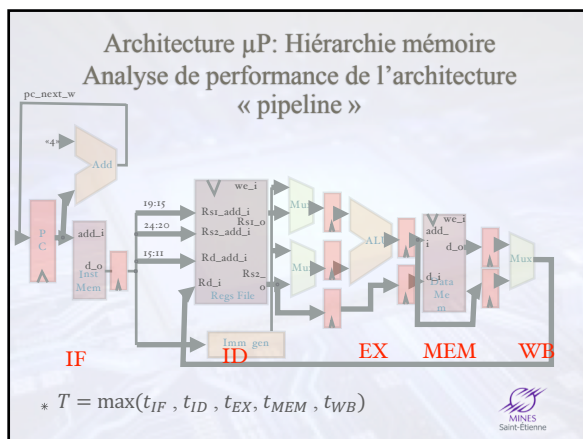


Architecture des microprocesseurs hiérarchie mémoire

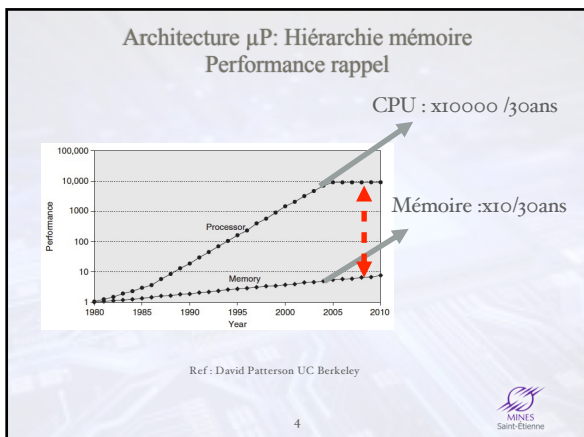
Michel Agoyan : michel.agoyan@st.com
 Marc Lacruche : marc.lacruche@st.com
 Simon Pontie : simon.pontie@cea.fr
 Olivier Potin : olivier.potin@emse.fr
 Come Allart : come.allart@emse.fr
 Raphaël Comps : rcomps@emse.fr

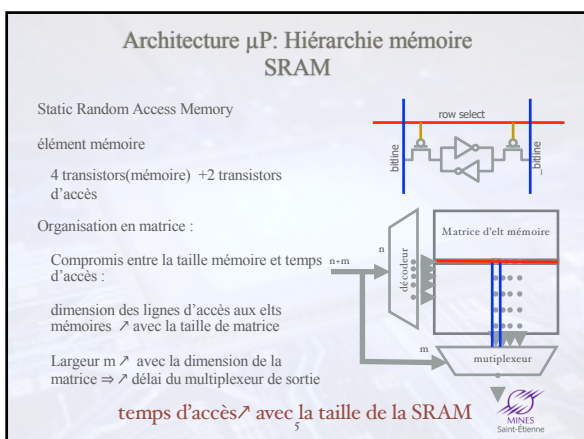


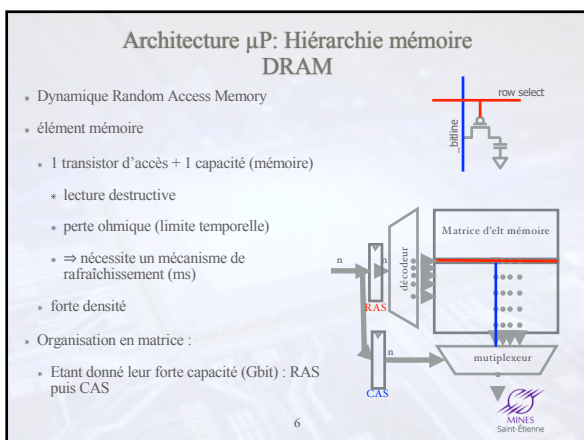
Architecture μP : Hiérarchie mémoire Analyse de performance de l'architecture « pipeline »

- * $T = \max(t_{IF}, t_{ID}, t_{EX}, t_{MEM}, t_{WB})$
- * Les étages les plus critiques sont ceux pour lesquels un accès mémoire est effectué : t_{IF} et t_{MEM}
- * Exemple si : $t_{IF} = t_{MEM} = 2 * t_u$ et $t_{ID} = t_{EX} = t_{WB} = t_u \rightarrow T = 2 * t_u$
- * Pour l'architecture monocycle : $T = 7 * t_u$
- * L'efficacité du pipeline devient maximale lorsque tous les étages ont une durée de traitement égale
- * Comment réduire le temps d'accès à la mémoire ?










