DE LA RECHERCHE À L'INDUSTRIE

cea

www.cea.fr

EXPLOITATION DE MOTIFS D'ACCÈS MÉMOIRE ET AMÉLIORATION DE LA COOPÉRATION DES CACHES DANS LES ARCHITECTURES MANY-COEURS

Journée Logiciels Embarqués et Architectures Matérielles du GDR SoC-SiP

Loïc Cudennec et Safae Dahmani

15 NOVEMBRE 2012

list

ACTIVITÉS HPC ET EMBARQUÉ

Laboratoire des fondements des systèmes temps-réel embarqués (LaSTRE)

- Systèmes temps-réels embarqués et sûreté de fonctionnement
  - OASIS : méthode de conception de systèmes temps-réels critiques, industrialisée dans le nucléaire (AREVA)
  - PharOS : application d'OASIS à l'automobile (Krono-Safe)

- Calcul haute performance
  - Recherche opérationnelle
  - Cloud computing, crypto-calcul
  - Manycoeurs et architectures massivement parallèles
    - Langages, méthodes de conception et de vérification
    - Modèle de programmation flot de données (Tau-C, Sigma-C / KALRAY)

      réduction parallélisme, dimensionnement, cadencement, placement, routage, génération de runtime, logiciel système
    - Modèle de programmation mémoire partagée

      NUMA, mémoire virtuellement partagée, cohérence des caches

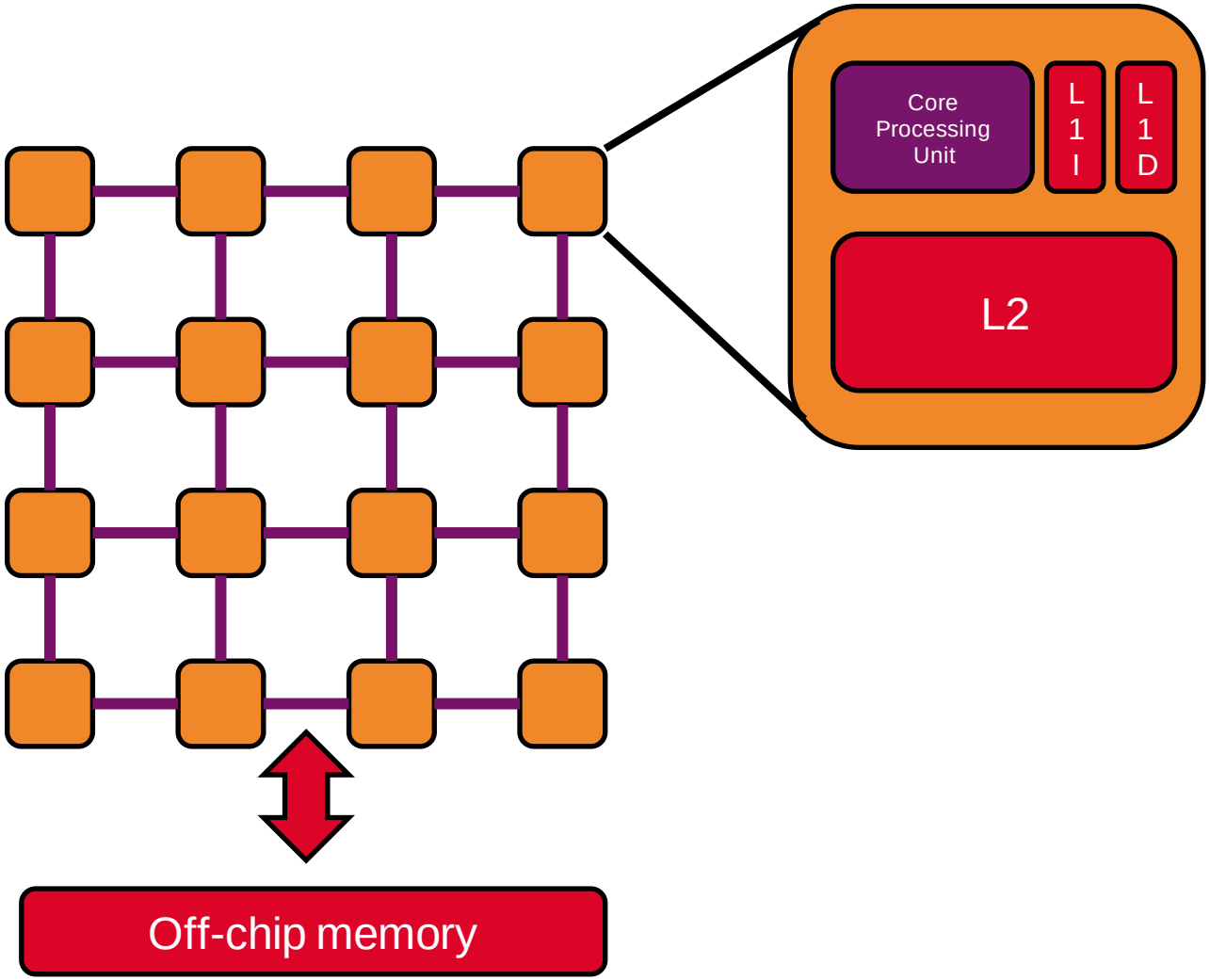
Architecture, Languages, Compilation and Hardware support for Emerging ManYcore systems (ALCHEMY Workshop)

- New challenges for efficient programmability of manycores
  - Advanced **compilers** for programming **languages** targeting massively parallel architectures
  - Advanced **architecture** support for massive **parallelism** management
  - Advanced architecture support for enhanced **communication** for CMP/manycores
  - New **OS**, or dedicated OS for massively parallel application
  - **Runtime** generation for parallel programming on manycores

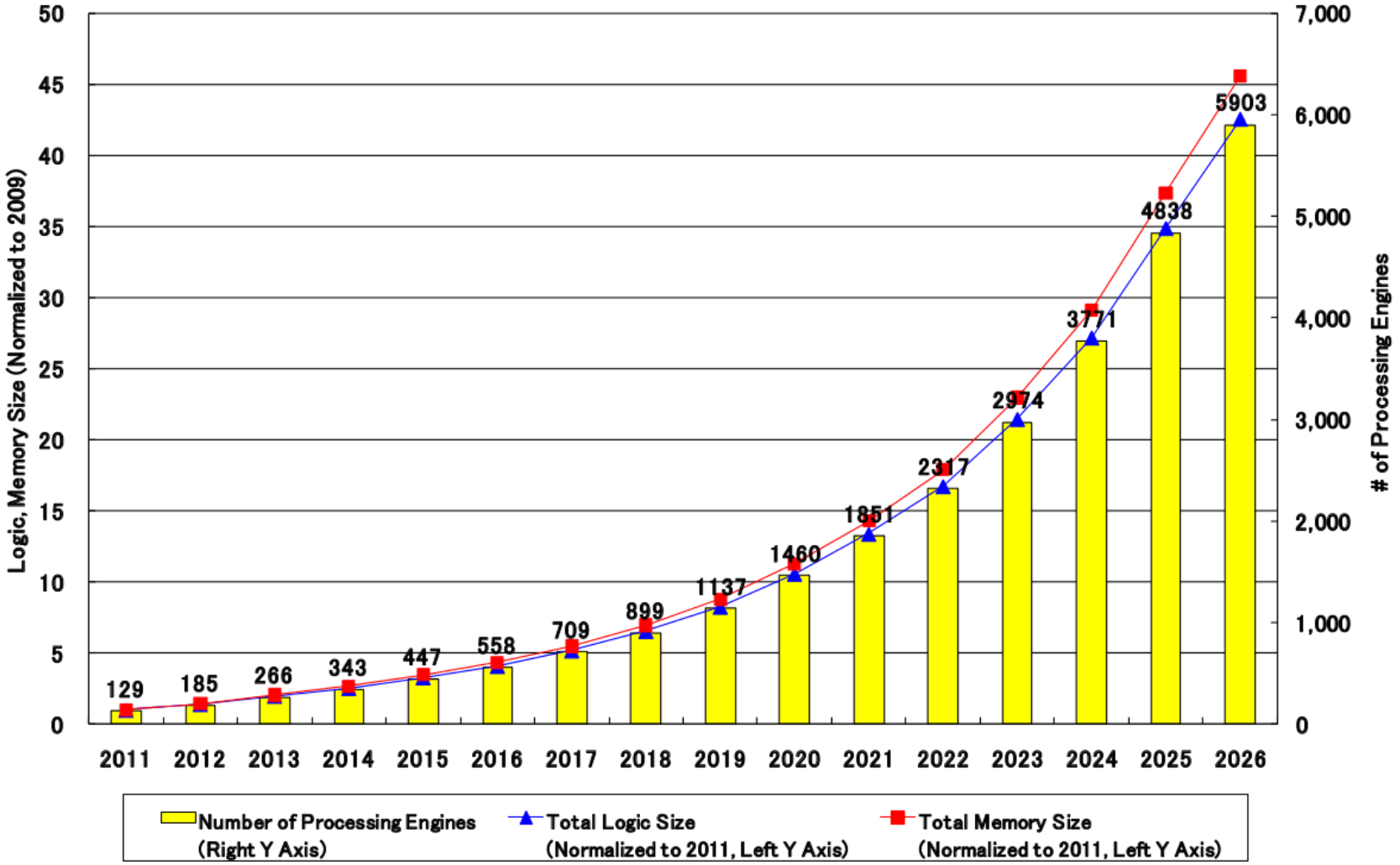
- Full paper submission (10 pages, Procedia Elsevier) **December 15, 2012**
- Program Committee

David Bader, Georgia Institute of Technology, USA Loïc Cudennec, CEA LIST, France Roberto Di Cosmo, University of Paris-Diderot, France Stephan Diestelhorst, AMD Dresden, Germany Aleksandar Dragojevic, Microsoft Research Cambridge, UK Benoît Dupond de Dinechin, KALRAY, France Daniel Etiemble, University of Paris-Sud, France Stéphane Louise, CEA LIST, France Eric Petit, University of Versailles Saint Quentin-en-Yvelines, France Antoniu Pop, Ecole Normale Superieure (ENS) de Paris, France Erwan Raffin, CAPS entreprise, France Etienne Rivière, University of Neuchâtel, Switzerland Osamu Tatebe, University of Tsukuba, Japan

