LABORATOIRE D'ELECTRONIQUE ANTENNES ET TELECOMMUNICATIONS

Méthodologie de conception pour la virtualisation et le déploiement d'applications parallèles sur plateforme reconfigurable matériellement

Soutenance de thèse du 24 octobre 2012

Clément Foucher

En présence du jury :

Jean-Luc Dekeyser, rapporteur

Guy Gogniat, rapporteur

Alexandre Nketsa, examinateur

Alain Giulieri, directeur de thèse

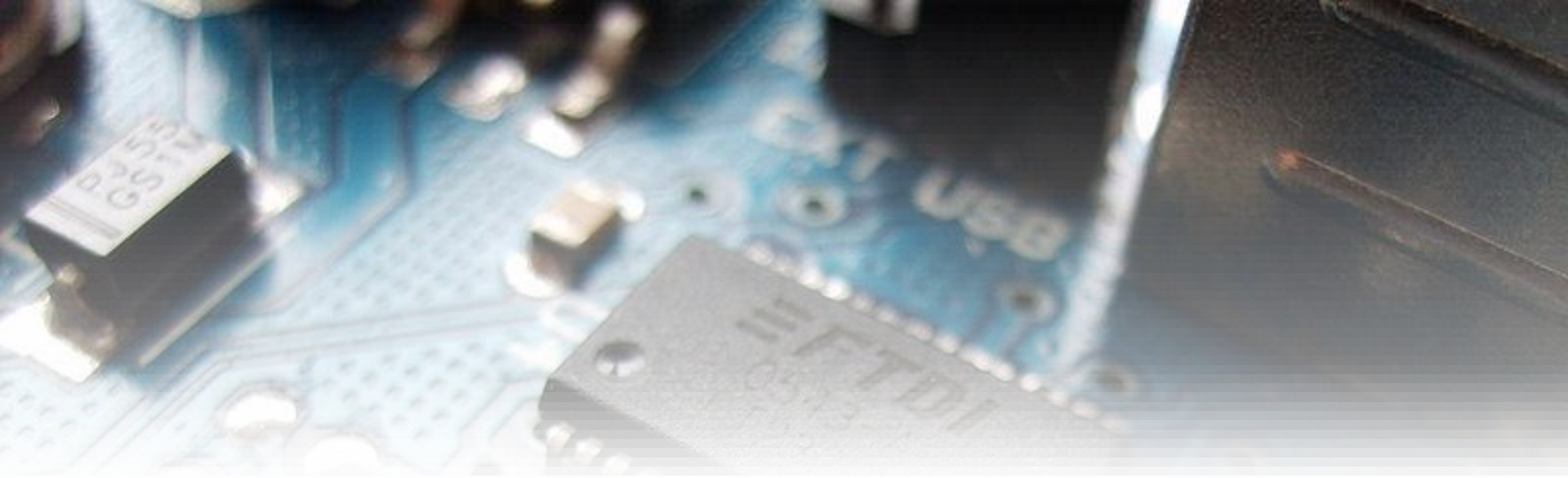
Fabrice Muller, co-encadrant

cnrs dépasser les frontières

Université Nice Sophia Antipolis

Plan

- Introduction
- Modèle d'application
- L'architecture SPoRE
- Plateforme logicielle MPI
- Plateforme matérielle reconfigurable
- Conclusion



INTRODUCTION

- Introduction
  - Contexte
  - Motivations
  - Objectifs
- Modèle d'application
- L'architecture SPoRE
- Plateforme logicielle MPI
- Plateforme matérielle reconfigurable
- Conclusion

Contexte

- Applications et systèmes de traitement des données
  - Utilisation quotidienne
    - Visible ou non
  - En constante évolution
    - Changement de paradigmes pour répondre aux obstacles
- Recherche de puissance
  - Parallélisation
  - Utilisation de ressources hétérogènes
    - Processeurs spécialisés (GPUs, DSPs, …)
    - Accélérateurs matériels
    - Reconfiguration matérielle

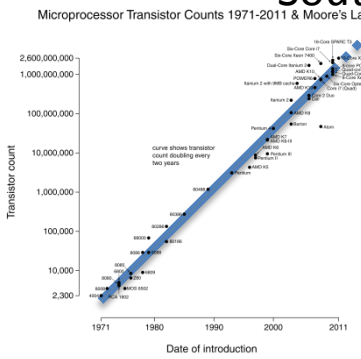
Médecine

Vie publique

Loisirs

Soutenable ?