Architecture μP : Hiérarchie mémoire SRAM/DRAM

SRAM				
64 KB	256 KB	1 MB		
1 ns	10 ns	20 ns		

DRAM				
capacité	type de DRAM	freq	temps d'accès CAS (nb de périodes)	temps d'accès RAS (ns)
↓	DDR2-1066	533MHz	4	7.5ns
	DDR3-1066	533MHz	7	13ns
	DDR3-1600	800MHz	6	7.5 ns
	DDR4-1600	800MHz	10	12.5ns


Temps d'accès \nearrow avec la taille mémoire $> (>>)$ temps de cycle processeur

7




Architecture μP : Hiérarchie mémoire Propriétés de localité des programmes

- Le flot d'exécution d'un programme est une succession de séquences linéaires, de boucles, manipulant les mêmes structures de données :
- Localité temporelle** : si un emplacement est accédé à un instant il a de forte chance d'être accédé de nouveau dans un futur proche
- Localité spatiale** : si un emplacement est accédé à un instant il a de forte chance qu'un emplacement voisin soit accédé dans un futur proche



8




Architecture μP : Hiérarchie mémoire Cache principe

Idee : pour diminuer l'impact du temps d'accès des mémoires de forte capacité :


Exploiter les propriétés de localité temporelle et spatiale :

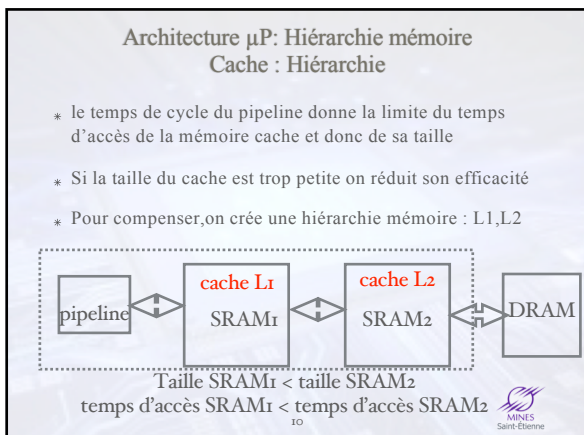
Stocker les données récentes dans une mémoire de plus faible capacité mais plus rapide (sans impact sur CPI) : **mémoire cache**

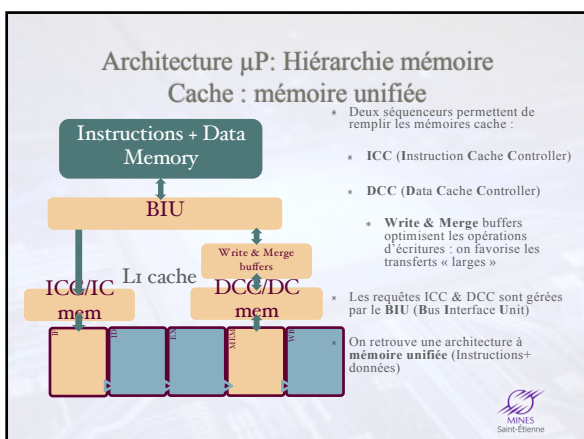


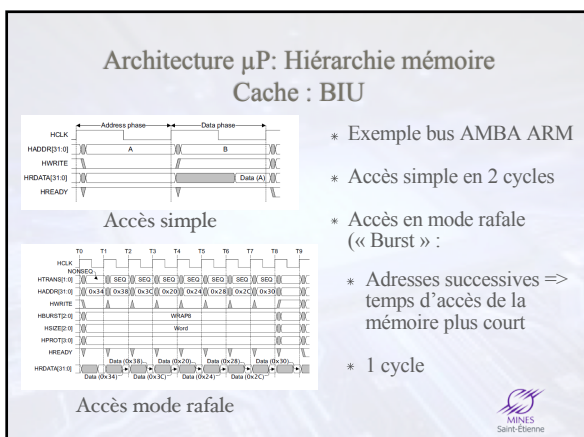
temps accès SRAM $<<$ temps accès DRAM
taille SRAM $<<$ taille DRAM

