANAGEMENT

Game theory is usually considered as a branch of applied mathematics. It aims at modeling the interactions between rational agents or decision makers. A game is basically a scenario composed of rational players, each one having a set of possible actions. Regarding the whole system including interactions, these actions produce consequences quantified as outcomes. In such scenario, each player makes strategic choices over the set of possible actions to execute the best one, pursuing some predefined objective in terms of outcome. The individual success of each player (the quantitative outcome) depends not only on its own choice but also on the actions chosen by other players in the game.

In this section, we present some basic definitions of GT useful to understand our approach. Then, the modeling approach is discussed to finally present the game-based optimization scheme.

### 3.1. Game-Theoretic Models

A non-cooperative game is a scenario with several players interacting by actions and consequences. Basically, the players individually choose an action within a defined set, resulting in consequences or outcomes. Each player tries to maximize its outcome according to its preferences. If this sequence is repeated, under certain conditions, the game will find an equilibrium solution with quasi-optimal outcomes.

Mathematically, non-cooperative games are represented in normal-form as follows:

$$\Gamma = \langle N, S_i, u_i \rangle, \quad \forall\, i \in N \tag{1}$$

where $N = \{1, \ldots, n\}$ is the set of $n$ players, $S_i$ is the set of actions for player $i$ and $u_i$ is a function describing its outcome.

The discrete set of actions for player $i$ is defined as:

$$S_i = \{s_{i1}, s_{i2}, \ldots, s_{im}\} \tag{2}$$

where $m_i$ is the number of possible actions for this player. Note that $m_i$ is not necessarily equal to $m_k$ for $i \neq k$ and $i, k \in N$, i.e., players can have different numbers, and thus heterogeneous actions.

The outcome of player $i$ is represented by the cost function $u_i$ coding its quantified outcome. Because of interactions with other players, this outcome is a function of the choices of current player as well as other players. Mathematically, it is represented by:

$$u_i\colon S_1 \times S_2 \times \cdots \times S_n \rightarrow \mathscr{R} \tag{3}$$

The notion of solution in a game is the way to describe how it should be played according to the preferences of each player. These descriptions specify which strategic set of actions will be played, thus predicting thus the resulting outcomes of the game. A powerful definition is the Nash equilibrium: it proposes an equilibrium point as a possible solution of a game. Nash proved that each $n$-player non-cooperative game has at least one equilibrium point, known as Nash Equilibrium (NE).[17] It can be defined by pure strategies or by mixed strategies. In the first case, players choose and play only one action among the possible set. On the contrary, in mixed strategies, the solution is chosen in a set of actions, each played with a given probability.

For pure strategies, a NE is defined as:

Definition 1. For a given non-cooperative strategic game $\Gamma = \langle N, S_i, u_i \rangle$, $\forall\, i \in N$ with $n$ players, a solution:

$$S^* = \{s_1^* \in S_1, s_2^* \in S_2, \ldots s_n^* \in S_n\} \tag{4}$$

is a Nash Equilibrium if:

$$u_i\{s_1^*, \ldots, s_i^*, \ldots, s_n^*\} \geq u_i\{s_1^*, \ldots, s_i, \ldots, s_n^*\} \tag{5}$$

$\forall\, i \in N, \forall s_i \in S_i$, being $s_i^*$ the Nash Equilibrium strategy for player $i$.

An interesting feature is when a NE is reached, that players cannot improve their outcomes by unilaterally changing their strategies. This indicates that if a NE is reached, there will be no deviation from this solution without cooperation, i.e., without making coalitions: a player in this situation cannot improve its own outcome, given the actions of the other players. It is at least a local maximum and it ensures the stability of the method.

### 3.2. Energy Consumption Model

In the context of this paper, the system under consideration is an MPSoC composed of $n$ tiles interconnected by an asynchronous NoC. Each PE integrates a DVFS engine that regulates the local voltage and frequency couple (see Fig. 1). We are interested on estimating the amount of energy needed to feed an operating tile during a given period of time. This magnitude must be optimized under certain constraints, for instance maintaining real-time performances in terms of calculation latency.

The DVFS device allows choosing among a finite number of clock periods, each one associated to the best corresponding voltage. We denote $T_i$ the clock period corresponding to the frequency of $\mathrm{PE}_i$. The energy consumption $E_i$ of the whole tile $i$ is considered during period $T_0$. For a CMOS circuit, it is given by the static and dynamic consumptions, for both memories and logic circuitries.

$$E_i(T_i) = E_{\mathrm{dyn},\,i} + E_{\mathrm{stat},\,i} \tag{6}$$

The static power dissipation $P_{\mathrm{stat},\,i}$ is considered constant. In a more complex model, this term can be defined as a function of different variables of the circuit such as

temperature, supply voltage, etc. The static energy needed to feed the tile during a given time $T_0$ is:

$$E_{\text{stat}} = P_{\text{stat},i} T_0 \tag{7}$$

The dynamic consumption of tile $i$ is a function of its activity and is modeled as follows. The task assigned to $\text{PE}_i$ is processed in $N_i$ clock periods, or in $N_i T_i$ seconds. During this period, tile $i$ has a high activity, needing a dynamic energy equal to $E_{\text{Hdyn},i}$. During the rest of the time until the task is launched again, the PE is in low-activity state, due to classical low-power techniques such as gated clocks. Thus, during $(T_0 - N_i T_i)$ seconds, it needs $E_{\text{Ldyn},i}$. This energy distribution over the time is shown in Figure 2. Note that typically $E_{\text{Ldyn},i} < E_{\text{Hdyn},i}$. Thus, the total dynamic energy required by tile $i$ becomes:

$$E_{\text{dyn},i} = E_{\text{Hdyn},i} + E_{\text{Ldyn},i} \tag{8}$$

We define $(1-\lambda_i)$ the power reduction of tile $i$ when it is using low-power mechanisms (gated clocks or similar):

$$\lambda_i = \frac{E_{\text{Ldyn},i}/(T_0 - N_i T_i)}{E_{\text{Ldyn},i}/N_i T_i} \tag{9}$$

Considering the voltage supply $V_{\text{dd}}$, the clock period $T$ and the average gate switching activity $\alpha$, the dynamic power consumption is represented by $P_{\text{dyn}} = \alpha V_{\text{dd}}^2/T$. Thus, the dynamic energy needed to feed the block during a nominal period $T_{\text{nom}}$ at a nominal voltage supply $V_{\text{nom}}$ is:

$$E_{\text{nom}} = P_{\text{nom}} T_{\text{nom}} = \alpha V_{\text{nom}}^2 \tag{10}$$

Since $\alpha$ remains constant, for a different voltage supply $V_{\text{dd}}$ we obtain $\alpha = E_{\text{nom}}/V_{\text{nom}}^2 = E_{\text{dd}}//{V_{\text{dd}}}^2$. Thus, the energy after applying DVFS becomes $E_{\text{dd}} = E_{\text{nom}}(V_{\text{dd}}/V_{\text{nom}})^2$. The circuit frequency is proportional to the supply voltage, i.e., $V_{\text{nom}} T_{\text{nom}} = V_{\text{dd}} T_{\text{dd}}$. Thus, for a given clock period $T_{\text{dd}}$, the corresponding dynamic energy $E_{\text{dd}}$ is:

$$E_{\text{dd}} = E_{\text{nom}}(T_{\text{nom}}/T_{\text{dd}})^2 \tag{11}$$

Using expression (11) in Eq. (8), we obtain the dynamic energy of tile $i$ for a given clock period $T_i$:

$$E_{\text{dyn,i}} = \left[N_i + \lambda_i\left(\frac{T_0}{T_i} - N_i\right)\right]\frac{E_{\text{nom},i} T_{\text{nom},i}^2}{T_i^2} \tag{12}$$

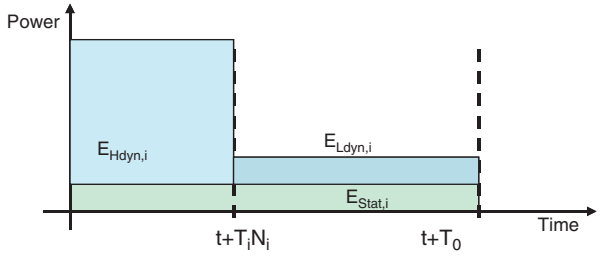



**Fig. 2.** Energy profile of $\text{PE}_i$.

Finally, replacing expressions (7) and (12) in (6), the energy required by tile $i$ when applying $T_i$ becomes:

$$E_i(T_i) = \left[N_i + \lambda_i\left(\frac{T_0}{T_i} - N_i\right)\right]\frac{E_{\text{nom},i} T_{\text{nom},i}^2}{T_i^2} + T_0 P_{\text{stat},i} \tag{13}$$

Note that $E_{\text{nom},i}$ should be measured and corresponds to the energy consumed when clocking at $T_{\text{nom},i}$. Parameter $\lambda_i$ should also be estimated considering the used technology and low-power techniques. Finally, parameters $N_i$ and $T_0$ are the number of clock periods needed to process the task and the total processing time of the given application.

### 3.3. Optimization Scheme for MPSoC Energy Management

In Section 3.1 the basic definitions of game theory were presented and in Section 3.2, a model was developed to describe the energy consumption of multi-core tiles of a parallel embedded system with fine-grain voltage and frequency controls. These two elements must now be combined to provide an optimization scheme for MPSoC energy management under calculation latency constraint. To do this, we will place an MPSoC in a game-theoretic context by modeling the PEs as players in a non-cooperative game. The latency and power consumption combination represent the local objective function which depends on other PEs global state.

As stated above, each PE integrates a DVFS engine able to set a finite number of different frequencies. We denote $T_i$ the clock period corresponding to the frequency of $\text{PE}_i$ and $T_{i-}$ as the frequency vector of all other PEs in the NoC:

$$T_{i-} = [T_1, T_2, \ldots, T_{i-1}, T_{i+1}, \ldots, T_n] \tag{14}$$

For the sake of explanation, we assume the MPSoC system running one application composed of a set of communicating tasks, each PE executing only one task. The objective is to minimize the global energy consumption of the whole system under application latency constraints.

The problem can be stated as follows:

$$\begin{aligned} \text{minimize} \quad & \sum_i U_i(T_i, T_{i-}), \quad \forall\, i \in N \\ \text{where} \quad & U_i(T_i, T_{i-}) = -(EC_i + LC_i + L_{\text{pen},i}) \end{aligned} \tag{15}$$

where $U_i$ represents the local utility function computed by $\text{PE}_i$ at the operating periods $(T_i, T_{i-})$, defined as the sum of three normalized terms characterizing the state of $\text{PE}_i$: $EC_i$, $LC_i$ and $L_{\text{pen},i}$. $EC_i$ is the energy contribution of $\text{PE}_i$ versus the whole system consumption, $LC_i$ is the local latency impact weighted by the global latency constraint, and $L_{\text{pen},i}$ is a penalty function expressing latency constraint violations in real-time system.

The energy model developed in the previous section describes the energy of each tile independently from the

other ones. For example, while tile 1 consumes energy $E_1$, tile 2 consumes $E_2$ independently of the first one for a given calculation latency. Nevertheless, when optimizing an MPSoC, we are interested not only in the tradeoff between these quantities (energy and latency) but also among different tiles. In other words, we are interested in optimizing $E_1$ together with $E_2$. Also, the energy model developed before should be modified for describing the interactions between quantities among all players in the system.

Our idea is to compare combine in the equation of the tile the quantity (energy or latency) of this tile with the total sum of the quantity available in the system. We proceed through a comparison of the contribution of a given quantity to the whole system. Mathematically, the energy contribution of $\mathrm{PE}_i$ to the total energy can be deduced:

$$EC_i(T_i, T_{i-}) = \frac{\rho_i E_i(T_i)}{\sum_k E_k(T_k)} \tag{16}$$

where $\rho_i = N_i / \sum_k N_k$ is the weight of $\mathrm{PE}_i$ in terms of number of clock cycles required over the total application requirements. Note that $EC_i$ is a function of the clock periods of all the tiles in the system.

In the same way, the latency contribution of $\mathrm{PE}_i$ to the total latency of the critical path of an application can be calculated as:

$$LC_i(T_i, T_{i-}) = \frac{\rho_i L_i(T_i)}{\sum_k L_k(T_k)} \tag{17}$$

Once again, $\rho_i$ is the weight of $\mathrm{PE}_i$ in terms of number of clock cycles required over the total application requirements and $LC_i$ depends on the clock periods of all tiles in the system. $\sum_k L_k(T_k)$ is the total latency of the application considering the critical path. The latency $L_i$ is given by the product between the clock period $T_i$ in seconds and the number of cycles $N_i$ that $\mathrm{PE}_i$ needs to process the assigned task.

The system is running a given application represented by a set of tasks introducing a computational latency $L_i$. The whole latency should not exceed $L_{\mathrm{Max}}$ for real-time purpose. We consider the critical path of the applications, meaning that the total latency is the sum of all local latencies. $L_{\mathrm{pen},i}$ is introduced to keep the total latency below $L_{\mathrm{Max}}$:

$$L_{\mathrm{pen},i}(T_i, T_{i-}) = \begin{cases} 0 & \text{if } \sum_k L_k(T_k) \le L_{\mathrm{Max}} \\ a(T_i - T_{ml}) & \text{otherwise} \end{cases} \tag{18}$$

Here, $a$ represents the penalty constant and $T_{ml}$ is the largest clock period for $\mathrm{PE}_i$ needed to satisfy latency constraint.

### 3.4. Distributed Algorithm

As explained in previous section, we model the PEs as reactive players holding the utility function $U_i$ (Eq. (15)) as its game objective. At game cycle $k$, $\mathrm{PE}_i$ starts to calculate $U_i$ for every possible $T_i$, between $T_{\min}$ and $T_{\max}$ with a granularity $\Delta T$; then, the best solution over the whole discrete range is pointed as the effective strategy for the next game cycle. This solution noted as $T^*_{i,k}$ is then applied using the DVFS engine.

In order to reduce the complexity of the algorithm implementation, we limit this search to only three values of possible $T_i$. We compare $U_i$ considering $T^*_{i,k-1}$ and its two adjacent periods $\{T^*_{i,k-1} \pm \Delta T\}$. Each PE executes this algorithm taking into account the state of the whole system resulted from the previous game. In other words, to process $U_i(k)$ the algorithm needs global information about latency and energy consumption resulted from the previous game decisions. In accordance with game theory models, we named this optimization process as Local Decision Maker (LDM).

*Local Decision Maker process*

1. Scan the network and get outcomes of previous cycle: $L_{\mathrm{tot}}(k-1)$ and $E_{\mathrm{tot}}(k-1)$.
2. Compute $\{U_i\ (T^*_{i,k-1})\ ,\ U_i\ (T^*_{i,k-1} \pm \Delta T)\}$ according to Eq. (15).
3. Compare the resulting values and find the new solution $T^*_{i,k}$

$$T^*_{i,k} = \arg\min \{U_i(T^*_{i,k-1}),\ U_i(T^*_{i,k-1} \pm \Delta T)\} \tag{19}$$

4. Stop the algorithm when a steady-state is reached.

## 4. DISTRIBUTED VERSUS CENTRALIZED ENERGY MANAGEMENT

In order to prove the efficiency of the proposed approach, it is necessary to compare it to a centralized optimization scheme. In this section, we first introduce a description of the centralized algorithm developed to solve the energy minimization problem detailed in Section 3. The method used as a reference in this paper is based on a standard gradient-type iterative strategy, commonly used to solve optimization problems in many engineering fields; it implements the "*augmented Lagrangian*."[18] Then, we compare both approaches on the same test-benches in terms of optimum quality and convergence speed for different number of PEs.

### 4.1. Reference Centralized Approach: Augmented Lagrangian Method

The Lagrangian technique permits maintaining application constraints during the energy minimization iterative process. The Lagrangian technique reformulates the constrained optimization into an unconstrained problem. The function to optimize includes both the energy consumption and a quadratic penalty term associated to the performance constraint. The penalization is expressed by the factor $\lambda$,

known as Lagrangian factor. For normalization purposes, we divide each part of this function with their initial values, $E_{\text{ini}}$ and $T_{\text{ini}}$ denote respectively the whole system consumption and latency measured in an initial state. In the following, the vector $T$ denotes the clock periods associated to a set of $n$ processors.

The new objective function becomes:

$$\text{minimize } \frac{\sum_i E_i(T_i)}{E_{\text{ini}}} + \lambda\left(\frac{\sum_i N_i T_i - T_0}{T_{\text{ini}}}\right)^2, \quad \forall i \in [1{:}n] \tag{20}$$

where $n$ is the size of the system.

The resolution of such system is strongly dependent on the value of the Lagrangian factor $\lambda$. The choice of a small $\lambda$ yields to a solution which not fulfils constraints, and on the contrary with a significant value of $\lambda$, the algorithm fails to find the optimum. The augmented Lagrangian approach was introduced to avoid this conditioning problem and to reach the exact solution independently of this factor. The Lagrange factor $\lambda$ is incremented during the optimization process to maintain the problem constraints when the program is getting closer to the optimum. A second factor noted as $\delta$ is introduced to calculate $\lambda$ increases between successive iterations.

The system (20) adapted to the augmented Lagrangian formulation, can be written as:

$$\text{minimize} \quad \frac{\sum_i E_i(T_i)}{E_{\text{ini}}} + \delta\left(\frac{\lambda}{2\delta} + \frac{\sum_i N_i T_i - T_0}{T_{\text{ini}}}\right)^2, \quad \forall i \in [1{:}n] \tag{21}$$

$$\text{where} \quad \lambda = \max\left\{\lambda + 2\delta\left(\sum_i N_i T_i - T_0\right), 0\right\}$$

In such a formulation the solution of the constrained problem is replaced by the solution of a sequence of quasi-unconstrained minimizations. A novel optimization problem is set each time the factor $\lambda$ is updated. The augmented lagrangian algorithm is described below, the index $k$ denoting the iteration number. Initially, we set the system with its best performances, $T_1$ corresponds to the rapid clock frequency of each processor, $\lambda$ and $\delta$ are fixed to 0 and 10 respectively, the constant $c$ used to increase $\delta$ after each iteration, is arbitrary chosen in the interval [0:10].

*Outline of the optimization solver*

1. solve the quasi-unconstrained problem:

$$\text{minimize } \frac{\sum_i E_i(T_i)}{E_{\text{ini}}} + \delta_{k-1}\left(\frac{\lambda_{k-1}}{2\delta_{k-1}} + \frac{\sum_i N_i T_i - T_0}{T_{\text{ini}}}\right)^2, \quad \forall i \in [1{:}n] \tag{22}$$

with $T_{i,k-1}$ as an initial solution.

2. Set $T_k$ as the solution of the sub-system (22).

3. Check if the constraint is sufficiently satisfied; if true then the iteration process terminates, and $T_k$ is the solution; else increase $\delta$ and pass to step 4.

$$\delta_k = \delta_{k-1} \times c \tag{23}$$

4. Update $\lambda$ for the next cycle according to formula (24), and then repeat the process from step (1):

$$\lambda_k = \max\left\{\lambda_{k-1} + 2\delta_k\left(\sum_i N_i T_{i,k} - T_0\right), 0\right\} \tag{24}$$

The search direction of the optimum in step (1) is not restricted by the constraints. Thus the search space is better explored and the optimization quality is guaranteed. To find the solution for each newly defined sub-problem (in step (1)), we chose to use the steepest descent method. Compared to other gradient techniques, the steepest method results in faster convergence and can operate with reasonable computing features. For example, simple descent gradient is ineffective unless the step size used is very small and it is then very slow. The steepest descent performs far better than the simple scheme by employing second-order information derived from the Hessian matrix of the objective function (Hessian matrix is the second partial derivatives of Eq. (22) in terms of $T$). On the other hand, conjugate gradient methods converge very quickly and are independent of the problem scaling, but from a computational aspect they can't be used in our context. Conjugate gradient methods require the computation of the inverse of Hessian matrix in every iteration, which is clearly time and resource consuming. The reader can refers to Ref. [18] for more detailed information on numerical optimization methods.

Before launching the steepest iterates, all the function parameters as well as the vector $T$ are set by the Lagrangian algorithm. Then the following procedure is performed to solve the sub-problem in step (1). This process is repeated as long as the maximum frequency variation among the processor is less than the DVFS resolution $R_{\text{DVFS}}$. Herein, $j$ denotes the number of iteration inside this loop, and $<, >$ is the Cartesian product in $\text{IR}^n$ space.

*Outline of the sub-problem solver*

1. Evaluate the gradient vector $G_j$ at $T_j$ and the Hessian matrix $H_j$.

2. Compute the search direction $\alpha_j$ according to the formula:

$$\alpha_j = \frac{\langle G_j, G_j\rangle}{\langle H_j G_j, G_j\rangle} \tag{25}$$

3. Update the next period vector in the direction of the gradient via the rule

$$T_{j+1} = T_j - \alpha_j \times G_j \tag{26}$$

4. Project the result on the permissible ranges: $[T_{\min}, T_{\max}]$

5. Check if max $(1/T_j) < R_{\text{DVFS}}$, then finish, otherwise go back to step (1).

**Table II.** Convergence properties of the steepest algorithm.

| | Speed (number of iterations) | | | |
|---|---|---|---|---|
| Platform size | Average | Best case | Worst case | STD* |
| 10 PEs | 31.4 | 17 | 39 | 3.11 |
| 20 PEs | 33.39 | 20 | 40 | 2.49 |
| 30 PEs | 34.27 | 24 | 40 | 2.17 |
| 50 PEs | 35.15 | 25 | 39 | 1.74 |
| 60 PEs | 35.39 | 29 | 39 | 1.7 |
| 70 PEs | 35.65 | 28 | 39 | 1.47 |
| 80 PEs | 35.61 | 28 | 39 | 1.69 |

*STD: Standard deviation value.

