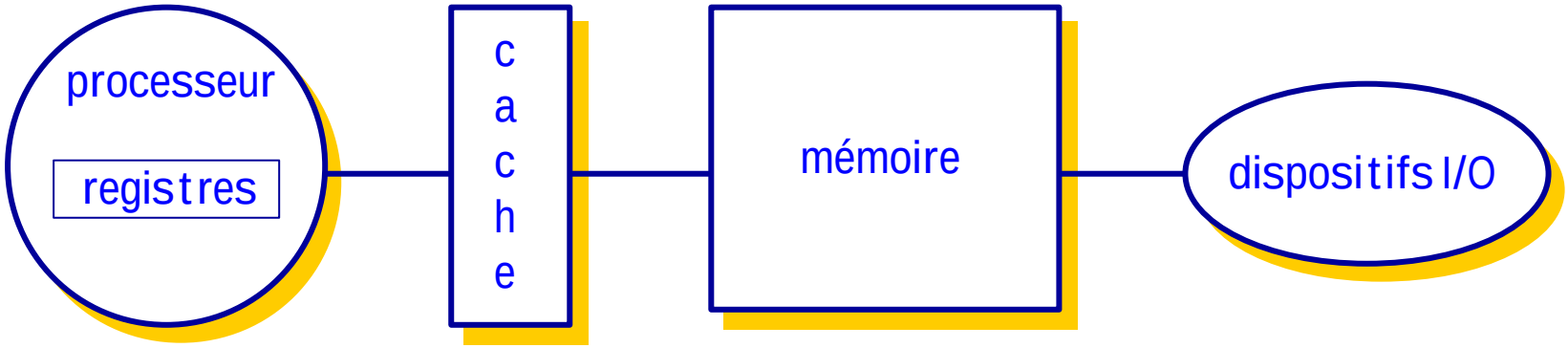
La mémoire cache

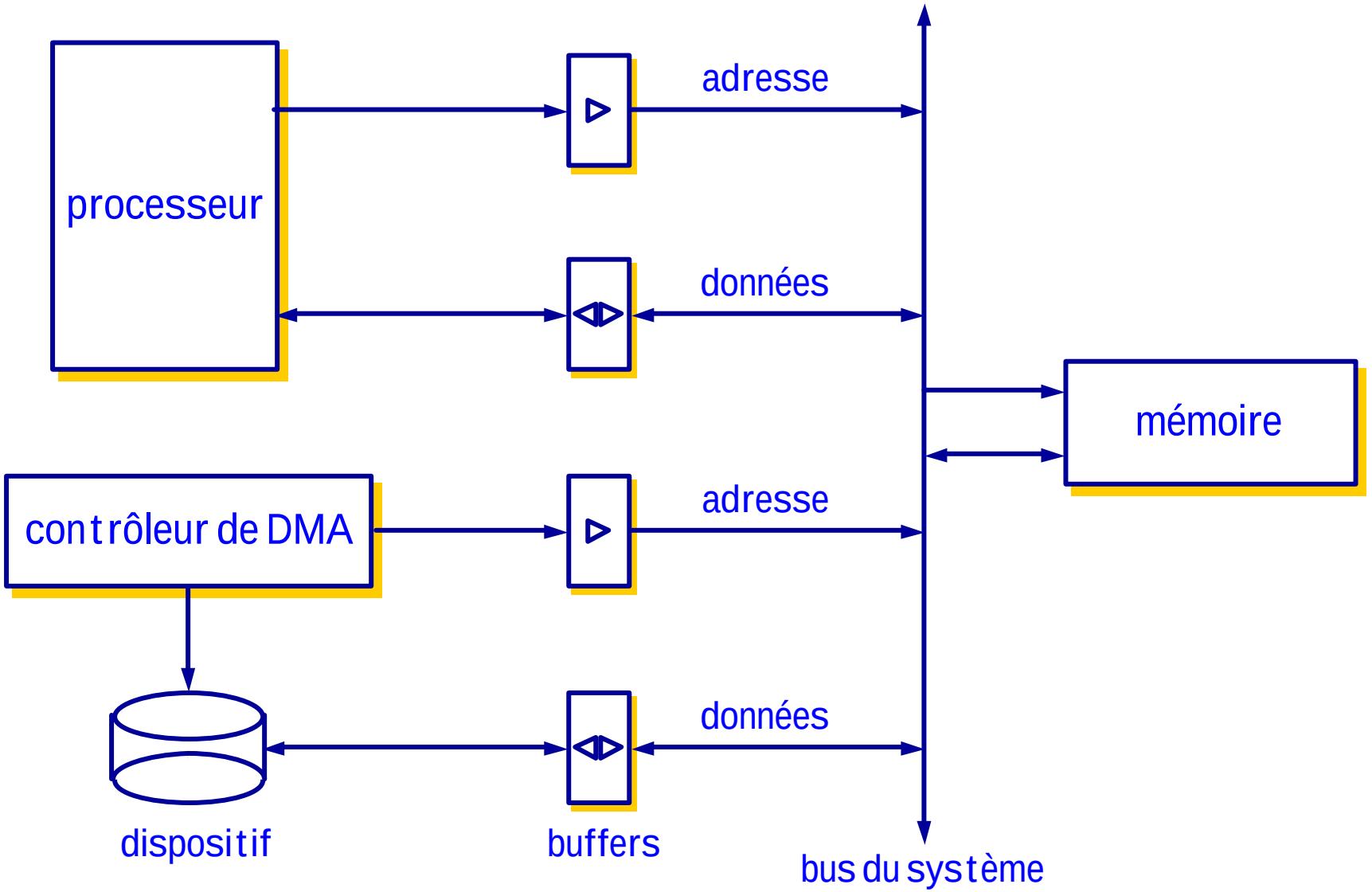
Eduardo Sanchez
Laboratoire de Systèmes Logiques
Ecole Polytechnique Fédérale de Lausanne

