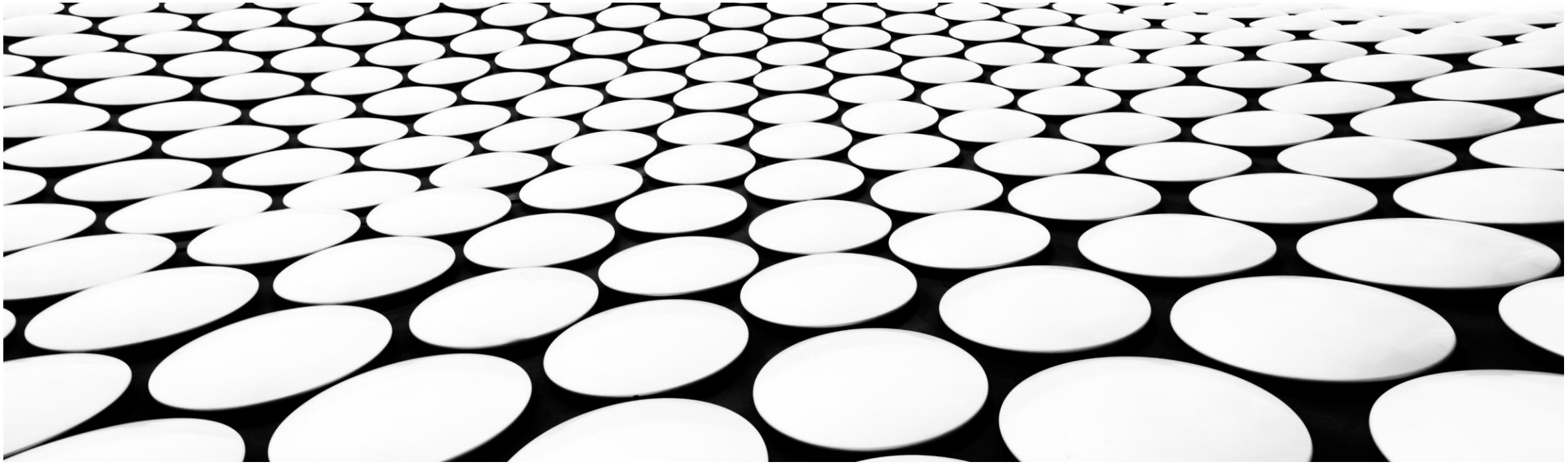
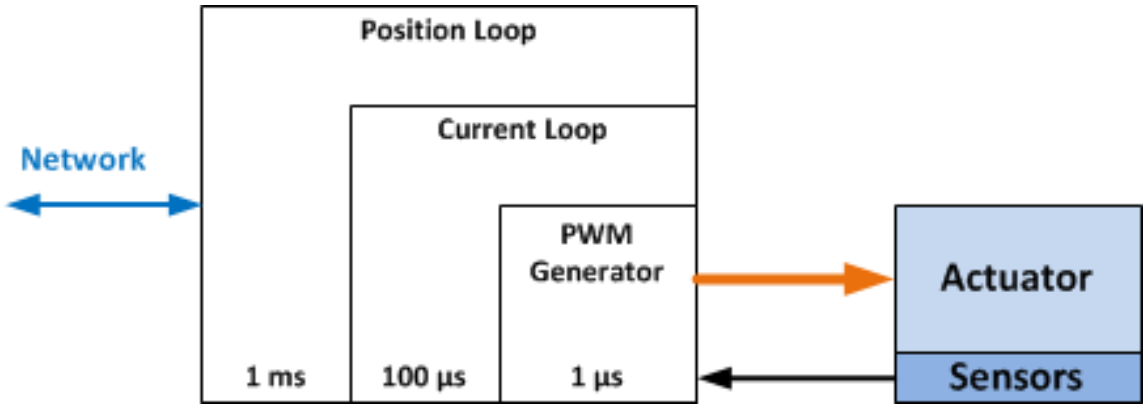
ES207 : COMPOSANTS ET ARCHITECTURES EMBARQUÉES

BRUNO MONSUEZ (ENSTA-PARIS/U2IS)



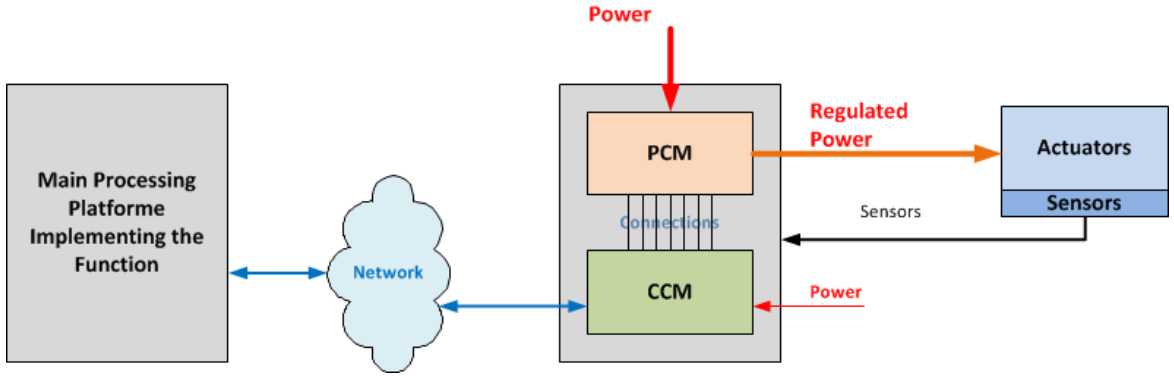
COMMANDER UN ACTIONNEUR

- Boucles de contrôle de l'actionneur



PWM : Pulse With Modulation
PCM : Power Control Module
CCM : Command Control Module

