FREIA ANR-AF-2007-004

Présentation finale projet FREIA
Congrès ANR à Lyon

4 janvier 2012

Michel Bilodeau (CMM MINES ParisTech/ARMINES)
Christophe Clienti (Thales TRT, CMM MINES ParisTech/ARMINES)
Fabien Coelho (CRI MINES ParisTech/ARMINES)
Serge Guelton (Télécom Bretagne)
François Irigoin (CRI MINES ParisTech/ARMINES)
Ronan Keryell (Télécom Bretagne)
Fabrice Lemonnier (Thales TRT)

Freia ANR-AF-2007-004

4 janvier 2012

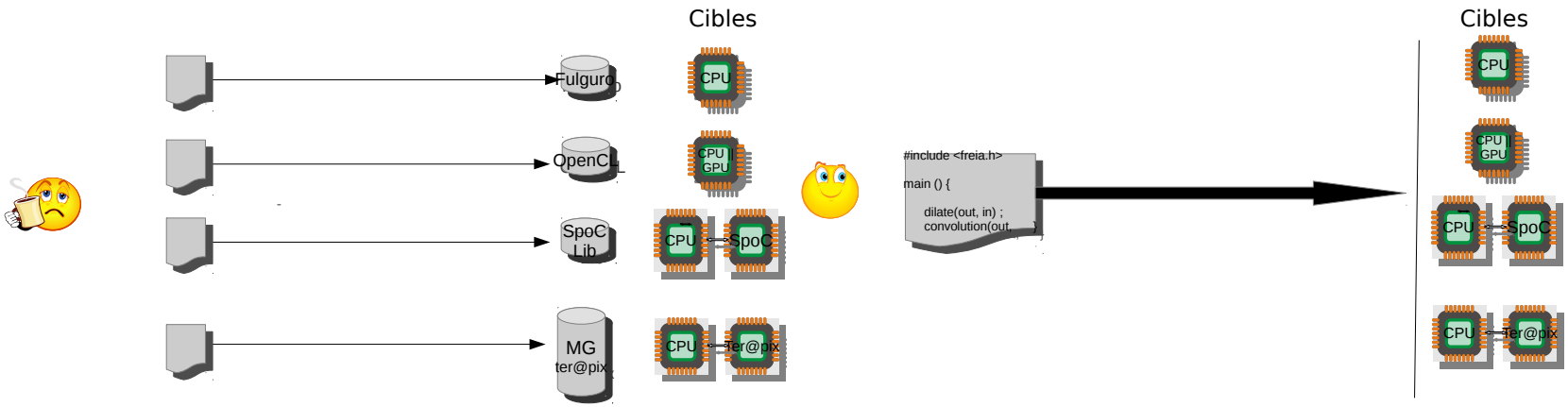
rev:2061

Objectifs

"... vise à concevoir et implémenter une plate-forme flexible et efficace de traitement d'images ET ses outils d'optimisation pour des applications en embarqué..."
"... dans les domaines de la surveillance, du transport..."

