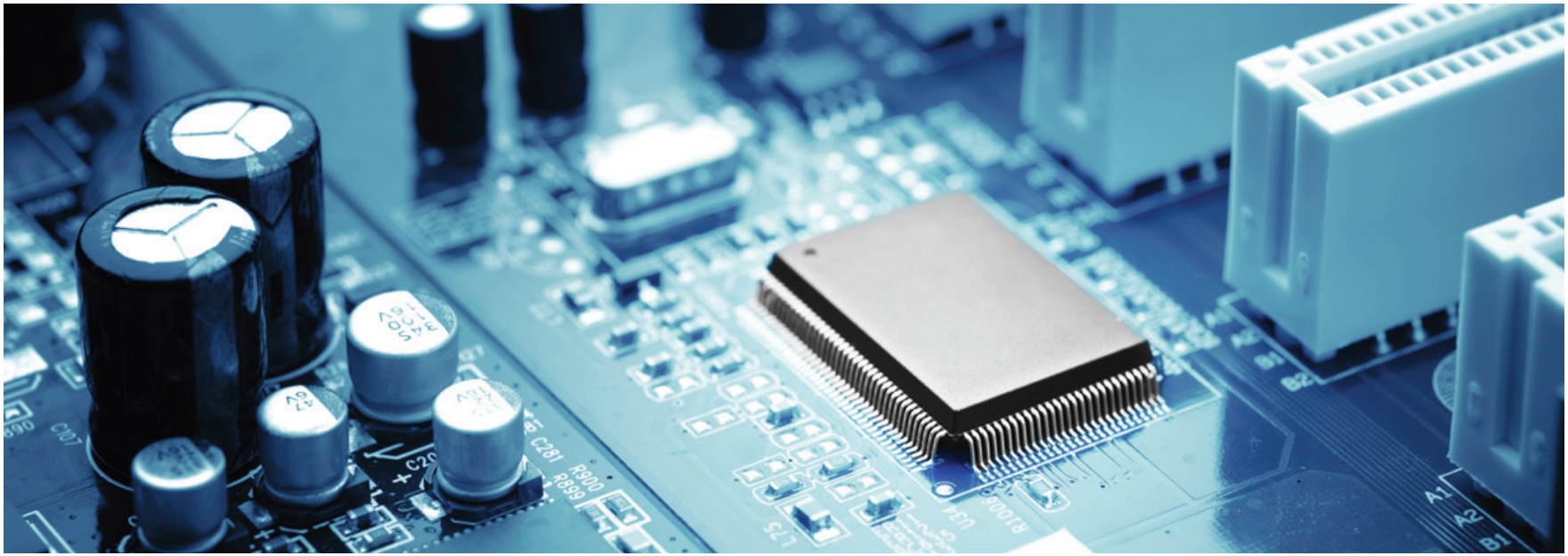
Architecture d'un processeur DSP



Filière Génie Réseaux et Télécommunications

Plan

- Le traitement numérique du signal (TNS) pourquoi ?

Plan

- Le traitement numérique du signal (TNS) pourquoi ?
- Exemples d'applications

Plan

- Le traitement numérique du signal (TNS) pourquoi ?
- Exemples d'applications
- Classification des processeurs

Plan

- Le traitement numérique du signal (TNS) pourquoi ?
- Exemples d'applications
- Classification des processeurs
- Spécificités des DSP

Plan

- Le traitement numérique du signal (TNS) pourquoi ?
- Exemples d'applications
- Classification des processeurs
- Spécificités des DSP
- Architecture générale d'un processeur

Plan

- Le traitement numérique du signal (TNS) pourquoi ?
- Exemples d'applications
- Classification des processeurs
- Spécificités des DSP
- Architecture générale d'un processeur
- Architecture d'un processeur DSP

Le traitement numérique du signal (TNS) pourquoi ?

Flexibilité

Les fonctions d'un système de TSN peuvent être facilement modifiées et mises à jour pour une utilisation sur le même matériel

Le traitement numérique du signal (TNS) pourquoi ?

Flexibilité

Les fonctions d'un système de TSN peuvent être facilement modifiées et mises à jour pour une utilisation sur le même matériel

Reproductibilité

Les performances d'un système TSN peuvent être répétées avec précision d'une unité à une autre

Le traitement numérique du signal (TNS) pourquoi ?

Flexibilité

Les fonctions d'un système de TSN peuvent être facilement modifiées et mises à jour pour une utilisation sur le même matériel

Reproductibilité

Les performances d'un système TSN peuvent être répétées avec précision d'une unité à une autre

Fiabilité

La mémoire et le système logique du matériel TSN ne se détériorent pas avec l'âge, contrairement aux composants de l'électronique analogique

Le traitement numérique du signal (TNS) pourquoi ?

Flexibilité

Les fonctions d'un système de TSN peuvent être facilement modifiées et mises à jour pour une utilisation sur le même matériel

Reproductibilité

Les performances d'un système TSN peuvent être répétées avec précision d'une unité à une autre

Fiabilité

La mémoire et le système logique du matériel TSN ne se détériorent pas avec l'âge, contrairement aux composants de l'électronique analogique

Complexité

L'utilisation du TSN permet des applications sophistiquées telles que la reconnaissance de la parole ou d'image pour des appareils portatifs légers à faible consommation

Le traitement numérique du signal (TNS) pourquoi ?

Deux types de TSN:

Le traitement numérique du signal (TNS) pourquoi ?

Deux types de TSN:

Traitement en temps non-réel

Consiste à manipuler des signaux qui ont été numérisés

Le traitement numérique du signal (TNS) pourquoi ?