- Architecture typique de contrôle d'un actionneur

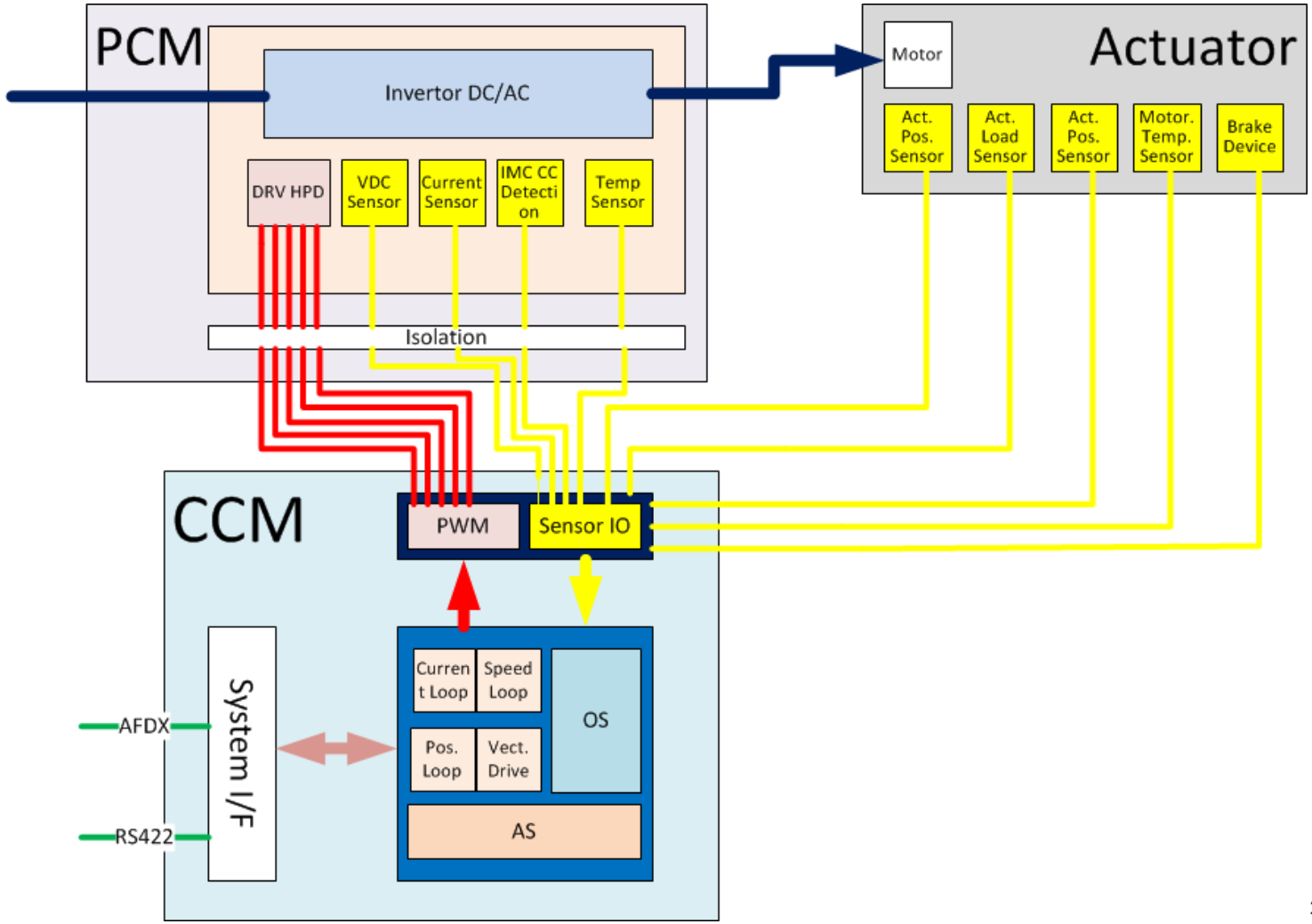


ELEMENTS D‘ARCHITECTURES

- Control Board (CCM)
  - Héberge le logiciel
  - Gère la communication avec le système
  - Pilote la carte PCM
- Power Drive (PCM)
  - Génère uniquement les signaux CCM

- Répartition des Fonctions
  - Boucle Position & Vitesse sur la CCM
  - Génération de Courant sur la CCM
  - Génération du PWM sur la PCM
- Avantages
  - Séparation entre la partie „électronique de commande“ et „électronique de puissance“
  - Carte complexe CCM
  - Carte simple PCM

EXEMPLE D‘IMPLANTATION PHYSIQUE



EXIGENCES SUR LES CARTES/EQUIPEMENTS

- Control Board (CCM)
  - Microprocesseur executant un OS (Operating System) ou un AS (Application System)
  - Support de communication vers d'autres équipements (CAN, Ethernet, …)
  - Interfaçage avec de nombreux capteurs
  - Fonctions implantées
    - Commande
    - Interaction et communications avec les autres systems
    - Initialisation/Configuration
    - Surveillance de la commande/détection de pannes
    - Support des mises à jours
- Power Drive (PCM)
  - Composant hardware simple (Microcontroleur, FPGA ou CPLD)
  - Connexion avec la carte de contrôle
  - Mise en sécurité électrique

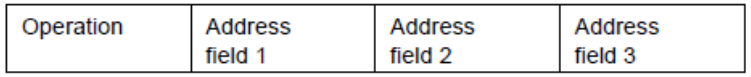
MICROPROCESSEUR

LE COEUR CENTRAL D‘UN SYSTÈME EMBARQUÉ

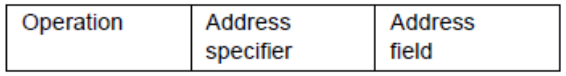
29.03.2022

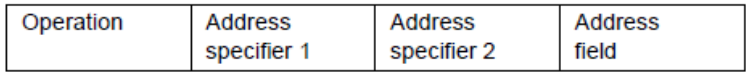
ENCODAGE ET TYPES D‘INSTRUCTIONS

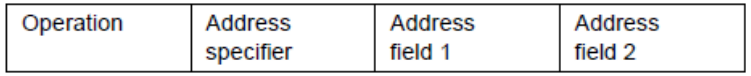
(a) Variable (e.g., VAX)



(b) Fixed (e.g., DLX, MIPS, Power PC, Precision Architecture, SPARC)



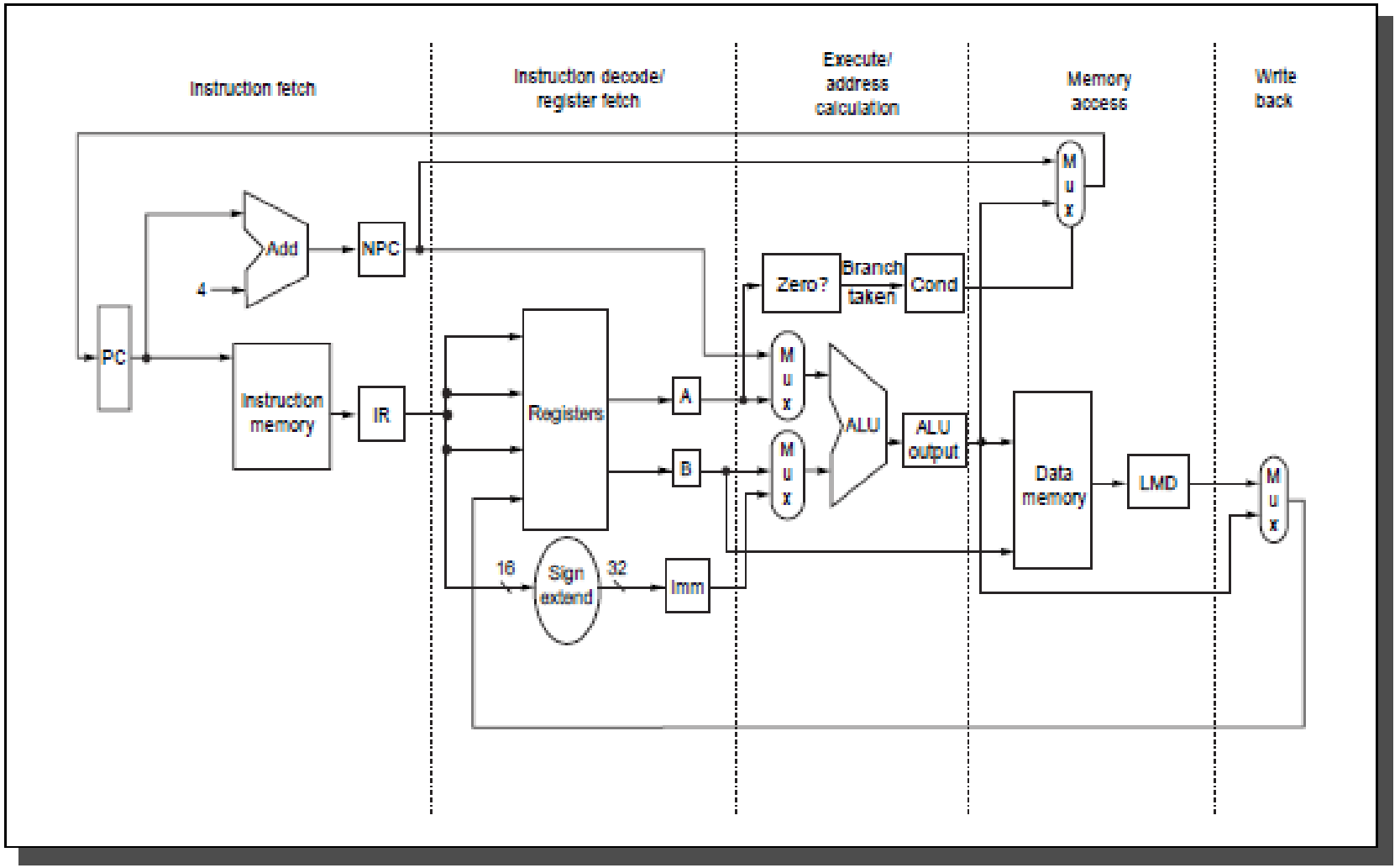




(c) Hybrid (e.g., IBM 360/70, Intel 80x86)

Operator type	Examples
Arithmetic and logical	Integer arithmetic and logical operations
Data transfer	Load, Store, Move
Control	Branch, Jump, Call and Returns
System	System call, memory management, cache…
Floating point	Floating-point operations
String	String move, compare, search
Multimedia	Vertex, Compresssion/Decompression, Mapping

UN SIMPLE MICROPROCESSEUR



FONCTIONNEMENT D‘UN PROCESSEUR

- *IF Instruction Fetch Cycle*

  IR ← Mem (PC)

  NPC ← PC + 4

- *ID Instruction Decode/Register Fetch*

  A ← Regs[$IR_{6..10}$]

  B ← Regs[$IR_{11..15}$]

  Imm ← ((IR_{11})^{16}##$IR_{11..15}$)

- *EX Execution/Effective address cycle*

  ALUOutput ← A + Imm

  ALUOutput ← A op B

  ALUOutput ← A op Imm

  ALUOutput ← NPC + Imm

  Cond ← (A op 0)