Motivations

Développement d'applications



Introduction

Plate-forme et interfaces logicielles

Outils d'optimisation

Conclusion

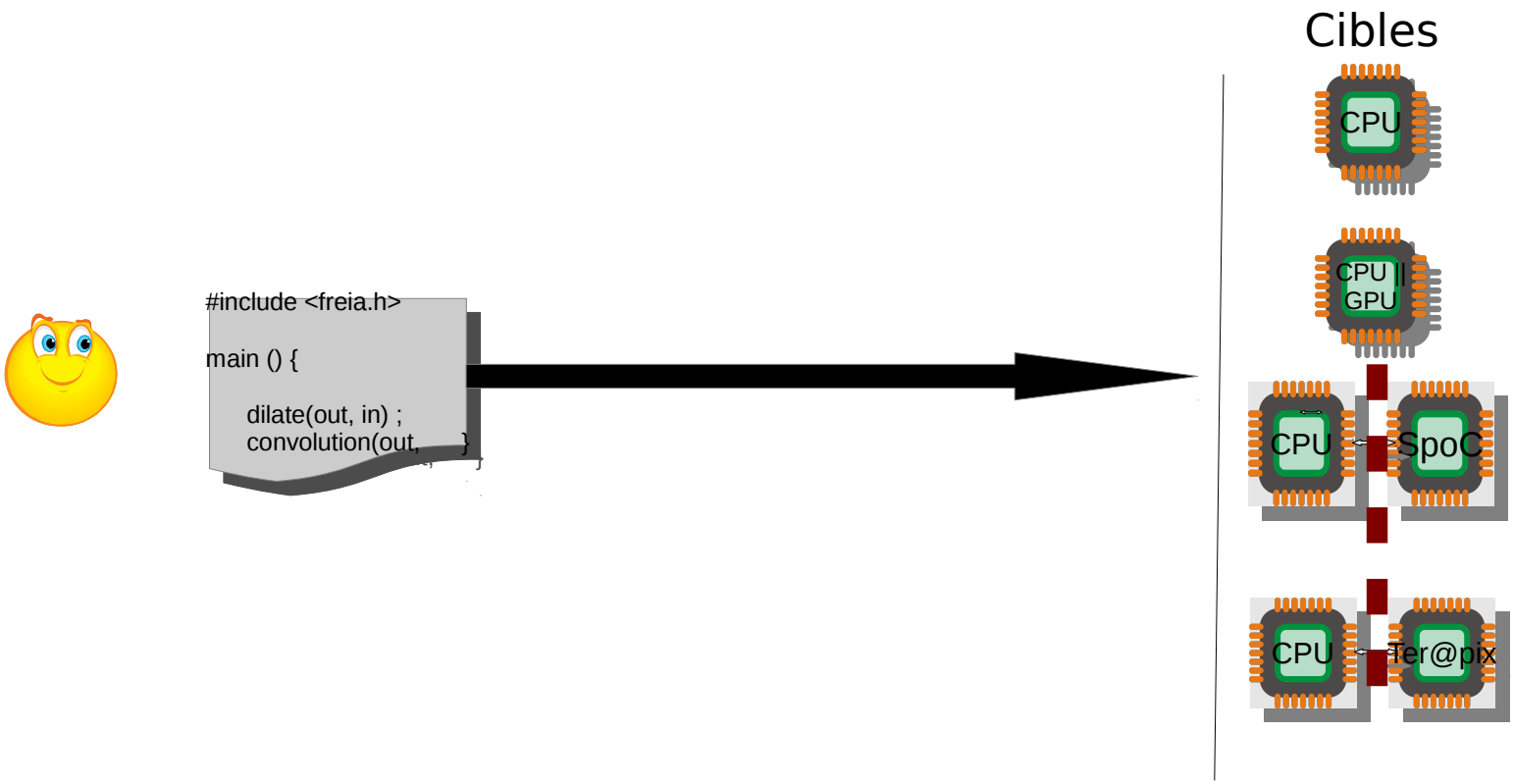
Introduction

Plate-forme et interfaces logicielles

Outils d'optimisation

Conclusion

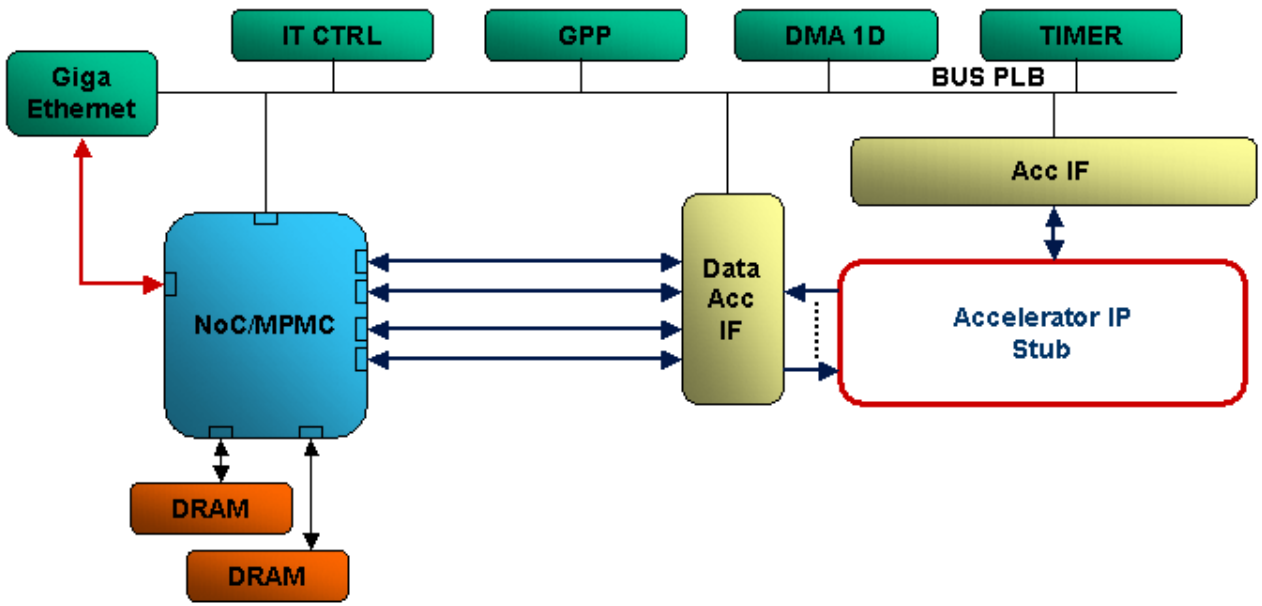
FREIA — Infrastructure et interfaces matérielles



Contributions: Thales, CMM

WP1: Infrastructure d'accueil

- Contrôle générique de l'accélérateur
- Pilotage générique du transfert de données
- Modèle maître/esclave asynchrone basé sur un protocole de requête et de synchronisation