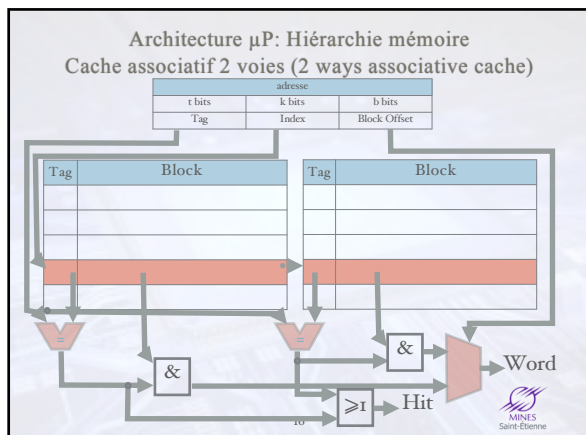
9

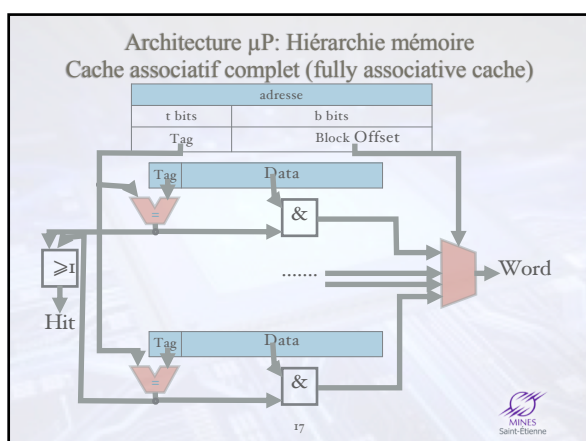


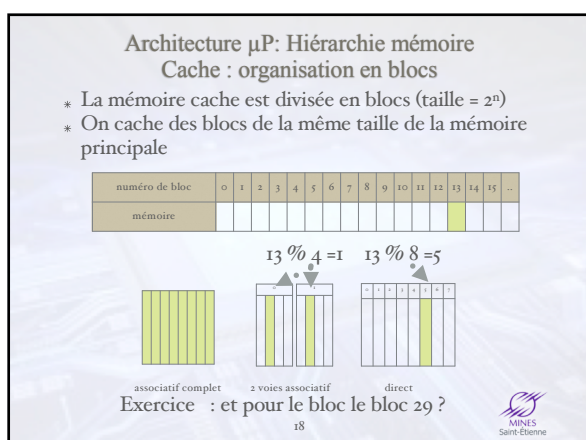












Architecture μP : Hiérarchie mémoire remplacement des lignes de cache

- * Que faire quand la ligne de cache « victime » est occupée ?
- * Plusieurs stratégies existent :
 - * Least Recently Used (LRU)
 - * Facile à implémenter sur des caches à 2 voies
 - * FIFO
 - * utilisé pour les caches associatifs à plusieurs voies
 - * Not Least Recently Used (NLRU)
 - * utilisé pour améliorer le mode FIFO
 - * MFU (Most Frequently Used)
 - * Aléatoire

19



Architecture μP : Hiérarchie mémoire Cache différent mode de gestion de l'écriture

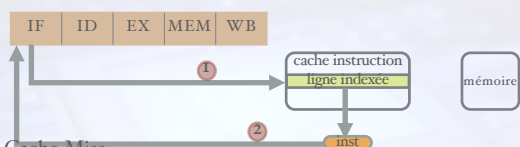
- * Cache Hit :
 - * « Write Through » :
 - * on met à jour à la fois la ligne de cache et la mémoire
 - * Simplifie la gestion de cohérence du cache mais impacte les performances
 - * limitation de l'impact par l'ajout d'un buffer d'écriture « Write buffer »
 - * « Write back » :
 - * On met seulement à jour la ligne de la mémoire cache
 - * Les données seront écrites en mémoire lorsque la ligne est invalidée (utilisation d'un bit indicateur : « dirty bit »)
- * Cache Miss :
 - * sans allocation (no write allocate) on ne met pas jour la mémoire cache
 - * avec allocation

20

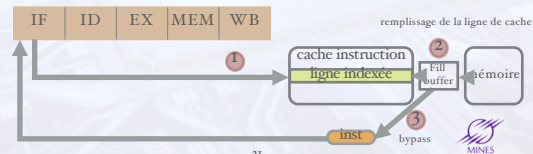


Architecture μP : Hiérarchie mémoire lecture du cache instruction

- * Cache Hit :