IBM Blue gene

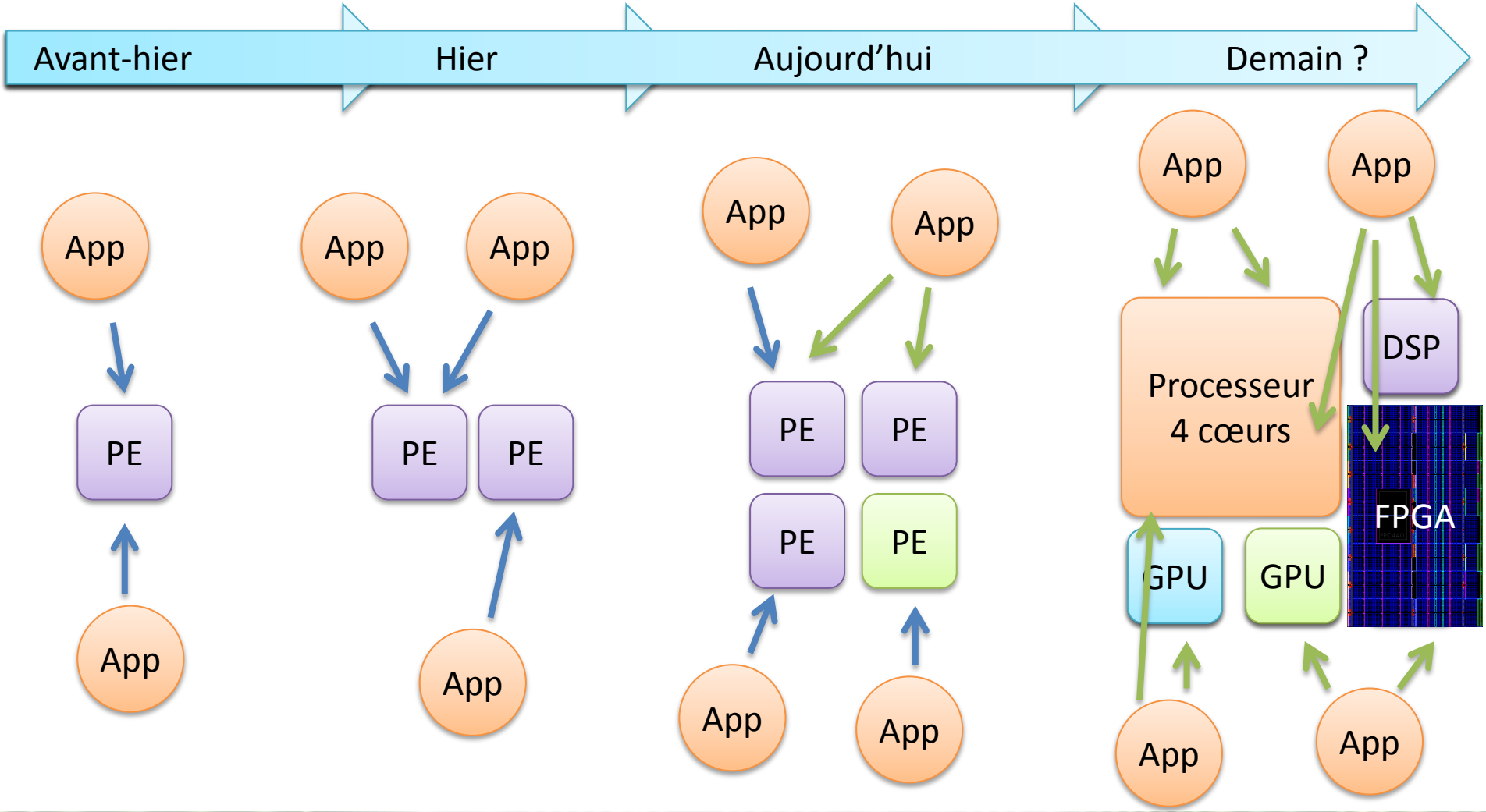


Loi de Moore

GPU

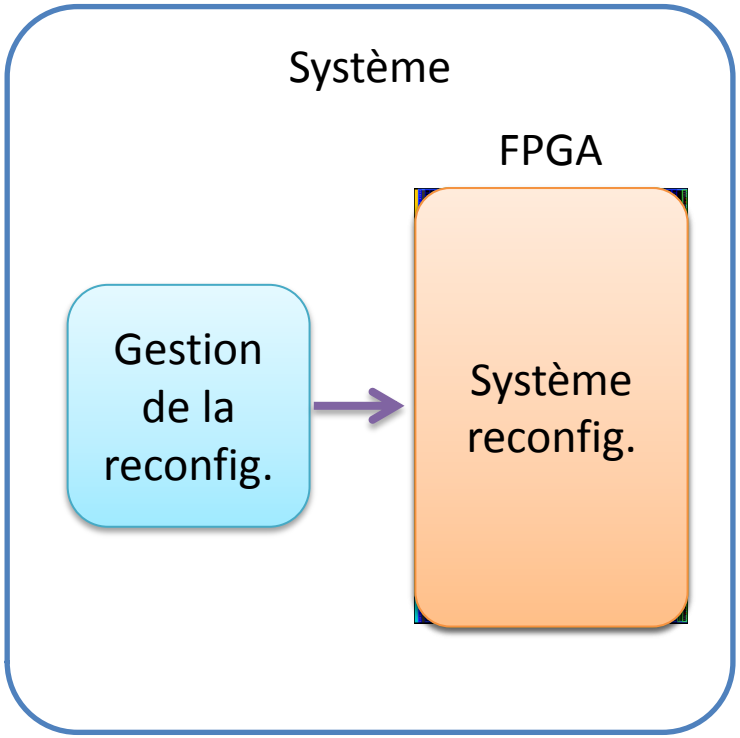
FPGA

Parallélisme et hétérogénéité



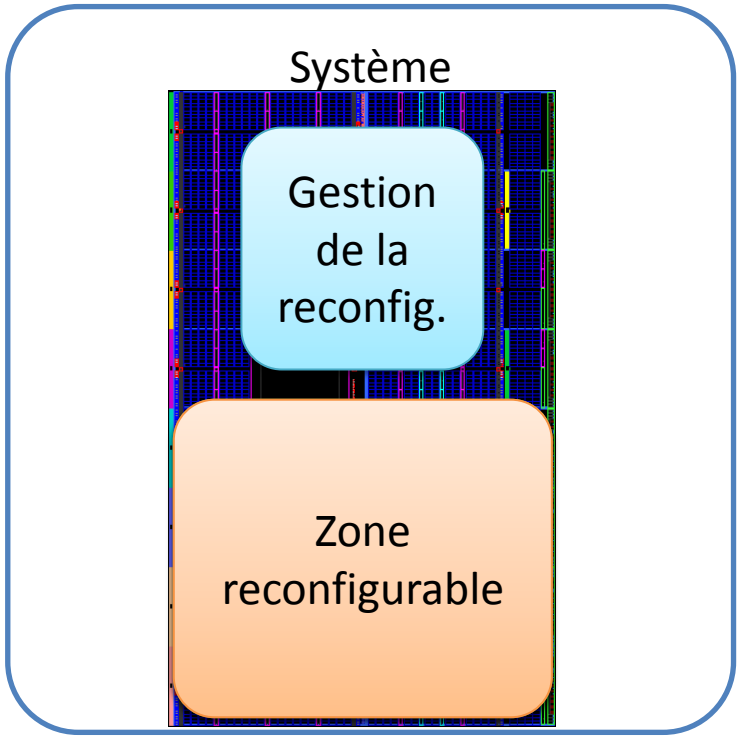
Reconfiguration matérielle

Reconfiguration totale



Nécessité d'un contrôleur externe

Reconfiguration partielle



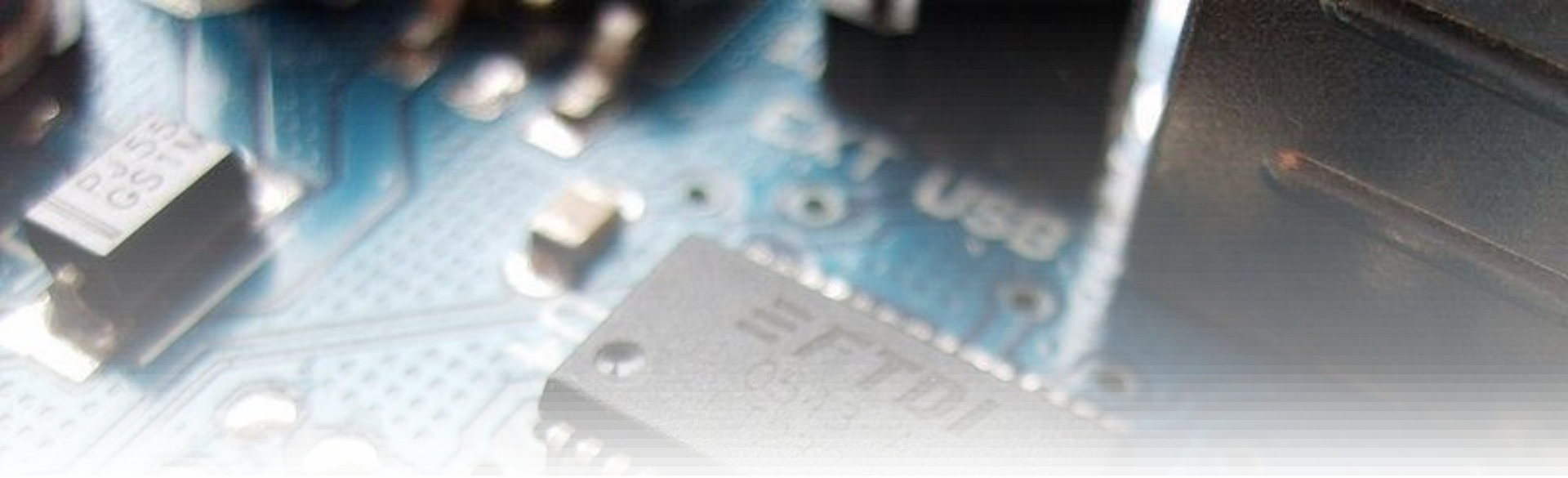
Possibilité de système entièrement sur FPGA (SoPC)

Motivations

- Conception des applications centrée sur le logiciel
  - Matériel vu comme une aide ponctuelle
    - ➔ Dépasser le clivage logiciel / matériel
- Difficulté de mise en œuvre de la reconfiguration partielle
  - Reconfiguration partielle inutilisée dans l'industrie
    - ➔ Gérer de manière transparente le processus

Objectifs

- Méthodologie pour la prise en charge du cycle de vie des applications parallèles
  - Modélisation
  - Développement / implémentation
  - Déploiement et exploitation
- Gestion de l'hétérogénéité des systèmes
  - Logiciels / matériels
  - Statiques / dynamiques (reconfigurables)



MODÈLE D'APPLICATION

Les applications hétérogènes

- Environnements et APIs
  - OpenCL [1]
    - API pour la coordination des calculs
    - Langage OpenCL-C pour l'implémentation des calculs
    - Cible : CPUs, GPUs, DSPs et… FPGAs [2]
      - « High level synthesis » depuis OpenCL-C
  - CUDA [3]
    - C for CUDA permet de programmer les CUDA cores
    - Cible : CPUs et GPUs basés sur les « CUDA cores »
- High-Level Synthesis
  - Xilinx Vivado High-Level Synthesis (ex-AutoESL) [4]
  - Catapult C [5]

[1] Peter Thoman *et al.*, University of Innsbruck, *Automatic OpenCL device characterization : Guiding optimized kernel design*, 2011

[2] http://www.altera.com/literature/wp/wp-01173-opencl.pdf

[3] John Nickolls *et al.*, University of Virginia, *Scalable parallel programming with CUDA*, 2008

[4] http://www.xilinx.com/products/design-tools/vivado/integration/esl-design/hls/index.htm

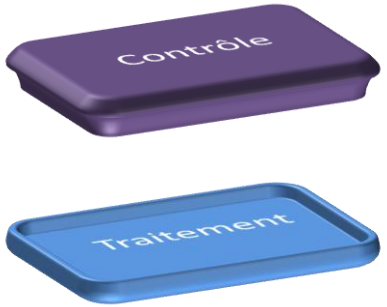
[5] http://www.calypto.com/catapult_c_synthesis.php

Spécifications du modèle

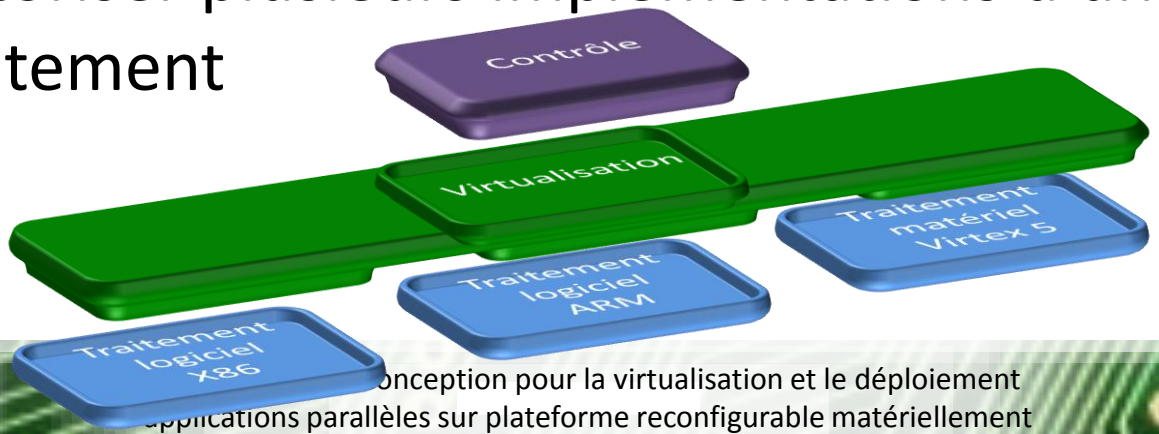
- Profil d'applications visé
  - Applications parallèles
  - Utilisation de ressources hétérogènes
- Portabilité
  - Calculateurs hautes performances
  - Embarqué
- Flexibilité
  - Auto-adaptativité

Hiérarchie

- Applications : deux couches interdépendantes mais distinctes
  - Contrôle : *coordonne* les calculs
  - Traitement : *réalise* les calculs

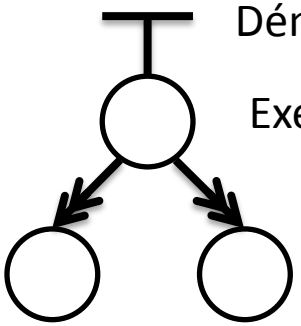


- Séparation des deux couches : virtualisation
  - Autoriser plusieurs implémentations d'un même traitement



Traitement (1/2)

- Notion de noyau
  - Algorithme exécuté sur une unité de calcul
- Application parallèle
  - Plusieurs noyaux exécutés simultanément

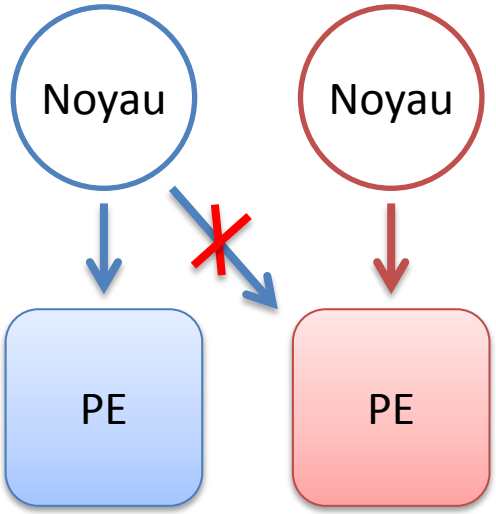


Démarrage de l'application

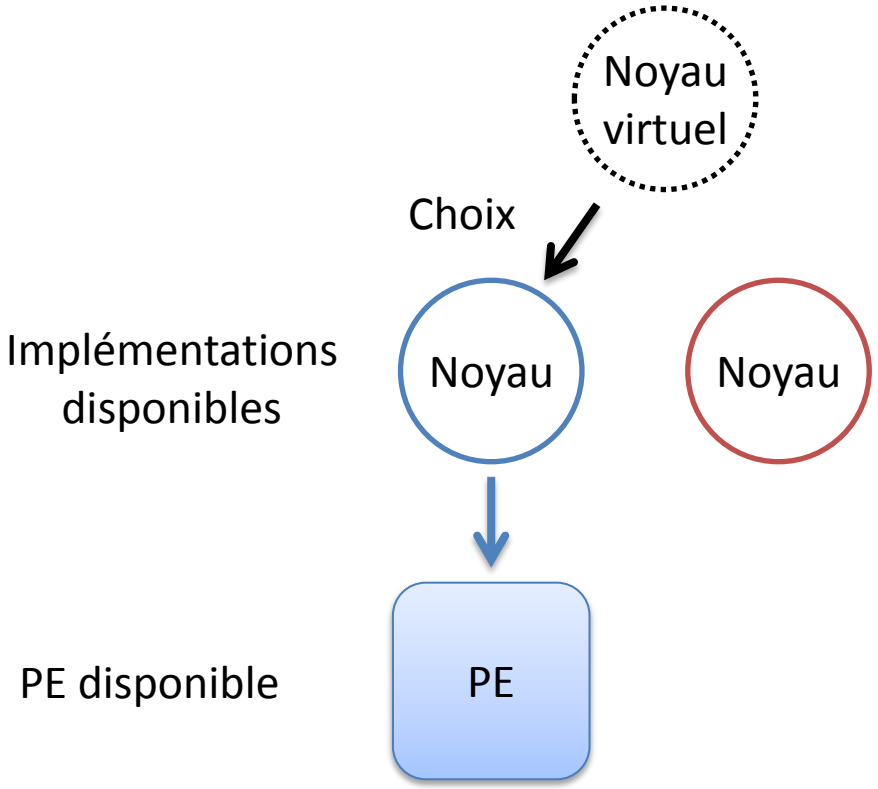
Exécution d'un noyau…

…suivie par l'exécution simultanée de deux noyaux

Traitement (2/2)