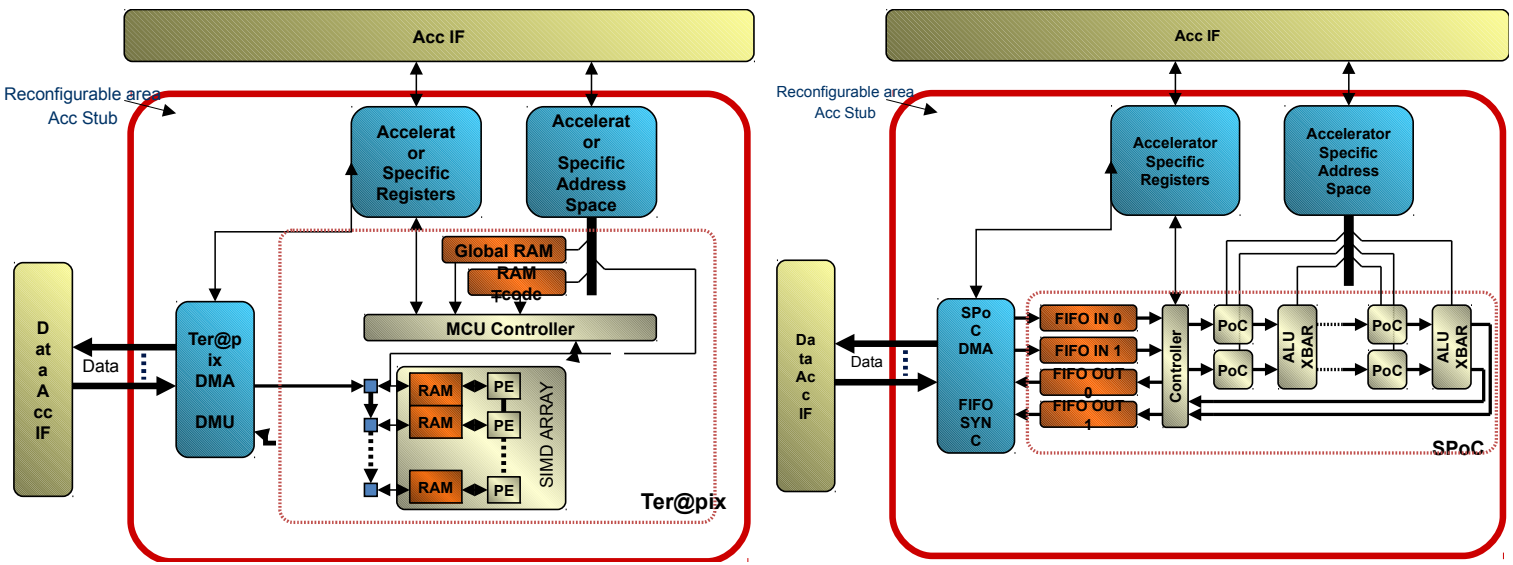


WP1: Interfaces matérielles

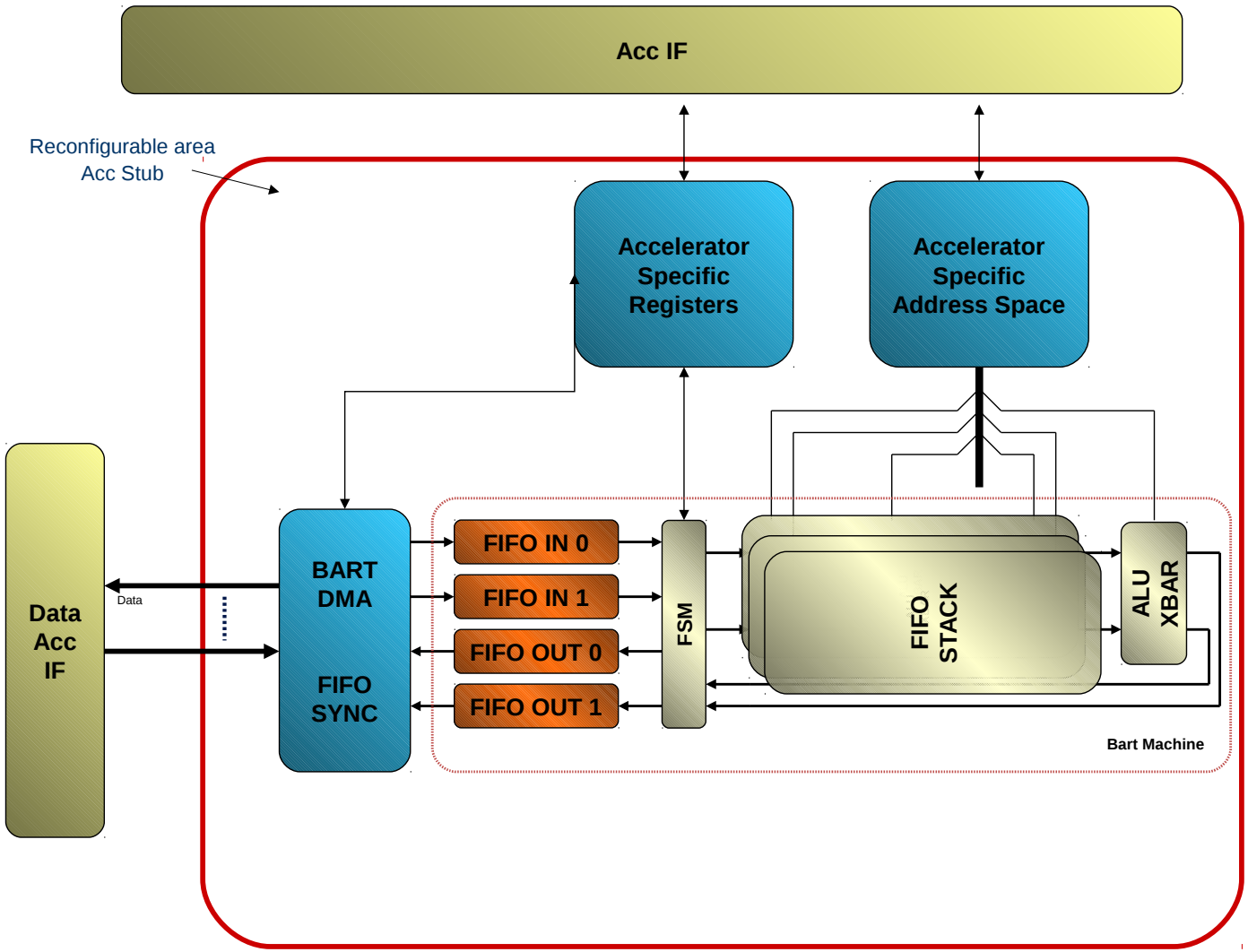
Définition d'interfaces matérielles génériques :

- Interface d'échange de données :
  - Livraison des données sur plusieurs canaux
  - Spécificité d'acheminement laissée à l'accélérateur
- Interface de contrôle :
  - Contrôle de l'acheminement des données
  - Contrôle du séquencement des calculs
  - Gestion des synchronisations transferts/calculs programmable

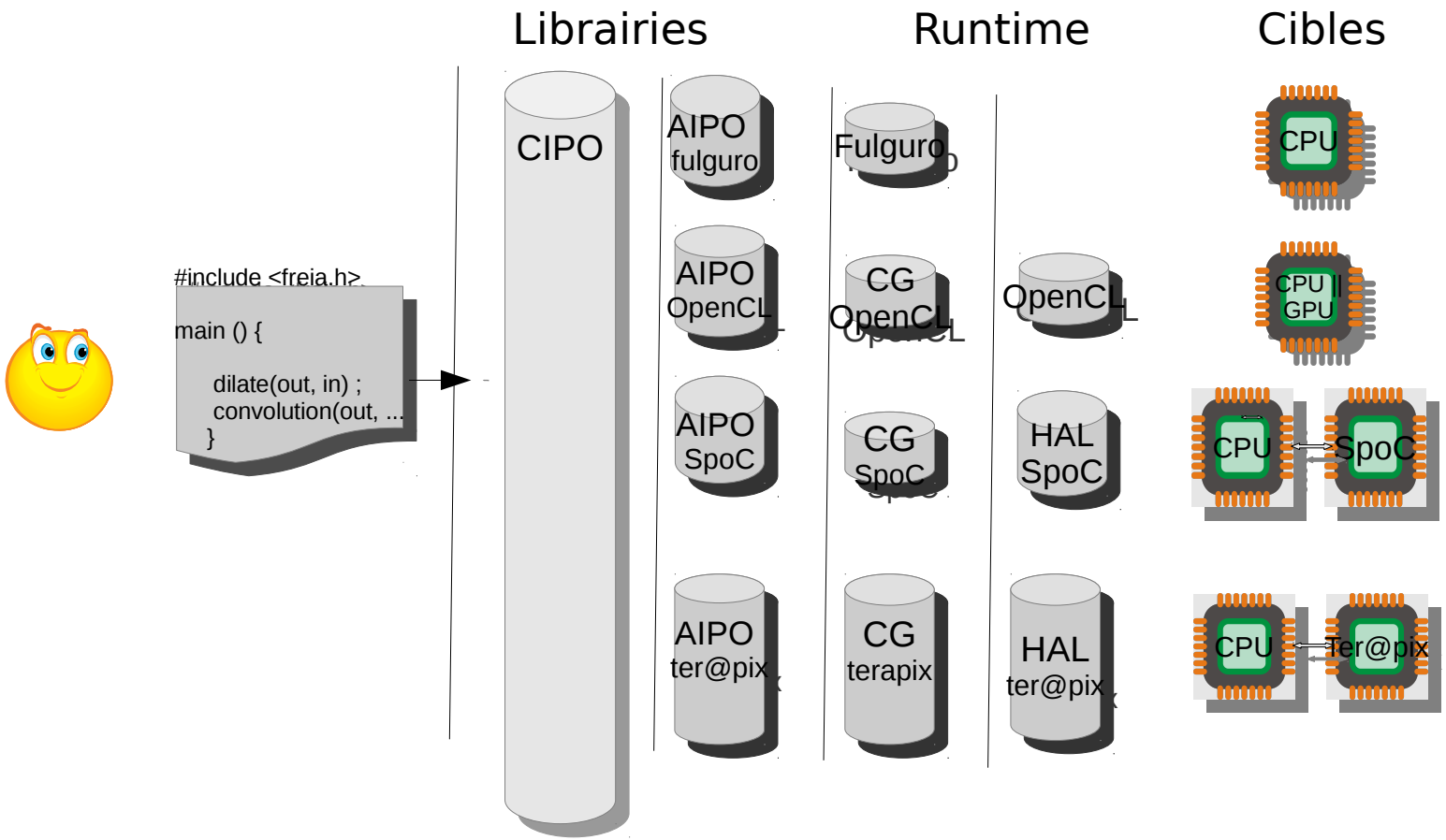
$\rightarrow$ Garantir l'interchangeabilité des accélérateurs au sein de la structure d'accueil