Deux types de TSN:

Traitement en temps non-réel

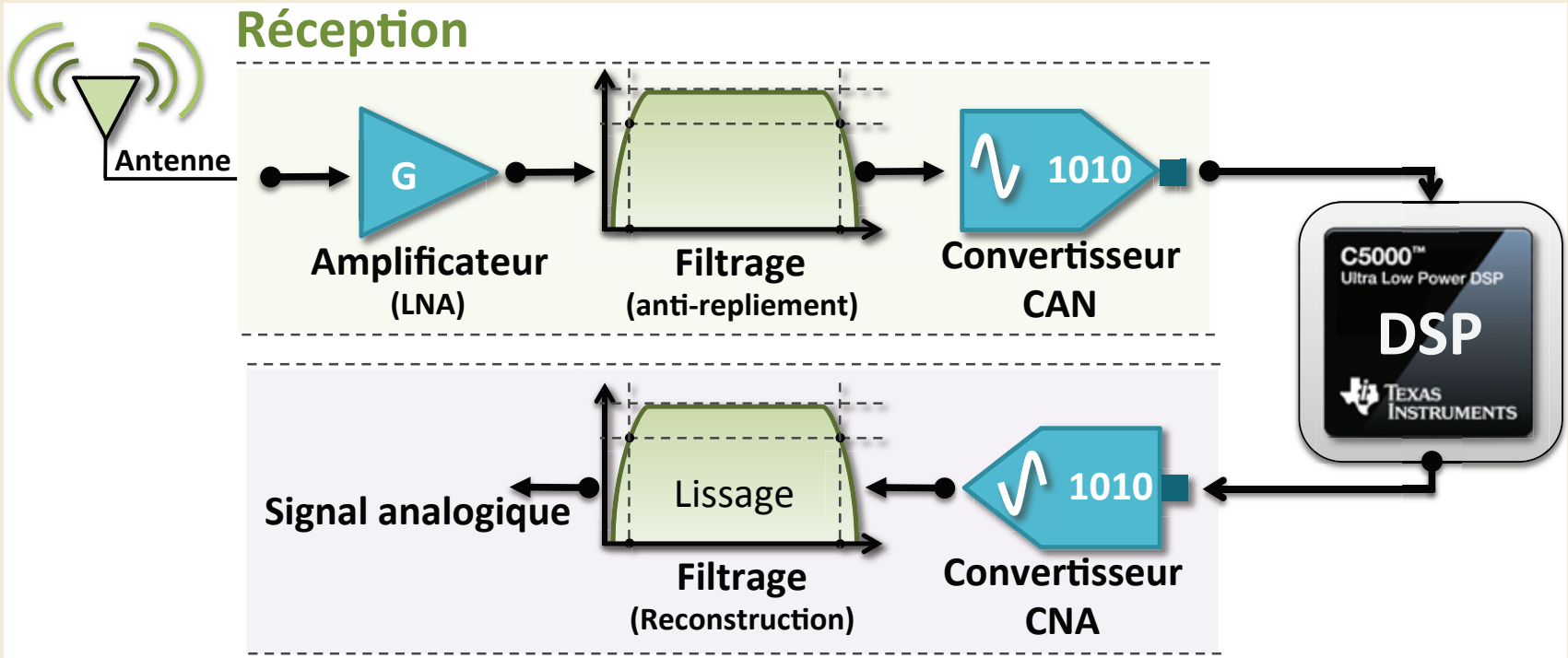
Consiste à manipuler des signaux qui ont été numérisés

Traitement en temps réel

Impose des exigences strictes sur le matériel de TSN ainsi que sur la conception du logiciel pour exécuter des tâches prédéfinies dans un certain laps de temps

Le traitement numérique du signal (TNS) pourquoi ?

Chaîne de traitement numérique



Le traitement numérique du signal (TNS) pourquoi ?

Cahier des charges

- Traitement temps réel: Latence maximale autorisée entre l'arrivée d'une donnée et la disponibilité du résultat du calcul correspondant

Le traitement numérique du signal (TNS) pourquoi ?

Cahier des charges

- Traitement temps réel: Latence maximale autorisée entre l'arrivée d'une donnée et la disponibilité du résultat du calcul correspondant
- Débits de données important: Etant donné que les signaux sont numériques, alors la quantité de données transmises séquentiellement est importante

Le traitement numérique du signal (TNS) pourquoi ?

Cahier des charges

- Traitement temps réel: Latence maximale autorisée entre l'arrivée d'une donnée et la disponibilité du résultat du calcul correspondant
- Débits de données important: Etant donné que les signaux sont numériques, alors la quantité de données transmises séquentiellement est importante
- Charge de calcul importante

Le traitement numérique du signal (TNS) pourquoi ?

Cahier des charges

- Traitement temps réel: Latence maximale autorisée entre l'arrivée d'une donnée et la disponibilité du résultat du calcul correspondant
- Débits de données important: Etant donné que les signaux sont numériques, alors la quantité de données transmises séquentiellement est importante
- Charge de calcul importante
  - Exemple de calcul: Convolution, FFT. . . etc

Le traitement numérique du signal (TNS) pourquoi ?

Cahier des charges

- Traitement temps réel: Latence maximale autorisée entre l'arrivée d'une donnée et la disponibilité du résultat du calcul correspondant
- Débits de données important: Etant donné que les signaux sont numériques, alors la quantité de données transmises séquentiellement est importante
- Charge de calcul importante
  - Exemple de calcul: Convolution, FFT. . . etc
  - Charge de calcul = Débit x complexité du calcul
- Maîtrise de la précision des calculs

Le traitement numérique du signal (TNS) pourquoi ?