Organisation de la mémoire



taille:	200B	64KB	32MB	2GB
vitesse:	5ns	10ns	100ns	5ms
largeur de bande (MB/sec):	4000-32000	800-5000	400-2000	4-32

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne



Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Principe de localité (*locality*)

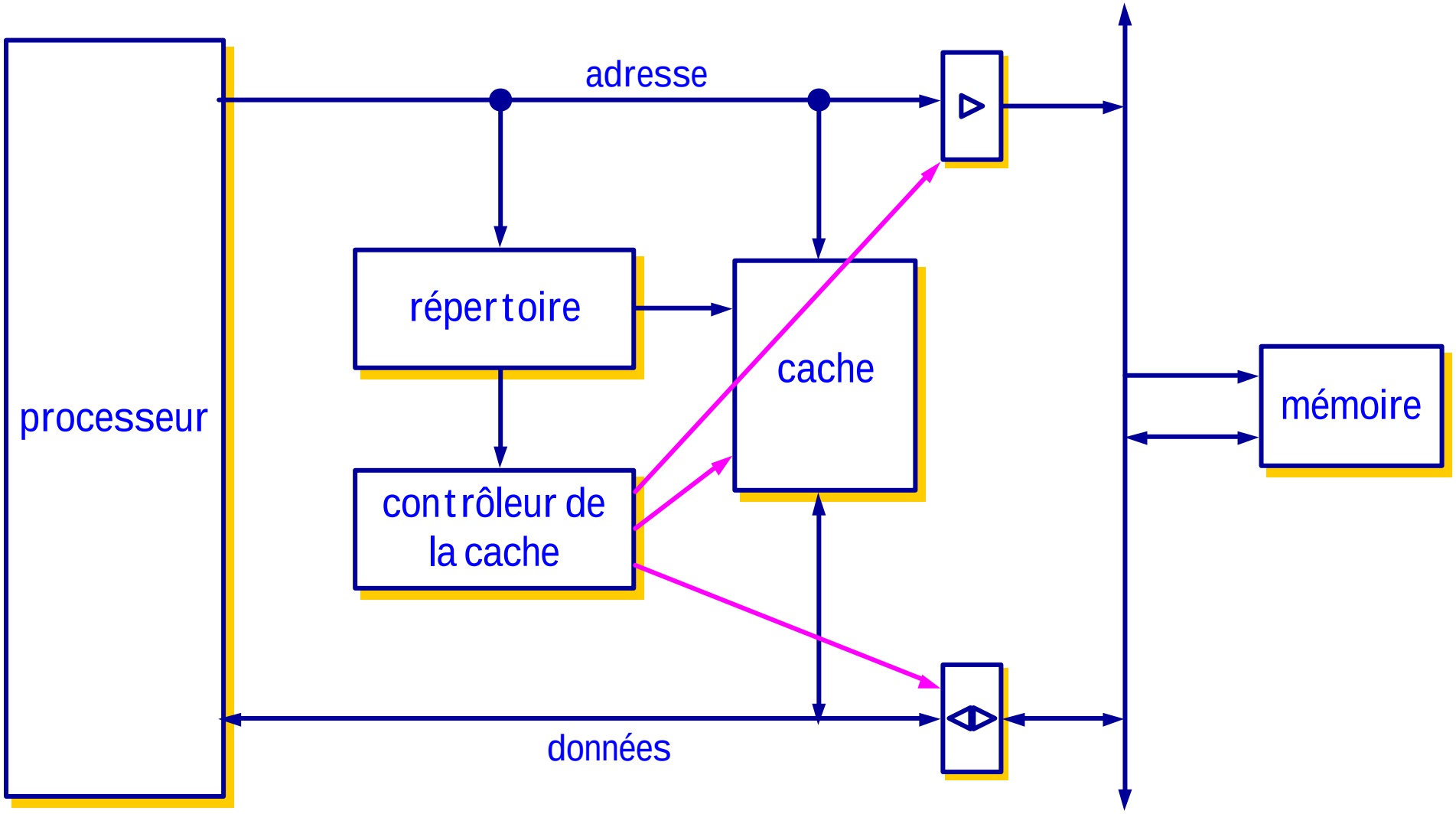
- Localité spatiale:
  le code d'un programme s'exécute toujours à l'intérieur de petites zones répétées de mémoire (des blocs correspondant à des boucles ou/et des sous-programmes)
- Localité temporelle:
  les blocs s'exécutent en séquences très proches (il y a plus de chances d'accéder à une position de mémoire utilisée il y a 10 cycles qu'à une autre utilisée il y a 10000 cycles)

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Principe de la mémoire cache

- Mémoire très rapide, mais de petite taille, placée entre le processeur et la mémoire principale
- Le processeur essaie d'accéder un mot d'abord dans la cache, avant de passer à la mémoire principale. En cas d'échec (*miss*), le mot est gardé dans la cache pour un accès futur. En cas de succès (*hit*), la mémoire principale n'est pas accédée
- La fréquence des succès (*hit rate*) dépend de la taille de la cache et de l'algorithme exécuté par le contrôleur de cache