WP1: Nouveau processeur voisinage grande taille



FREIA — interfaces logicielles et run-time



Contributions: Thales, CMM

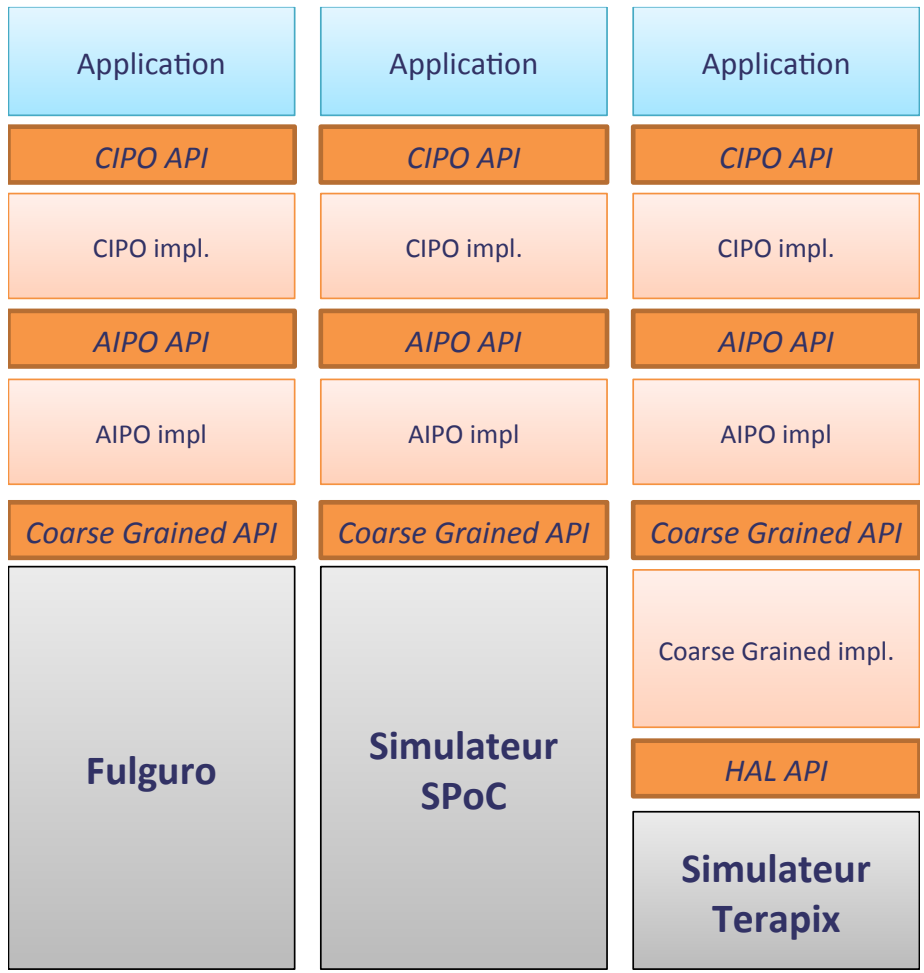
WP1: Librairie d'exécution – Ordonnancement

Ordonnancement mis en œuvre dans la couche CG, deux grandes classes :

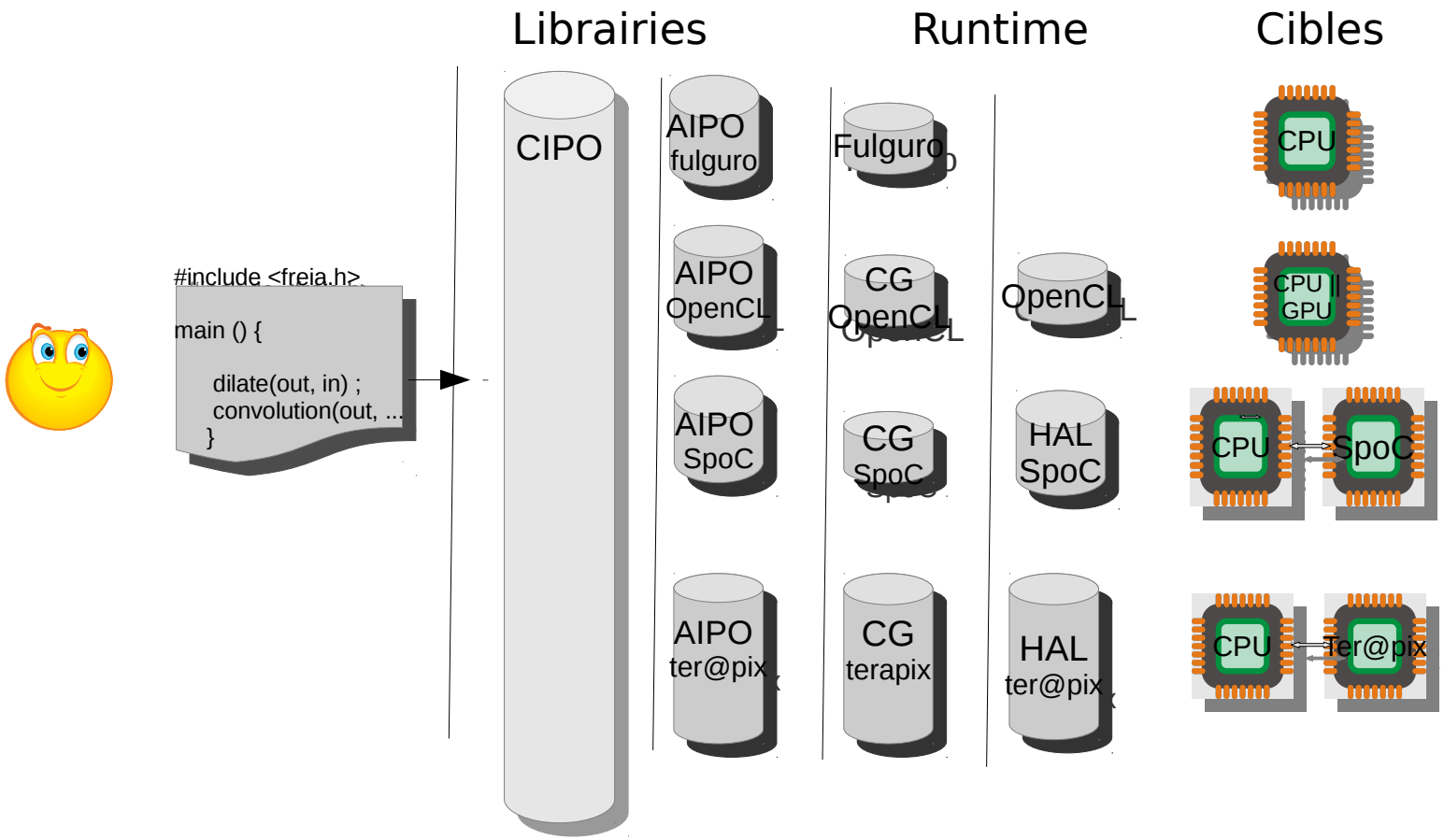
- Ordonnanceur simple :
  - La couche CG vers Fulguro est un simple proxy vers une librairie rapide de traitement d'images
  - SPoC étant un accélérateur pipeline travaillant sur des images, la distribution des données est simplifiée.
- Ordonnanceur complexe à cause de la gestion mémoire sur l'accélérateur :
  - Optimisation des transferts mémoires CPU–GPU pour l'OpenCL
  - Optimisation de la mémoire interne de *Ter@pix* qui ne peut contenir que des sous-images