- MEM Memory Access/Branch Completion

  LMD ← Mem[ALUOutput]
  Mem[ALUOutput] ← B

  If (Cond) PC ← ALUOutput

- WB Write/Back Cycle

  Regs[$IR_{16..10}$] ← ALUOutput

  Regs[$IR_{11..15}$] ← ALUOutput

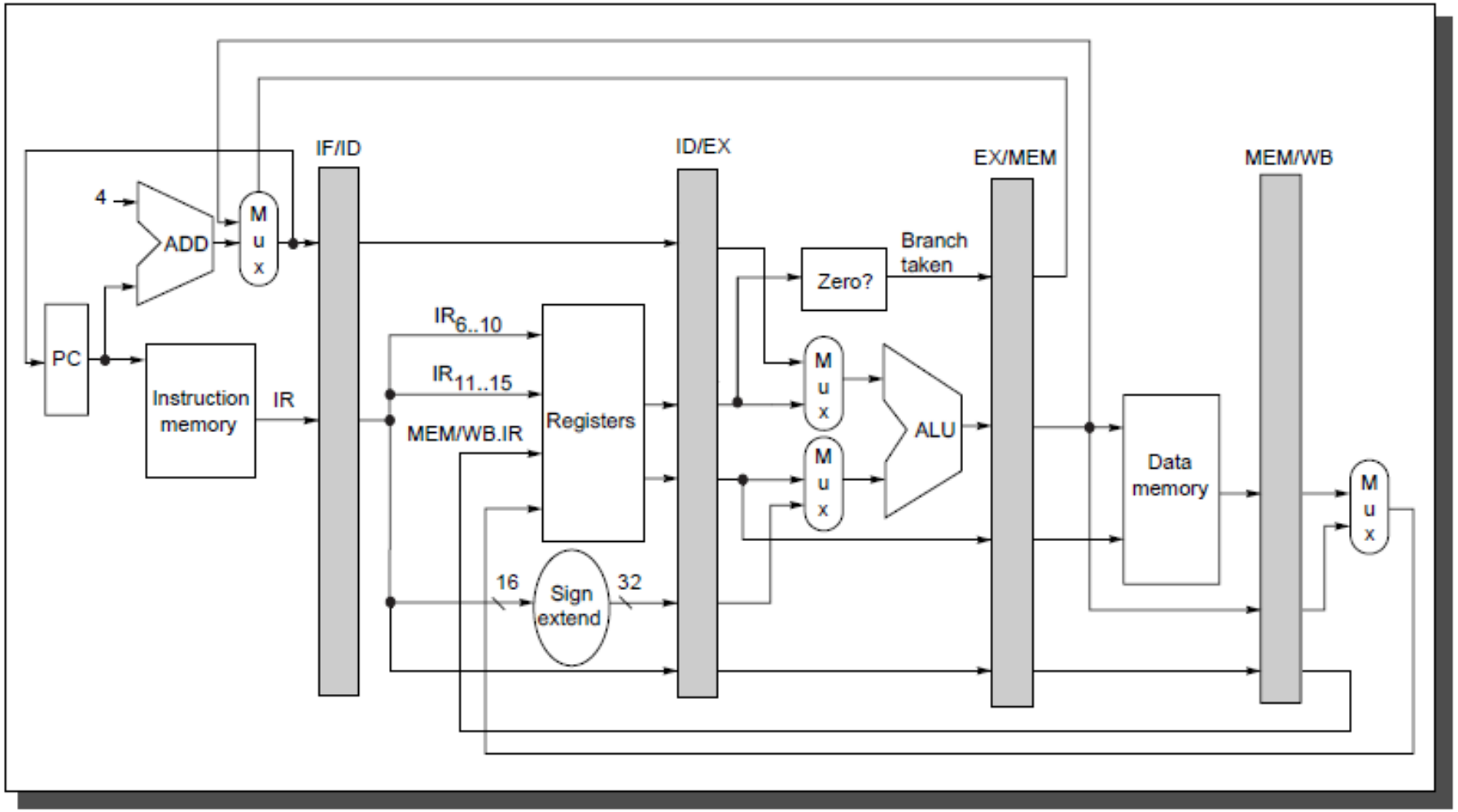
  Regs[$IR_{11..15}$] ← LMD

ACCELERER LE CALCUL/PIPELINER

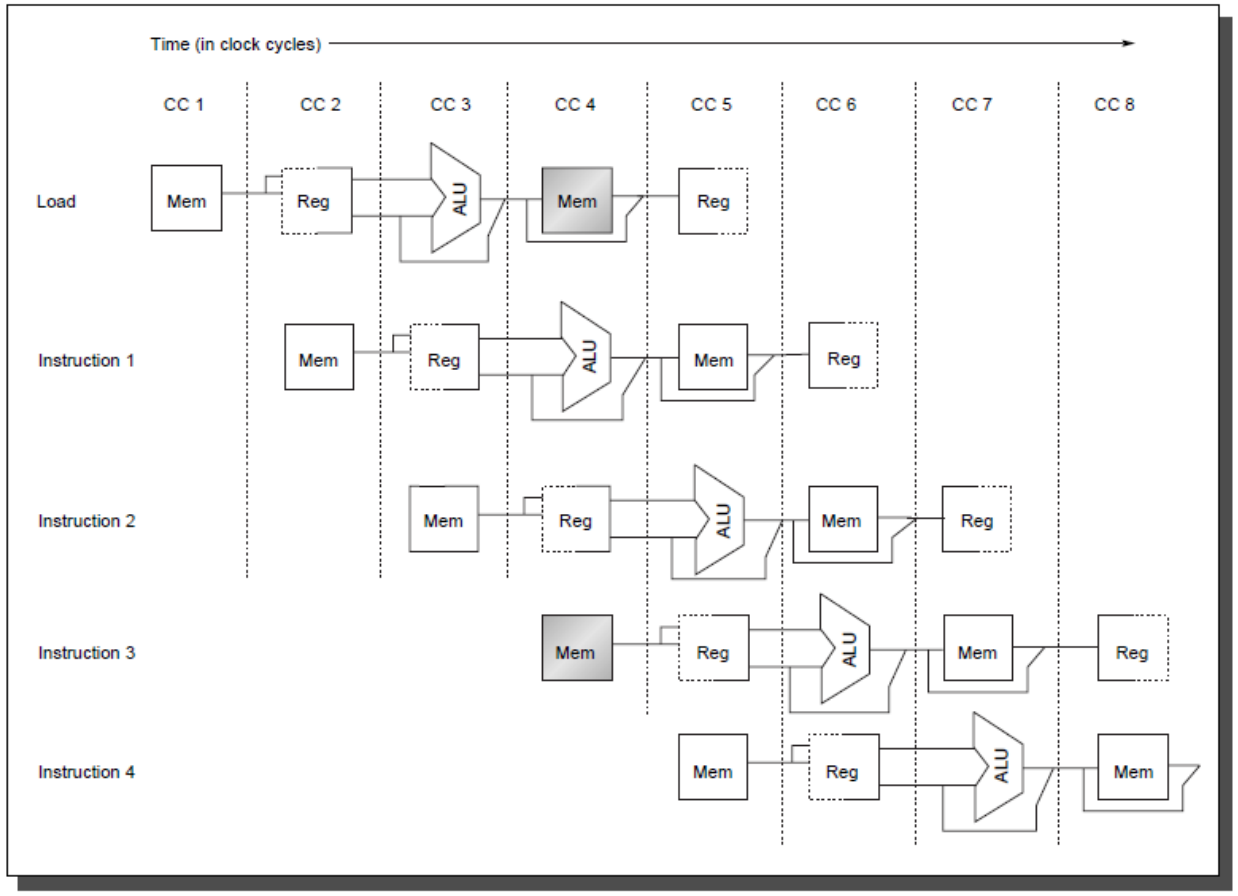
Instruction	1	2	3	4	5	6	7	8	9
Inst	IF	ID	EX	MEM	WB				
Inst +1		IF	ID	EX	MEM	WB			
Inst +2			IF	ID	EX	MEM	WB		
Inst +3				IF	ID	EX	MEM	WB	
Inst + 4					IF	ID	EX	MEM	WB

- INTERETS
  - Execution de plus d‘une instruction en parallèle
  - Réduction de la taille d‘un chemin d‘éxécution => Augmentation de la fréquence