### 4.2. Convergence Speed and Optimization Quality of Distributed Versus Centralized Approaches

We analyze the performances of both centralized and the game theoretic based approaches, as the problem size increases using a statistical method. In order to have a fair comparison, we analyze the properties of each algorithm using the same initial configuration (default starting point) and we use the same stopping criteria. In our experiments, we generate different application graphs sized from 10 to 120 mapped to a 120-PE platform, for simplicity purposes, each PE is handling a unique task. The latency of each task in the graph and the circuit parameters (figuring in the energy model) are randomly reconfigured at each simulation following a uniform distribution. All tests are performed on a 2 GHz PC with 4 G bytes of memory. The aim of this experiment is to cover a large bunch of parameters proper to the system to optimize, so that we can better characterize each method.

Simulations show that for the same precision on the applied constraint, the principal loop of the lagrangian method maintains a nearly constant number of iterations. However, the sum of performed iterations in the secondary process (the steepest algorithm) is variable and depends on the number of processors; the gradient optimization is more sensitive to the system parameters when the number of cores is limited. These results are highlighted in Table II. As for the distributed approach, we observed that the game theoretic algorithm converges with the same number of iterations independently of the number of cores. This is due to the greedy search used to limit the computational load of our proposal (see Section 3.4). In fact, by testing just three sequential frequencies for each iteration, the convergence rate of this algorithm is constant, and the game theory approach is more likely to stagnate in the close local minima. Figure 4 summarizes optimum error by reference to *fmincon* results of Matlab. Results show that our method is efficient in most of the cases, despite the local search algorithm used.

The total time to solution for the overall algorithm (augmented Lagrangian combined with the steepest descent) is given in the second plot of Figure 3. To study the scalability of one iteration of this algorithm, we measured the average processing time needed by one iteration inside the steepest gradient. The first plot in Figure 3 shows how a single iteration of this algorithm becomes dramatically more expensive as the problem size increases. Meanwhile,

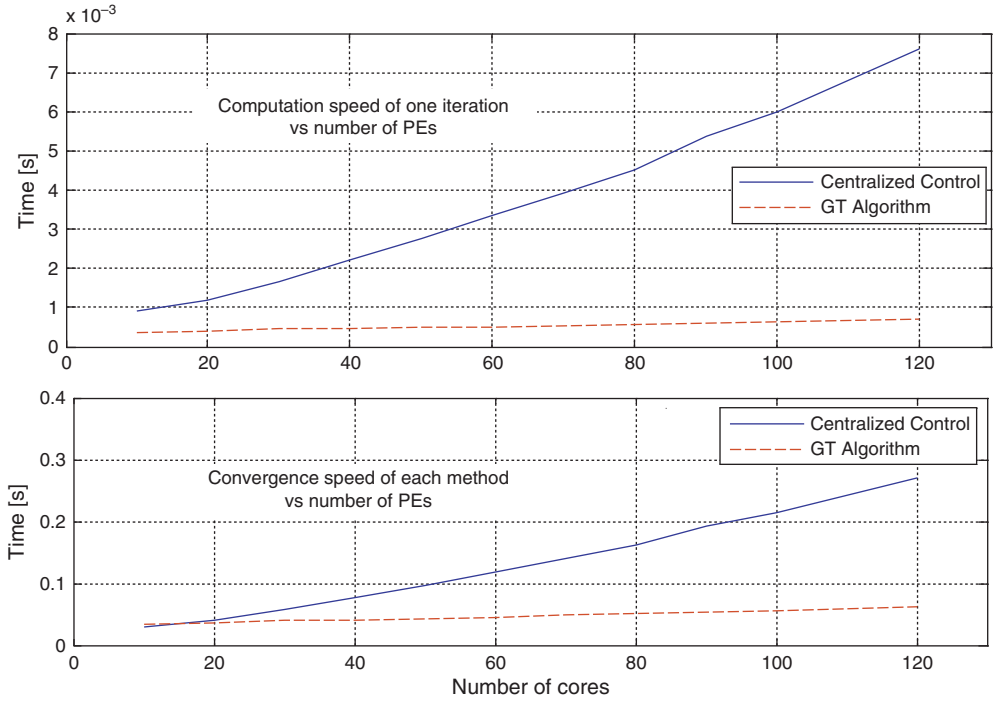



**Fig. 3.** Convergence speed of the Lagrangian and GT methods versus number of cores.

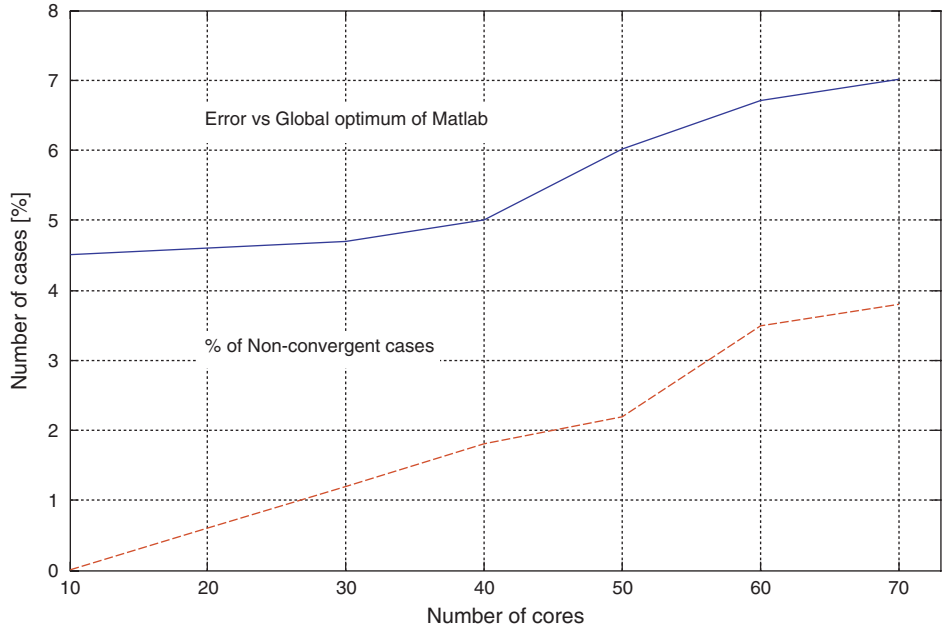



**Fig. 4.** Optimization quality of the GT method versus number of cores.

the game theoretic process converges faster in each test case, and as we expected, it is linearly scalable with the number of PE.

## 5. IMPLEMENTATION OF THE GAME-THEORETIC CONTROLLER

In the following, we analyze the efficiency and the complexity of the game theoretic approach in a concrete context. For this purpose, we develop two implementations, hardware and software schemes, to embed the Local Decision Maker (LDM) scheme. We proved the efficiency of our method through a real test-case application running under GENEPY[19] (simulated GALS platform). Finally, the cost in terms of area and reactivity of both implementations was evaluated and compared to other techniques from the state-of-the-art.

### 5.1. GENEPY Platform and Evaluation Context

The proposed algorithm was implemented on the homogeneous MPSoC platform GENEPY (see Fig. 5). This architecture is a processor array composed of DSP-centered processing elements, called SMEP units, interconnected with an asynchronous Network-on-Chip (NOC). The LDM block in Figure 5, refers to the integrated Local Decision Maker in each $\text{SMEP}_i$ running the task $i$.

This platform is based on GALS architecture with a Local Clock Generator in each unit. This distributed DVFS engine can generate frequencies in a range from 50 to 400 Mhz, by step of 30 Mhz. Each SMEP integrates two VLIW DSPs to provide a high computation performance, a Network Interface (NI) to manage data-transfer between units and a control processor to schedule the application. The control processor is a 32-bit MIPS processor[20] with

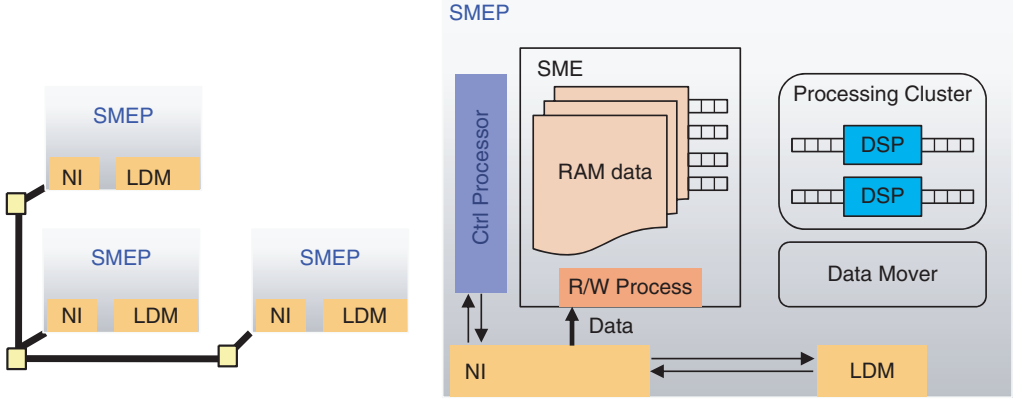



**Fig. 5.** GENEPY and SMEP units.

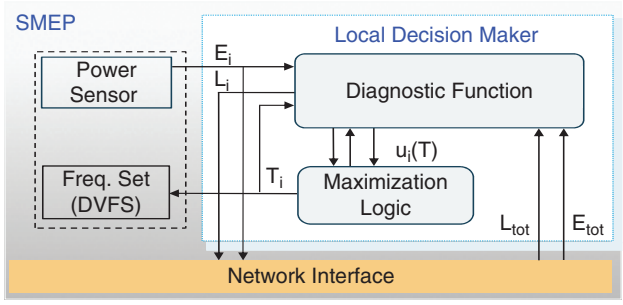



**Fig. 6.** LDM: Functional view.

several features: timer extension to check real-time constraints, and Input/Output extension to exchange control information between units. The application hosted by the GENEPY plateform, executes the demodulation part of the LTE Telecom standard,[21] it is composed of 3 tasks:

- Channel Estimation based on Wiener-filtering.
- Interpolation algorithms of the channel coefficients over the whole bandwidth.
- 1 MIMO decoder that implements a $4 \times 2$ double Alamouti algorithm.

### 5.2. The LDM Embedded Implementation

We worked on two different modules, software and hardware, to decide on which scheme is more suitable for the Local Decision Maker implementation. A functional view of the LDM is reported in Figure 6. The LDM's inputs are based on a local sensors readings and the couple ($L_{\text{tot}}$, $E_{\text{tot}}$) from the network. The controller is composed of two blocks: the diagnostic block computing $U_i$ corresponding to one frequency, and the maximization unit used to select the appropriate frequency for the next game cycle.

The LDM functionalities were first written in C-code, then cross-compiled with the MIPS-elf and loaded into the MIPS memory. The memory footprint required is about 4.7 Kb, and to limit computation resources we used the MIPS without a floating-point unit. Hence, parameters of $U_i$ were scaled at 32 bits integers, and computations were scheduled to manage precision loss due to binary calculations.

The LDM process was emulated within a Matlab simulink model, and then fixed-point precisions were introduced based on a 16-bit datapath width. A first design was automatically generated using simulink HDL coder (HW Matlab model in Table IV). We introduced pipelining to minimize the hardware, and we added memories, in this way the module frequency was improved. The modified HW module aims to compute each component of the utility function $U_i$ using a shared arithmetic unit composed of a parallel multiplier and a serial divider. Figure 7 shows the internal architecture of the hardware diagnostic block after optimization.

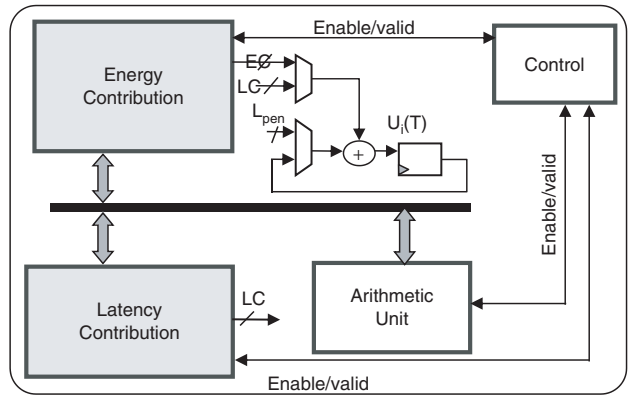



**Fig. 7.** Diagnostic module: internal HW architecture.

## 6. RESULTS

The LDM controller was used in a concrete context using GENEPY platform. Parameters characterizing the test-case application once mapped to GENEPY, are given in Table III. The nominal energy values were measured at 400 MHz in a post back-end retro-annotated *netlist*. In this section, we present results showing the dynamic behavior of the proposed method, and then we evaluate the cost of such mechanism through the SW and HW implementations.

### 6.1. Performance of the Controller

The LDM role is to adapt the system to new operating conditions as well as changes in the application demands. As illustrated in Figure 8(a), we changed periodically the global constraint (the first plot) and we observed how the system consumption is tuned to fit the applied constraint (the second plot). For instance, between iterations numbered 0 and 100, the global latency constraint

**Table III.** Test-case parameters.

| | Ch. estimation | Coef. interpolation | MIMO deco. |
|---|---|---|---|
| $E_{\text{nom}}$ [$nj$/cycle] | 92.5 | 98 | 110 |
| $N_i$ [cycles] | 91000 | 17500 | 120000 |
| $L_i/T_i$ [cycles] | 91000 | 1 | 120000 |

**Table IV.** Implementation results.

| | Model | | |
|---|---|---|---|
| Criteria | $\mu$ prog MIPS | HW from MATLAB | HW optimized |
| Frequency [MHz] | 400 | 25 | 100 |
| Game cycle duration [clk c.] | 2420 | $3 \times 142 + 35$ | $3 \times 239 + 35$ |
| Area (mm$^2$) | 0.122 | 0.061 | 0.014 |
| Precision loss | 8% | 3% | 3% |
| Resources | CPU 32 bits 32 Kb RAM | 11 mul $16 \times 16$ 3 dividers | mul $9 \times 9$ 1 divider |

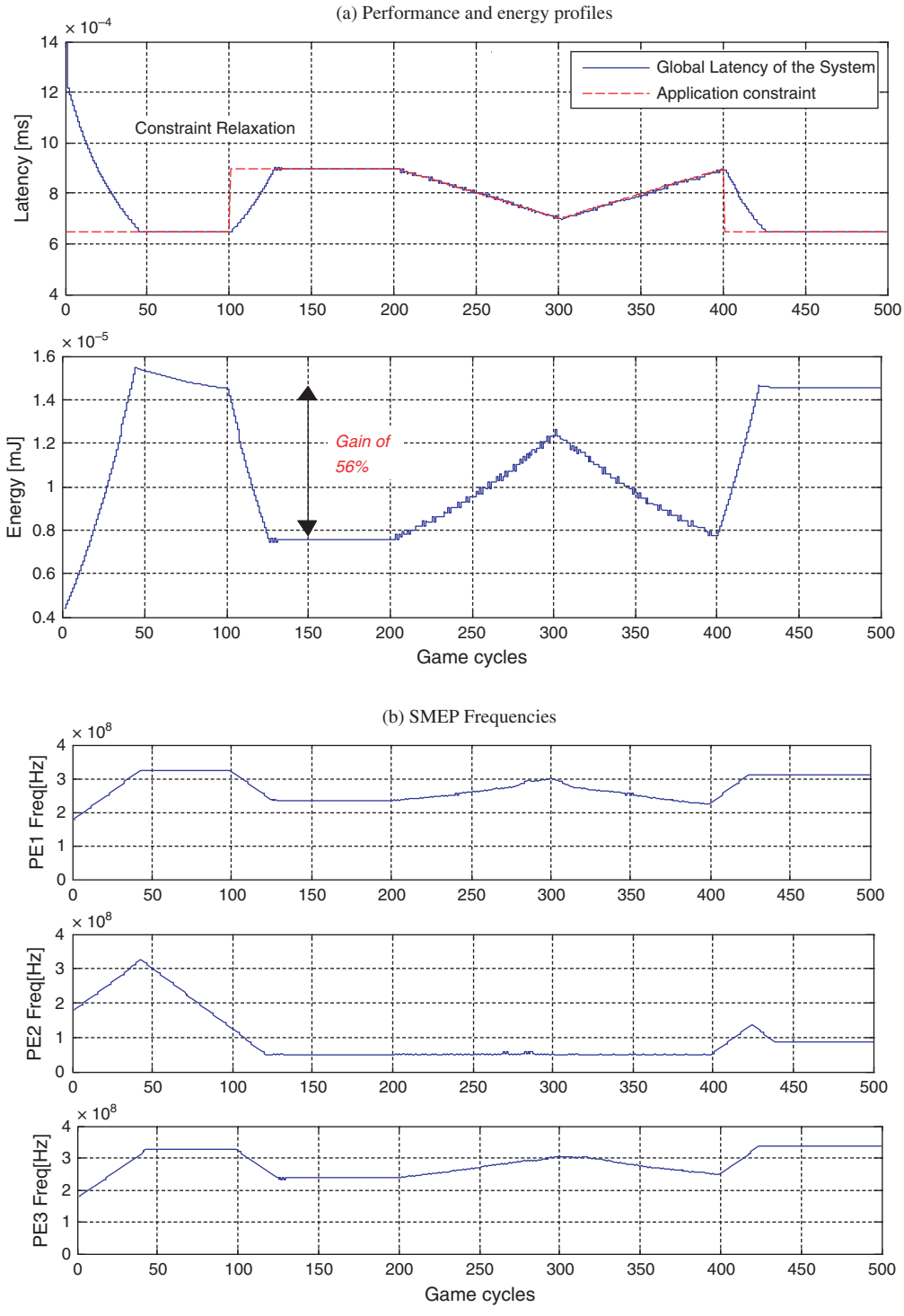



**Fig. 8.** (a) Performance and energy profiles when minimizing the system consumption under latency constraints. (b) SMEP Frequencies during the adaptation process.

increases so that PEs adapt their frequencies to tackle new changes. During iterations 100 and 200, when constraints are released, the global energy consumption is decreased by around 50% compared to operating under static conditions. Figure 8(b) describes the evolution of frequencies throughout simulation times.

### 6.2. Implementation Overhead

Communication aspects and computing load are both considered in our analysis to evaluate the LDM overhead. To collect global information from other PEs in the network, we chose to implement a simple circular scheme;

energy and latency information are transported as packets of data, their values are incremented from one SMEP to another, and once the whole NoC is scanned the accumulated data is broadcasted among the $n$ SMEP. The communication latency devoted to the LDM controller $L_{\text{Netcom}}$ for a given $k \times n$-mesh Network, is given below:

$$L_{\text{Netcom}} = 2 \cdot (kn - 1) \cdot D_m \quad (27)$$

$D_m$ denotes the mean data packet delivery delay between two neighboring LDM in GENEPY; when the LDM is embedded by the MIPS, $D_m$ was estimated to 58 clock cycles, while in the hardware implementation it was of about 7. Taking into account the communication latency besides to the needed computation time (see Table IV), one game cycle takes around 752 clock periods for the hardware dedicated module; 2420 cycles in the MIPS processor.

Synthesis results of the HW Local Decision Maker targeting 65 nm ST technology library are given in Table IV. The resulting netlist occupies about 0.014 mm$^2$ and consumes 0.57 mW (the average power consumption during one game cycle). The hardware implementation has a performance/area ratio up to a factor of 10 times better than the software alternative. Besides, it scales better for large designs, due to the increasing impact of communication latency on the SW scheme. In the other side, the software approach still attractive solution basically for two advantages: easy programmable in a high level language, it lets the opportunity to tune the algorithm's parameters whenever it is needed, besides when a processor (or microcontroller) already exists in the platform (which is the case in GENEPY) there will be no need to add extra features to the chip.

In an interval-based scheme, the LDM controller is triggered each 5000 clock cycle. Based on our analysis, this period is large enough to perform calculations and compensate the communication delay for both deployed implementations. By reference to the second curve of Figure 8, the worst convergence rate is around 100 game cycles for the considered simulation scenario. Therefore, the LDM reactivity time is estimated to 5 ms. Simulations demonstrate that the energy control process has no impact on the application execution; the overall throughput degradation is estimated to 0.17%. In Table V, we refer to the HW dedicated module performances to compare the efficiency of our energy controller with other schemes from state-of-the-art. Based on this comparison, we can conclude that our LDM achieves good performances and could be embedded in future designs.

**Table V.** Comparison table of several MPSoC controller.

| | Referenced dynamic controller | | | |
|---|---|---|---|---|
| | [12] | [7] | [6] | HW GT model |
| Optimization technique | Off-line exploration | Kirchoff's current law analogy | Non-linear lagrange | Game theory model |
| Implementation | Distributed | Centralized | Centralized | Distributed |
| Actuator | OS-based | Global power manger | Vdd and freq | DVFS |
| Latency (s) | $1.10^{-3}$ | $50.10^{-6}$ | — | $0.5.10^{-3}$ |

## 7. CONCLUSION

In this paper, we have presented a distributed scheme to optimize energy consumption in MPSoC architectures while maintaining application constraints. Our proposal inspired by game theory, aims at modeling PEs as independent players, which can adjust their local frequencies via an *in situ* DVFS engine. Each PE tries to minimize its consumption with respect to the global system performances when running a given application. Game theory concepts are used to lead the system to a global satisfying equilibrium.