ARCHITECTURES MANYCOEURS



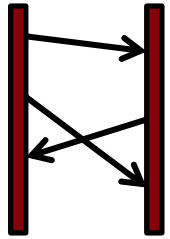
DU MULTI-CŒUR AU MANYCOEUR



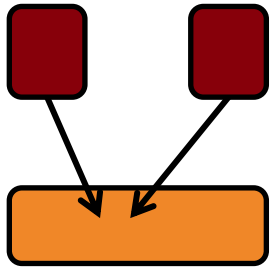
INTERNATIONAL TECHNOLOGY ROADMAP FOR SEMICONDUCTORS (ITRS) – System Driver Chapter 2011
SoC Consumer Portable Design Complexity Trends

MODÈLES DE PROGRAMMATION POUR MANYCOEURS

Les deux principaux paradigmes de programmation pour les architectures parallèles à large échelle

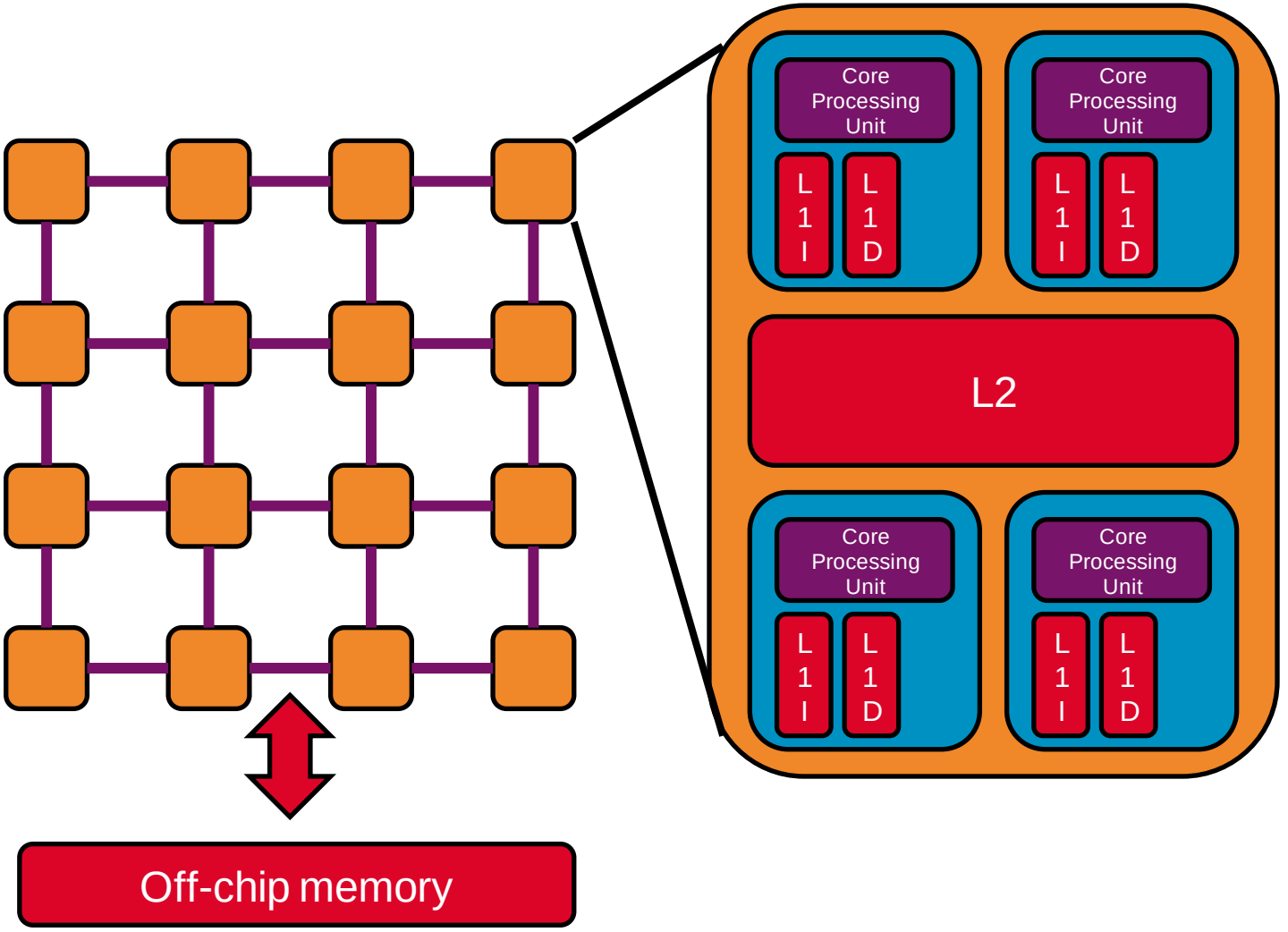


- Systèmes à passage de messages
  - Programmation répartie
  - Langages plus au moins structurés : MPI, JADE, Lustre, StreamIt, Sigma-C..
  - Pro : intergiciel léger pour gérer les communications, performances
  - Cons : travail de conception et de mise au point pour exprimer le parallélisme



- Systèmes à mémoire partagée
  - Localisation et transfert des données transparents pour l'application
  - Langages : C/POSIX, OpenMP, Cilk..
  - Pro : programmation intuitive, abstraction du matériel
  - Cons : coût des mécanismes matériels ou logiciels
  - Point dur : assurer le partage des données tout en passant à l'échelle

LOCALISATION DES DONNÉES DANS LES ARCHITECTURES MANYCOEURS HIÉRARCHIQUES



MODÈLES ET PROTOCOLES DE COHÉRENCE DES CACHES

Pour les architectures manycoeurs, la conception d'une mémoire virtuellement partagée repose sur les mécanismes de cohérence des caches

- Modèles de cohérence des données
  - Contraintes permettant d'encadrer la valeur d'une lecture en fonction des écritures précédentes
  - Cohérence forte : contraintes importantes entre une lecture et les écritures

    Ex: modèle strict, séquentiel, causal (Herlihy, Wing, Lamport, Raynal)
  - Cohérence relâchée : passage à l'échelle

    Ex : modèle faible, libération paresseuse, à l'entrée, de portée (Munin, Treadmarks, Midway)
  - Largement étudiés dans les multi-processeurs, les multi-coeurs, les grappes et grilles de calculateurs
- Problématique : proposer des modèles et protocoles de cohérence adaptés à une architecture SoC (NUMA) massivement parallèle
- Deux contributions pour la gestion des caches dans les manycoeurs
  - Prise en compte des motifs d'accès mémoire
  - Amélioration d'un système de cache élastique

PRISE EN COMPTE DES MOTIFS D'ACCÈS MÉMOIRE DANS LES ARCHITECTURES MANYCOEURS

(SoC 2012)

Jussara Marandola, Georgia Institute of Technology
Stéphane Louise, CEA LIST
Loïc Cudennec, CEA LIST
Jean-Thomas Acquaviva, CEA LIST
David Bader, Georgia Institute of Technology

DIMINUER LE COÛT D'UTILISATION D'UNE MÉMOIRE VIRTUELLEMENT PARTAGÉE

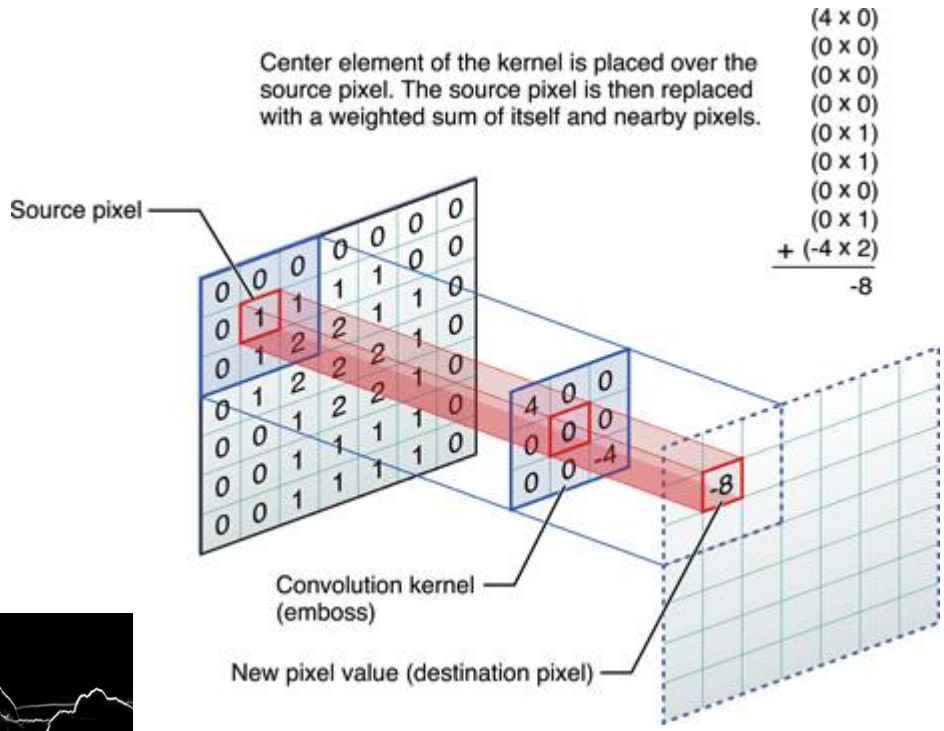
Augmenter les performances d'exécution des applications en diminuant les temps d'accès aux données

- Diminuer le nombre de messages coûteux du protocole de cohérence
  - Messages bloquants (programmation synchrone)
  - Messages de contrôle vs. de données (nombre de cycles réseau)
  - Chemins longs sur puce (nombre de hops), en mémoire externe (DMA)
  - Contention sur les liens ou les cœurs (importance du placement routage)

- Placer les données au plus proche des tâches
  - Mise à disposition des données proche de la demande (voisinage, coopération)
  - Migration, réplication
  - Anticipation des accès (*prefetching*)
  - Spéculation sur les prochains accès