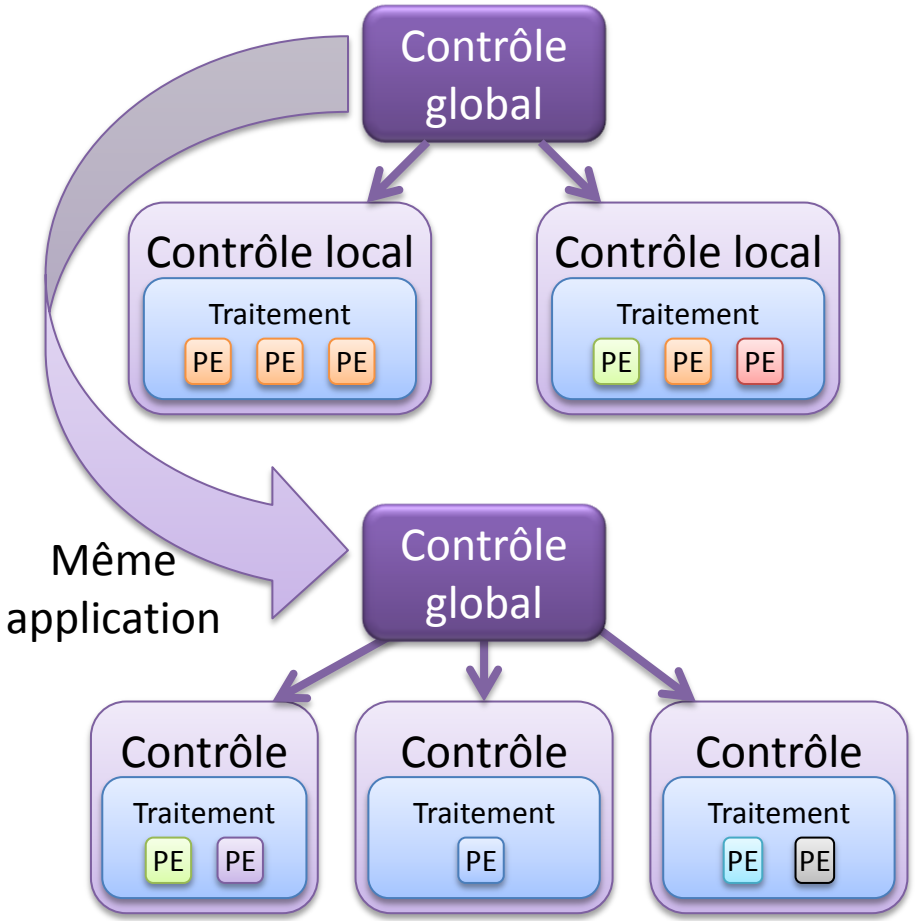


Noyau lié à une unité d'exécution

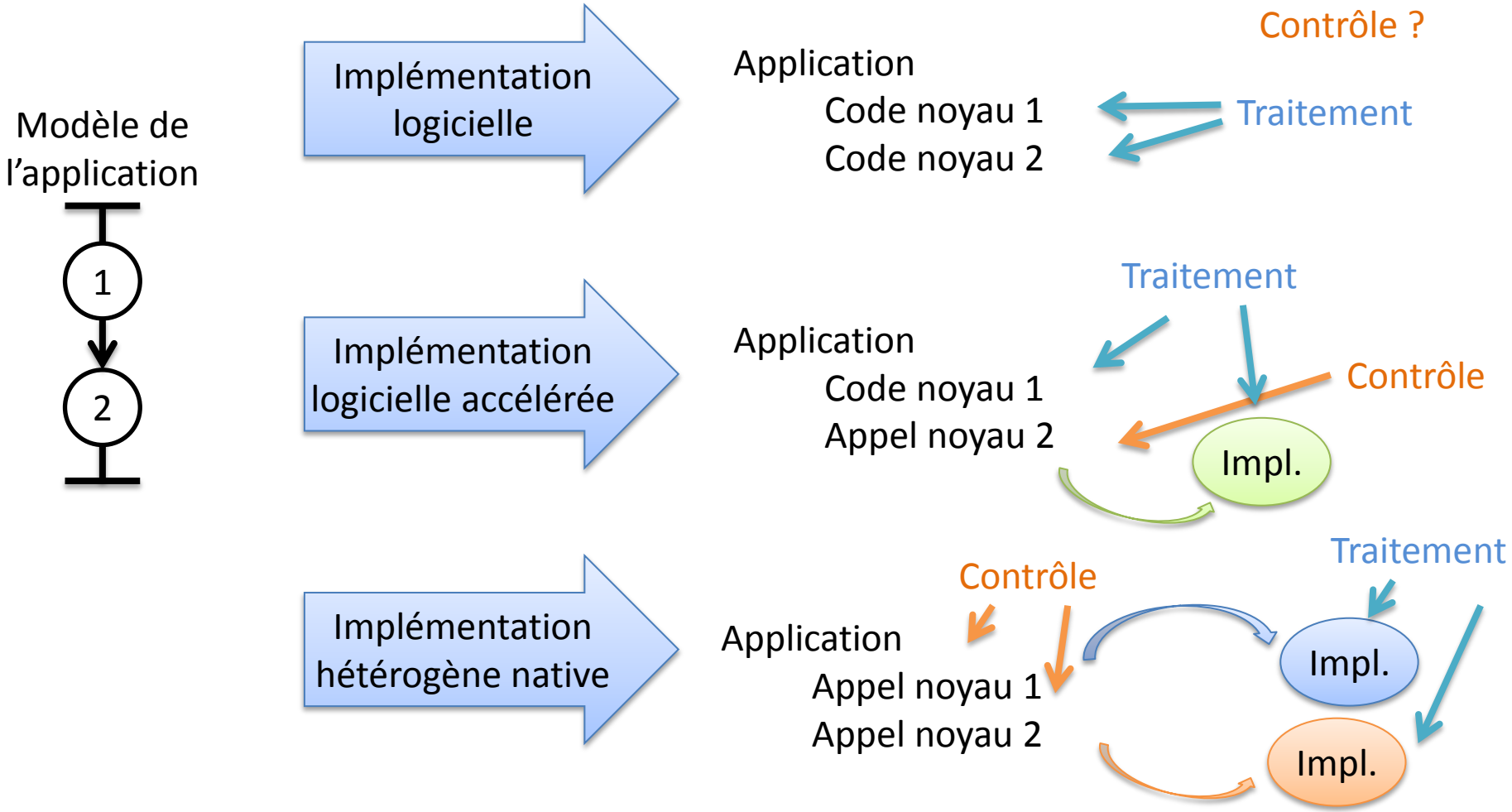


Contrôle

- Hiérarchie
  - Distribution de la couche de contrôle
  - Modèle général utilisé dans les hauts degrés de parallélisme
- Indépendance vis-à-vis des unités d'exécution
  - Portabilité



Modélisation et implémentation



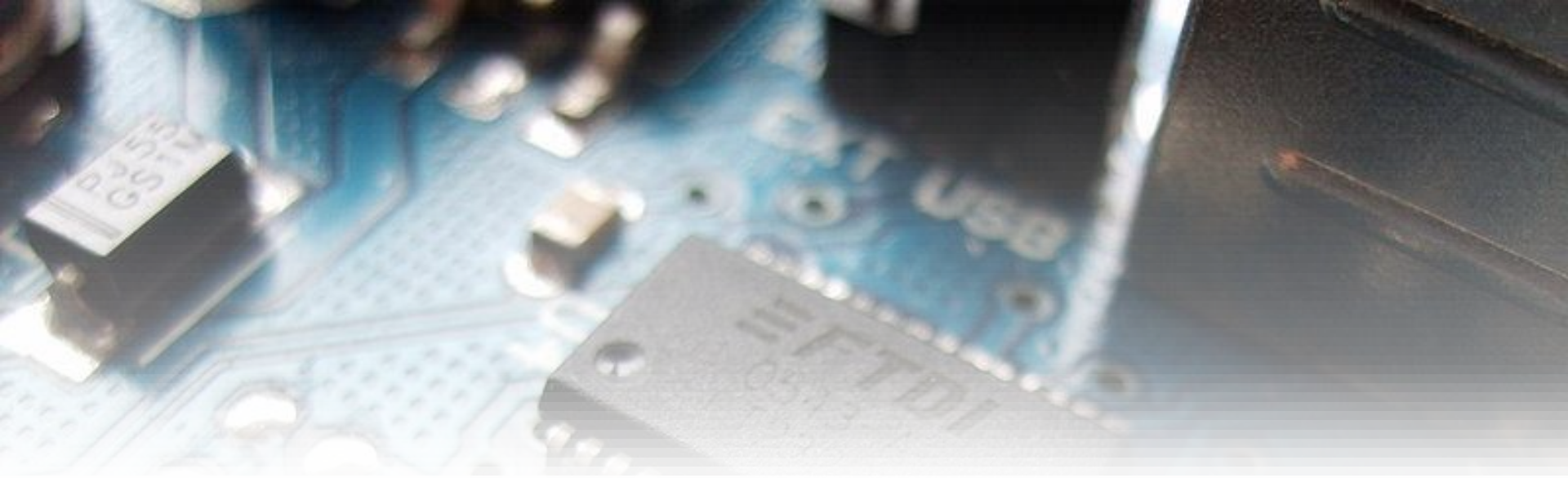
Conclusion

- Destiné aux applications parallèles et à la prise en charge du matériel reconfigurable
- Autoriser plusieurs implémentations pour chaque noyau
  - Logicielles
  - Matérielles
  - Reconfigurables
- Description du contrôle indépendamment de l'architecture d'exécution

Portabilité

Publication : modèle présenté dans une conférence nationale

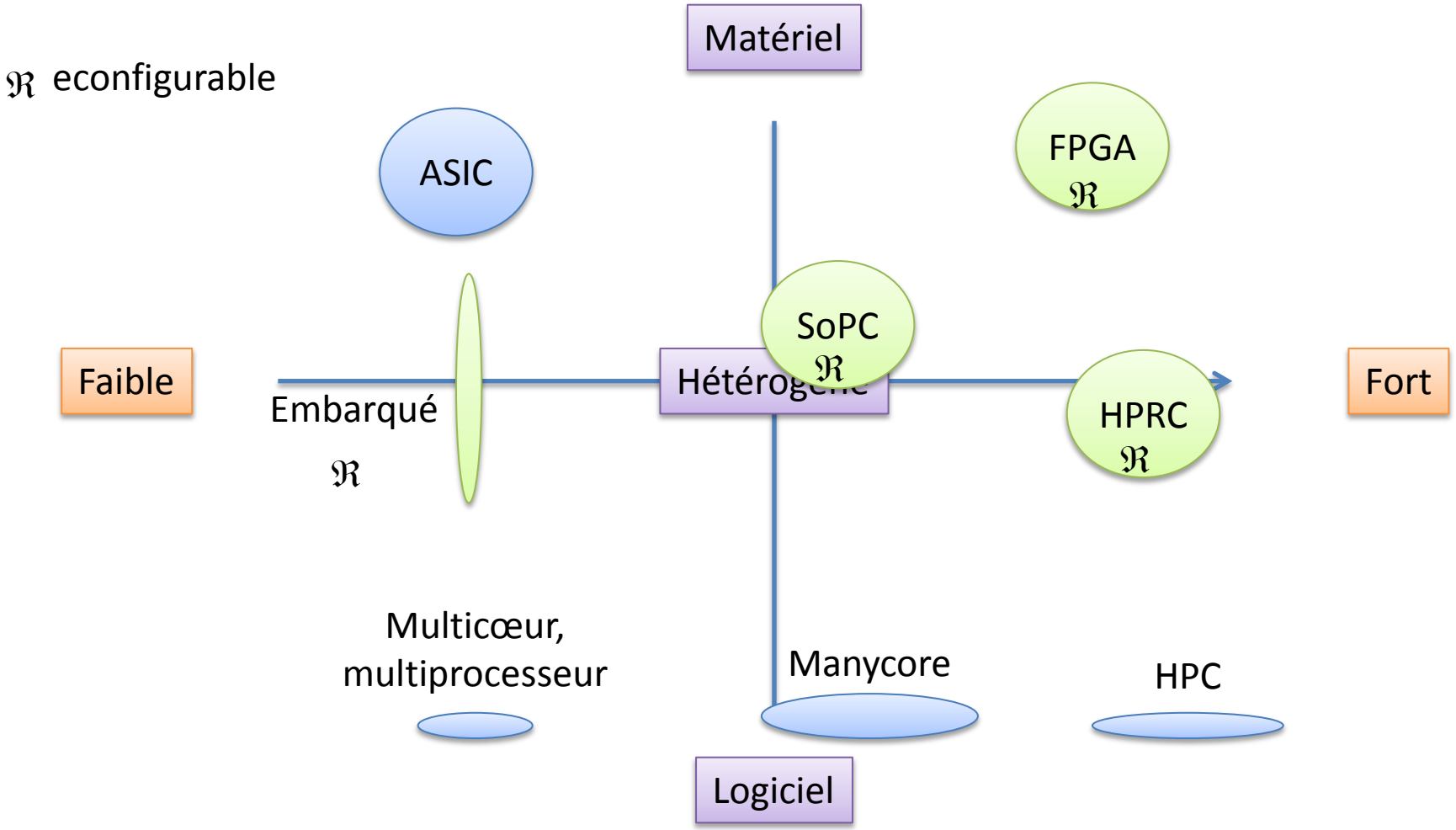
C. Foucher, F. Muller et A. Giulieri, *Flot de conception d'applications parallèles sur plateforme reconfigurable dynamiquement*, 14ème Symposium en Architectures nouvelles de machines (SympA), St. Malo, France, 2011