We have studied the convergence rate and the scalability of a centralized approach based on the augmented lagrangian method, which is widely used to solve constrained optimization problem. As results show, the centralized algorithm computations scale poorly when the problem dimension increases. Hence, the reactivity time of such scheme increases when we consider large scale systems. Contrarily, the game theory approach has similar performances with linear scalability; even it may converge to a local minimum. Simulations show that, this decentralized method can achieve high optimization quality and requires much less computations.

Furthermore, we have shown how the GT technique could be implemented in practice to manage energy at run time. We have proposed two different schemes: software implementation ensured by a RISC processor and hardware dedicated module. Implementation results of the GT controller are promising, we have showed: a low area overhead considering the overall SMEP area for a 65 nm CMOS technology (3.1 mm$^2$), and a response time satisfying real time constraints, the time required to reach steady-state was estimated to 5 ms in the worst case by reference to our simulation scenario.

Further extensions of this work will be on the use of more localized communication scheme. Instead of gathering the global consumption and latency before each game cycle, one PE can calculate its utility function using partial information received from nearby neighbors. Our current work includes the investigation of "consensus" algorithms with limited neighborhood, which may reduce the communication overhead and improve the scalability of the optimization scheme.

## References


1. O. Khan and S. Kundu, A self-adaptive system architecture to address transistor aging **(2009)**.
2. E. Beigné, F. Clermidy, H. Lhermet, S. Miermont, Y. Thonnart, T. T. Xuan, A. Valentian, D. Varreau, P. Vivet, X. Popon, and H. Lebreton, An asynchronous power aware and adaptive NoC based circuit. *Journal of Solid State Circuit (JSSC)* 44 **(2009)**.
3. D. Puschini, F. Clermidy, P. Benoit, G. Sassatelli, and L. Torres, Temperature-aware distributed run-time optimization on MP-SoC using game theory. *ISVLSI '08: Proceedings of the 2008 IEEE Computer Society Annual Symposium on VLSI Motpellier*, IEEE Computer Society, France **(2008)**, pp. 375–380.
4. D. Puschini, F. Clermidy, P. Benoit, G. Sassatelli, and L. Torres, Dynamic and distributed frequency assignment for energy and latency constrained MP-SoC. *DATE '09: Proceedings of the 2009 Conference on Design, Automation and Test in Europe*, Nice, France **(2009)**.
5. A. K. Coskun, T. S. Rosing, K. K. Mihic, G. De Micheli, and Y. Leiblebici, Analysis and optimization of MPSoC reliability. *Journal of Low Power Electronics* 59 **(2006)**.
6. K. Niyogi and D. Marculescu, Speed and voltage selection for GALS systems based on voltage/frequency islands. *ASPDAC '05: Proceedings of the 2005 Conference on Asia South Pacific Design Automation*, ACM, New York, NY, USA **(2005)**, pp. 292–297.
7. Z. T. Deniz, Y. Leblebici, and E. A. Vittoz, On-line global energy optimization in multi-core systems using principles of analog computation. *IEEE Journal of Solid-State Circuits* 42, 1593 **(2007)**.
8. S. Murali, A. Mutapcic, D. Atienza, R. Gupta, S. Boyd, and G. D. Micheli, Temperature-aware processor frequency assignment for MPSoCs using convex optimization. *CODES + ISSS '07: Proceedings of the 5th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis,* ACM, New York, NY, USA **(2007)**, pp. 111–116.
9. S. Murali, A. Mutapcic, D. Atienza, R. Gupta, S. Boyd, L. Benini, and G. D. Micheli, Temperature control of high performance multi-core platforms using convex optimization. *DATE'08: Design, Automation and Test in Europe*, IEEE Computer Society, Munich, Germany **(2008)**, pp. 110–115.
10. Q. Wu, P. Juang, M. Martonosi, and D. W. Clark, Voltage and frequency control with adaptive reaction time in multiple-clock-domain Processors, HPCA. IEEE Computer Society, Washington, DC, February **(2005)**, pp. 178–189.
11. C. Ykman-Couvreur, E. Brockmeyer, V. Nollet, T. Marescaux, F. Catthoor, and H. Corporaal, Design-time application exploration for MP-SoC customized run-time Management. *SOC'05: Proceedings of the International Symposium on System-on-Chip*, Tampere, Finland, November **(2005)**, pp. 66–73.
12. C. Ykman-Couvreur, V. Nollet, T. Marescaux, E. Brockmeyer, F. Catthoor, and H. Corporaal, Pareto-based application specification for MP-SoC customized run-time management. *IC-SAMOS 2006*, July **(2006)**, pp. 78–84.
13. J.-J. Chen and C.-F. Kuo, Energy-efficient scheduling for real-time systems on dynamic voltage scaling (DVS) platforms. *RTCSA '07: Proceedings of the 13th IEEE Internationa Conference on Embedded and Real-Time Computing Systems and Applications*, IEEE Computer Society, Washington, DC, USA **(2007)**, pp. 28–38.
14. A. K. Coskun, T. S. Rosing, and K. Whisnant, Temperature aware task scheduling in MPSoCs. *DATE '07: Proceedings of the Conference on Design, Automation and Test in Europe*, EDA Consortium, San Jose, CA, USA **(2007)**, pp. 1659–1664.
15. A. K. Coskun, T. S. Rosing, K. A. Whisnant, and K. C. Gross, Temperature-aware MPSoC scheduling for reducing hot spots and gradients. *ASP-DAC '08: Proceedings of the 2008 Conference on Asia and South Pacific Design Automation*, IEEE Computer Society Press, Los Alamitos, CA, USA **(2008)**, pp. 49–54.
16. D. Puschini, F. Clermidy, P. Benoit, G. Sassatelli, and L. Torres, Convergence analysis of run-time distributed optimization on adaptive systems using game theory. *FPL '08: Proceedings of the 2008 International Conference on Field Programmable Logic and Applications*, Hidelberg, Germany, September **(2008)**, pp. 555–558.
17. J. F. Nash, Non cooperative games. *Annals of Mathematics* 54, 286 **(1951)**.
18. J. Nocedal and S. Wright, Numerical Optimization, Springer Verlag **(1999)**.
19. C. Jalier and al Heterogeneous versus Homogeneous MPSoC Approach for a 3G LTE Modem DATE '10.
20. Plasma CPU Model. http://www.opencores.org/ projects/mips.
21. 3GPP TSG-RAN, 3GPP TS36.211, Physical channels and modulation. (Release 8), v8.1.0 (2007-11).



### Imen Mansouri

Imen Mansouri *is a Ph.D. candidate at CEA-leti Grenoble, France. She is working jointly with the LIRMM Laboratory on Embedded system optimization and system level power modeling. She has a master degree in microelectronics from UJF-Grenoble University. Imen Mansouri received a Diploma in telecom engineering from Tunisia Polytechnic School in 2007.*

### Pascal Benoit

Pascal Benoit *received a Master Degree in Microelectronics and Automated Systems from the University of Montpellier, France, in 2001. He obtained his Ph.D. degree in Computer Engineering from the University of Montpellier in 2004. In November 2004, he joined the Karlsruhe Institute of Technology at the University of Karlsruhe in Germany where he worked as a scientific assistant. Since September 2005, Dr. Benoit is a permanent Associate Professor at the University of Montpellier 2. He teaches the design and programming of embedded systems at the "Ecole Polytechnique Universitaire de Montpellier" and performs his research activities at the LIRMM, the Montpellier Laboratory of Informatics, Robotics, and Microelectronics which is a cross-faculty research entity of the University of Montpellier 2 (UM2) and the National Center for Scientific Research (CNRS)—Department of Information and Engineering Sciences and Technologies (IEST). The objective of his research works is to design self-adaptive architectures to reduce their power consumption, compensate their technological variability and improve their reliability. He is involved in major international conferences and projects in the domain, and he's the (co-)author of more than 50 publications.*

### Diego Puschini

Diego Puschini *is a Research Engineer at CEA-LETI Minatec, Grenoble, France. He received his Dipl. Ing. degree in Electronic from "Universidad Nacional del Sur," Bahia Blanca, Argentina in 2004 and his Ph.D. in Microelectronic from University of Montpellier II, France in 2009. He has published 10 scientific contributions to world leading conferences, international journals and book chapters*

*concerning power management in distributed architectures. His research interests include energy-aware design and emergent control techniques for energy management in multi-core embedded systems.*

### Lionel Torres

Lionel Torres *obtained respectively his Master and Ph.D. degree in 1993 and 1996 from the University of Montpellier 2. From 1996 to 1997 he wad in ATMEL company as IP core methodology R&D engineer. From 1997 to 2000 he was assistant professor at the University of Montpellier 2, Polytech'Montpellier (Microelectronic design) and LIRMM laboratory. Since 2004 he is full Professor and was at the head of the Microelectronic dpt of the LIRMM from 2007 to 2010. He is now deputy head of the engineering school of the University of Montpellier 2, Polytech'Montpellier. His research interests and skills concern reconfigurable computing and system level architecture. He leads several European, national and industrial projects in this field. He is involved in different major conference as DATE, VLSI, FPL, ISVLSI, DAC and is (co)author of more than 150 publications.*

### Fabien Clermidy

Fabien Clermidy *obtained his Masters degree in 1994 and his Ph.D. in Engineering Science in 1999 on fault-tolerant architectures. He then started working at the Center for Atomic Energy (CEA) in Saclay (near Paris) in 2000 as a research engineer. In this position, he participated in the design of a massively parallel and reconfigurable computer, with fault-tolerant features. He then moved to Grenoble and became the main architect of the Network-on-Chip activity at the microelectronic laboratory of CEA. He is currently a project leader in the development of complex and energy-efficient chips for advanced telecommunication protocols applications. Fabien Clermidy has published more than 50 papers in international conferences and journals including ISSCC, JSSC, Symposium on VLSI circuits and DATE, and is the main inventor or co-inventor of 9 patents.*

### Gilles Sassatelli

Gilles Sassatelli *obtained his Ph.D. thesis in electrical engineering in 2002. He then joined the group of microelectronic systems at the Darmstadt University of Technology, Germany. He currently occupies a full-time researcher position at LIRMM, a joint research unit between CNRS and the University of Montpellier II. He conducts his research in the area of adaptive multiprocessor architectures in the flexible and reconfigurable computing group that he leads. He has published over 120 publications in a number of renowned international conferences and journals, and regularly serves as track or topic chair in the major conferences in the area of reconfigurable computing (IEEE FPL, IEEE Reconfig, Worldcomp ERSA, etc.).*

Microelectronics Reliability 54 (2014) 2344–2348

Contents lists available at ScienceDirect

Microelectronics Reliability

journal homepage: www.elsevier.com/locate/microrel

# Voltage scaling and aging effects on soft error rate in SRAM-based FPGAs

F.L. Kastensmidt [a,*], J. Tonfat [a], T. Both [b], P. Rech [a], G. Wirth [b], R. Reis [a], F. Bruguier [c], P. Benoit [c], L. Torres [c], C. Frost [d]

[a] Instituto de Informática, Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, Brazil
[b] Engenharia Elétrica, Universidade Federal do Rio Grande do Sul (UFRGS), Porto Alegre, Brazil
[c] LIRMM, Université Montpellier 2, France
[d] ISIS, STFC, United Kingdom

ARTICLE INFO

*Article history:*
Received 27 June 2014
Accepted 14 July 2014
Available online 22 August 2014

*Keywords:*
Soft error rate in FPGAs
Neutron induced soft error
Voltage and aging in FPGAs

ABSTRACT

This work investigates the effects of aging and voltage scaling in neutron-induced bit-flip in SRAM-based Field Programmable Gate Array (FPGA). Experimental results show that aging and voltage scaling can increase in at least two times the susceptibility of SRAM-based FPGAs to Soft Error Rate (SER). These results are innovative, because they combine three real effects that occur in programmable circuits operating at ground-level applications. In addition, a model at electrical level for aging, soft error and different voltages in SRAM memory cells was described to investigate by simulation in more details the effects observed at the practical neutron irradiation experiment. Results can guide designers to predict soft error effects during the lifetime of devices operating in different power supply mode.



## 1. Introduction

SRAM-based Field Programmable Gate Arrays (FPGAs) are very attractive to critical applications because they provide high logic density with reasonable performance capability. In addition, reconfigurability in the field allows to update circuit's configuration and to correct them remotely in harsh environments [1]. Aging and soft errors have become the two most critical reliability issues for nano-scaled CMOS designs. Aging is defined as a set of degeneration effects, such as Hot-Carrier Injection (HCI), electromigration, and Bias Temperature Instability (BTI) and others. Negative BTI (NBTI) affects PMOS transistors, increasing their threshold voltage and is considered to be the most important long-term effect to degrade circuit performance. It increases transistor-switching delays that may eventually lead to timing errors. The impact of NBTI on SRAM cells performance has been under research and NBTI modelling in static and dynamic operation has been investigated [2].

Various timing errors in CMOS designs can be avoided by reducing the clock frequency of the circuit during its lifetime. However, the impact of the transistor switching delay may lead to an increase in the susceptibility to neutron-induced bit-flip in SRAM memory cells due to its slow answer to signal recovery, as discussed in [3,4]. High-energy neutrons are produced in nuclear cascade showers created by nuclear spallation reactions between cosmic rays (mainly protons) and atmospheric nuclei (nitrogen and oxygen) and reach the ground level. According to the altitude and position on Earth it may have different neutron fluxes. Neutron-induced soft errors can occur due to the interaction of neutrons and the material producing secondary particles [5].

These particles can ionize the sensitive p–n junctions, which may collect some charge. If the collected charge in the struck node is higher than the critical charge ($Q_{crit}$) of the junction node, an undesirable transient current pulse is generated that is characterized as a soft error. Systems operating in harsh environments during long period of time, *e.g.* in automotive, medical, and avionic applications, are the most critical ones as they are stressed during their lifetime. In consequence, they may present a significant aging effect and must be tolerant to neutron-induced soft errors. Related works have already shown that NBTI can lead to a small increase of the Soft Error Rate (SER) in SRAM cells fabricated in 45 nm CMOS technologies [3,4,6].

However, to the best of our knowledge, the literature has not reported SER studies regarding the influence of aging in SRAM-based FPGAs, where millions of SRAM cells are used as configuration memories to customize the configurable matrix in the user's design. In this article, we investigate the SER in an SRAM-based FPGA fabricated in 45 nm CMOS technology under neutron-induced effects when accelerated aging has been performed. We compare the measured sensitive area (cross-section) of the device post aging exposed under neutron to the sensitive area reported by

* Corresponding author.
*E-mail address:* fglima@inf.ufrgs.br (F.L. Kastensmidt).

http://dx.doi.org/10.1016/j.microrel.2014.07.100
0026-2714/

the fabricant. Results show that the sensitive area can increase more than twice due to aging effects.

In addition, some FPGAs working in harsh environment may operate in systems with serious limitations of power due to its remote access. It is well known that SRAM-based FPGAs present a relative high static and idle power due to its millions of SRAM cells presented in the configurable memory bits. In order to reduce this power consumption, one common technique is to reduce the voltage supply of the entire FPGA core [7]. However, when doing that, the FPGA may be more susceptible to soft errors as well.

We analyse the combination of three different effects: soft error, aging and voltage scaling in SRAM cells at electrical simulation level to understand these three combined effects. The goal is to predict (based on a simulation method) the impact of these three effects on SRAM memory cells and further the impact in SRAM-based FPGA. We performed practical neutron-induced soft error rate experiments in SRAM-based FPGA under aging-induced variations and voltage scaling, and compare the results with the ones from the electrical simulation method. Results at electrical level could predict an increase of the soft error susceptibility by at least twice, when aging is also considered. The same proportion was measured under radiation.

## 2. Soft errors, aging and voltage scaling in SRAM-based FPGAs

SRAM-based FPGAs are complex integrated circuits composed of an array of configurable logic blocks (CLB) surrounded by a complex routing architecture, embedded memories (Block RAM), digital signal processing components (DSP), and a set of control and management logic. The CLBs are divided into slices. Each one is composed of a Look-up Table (LUT), which implements the combinational logic, and flip-flops (DFF) for the sequential elements. The routing architecture is composed of millions of pre-defined wires that can be configured by multiplexers and switches to build the desirable routing. The configuration of all CLBs, routing, DSP blocks and IO blocks is done by a set of configuration memory bits called bitstream. This one is loaded into SRAM memory cells, which are reprogrammable and volatile. According to the size of the FPGA device, the bitstream may contain millions of bits.

The selected device is the XC6SLX45-3CSG324 Spartan-6 Xilinx FPGA on a Digilent Atlys prototyping board [8]. This device is manufactured with a 45 nm technology and has a nominal core voltage of 1.2 V. The slices of this device can be divided into 3 different types. SLICEXs are the basic slices composed of LUTs and flip-flops. SLICELs include in addition an arithmetic carry structure and wide multiplexers. SLICEMs allow using the LUTs as distributed RAM and shift registers. All these resources are configured by a bitstream composed of 11,939,296 bits that are loaded into the SRAM configuration memory.

### 2.1. Radiation-induced soft error

At ground level, integrated circuits are susceptible to soft errors due to neutron effect [5]. As FPGAs have a large amount of SRAM memory cells, they are very susceptible to bit-flips in their configuration memory. According to the FPGA process technology the bitstream cells can be more or less susceptible to bit-flips. When a soft error occurs in the configuration memory, it can provoke a bit-flip. This one can change the configuration of a routing connection, the configuration of a LUT or a flip-flop in the CLB. This can be dramatic, since the bit-flip may change the functionality of it. Soft error in the configuration memory bits of an SRAM-based FPGA has a persistent effect and it can only be corrected when a new bitstream is loaded to the FPGA.

Two Soft Error Rates (SERs) can be analysed when SRAM-based FPGAs are exposed to neutron-induced soft errors. One is the static error rate, which is the number of bit-flips that occur per time in the bitstream. The other is the dynamic error rate, which is the number of errors in the design output synthesized into the FPGA. For the static error rate, the bitstream is continuously read and compared to the golden bitstream. Previous work performing static testing in this device at a neutron radiation facility [9] using an average neutron flux of $3.43 \times 10^4$ n/(cm$^2$ s) has shown a bit-flip upset rate of 0.27 upsets per minute.

### 2.2. Aging effects aging effect

Aging is a natural process that any integrated circuit suffers during its lifetime. The effects of aging can be seen as the degradation of a circuit in terms of its performance and also an increase in the leakage current. In order to analyse the effect of aging in FPGAs, it is possible to perform aging acceleration. This one is achieved by exposing the FPGA to an elevated temperature and core voltage [10]. For this purpose, the core was supplied with an external power supply at 1.8 V (above its nominal value of 1.2 V). The FPGA was heated to 80 °C using a thermal chamber, while a stress-design was in operation (to maintain a switching activity at given locations). The FPGA was configured with as array of Ring Oscillators (RO) as depicted in Fig. 1. Four gates were used for the structure of the RO, manually placed and routed in SLICEX and Switch Matrix. The manual mapping ensures an identical physical implementation of all ROs. In the following, the aging period refers to 10 days, including 7 days of effective aging and 3 days of recovery, required to clear the effects of reversible aging.

To measure the impact of aging on FPGA, the FPGA was characterized before and after aging using an ElectroMagnetic (EM) method, first proposed in [11]. In this method, the RO (Fig. 1) is successively placed in each of the 3411 CLBs of the Spartan-6 FPGA. Approximately half of the CLBs are based on one SLICEX and one SLICEL, and the other half on one SLICEX and one SLICEM (we will refer to CLBXL and CLBXM in the following). Each RO oscillates at a specific frequency, which is directly related to process, voltage and temperature ($P$, $V$, $T$).

Voltage is controlled using an external high precision DC power supply. The temperature is fixed using a thermal chamber. As demonstrated in [12], the measurement error is lower than 100 kHz with this equipment. Therefore, the variations between two different

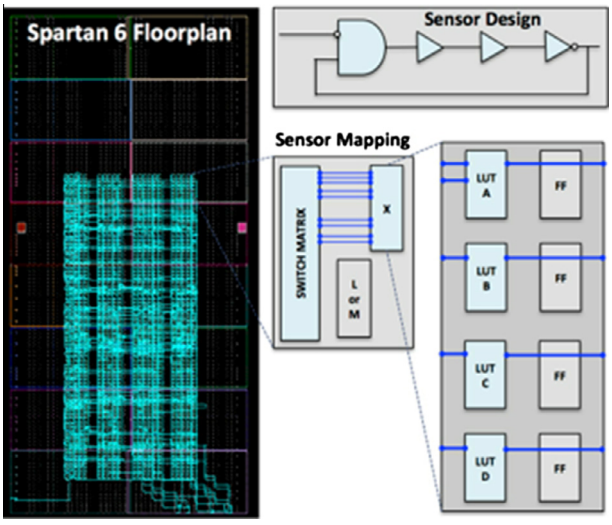



**Fig. 1.** Group of ring oscillators into the FPGA.

positions of the sensor are mainly due to process variations. During measurement, a near-field EM probe, placed over the chip, captures the EM emissions generated by the RO inside the chip. The signal from the sensor is amplified by a low noise amplifier and digitalized using an oscilloscope. An FFT is realized to obtain the frequency of each RO and then the chip cartography is build. The whole measurement system is illustrated in Fig. 2.