WP1: Plateforme de simulations

- La cible Fulguro permet le développement rapide sur station de travail
- Deux simulateurs au niveau fonctionnels ont été réalisés
- Ces simulateurs permettent la validation de l'application sur les accélérateurs cibles avant optimisation par les outils

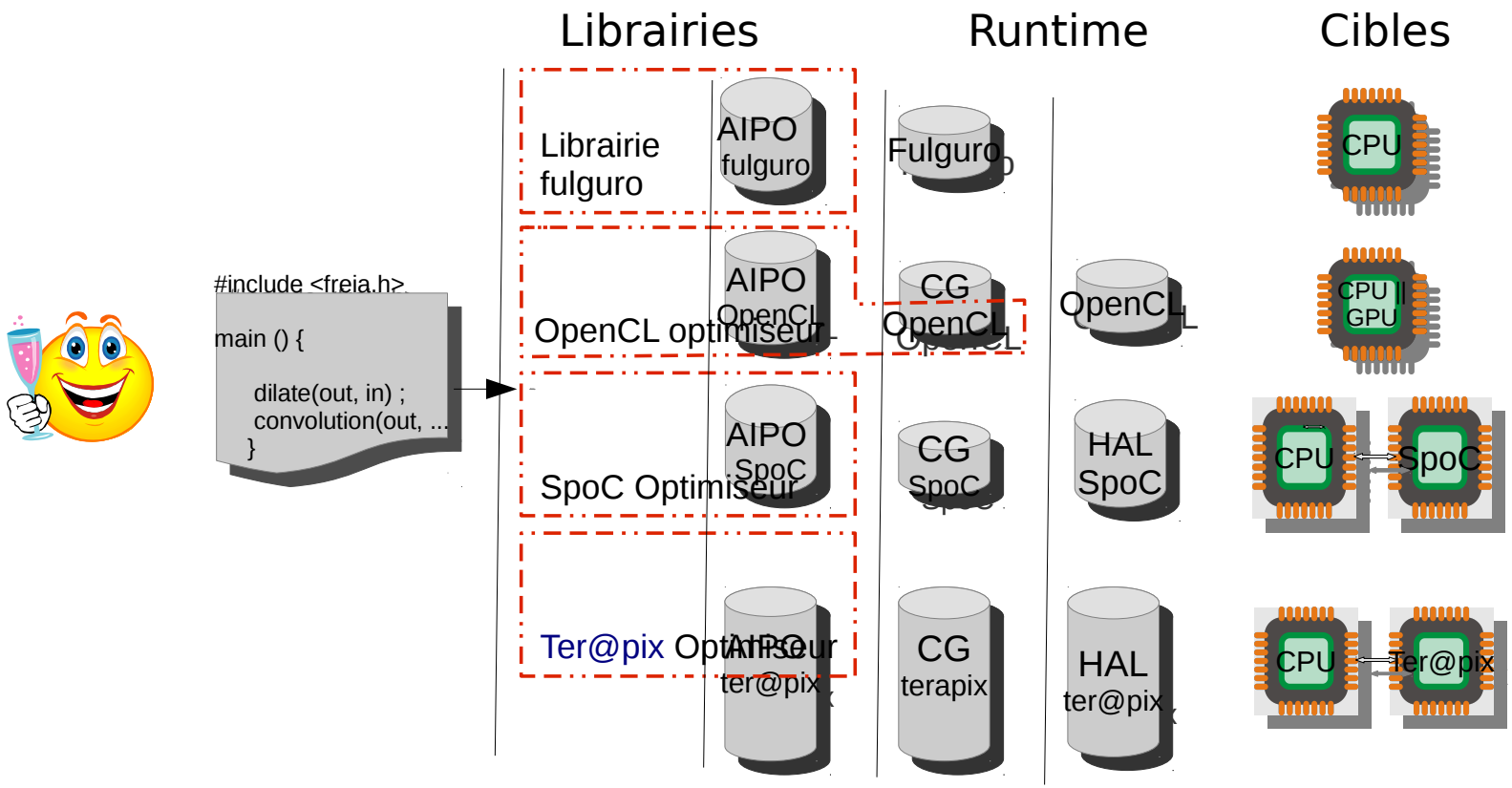


FREIA — outils d'optimisation à gros grain avec PIPS



Contributions: CRI

FREIA — outils d'optimisation à gros grain avec PIPS



Contributions: CRI

WP3.1/3.5/3.6 Compilateur gros grain

entrée code C code avec appels à API FREIA

- allocation, I/O, opérations images
- CIPO (complex) et AIPO (atomic)
- hypothèses: code sans bug, un seul type image

```
freia_data2d *im1 = freia_common_create_data(...);
freia_common_rx_image(im1, &in);
freia_aipo_global_min(im1, &min);
freia_cipo_dilate(im2, im1, 8, 10);
freia_aipo_subsat(im2, im1, im2);
freia_common_tx_image(im2, &out);
freia_common_destruct_data(im1);
```

sortie code C, configuration des accélérateurs
environnement d'exécution adapté à la cible

- *SPoC* : vecteurs d'instructions combinées
- *Ter@pix* : allocation d'images pour proc. SIMD
- OpenCL : aggrégation d'opérateurs simples