Cahier des charges

- Traitement temps réel: Latence maximale autorisée entre l'arrivée d'une donnée et la disponibilité du résultat du calcul correspondant
- Débits de données important: Etant donné que les signaux sont numériques, alors la quantité de données transmises séquentiellement est importante
- Charge de calcul importante
  - Exemple de calcul: Convolution, FFT. . . etc
  - Charge de calcul = Débit x complexité du calcul
- Maîtrise de la précision des calculs
- Traitements répétitifs

Le traitement numérique du signal (TNS) pourquoi ?

Opérations classiques en TNS

- MAC: Multiplication-Accumulation

Le traitement numérique du signal (TNS) pourquoi ?

Opérations classiques en TNS

- MAC: Multiplication-Accumulation
  - acc $\longleftarrow$ $acc + b_i x_i$

Le traitement numérique du signal (TNS) pourquoi ?

Opérations classiques en TNS

- MAC: Multiplication-Accumulation
  - acc $\longleftarrow$ $acc + b_i x_i$
  - Produit scalaire

Le traitement numérique du signal (TNS) pourquoi ?

Opérations classiques en TNS

- MAC: Multiplication-Accumulation
  - acc $\longleftarrow$ $acc + b_i x_i$
  - Produit scalaire
  - Produit de convolution

Le traitement numérique du signal (TNS) pourquoi ?

Opérations classiques en TNS

- MAC: Multiplication-Accumulation
  - acc $\longleftarrow$ $acc + b_i x_i$
  - Produit scalaire
  - Produit de convolution
  - Filtre à réponse impulsionelle finie FIR
    $y[n] = \sum_{i=0}^{N-1} b[i]x[n-i]$

Le traitement numérique du signal (TNS) pourquoi ?

Opérations classiques en TNS

- MAC: Multiplication-Accumulation
  - acc $\longleftarrow acc + b_i x_i$
  - Produit scalaire
  - Produit de convolution
  - Filtre à réponse impulsionelle finie FIR
    $y[n] = \sum_{i=0}^{N-1} b[i]x[n-i]$
  - Filtre à réponse impulsionelle infinie IIR
    $y[n] = \sum_{i=0}^{N-1} b[i]x[n-i] + \sum_{j=1}^{M-1} a[j]y[n-j]$

Le traitement numérique du signal (TNS) pourquoi ?

Opérations classiques en TNS

- MAC: Multiplication-Accumulation
  - acc $\longleftarrow acc + b_i x_i$
  - Produit scalaire
  - Produit de convolution
  - Filtre à réponse impulsionelle finie FIR
    $y[n] = \sum_{i=0}^{N-1} b[i]x[n-i]$
  - Filtre à réponse impulsionelle infinie IIR
    $y[n] = \sum_{i=0}^{N-1} b[i]x[n-i] + \sum_{j=1}^{M-1} a[j]y[n-j]$
- FFT: Analyse spectrale

Le traitement numérique du signal (TNS) pourquoi ?

Opérations classiques en TNS

- MAC: Multiplication-Accumulation
  - acc $\longleftarrow acc + b_i x_i$
  - Produit scalaire
  - Produit de convolution
  - Filtre à réponse impulsionelle finie FIR
    $y[n] = \sum_{i=0}^{N-1} b[i]x[n-i]$
  - Filtre à réponse impulsionelle infinie IIR
    $y[n] = \sum_{i=0}^{N-1} b[i]x[n-i] + \sum_{j=1}^{M-1} a[j]y[n-j]$
- FFT: Analyse spectrale
- DCT: Compression d'images

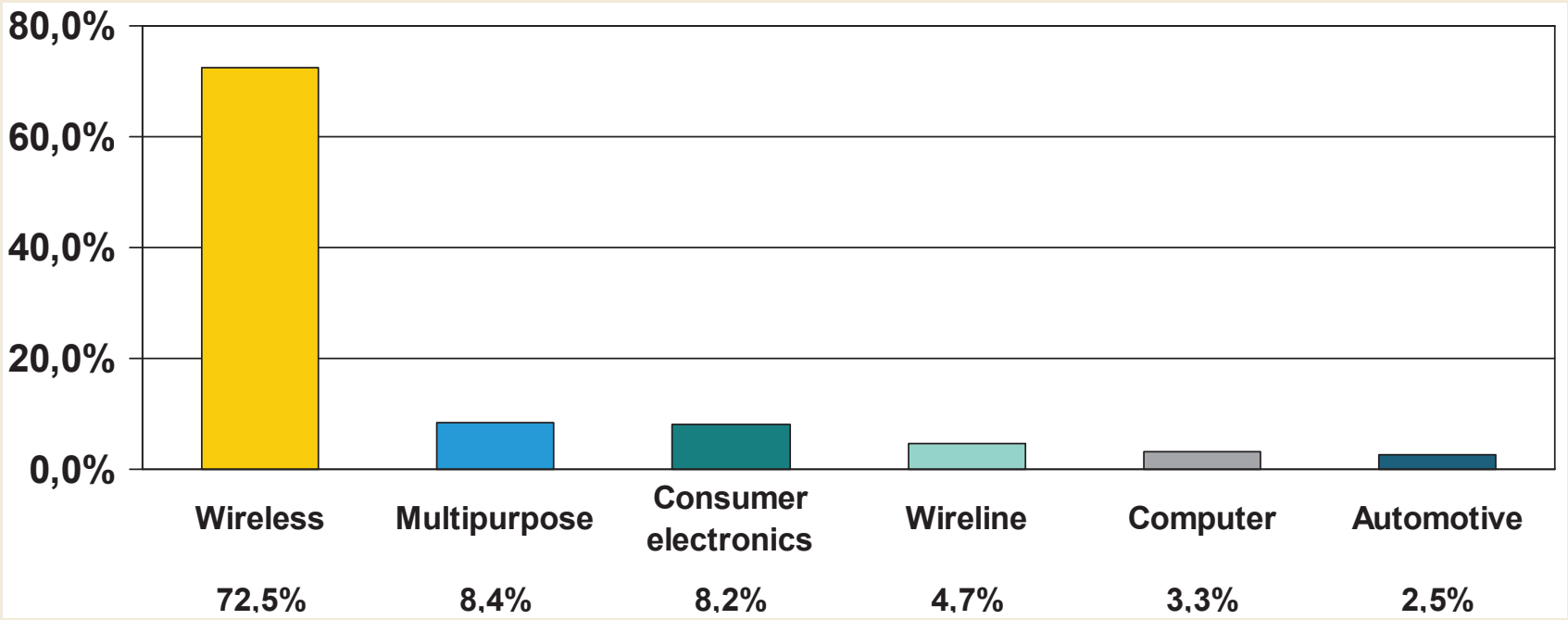
Le traitement numérique du signal (TNS) pourquoi ?