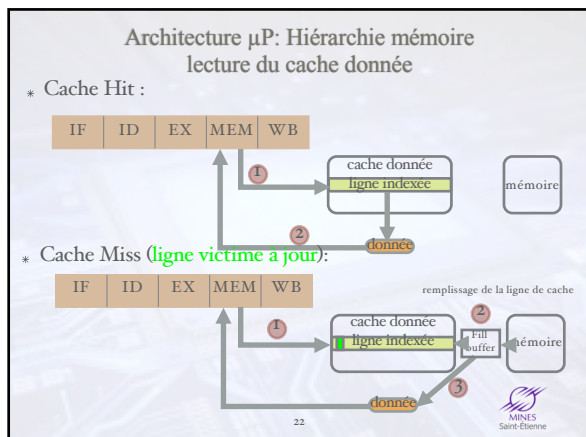


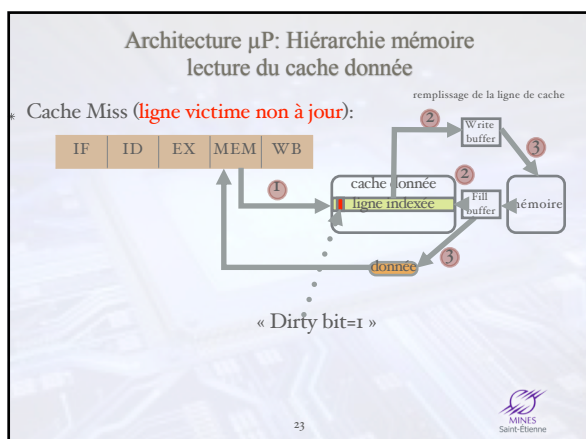
- * Cache Miss:

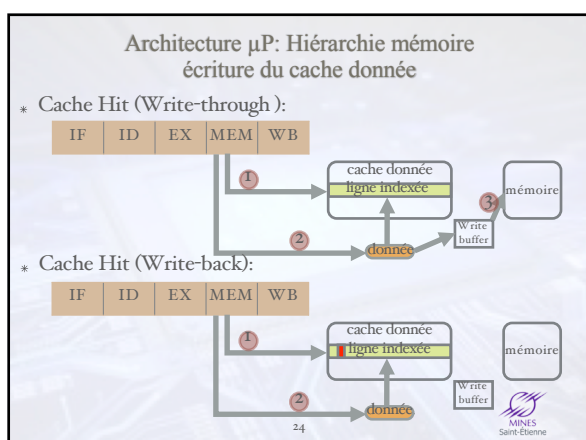


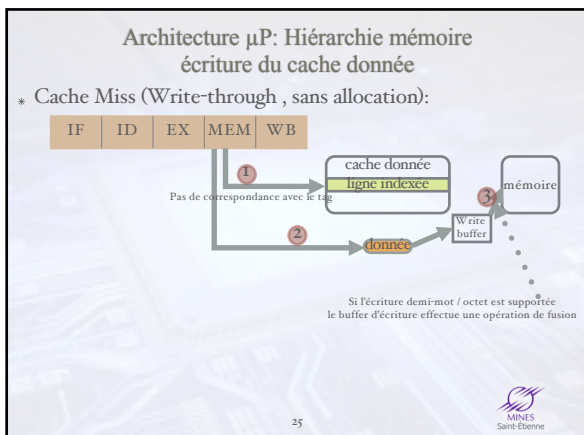
21

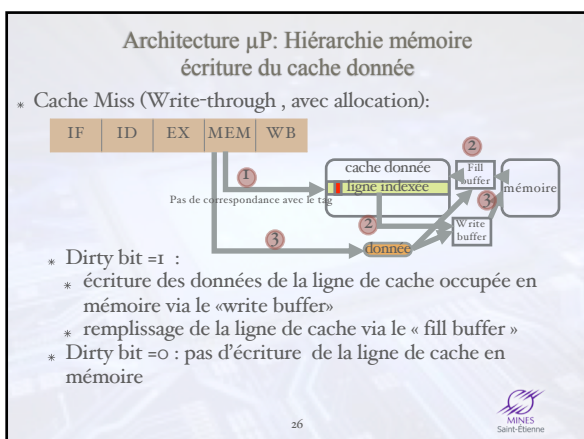


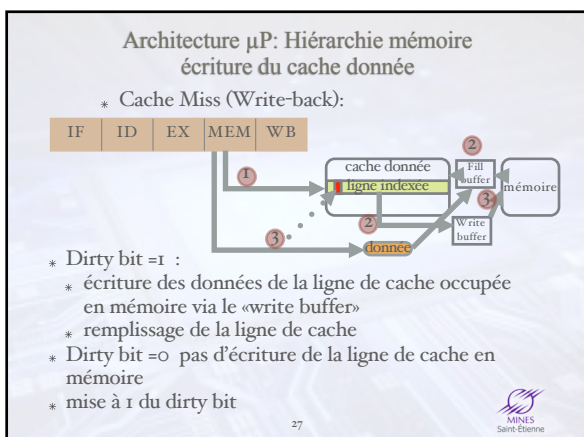












Architecture des microprocesseurs Unité de Gestion Mémoire MMU (Memory Management Unit)

28



Architecture μ P: Unité de gestion Mémoire

Les problèmes à résoudre sur les OS modernes (multi utilisateurs , multitâches)

Fragmentation de la mémoire :

Plusieurs utilisateurs => Allocation dynamique de la mémoire

Espace d'adressage privé par Utilisateur

Espace mémoire limité :

Plusieurs utilisateurs => ↗ taille mémoire système requise



Architecture μ P: Unité de gestion Mémoire

* Fragmentation :