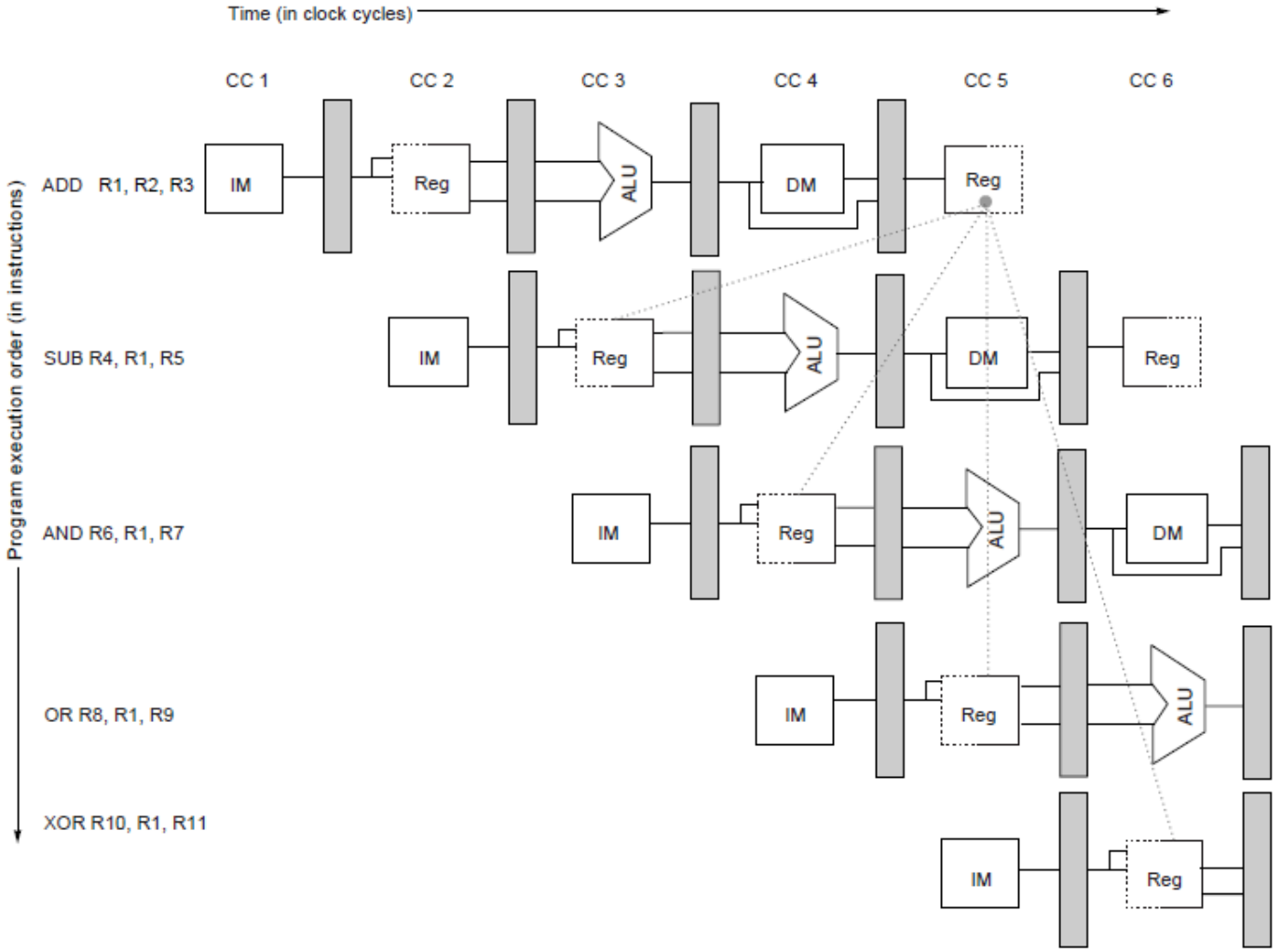
LE MICROPROCESSEUR DLX PIPELINE



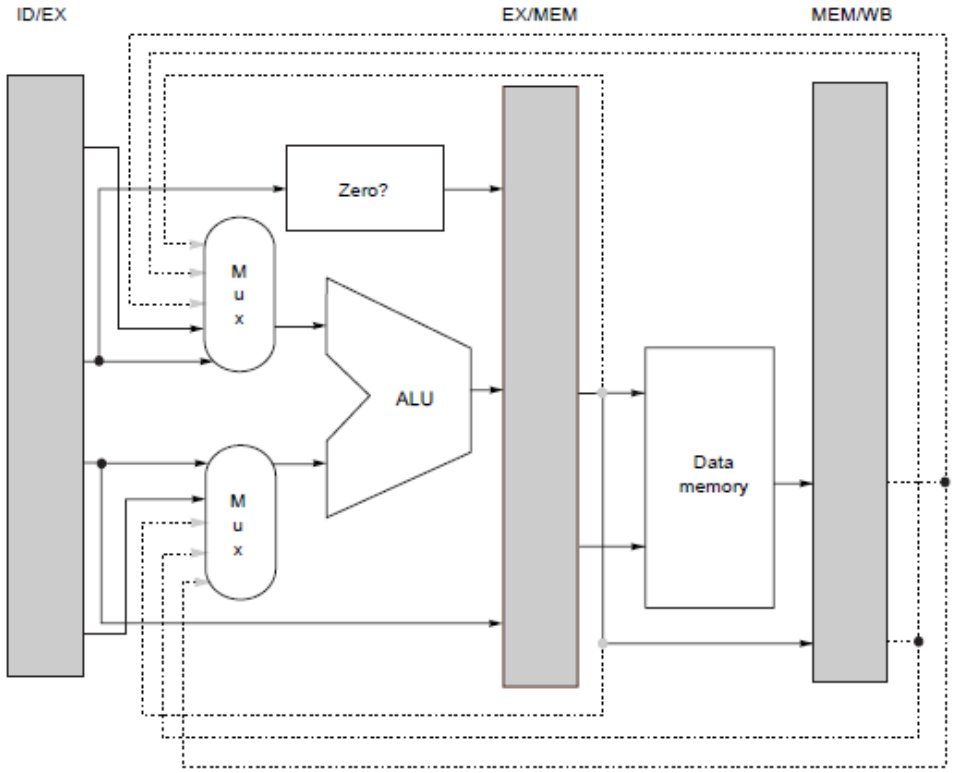
STRUCTURAL HAZARD/ACCES A UNE RESSOURCE PARTAGEE



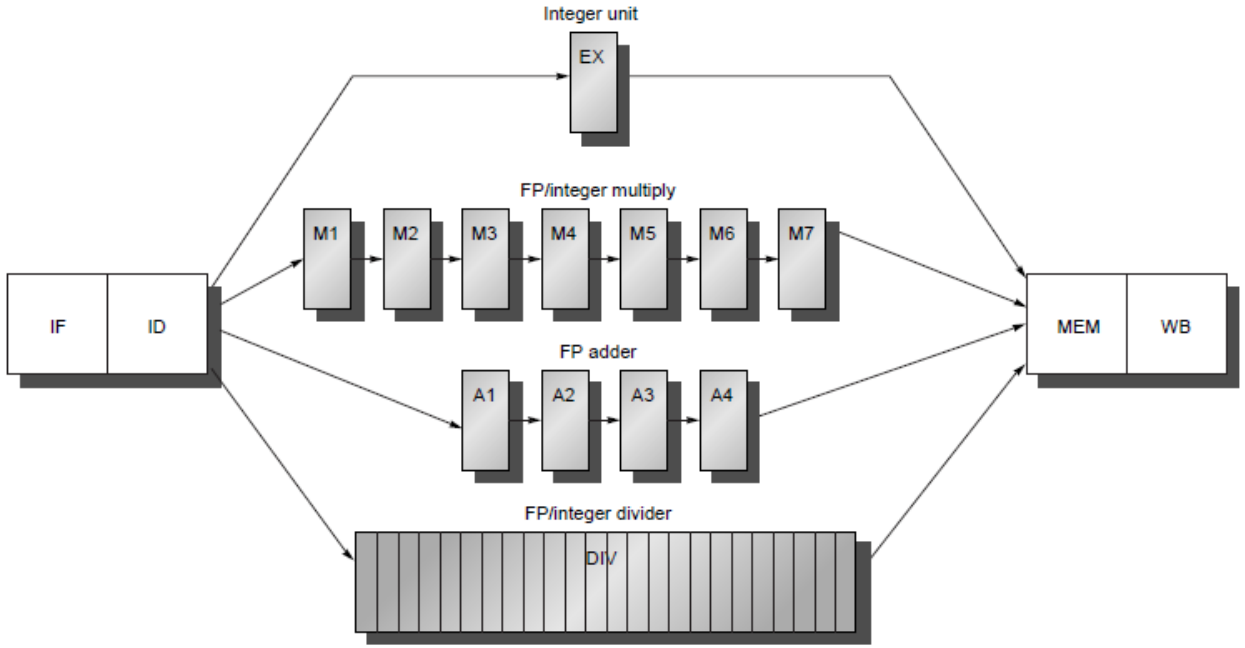
DATA HAZARD/ACCÈS À UNE DONNEE NON ENCORE DISPONIBLE



MASQUER LA LATENCE/DATA FORWARDING UNIT



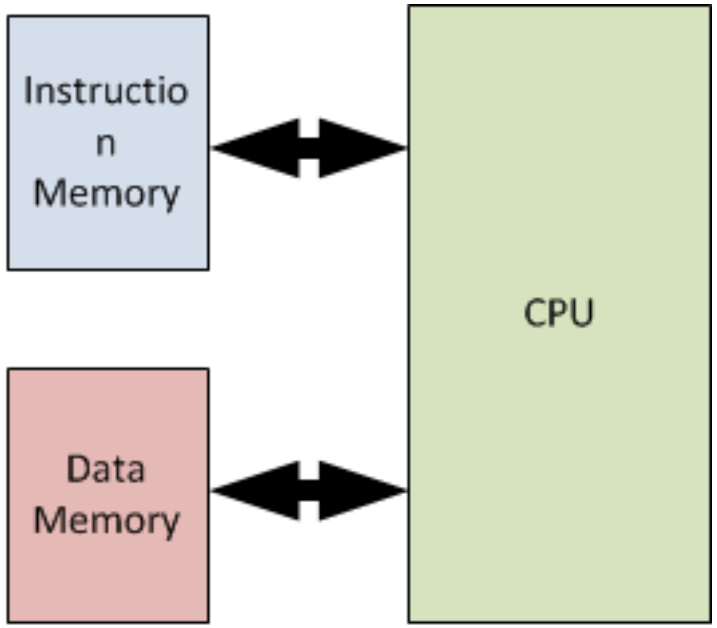
MICRO-PIPELINES



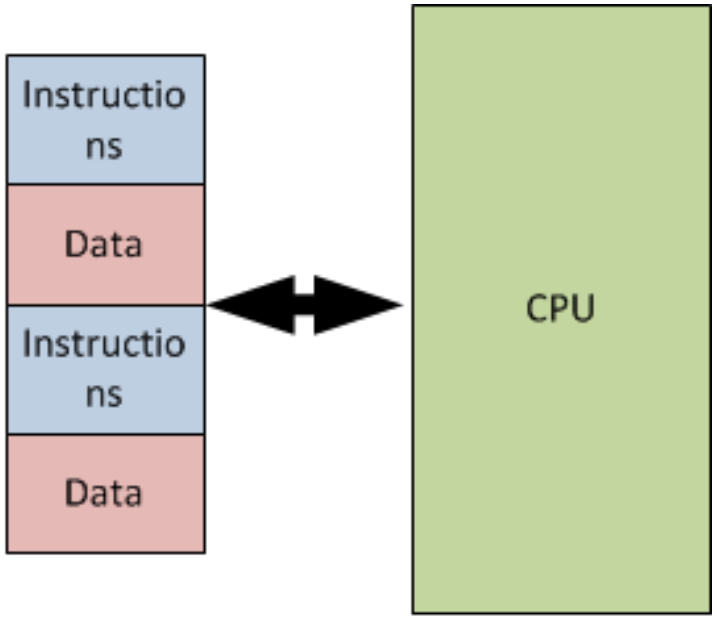
- Introduction d‘instructions multi-cycles
- Introduction de micro-pipeline pour parallèliser ces instructions bloquantes ou non bloquantes.

ACCÈDER À LA MÉMOIRE

MÉMOIRE SEPARÉE OU MÉMOIRE UNIFIÉE

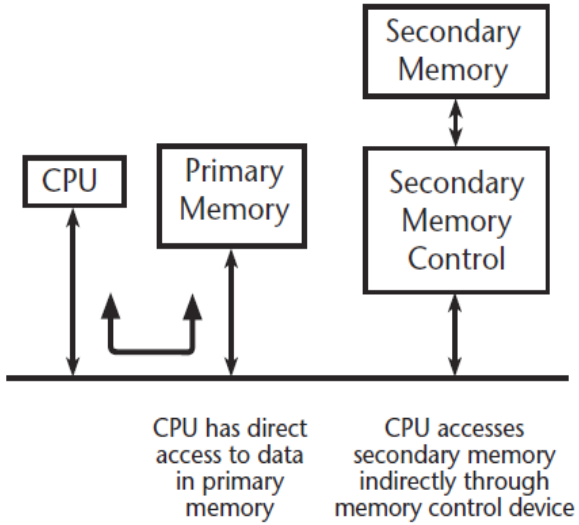


Harvard
Architecture



Von Neumann
Architecture

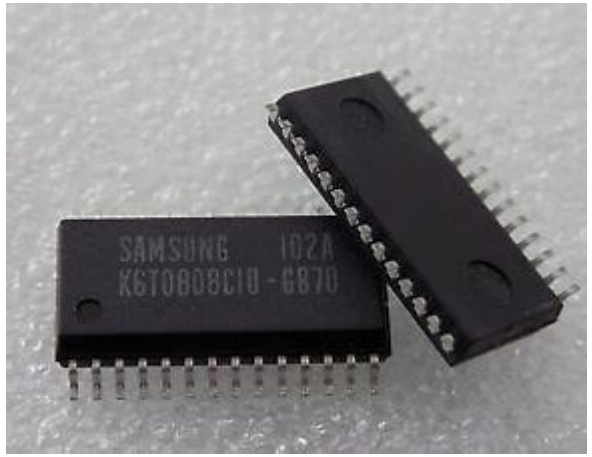
LES DIFFERENTS TYPES DE MÉMOIRE À ACCÈS DIRECT (RANDOM ACCESS MEMORY)



- Deux types de mémoire
  - La mémoire principale (directement connectée au processeur)