Dans l'embarqué, un grand nombre d'applications utilisent des motifs d'accès mémoire

- Domaines d'application
  - Traitement d'image encodage, décodage audio et vidéo
  - Traitement du signal, analyse, réduction du bruit
  - Sécurité, chiffrement
- Algorithmes
  - Convolutions
  - Transformations usuelles (FFT, DCT..)
  - Opérations vectorielles



Kernel convolution.
Source : Apple Developer

EXEMPLE D'ACCÈS RÉGULIER : LE STRIDE

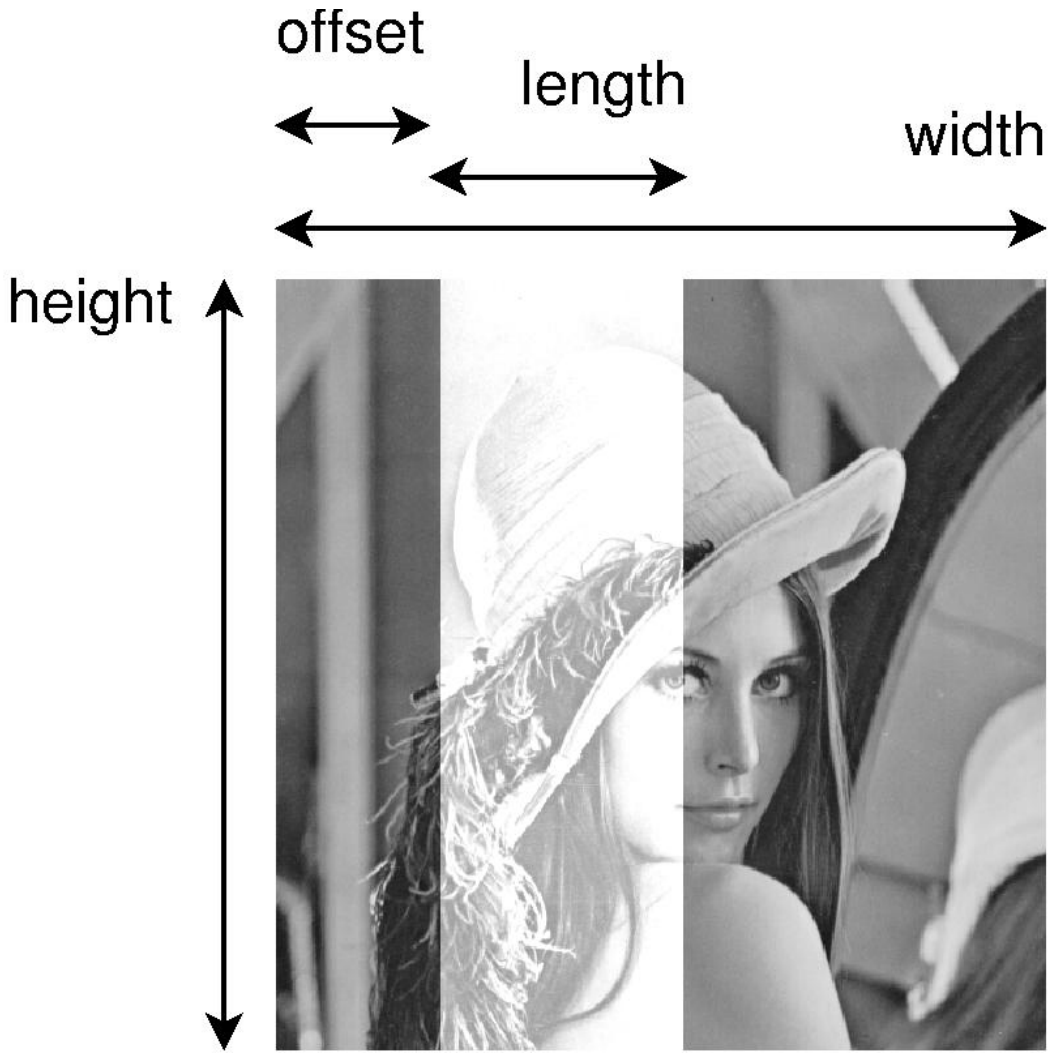


Image stockée sous forme de lignes contigües en mémoire

Accès *stridés* à 2 dimensions

Accès simples, réguliers et déterministes

1. Se déplacer à *@offset*
2. Accéder *length* pixels
3. Se déplacer de (*width-length*) pixels
4. Répéter 2,3 (*height-1*) fois

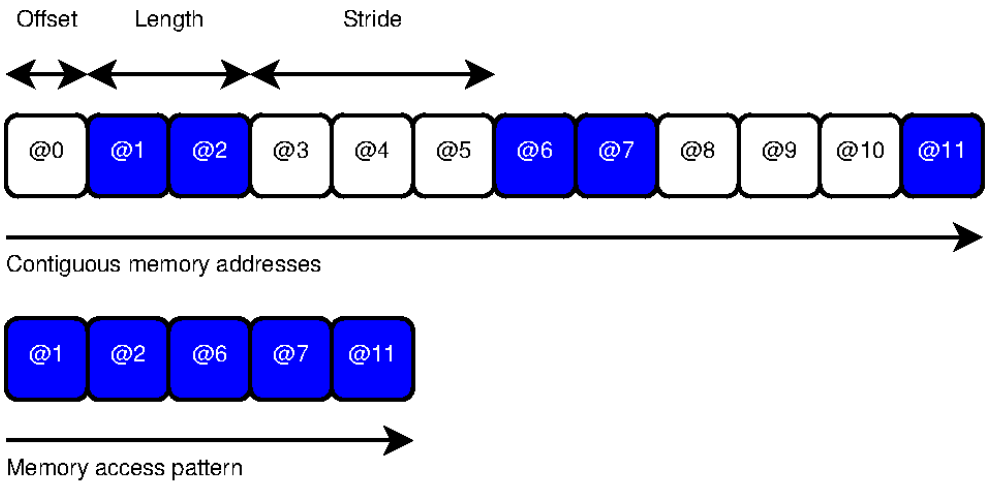
LE PROTOCOLE HOME-BASED MESI (BASELINE)

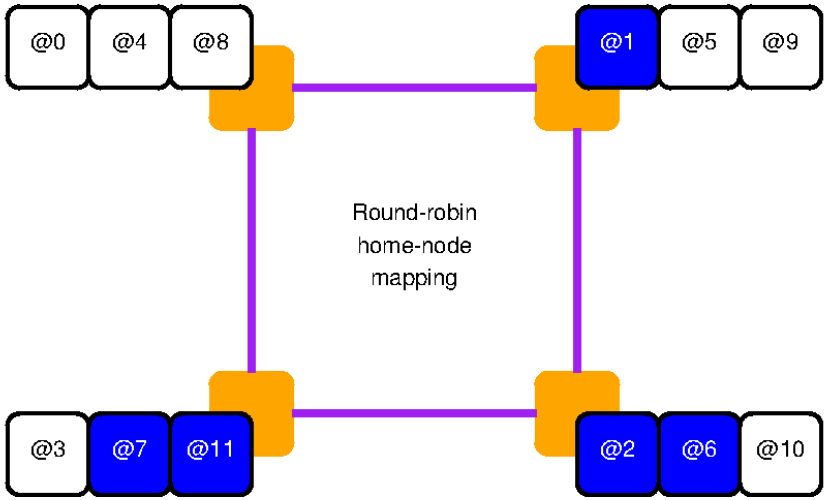
- Protocole MRSW à 4 états de cohérence
  - M : modifié
  - E : exclusif
  - S : partagé
  - I : invalide
- Nœuds gestionnaires (*home-nodes*)
  - Chaque adresse mémoire est gérée par un nœud gestionnaire
  - Les gestionnaires sont alloués en *round-robin* en appliquant un masque sur les adresses
  - Un masque processeur conserve la trace des copies sur les différents cœurs

Les protocoles dérivés de MSI, MOSI, MESI, MESIF, MOESI sont largement utilisés dans les architectures multi-coeurs.

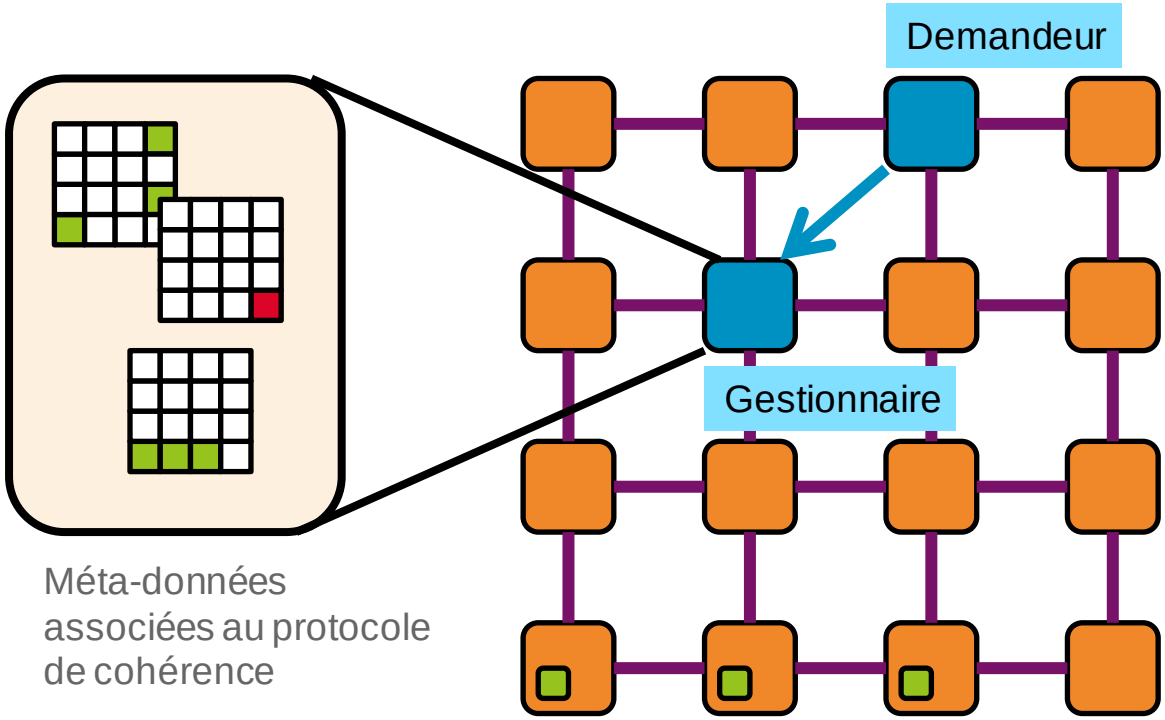
LE PROTOCOLE HOME-BASED MESI (BASELINE)