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne



Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Importance de la fréquence de succès

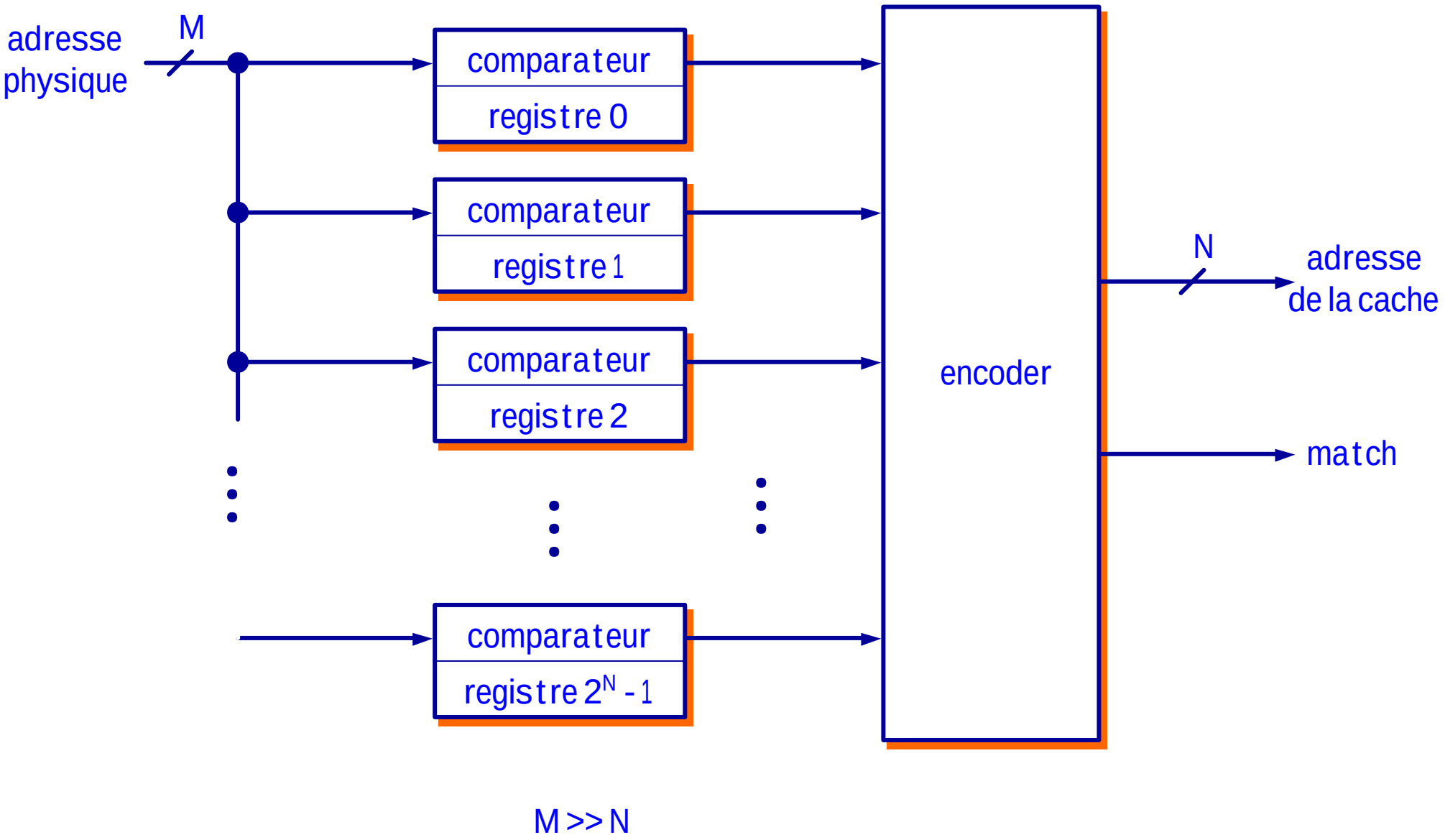
- Si les accès à la mémoire demandent 3 *wait states* et la fréquence de succès est de 90%, le nombre moyen de *waits states* est de
  10% x 3 = 0.3 *wait state* / cycle de mémoire
  la chute de la fréquence de succès à 80% fait doubler cette valeur
- Si une instruction demande 2 cycles sans *wait state* et 5 cycles avec, et si la fréquence de succès est de 80%, l'exécution de 10 instructions demande:
  (10 x 0.8 x 2) + (10 x (1 - 0.8) x 5) = 26 cycles
  dont 16 sont faits avec la cache.
  C'est-à-dire: le processeur passe 40% du temps (10/26) à chercher 20% du code: pendant ce temps le bus du système est occupé par les transferts entre le processeur et la mémoire

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Le répertoire de la cache

- Le répertoire de la cache a deux fonctions:
  - indiquer si le mot accédé par le processeur se trouve dans la cache (signal *match*)
  - si c'est le cas, indiquer l'adresse de la cache où se trouve le mot
- Le répertoire de la cache peut donc être vu comme une mémoire associative: on parle alors d'une mémoire cache complètement associative
- Dans ce cas, un mot de la mémoire principale peut être stocké n'importe où dans la cache