L'ARCHITECTURE SPoRE

Simple
Parallel
platform for
Reconfigurable
Environment

Plateformes : nature et parallélisme



Plateformes reconfigurables

- High Performance Reconfigurable Computers (HPRC) [6]
  - HPC + FPGA
- Diverses plateformes : MpSoC reconfigurables [7, 8]
  - Multicores/manycores modulaire
- ReMAP [9]
  - PEs logiciels couplés à des zones reconfigurables
  - Utilisation des zones reconfigurables pour communication + calcul
- BORPH [10]
  - « Enrobage » des tâches matérielles dans des processus UNIX
- Egret [11]
  - Plateforme modulaire, possibilité d'introduire des « cartes d'extension » reconfigurables

[6] El-Ghazawi *et al*., George Washington University, *The promise of High-Performance Reconfigurable Computing*, 2008

[7] G. Beltrame *et al.*, European Space Agency and Politecnico di Milano, *High-Level Modeling and Exploration of Reconfigurable MPSoCs,* 2008

[8] L. Ye *et al*., Université Européenne de Bretagne, *Modeling of Reconfigurable MPSoCs for On-Demand Computing*, 2009

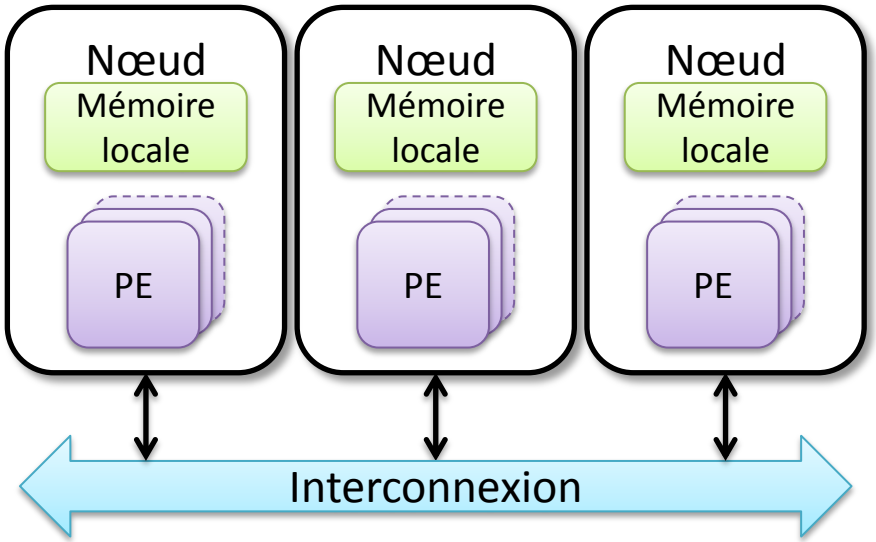
[9] M. Watkins and D. Albonesi, Cornell University, *ReMAP: A Reconfigurable Heterogeneous Multicore Architecture*, 2010

[10] H. So and R. Brodersen, University of California, *A unified hardware/software runtime environment for FPGA-based reconfigurable computers using BORPH*, 2008

[11] Bergmann *et al.*, University of Queensland, *Egret : A flexible platform for real-time reconfigurable systems on chip*, 2003

Mémoire et communication [12]

- Localement partagée
  - Open Multi Processing (OpenMP) [13] : communication par mémoire partagée
- Globalement distribuée
  - Message Passing Interface (MPI) [14] : communication par échange de messages



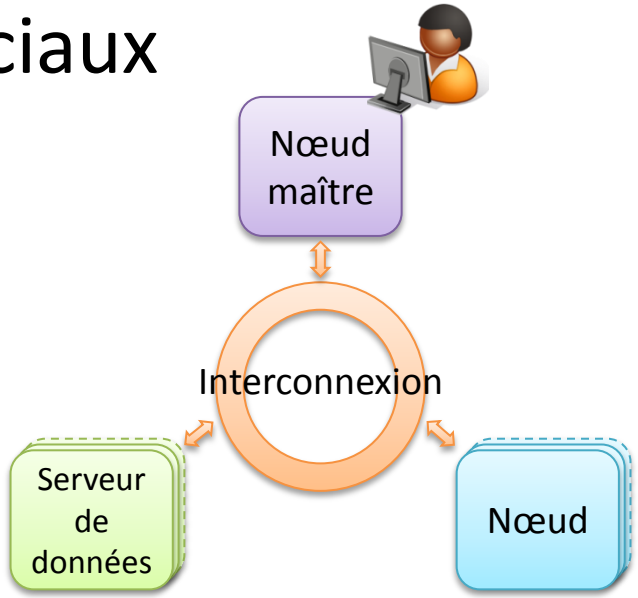
[12]Yoshio Oyanagi, University of Tokyo, *Future of supercomputing*, 2002

[13] Jay P. Hoeflinger *et al.*, Intel, *Parallel programming in OpenMP*, 2008

[14] R. Hempel and D. Walker, Computations and Communications Research Laboratories and University of Wales, *The emergence of the MPI message passing standard for parallel computing*, 1999

La plateforme SPoRE

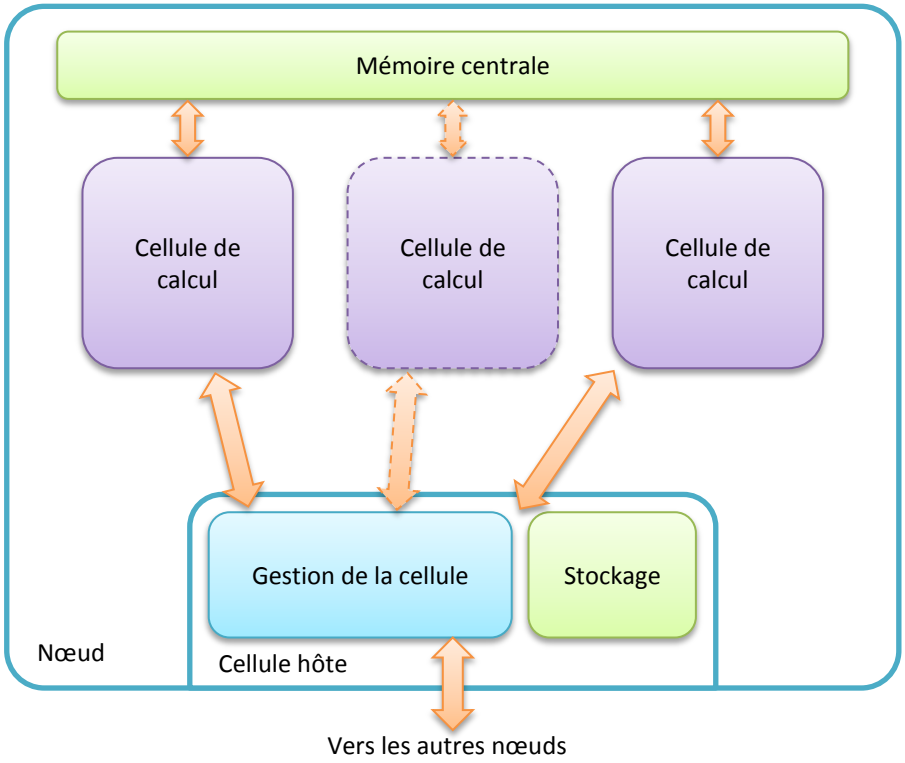
- Architecture mémoire
  - Localement partagée, globalement distribuée
- Introduction de nœuds spéciaux
  - Serveur(s) de données [15]
  - Nœud maitre
    - Ordonnanceur global
    - Point d'entrée



[15] J. Crenne *et al.*, Université Européenne de Bretagne, *End-to-end bitstreams repository hierarchy for FPGA partially reconfigurable systems*, 2011

Les nœuds SPoRE

- Séparation en *cellules*
  - Cellules de calcul
    - Traitement
  - Cellule hôte
    - Contrôle
      - Ordonnanceur local
    - Communication
      - Locale
      - Distante



SPoRE et les architectures existantes

Autres plateformes

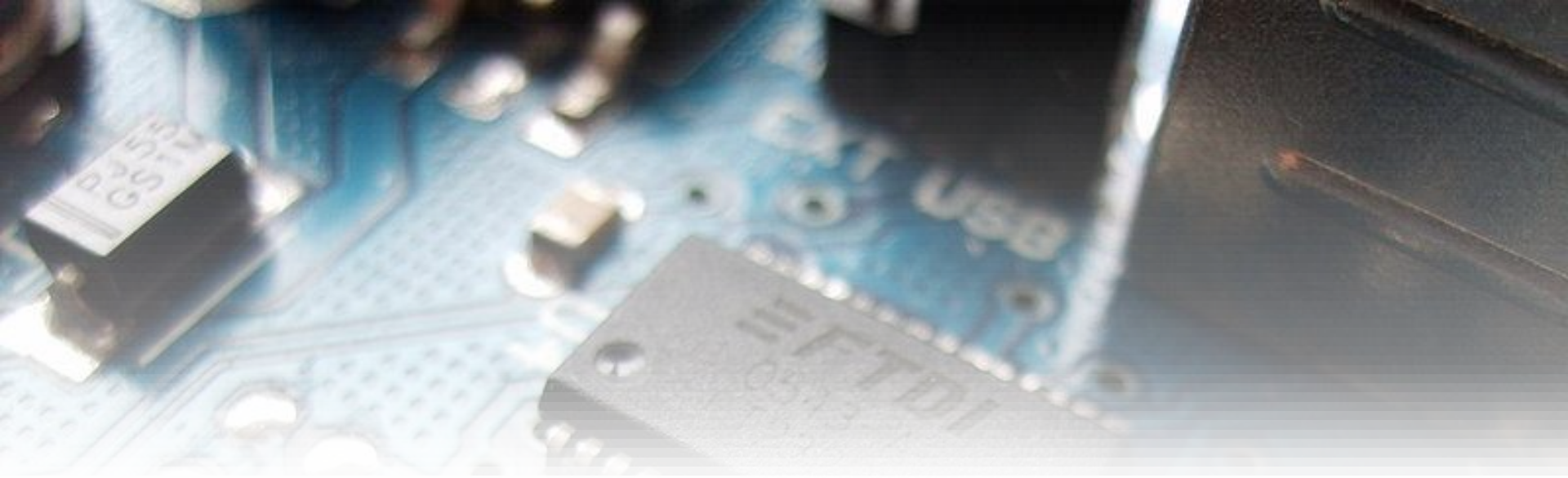
- HPRC
  - Vision logiciel accéléré
- ReMAP
  - Base logicielle, utilisation marginale du matériel (communication, accélération)
- MpSoC
  - Uniquement logiciel
- BORPH
  - Très bonne intégration dans le standard UNIX
  - Difficulté d'adaptation des IPs matériels
- Egret
  - Très forte modularité
  - Intégration temps réel
  - Pas de prise en charge de la reconfiguration partielle