  - La mémoire secondaire (les données doivent transiter par la mémoire principale)
- RAM (Random Access Memory)
  - Accès à une donnée située à un endroit dans la mémoire dans un temps quasi-immédiat
- Nombreuses technologies de mémoires à accès direct
  - SRAM, DRAM (DDR…), Q-DRAM, RD-RAM, V-RAM, PPM DRAM
  - ROM, NAND Flash, NOR Flash…

ORGANISATION DE LA MÉMOIRE

- Accès par ligne et colonne (banque)
  - Sélectionne le bit devant être lu
  - Sélectionne le chip contenant la donnée
  - Un ensemble de chips peut-être vue comme un seul chip (8 IC fournissant un octet)
- Taille des mémoires exprimées par 2^n x m
  - 2^n le nombre de lignes stockées dans l'IC
  - M le nombre de bits stockés pour chacune des lignes.
- 32K x 8
  - Taille du mot : 8 bits
  - Taille du composant : 32 Kilooctets ou 4096 Kilobits
  - 15 lignes d'addresse (2^15 = 32k)
  - 8 bits par ligne de données.



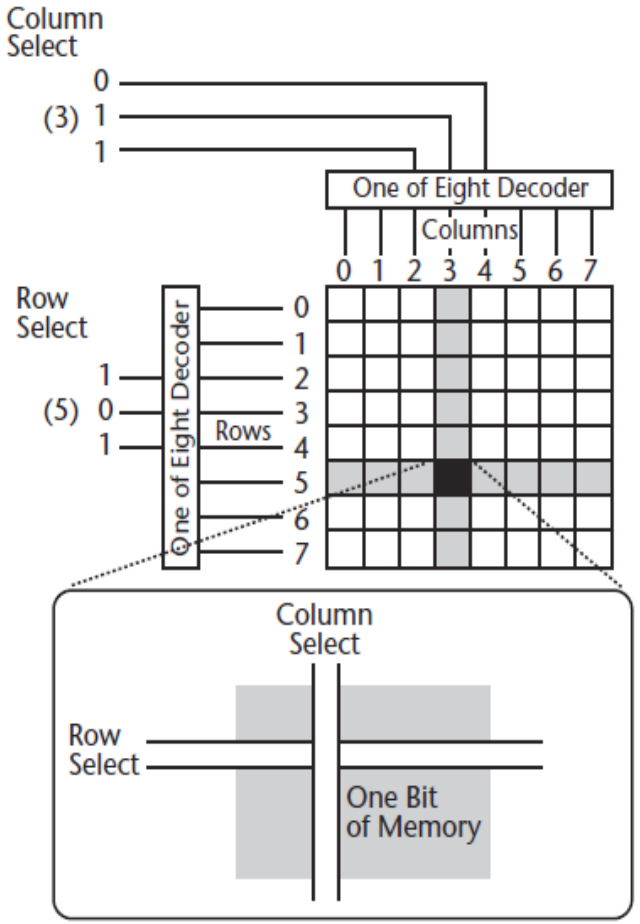


Figure 4-3: Random access memory (RAM).