Phases de la compilation (1/3)

1. Préparation du source – PIPS

- matérialisation des séquences d'appels AIPO de l'arbre d'appel
- *inlining*, évaluation partielle, déroulage des boucles, élimination code mort, applatissement...
- élimination des images inutilisées

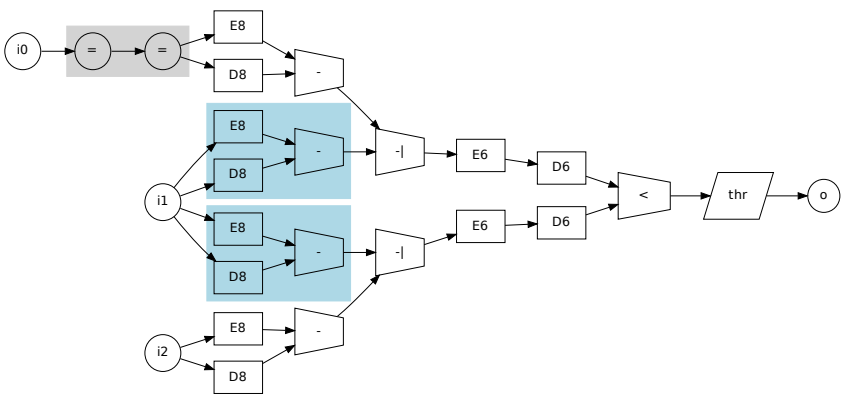
```
freia_aipo_global_min(im1, &min);
freia_aipo_dilate_8c(im2, im1, freia_morpho_kernel_8c);
freia_aipo_dilate_8c(im2, im2, freia_morpho_kernel_8c);
freia_aipo_dilate_8c(im2, im2, freia_morpho_kernel_8c);
freia_aipo_dilate_8c(im2, im2, freia_morpho_kernel_8c);
freia_aipo_dilate_8c(im2, im2, freia_morpho_kernel_8c);
freia_aipo_dilate_8c(im2, im2, freia_morpho_kernel_8c);
freia_aipo_dilate_8c(im2, im2, freia_morpho_kernel_8c);
freia_aipo_dilate_8c(im2, im2, freia_morpho_kernel_8c);
freia_aipo_dilate_8c(im2, im2, freia_morpho_kernel_8c);
freia_aipo_subsat(im2, im1, im2);
```

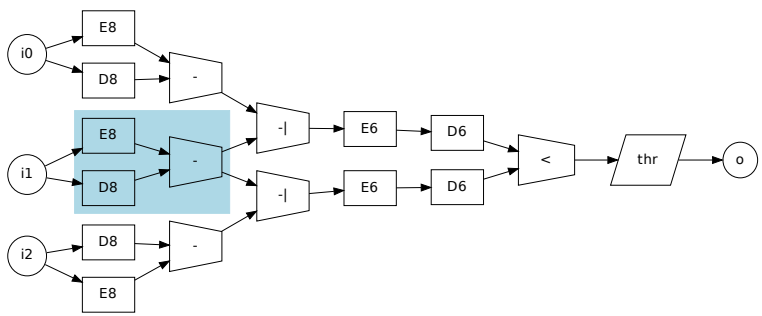
Phases de la compilation (2/3)

2. DAG d'expressions image

- construction du DAG des séquences d'appels AIPO
- normalisation, simplifications algébriques
- propagation de constantes, élimination expressions communes
- élimination des copies inutiles (avant et arrière)

DAG résultants pour application OOP *(Out of position)*





Phases de la compilation (3/3)

3. génération de code ou de configurations

Fulguro	librairie logicielle regénération des appels AIPO du DAG optimisé
SPoC	deux images vivantes max découpage du graphe plutôt horizontal
Ter@pix	calculs sur imagettes avec usure des bords découpage du graphe vertical, par niveau d'érosion
OpenCL	regroupement d'opérateurs simples (1-1) génération de noyaux de calculs optimisés

Performances obtenues

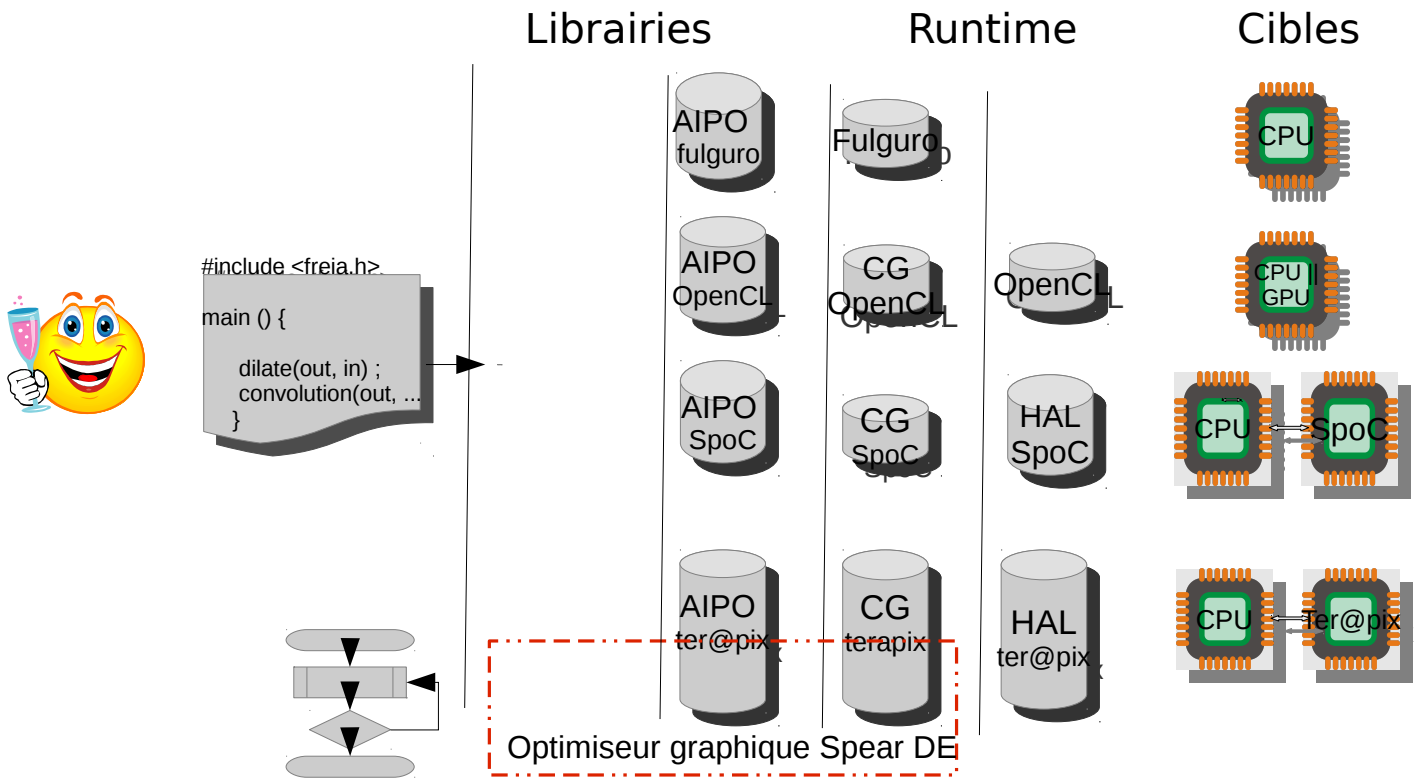
- applications FREIA (vidéo, médicales. . .)
- moyenne géom. des accélération vs librairie optimisée