Characterization results obtained before the aging process are summarized in Table 1. As there are two kinds of CLBs (CLBXL and CLBXM), the mean frequencies are not the same for all the ROs. After aging, the FPGA is characterized again (Table 2). It can be noticed that the entire FPGA is affected by aging, even though from experimental results, it is clear that locations configured by ROs are more impacted. Compared to the measurements done before stress, we can observe clearly that the aging process has decreased the mean frequency of the ROs, as expected. Fig. 3 depicts the histograms of the ROs frequencies before and after aging. Both bimodal distributions have the same aspect and are only translated.

### 2.3. Voltage scaling

Scaling down the voltage in Spartan-6 FPGA reduces the mean frequencies of the RO (as all the transistors operating in a lower voltage supply also increase transistor-switching delays, as the aging effect). Voltage scaling in the SRAM cells also reduces the noise margins, which means that under neutron-induced soft errors, cells are more susceptible to flip, since the transistors that are at on state and can recover the upset node are slower and have more difficulties to restore the signal [13]. A previous experiment under neutron-induced soft errors has shown that the upset bit-flip rate can increase up to 50% for a 20% reduction in the supply voltage [9].

In this work, we analyse the bit-flip error rate under the effect of aging and voltage scaling combined by means of modelling the effects in SRAM cell at electrical simulation and by exposing the devices under neutron irradiation. These results are innovative because they associate three real effects that occur in FPGAs operating at ground-level applications and must be considered to evaluate the actual SER.

## 3. Analysing soft errors, aging and voltage scaling in SRAM cells by an electrical simulation method

An important measurement for soft-error sensitivity of a circuit is the critical charge ($Q_{crit}$), which is the minimum amount of charge to inject in a circuit node to produce a soft error [14]. The critical charge can be determined from circuit simulation of the SRAM cell. In this work, $Q_{crit}$ of a storage cell is computed by injecting a double-exponential current pulse at the storage node. Current collection is simulated at the reverse-biased drain junction of the turned-off NMOS transistor of the 6T-SRAM cells. Its amplitude is increased after each simulation, in order to find the threshold value between cell recovery and an upset. The critical charge is computed as the amount of charge injected by the current pulse.

It is important to observe that the critical charge is dependent on circuit parameters, such as bias voltage and transistor sizing, as well as device parameters, such as carrier mobility and threshold voltage. Consequently, $Q_{crit}$ is sensitive to process variability and transistor aging. In this regard, $Q_{crit}$ of modern SRAM memories are vulnerable to Bias Temperature Instability (BTI) effects due to the fact that BTI increases transistor threshold voltage, reducing current drive and, consequently, the critical charge. For digital circuits (such as SRAM cells) integrated in sub-100 nm process technologies the dominating aging mechanism is NBTI. This is due to the short times of hot carrier stress during active switching events, in comparison to the longer static time with BTI stress conditions [15]. Also, stress at high temperature tends to exacerbate the relevance of NBTI even more.

In order to account for the critical charge variability and aging effects due to NBTI (negative bias temperature instability), Monte Carlo simulations using a SPICE tool were performed to compute the critical charge distribution of 1000 6T-SRAM cells under different bias conditions. The SRAM cells were simulated using 45 nm Predictive Technology Model (PTM) transistor model [16].

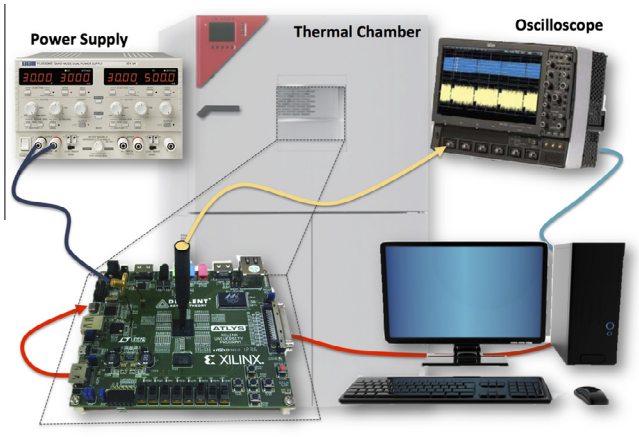



**Fig. 2.** EM measurement setup.

**Table 1**
Characterization results before aging process of ring oscillators USING CLBXL and CLBXM.

| | $F_{mean}$ (MHz) | Max var. (MHz) | 3$\sigma$ (MHz) |
|---|---|---|---|
| CLBXL | 427.1 | 20.0 | 9.2 |
| CLBXM | 414.2 | 26. 8 | 10.8 |

**Table 2**
Characterization results after aging process of ring oscillators using CLBXL and CLBXM.

| | $F_{mean}$ (MHz) | Max var. (MHz) | 3$\sigma$ (MHz) |
|---|---|---|---|
| CLBXL | 423.1 | 20.3 | 8.8 |
| CLBXM | 410.0 | 19.0 | 8.4 |

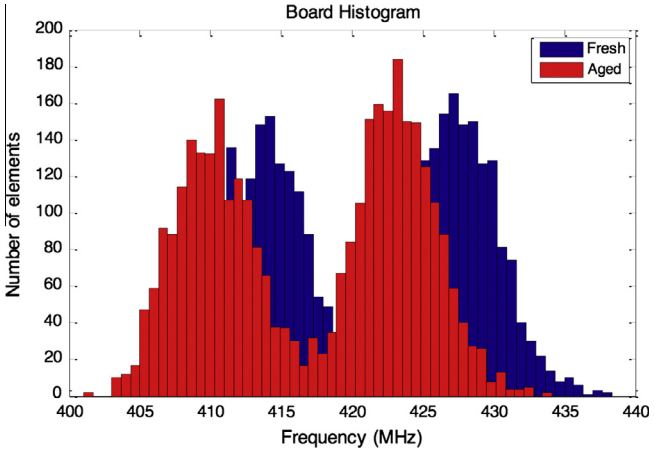



**Fig. 3.** Frequency degradation due to aging.

Transistor aging and process variations were modelled as threshold voltage shifts, whose mean and standard deviation were estimated by comparing simulation results to measurements of a number of implemented ROs during the EM analysis. Threshold voltage shifts are presented in Table 3.

The critical charge was computed only for strikes at logic 1 node. This decision is related to the fact that this node has a smaller critical charge compared to the logic 0 node, as shown by [17]. This is a consequence of the fact that the pull-up PMOS transistor usually has a weaker current drive compared to the pull-down NMOS transistor; therefore, a larger current pulse is required to flip a storage node from 0 to 1 than from 1 to 0. Also, BTI effects are more pronounced in PMOS transistors (NBTI) than in NMOS transistors (PBTI) for the 45 nm technology SRAM cell; hence, the logic 1 node is also more severely affected by BTI in this technology than the logic 0 node.

Aiming to provide an insight on how the degradation of the critical charge affects the probability of a soft error on a SRAM cell, a simplified situation may be analysed. Considering that the critical charge of a storage node is a random variable normally distributed and that the collected charge is also a normally distributed random variable, independent of the bias voltage, it is possible to determine the probability of a single event as:

$$CDF = 0.5\left[1 + erf\left(\frac{\mu_1 - \mu_2}{\sqrt{\sigma_1^2 + \sigma_2^2}}\right)\right] \quad (1)$$

where $\mu_1$ and $\mu_2$ are the mean collected charges and the mean critical charges, respectively; $\sigma_1$ and $\sigma_2$ are the standard deviations of the collected charge and of the critical charge, respectively.

For each 6T-SRAM cell, the critical charge was calculated by sweeping the amplitude of a current source injected at a storage node of the cell, and integrating the threshold current that caused a bit-flip. The current source used was modelled as a double-exponential. The mean and standard deviation of the critical charge calculated are presented in Table 4.

It is possible to observe, based in Table 4, that the mean critical charge of the SRAM cells was reduced by roughly 8.2% due to stress. At first, this result may indicate that the tolerance to bit-flip of the 6T-SRAM is mildly affected by transistor aging. That is not always the case, however, as the number of bit-flips does not increase linearly with the reduction of the critical charge. It is important to consider that the probability of a bit-flip is also related to the charge deposited by an incoming particle. This charge is not a constant value, but rather a statistical distribution. In this work, for simplicity, this distribution is assumed to be normal. Hence, the probability of a bit-flip is related to the probability of the deposited charge of an incoming particle to be larger than the critical charge of the node it strikes.

Fig. 4 shows that $Q_{crit}$ is strongly related to the bias voltage, in a trend, which is in accordance to that observed by [14]. Also, it is clear that the threshold voltage increase in the pull-up PMOS transistor due to BTI effects, degrades $Q_{crit}$ of the cell, reducing it by roughly 8% for the applied threshold voltage shift. On the other hand, there is no indication that threshold voltage shifts due to transistor aging affects the relation $\partial Q_{crit}/\partial V_{DD}$. This relation, however, is affected by the width of the current pulse employed in simulation, as presented by [14]. The width of the current pulse can be estimated through device level simulations, based on device characteristics and the Liner Energy Transfer (LET) of the incoming particle. For smaller current pulse widths, the impact of bias voltage reduction on $Q_{crit}$ is decreased, which may cause the impact of aging to dominate the critical charge degradation.

**Table 3**
Estimated mean and standard deviation of the threshold voltage shift before and after stress.

| | NMOS | | PMOS | |
|---|---|---|---|---|
| | Mean (mV) | Standard deviation (mV) | Mean (mV) | Standard deviation (mV) |
| Before stress | 0 | 15 | 0 | 15 |
| After stress | 0 | 15 | −50 | 15 |

**Table 4**
Calculated mean and standard deviation of the critical charge of 1000 6T-SRAM cells.

| | Mean (aC) | Standard deviation (aC) |
|---|---|---|
| Before stress | 287.2 | 7.4 |
| After stress | 263.7 | 8.1 |

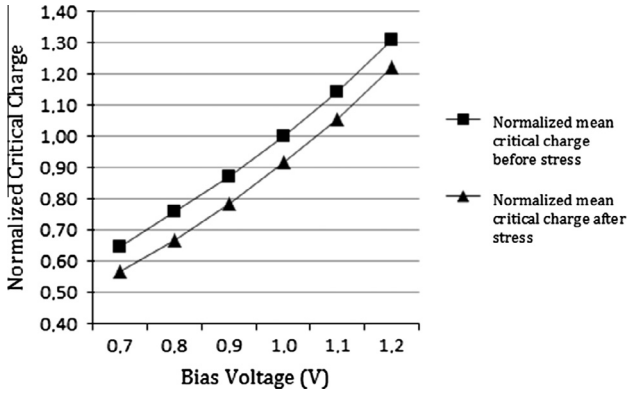



**Fig. 4.** Effect of the critical charge in a SRAM cell before and after stress (aging effect).

## 4. Neutron-induced soft error test experiment with aging and voltage scaling effects

Two FPGAs were tested: one FPGA without stress (here after "FPGA before stress") and one FPGA stressed (here after "FPGA after stress") under neutron radiation tests, performed at the ISIS facilities located at the Rutherford Appleton Laboratory (RAL) in the Didcot, UK. Irradiation was performed at room temperature with normal neutron incidence. The FPGA boards and the power supply are placed in the radiation chamber, while the computer used to monitor the test remotely is located in the control room. One USB connection is used between the FPGA board and the computer for the FPGA configuration memory readback via JTAG. The two FPGAs were irradiated with an average neutron flux of $3.43 \times 10^4 \pm 10\%$ n/(cm$^2$ s) and $4.10 \times 10^4 \pm 10\%$ n/(cm$^2$ s) respectively. The neutron fluence is calculated as the product of the average neutron flux and the time the FPGAs are exposed to the neutron flux.

The experiment consists of configuring the FPGA with a *golden* bitstream containing the test-design and then constantly read back the FPGA configuration memory with the Xilinx iMPACT tool through the JTAG interface. In the experiment control computer, the *golden* bitstream is compared against the *readback* bitstream. If differences are found, the FPGA is reconfigured with the *golden* bitstream and the differences are stored in the computer. This procedure is repeated for each core supply voltage and for both FPGAs. For both FPGAs, the errors are defined as any bit-flip in the configuration memory detected by the *readback* procedure.

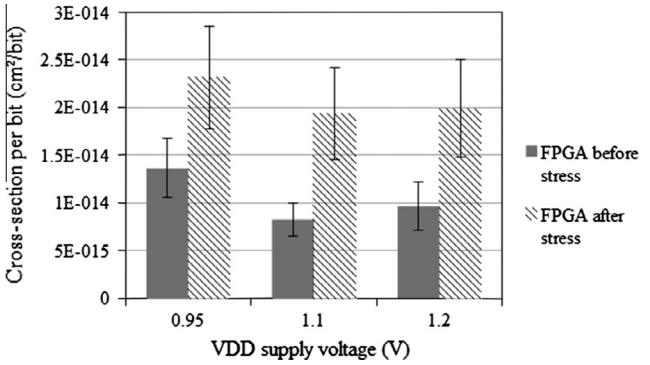



**Fig. 5.** Neutron cross-section of the configuration memory bits for Spartan-6 FPGAs before and after stress (aging effect) for different power supply modes.

With the aim of characterizing the soft error rate susceptibility of both FPGAs, the bit-flip cross-section per bit is calculated as defined:

$$\sigma_{SEU-bit} = \frac{N_{SEU}}{\Phi_{neutron} \times N_{bits}} \quad (2)$$

where $N_{SEU}$ is the number of bit-flips (SEU), $\Phi_{neutron}$ is the neutron fluence and $N_{bits}$ is the number of bits of the device, as presented in [18].

In order to evaluate the soft errors under the effects of voltage scaling, three voltages were selected including the nominal voltage value: 1.2 V (nominal), 1.1 V and 0.95 V. The experimental data for FPGA before stress were obtained from [9].

Fig. 5 presents the cross-section per bit of configuration memory bits for both FPGAs and for the three different supply voltages. The error bars are calculated using Poisson statistics. As the power supply voltage is reduced, the nominal neutron cross-section increments for both FPGAs. Moreover, we also clearly observe that the aging process impacted more significantly the cross-section than the voltage scaling. These results are consistent with the predicted results in the latter section.

## 5. Conclusions

Based on Monte-Carlo electrical simulations and neutron experiment results, we observe the influence of aging and voltage scaling to soft error rate in SRAM-based FPGAs. Results have shown that the error rate can increase more than twice when considering aging and voltage scaling, so it is important to add this type of measurement and discussions when considering FPGAs for high-reliable applications. Based on the proposed methodology based on electrical simulation, it is possible to predict the increase in error rate due to aging and voltage scaling and this can guide designers to minimize those effects.

## References

[1] Berg M. Fault tolerance implementation within SRAM based FPGA designs based upon the increased level of single event upset susceptibility. In: On-Line Testing Symposium, IOLTS; 2006.
[2] Ceratti A, Copetti T, Bolzani L, Vargas F. On-chip aging sensor to monitor NBTI effect in nano-scale SRAM. Design and Diagnostics of Electronic Circuits & Systems (DDECS). In: 15th international symposium on IEEE; 2012. p. 354–9.
[3] Bagatin M, Gerardin S, Paccagnella A, Faccio F. Impact of NBTI aging on the single-event upset of SRAM cells. IEEE Trans Nucl Sci 2010;57(6):3245–50.
[4] Cannon EH, KleinOsowski A, Kanj R, Reinhardt DD, Joshi RV. The impact of aging effects and manufacturing variation on SRAM soft-error rate. IEEE Trans Dev Mater Reliab 2008;8(1):145–52.
[5] Flament O, Baggio J, D'hose C, Gasiot G, Leray JL. 14 MeV neutron-induced SEU in SRAM devices. IEEE Trans Nucl Sci 2004;51(5):2908–11.
[6] Lin CYH, Huang RH-M, Wen CH-P, Chang AC-C. Aging-aware statistical soft-error-rate analysis for nano-scaled CMOS designs. Int Symp VLSI Des Autom Test 2013;2:1–4.
[7] Chow CT, Tsui LSM, Leong PH-W, Luk W, Wilton SJE. Dynamic voltage scaling for commercial FPGAs. Field-Programmable Technology. In: Proceedings of the international conference on IEEE, 11–14 December; 2005. p. 173–80.
[8] Xilinx Inc. Spartan-6 documentation. Available: <http://www.xilinx.com/support/documentation/spartan-6.htm>.
[9] Tonfat J, et al. Analyzing the influence of voltage scaling for soft errors in SRAM-based FPGAs. In: Presented at the 2013 RADECS, Oxford; 2013.
[10] Maiti A, McDougall L, Schaumont P. The impact of aging on a FPGA-based physical unclonable function. IEEE FPL'11: Field Programmable Logic and Applications (2011), Greece; 2011. p. 151–6.
[11] Bruguier F, Benoit P, Maurine P, Torres L. A new process characterization method for FPGAs based on electromagnetic analysis. IEEE FPL'11: Field Programmable Logic and Applications (2011), Greece; 2011. p. 20–3.
[12] Bruguier F, Benoit P, Torres L. Investigation of digital sensors for variability characterization on FPGAs. In: Proceedings of the 5th international workshop on reconfigurable communication-centric systems on chip 2010 – ReCoSoC'10; 2010. p. 95–100.
[13] Chandra V, Aitken R. Impact of technology and voltage scaling on the soft error susceptibility in nanoscale CMOS. IEEE Int Symp Defect Fault Tolerance VLSI Syst 2008;2008:114–22.
[14] Heijmen T, Giot D, Roche P. Factors that impact the critical charge of memory elements. In: 12th IEEE Int On-Line Test Symp; 2006. p. 57–62.
[15] Schlunder C, Aresu S, Georgakos G, Kanert W, Reisinger H, Hofmann K, Gustin W. HCI vs. BTI? Neither one's out. In: 2012 IEEE Int Reliab Phys Symp (IRPS), 15–19 April; 2012.
[16] Predictive Technology Model [Online]. Available: <http://ptm.asu.edu>.
[17] Dodd PE, Massengill LW. Basic mechanisms and modeling of single-event upset in digital microelectronics. IEEE Trans Nucl Sci 2003;50(3):583–602.
[18] Xilinx, Inc. Device reliability report third quarter 2013. [Online]. Available: <http://www.xilinx.com/support/ documentation/user_guides/ ug116.pdf>.

# Fine-Grain Dynamic Energy Tracking for System on Chip

I. Mansouri, P. Benoit, L. Torres, and F. Clermidy

***Abstract*—In this brief, we model system-on-chip consumption as a dynamic process that can be easily tracked over time through a set of power modes. The proposed method provides precise information on each block dissipation inside the system. Each generated mode defines a specific model that links power to block activity reported by a set of critical signals. The modeling approach is illustrated by real experiments. When applied to memory blocks, the model showed 10% maximum error, considering power at very fine granularity (quasi-instantaneous power). For long simulations, average and maximum energy are estimated with errors of 5.3% and 4.8%, respectively. For a memory controller and a multiply accumulate unit, only one probe is used leading to a very limited area overhead.**

***Index Terms*—Hidden Markov models, power system modeling, regression analysis, system-on-chip.**

## I. Introduction

ENERGY EFFICIENCY is a critical concern for battery-powered systems. Smartphones and tablets are two examples of the growing trend toward high-performance embedded system-on-chip (SoC) whose power consumption is dramatic. Circuits will have to be adapted to application requirements, so that the minimum amount of energy will be dissipated without loss of performance. One way to address this challenge is to monitor the application workload being executed, estimate the resulting dissipation, and adjust the core performance on the fly. The use of power monitors is one way to obtain instant information about core consumption (but with limited latency).

Power probes that measure current [1] are neither accurate nor fast enough to measure the exact load of the core. These bulky probes are generally applied to the whole circuit, making it difficult to analyze the consumption of each core separately and, hence, obtaining an efficient optimization strategy that would allow local configuration of each core. Another approach models power at instruction-set level. A power cost is assigned for program segments [2] or instructions [3] at finer grain. This method lacks precision and depends on the benchmarks used. During characterization, an instruction involves more than one functional block (pipeline, decoder, etc.), explaining why one particular block cannot be targeted. *In situ* event counters [4] have proved to be efficient for real-time monitoring of the activity of a core. They were first developed to better understand application behavior, facilitate debugging, and improve execution flow at run time. For this task, they are also called performance counters. The probes are a set of *in situ* registers used to record the activity (i.e., the number of events on selected signals) in the processor. Events with an architectural meaning are typically used, including instruction execution, floating point operation, memory transactions, and pipeline stalls. Performance counters, which were originally used for debugging applications, were then adapted for power estimation [5] and thermal profiling [6]. Activity counters were also used for run-time management of embedded systems. SoC is more limited in terms of area and power consumption. For that reason, monitors are much more limited in number than general purpose processors. Readings from these features come in two forms: either periodic or using interrupts. In the first case, the time period is assumed to be large enough to limit heavy communications. In the second, whenever a certain threshold is reached, an interrupt request is sent to the host system making these monitors application intrusive.

In this brief, we describe new monitoring technique that tracks power consumption at fine-grain level using a minimum of monitored events. To avoid the time dependence inherent to this kind of features, we used a dynamic modeling approach based on *hidden Markov models* (HMMs). The rest of this brief is organized as follows. Section II reviews some related works. Section III provides an overview of our method, and Section IV describes the proposed power modeling flow. In the final section, we discuss the quality and efficiency of power estimation using the new technique.