SPoRE

- Vs. HPRC, ReMAP, MpSoC
  - Utilisation de logiciel ou matériel selon les besoins, sans préférence
- Vs. BORPH
  - Meilleure portabilité (indépendance par rapport à UNIX)
  - Réutilisation facilitée d'IPs matériels
- Vs. Egret
  - Prise en charge automatisée du processus de reconfiguration partielle

Conclusion

- Plateforme
  - Distribuée
    - Passage à l'échelle par ajout de nœuds
  - Cohérence avec le modèle d'application
    - Passage immédiat du modèle vers l'implémentation
  - Prise en charge transparente de la reconfiguration dynamique partielle
- Validation du modèle
  - Prototypage
    - Plateforme logicielle MPI
    - Plateforme matérielle reconfigurable flot de données



Plateforme logicielle MPI

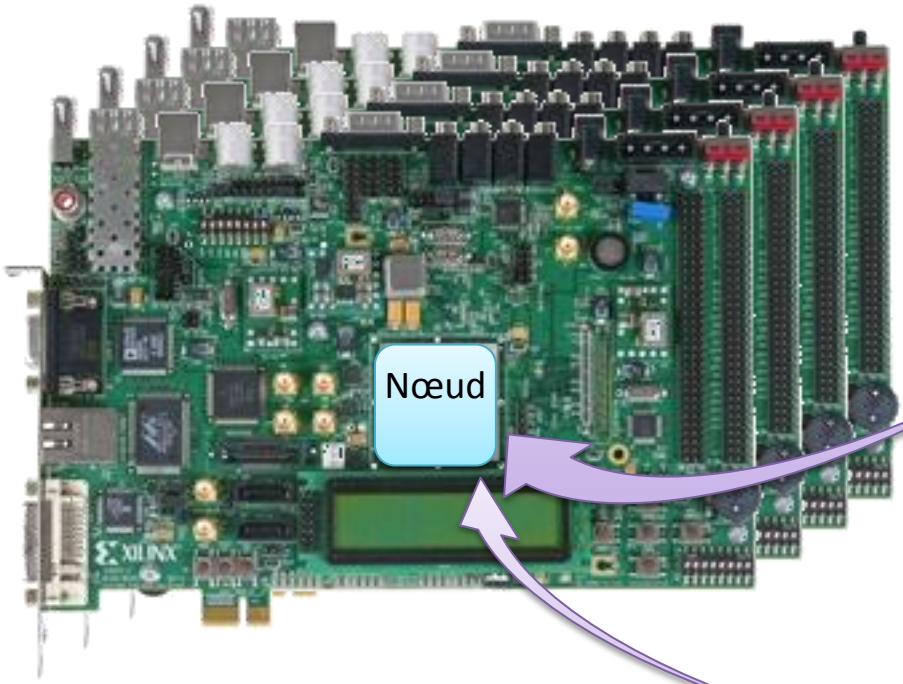
Objectifs

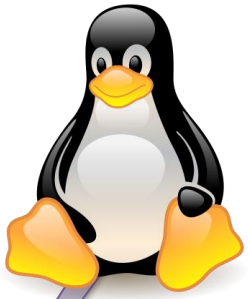
- Vérification de l'adaptation aux forts taux de parallélisme
  - Modèle type HPC
- Validation de la structure locale des nœuds
  - Pertinence de la séparation des rôles
    - Cellule de calcul
    - Cellule hôte
  - Validité du modèle mémoire
    - Mémoire partagée gérée par la cellule hôte

Comment ?

Architecture des nœuds SPoRE

Déploiement sur carte de prototypage Xilinx ml507

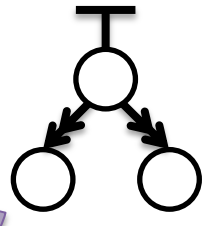




Linux embarqué

SoPC

Exécution d'une application de test



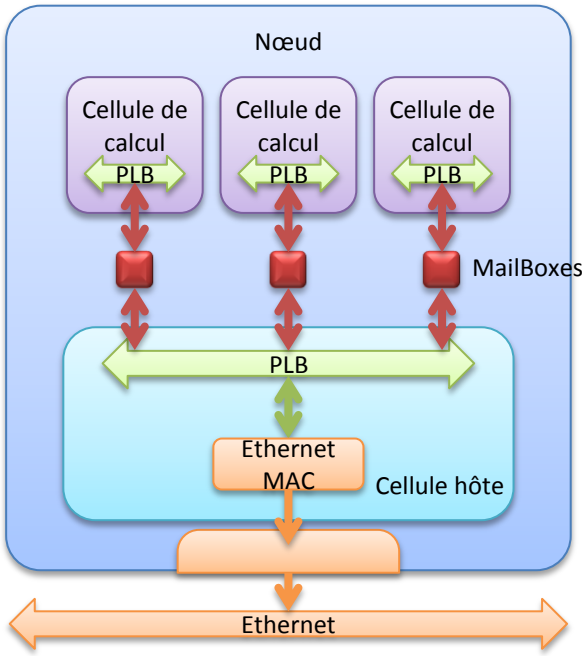
Evaluation des performances

Architecture niveau système

- Plateforme homogène
  - Nœuds identiques
  - Cellules de calcul identiques
    - Exécution logicielle
- Basée sur MPI
  - Nœud spécial : nœud maître
    - Ne procède à aucun calcul
    - Sert de point d'entrée

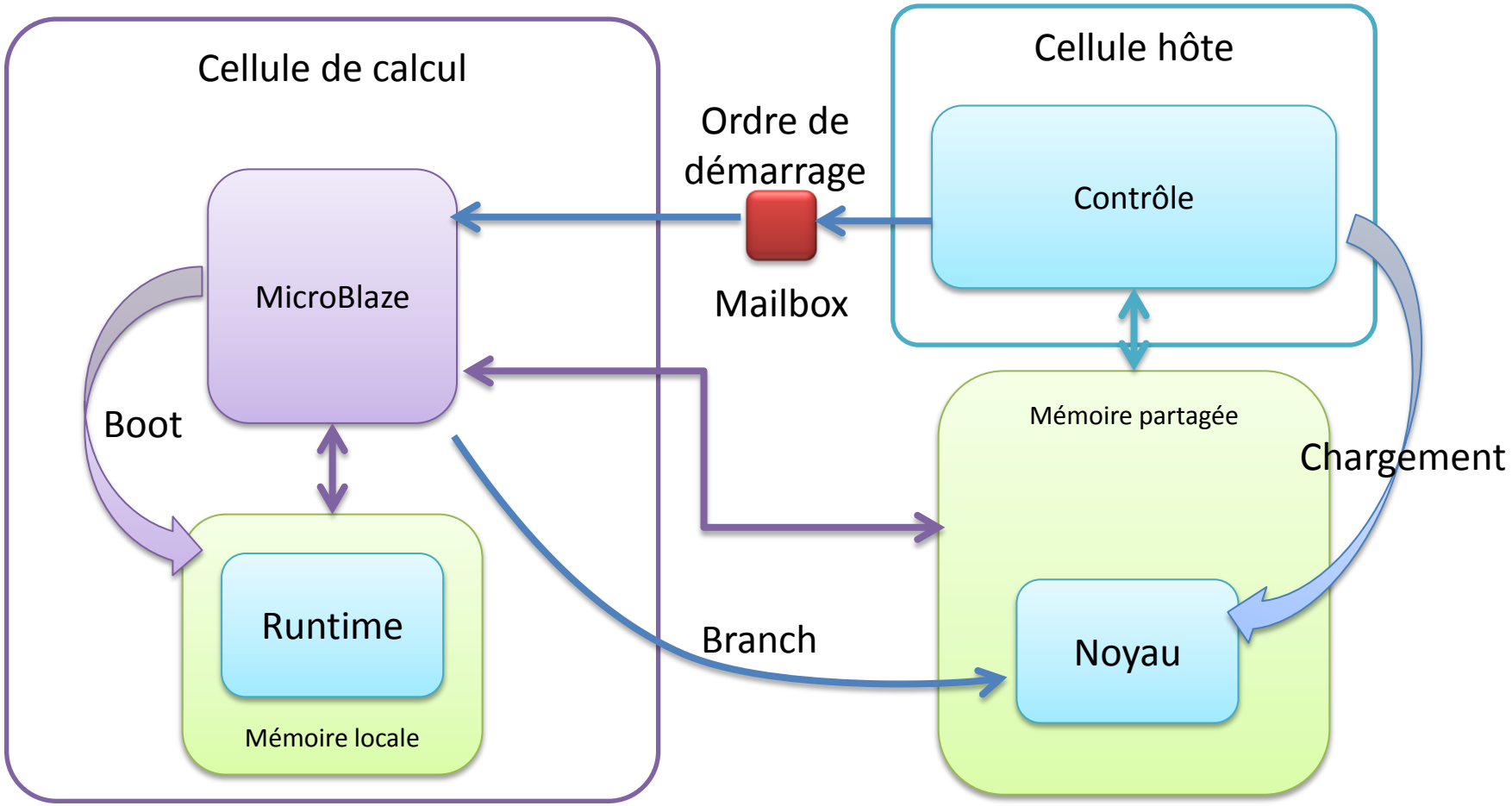
Architecture des nœuds

- Cellule hôte
  - Architecturée autour d'un processeur
    - Gestion logicielle du nœud
      - Linux embarqué
    - Ordonnanceur MPI
  - Pas d'intégration d'OpenMP
    - Nécessiterait de développer un compilateur
- Cellules de calcul
  - MicroBlaze (Xilinx) [16]



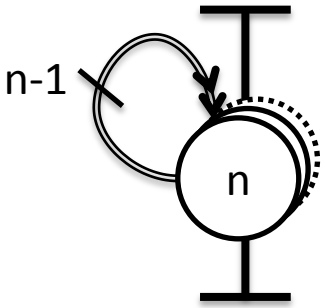
[16] http://www.xilinx.com/tools/microblaze.htm

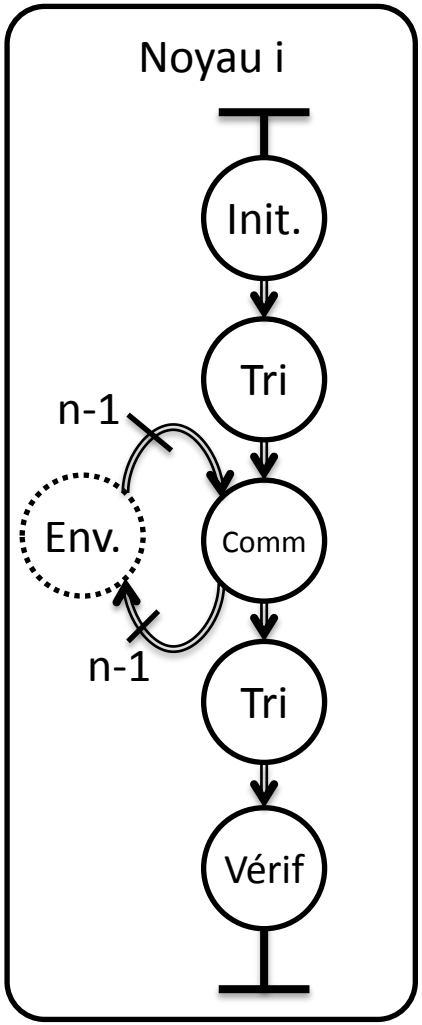
Runtime des cellules de calcul



Application de test

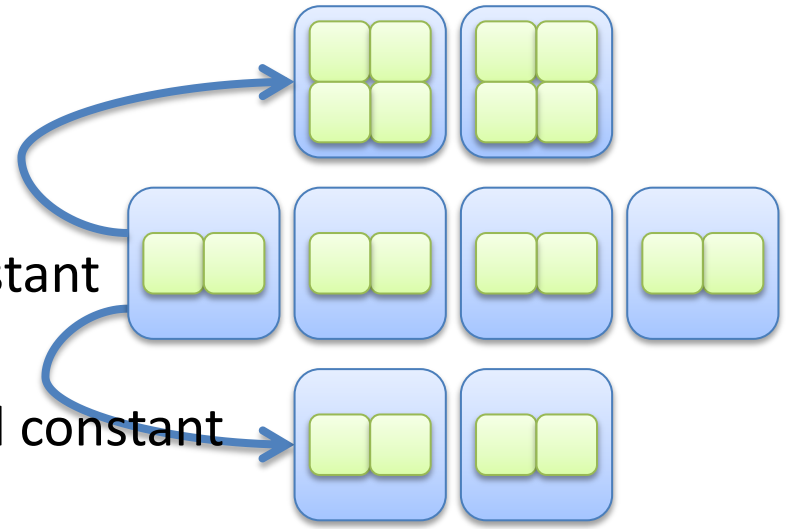
- Utilisation du benchmark NPB IS
  - Teste principalement les communications
- Chaque noyau :
  - Initialisation aléatoire
  - « Tri casier » local
  - Envoi de chaque casier au noyau correspondant
  - Tri local
  - Vérification