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne



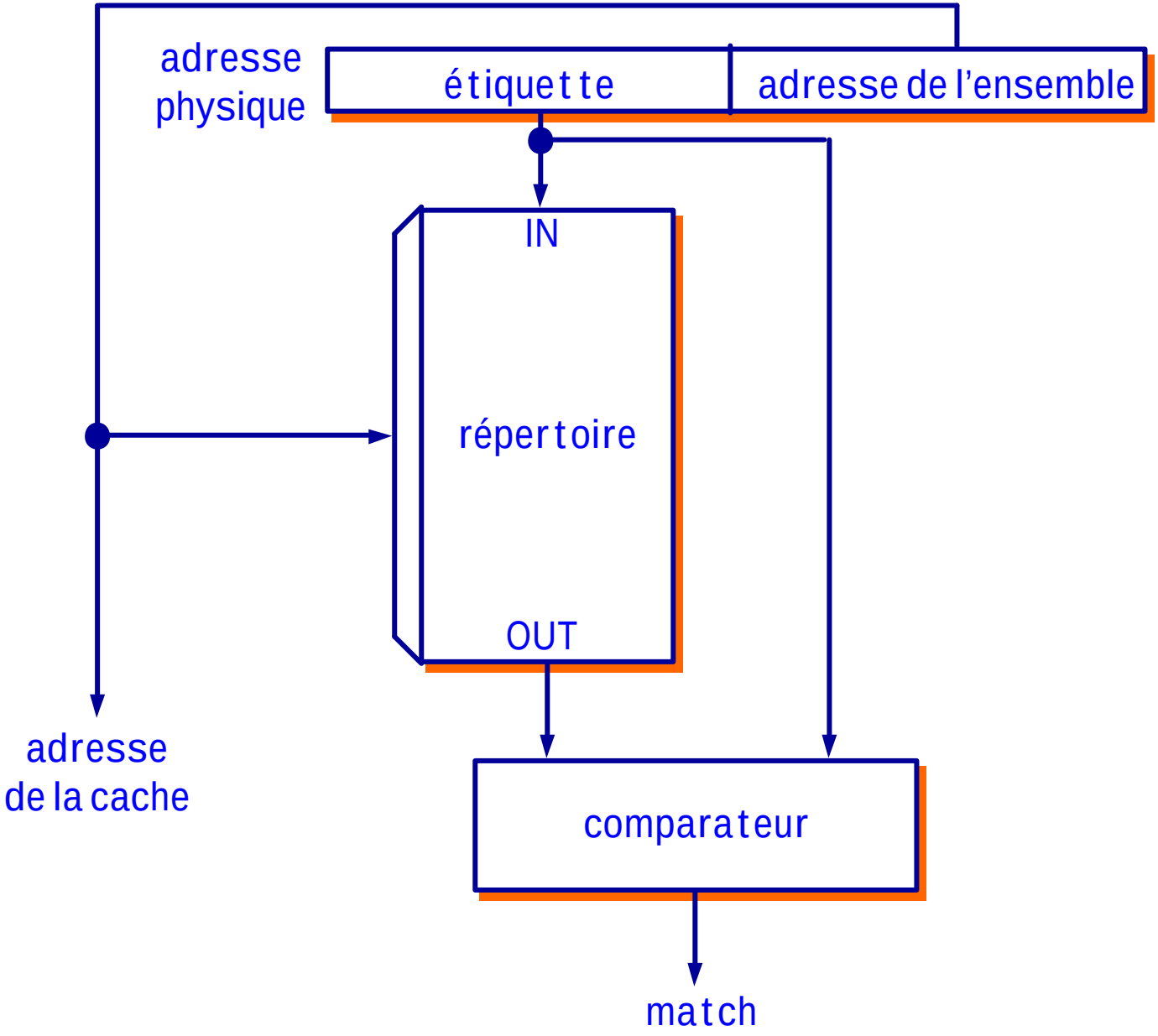
M >> N

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Cache associative par ensembles

- La mémoire principale est divisée en ensembles et la cache est complètement associative pour chaque ensemble
- L'adresse physique est divisée en deux parties:
  - les bits de poids faible contiennent l'adresse de l'ensemble (adresse de la cache): toutes les adresses physiques qui partagent ces bits seront stockées à la même position dans la cache
  - les bits de poids fort forment une étiquette (*tag*), à comparer avec la valeur stockée dans le répertoire pour l'ensemble sélectionné
- Plus rapide que la cache complètement associative: la cache est accédée sans attendre la réponse du répertoire
- Au moment du démarrage, le contenu de la cache est quelconque. Chaque ligne de la cache contient un bit de validité (*valid bit*): il est mis à 1 si le contenu de la ligne est valable