Opérations classiques en TNS

- MAC: Multiplication-Accumulation
  - acc $\longleftarrow$ $acc + b_i x_i$
  - Produit scalaire
  - Produit de convolution
  - Filtre à réponse impulsionelle finie FIR
    $y[n] = \sum_{i=0}^{N-1} b[i]x[n-i]$
  - Filtre à réponse impulsionelle infinie IIR
    $y[n] = \sum_{i=0}^{N-1} b[i]x[n-i] + \sum_{j=1}^{M-1} a[j]y[n-j]$
- FFT: Analyse spectrale
- DCT: Compression d'images
- Génération de formes d'ondes: Communications numériques

Le traitement numérique du signal (TNS) pourquoi ?

Parts de marché vente DSP programmables (2006)



1

[1] **Source**: *Will Strauss*, Forward Concepts, publié dans *Programmable Logic 2007*

Exemples d'applications

Communications

- Software Defined Radio (SDR)

Exemples d'applications

Communications

- Software Defined Radio (SDR)
- Sans fil: Cellulaires, télévision numérique, radio numérique

Exemples d'applications

Communications

- Software Defined Radio (SDR)
- Sans fil: Cellulaires, télévision numérique, radio numérique
- VoIP

Exemples d'applications

Communications

- Software Defined Radio (SDR)
- Sans fil: Cellulaires, télévision numérique, radio numérique
- VoIP

Audio

- Mixage et édition

Exemples d'applications

Communications

- Software Defined Radio (SDR)
- Sans fil: Cellulaires, télévision numérique, radio numérique
- VoIP

Audio

- Mixage et édition
- Annuleur d'echo

Exemples d'applications

Communications

- Software Defined Radio (SDR)
- Sans fil: Cellulaires, télévision numérique, radio numérique
- VoIP

Audio

- Mixage et édition
- Annuleur d'echo

Image/vidéo

- Compression/Codage

Exemples d'applications

Communications

- Software Defined Radio (SDR)
- Sans fil: Cellulaires, télévision numérique, radio numérique
- VoIP

Audio

- Mixage et édition
- Annuleur d'echo

Image/vidéo

- Compression/Codage
- Vidéosurveillance

Exemples d'applications

Communications

- Software Defined Radio (SDR)
- Sans fil: Cellulaires, télévision numérique, radio numérique
- VoIP

Audio

- Mixage et édition
- Annuleur d'echo

Image/vidéo

- Compression/Codage
- Vidéosurveillance

Militaire

- Cryptographie

Exemples d'applications

Communications

- Software Defined Radio (SDR)
- Sans fil: Cellulaires, télévision numérique, radio numérique
- VoIP

Audio

- Mixage et édition
- Annuleur d'echo

Image/vidéo

- Compression/Codage
- Vidéosurveillance

Militaire

- Cryptographie
- GPS, Radar, Sonar

Exemples d'applications

Biomédical

- Electro-EncéphaloGramme (EEG)

Exemples d'applications

Biomédical

- Electro-EncéphaloGramme (EEG)
- Electro-CardioGramme (ECG)

Exemples d'applications

Biomédical

- Electro-EncéphaloGramme (EEG)
- Electro-CardioGramme (ECG)

Instrumentation

- Analyseurs de spectre

Exemples d'applications

Biomédical

- Electro-EncéphaloGramme (EEG)
- Electro-CardioGramme (ECG)

Instrumentation

- Analyseurs de spectre
- Générations de fonctions

Exemples d'applications

Biomédical

- Electro-EncéphaloGramme (EEG)
- Electro-CardioGramme (ECG)

Instrumentation

- Analyseurs de spectre
- Générations de fonctions

Automatisation

- Commande de machines

Exemples d'applications

Biomédical

- Electro-EncéphaloGramme (EEG)
- Electro-CardioGramme (ECG)

Instrumentation

- Analyseurs de spectre
- Générations de fonctions

Automatisation

- Commande de machines
- Contrôle de moteurs

Exemples d'applications

Biomédical

- Electro-EncéphaloGramme (EEG)
- Electro-CardioGramme (ECG)