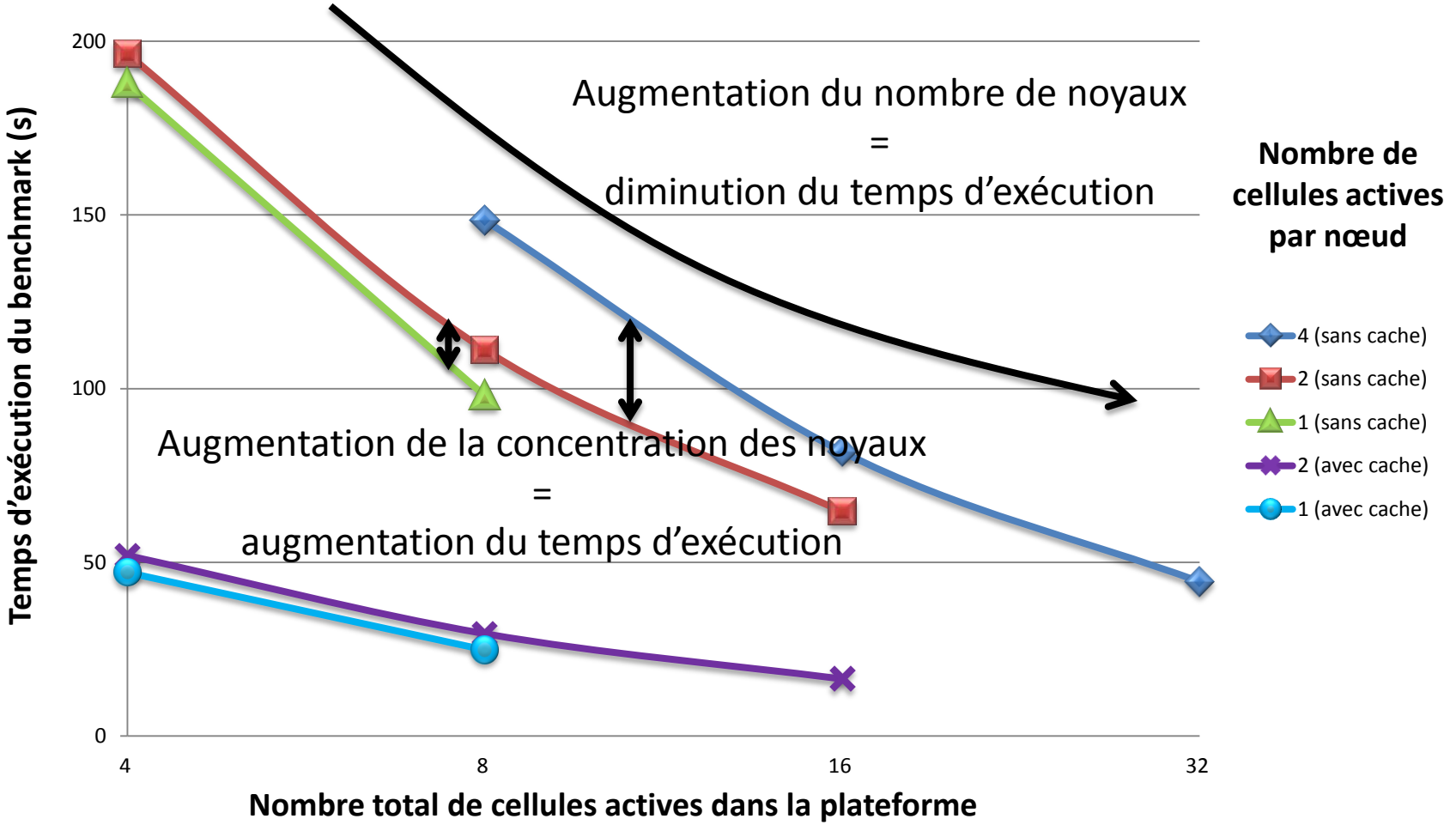


Adaptation et configuration

- Adaptation de l'application
  - Extraction des noyaux tri / communication
    - Remplacement par un *proxy*
    - Création du noyau MicroBlaze correspondant
    - Exécution
- Plusieurs configurations
  - Avec et sans cache
  - Nombre de cellules par nœud
    - A nombre de cellules total constant
  - Nombre de nœuds
    - A nombre de cellules par nœud constant



Validation : résultats



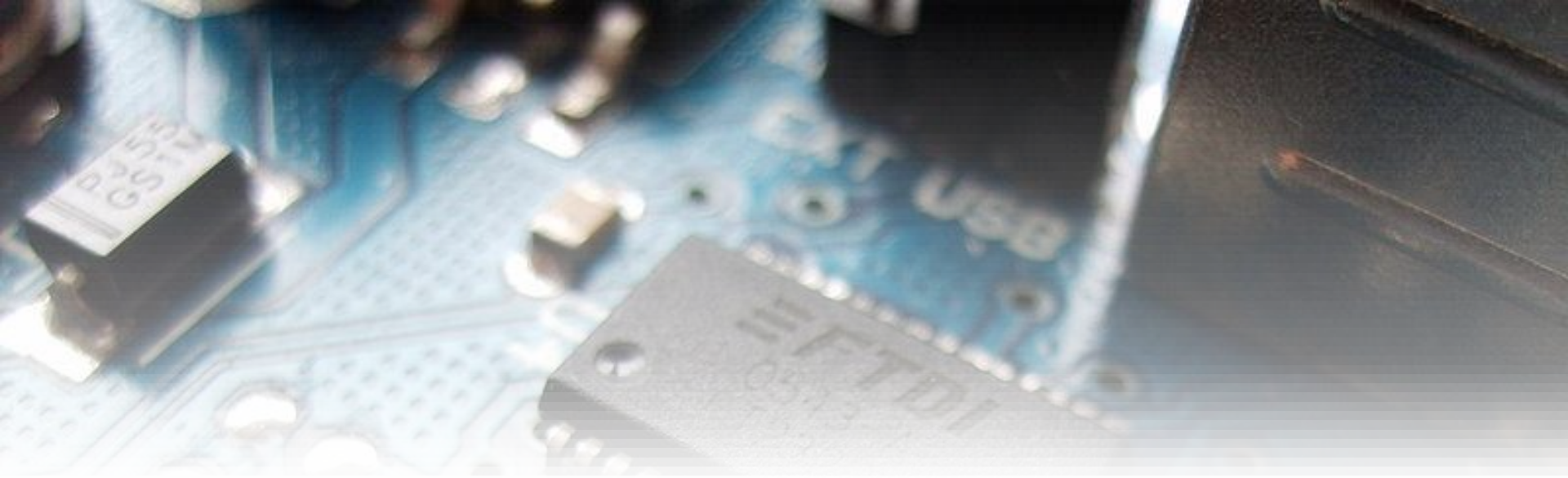
Conclusion

- Validation du principe
  - Adapté aux applications de type HPC
- Problèmes de performance
  - Dus aux échanges MPI au sein du nœud
- Perspectives
  - Modification du modèle mémoire
    - Utilisation d'un système de communication type ReMAP ?
    - MPI matériel [17]
  - Extension aux processeurs hétérogènes

[17] W. Chung *et al.,* Yonsei University , *A Low-Cost Standard Mode MPI Hardware Unit for Embedded MPSoC,* 2011

Publication : plateforme présentée dans une conférence internationale IEEE

C. Foucher, F. Muller and A. Giulieri, *Exploring FPGAs capability to host a HPC design*, 28th Norchip conference, Tampere, Finland, 2010



PLATEFORME MATÉRIELLE RECONFIGURABLE

Objectifs

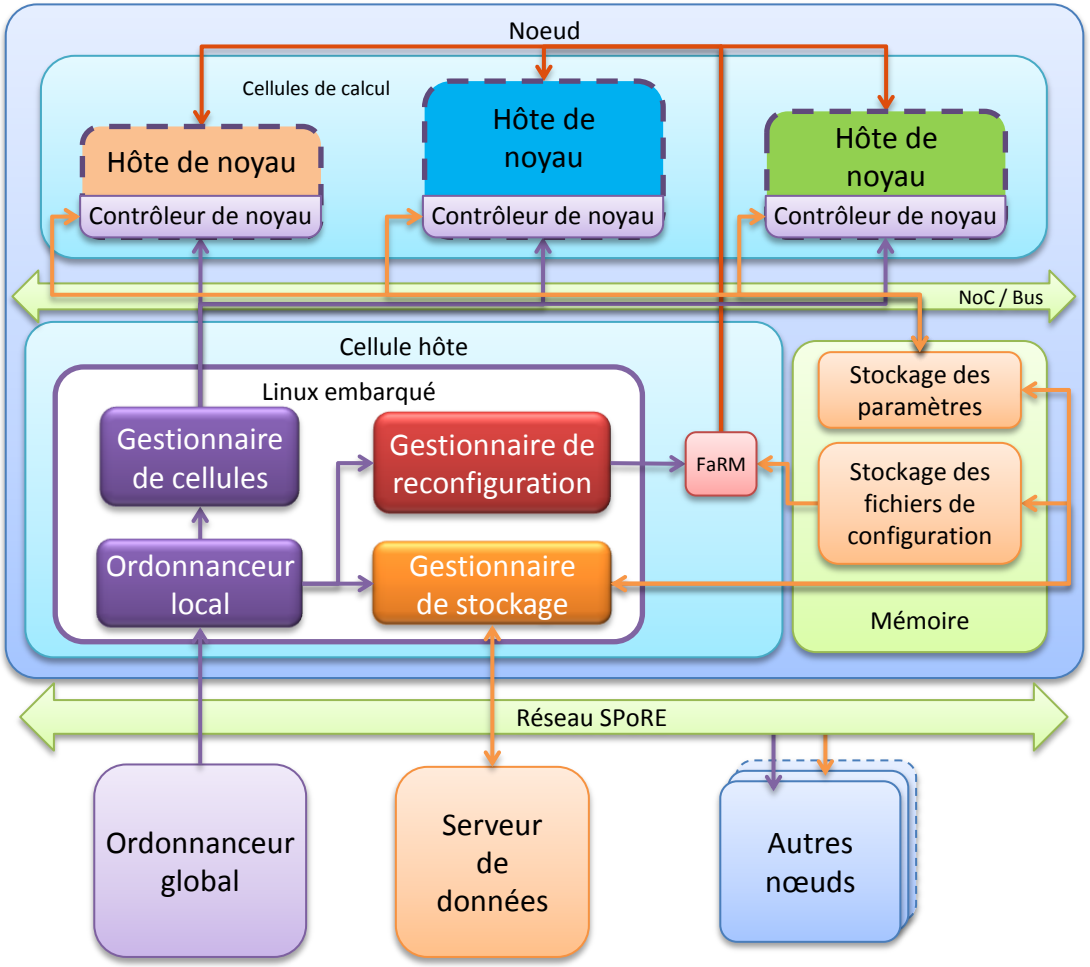
- Gérer la reconfiguration matérielle partielle
  - Processus transparent pour le développeur
- Cohabitation logiciel / matériel
- Concevoir la couche de virtualisation
  - Possibilité de fournir plusieurs implémentations
  - Standardisation des interactions avec les noyaux