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

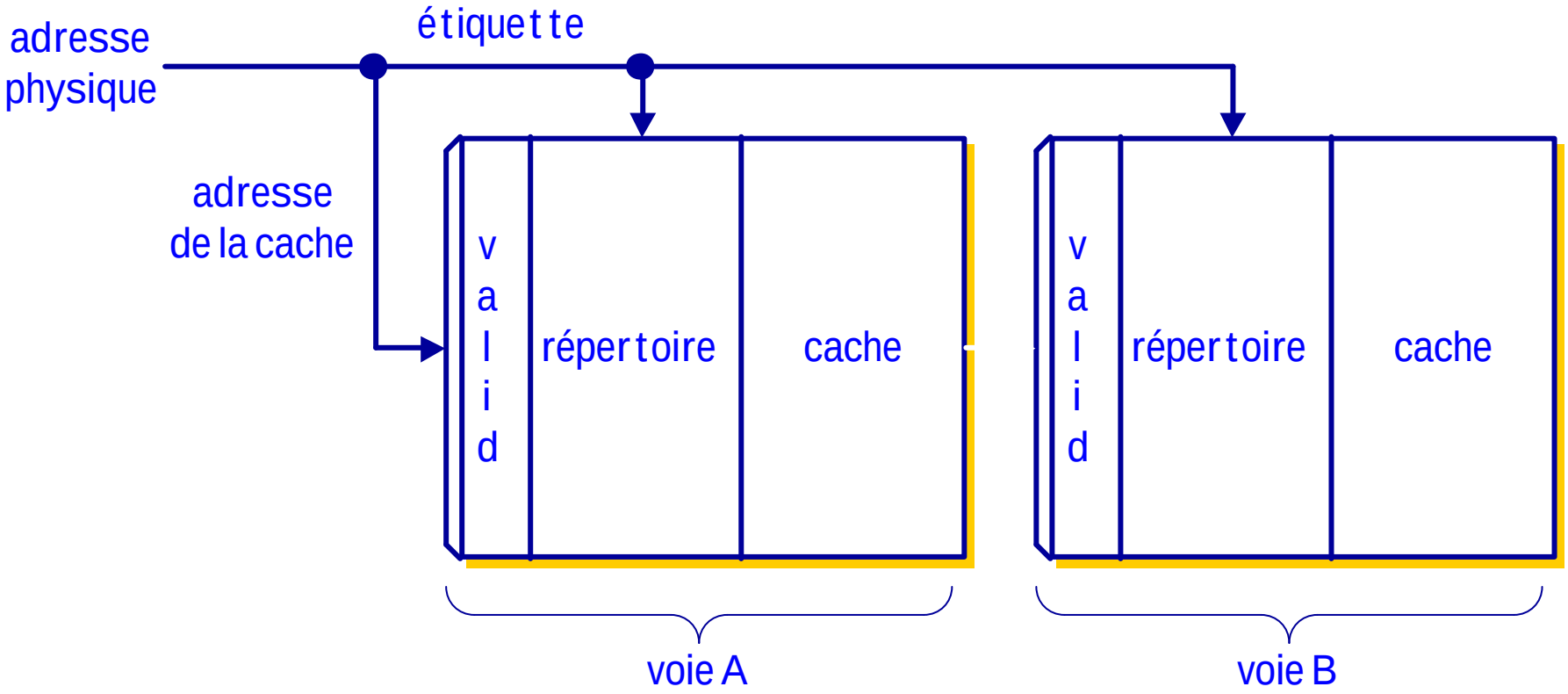


Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Associativité à plusieurs voies

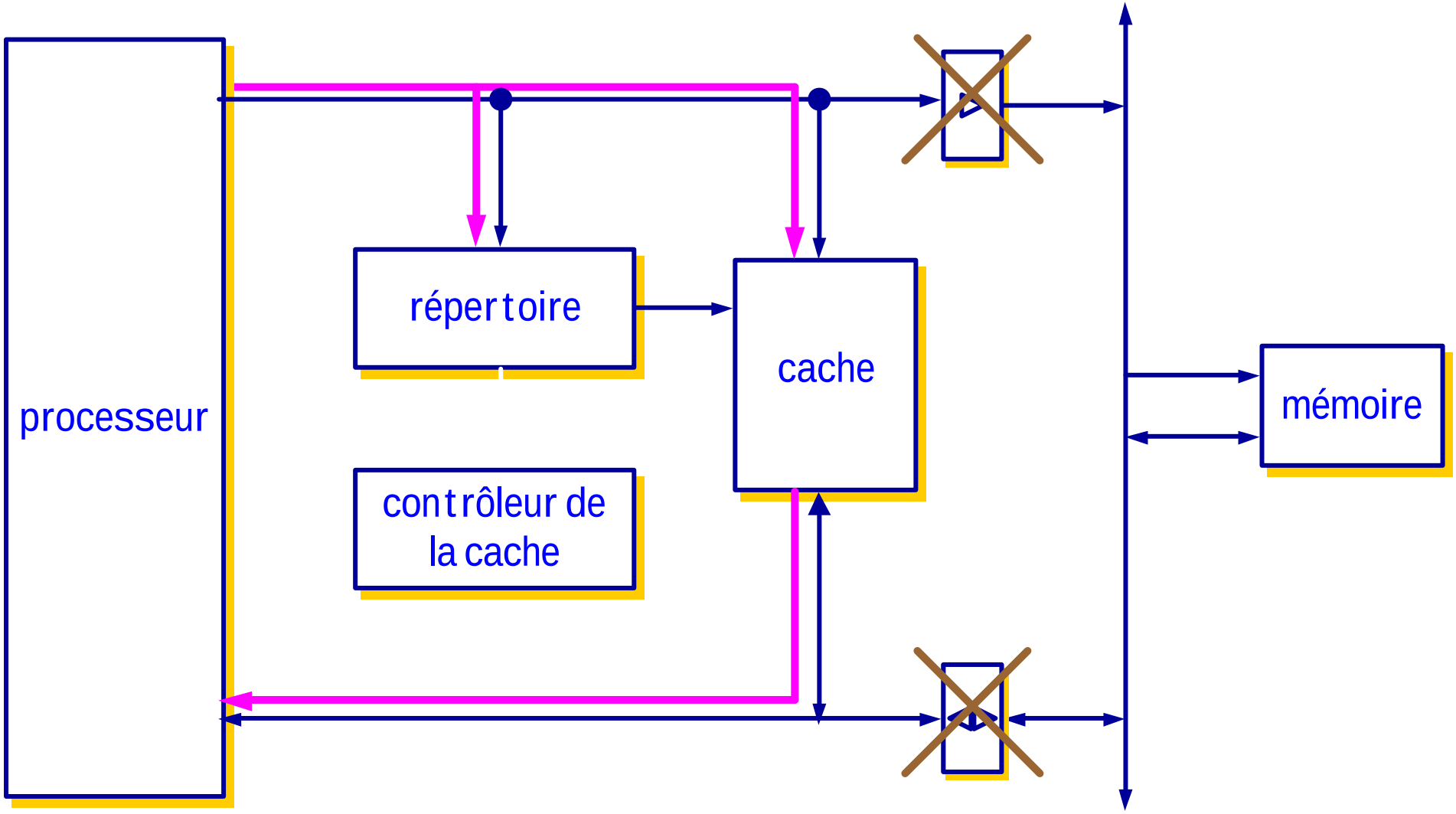
- Dans une cache associative par ensembles, un mot de la mémoire principale peut être stocké à une seule position dans la cache. Ce type de cache est appelé également *direct-mapped*. Si le processeur accède à deux mots dont les adresses partagent les bits de poids faible (l'adresse de l'ensemble), un échec arrive
- Ce problème est résolu dans une cache associative par ensembles à N voies: un mot de la mémoire principale peut être stocké en N positions différentes de la cache
- En général:
  - doubler l'associativité implique une diminution de 20% de la fréquence d'échec
  - doubler la taille de la cache implique une diminution de 69% de la fréquence d'échec