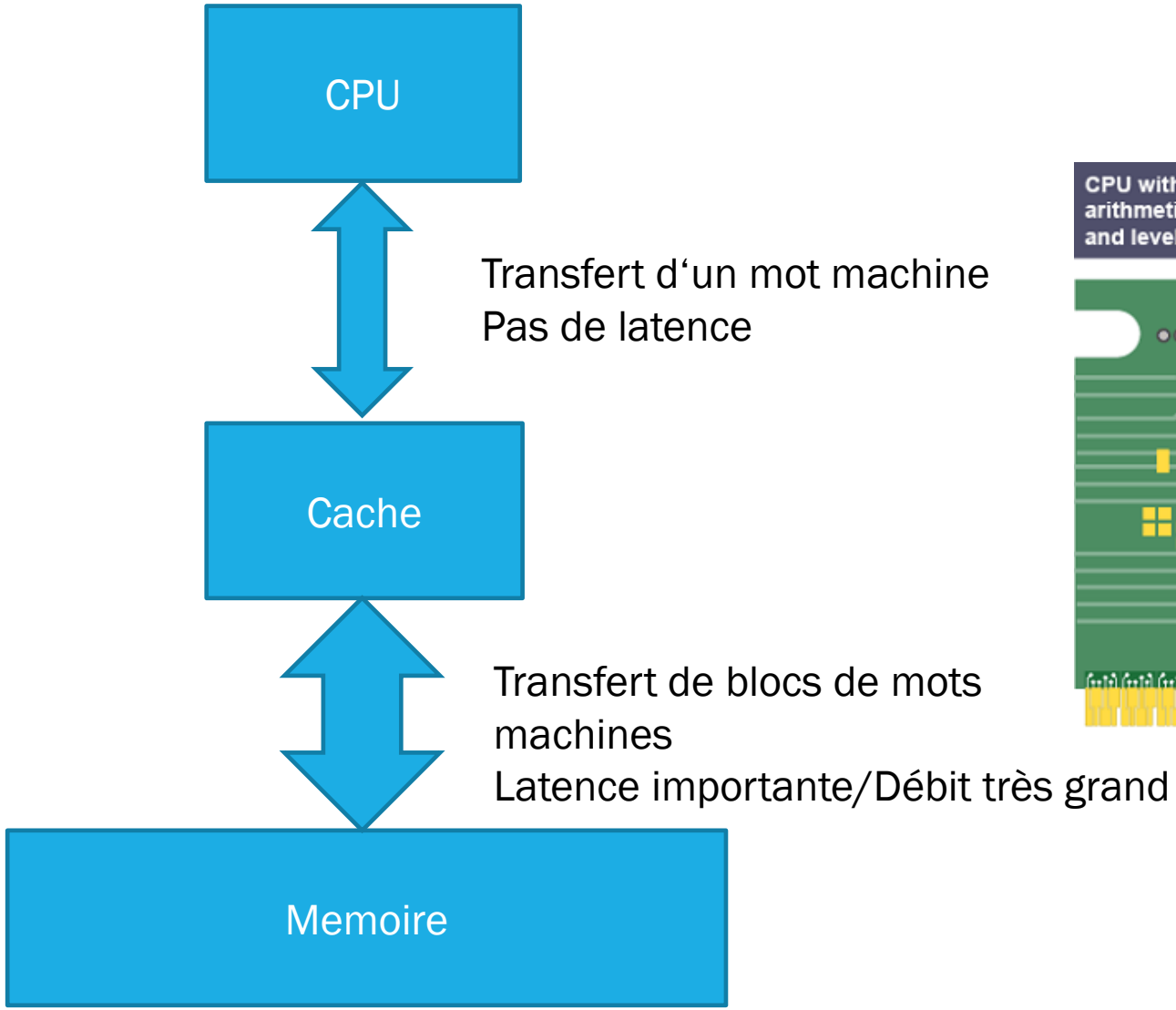
REGISTERS VS SRAM VS DRAM

- Register
  - Ensemble de bascules D
  - Coût en silicium
  - Fréquence de fonctionnement élevée
  - Latence quasi-nulle

- SRAM (Static Ram)
  - Flip/Flop (4 à 6 transistors)
  - Organisation Matricielle (Colonne/Ligne)
  - Latence faible
  - Moins dense que la DRAM et plus couteuse

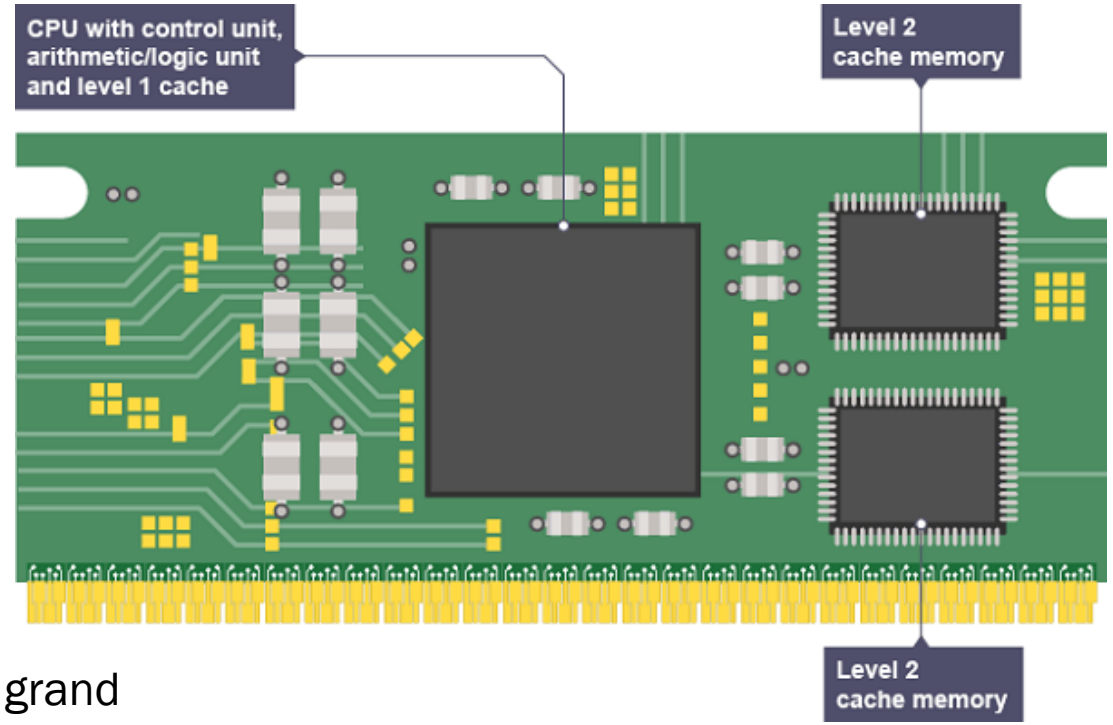
- DRAM (Dynamic RAM)
  - Capacité stockant chaque bit (1 transistor par bit)
  - Besoin de rafraichir la charge présente dans la capacité à une certaine fréquence (ms)
  - Accès multiplexé
    - Sélection de la banque
    - Sélection de la ligne
    - Sélection de la colonne
  - Latence relativement élevée
  - Débit pouvant être important

L‘INTRODUCTION DES CACHES



Transfert d‘un mot machine
Pas de latence

Transfert de blocs de mots machines
Latence importante/Débit très grand



TYPES DE CACHE

- **Fully associative**

Adresse de la donnée$_1$	Donnée$_1$
Adresse de la donnée$_2$	Donnée$_2$
Adresse de la donnée$_3$	Donnée$_3$
Adresse de la donnée$_4$	Donnée$_4$