## II. Related Works

Many research efforts have focused on how monitors can be managed efficiently despite their increasing number. User-level application programming interfaces were developed to provide an abstraction layer to access these features. For instance, in the IBM BlueGene/P superprocessor, users can rotate through 52 events [4]. Similarly, Intel XScale processor contains an extended instruction set [7] to measure two events at the same time. Temporal multiplexing has been proposed to reduce the hardware overhead, which assumes that the application has unvarying behavior with respect to sampling intervals [8]. Special monitoring threads characterize gathered events with specific tags: time-indexed, process identifier, etc. Still, communication throughput is problematic and may penalize applications with hard performance constraints.

Manuscript received September 17, 2012; revised November 13, 2012; accepted March 10, 2013. Date of publication May 16, 2013; date of current version June 12, 2013. This brief was recommended by Associate Editor M. Alioto.

I. Mansouri, P. Benoit, and L. Torres are with the Laboratory of Informatics, Robotics, and Microelectronics of Montpellier, National Center for Scientific Research (CNRS), University of Montpellier II, 34095 Montpellier, France.

F. Clermidy is with Micro- and Nano-Technologies, Commission for Atomic Energy–Laboratory of Electronics and Information Technology (Laboratoire d'Electronique et de Technologie de l'Information), 38054 Grenoble, France (e-mail: pascal.benoit@lirmm.fr).

Digital Object Identifier 10.1109/TCSII.2013.2258246

Solutions advanced so far for processors are not feasible for SoC as they require a lot of power and a large number of extra hardware resources. Recently, Chen *et al.* [9] have presented special operating system drivers that allow nonintrusive collection of event data in embedded systems. Tracked events are user defined. In [10], the authors suggest to intuitively select a set of events and, then, to build several linear power models to test different subsets of events. The model with the lowest margin of error is then selected. This procedure is more painful as the circuit is more complex and less conventional. In addition, power is averaged over time after periodic readings are provided by incorporated monitors. Monitors have to be large to avoid frequent saturation, so more information about system activity is needed to compensate for the long sampling time. Once again, a tradeoff between additional area and precision has to be found when implementing this technique. Our proposal is also based on activity sensing to estimate dissipated power but uses monitors optimally and has no sampling load. Here, we propose a systematic method that tracks power variations over time and stores activity information related to each phase in a state machine. Each state is associated with one model able to present the system power even with limited information on the activity concerned.

## III. Motivation and General Methodology

Power consumption $P_T$ can be approximated over a period of time $T$ by a linear function as follows:

$$P_T = c + \alpha_1 N_{e_1} + \cdots + \alpha_i N_{e_i} + \cdots + \alpha_n N_{e_n} \qquad (1)$$

where

- $\{e_i\}_{i=1:p}$ denotes events to which counters will be connected, an event corresponds to a specific control signal;
- $N_{e_i}$ number of occurrences of $e_i$ event during sampling period $T$;
- $\alpha_i$ regression coefficient describing the power contribution of event $e_i$;
- $c$ constant term representing static consumption;
- $n$ controls model complexity.

Our goal is to provide a set of models matching running workloads instead of a single expression like in (1). This would have two advantages: First, fewer events would be monitored thereby limiting model complexity but with no loss of precision, and, second, a periodic model (in which probes are periodically scrutinized by referring to $T$) would no longer be required thus simplifying probe management. In other words, event probes act as activity containers to be emptied at the beginning of each phase.

The challenge is to find a systematic method to construct such a model, regardless of the complexity of the architecture. As can be seen in Fig. 1, our modeling flow is based on four steps. First, instrumentation is performed to collect power traces and design activity. A postsynthesis simulation is used to generate a value change dump (VCD) file containing net transitions over time. The hierarchical structure of the design is preserved during synthesis, allowing the consumption of each

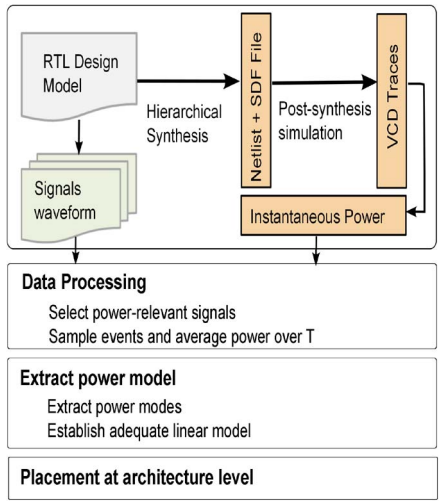



Fig. 1. Power modeling flow for SoC.

block to be tracked separately. The VCD is used as input to the *Prime Time* tool to extract instantaneous power values. In parallel, the register transfer level model of the design is simulated. Events occurring on each controlling signal are time stamped and then reported as a set of binary series. At this stage, the choice of the testbench is crucial; the model may fail to report the real system consumption if another program is run later because of the lack of meaningful data delivered by the measurement environment. In other words, if testbench coverage is insufficient (for example, a specific register array is not accessed), the power model cannot compensate for the consumption of that component. One way to overcome this constraint and to ensure enough variations within the applied workloads is to use test programs specifically developed to test the circuit functionalities.

A data processing step is then required to prepare gathered data for the modeling phase. It ensures two functions: the preselection of events and sampling. Events collected during the instrumentation step exhibit the redundancy inherent to integrated circuit design: duplicated signals, logical dependence, complemented signals, etc. Events with a high impact on the power profile must be identified among raw data for two reasons: First, redundancy might confuse the modeling process and, second, may render computation very complex. One-way analysis of variance (ANOVA) was used to evaluate the immediate impact of each event on instantaneous power consumption. Only influential events that are closely linked with the dynamics of power consumption are retained and considered to be critical in our analysis. The sampling process computes the average power $P_T$ due to circuit activity during a time period $T$. The period $T$ is set to a small number of clock cycles so that our initial hypothesis about time-independent model remains valid. $T$ is also required to process events as numeric data and not as binary series. Counting routines are added to record selected events during that period.

## IV. Dynamic Modeling of Power Consumption

Consumption modes are not observed directly but are "hidden." We can only measure power and block activity. An

HMM [11] can be used to track modes and to characterize each mode in terms of monitored events. In the following, we briefly describe some necessary backgrounds of HMMs before explaining how the tool can be used for power modeling.

### A. Background

We assume a discrete process $(Y_1, \ldots, Y_t)$ that is randomly observed at different time steps $t$. $Y_t$ evolves over time independently of previous observations, this being the standard assumption of a Markov process. HMM formalism transforms $Y_t$ observations into a sequence of instructive states denoted as $S$. HMM identifies the optimal sequence of states that best corresponds to the $Y_t$ process. An HMM can be defined as a four-tuple $\Theta = (S, \Pi, E, (\pi_i))$, where

- $S$ — finite set of states specifying necessary rules from which an approximation of $Y_t$ can be deduced, $S \in \{S_1, \ldots, S_m\}$;
- $\Pi$ — $m \times m$ state transition matrix with its $ij^{\text{th}}$ entry denoting the probability of passing from state $S_i$ to state $S_j$;
- $n \times m$ — emission matrix denoted $E$, which determines the probability that a particular observation $Y_t$ will be emitted given state $S$;
- $\pi_i$ — initial probability vector predicting the occurrence of each state $S_i$, $(\pi_i) \in \mathbb{R}^m$ will be refined during model training.

Let $\widetilde{Y}_t$ denote an approximation of $Y_t$ generated by our model. The analytic expression of $\widetilde{Y}_t \backslash S_i$ varies when the system switches between different states. An iterative procedure is used to adjust $\Theta$ parameters so that $\widetilde{Y}_t$ fits $Y_t$ observations satisfactorily. In [12], HMM was extended to deal with regression plans. Herein, each state $S$ describes a linear relationship between $Y_t$ and a set of explanatory variables recorded at time $t$.

### B. Power Modeling Flow

Mapped to our problematic, a Markov state can be seen as a specific consumption mode or regime characterized by a set of regressive parameters linking power to monitored events. Power measurements depict our Markov process denoted as $P_t$. Events are listed in $n$-dimensional time-series vector $X_t = (e_{t1}, \ldots, e_{tn})$, where $n$ stands for the number of preselected events. Power and events are extracted from real experiments on the chip based on the previously described flow. Events active at low level (i.e., when set to zero) are negatively correlated with power, to get positive functions they were inverted during computations.

We assume $m$ power modes $S_i$ governing the consumption behavior of the design during our experiments. Regressive parameters related to mode $S_i$ are enclosed in a constant vector labeled $\alpha_{S_i}$, $\alpha_{S_i} \in \mathbb{R}^{n+1}$. $\alpha$ dimension is incremented by one to include a constant term (for static power and error correction) in addition to the weights of $n$ events. To simplify notations, we use $\alpha$ as a reference for the matrix holding $m \times \alpha_{S_i}$ vectors.

Dynamic power is defined as $P_T = \mu V^2/T$, where $\mu$ is the average gate switching activity, $T$ is the clock period, and $V$ is voltage supply. $\alpha$ matrix is calculated for one operating point (frequency–voltage couple $F_1 - V_1$). Since $\mu$ remains constant for different operating points, our model based on sensing design activity can be easily adapted to deal with different voltage/frequency configurations (even corner values $F_2 - V_2$). Regression coefficients have to be scaled using the ratio of running frequency to modeling frequency $(F_2/F_1)$; for voltage, quadratic values $(V_2/V_1)^2$ were used. Static power can be extrapolated to match new configurations.

Based on previous definitions, power model at time $t$ is structured as follows: $P_t = X_t^{\mathrm{T}} \times \alpha_{S_t} + \epsilon_{S_t}$, where $\epsilon_{S_t} \approx N(0, \sigma_{S_t}^2)$, $\epsilon_{S_t}$ is a white noise error with variance $\sigma_{S_t}$. The goal of the Markov model is to define $\Theta^*$ by maximizing the fitness function given by likelihood equation (2)

$$Ln(P_1, \ldots, P_p; \Theta) = \Pi_{i=1}^{p} P_\Theta \left( \{\widetilde{P}_i = P_i\} \right) \tag{2}$$

where $p$ is the number of sampled data $((P_t), (X_t)$ with $t = 1 : p)$, $P_\Theta(\{\widetilde{P}_i = P_i\})$ is the probability that $\widetilde{P}_i$ is equal to $P_i$ given parameters $\Theta$. $\Theta^*$ is computed for the logarithm of (2) since the derivative function of $\Pi_{i=1}^{n}$ is simpler. Hence, $\Theta^*$ corresponds to the optimum of

$$L\left(\Theta; (Y_t); (X_t)\right) = \sum_{t=1}^{p} \sum_{S_i} \log \left( \widetilde{P}_i \cdot P(S_t \setminus \Pi) \right). \tag{3}$$

$P(S_t \backslash \Pi)$ is the probability that $S_t$ occurs given the transition matrix $\Pi$. Assuming that the system is determined by state $S_t$ at time $t$, $\widetilde{P}_i$ is an estimation of $(P_t)$ given $(X_t)$ with regard to $\alpha$ parameters. We use a special Matlab package called *MS_Regress* [13] to solve this complex optimization problem. The design shifts from one power regime to another in response to a particular critical event. A brief analysis of the design functioning explains these transitions. This dynamic is also expressed through the transition matrix $\Pi$. The number of states $m$ is progressively incremented until one mode is captured but with no real significance. Probabilities extracted from matrix $\Pi$ reflect the significance of each mode.

## V. Validation and Simulation Results

We applied our proposal to a very long instruction word DSP core particularly designed for Telecom applications [14]. The blocks we focused on to validate our approach were the multiply accumulate (MAC) and random access memory (RAM) inside the DSP core. Two algorithms were executed to extract raw data and then to evaluate the results: channel estimation and interpolation programs of the 3rd generation partnership project-long term evolution standard.

### A. Extracted Models

Input data used to model RAM consumption are power and transitions across one important signal revealed by *ANOVA* and named *chip select*. That signal was always active whenever the RAM was accessed. It is possible to monitor more signals, but our aim was to prove that the concept of many regimes guarantees acceptable performance even with minimum resources. That is why we choose to work with only one signal

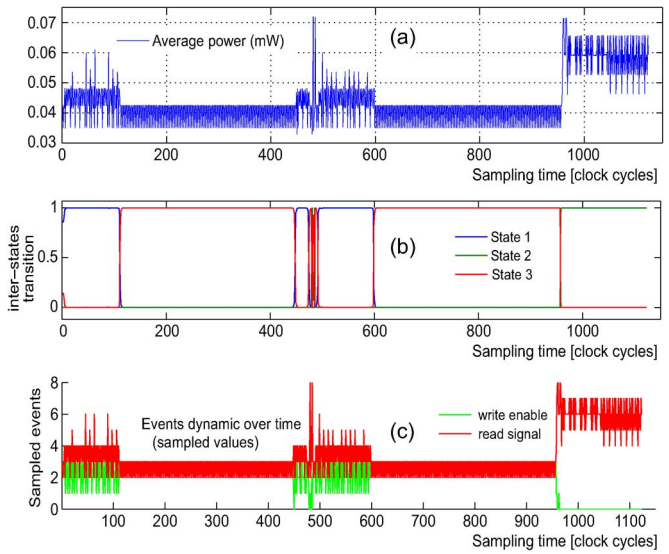



Fig. 2. (a) RAM power (in milliwatts), (b) extracted Markov modes, and (c) monitored events, versus sampling time (sampling period is eight clock cycles).

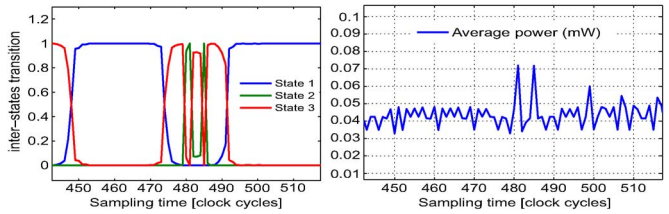



Fig. 3. Close-up of interstate transition and corresponding dynamics of the power consumed (in milliwatts) by the RAM.

(to prove the efficiency of the technique but also to limit the number of counter to implement). We investigated fine-grain sampling period $T$ of eight clock cycles during data processing. The modeling process generated three modes $(m = 3)$ for the RAM. The second subplot in Fig. 2 shows the sequencing of captured modes. Transition probabilities $\Pi$ are equal to the identity matrix, which means that, in each execution phase, only one mode dominates. To judge the significance of the modes, we examine the rest of the signals not used for model processing, namely, read and write enable signals. Mapping these signals along with the sequence of the states showed that state 3 corresponds to a writing phase, state 2 corresponds to reading only, and state 1 denotes alternating reading and writing requests to the RAM. The power weight associated with *chip select* activity characterizes each regime. Fig. 3 is a zoom on $x$-axes between sampling times 450 and 500 where three states overlap showing how our model is able to track each regime at a fine-grain level although only one signal is monitored.

The same approach was used when modeling MAC consumption. One signal-enabling activity on that block was introduced to establish power modes. As can be seen in Fig. 4, three significant modes were recorded. They can be interpreted as follows.

1) State 1: Indicates that two multipliers inside the MAC are in use.
2) State 2: Denotes low MAC activity (as in standby).
3) State 3: When one multiplier is active.

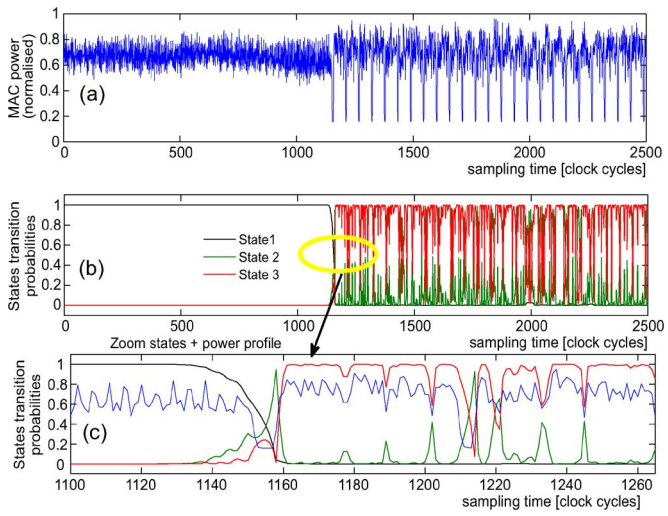



Fig. 4. (a) Dynamic state sequencement matching MAC power profile. (c) States probabilities highlight dominant power mode through sampling time.

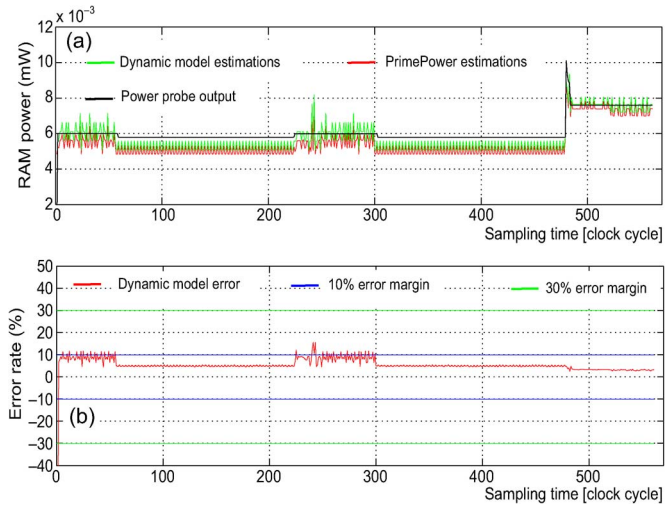



Fig. 5. (a) RAM test case: Dynamic model and resulted power probe performances tracked at very fine-grain level. (b) Subplot presents error profile.

Signals controlling each multiplier define transitions between detected states. Here, again, these signals were excluded when running the modeling flow, and we only used them to interpret findings. Aside from states, $\alpha^*$ matrix is derived by means of a training phase leading to the best fit with power and event traces. The resulting models are evaluated in the following section.

### B. Validation and Simulation Results

Previous models can be transformed into logic probes in different ways. In the case of RAM and based on previous findings, we used a state machine with three entries. Only one counter was attached to chip_select. Finally, we needed three registers to store the counter output associated with each state. In this way, we were easily able to construct the historic consumption profile depicted in Fig. 5, which shows the efficiency of the dynamic model in reporting accurate power densities. The dynamic model error fluctuates between 3% and 10% (equivalent to 0.5 $\mu$W); here, bear in mind that we are tracking quasi-instantaneous power at very fine granularity.

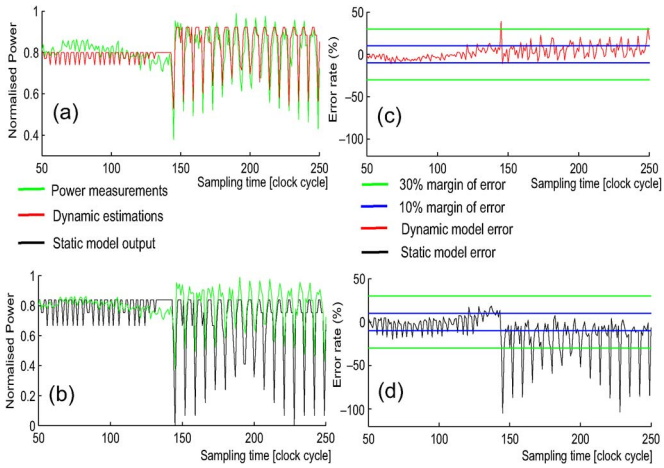



Fig. 6. MAC test case: normalized power given by (a) dynamic and (b) static models versus PrimePower estimation. Error profiles of each model are reported in subplots (c) and (d).

Long simulations (millions of clock cycle) show an average energy accuracy with 5.3% error and maximum energy error of about 4.8%. In the same way, the enable signal of the MAC component is permanently monitored. A 32-b counter records events on that signal during each power mode. To track power dynamics, we used one finite state machine implemented as a 2-b register. When moving from one state to another, the counter content is stocked in a dedicated register before being reinitialized. For fair comparison of precision, we used one counter attached to the enable signal (like in the dynamic method) to implement the static power model of the MAC. This counter is scrutinized periodically, hence, the name "static." Simulation results are shown in Fig. 6. The MAC model tracks power drops and peaks better than static implementation. Maximum energy tracking error was estimated to 7.8%, and average energy error was about 6%. However, in some cases, model error increased to 30%; to reduce that margin, more events can be used for modeling. Fig. 6 shows that, with equivalent resources, the static model is absolutely unreliable.

### C. Result Analysis

The proposed solution is quite new in the literature; the most similar approach is described in [5]: five signals are monitored with 32-b counters to track average power over 50 ms. Obtained accuracy is about 10%. In our solution, we showed that, with only one monitored signal and one 32-b counter, it is possible to manage an accuracy of 10% at finer granularity. Of course, these results should be better compared, but clearly, our solution allows to limit the overhead induced by activity monitors.

## VI. Conclusion

In this brief, we have described a new method to dynamically track SoC consumption using conventional event counters. Markov modeling was used to set power modes and related transition rules. Power variations are smoothed when using static monitors because of long interval readings, whereas our dynamic approach is designed to detect critical variations. The dynamic model has many proven advantages over a static approach. RAM and MAC components were used as test cases, and the results obtained are promising. Our future work includes modeling larger blocks and evaluating related models in different application contexts.