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne



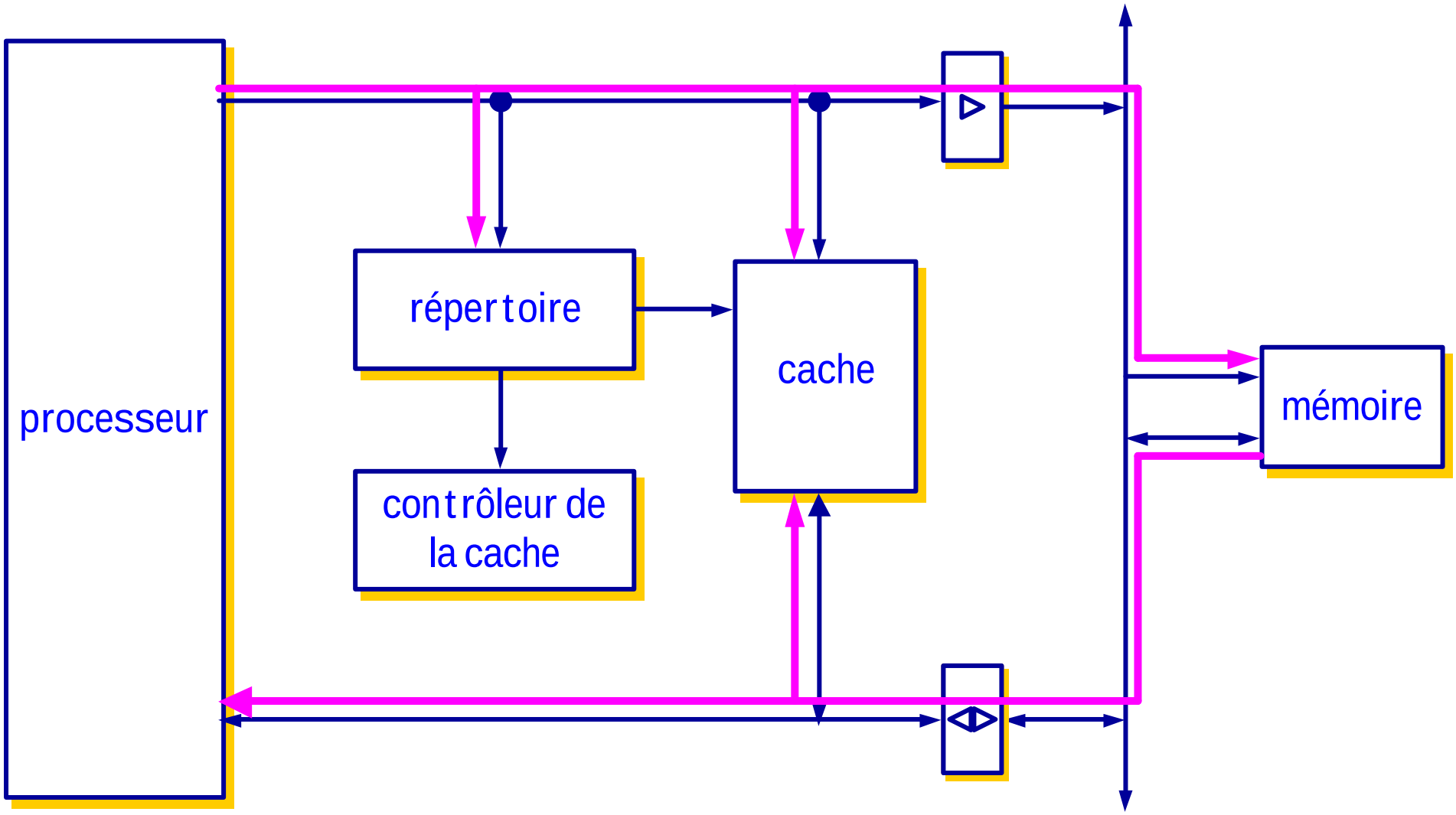
Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Succès en lecture