Instrumentation

- Analyseurs de spectre
- Générations de fonctions

Automatisation

- Commande de machines
- Contrôle de moteurs
- Robots

Exemples d'applications

Biomédical

- Electro-EncéphaloGramme (EEG)
- Electro-CardioGramme (ECG)

Instrumentation

- Analyseurs de spectre
- Générations de fonctions

Automatisation

- Commande de machines
- Contrôle de moteurs
- Robots

Electronique Automobile

- Assistance au freinage

Exemples d'applications

Biomédical

- Electro-EncéphaloGramme (EEG)
- Electro-CardioGramme (ECG)

Instrumentation

- Analyseurs de spectre
- Générations de fonctions

Automatisation

- Commande de machines
- Contrôle de moteurs
- Robots

Electronique Automobile

- Assistance au freinage
- Commandes vocales

Classification des processeurs/Famille TMS320 Texas Instruments

Gamme C2000: Contrôle des systèmes

C2000

- Contrôle des moteurs

Classification des processeurs/Famille TMS320 Texas Instruments

Gamme C2000: Contrôle des systèmes

C2000

- Contrôle des moteurs
- Contrôle numérique des systèmes (DCS)

Classification des processeurs/Famille TMS320 Texas Instruments

Gamme C5000: Télécommunication: Clients

C5000

- Téléphones portables

Classification des processeurs/Famille TMS320 Texas Instruments

Gamme C5000: Télécommunication: Clients

C5000

- Téléphones portables
- Caméras numériques

Classification des processeurs/Famille TMS320 Texas Instruments

Gamme C5000: Télécommunication: Clients

C5000

- Téléphones portables
- Caméras numériques
- VoIP

Classification des processeurs/Famille TMS320 Texas Instruments

Gamme C6000: Télécommunication: Infrastructures

C6000

- Stations de base

Classification des processeurs/Famille TMS320 Texas Instruments

Gamme C6000: Télécommunication: Infrastructures

C6000

- Stations de base
- Serveurs multimédias

Spécificités des DSP

Un processeur DSP est préférable pour:

- Minimiser la taille

Spécificités des DSP

Un processeur DSP est préférable pour:

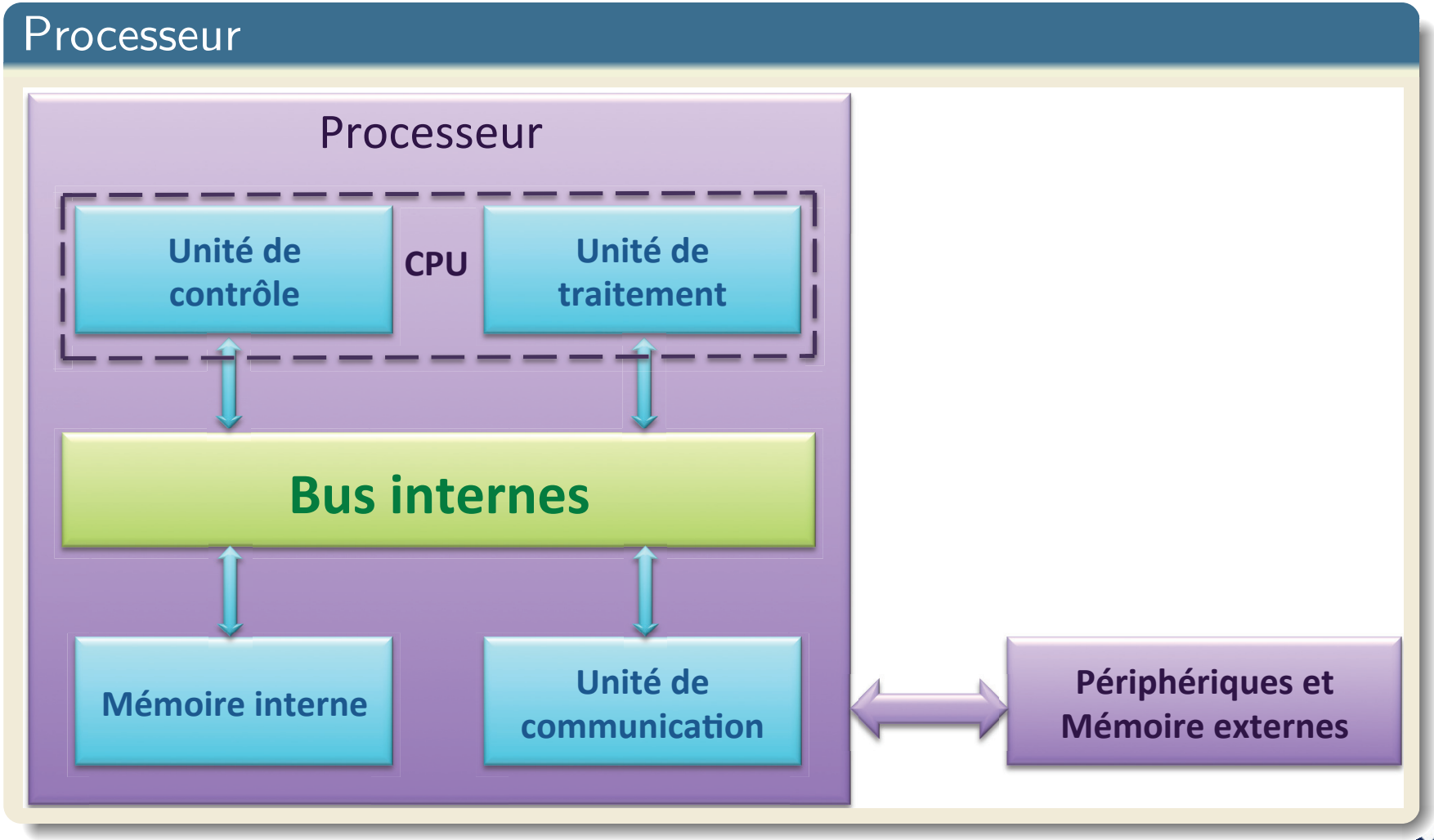
- Minimiser la taille
- Minimiser la consommation