SPoC **7.09** matériel bien adapté aux applications
  - code optimal sur la plupart des exemples

Ter@pix **2.92** borné par recouvrement calculs/comm.

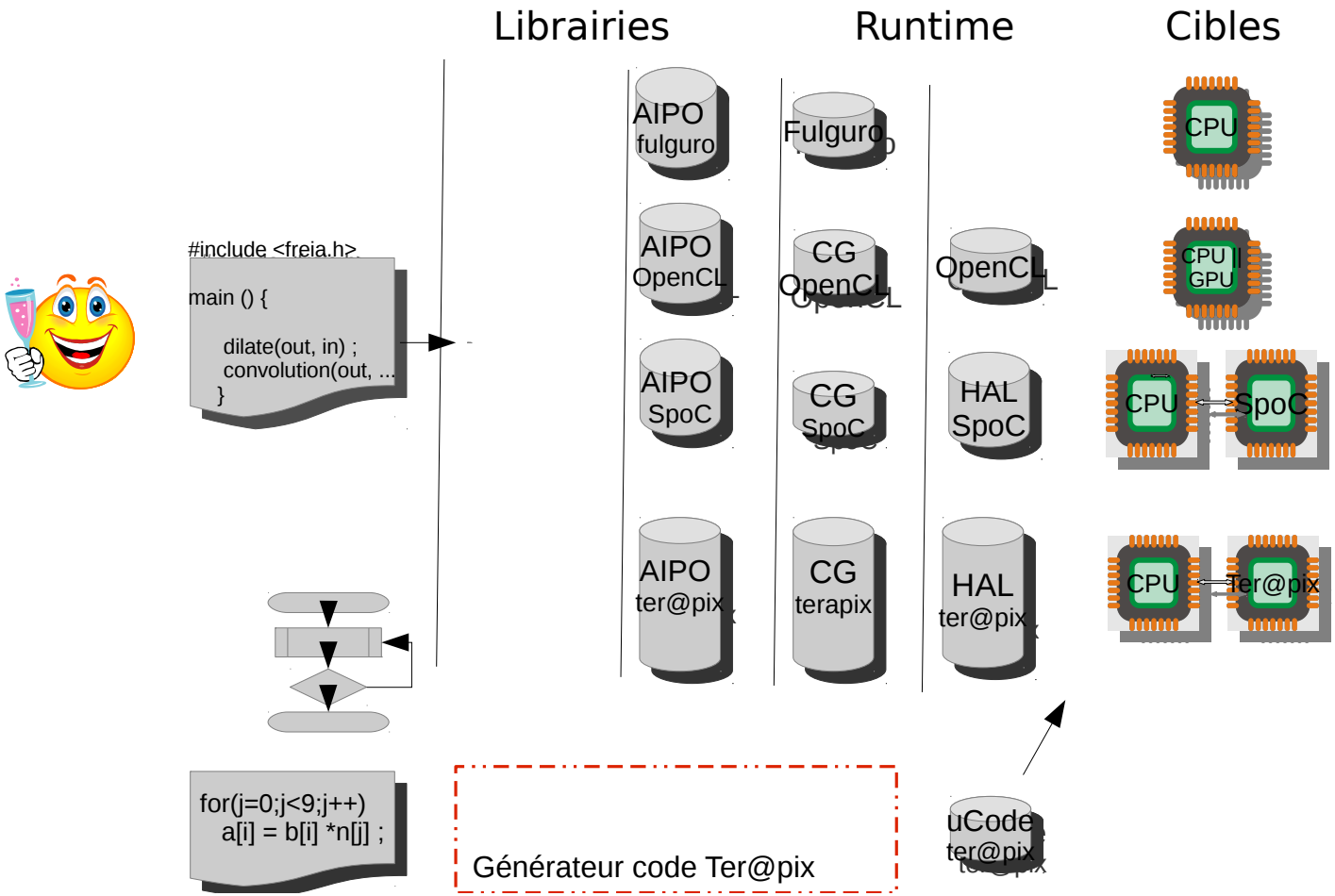
OpenCL **1.59** version librairie bien optimisée
  - pas de fusion des érosions/dilatations successives
  - l'exécutif minimise déjà les comm. hôte/carte

FREIA — outil d'optimisation SpearDE



Contributions: Thales

FREIA — outils d'optimisation grain fin avec PIPS



Contributions: TÉLECOM Bretagne, co-tutelle CRI

WP3.2/3.4: Génération de code pour le grain fin et moyen

Objectif

Générer automatiquement un microcode *Ter@pix* à partir de sa version C.

Méthodologie

1. Identifier les contraintes matérielles propre à la cible
2. Utiliser des passes de compilation originales pour lever ces contraintes
3. Assembler un compilateur qui chaîne ces transformations de façon automatique