- Protocole MRSW à 4 états de cohérence
  - M : modifié
  - E : exclusif
  - S : partagé
  - I : invalide
- Nœuds gestionnaires
  - Chaque adresse mémoire est gérée par un nœud gestionnaire
  - Les gestionnaires sont alloués en *round-robin* en appliquant un masque sur les adresses
  - Un masque processeur conserve la trace des copies sur les différents cœurs



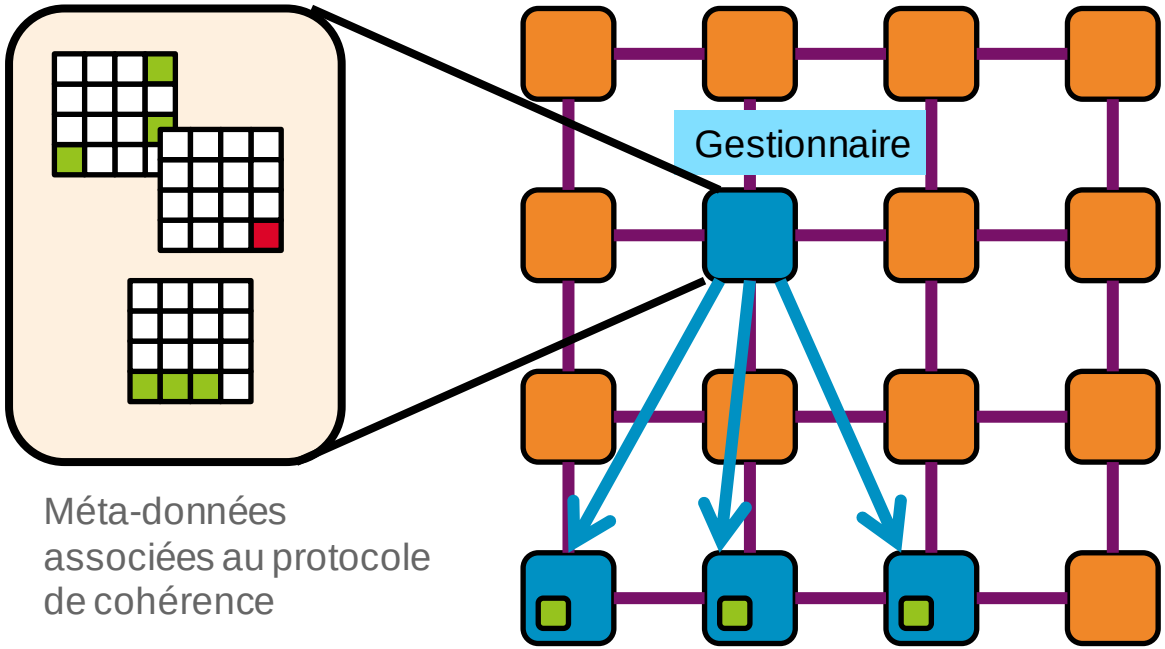


MESI : EXEMPLE D'ACCÈS EN ÉCRITURE



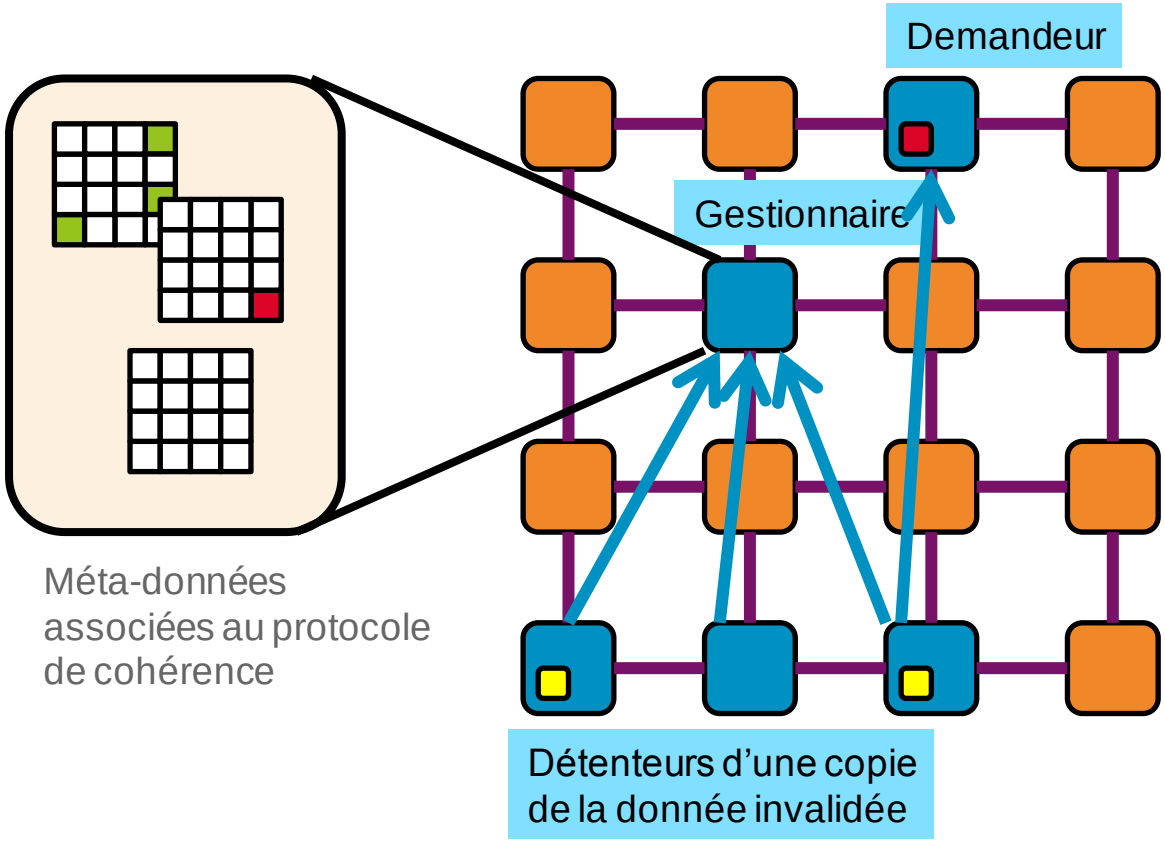
Requête en écriture

MESI : EXEMPLE D'ACCÈS EN ÉCRITURE



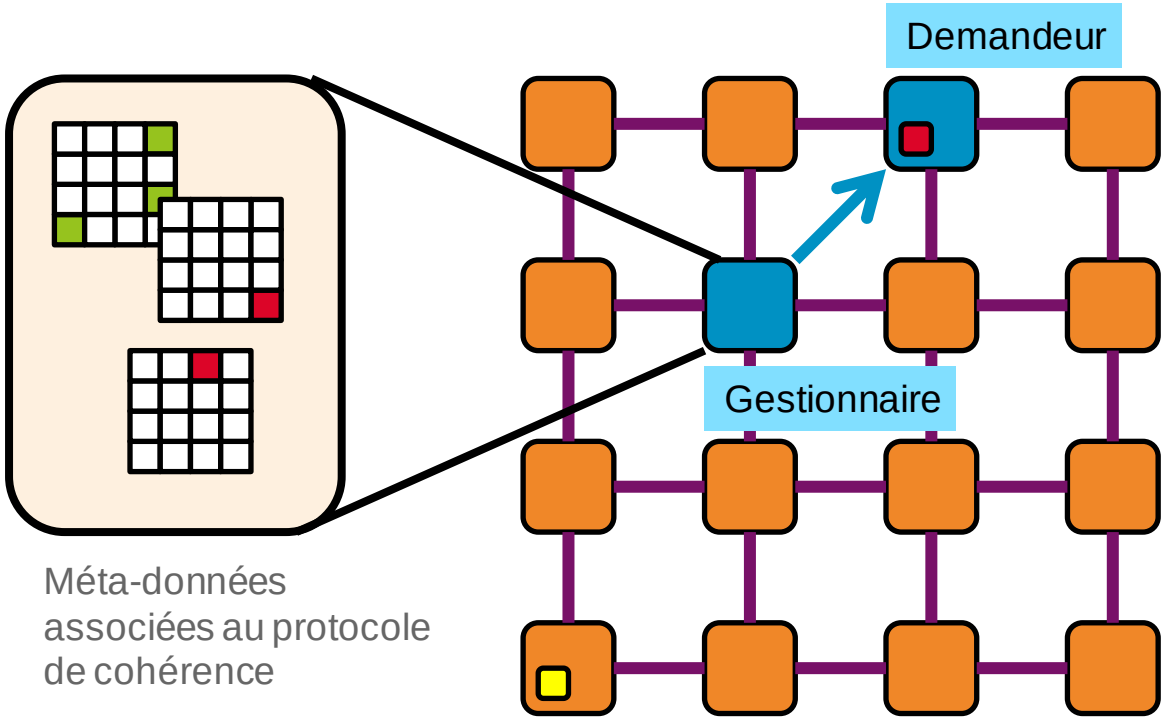
Demande d'invalidation et transfert

MESI : EXEMPLE D'ACCÈS EN ÉCRITURE



Acquittement des
invalidations et transfert

MESI : EXEMPLE D'ACCÈS EN ÉCRITURE



Acquittement de requête
d'écriture

PRENDRE EN COMPTE LES ACCÈS RÉGULIERS

Améliorer le comportement du protocole Baseline en présence d'accès réguliers à la mémoire

- MESI et le cas d'un accès réguliers à la mémoire
  - Chaque accès déclenche les 4 étapes du protocole (requête, invalidation, transfert et acquittement)
  - Les étapes sont séquentialisées par le protocole