Spécificités des DSP

Un processeur DSP est préférable pour:

- Minimiser la taille
- Minimiser la consommation
- Traitement temps-réel à grand débit

Architecture générale d'un processeur



Accès à la mémoire

Bus

- Bus adresse: Sélection du périphérique et d'une donnée

Accès à la mémoire

Bus

- Bus adresse: Sélection du périphérique et d'une donnée
- Bus donnée: valeur de la donnée à échanger

Accès à la mémoire

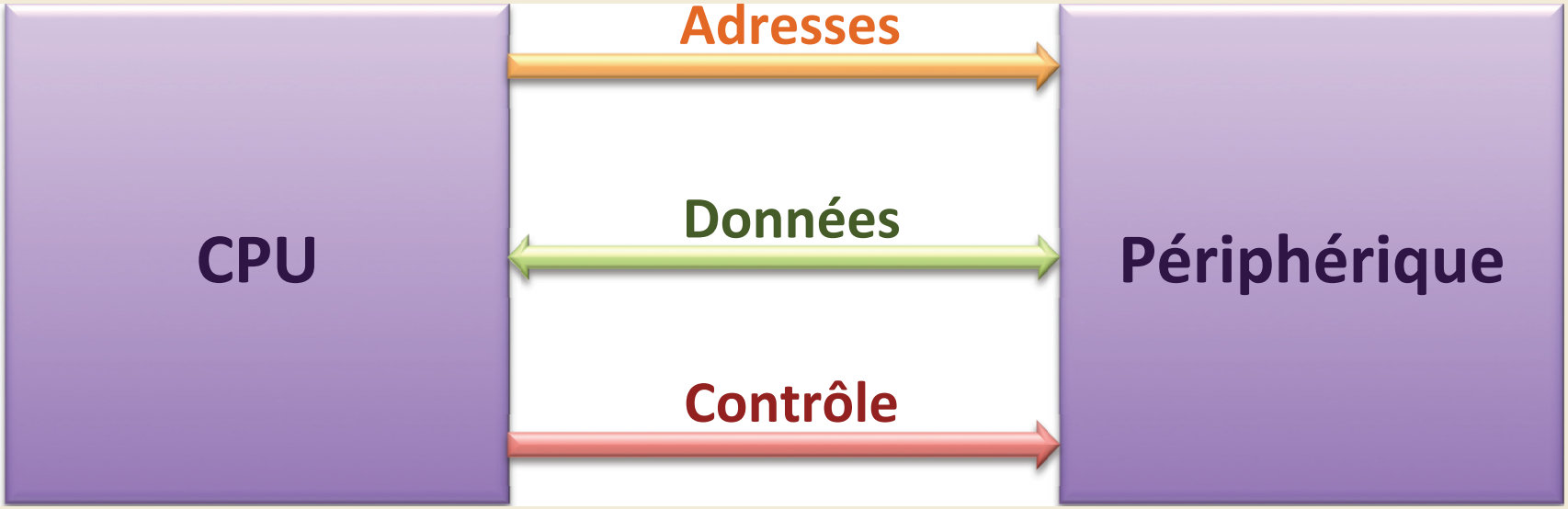
Bus

- Bus adresse: Sélection du périphérique et d'une donnée
- Bus donnée: valeur de la donnée à échanger
- Bus contrôle: lire ou écrire une donnée