WP3.2/3.4: Génération de code pour grain fin et moyen

Approche basée sur l'infrastructure de compilation PIPS

Compilation source-à-source

Pour interagir avec les outils existants chez Thales

Réutilisation de briques existantes

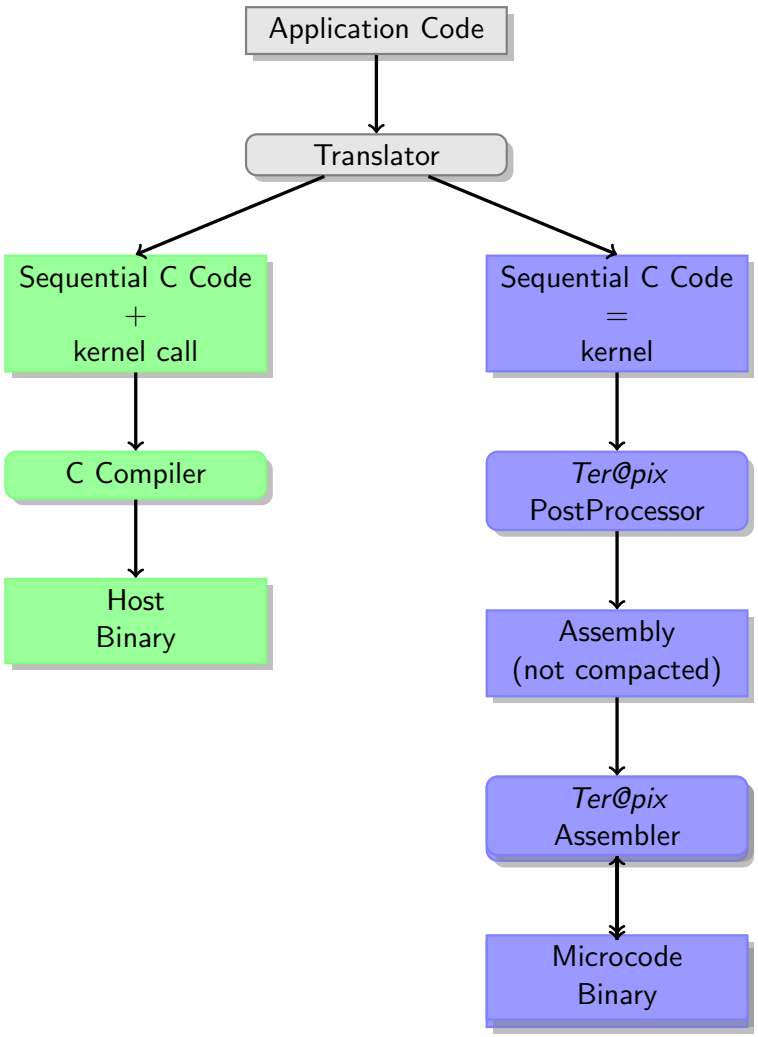
Pour la détection de boucles parallèles, la transformation de nids de boucles et les analyses interprocédurales

Contributions

- Transformations de code pour extraire une procédure d'un bloc [outlining], générer les communications [statement isolation], dimensionner un pavage de boucle [symbolic tiling] ...
- Gestionnaire de passe programmable pour exprimer des schémas de compilation complexes

WP3.2/3.4: Génération de code pour grain fin et moyen

Schéma de compilation



Involves

- 1+1 source-to-source compilers
- 1+1 source-to-binary compilers
- 1 code compactor

WP3.2/3.4: Génération de code pour grain fin et moyen

Mesure sur l'efficacité du code produit

Manuel *vs.* Automatique

Rapport entre le nombre de cycle de microcode en automatique et manuel.

	brightness	horizontal convolution	vertical erode	convolution
$\frac{\text{automatic}}{\text{manual}}$	$\times 1$	$\times 1.31$	$\times 2.12$	$\times 1.31$

$\rightsquigarrow$ Bonne performance pour un outil automatique

Ralentissement dû à:

- Sélection des instructions locales
- Utilisation de registres lents plutôt que des accumulateurs plus rapides

Dissémination

communication

- Fabrice Lemonnier. *Architecture distribuée programmable pour traitement d'images implémentée sur FPGA.* GDR ISIS C, juin 2008
- Fabien Coelho, *Compilation and Code Generation of Image Processing Applicaations for Hardware Accelerators* , Séminaire du LIP6, Paris, France, feb 2012
- Christophe Clienti, *Présentation du démonstrateur SPoC/FREIA*, Technoday Thales, 2011
- Christophe Clienti, *Présentation de l'outil SpearDE dans le contexte Terapix/FREIA*, Technoday Thales, 2010
- Fabien Coelho, François Irigoin, *Compiling for a Heterogeneous Vector Image Processor*, PIPS Developer Day, Lille 2010

Groupes de travail

- R. Keryell: GPU and multi-core GDR ASR and INRIA

Dissémination

Conférences

- Ronan Keryell. *Par4All: Auto-Parallelizing C and Fortran for the CUDA Architecture.* Nvidia GPU Technology Conference, Sep 30 2009
- Fabien Coelho, François Irigoin, *Compiling for a Heterogeneous Vector Image Processor*, Workshop on Optimizations for DSP and Embedded Systems (ODES'9), april 2011
- Serge Guelton, Sébastien Varrette. *Une approche génétique et source à source de l'optimisation de code.* conférence `RenPar'19/SympA'13/CFSE'6`, 9 septembre 2009
- Serge Guelton, Mehdi Amini, Ronan Keryell, and Beatrice Creusillet. *PyPS, a programmable pass manager.* poster for International Workshop on Languages and Compilers for Parallel Computing, September 2011. Fort Collins, Colorado, USA.

Dissémination

Conférences

- Serge Guelton, Adrien Guinet, and Ronan Keryell. *Building retargetable and efficient compilers for multimedia instruction sets.* poster for Parallel Architectures and Compilation Techniques, October 2011. Galveston, Texas, USA
- Serge Guelton, François Irigoin, and Ronan Keryell. *Automatic and source- to-source code generation for vector hardware accelerators.* Colloque National du GDR SOC-SIP, June 2010. Cergy, France.
- Serge Guelton. *A genetic and source-to-source approach to iterative compilation*. ACM Student Research Competition Posters, Parallel Architectures and Compilation Techniques, September 2009. Raleigh, North Carolina, USA.

Dissémination

Conférences

- Serge Guelton. *Automatic source-to-source code generation for vector hardware accelerators* . poster at International Workshop on Languages and Compilers for Parallel Computing, October 2010. Houston, Texas, USA.
- J. Bartovsky, E. Dokladalova, P. Dokladal, and V. Georgiev. *Pipeline architecture for compound morphological operators* IEEE International Conference on Image Processing (ICIP), pages 3765 –3768, Sept. 2010.
- Clienti C., Beucher S., Bilodeau M. *A System on Chip Dedicated to Pipeline Neighborhood Processing For Mathematical Morphology.* EUSIPCO 2008, Aug 25-29, Lausanne

Conclusion

- Chaîne complète d'outils libres d'optimisation pour le gros grain et moyen
  - Définition d'interfaces logicielles et de leur implémentation
  - Bibliothèque d'exécution
  - Simulateurs multi-niveaux
    - Fonctionnel (haut niveau, effet de bord source à source)
    - Comportemental (bas niveau)
  - Génération de code avec de très bonnes performances pour plusieurs cibles.
- Interface matérielle standardisée testée sur un nouvel IP
- Base outil plus générique Par4All
  - Création entreprise HPC Project (30+ personnes, recrute...)
    - Transfert équipe de TÉLÉCOM Bretagne dont thésard du projet
  - Utilisé dans autres projets pour autres cibles (GPU, CEA SCMP, ST P2012, Kalray MPPA...)

Liens

- http://freia.enstb.org
- http://pips4u.org