- Quelques références sur les motifs d'accès mémoire
  - **Intel** Patent US 7,143,264 (2006)
    Instructions dédiées aux accès réguliers, limité à un cœur
  - **IBM** Patent US 7,395,407 (2008)
    Instructions utilisées pour la détection et l'exploitation de motifs d'accès, limité à un cœur
  - **Fujitsu** Patent US 5,829,017 (1998)
    Motifs d'accès mémoire sous forme d'une liste d'adresses mémoire déjà instanciée
  - **Sun Microsystems** Patent US 5,734,922 (1998)
    Motifs d'accès processeur sur une même adresse mémoire

PRENDRE EN COMPTE LES ACCÈS RÉGULIERS

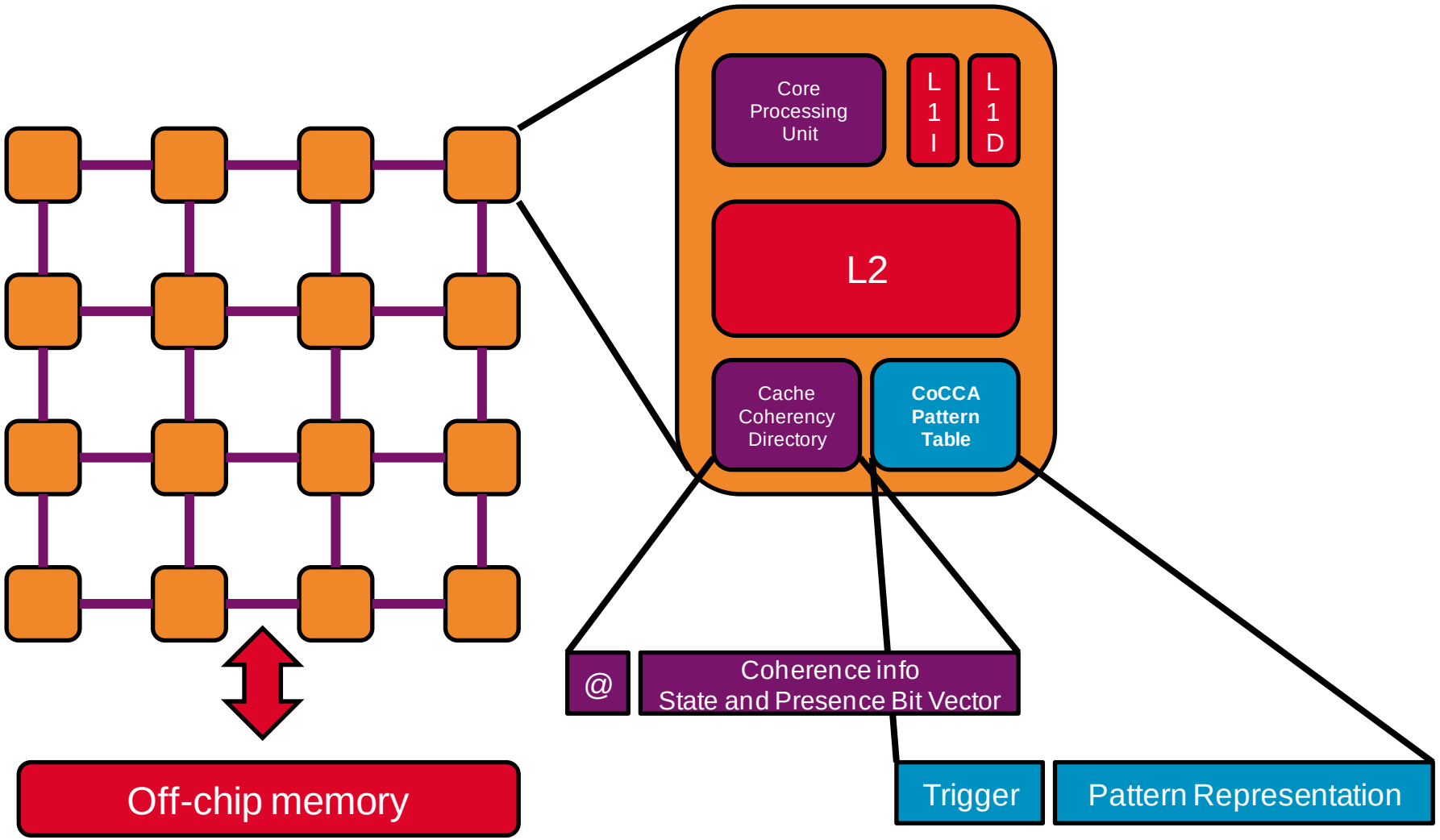
Améliorer le comportement du protocole Baseline en présence d'accès réguliers à la mémoire

- MESI et le cas d'un accès réguliers à la mémoire
  - Chaque accès déclenche les 4 étapes du protocole (requête, invalidation, transfert et acquittement)
  - Les étapes sont séquentialisées par le protocole

- Proposition : exploiter les motifs sur chaque cœur gestionnaire
  - Extension du protocole Baseline (pas de modification profonde du protocole)
  - Mise à jour du protocole avec des cas particuliers pour l'exploitation des motifs
  - Approche spéculative sur les adresses suivantes dans un motif
  - Extensions matérielles pour stocker et détecter les motifs sur chaque cœur
  - Construction statique des motifs à la compilation (ou dynamique par profilage)
  - Désignation d'un cœur gestionnaire pour chaque motif, choisi comme étant le cœur gestionnaire du premier accès

- Co-designed Cache Coherent Architecture (CoCCA)

COCCA CORE IP PATTERN TABLE



REPRÉSENTATION D'UN MOTIF COCCA

Les motifs sont représentés par une fonction mathématique paramétrée

- Une fonction paramétrée permet de calculer une suite d'adresses mémoire en l'appliquant à une adresse de base
  - Exemple de fonction : un stride 2D
    S2D(length,size,stride)(base)
  - Exemple de fonction paramétrée :
    un stride de 3 blocs de 2 adresses espacées de 4 adresses
    S2D(**3,2,4**)()
  - Exemple de fonction instanciée : application à l'adresse @7
    S2D(3,2,4)(@**7**) = **{@7, @8, @12, @13, @17, @18}**

REPRÉSENTATION D'UN MOTIF COCCA

Les motifs sont représentés par une fonction mathématique paramétrée

- Une fonction paramétrée permet de calculer une suite d'adresses mémoire en l'appliquant à une adresse de base
  - Exemple de fonction : un stride 2D
    S2D(length,size,stride)(base)
  - Exemple de fonction paramétrée :
    un stride de 3 blocs de 2 adresses espacées de 4 adresses
    S2D(**3,2,4**)()
  - Exemple de fonction instanciée : application à l'adresse @7
    S2D(3,2,4)(@**7**) = **{@7, @8, @12, @13, @17, @18}**

- Un déclencheur (**trigger**) est défini pour chaque motif afin de déterminer si l'accès à une adresse doit déclencher le transfert du motif complet
  - Exemples : une fonction de hachage, un masque sur les adresses, une suite d'adresses appartenant au motif..
  - Exemple simple : la première adresse du motif

REPRÉSENTATION D'UN MOTIF COCCA

Stockage des différents éléments d'un motif

S2D(3,2,4)(@7) = {@7, @8, @12, @13, @17, @18}

Il n'est pas raisonnable de stocker directement la forme instanciée du motif

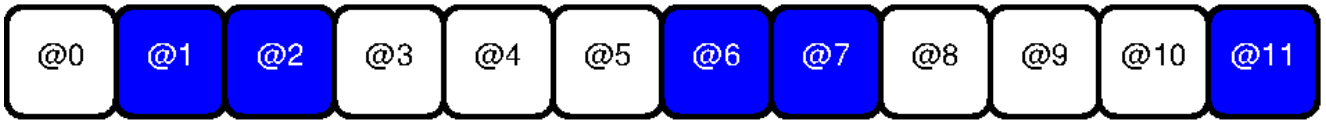
Bibliothèque de microcodes d'accès (type DMA) commune à tous les cœurs

Trigger	Représentation du motif : référence au microcode de la fonction et paramètres
@7	S2D 3 2 4