Architecture niveau système

- Serveur de données
  - Stockage des éléments de l'application
    - Implémentations, données, description
- Nœud maître
  - Pas d'implémentation MPI
    - Ordonnanceur indépendant

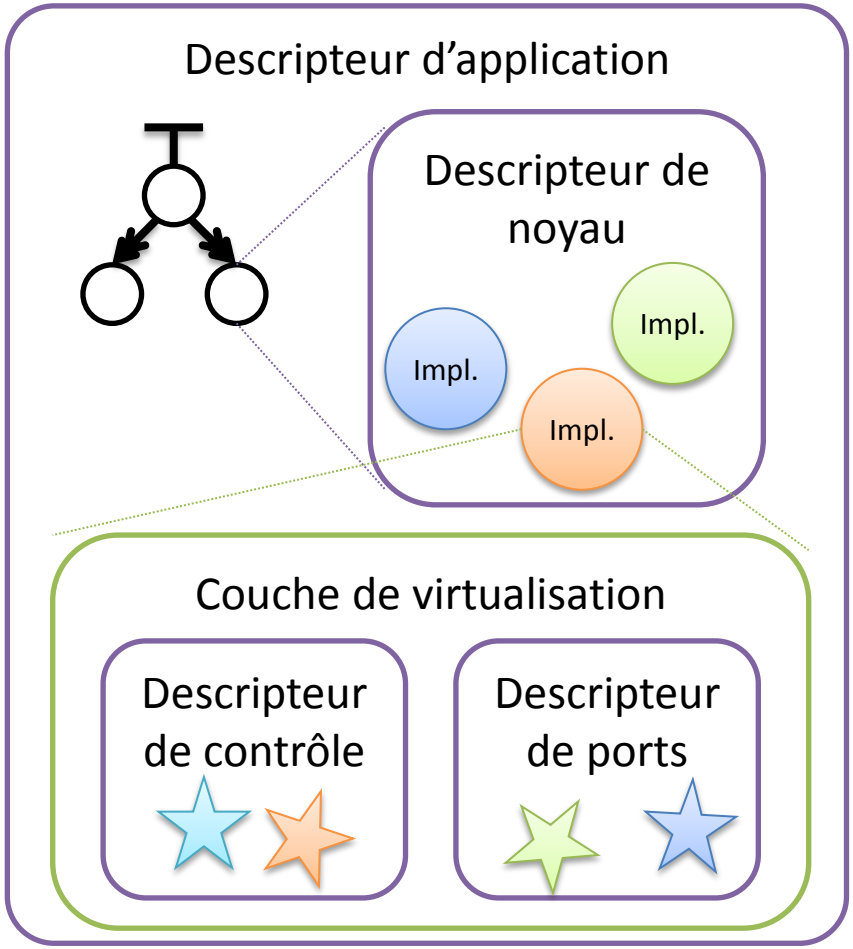
Architecture des nœuds

- Base identique à la plateforme précédente
  - Sauf communication basée sur NoC / Bus
- Cellules de calcul
  - Contrôleur statique
    - Interface avec le reste du nœud
    - Fonctionnalités de monitoring
  - Hôte reconfigurable
- Cellule hôte
  - Runtime de gestion complet



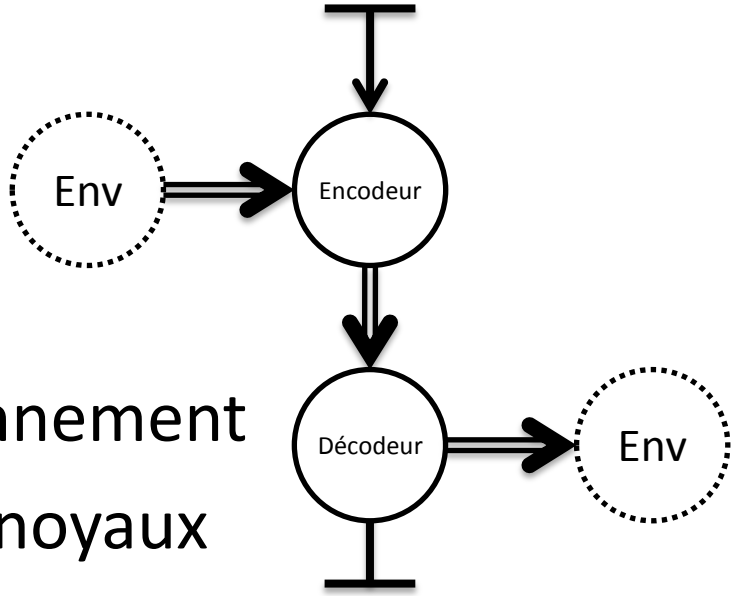
Structure des applications

- Descripteur d'application
  - Liste les noyaux et leurs relations
  - Couche de contrôle
- Descripteur de noyau
  - Implémentations d'un noyau
  - Interaction avec la couche de virtualisation
- Descripteurs de contrôle / ports
  - Décrit les interactions au moyen d'accesseurs
  - Préemption « maîtrisée » du matériel

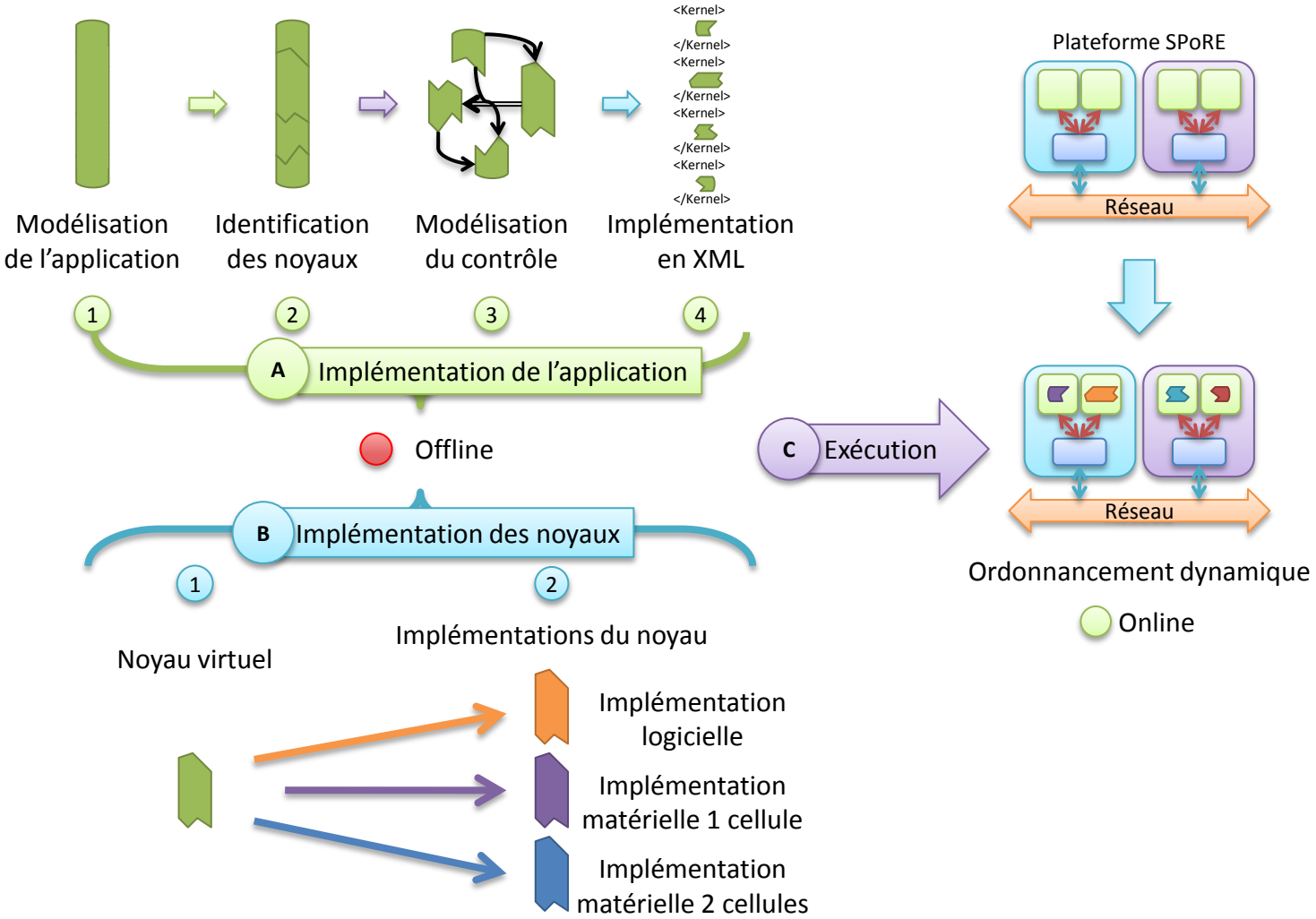


Application de test : AES

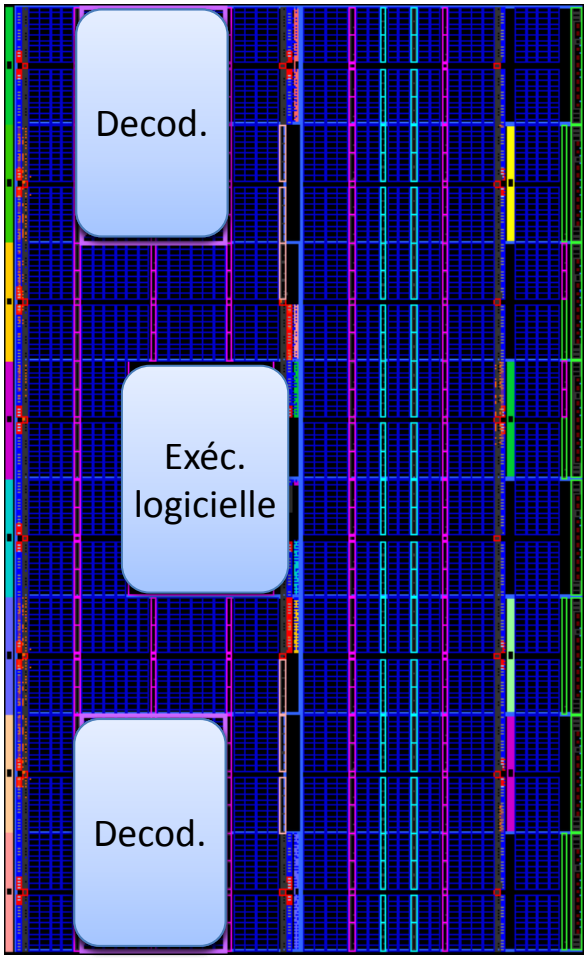
- Modèle d'application « flot de données »
  - Encodage d'un fichier
  - Décodage du fichier
- Transmission de données
  - Récupération depuis l'environnement
  - Passage de résultat entre les noyaux
  - Restitution du résultat à l'environnement