Avantages

Très flexible

Inconvénients

Beaucoup de place occupée pour stocker les adressees

Lent pour retrouver une adresse mémoire dans la table

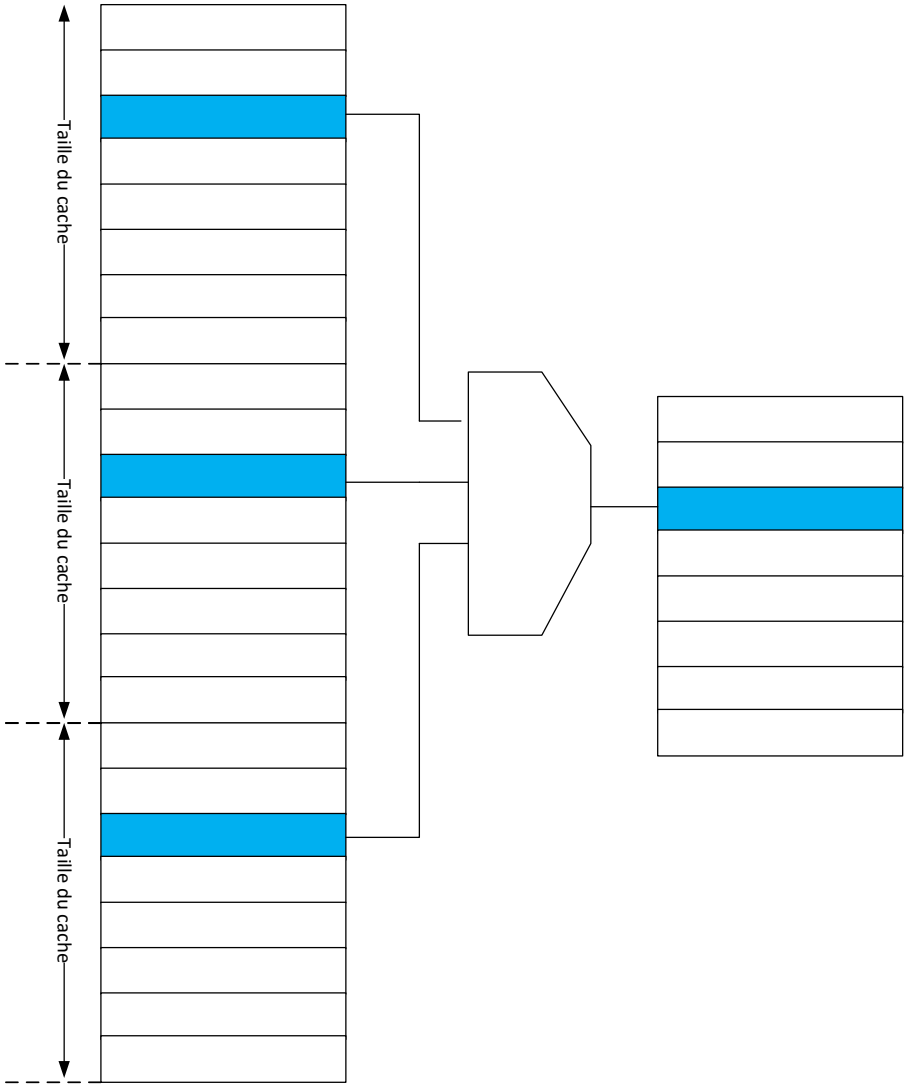
- **Direct Map Cache**

Avantages

Simplicité à mettre en oeuvre
Rapide d'accès

Inconvénients

Seule une ligne peut-être stockée dans le cache.



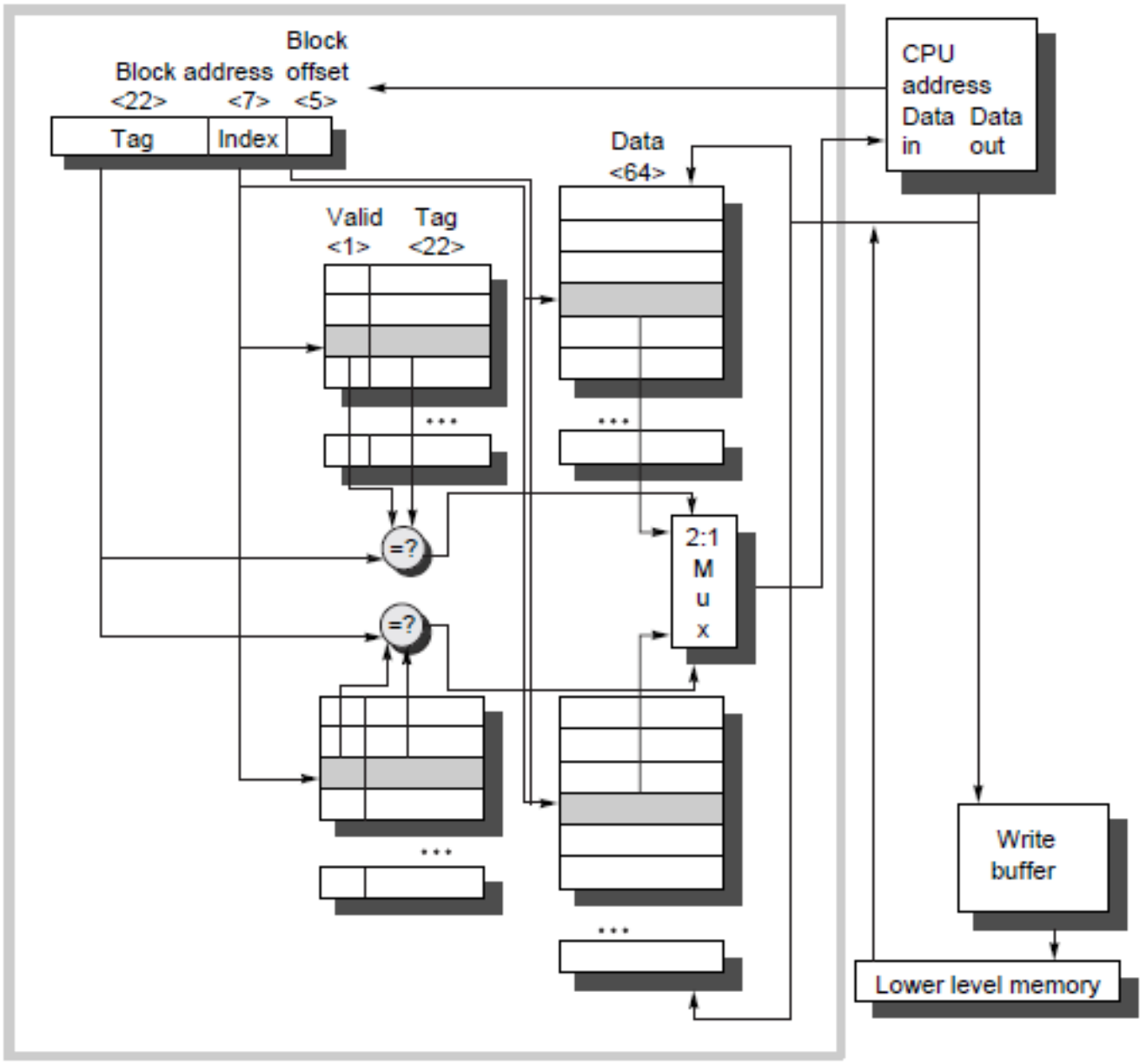
TYPES DE CACHES (2)

- X Ways Set-Associative Cache

  Idée: *mettre plusieurs „direct map cache en parallèle“*

  Avantages

  Reste simple même si plus complexe
  Quasiment aussi performant qu‘un „fully associative cache“.



GÉRER LA MISE A JOUR DES DONNÉES

- **Read**
  - Donnée présente dans le cache (Charge la donnée)
  - Donnée absente du cache (Read Miss)
- **Read Misses**

  Charge le bloc contenant les données requises
  - Sélectionne une ligne pour charger les données selon différentes stratégies:

  LRU : least recent used

  Round Robin : chaque ligne est déchargée de manière cyclique

  Random : la ligne est sélectrionnée de manière aléatoire.
  - Charge les données

  Comment limiter les „Read Miss“
  - Augmenter la taille des lignes
  - Augmenter l‘associativité

- **Write**

  Donnée présente dans le cache
  - Ecrit la donnée dans le cache
  - Si en Write-Through, la modification immédiatement faite en mémoire
  - Si en Write-Back, on indique que la ligne est modifiée et la modification en mémoire sera fait au moment où l‘on décharge la ligne?