Table des motifs CoCCA implantée sur le cœur responsable de ce motif

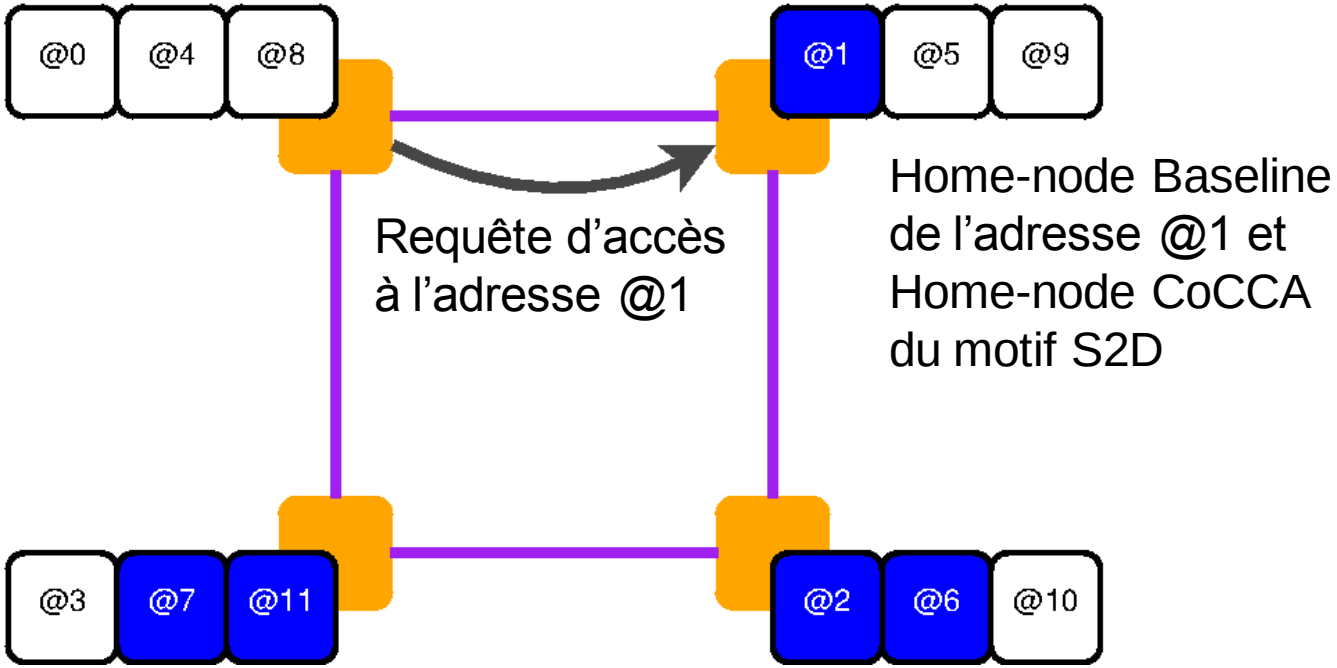
COMPORTEMENT DU PROTOCOLE DE COHÉRENCE BASELINE MODIFIÉ EN PRÉSENCE D'UN MOTIF

Mémoire contigüe

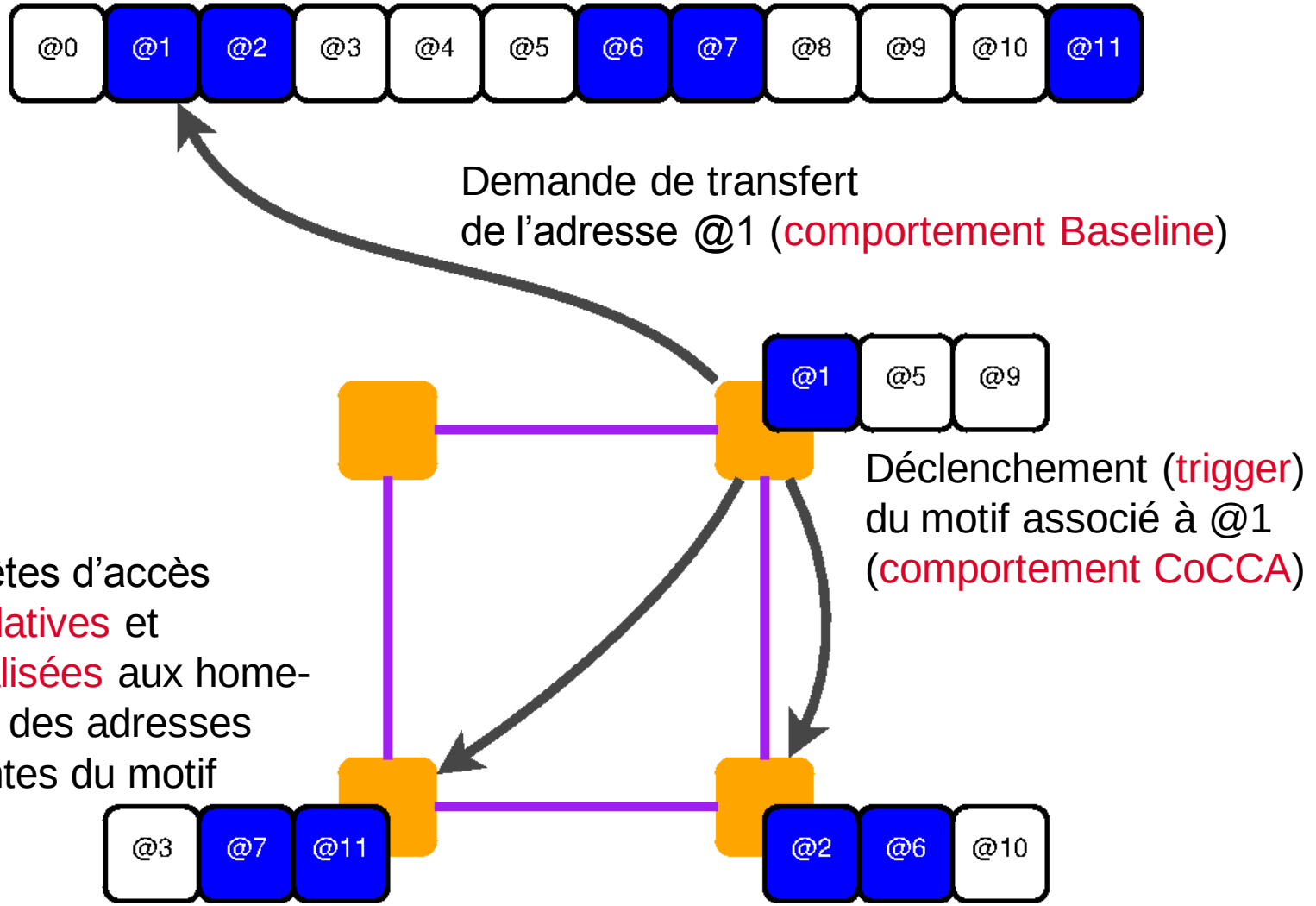


S2D(length=n, size=2, stride=3)(@1), Trigger={@1}

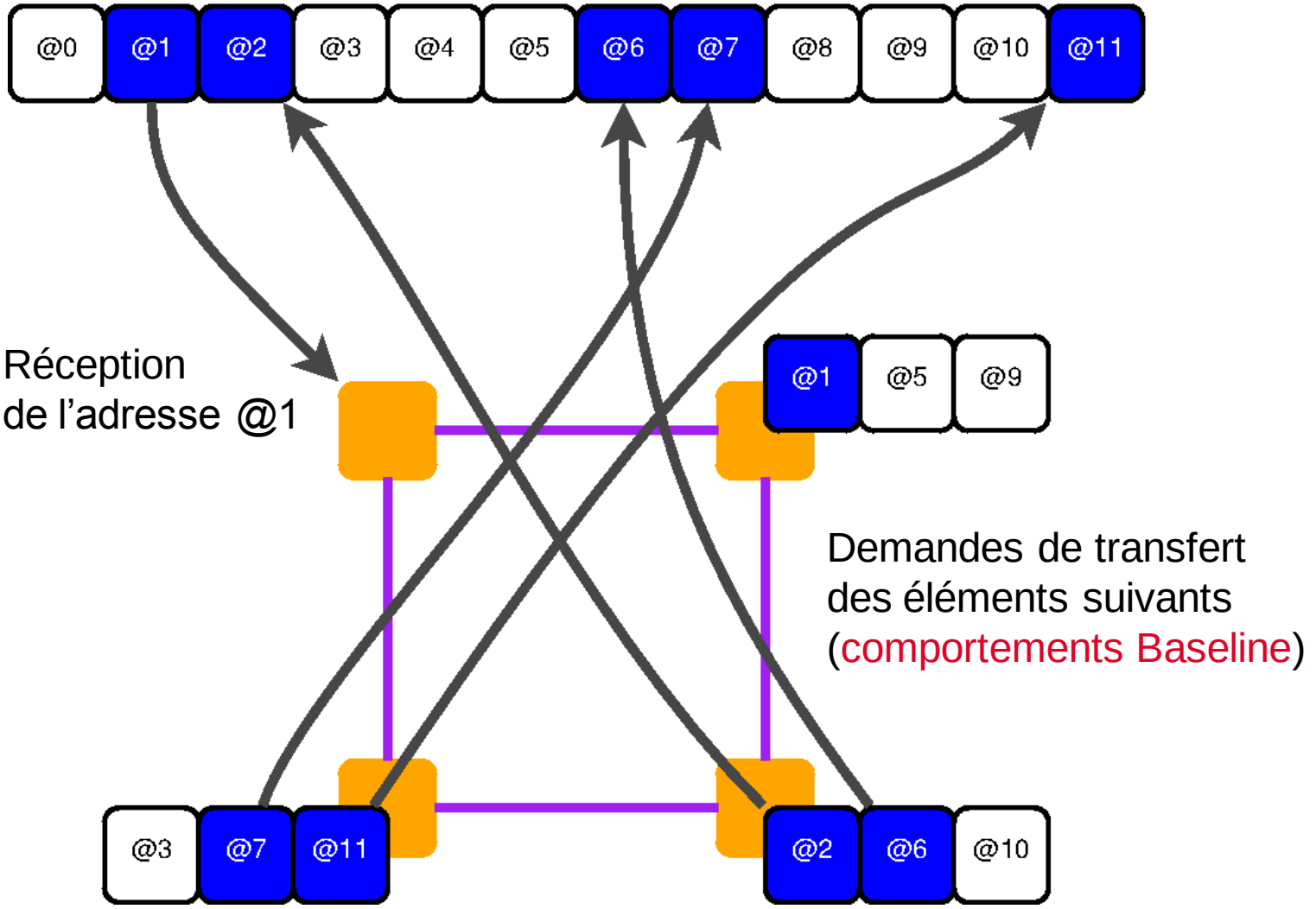
Assignation des home-nodes



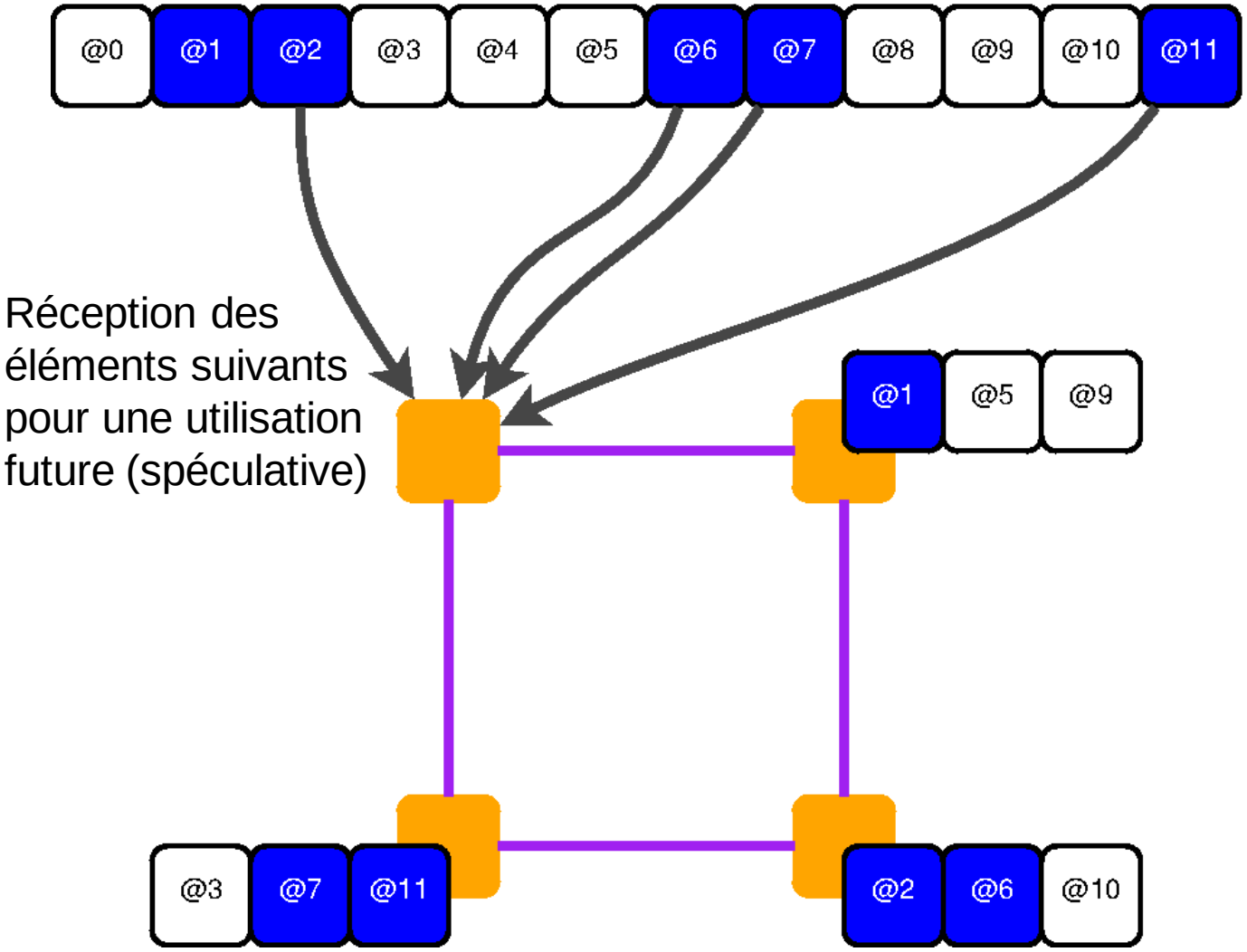
COMPORTEMENT DU PROTOCOLE DE COHÉRENCE BASELINE MODIFIÉ EN PRÉSENCE D'UN MOTIF



COMPORTEMENT DU PROTOCOLE DE COHÉRENCE BASELINE MODIFIÉ EN PRÉSENCE D'UN MOTIF

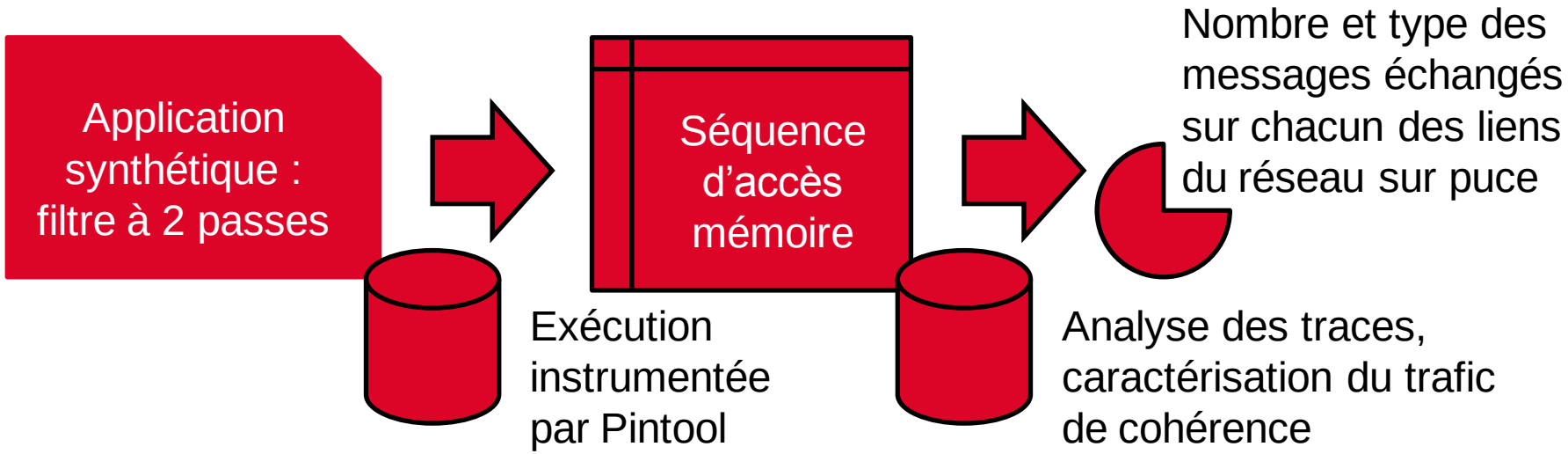


COMPORTEMENT DU PROTOCOLE DE COHÉRENCE BASELINE MODIFIÉ EN PRÉSENCE D'UN MOTIF



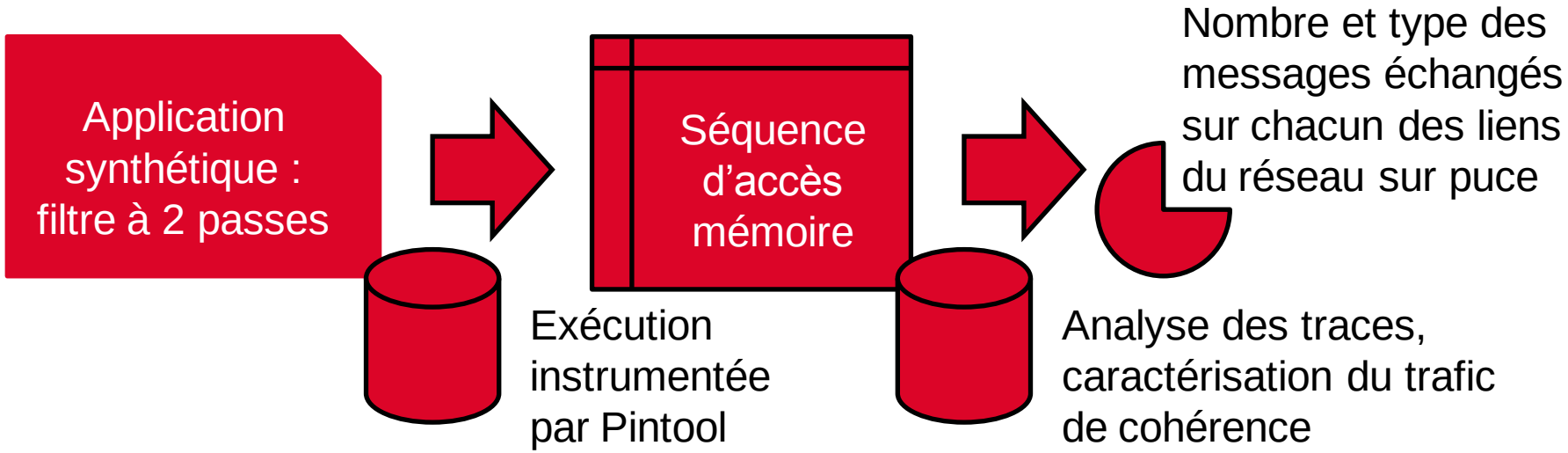
ÉVALUATION PRÉLIMINAIRE DU PROTOCOLE

Approche analytique avec étude statistique des messages induits par le protocole



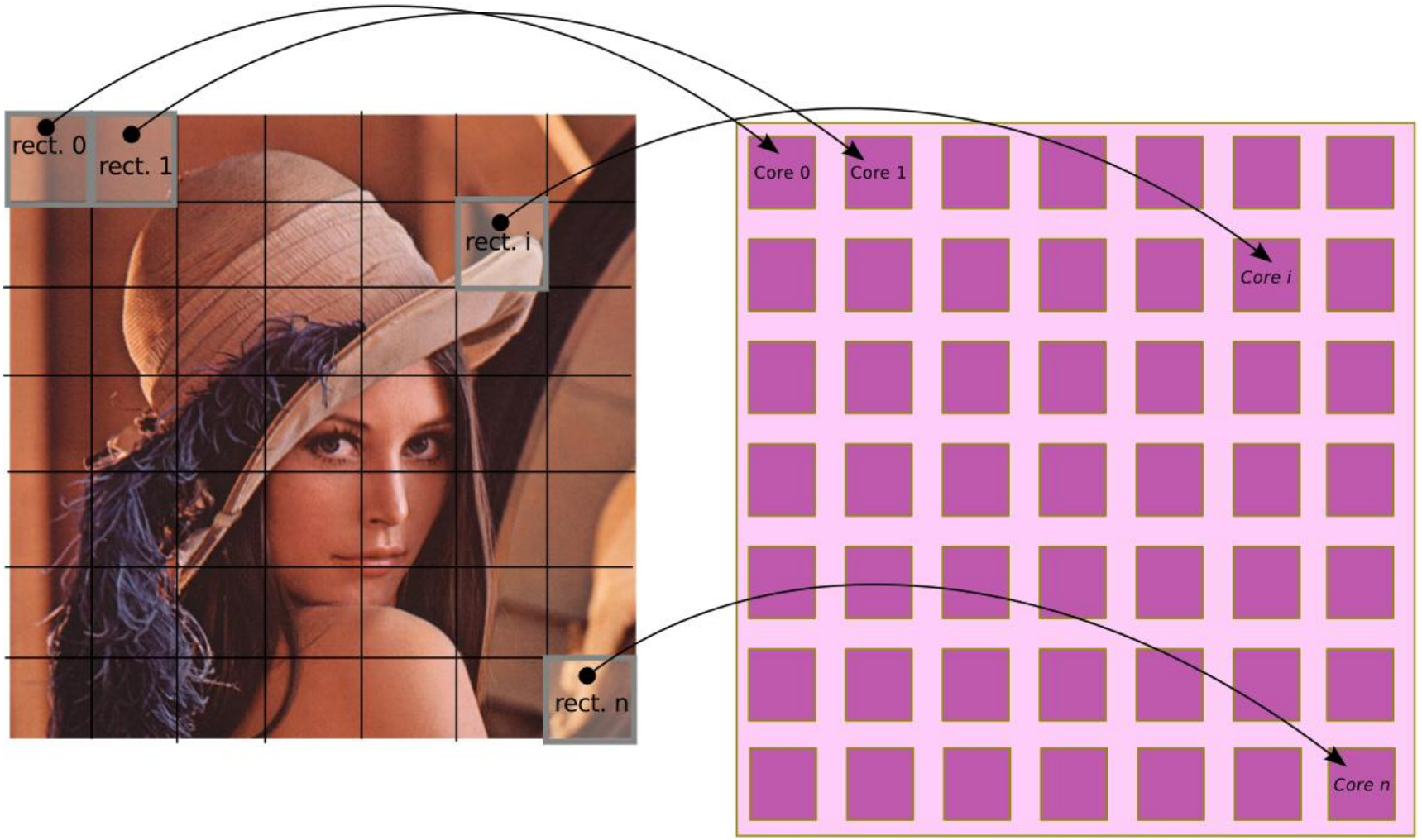
ÉVALUATION PRÉLIMINAIRE DU PROTOCOLE

Approche analytique avec étude statistique des messages induits par le protocole

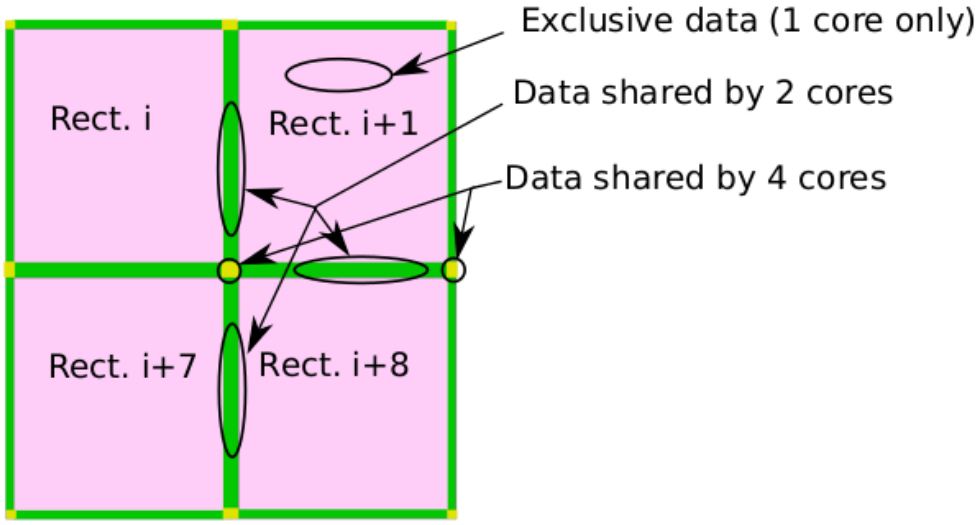


- Le résultat de la première passe du filtre est utilisé comme entrée de la seconde passe
- Noyau de convolution de taille 5x5 pixels
- Appliqués sur des macro-blocks rectangles de l'image (génère de nombreux partages de lignes de cache)
- Le résultat final est écrit sur l'image source (génère de nombreuses invalidations)

PROJECTION DE L'IMAGE SUR LES CŒURS DE LA PUCE