## References

[1] L. Chen, Y. Chen, and I. Huang, "A real-time power analysis platform for power-aware embedded system development," *J. Inf. Sci. Eng.*, vol. 27, no. 3, pp. 1165–1182, May 2011.

[2] J. Flinn and M. Satyanarayanan, "PowerScope: A tool for profiling the energy usage of mobile applications," in *Proc. 2nd IEEE Workshop Mobile Comput. Syst. Appl.*, New Orleans, LA, USA, Feb. 1999, pp. 2–10.

[3] V. Tiwari, S. Malik, and A. Wolfe, "Power analysis of embedded software: A first step towards software power minimization," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, Los Alamitos, CA, USA, 1994, pp. 384–390.

[4] K. Ganesan, L. John, V. Salapura, and J. Sexton, "A performance counter based workload characterization on blue gene/P," in *Proc. 37th Int. Conf. Parallel Process.*, Washington, DC, USA, 2008, pp. 330–337.

[5] F. Bellosa, "The case for event driven energy accounting," Univ. Erlangen, Erlangen, Germany, Tech. Rep. TR-14-01-07, 2001.

[6] S. W. Chung and K. Skadron, "Using on-chip event counters for high-resolution, real-time temperature measurement," in *Proc. 11th Intersoc. Conf. Thermal Thermomech. Phenom. Electron. Syst.*, San Diego, CA, USA, May 2006, pp. 114–120.

[7] *3rd Generation Intel Xscale Microarchitecture Developers's Manual*, Intel, Santa Clara, CA, USA, May 2007152170, Order Number: 316283-002US.

[8] J. M. May, "MPX: Software for multiplexing hardware performance counters in multithreaded programs," in *Proc. 15th Int. Parallel Distrib. Process. Symp.*, San Francisco, CA, USA, Apr. 2001, pp. 22–30.

[9] P. Chen, C. King, Y. Chang, and S. Tseng, "Multiprocessor system-on-chip profiling architecture: Design and implementation," in *Proc. 15th Int. Conf. Parallel Distrib. Syst.*, Shenzhen, China, 2009, pp. 519–526.

[10] L. Bircher and K. John, "Complete system power estimation: A trickle-down approach based on performance events," in *Proc. Int. Symp. Perform. Anal. Syst. Softw.*, San Jose, CA, USA, Apr. 2007, pp. 158–168.

[11] R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," in *Readings in Speech Recognition*. San Francisco, CA, USA: Morgan Kaufmann, 1989, pp. 267–296.

[12] J. D. Hamilton, *Regime Switching Models,* 2nd ed., S. N. Durlauf and L. E. Blume, Eds. Hampshire, U.K.: New Palgrave Dictionary of Economics, 2008. [Online]. Available: http://www.dictionaryofeconomics.com/article?id=pde2008_R000269

[13] SSRN M. Perlin, MS regress—A package for Markov regime switching models in Matlab, Aug. 2007. [Online]. Available: http://ssrn.com/abstract=1714016 SSRN

[14] C. Bernard and F. Clermidy, "A low-power VLIW processor for 3GPP-LTE complex numbers processing," in *Proc. Des. Autom. Test Europe*, Grenoble, France, Mar. 2011, pp. 1–6.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2015.2407832, IEEE Transactions on Emerging Topics in Computing

# Cost-Effective Design Strategies for Securing Embedded Processors

Florent Bruguier, *Member, IEEE,* Pascal Benoit, *Member, IEEE,* Lionel Torres, *Member, IEEE,* Lyonel Barthe, Morgan Bourree, Victor Lomne

**Abstract**

Side-Channel Attacks (SCAs) such as Differential Power or ElectroMagnetic Analysis (DPA/DEMA), pose a serious threat to the security of embedded systems. In the literature, few articles address the problem of securing General Purpose Processors (GPPs) with resourceful countermeasures. However, in many low-cost applications where security is not critical, cryptographic algorithms are typically implemented in software. Since it has been proved that GPPs are vulnerable to SCAs, it is desirable to develop efficient mechanisms to ensure a certain level of security. In this paper, we extend side-channel countermeasures to the Register Transfer Level (RTL) description. The challenge is to create a new class of processor that executes embedded software applications, which are intrinsically protected against SCAs. For that purpose, we first investigate how to integrate into the datapath two countermeasures based on masking and hiding approaches. Through an FPGA-based processor, we then evaluate the overhead and the effectiveness of the proposed solutions against time-domain first-order attacks. We finally show that a suitable combination of countermeasures significantly increases the side-channel resistance in a cost-effective way.

**Index Terms**

Cryptography, Side-Channel Attacks, RISC Processor, Countermeasures, Masking, Hiding, FPGA, Time-Domain



**EDICS Category: 3-BBND**

- *F. Bruguier, P. Benoit, L. Torres, L. Barthe, M. Bourree, and V. Lomne,are with the Microelectronics Department of Montpellier Laboratory of Informatics, Robotics and Microelectronics (LIRMM), UMR 5506, University of Montpellier - CNRS, Montpellier, France. E-mail: secnum@lirmm.fr*

# Cost-Effective Design Strategies for Securing Embedded Processors

## 1 Introduction

### 1.1 Context

To meet the stringent security requirements of electronic systems, various cryptographic tools have been developed. Cryptography, in its traditional approach, studies the security of an algorithm as a mathematical function. For that purpose, it assumes a *black-box* model, in which attackers have only access to inputs and outputs of algorithms in order to obtain the secrets, the cryptographic keys. However, when these algorithms are executed on hardware devices that have the ability to store, process, and output digital data, adversaries have also the opportunity to tamper, perturbate, and spy their physical implementation.

As a consequence, new threats have been identified. In particular, attacks based on run-time information gained from physical characteristics of cryptographic implementations have received an ever increasing interest since their public introduction in 1996 by Paul. C. Kocher [1]. These cryptanalysis techniques, referred as *Side-Channel Attacks (SCAs)*, exploit different forms of information leakage such as the power consumption, electromagnetic emanations, or execution time from integrated circuits. Because such side-channels are dependent on the processed data and operations, simple and complex statistical analyses of collected samples can be conducted to retrieve the encryption keys of cryptographic algorithms.

With the discovery of SCAs, the security at the physical level has thus become a major priority for designers and developers of secure embedded systems. To prevent, or at least to mitigate the potential of these attacks, both industrial and research organisations have investigated a number of solutions, called *countermeasures* [2]. Although most of these methods do not guarantee absolute security, they can significantly reduce the side-channel leakages to make the attacks impractical [3]. However, regardless of the technique used, the implementation of countermeasures does not come for free. Most hardware and software solutions introduce considerable power, area, and performance overheads that are not suitable for resource-limited embedded systems.

### 1.2 Objective

Most of the papers in the literature are only focused on cryptoprocessors, *i.e.* dedicated hardware cryptographic cores for encryption and decryption processing. However, in various applications, cryptographic algorithms are directly implemented as pieces of software in General Purpose Processors (GPPs) for cost and flexibility reasons. Generally, for such applications, the required security is not the highest one, but it is important though to guarantee a certain level of trust. Within this context, the present paper investigates the security issues related to software cryptographic implementations running on embedded processors. More precisely, the attention is focused on the threat of time-domain first-order power and electromagnetic analysis attacks that predict intermediate values such as the *Differential Power Analysis (DPA)* [4] and *Correlation Power Analysis (CPA)* [5].

The objective is to enhance efficiently at low-cost the robustness of the core of embedded processor architectures by introducing design strategies at the Register Transfer Level (RTL), so that the executed software is intrinsically protected against these attacks. The RTL approach is motivated by the need to offer attractive solutions that are independent of the target technology. The real challenge is to provide efficient countermeasures that strike the balance among security, area cost, computing performance, and power consumption in order to meet the requirements of embedded systems. Another fundamental step is to describe the development of a prototype system based on Field-Programmable Gate Array (FPGA) technology, which implements the proposed methods. This approach has indeed the considerable advantage of allowing a security evaluation process under real-environment conditions, which is all the more relevant in the framework of SCAs.

### 1.3 Contributions

This paper presents novel design strategies to mitigate SCAs on embedded processors. Even if they are based on known principles, their application and their implementation on general processor architectures constitutes the originality of this work. The major contributions include:

- a *masking* countermeasure that conceals intermediate values with random values throughout the pipelined architecture of embedded processors (Section 2),
- a *hiding* countermeasure that efficiently randomises the execution of operations at the data-path level by exploiting micro-architectural functionalities of processors (Section 3), and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2015.2407832, IEEE Transactions on Emerging Topics in Computing



- a practical evaluation of the Side-Channel Resistant (SCR) implementation of the *SecretBlaze* [6], an open-source 32-bit soft-core processor developed to investigate the problem of time-domain first-order SCAs on embedded architectures (Section 4).

## 2 Investigation of a Masking Countermeasure

The masking has been extensively studied by both academic and industrial research groups. Its principle is based on *secret sharing*, in which intermediate values are shared by means of random numbers called *masks*, such that each share alone is independent of the secret. For instance, an $i^{th}$-order masking scheme shares each intermediate value in $(i+1)$ shares with $i$ random masks. Several masking schemes have been proposed in the literature, most targeting block ciphers [7], [8], [9], [10], [11]. Such countermeasures have become popular as their soundness can be formally proven [12].

In this context, we propose to study and extend the principle of masking to the datapath of embedded processors. For that purpose, we only consider the $1^{st}$-order masking scheme, which is briefly introduced in the next subsection.

### 2.1 General Description

#### 2.1.1 $1^{st}$-order Masking Principle

To remove the dependence between the side-channel leakages and the internal *sensitive values* of a cipher implementation, the fundamental idea of a $1^{st}$-order masking scheme is to share a sensitive data $d$ into two shares; a mask $m$ and a *masked data* $d_m$ such that:

$$d \rightarrow (d \circ m,\ m) = (d_m,\ m) \tag{1}$$

where $\circ$ is a *group operation*, which can be for instance Boolean or arithmetic.

From (1), the original value can be obtained from the inverse element according to:

$$(d_m,\ m) \rightarrow d = d_m \circ m^{-1} \tag{2}$$

For convenience, we use the terms *to mask* and *to unmask* to refer to equations (1) and (2), respectively.

When $m$ is randomly and uniformly chosen from $d$, each of the shares is independent of $d$. As a consequence, the side-channel leakages of the overall cipher execution are independent of the secret, which thus provides a protection against $1^{st}$-order statistical power and electromagnetic analyses.

The challenge of this approach lies in the difficulty to reconstruct the expected value: this step is the *mask correction*. Indeed, for a proper masking scheme, both the mask and the masked data have to be uniformly distributed throughout the various processes of a cryptoalgorithm implementation. There are two cases to consider:

- when the transformation process has a linear system property, and
- when the transformation process has a non-linear property.

For the first case, a linear transformation $\mathfrak{L}$ can be applied independently to each share according to equation (3):

$$(\mathfrak{L}(d \circ m), \mathfrak{L}(m)) = (\mathfrak{L}(d) \circ m', m') \tag{3}$$

where $m' = \mathfrak{L}(m)$.

Hence, the correct value can be extracted at the end of the process by computing its inverse transformation, as illustrated by equations (4) and (5):

$$\forall d, m,\ \ \mathfrak{L}(d \circ m) = \mathfrak{L}(d) \circ m' \tag{4}$$

$$\Rightarrow \mathfrak{L}(d) = \mathfrak{L}(d \circ m) \circ m'^{-1} \tag{5}$$

For the second case, the masking structure becomes more complex due to the non-linear property of the transformation. To overcome this problem, a common technique is to modify the non-linear part in order to produce the expected value. The secure solution is to pre-compute and store the values resulting from all possible combinations of masks and masked values into special *masked tables* [13]. Although the overhead of this approach is not negligible, it has the advantage to guarantee the correct execution of a cryptographic algorithm without involving the use of sensitive data.

To be an effective method, it is essential to remark that a particular attention should be given to operations with masked data and mask values. For instance, if two masked intermediate values are processed, we need to ensure that the result is still masked, *i.e.* it is crucial to avoid intermediate data sharing the same mask. In addition to this problem, all steps related to the mask correction should be carefully implemented to limit the information leakage. In practice, the usefulness of the masking method may be compromised if these details are not properly considered.

#### 2.1.2 Dual Pipelined Datapath Masking Scheme

As the masking technique is an efficient method to remove the dependence between processed data and the side-channel leakages, we suggest to study the integration of its algorithmic description into the architecture of embedded processors at the RTL. Our solution consists of implementing a *dual pipelined datapath*.

Basically, the idea is to introduce a special datapath for the mask itself, which can be coupled to a classic RISC-based datapath. Hence, instead of directly handling raw data, the processor operates on a dual datapath with masked data. The main role of the new datapath is to keep the corresponding mask for each masked data along the pipeline structure of the processor. It thus allows to implement all steps related

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2015.2407832, IEEE Transactions on Emerging Topics in Computing



to the mask correction to ensure the correct execution of instructions. Besides, one or more Pseudo Random Number Generators (PRNGs) are also included to generate the masks, which should be updated at each step of the datapath for a more efficient masking scheme.

The simplified model of our approach is depicted in Figure 1. Green (long) dash lines illustrate the pipeline with masked data, blue dot lines indicate the pipeline with masks, whereas black (short) dash lines point out the optional hardware. The direction of the data flow is indicated by arrowheads. In addition, filled circle arrows are used to denote the interaction between each datapath to properly implement the mask correction.

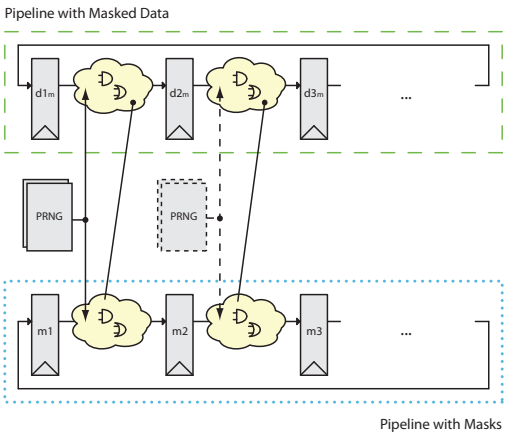



Fig. 1. The simplified model of the dual pipelined datapath masking scheme.

Despite the apparent simplicity of the concept, there are significant challenges to overcome, in particular, about the mask correction and the memory management of the masks.

## 2.2 Architectures

To implement the idea of the dual pipelined datapath, different approaches can be considered, along with their advantages and disadvantages. In the following paragraphs, we present various architectures of an embedded masked processor that attempts to address the side-channel issues. Thanks to a RISC-based design, we examine the masking strategy for two cases:

- the datapath related to register-to-register instructions, and
- the datapath related to memory-reference instructions.

### *2.2.1 Masked Datapath for Register-to-Register Instructions*

The first datapath to consider is the one that takes place in the core of a processor: the computational datapath, which is exclusively used by register-to-register instructions. It starts from the register file, then goes through several pipeline stages, performs mathematical operations, and finally returns back to the register file.

With the proposed masking strategy, the processor implements a new datapath dedicated to masks, operating in parallel to the original datapath. It thus follows the same steps, starting from fetching operands from the register file and ending to store the result into the register file. A *register file of masks* is provided to store each mask associated with the respective masked data. The architecture also includes various *pipeline registers of masks*, carrying the mask value from one stage to another.

Implementing the mask correction is more challenging. For this purpose, we distinguish two categories of register-to-register instructions:

- arithmetic and logic instructions, and
- control flow instructions.

In case of arithmetic and logic instructions, masked data are transferred from stage to stage without any loss of data integrity until the execute phase, where all mathematical operations are implemented. While masked cryptographic co-processors usually involve to pre-compute and store masked tables for specific operations, the large number of instructions and their relative complexity require a trade-off between the cost in terms of gates and the efficiency of the masking method. As a consequence, we propose, in a first step, to perform ALU operations with unmasked data and, in a second step, to mask the result of the ALU with a new Pseudo Random Number (PRN). This crucial design choice is also motivated by the fact that, even if SCAs are still effective on combinational logic, experimental results suggest that the leakage at the register stages is predominant [14], [15] and masking ALU operations results in large overhead [16]. Hence, although this solution is not perfect, it has the advantage to protect most critical parts of the datapath while achieving an attractive trade-off between performance and security.

In case of control flow instructions, the mask correction is straightforward. Since they are not the target of the studied model of attacks, it is therefore allowed to unmask the masked data for all related processes (computations, address assignments, branch evaluations, etc.) without breaking the efficiency of the masking scheme.

Figure 2 illustrates the proposed masked datapath for register-to-register instructions. The direction of the data flow is indicated by arrowheads while the optional hardware used to update the mask is depicted with black dash lines. Unmask and mask modules are also included in the execute phase of the pipeline. Note that, if the execute phase requires more stages, the same approach should be adopted. Finally, filled circle arrows are used to denote other steps of the mask correction that are used for control flow instructions and do not provide useful information for

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2015.2407832, IEEE Transactions on Emerging Topics in Computing



side-channel analysis.

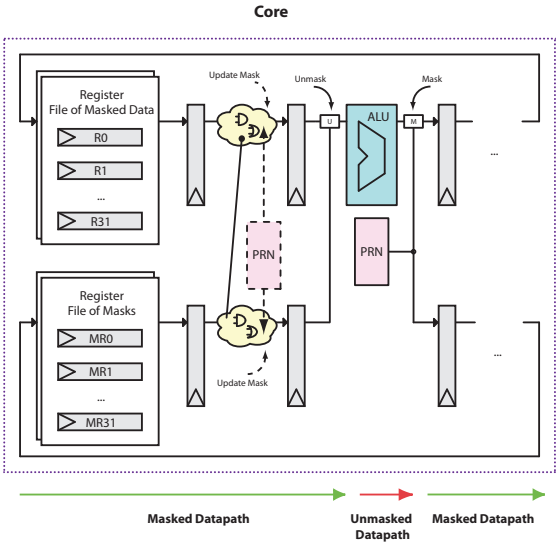



Fig. 2. The architecture of the masked datapath for register-to-register instructions.

#### 2.2.2 Masked Datapath for Memory-Reference Instructions

The second datapath that leaks sensitive information comes from memory-reference instructions. As a load-store architecture, memory instructions transfer data between the register file and the memory sub-system of the processor.

Like the masked datapath for register-to-register instructions, the idea of the countermeasure is to secure the data flow of the architecture by introducing a new pipeline of masks with various pipeline registers throughout all steps of load and store instructions, starting from the register file and ending to the memory sub-system of the processor. To reduce the overhead of the solution, some parts of the masked datapath for such instructions can be shared with the one for register-to-register instructions. The structure of the data alignment process also facilitates the integration of the masking scheme into the core of the processor, reducing thus the complexity of the mask correction for load and store instructions.

In this article, the memory management of masks that depends on the depth and complexity of the memory hierarchy is not addressed in detail. A simple approach that consists in generalising the dual pipelined concept to the whole memory architecture is assumed. Hence, all registers and RAMs used by the data part of the memory sub-system have to be duplicated to store the masks associated to the masked data. This concerns not only the data cache memory, but also all components linked to the cache architecture such as the memory controller, the bus structure, and the main memory of the system. More efficient approaches are discussed in detail in [17].

TABLE 1
Performance and resource overhead of the masking countermeasure.

| | Without Masking | With Masking | Overhead |
|---|---|---|---|
| # Flip-Flops | 897 | 1234 | +37.6% |
| # LUT | 2423 | 3389 | +39.9% |
| #BRAM | 18 | 22 | +22.2% |
| fMAX (MHz) | 64.5 | 52.4 | −18.8% |

TABLE 2
Performance and resource overhead of the SecretBlaze's PRNG.

| | SecretBlaze's PRNG |
|---|---|
| # State Bits | 1024 |
| # Random 0utput per cycle | 32 |
| Maximum SR length | 32 |
| # Flip-Flops | 350 |
| # LUT | 510 |
| fMAX in MHz | 125 |

### 2.3 Implementation and Overhead Evaluation

Choosing the *SecretBlaze* [6] – a 32-bit embedded processor – as a case study, we implemented the concept of the masked datapath based on the Boolean group. At the hardware level, this group has not only a low overhead cost, but also the advantage to reduce the complexity of the integration of masking and unmasking operations into the pipelined architecture of the processor.

The performance impact and the resource overhead of the studied masking method were estimated in Table 1. A cost-effective configuration of the SecretBlaze (instruction and data caches were set to 8 KB and the core included a barrel shifter unit) was chosen to conduct this evaluation. The generation process of the mask values was implemented using a PRNG optimised for FPGAs that produces three 32-bit mask values each clock cycle. This PRNG is based on the use of Shift Registers (SRs). It takes advantage of bit-wise XOR operations and the ability to turn Look-Up Tables LUTs into SRs, providing a good balance between quality and area. The complete description of the PRNG can be found in [18].

These results were obtained with Xilinx's XST 12.1 using the 90 nm Spartan-3 technology (speedgrade -4). Synthesis options were set at the highest speed optimisation level with strong timing constraints. The resource usage was evaluated with the number of flip-flops, 4-inputs (LUTs), and Block RAMs (BRAMs) while the maximum operating frequency (fMAX) was estimated after the place and route process. Note that the results of the PRNG are given separately in Table 2. Clearly, a better (or worse) PRNG would lead to higher (or lower) resource requirements.

Not surprisingly, the overhead on the number of flip-flops is significant, owing to the introduction of several 32-bit pipeline registers for the mask values.