Accès à la mémoire

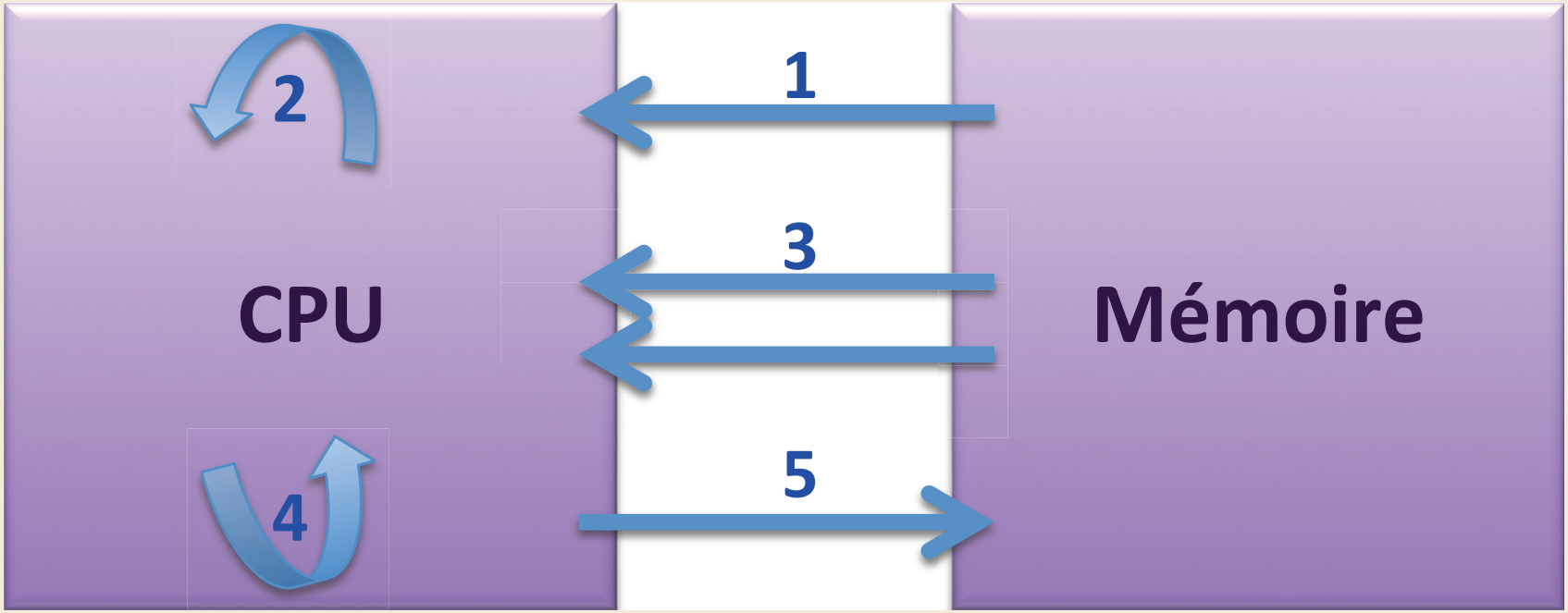
Bus

- Bus adresse: Sélection du périphérique et d'une donnée
- Bus donnée: valeur de la donnée à échanger
- Bus contrôle: lire ou écrire une donnée