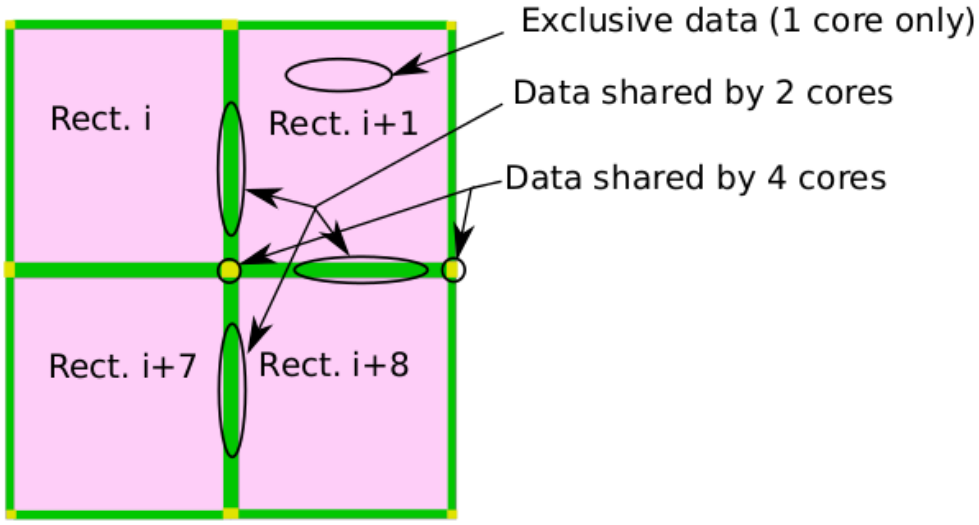


PARTAGE DE DONNÉES EN LECTURE ENTRE RECTANGLES VOISINS



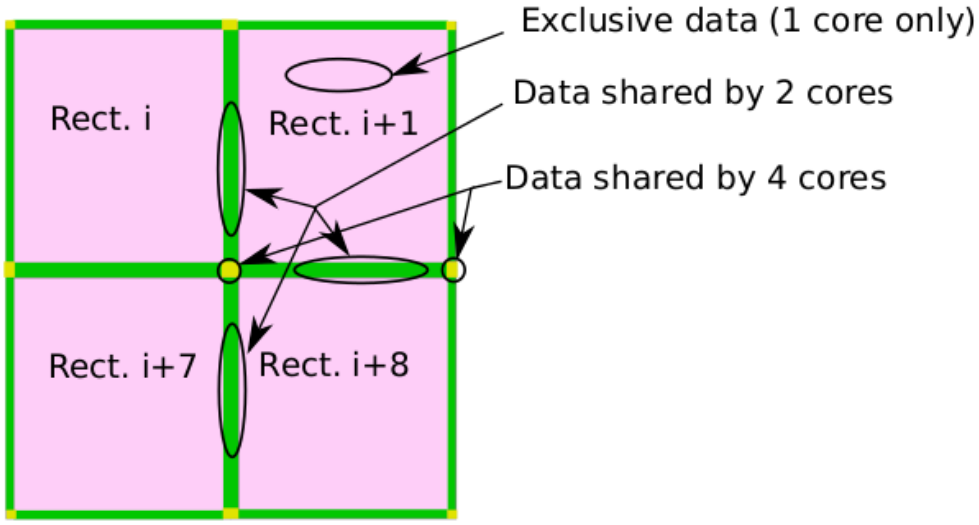
- Trois types de motifs (type 1)
  - Pré-chargement de l'image source dans les caches des cœurs
  - Modification de l'état partagé de l'image source vers l'état exclusif pour préparer l'écriture de l'image finale
  - Génère 2 motifs par cœur

PARTAGE DE DONNÉES EN LECTURE ENTRE RECTANGLES VOISINS



- Trois types de motifs (type 2)
  - Faux partage sur les accès en écriture entre 2 rectangles de l'image finale
  - Les frontières ne sont pas alignées avec les lignes de cache L2
  - Génère 2 motifs par cœur

PARTAGE DE DONNÉES EN LECTURE ENTRE RECTANGLES VOISINS



- Trois types de motifs (type 3)
  - Données partagées en lecture
  - Les noyaux de convolution lisent des données dans les rectangles voisins
  - Génère 2 motifs par cœur
- Total de **6 motifs par cœur**

RESULTATS ANALYTIQUES

Condition	MESI	CoCCA
Shared line invalidation	34560	17283
Exclusive line sharing (2 cores)	12768	12768
Exclusive line sharing (4 cores)	1344	772
Total throughput	48672	30723

- Diminution de 37% du trafic de cohérence
- Le pré-chargement correspond à 10% des accès aux caches
- Sans le pré-chargement, l'application s'exécute 67% plus lentement
  20 cycles pour les accès aux caches partagés
  80 cycles pour les accès à la mémoire externe

CONTRIBUTIONS

Proposer des mécanismes de cohérence des caches est primordial pour la programmabilité des manycoeurs