As regard the number of LUTs, the overhead is about 40% and mainly results from the introduction of the register file of masks (synthesised as distributed resources with 484 LUTs). The number of BRAMs also increases by 22% with 8KB dedicated to the cache of masks. Finally, the fMAX is reduced by 19%, which comes from special mask modules implemented within some critical paths.

In summary, the overhead costs introduced by the proposed masking method are relatively small. Furthermore, the SecretBlaze does not implement debug modules, exceptions and exception handling, a power management system, and a Memory Management Unit (MMU), which are not related to the masking countermeasure. That is why the performance impact and the resource overhead of the masking method for a similar industrial processor should be even smaller.

### 2.4 Comparison with Related Work

To the best of our knowledge, there is one example in the literature of a masking countermeasure that takes place into the datapath of embedded processors.

In [19], authors provide a complete masking framework for the LEON3 processor based on the Boolean masking with a secure zone (note that the original idea was introduced early in [20] with the LEON2 processor). However, their approach is only focused on the protection of cryptographic instruction set extensions and does not address the problem for the whole Instruction Set Architecture (ISA). Despite its efficiency, this method has some drawbacks including the development of specific instruction set extensions for an application, the customisation of the compiler, and the overhead due to the use of DPA-resistant logic styles. Unlike our method, their solution is not generic and circumvents the side-channel issues of embedded processors by treating only the inner blocks related to cryptographic extensions.

## 3 Investigation of a Hiding Countermeasure

The goal of hiding countermeasures is to make the physical characteristics of integrated circuits independent of intermediate values and operations performed during cryptographic applications. Among hiding countermeasures, we essentially distinguish two strategies: one based on the randomisation of the execution of cryptographic algorithms [21], [22] and one based on balanced DPA-logic styles [23], [24]. Note that, unlike masking-based countermeasures that are only efficient against statistical power and electromagnetic-based attacks, methods based on the randomisation of the execution increase the robustness of cryptosystems against most SCAs, including simple power analysis and timing-based attacks.

To be exhaustive in our study, we have chosen to study and integrate a hiding countermeasure that operates into the datapath of embedded processors.

### 3.1 General Description

Because of the large overhead of secure logic styles, our research efforts were focused on the integration of a cost-effective module that randomises the instruction stream at the hardware level.

Our idea is essentially based on the concept of a *non-deterministic processor* [25], in which the software can be executed with random additional operations that are generated by the hardware architecture of the processor. It allows not only to randomly insert dummy cycles that change the execution time, but also to randomise the usage of available hardware resources in order to increase the "noise", scrambling the patterns in the power consumption as well as the electromagnetic waves to prevent advanced signal processing, statistical, and modelling methods [26]. The countermeasure hence affects both time and amplitude dimensions of the physical leakages.

One solution consists of implementing a *pipeline randomiser* that handles a RISC datapath in a non-deterministic fashion through dummy control and data signals. Within this structure, a PRNG is required to provide a random information that should be used by the pipeline randomiser for taking a decision for each instruction being processed at each stage of the pipeline: either to keep a normal execution or to perform a dummy cycle. Additional PRNGs can also be implemented to generate random numbers for data operands during dummy cycles. This increases the random switching activity in order to lower the information leakage of the circuit.

The simplified model of our approach is depicted in Figure 3, in which the direction of the data flow is indicated by arrowheads and the additional hardware is illustrated with black (short) dash lines.

The challenge of this method lies in the difficulty to randomise both the execution of instructions and the content of registers in an efficient manner without altering the behaviour of the application.

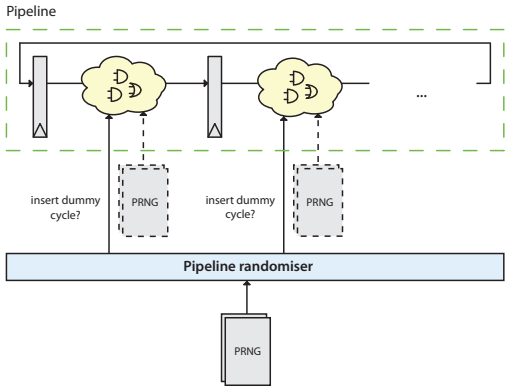



Fig. 3. The simplified model of the pipeline randomiser.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2015.2407832, IEEE Transactions on Emerging Topics in Computing



### 3.2 Architecture

In this section, we introduce an efficient and cost-effective architecture to implement the pipeline randomiser method within the datapath of embedded processors. For that, we examine two key elements:

- the mechanism that implements a non-deterministic execution, and
- the mechanism that increases the noise.

#### 3.2.1 Non-Deterministic Execution

When considering most RISC architectures of embedded processors such as ARM, MIPS, and PowerPC, a pipeline interlock is usually implemented to detect and solve data dependencies or control flow conditions specified in a sequence of instructions, known as *hazards*. This mechanism is often called a *hazard controller*, and thus manages the state of each pipeline stage throughout the whole architecture. A common technique to solve hazards is to *stall* all registers related to a pipeline stage until the data dependency is cleared (also known as pipeline bubbling). Sometimes it is also necessary to *flush* the pipeline stage to remove the execution of an instruction. Note that in case of an out-of-order execution (typically superscalar architectures for high-end embedded systems), advanced methods such as *scoreboarding* [27] or *Tomasulo algorithm* [28] are implemented to reduce the number of pipeline stalls. Nevertheless, a similar mechanism with stall, flush, and enable control signals remains necessary to ensure the correct execution of instructions.

To implement a non-deterministic execution, we can take advantage of the hardware features of the hazard controller (stall and flush control signals) to insert dummy cycles along the pipelined datapath. Indeed, although stall and flush are used primarily for hazards management, they can be also reused to insert dummy cycles using the same control logic, which does not allow to identify dummy cycles from control and data hazards. This design choice greatly reduces the overhead costs of the countermeasure (area, power, and may not affect the fMAX of the design) while achieving its purpose. Hence, by adding one or more false conditions to the controller, the pipeline operates in an unexpected way that can be efficiently exploited to insert additional calculations. We called it the *ghost hazard generation*.

To be an efficient technique, ghost hazards should be randomly generated. Nevertheless, an excessive use of random cycles can seriously decrease the throughput of the architecture. In practice, a suitable compromise needs to be found. To meet user application requirements, the probability of ghost hazards should be configurable by the software. One solution is to implement a module that compares a threshold value defined in a control register with a PRN generated at each clock cycle. The sign of the result hence determines the generation of ghost hazards. Note also, for real-time systems, it can be important to add a counter that determines the maximum number of consecutive dummy cycles. This can help to estimate the worst case execution time.

The architecture of the ghost hazard controller is illustrated in Figure 4. The direction of the data flow is indicated by arrowheads.

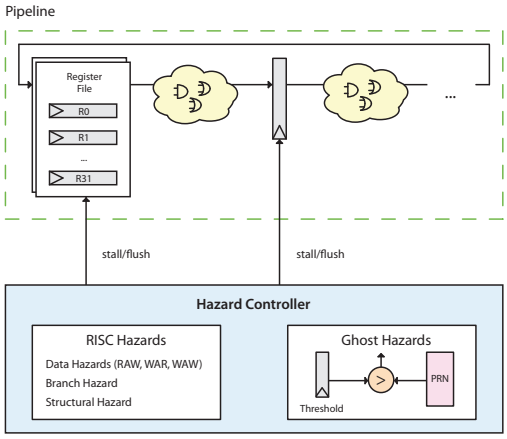



Fig. 4. The architecture of the ghost hazard controller.

#### 3.2.2 Noise Generation

Non-deterministic processor can also introduce random additional calculations to lower the side-channel leakages. Like the masking countermeasure, we consider to randomise the architecture for two cases:

- the datapath related to register-to-register instructions, and
- the datapath related to memory-reference instructions.

The final architecture of the noise generation mechanism is depicted in Figure 5.

3.2.2.1 Randomised Datapath for Register-to-Register Instructions: For register-to-register instructions, a simple approach is to enable the propagation of random data operands from the register file to several functional units. It can be easily done by inserting additional muxes at the beginning of one or more pipeline stages. Hence, through the use of PRNs, pipeline buffers as well as functional units can be randomised to thwart an attacker. To ensure the integrity of the instruction flow, the designer has to ensure that the architecture does not write-back the result of a dummy calculation to the register file, for instance by forcing the write-back control signal to a no-operation state (by definition, this is the purpose of a flush signal).

For the sake of clarity, we would like to precise that most of functional units of the ALU (AND, OR, XOR, ROTATE, ADD, CMP, etc.) operate in parallel. This implies that, according to the type of the instruction being executed, only the result of the decoded

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2015.2407832, IEEE Transactions on Emerging Topics in Computing



functional unit is stored into the pipeline register. In other words, all computations are done at the same time with more or less the same operands. It is hence not necessary to randomly modify all control signals during a dummy cycle (for instance, the one used to select the result from the functional unit). It also appears difficult to distinguish most logical and arithmetic instructions by simple analysis. Typical exceptions are multiply and divide instructions, which are usually handled as multi-cycle instructions with enable control signals to reduce the power consumption of these modules. Depending on the complexity of the processor architecture, it can also be more powerful to randomly activate such modules.

3.2.2.2 Randomised Datapath for Memory-Reference Instructions: Instructions that perform operations with the memory sub-system can seriously modify the leakages of a circuit by activating large memory banks. It is thus essential to add dummy memory operations for increasing the noise. A simple approach is to insert dummy load operations that does not write-back the result into the register file. For that, the address is randomly computed using a PRN and then is assigned to the data memory sub-system. Hence, depending on the available hardware, it may active the data cache memory sub-system, internal memories, or request a data from peripheral devices. In case of incomplete/partial memory map, the designer has to ensure that the random address is properly handled by the decoder of the processor and does not lead to a deadlock state.

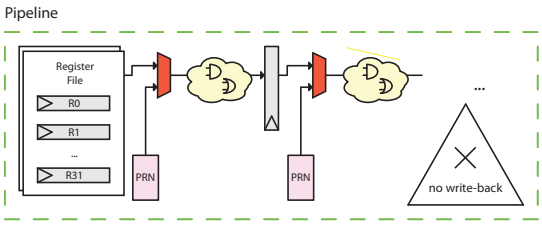



Fig. 5. The architecture of the noise generation mechanism.

### 3.3 Implementation and Overhead Evaluation

Both ghost hazard and noise generation mechanisms were implemented into the 5-stage pipelined architecture of the SecretBlaze. Unlike the masking countermeasure that requires attention to detail with mask values, the randomisation of the instruction stream can be done using a cost-effective design with a PRNG producing a 32-bit value each clock cycle without opening security holes. We arbitrarily chose to implement the noise generation mechanism during the execute phase of the SecretBlaze's pipeline, which significantly affects the side-channel leakages because of multiple functional units. Furthermore, the ghost hazard mechanism was implemented using the same source of PRNs to reduce the cost of the countermeasure.

The performance impact and the resource overhead of the hiding method were estimated in Tables 3 and 4. The evaluation was conducted using the same processor configuration and synthesis options to those described in Subsection 2.3.

TABLE 3
Performance and resource overhead of the hiding countermeasure.

| | Without Hiding | With Hiding | Overhead |
|---|---|---|---|
| **# Flip-Flops** | 897 | 910 | +1.4% |
| **# LUT** | 2423 | 2653 | +9.5% |
| **# BRAM** | 18 | 18 | 0.0% |
| **fMAX in MHz** | 64.5 | 62.7 | −2.9% |

TABLE 4
Performance and resource overhead of the SecretBlaze's PRNG.

| | SecretBlaze's PRNG |
|---|---|
| **# State Bits** | 1024 |
| **# Random 0utput per cycle** | 32 |
| **Maximum SR length** | 32 |
| **# Flip-Flops** | 116 |
| **# LUTs** | 181 |
| **fMAX in MHz** | 125 |

According to these figures, it can be concluded that the proposed method has a negligible hardware overhead compared to conventional countermeasures. This observation mainly results from the ghost hazard mechanism that requires minor modifications to the original design. Synthesis reports also suggest that the increase in the number of LUTs is principally due to the introduction of larger muxes involved for the noise generation. At last, it should be noted that these tables do not include the execution time overhead, which is configurable by the software designer.

### 3.4 Comparison with Related Work

Non deterministic processors were introduced in [25] to thwart side-channel analysis. In this study, authors provide a framework to shuffle the instruction stream in a more or less random fashion. It takes advantage of an out-of-order execution mechanism, typically found in superscalar architectures. The proposed implementation was improved with a random register renaming technique described in [29]. The efficiency of the approach relies on the parallelism of the executed code that is, in practice, fairly limited. To address this issue, an additional pipeline stage was developed in [30], in which some random operations are generated without modifying the effective data. However, all these works can introduce a significant overhead and

can require both hardware and compiler tool-chain modifications.

An alternative method was presented in [31], where additional instructions are injected at run-time by the hardware. It requires small hardware and ISA modifications. Nevertheless, to reduce the complexity of the method, only a subset of instructions is allowed to be randomly executed, which clearly limits the non-determinism method as well as the noise generation.

Compared to these previous works, our contribution brings the following benefits. First of all, the proposed method aims to strike the balance between security and performance by exploiting available resources of most embedded processors such as the pipeline interlock mechanism. It also does not require the customisation of the software tool-chain with additional opcodes to generate dummy instructions. At last, it has the advantage to randomise the data-path at a low abstraction level (control and data signals rather than instruction codes), which gives the designer a greater control over the physical leakages.

## 4 Security Evaluation of the SecretBlaze-SCR Processor

In this section, we propose to evaluate the robustness of the masking and hiding countermeasures through prototyping. The motivation of this approach is to quantify the relative performance offered by the proposed masking and hiding strategies.

The lab instruments used in our experimental setup are a high-performance oscilloscope (3.5 GHz bandwidth, 40 GS/s sampling rate), a low-noise 63 dB amplifier, a near-field electromagnetic probe (500 $\mu$m diameter), a motorised X-Y-Z table allowing accurate positioning, and finally a Spartan-3 FPGA board. The whole measurement process and the data acquisition were controlled by a computer over RS232 and Ethernet communication protocols. All attacks were carried out on a desktop computer powered by a quad-core Intel Core 2 at 2.83 GHz with 8 GB of RAM. They were implemented through a MatLab program, in which the kernel was accelerated using C language through the use of MEX-functions. We detail in the following subsections the framework used for that purpose.

### 4.1 Processor Configurations

The SCR implementation of the SecretBlaze was synthesised at 50 MHz on a low-cost Spartan-3 FPGA. Several functional configurations of the SecretBlaze-SCR processor were defined to evaluate the benefit of the countermeasures separately, but also when combined. Note that it was crucial to ensure that the same FPGA bitstream (with fixed design and implementation characteristics) was used to allow a valid relative comparison between the different modes. Thanks to the flexibility provided by the countermeasures (control through enable bits), the SecretBlaze-SCR was evaluated for the following configurations:

- the *unprotected configuration*, used as the reference design,
- the *protected configuration with masking*,
- the *protected configuration with hiding*, and
- the *protected configuration with masking and hiding*.

Since our main objective was to achieve a secure implementation without jeopardising performance, the hiding countermeasure was arbitrarily configured to randomise the instruction flow with a maximum penalty of 35% (*i.e.* a weak randomisation in the context of side-channel analysis). More precisely, the probability to insert dummy operations was updated every 100 encryptions in software, varying the execution time overhead between 20% and 35%. The goal was to increase the efficiency of the proposed countermeasure by frequently changing the global distribution of dummy operations. Note also that the software overhead induced by the management of both masking and hiding strategies were negligible (less than 20 instructions to initialise and control the processor during the run-time execution).

### 4.2 Attack Scenario

The evaluation of the SecretBlaze-SCR was performed with the *Data Encryption Standard (DES)*, which has been the object of several studies regarding SCAs. The DES was implemented in C programming language, using a straightforward code that was not specifically optimised to take advantage of the 32-bit RISC architecture of the processor.

Using the Hamming Weight (HW) leakage model, the CEMA which is a time-domain attack was used as the reference model of attacks. This choice was mostly motivated by our previous experience with side-channel analysis on embedded processors, which has shown that the electromagnetic side-channel using the Pearson's correlation distinguisher provides the best results from an attacker's point of view.

### 4.3 Security Metrics

#### 4.3.1 Measurement To Disclosure

*Measurement To Disclosure (MTD)* is the first metric that was introduced with the advent of SCAs. It is defined as the minimal number of power or electromagnetic traces required to correctly find the secret key. However, we recall that CEMA attacks rank among divide and conquer cryptanalysis methods that provide distinguishers for small key chunks, called subkeys, which can be recovered independently. In case of the DES algorithm, the attack divides the problem into 8 subkeys of 6 bits each. Consequently, there is a total of 64 combinations for each subkey, which gives a probability of $1.56\%$ to pick

the correct one (assuming an equiprobable distribution). That is why, in practice, this metric is limited when applied for the 8 subkeys of the DES, since the probability to randomly find the correct subkey is clearly high. Nevertheless, MTD still can give an information about the efficiency of the attack, and thus was selected for the security evaluation process.

#### 4.3.2 Measurement To Disclosure with Stability

To overcome the previous limitation, an extended version of the MTD metric was adopted: this is the *Measurement To Disclosure with Stability (MTDwS)*. MTDwS defines the amount of traces needed to guess the correct key with a stability criterion, which states that the distinguisher of an attack continuously output the correct key hypothesis. For that purpose, a threshold value is arbitrarily chosen. Due to the presence of countermeasures, we chose a value of 1000 traces to ensure high stability of the correct key hypothesis. The latter metric is particularly useful to estimate the statistical convergence of attacks.

#### 4.3.3 Percentage of Correct Guesses

One restriction of the MTDwS metric is that the use of the stability as a criterion is not directly related to the security aspects. In the context of SCAs, the key recovery is mainly considered to evaluate the robustness of an integrated circuit. A classical approach is to measure the frequency of the correct key candidate according to the result of the distinguisher: this is the goal of the *Percentage of Correct Guesses (PCG)* metric. For more information about the effectiveness of the attack, we can also compute the percentage of guesses for other key hypotheses in order to determine the *rank* of the PCG.

#### 4.3.4 Guessing Entropy

Even if the previously proposed metrics give us a good starting point to evaluate the countermeasures, these metrics are data-dependent. This might involve different results with a same set of measurements used in a different order. That is the reason why, we employ the *Guessing Entropy (GE)* metric to evaluate the different configurations of our processor [32]. This metric helps us to properly quantify weaknesses and to conclude whether or not the cryptosystem is successfully broken.

### 4.4 Experimental Results

Experimental results for each configuration of the processor are given in Table 5 while the CEMA traces obtained for the first subkey are depicted in Figure 6. We acquired 50,000 traces for the unprotected configuration, 100,000 traces for the protected configuration with masking, 100,000 traces for the protected configuration with hiding, and 200,000 traces for the protected configuration with masking and hiding. These acquisitions were done 3 times with 3 independent data sets. Note that, these data sets were divided into smaller data sets to calculate GE for the unprotected configuration and for the protected configuration with masking. During these experiments the bandwidth of the oscilloscope was adjusted at 2 GHz while the sampling rate was set at 40 GS/s. It took more than a week with our measurement setup to acquire all traces. The compressed traces required about 50 GB of disk space. It should be noted that we favored the quality of the measurements over the quantity.

Note that, the whole design was constrained using PlanAhead tool. We had therefore comprehensive knowledge of the spatial location of logic and sequential elements that implement the processor within the FPGA architecture. The probe was firstly located above the position of the pipeline registers. This one was finally adjusted after doing some measurements and computations before launching the whole measurement campaigns.

### 4.5 Analysis and Interpretation

#### 4.5.1 Unprotected Configuration

The CEMA attack performed with 50,000 electromagnetic traces was successful. All subkeys were quickly broken, as evidenced by the three metrics (MTDwS, PCG, and GE): only 1,449 traces were necessary to fulfill the stability criterion for all subkeys while the PCG values reached more than 99%. Furthermore, the CEMA traces obtained for the subkey S1 (Figure 6(a)) highlight the security issues of the unprotected implementation by giving a time dimension to the leakage. The "accordion effect" clearly visible from the picture (black curve) demonstrates the vulnerabilities of the SecretBlaze-SCR's pipelined architecture.

#### 4.5.2 Protected Configuration with Masking

From the figures given in the table, we first conclude that the SecretBlaze-SCR with the masking countermeasure offers a better resistance against CEMA attacks. The data associated to the stability criterion support this analysis. The MTDwS metric was fulfilled for all subkeys with 14,836 electromagnetic traces. Compared to the unprotected implementation of the processor, the MTDwS was thus increased by a maximum factor of 10.2 for this attack scenario. However, as evidenced by the results of average GE, all subkeys were recovered, which means that the CEMA attack was successful with less than 500 electromagnetic traces (Figure 7).