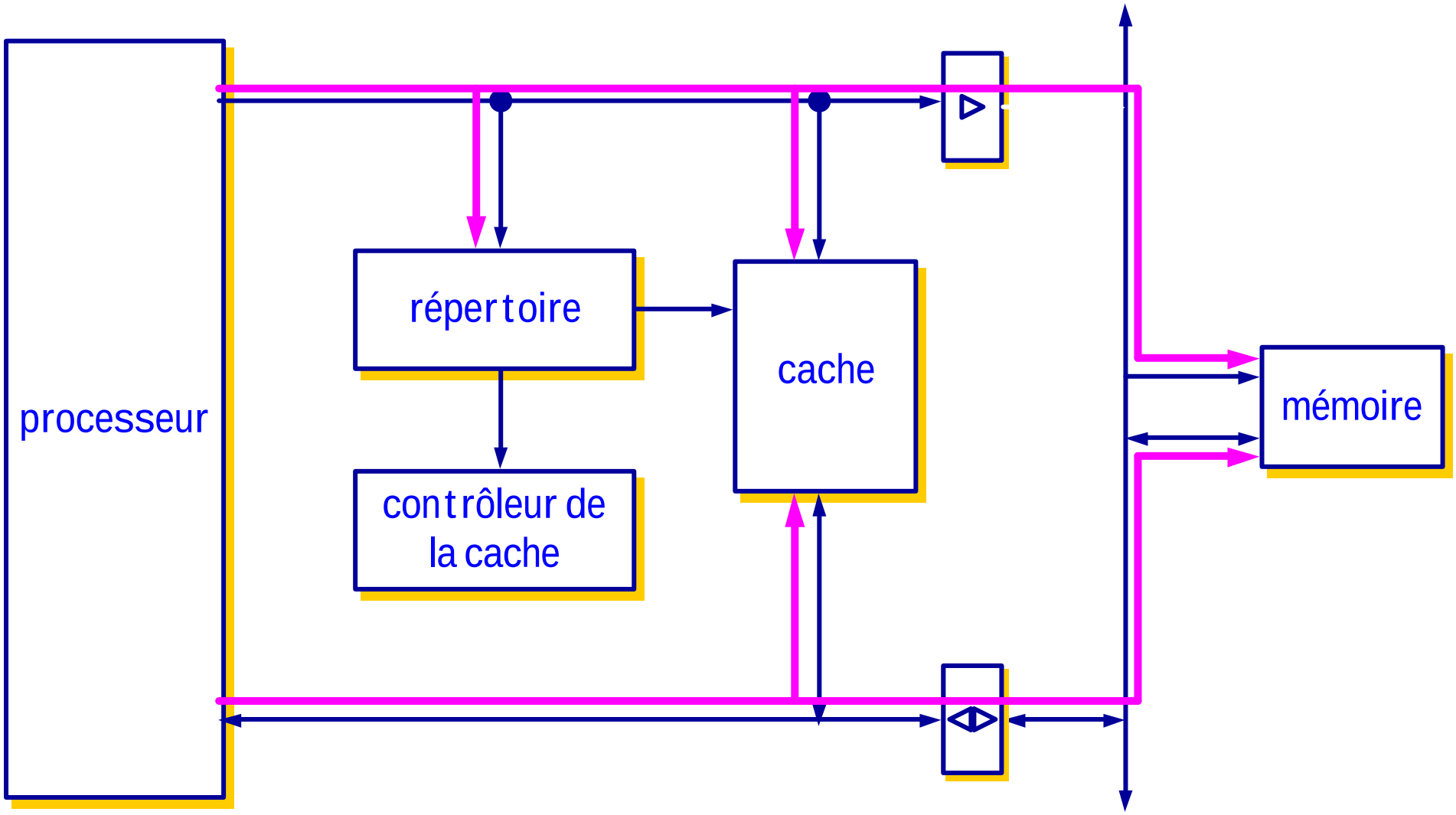
Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Echec en lecture