- Amélioration du comportement du protocole de cohérence des caches
  - Prendre en compte les comportements spécifiques des applications
  - Extension d'un protocole Baseline pour gérer les accès réguliers
  - Dans le contexte de l'embarqué, les motifs sont générés à la compilation
  - Conception mixte logicielle (protocole de cohérence) et matérielle (stockage et détection des motifs)

- Observations
  - Un petit nombre de motifs est suffisant pour prendre en charge les filtres bas-niveaux
  - Les motifs réduisent significativement le trafic du protocole de cohérence
  - Le préchargement des données accélère l'exécution des applications
  - CoCCA peut être une piste de recherche pertinente pour les manycoeurs

TRAVAUX FUTURS ET PERSPECTIVES

- Etendre les expérimentations à des applications ciblant les manycoeurs afin de tirer des conclusions plus générales
- Implémenter l'extension du protocole dans un simulateur de réseau sur puce « cycle accurate » afin d'évaluer les gains en temps d'exécution
- Utiliser un simulateur complet (i.e. SoCLib/Lip6)
- Etendre les travaux à une architecture HPC capable de construire les motifs dynamiquement

Commissariat à l'énergie atomique et aux énergies alternatives
Centre de Saclay | 91191 Gif-sur-Yvette Cedex
T. +33 (0)1 XX XX XX XX | F. +33 (0)1 XX XX XX XX

DRT
LIST
DACLE

Etablissement public à caractère industriel et commercial | RCS Paris B 775 685 019