- **Write Misses**

  Ecrit les données requises

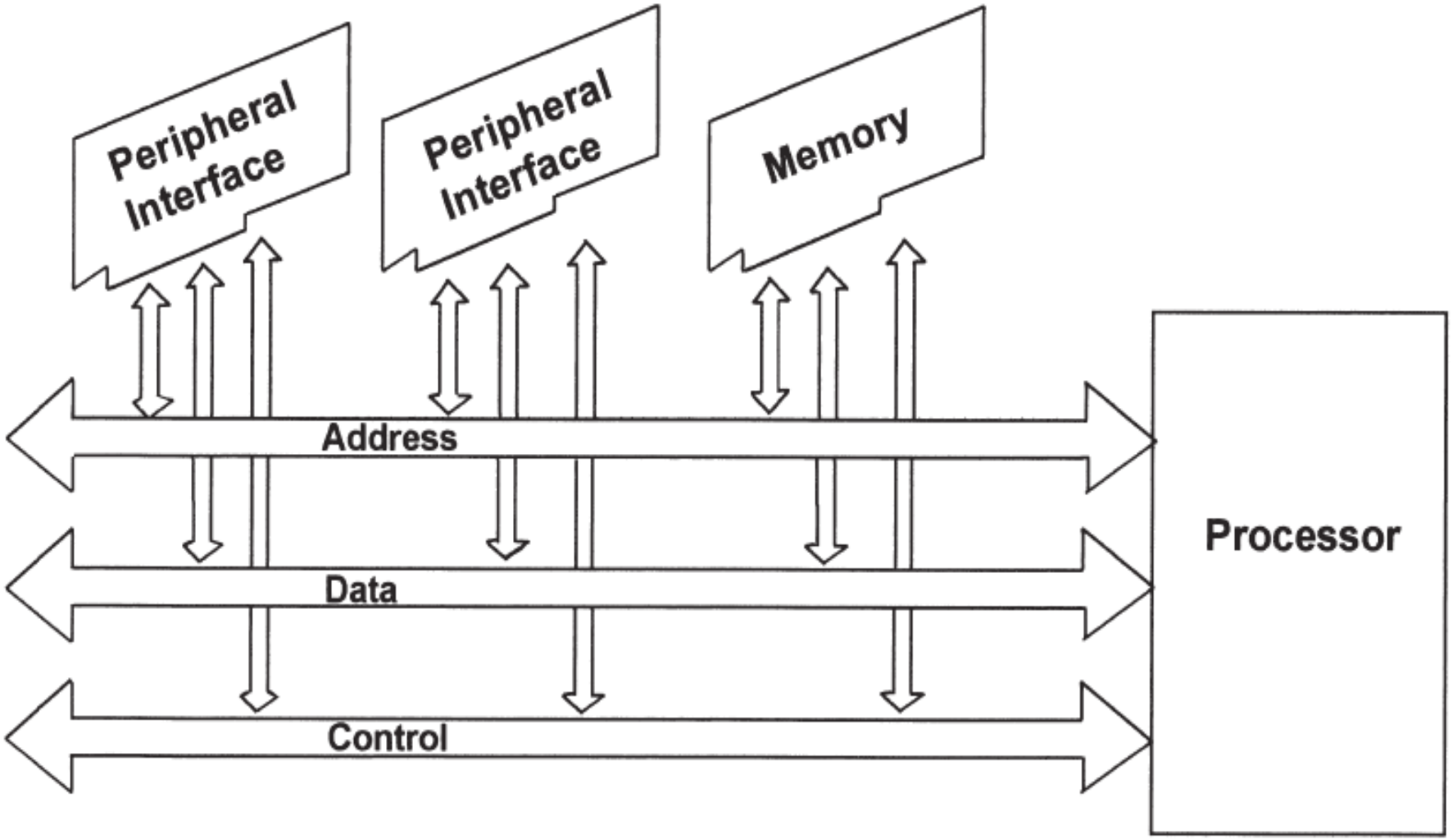
  soit sans charger les données dans le cache (Write Around),

  Soit en chargant les données à partir de la mémoire et en les modifiant

CONNECTER UN MICROPROCESSEUR À SON ENVIRONNEMENT

29.03.2022

SCHEMA FONCTIONNEL D‘UN BUS



IMPLANTATIONS ET FONCTIONALITES DES BUS

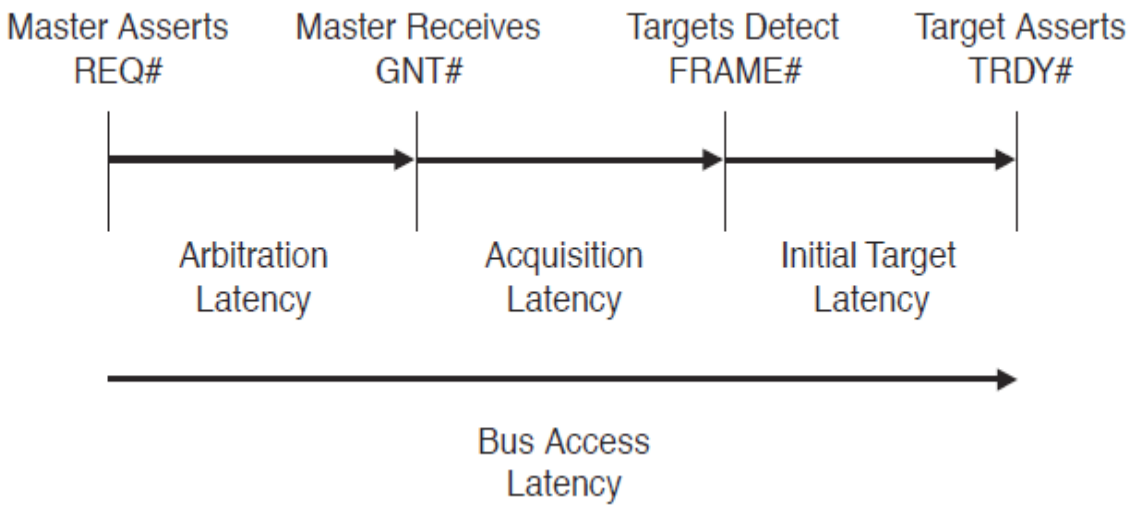
- Un Bus implante plusieurs groupes fonctionnels :
  - *Address* : Sélection du périphérique ainsi que des registres ou adresses des mémoires exposées par le périphérique.
  - *Data* : Les données qui sont échangées entre le périphérique et le processeur ou entre deux périphériques.
  - *Control* : Les commandes qui permettent de définir la nature des accès au bus ainsi que la nature des requêtes.
- Un Bus peut partager des fils entre les groupes fonctionnels
  - Bus non-multiplexé : un ensemble de fils transfère les données correspondant à un groupe fonctionnel
  - Bus multiplexé : le même ensemble de fils transfère l'ensemble des groupes fonctionnels, chaque groupe fonctionnel à son temps de parole.
- Un bus peut fournir des fonctionnalités complémentaires
  - Distribution de puissance

ARCHITECTURE DE BUS

- Architecture „Maître/Esclave“
  - Un maître coordonne l‘accès au bus (souvent le microprocesseur), les esclaves (typiquement les périphériques) répondent aux requêtes du maître.
- Plusieurs „Maitres“/Esclave
  - Concurrence entre plusieurs maître pour avoir l‘accès au bus
  - Néccesite un mécanisme pour garantir la „Fairness“
- DMA
  - Direct Memory Access : autorise une communication directe entre deux périphériquesr
  - Les bus modernes supportent plusieurs communications directes entre périphériques de manières concurrente.

LATENCE VERSUS DÉBIT

Latence d'accès au bus



- Latence faible
  - Petits messages
  - Débits plus faibles
- Latence importance
  - Grands messages,
  - Débit plus important

Impact de la taille du message sur la latence et le débit pour un bus PCI

Data Phase	Bytes Transferred	Total Clocks	Bandwith (Mb/sec)	Latency (us)
8	32	16	60	0.48
16	64	24	80	0.72
32	128	40	96	1.20
64	256	107	107	2.16

BUS HAUTE ET BASSE PERFORMANCE

High bandwidth busses

Accès aux périphériques nécessitant des fortes bandes passantes RAM

- Mémoire
- Stockage
- Co-Processors (DSP…)
- GPU

Low bandwidth busses

Accès aux périphériques communs n'ayant ni besoin de beaucoup de bandes passantes, ni besoin de fonctions avancées

- Cartes Entrées/Sorties
- Connexion Réseau
- …
- Souvent connecté par une passerelle au bus à haut debit.

Peuvent être une implantation du même bus mais avec des tailles, fréquences et technologies différentes

AMBDA/Axi pour l'architecture Arm

STBus

HIERACHIE DE BUS SUR UN SYSTÈME A BASE DE MICROPROCESSEUR