Flot de création de l'application



Noyaux matériels reconfigurables



Evaluation des performances

Quantité de données traitées			512 Kio	1 Mio	5 Mio
Nature des noyaux	Noyau	Durée ...			
Logiciels	Encodeur	d'exécution	426 580 μs ± 0,37%	717 269 μs ± 0,63%	3 049 773 μs ± 0,83%
	Décodeur	d'exécution	550 588 μs ± 0,87%	972 432 μs ± 0,23%	4 349 304 μs ± 0,63%
Matériels (1 cellule)	Encodeur	d'exécution	52 926 μs ± 0,03%	105 847 μs ± 0,02%	529 166 μs ± 0,01%
	Décodeur	exécution	52 [illegible] ± 0,06%	105 [illegible] ± 0,01%	529 178 μs ± 0,01%
	Encodeur	de reconfiguration	825 μs ± 0,67%		
	Décodeur	de reconfiguration	906 μs ± 0,12%		
	Encodeur	totale	53 751 μs ± 0,03%	106 673 μs ± 0,02%	529 991 μs ± 0,01%
	Décodeur	totale	53 848 μs ± 0,06%	106 759 μs ± 0,01%	530 084 μs ± 0,01%
Matériels (2 cellules)	Encodeur	d'exécution	28 541 μs ± 0.15%	57 974 μs ± 0.08%	285,656 μs ± 0.08%
	Décodeur	exécution	28 [illegible] ± 0.03%	57 [illegible] ± 0.04%	271,887 μs ± 0.05%
	Encodeur	de reconfiguration	1,671 μs ± 0.46%		
	Décodeur	de reconfiguration	1,820 μs ± 0.21%		
	Encodeur	totale	30,177 μs ± 0.15%	59,642 μs ± 0.09%	287,329 μs ± 0.08%
	Décodeur	totale	28,670 μs ± 0.03%	55,912 μs ± 0.04%	273,706 μs ± 0.05%

-87 %

-85 %

-82 %

-44 %

-44 %

-46 %

Conclusion

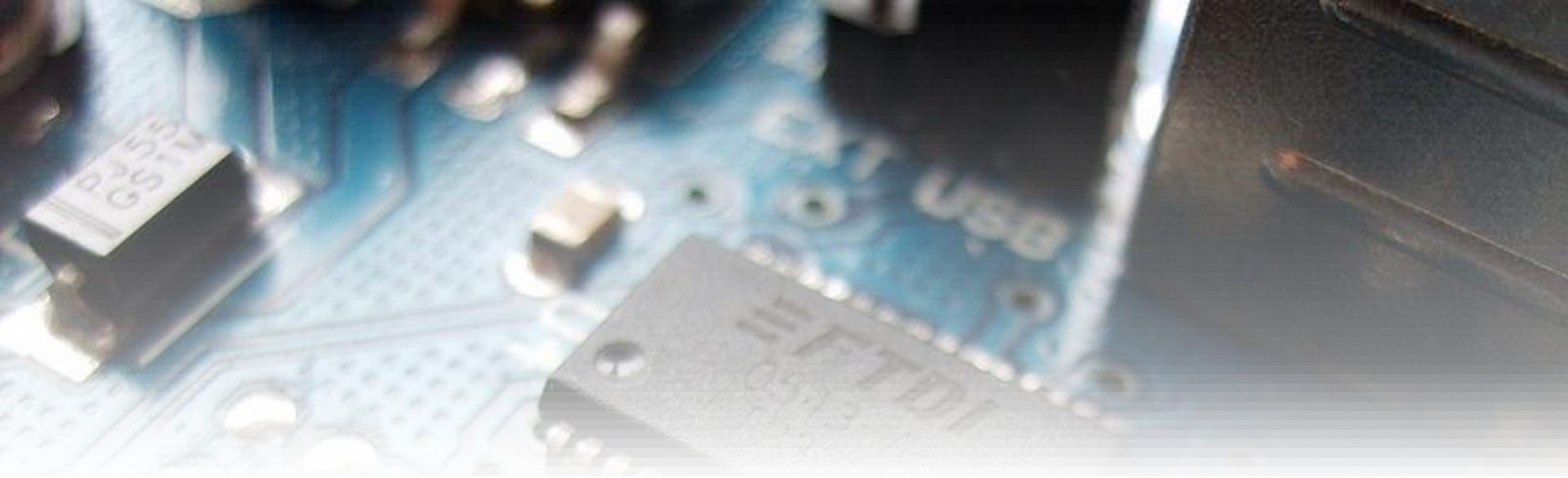
- Fonctionnement
  - Interactions avec les IPs de type registre / plage mémoire
  - Echanges possibles par mémoire partagée
- Perspectives
  - Développement de l'aspect ordonnancement
  - Utilisation des données de monitoring pour l'auto-adaptativité

Publications : travaux sur la plateforme et résultats

Journal international : C. Foucher, F. Muller and A. Giulieri, *Online codesign on reconfigurable platform for parallel computing*, Microprocessors and Microsystems, ISSN 0141-9331, 10.1016/j.micpro.2011.12.007, 2012

Journal national : C. Foucher, F. Muller et A. Giulieri, *Méthodologie dédiée aux applications parallèles sur plateforme reconfigurable dynamiquement*, Technique et Science Informatiques, 2012

Conférence internationale IEEE : C. Foucher, F. Muller and A. Giulieri, *Fast Integration of Hardware Accelerators for Dynamically Reconfigurable Architecture*, 7th International Workshop on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), York, U.K., 2012



CONCLUSION

- Introduction
- Modèle d'application
- L'architecture SPoRE
- Plateforme logicielle MPI
- Plateforme matérielle reconfigurable
- Conclusion
  - Bilan
  - Perspectives

Bilan

- Proposition d'une méthodologie de développement d'applications et d'un modèle associé
  - Applications parallèles hétérogènes
  - Gestion transparente de la reconfiguration partielle
- Validation au travers de deux plateformes
  - Plateforme logicielle
    - Orientée HPC
  - Plateforme matérielle reconfigurable
    - Orientée flot de données
    - Modèle embarqué
- Publications
  - Nationales et internationales, conférences et journaux

Perspectives d'évolution

- Plateforme « de la grande unification »
  - MPI
  - Reconfiguration matérielle
  - Réseaux MPI matériels reconfigurables [18]
- Intégration dans un flot de conception automatisé
  - Outils FoRTReSS (en cours de publication)
    - Flow for Reconfigurable archiTectures in Real-time SystemS
  - Conception graphique, génération automatique des XML
    - IP-XACT [19]
- Développement de l'aspect auto-adaptatif
  - Intégration de nouvelles métriques
    - Energie
    - Température
  - Contrôleur de noyau « intelligent »
    - Déclenchement d'interruptions pour informer le système de manière asynchrone

[18] S. Gao *et al*., University of Southern California,*Impact of reconfigurable hardware on accelerating MPI_Reduce*, 2010

[19] www.spiritconsortium.org

Perspectives d'utilisation

- Accélération du time-to-market.
  - Création d'applications sur la base de la réutilisation
  - Prise en charge de la reconfiguration partielle y compris pour des IPs statiques existants
    - Faible niveau d'efforts requis pour l'adaptation
- Plateforme de développement
  - Rapidité de déploiement de tests sur des IPs matériels
  - Exécution simultanée de plusieurs tests différents
- Test d'algorithmes d'ordonnancement
  - Structure modulaire autorisant la modification des deux niveaux d'ordonnanceurs
  - Possibilité de faire remonter des informations
    - Auto-adaptativité
    - Algorithmes évolutifs et apprentissage

Merci pour votre attention

cnrs dépasser les frontières

Université Nice Sophia Antipolis

LABORATOIRE D'ELECTRONIQUE
ANTENNES ET TELECOMMUNICATIONS

Questions

dépasser les frontières

Université Nice Sophia Antipolis

Bibliographie

[1] Peter Thoman, Klaus Kofler, Heiko Studt, John Thomson, and Thomas Fahringer. Automatic OpenCL device characterization : Guiding optimized kernel design. In Emmanuel Jeannot, Raymond Namyst, and Jean Roman, editors, Euro-Par 2011 Parallel Processing, volume 6853 of Lecture Notes in Computer Science, pages 438–452. Springer Berlin / Heidelberg, 2011

[2] http://www.altera.com/literature/wp/wp-01173-opencl.pdf

[3] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. Scalable parallel programming with CUDA. In ACM SIGGRAPH 2008 classes, SIGGRAPH '08, pages 16 :1–16 :14, New York, NY, USA, 2008. ACM.

[4] http://www.xilinx.com/products/design-tools/vivado/integration/esl-design/hls/index.htm

[5] http://www.calypto.com/catapult_c_synthesis.php

[6] Tarek El-Ghazawi, Esam El-Araby, Miaoqing Huang, Kris Gaj, Volodymyr Kindratenko, and Duncan Buell. The promise of High-Performance Reconfigurable Computing. Computer, 41 :69–76, February 2008.

[7] Giovanni Beltrame, Luca Fossati, and Donatella Sciuto. 2008. High-Level Modeling and Exploration of Reconfigurable MPSoCs. In Proceedings of the 2008 NASA/ESA Conference on Adaptive Hardware and Systems (AHS '08). IEEE Computer Society, Washington, DC, USA, 330-337. DOI=10.1109/AHS.2008.15 http://dx.doi.org/10.1109/AHS.2008.15

[8] Ye L., Diguet J.-P., Gogniat G., Reconfigurable MPSoCs for On-Demand Computing. In GRETSI 2009, Dijon : France (2009)

[9] Matthew A. Watkins and David H. Albonesi. 2010. ReMAP: A Reconfigurable Heterogeneous Multicore Architecture. In Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO '43). IEEE Computer Society, Washington, DC, USA, 497-508. DO=10.1109/MICRO.2010.15 http://dx.doi.org/10.1109/MICRO.2010.15

[10] Hayden Kwok-Hay So and Robert Brodersen. A unified hardware/software runtime environment for fpga-based reconfigurable computers using borph. ACM Trans. Embed. Comput. Syst., 7 :14 :1–14 :28, January 2008.

[11] Neil Bergmann, John Williams, and Peter Waldeck. Egret : A flexible platform for real-time reconfigurable systems on chip. In International Conference on Engineering of Reconfigurable Systems and Algorithms, pages 300–303, Las Vegas, USA, 2003.

[12] Yoshio Oyanagi. Future of supercomputing. J. Comput. Appl. Math., 149(1):147–153, 2002.

[13] Jay P. Hoeflinger and Bronis R. De Supinski. The OpenMP memory model. In Proceedings of the 2005 and 2006 international conference on OpenMP shared memory parallel programming, IWOMP'05/IWOMP'06, pages 167–177, Berlin, Heidelberg, 2008. Springer-Verlag.

[14] Rolf Hempel and David W. Walker. The emergence of the MPI message passing standard for parallel computing. Comput. Stand. Interfaces, 21 :51–62, May 1999.

[15] J. Crenne, P. Bomel, G. Gogniat, J.P. Diguet, End-to-end bitstreams repository hierarchy for FPGA partially reconfigurable systems, in: G. Gogniat, D. Milojevic, A. Morawiec, A. Erdogan (Eds.), Algorithm–Architecture Matching for Signal and Image Processing, Lecture Notes in Electrical Engineering, vol. 73, Springer, 2011, pp. 171–194.

[16] http://www.xilinx.com/tools/microblaze.htm

[17] W. Chung *et al.,* A Low-Cost Standard Mode MPI Hardware Unit for Embedded MPSoC. *IEICE Transactions 94-D(7):1497-1501* (*2011*)

[18] Gao, S.; Schmidt, A. G. & Sass, R. (2010), Impact of reconfigurable hardware on accelerating MPI_Reduce., *in* Jinian Bian; Qiang Zhou; Peter Athanas; Yajun Ha & Kang Zhao, ed., 'FPT' , IEEE, , pp. 29-36 .

[19] SPIRIT Schema Working Group Membership. IP-XACT User Guide v1.2. www.spiritconsortium.org, July 2006.