Despite a moderate improvement, the proposed masking countermeasure does not seem to provide a sufficient level of security for most applications. To figure out the reasons behind this observation, we propose to investigate the origin of the leakage by analysing the CEMA traces. From them (Figure 6(b)),

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2015.2407832, IEEE Transactions on Emerging Topics in Computing



TABLE 5
CEMA results.

| Subkey # | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 |
|---|---|---|---|---|---|---|---|---|
| **MTD** | 141 | 101 | 101 | 144 | 101 | 165 | 108 | 219 |
| **MTDwS** | 1141 | 1104 | 1168 | 1243 | 1101 | 1389 | 1164 | 1449 |
| **PCG** | 99.75% | 99.80% | 99.72% | 99.57% | 99.80% | 99.39% | 99.74% | 99.21% |
| **Rank** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **Broken** | **success** | **success** | **success** | **success** | **success** | **success** | **success** | **success** |

(a) Unprotected configuration.

| Subkey # | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 |
|---|---|---|---|---|---|---|---|---|
| **MTD** | 512 | 2392 | 801 | 4932 | 5325 | 101 | 101 | 10368 |
| **MTDwS** | 3057 | 4751 | 3045 | 14836 | 13784 | 5612 | 4293 | 11511 |
| **PCG** | 98.21% | 96.93% | 98.40% | 88.98% | 88.56% | 95.67% | 96.69% | 81.49% |
| **Rank** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **Broken** | **success** | **success** | **success** | **success** | **success** | **success** | **success** | **success** |

(b) Protected configuration with masking.

| Subkey # | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 |
|---|---|---|---|---|---|---|---|---|
| **MTD** | 147 | 861 | 8111 | 8612 | 373 | failure | 32483 | 2394 |
| **MTDwS** | 24747 | 28733 | 14017 | 11376 | 59384 | failure | 83165 | failure |
| **PCG** | 74.33% | 58.91% | 74.76% | 20.87% | 39.63% | 0.00% | 15.66% | 0.17% |
| **Rank** | 1 | 1 | 1 | 1 | 1 | 63-64 | 1 | 62 |
| **Broken** | **success** | **success** | **success** | **success** | **success** | failure | **success** | failure |

(c) Protected configuration with hiding.

| Subkey # | S1 | S2 | S3 | S4 | S5 | S6 | S7 | S8 |
|---|---|---|---|---|---|---|---|---|
| **MTD** | failure | 23738 | 27465 | failure | 35008 | 111708 | 377 | 35856 |
| **MTDwS** | failure | failure | failure | failure | 156132 | 103069 | failure | 38305 |
| **PCG** | 0.00% | 0.35% | 0.23% | 0.00% | 7.87% | 0.24% | 0.09% | 2.62% |
| **Rank** | 64 | 58 | 62 | 63-64 | 3 | 59 | 61 | 8 |
| **Broken** | failure | failure | failure | failure | failure | failure | failure | failure |

(d) Protected configuration with masking and hiding.

it is apparent that the CEMA attack was successful on the subkey S1, but the leakage appears to be confined to a small number of operations. We can indeed observe two peak areas related to the correct subkey hypothesis. However, the "accordion effect" due to the pipelined architecture is no longer observable. By cross-referencing the behaviour of the SecretBlaze-SCR's pipeline with the list of the executed instructions, we were able to determine which part of the processor was still leaking sensitive information: the ALU was identified as the main source of the leakage. This observation is in fact not surprising. Indeed, it is the consequence of the design choices made during the integration of the masking scheme into the datapath of the SecretBlaze-SCR. We recall that, to achieve a cost-effective implementation, the processor performs ALU operations with unmasked data, which corroborates the results of the CEMA traces. Furthermore, these results confirm that all pipeline registers as well as the data memory sub-system are no longer leaking sensitive information thanks to the masking protection.

To conclude, the experiment evaluation demonstrates that our masking countermeasure is still vulnerable against first-order CEMA attacks. However, its most striking benefit is the confinement of the leakage to a small part of the processor, which modestly enhances the overall robustness of the processor architecture.

#### 4.5.3 Protected Configuration with Hiding

From Table 6c, we can see that six subkeys were broken, but the ranks obtained for other two subkeys were very low. Besides, the subkey S6 was never found according to the MTD metric. To recover the full key, we should have made more measurements.

We thus conclude that the proposed hiding countermeasure works as expected, since the randomisation does not prevent SCAs, but makes them more difficult to perform. Clearly, the more dummy operations are inserted, the more the side-channel resistance of the processor is increased. This experimental evaluation also reveals the statistical effect of the proposed hiding countermeasure over the leakage. Indeed, by analysing the CEMA traces from Figure 6(c), we still are able to distinguish the "accordion effect" related to the pipelined architecture, reflecting a large number of critical operations handled by the processor. This observation suggests that the instruction stream is not sufficiently randomised to thwart an attacker, since

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2015.2407832, IEEE Transactions on Emerging Topics in Computing



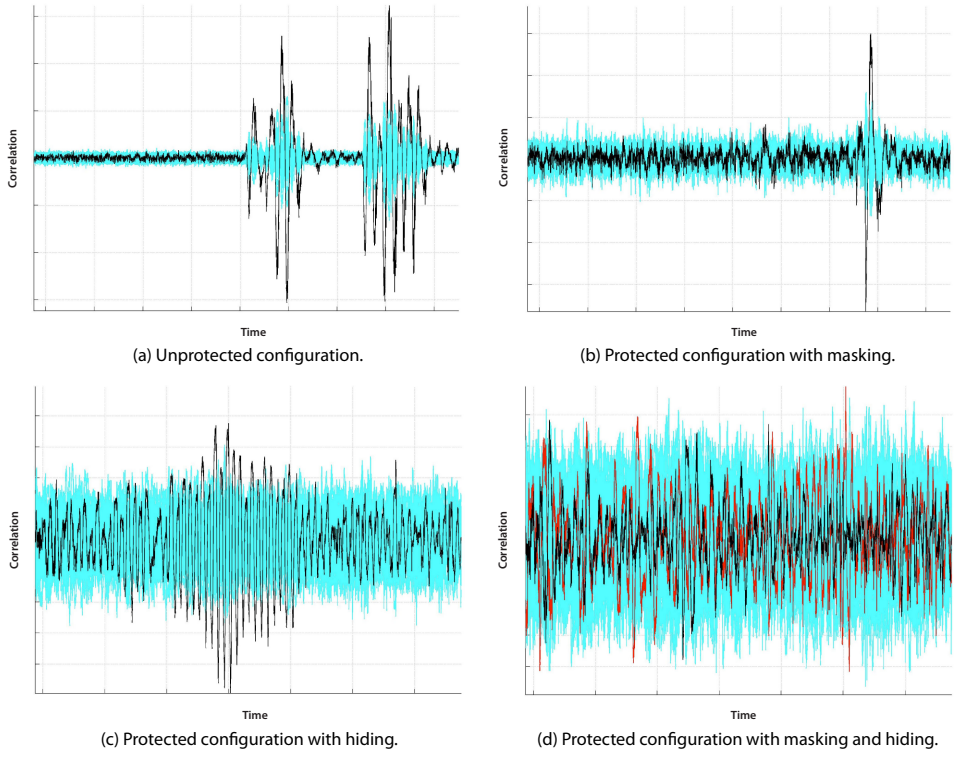



Fig. 6. CEMA traces obtained for the first subkey (*cyan* = wrong subkey hypothesis, *black* = correct subkey hypothesis, *red* = wrong guessed subkey hypothesis).

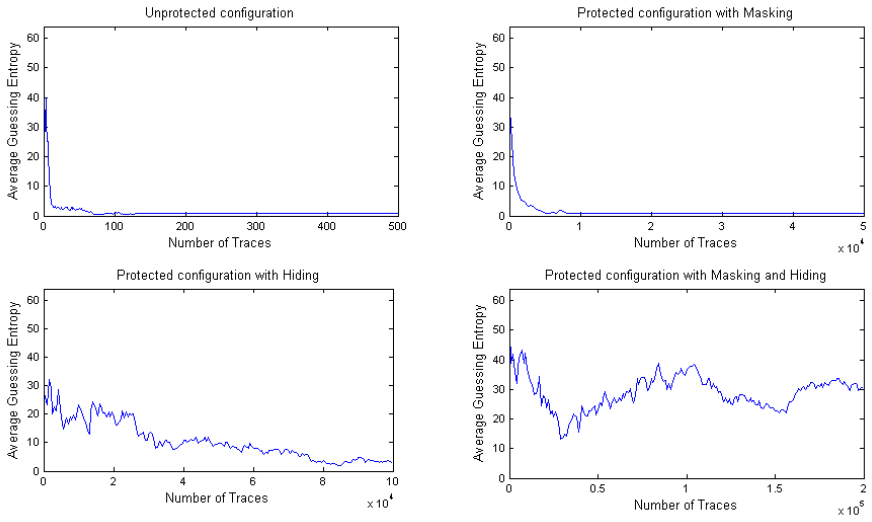



Fig. 7. Number of traces vs average guessing entropy.

the overlapping of critical instructions in the pipeline may intersect with a low randomisation.

#### *4.5.4 Protected Configuration with Masking and Hiding*

Using both masking and hiding countermeasures, the CEMA attack was not successful. This observation

proves that the robustness of the processor was deeply increased.

From an attacker's point of view, the best result was obtained for the subkey S5, which was correctly guessed many times, as indicated by the PCG metric (7.87%). It is difficult to conclude if this subkey was correctly found due to a residual leakage or due to a random situation. The margin, defined as the percentage of the difference between the amplitude of the highest correlation obtained for the guessed subkey hypothesis and the amplitude of the correlation obtained for other subkey hypotheses, was very low (less than 5%) for this subkey, which highlights again the efficiency of the countermeasures. Besides, the CEMA traces obtained for the first subkey support this analysis. From Figure 6(d), it is apparent that the correlation is not relevant, and compared to the previous configurations, the accordion effect is no more perceptible. These results are confirmed by the Figure 7 which shows that the average GE is not converging to one.

Hence, we conclude that the combination of masking and hiding countermeasures can offer a significant protection against CEMA attacks. This is all the more remarkable given that the randomisation of the execution was very low (as a reminder, the timing overhead was 35% at maximum).

## 5 Summary and Conclusions

The research works presented in this paper have been conducted to address the threat of power and electromagnetic SCAs that predict intermediate values. They have been specifically focused on software cryptographic implementations running on embedded GPPs.

Several strategies for securing embedded processors at the RTL have been examined. We have first introduced the concept of the dual pipelined datapath masking scheme, which allows to conceal intermediate values of cryptographic algorithms within embedded processor architectures. Then, we have suggested the concept of the ghost hazard generation, a cost-effective hiding countermeasure that randomises the flow of instructions. Both solutions have been explored to efficiently balance the security needs with the performance and resource overhead. An experimental processor implementing the proposed countermeasures, the SecretBlaze-SCR, has been developed on Xilinx's Spartan-3 FPGA technology. As summarised in Table 6, the performance evaluation of the resulting design has shown that the average overhead induced by the countermeasures is suitable for many embedded systems.

Then, we have performed a practical security evaluation of the SecretBlaze-SCR. From the results obtained with CEMA attacks, we have shown that each countermeasure moderately enhances the robustness against some statistical attacks. While the masking countermeasure has the advantage to confine the leaking operations to the ALU of the processor, the hiding strategy configured with a low randomisation (35% at maximum) also provides a protection by affecting both time and amplitude dimensions of the physical leakages. Furthermore, we have demonstrated the complementary of these methods that can be combined to significantly increase the side-channel resistance of the processor. For an attack scenario of a DES software implementation, we have experimentally shown that the security of the SecretBlaze-SCR was deeply increased.

TABLE 6
Performance and resource overhead of the countermeasures.

| | Masked Datapath | Ghost Hazard Mechanism | Both |
|---|---|---|---|
| **# Flip-Flops** | +37.6% | +1.4% | +39.0% |
| **# LUT** | +39.9% | +9.5% | +48.9% |
| **# BRAM** | +22.2% | 0.0% | +22.2% |
| **fMAX in MHz** | −18.8% | −2.9% | −18.9% |

As a main conclusion, these research works have given some valuable and practical information about the design and implementation processes of a secure embedded processor. We have developed resourceful countermeasures against time-domain first-order SCAs. This contribution may be further investigated in future works, against frequency domain attacks as well as high-order analysis. Although the proposed countermeasures are based on well-known concepts, they have been developed at the RTL description of processor architectures in order to prove their feasibility on silicon while addressing the requirements of many embedded systems. They have the advantage of being independent of the executed cryptographic algorithms, which gives the designer an attractive degree of flexibility when designing a secure system. Due to their construction, they also offer full compatibility with most existing software and technological countermeasures to achieve a higher level of security.

## References

[1] P. C. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," in *Advances in Cryptology*, 1996, pp. 104–113.

[2] P. Kocher, "Design and Validation Strategies for Obtaining Assurance in Countermeasures to Power Analysis and Related Attacks," in *Proceedings of the NIST Physical Security Workshop*, 2005.

[3] C. Clavier, J.-S. Coron, and N. Dabbous, "Differential power analysis in the presence of hardware countermeasures," in *Cryptographic Hardware and Embedded SystemsCHES 2000*. Springer, 2000, pp. 252–263.

[4] P. C. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *Advances in Cryptology*, 1999, pp. 388–397.

[5] E. Brier, C. Clavier, and F. Olivier, "Correlation Power Analysis with a Leakage Model," in *Proceedings of the 2004 Cryptographic Hardware and Embedded Systems Workshop*, 2004, pp. 16–29.

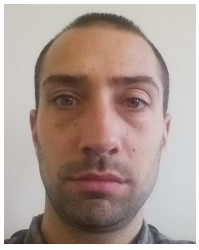
[6] L. Barthe, L. V. Cargnini, P. Benoit, and L. Torres, "A Configurable and Cost-Effective Open-Source Soft-Core Processor," in *Proceedings of the 2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and PhD Forum*, 2011, pp. 305–308.
[7] L. Goubin and J. Patarin, "DES and Differential Power Analysis – The Duplication Method," in *Proceedings of the 1999 Cryptographic Hardware and Embedded Systems Workshop*, 1999, pp. 158–172.
[8] Y. Ishai, A. Sahai, and D. Wagner, "Private Circuits: Securing Hardware against Probing Attacks," in *Advances in Cryptology*, 2003, pp. 463–481.
[9] E. Oswald, S. Mangard, N. Pramstaller, and V. Rijmen, "A Side-Channel Analysis Resistant Description of the AES S-Box," in *Fast Software Encryption*, 2005, pp. 413–423.
[10] K. Schramm and C. Paar, "Higher Order Masking of the AES," in *Topics in Cryptology, The Cryptographers Track at the RSA Conference*, 2006, pp. 208–225.
[11] E. Prouff and M. Rivain, "Provable Secure Higher-Order Masking of AES," in *Proceedings of the 2010 Cryptographic Hardware and Embedded Systems Workshop*, 2010.
[12] S. Chari, C. Jutla, J. Rao, and P. Rohatgi, "Towards Sound Approaches to Counteract Power-Analysis Attacks," in *Advances in Cryptology*, 1999, pp. 398–412.
[13] T. Messerges, "Securing the AES Finalists against Power Analysis Attacks," in *Proceedings of the 7th International Workshop on Fast Software Encryption*, 2000, pp. 150–164.
[14] B. Vaquie, S. Tiran, and P. Maurine, "A Secure D Flip-Flop against Side Channel Attacks," in *Proceedings of the 21st International Conference on Integrated Circuit and System Design: Power And Timing Modeling, Optimization, and Simulation*, 2011, pp. 331–340.
[15] H. Maghrebi, J.-L. Danger, F. Flament, S. Guilley, and L. Sauvage, "Evaluation of Countermeasure Implementations Based on Boolean Masking to Thwart Side-Channel Attacks," in *Proceedings of the 2009 International Conference on Signals, Circuits and Systems*, 2009.
[16] S. Tillich, M. Kirschbaum, and A. Szekely, "Implementation and evaluation of an sca-resistant embedded processor," in *Smart Card Research and Advanced Applications*. Springer, 2011, pp. 151–165.
[17] L. Barthe, "Stratgies pour scuriser les processeurs embarqus contre les attaques par canaux auxiliaires," Ph.D. dissertation, 2012.
[18] D. B. Thomas and W. Luk, "FPGA-Optimised Uniform Random Number Generators Using LUTs and Shift Registers," in *Proceedings of the 2010 Field Programmable Logic and Applications*, 2010, pp. 77–82.
[19] S. Tillich and M. Kirschbaum and A. Szekely, "Sca-resistant embedded processors: The next generation," in *Proceedings of the 26th Annual Computer Security Applications Conference*, 2010, pp. 211–220.
[20] S. Tillich and J. Groschädl, "Power Analysis Resistant AES Implementation with Instruction Set Extensions," in *Proceedings of the 2007 Cryptographic Hardware and Embedded Systems Workshop*, 2007, pp. 303–319.
[21] S. Mangard, E. Oswald, and T. Popp, *Power Analysis Attacks - Revealing the Secrets of Smart Cards*. Springer, 2007.
[22] R. Soares, N. Calazans, V. Lommé, A. Debahoui, P. Maurine, and L. Torres, "A Gals Pipeline Architecture to Increase Robustness against DPA and DEMA Attacks," in *Proceedings of the 23rd Symposium on Integrated Circuits and Systems Design*, 2010.
[23] K. Tiri, D. Hwang, A. Hodjat, B.-C. Lai, S. Yang, P. Schaumont, and I. Verbauwhede, "Prototype IC with WDDL and Differential Routing - DPA Resistance Assessment," in *Proceedings of the 2005 Cryptographic Hardware and Embedded Systems Workshop*, 2005, pp. 354–365.
[24] V. Lomné, P. Maurine, L. Torres, M. Robert, R. Soares, and N. Calazans, "Evaluation on FPGA of Triple Rail Logic Robustness against DPA and DEMA," in *Design, Automation and Test in Europe Conference*, 2009, pp. 634–639.
[25] D. May, H. L. Muller, and N. P. Smart, "Non-Deterministic Processors," in *Australasian Conference on Information Security and Privacy*, 2001.
[26] F. Durvaux, M. Renauld, F.-X. Standaert, L. van Oldeneel tot Oldenzeel, and N. Veyrat-Charvillon, "Cryptanalysis of the CHES 2009/2010 Random Delay Countermeasure," in *IACR Cryptology ePrint Archive*, 2012.
[27] J. E. Thorton, "Parallel Operation in the CDC 6600," in *AFIPS Proc. FJCC*, 1964, pp. 33–40.
[28] R. M. Tomasulo, "An Efficient Algorithm for Exploiting Multiple Arithmetic Units," *IBM Journal of Research and Development*, pp. 25–33, 1967.
[29] D. May, H. L. Muller, and N. P. Smart, "Random Register Renaming to Foil DPA," in *Proceedings of the 2001 Cryptographic Hardware and Embedded Systems Workshop*, 2001, pp. 28–38.
[30] J. Irwin, D. Page, and N. P. Smart, "Instruction Stream Mutation for Non-Deterministic Processors," in *Proceedings of the IEEE International Conference on Application-Specific Systems, Architectures, and Processors*, 2002, pp. 286–295.
[31] J. A. Ambrose, R. Ragel, and G. Parameswaran, "RIJID: Random Code Injection to Mask Power Analysis based Side Channel Attacks," in *Proceedings of the 2007 Design Automation Conference*, 2007, pp. 489–492.
[32] F.-X. Standaert, T. Malkin, and M. Yung, "A unified framework for the analysis of side-channel key recovery attacks," *Advances in Cryptology-Eurocrypt 2009*, pp. 443–461, 2009.

**Florent Bruguier** received a M.S. and PhD degrees in Microelectronics from the University of Montpellier, France, in 2009 and 2012, respectively. Since 2012, he is scientific assistant at LIRMM. He has co-autored over 20 publications. His research interests are focused on self-adaptive and secured approaches for embedded systems.

**Pascal Benoit** received a M.S. and PhD degrees in Microelectronics from the University of Montpellier, France, in 2001 and 2004, respectively. Then he joined the Karlsruhe Institute of Technology at the University of Karlsruhe in Germany where he worked as a scientific assistant. Since 2005, he is a permanent Associate Professor at LIRMM / University of Montpellier. He has co-authored over 130 publications in books, journals and conference proceedings, and holds 5 patents. His present research interests are self-adaptive and secured approaches for embedded systems.

**Lionel Torres** obtained respectively his Master and PhD degree in 1993 and 1996 from the University of Montpellier. From 1996 to 1997, he was in ATMEL company as IP core methodology R&D engineer. From 1997 to 2004, he was assistant professor at Polytech Montpellier engineering school and LIRMM laboratory. Since 2004, he is full Professor and was at the head of the Microelectronic department of the LIRMM from 2007 to 2010. He is now deputy head of Polytech'Montpellier in charge of research, industrial, and international relationship. His research interests and skills concern system level architecture, with a specific focus in the security and cryptographic applications and non-volatile computing based on emerging technologies. He leads several European, national and industrial projects in this field and is (co)author of more than 30 journal papers, 150 conference publications, and 7 patents.

**Lyonel Barthe** obtained his engineering degree in electronics at Polytech'Montpellier, France, in 2009, and his Ph.D in electrical and computer engineering at the University of Montpellier 2, Montpellier, France, in 2012. He is now working at Thales Alenia Space, Toulouse, France, as a FPGA designer for SATCOM modem solutions.



**Victor Lomne** is currently expert in hardware security at the hardware security lab of the ANSSI (french cybersecurity agency). He received the MS degree in cryptology and computer security from the University of Bordeaux, France, in 2007, and the PhD degree in electrical engineering from the University of Montpellier, France, in 2010. His research interests include embedded systems security, cryptographic implementations, physical cryptanalysis and hardware security.

**Morgan Bourree** obtained his engineering degree in electronics at Polytech'Montpellier, France, in 2008. From 2009 to 2011, he worked at IBM Burlington and at LIRMM until 2012. Since 2014, he is automotive consultant at Altran.