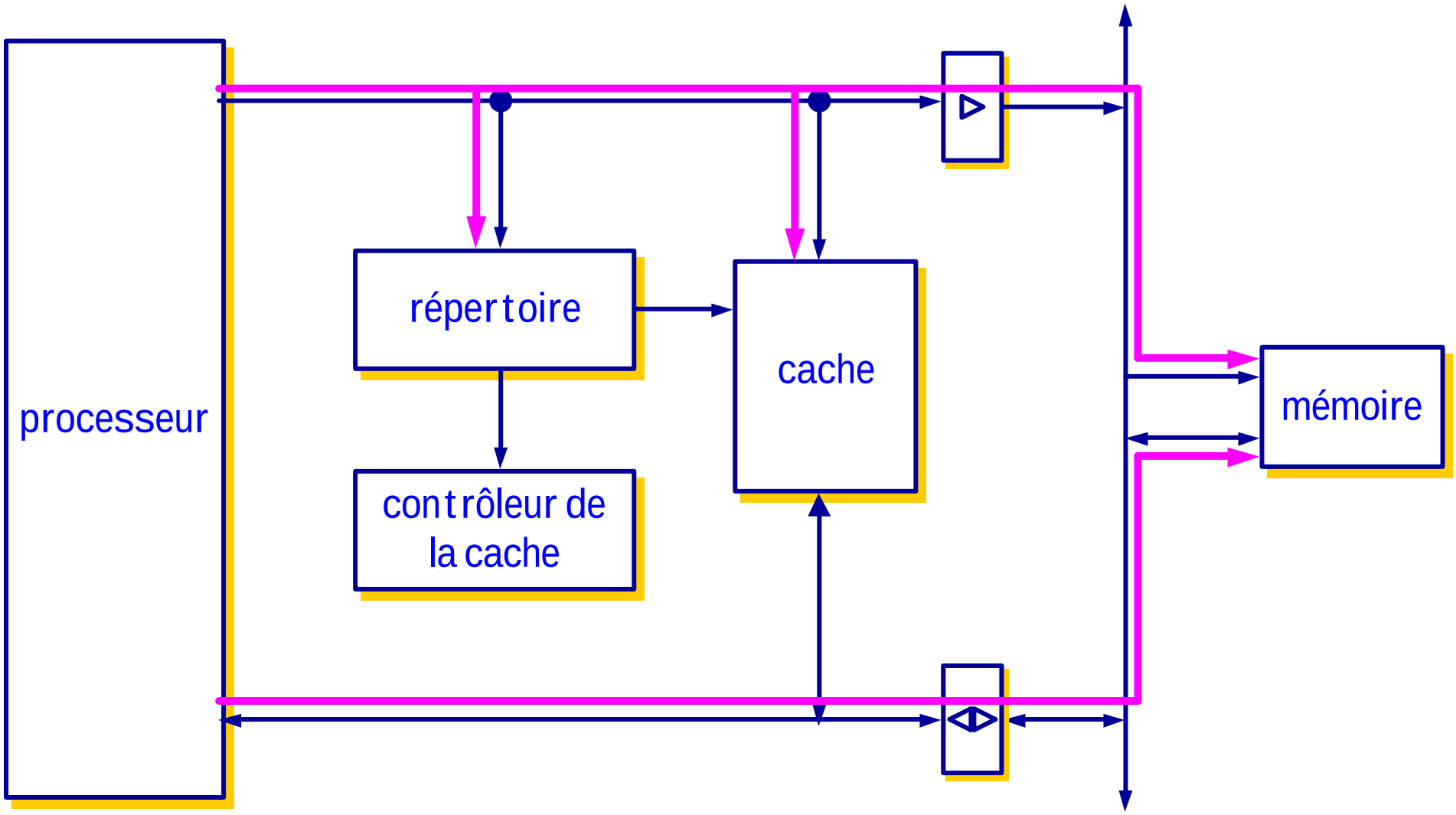
Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Succès en écriture



Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Echec en écriture



Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Algorithmes d'écriture

- *Write-through*:
  la mémoire principale est toujours modifiée lors d'un cycle d'écriture, qu'il s'agisse d'un succès ou d'un échec
- *Copy-back* ou *write-back*:
  l'écriture se fait seulement dans la cache.
  Pour garder la cohérence avec la mémoire principale, on peut écrire dans la mémoire principale toute ligne qu'on enlève de la cache avec une information valable. Pour éviter des écritures inutiles, on ajoute un bit par ligne, le *dirty bit*, pour indiquer que le contenu de la cache est différent de celui de la mémoire
- Dans tous les cas, on peut améliorer le temps d'écriture en ajoutant un buffer d'écriture (*write buffer*): les données sont écrites dans ce buffer et le contrôleur de cache se charge de les envoyer par la suite dans la mémoire

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Algorithme d'effacement

- Méthode LRU (*least recently used*):
  chaque ligne de la cache (même ensemble mais voies différentes) possède des bits pour indiquer l'ordre d'utilisation des voies.
  Pour une cache à 4 voies il y a 24 possibilités (4!): il faudrait 5 bits de codage au minimum. Pour une cache à 16 voies il faudrait 45 bits...
- Méthode NLU (*not last used*):
  on n'écrit pas dans la dernière voie accédée. Une RAM stocke cette information pour chaque ligne
- Remplacement aléatoire

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Taille de la ligne

- Une ligne (ou bloc) de cache est la plus petite portion de la cache avec une étiquette unique
- Les raisons pour avoir des lignes à plus d'un mot sont:
  - avoir une RAM de répertoire plus petite que la RAM de cache
  - avoir des transferts multi-mots (*burst*) plus rapides
- Une ligne possède normalement entre 2 et 8 mots
- Pour réduire le temps de mise à jour lors d'un échec, le bus entre la mémoire et la cache est généralement plus large que celui entre la cache et le processeur

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne

Position de la cache

- Cache physique:
  la cache est placée après le gestionnaire de mémoire virtuelle (MMU): elle reçoit une adresse physique
- Cache logique:
  la cache est placée avant le gestionnaire de mémoire virtuelle (MMU): elle reçoit une adresse virtuelle.
  Ce cas demande une cache moins rapide. Mais une adresse physique peut être traduite à plusieurs adresses logiques: si elles sont toutes dans la cache, l'écriture dans l'une laisse les autres inchangées

Eduardo Sanchez
Ecole Polytechnique Fédérale de Lausanne