Architecture générale d'un processeur

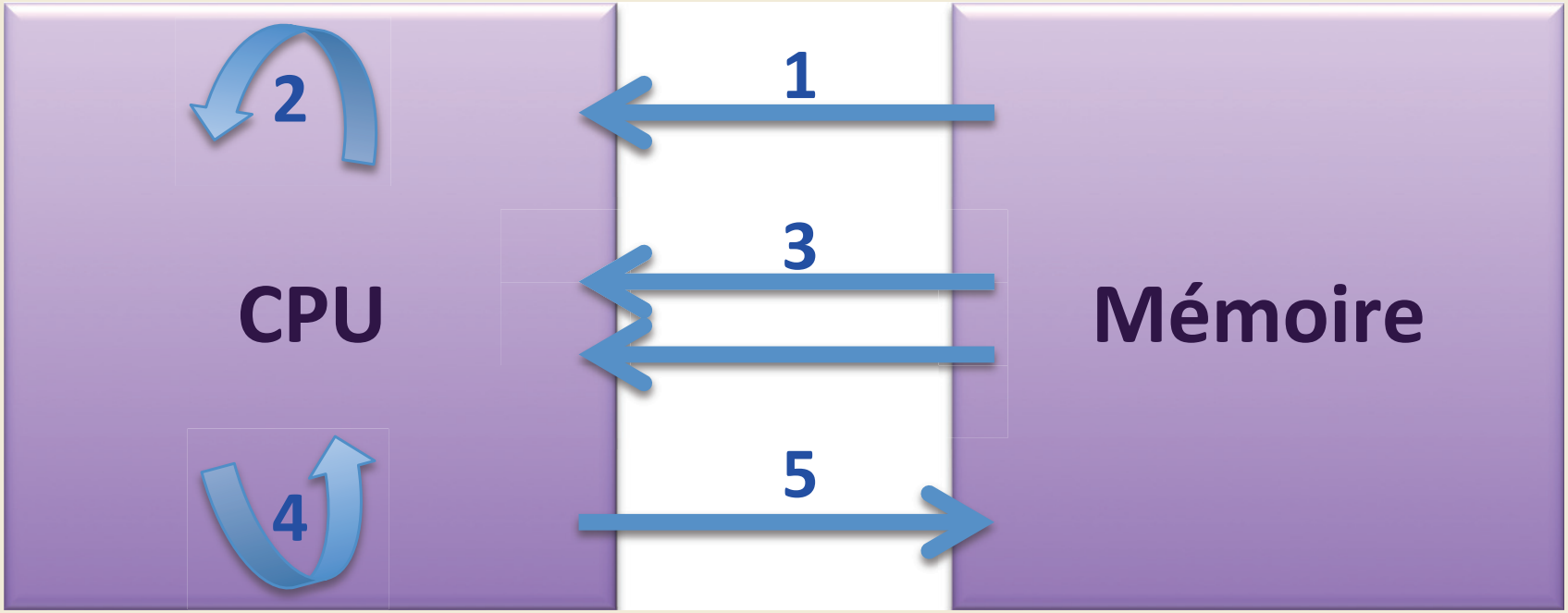
Etapes d'une opération de calcul



1. Charger une instruction

Architecture générale d'un processeur

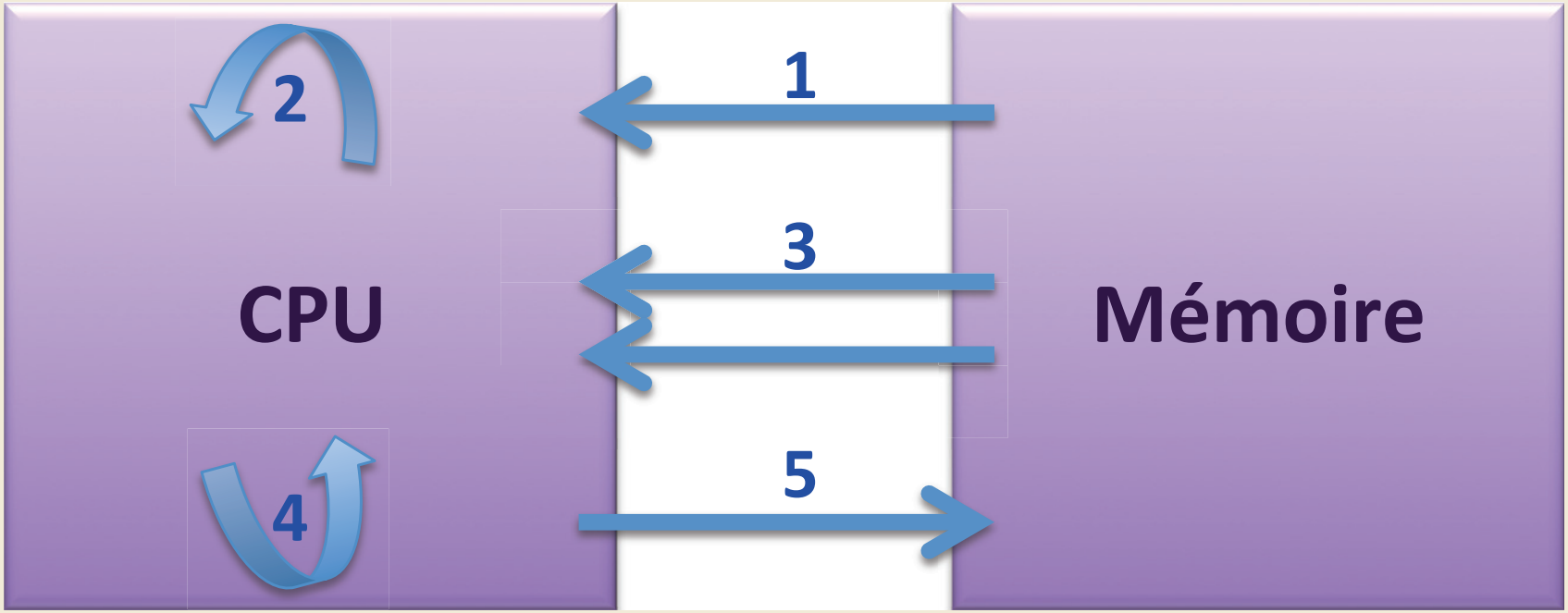
Etapes d'une opération de calcul



1. Charger une instruction
2. Décoder l'instruction

Architecture générale d'un processeur

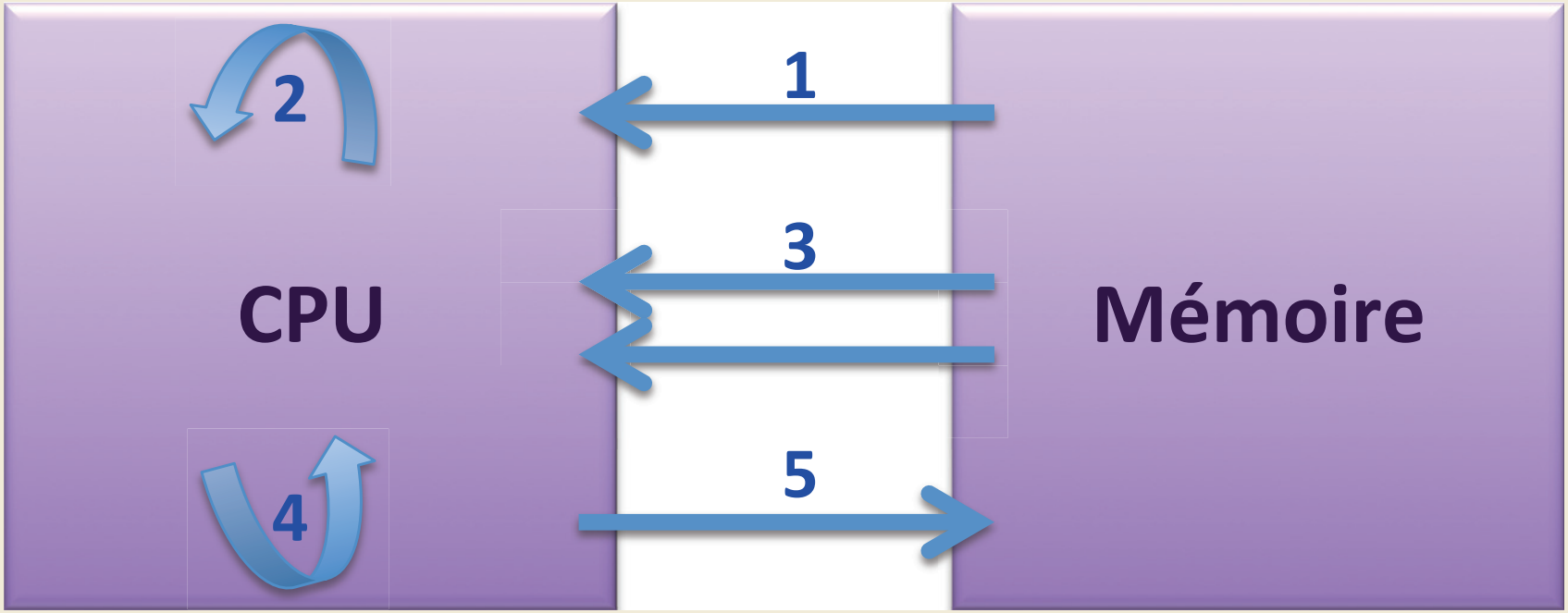
Etapes d'une opération de calcul



1. Charger une instruction
2. Décoder l'instruction
3. Charger les opérandes

Architecture générale d'un processeur

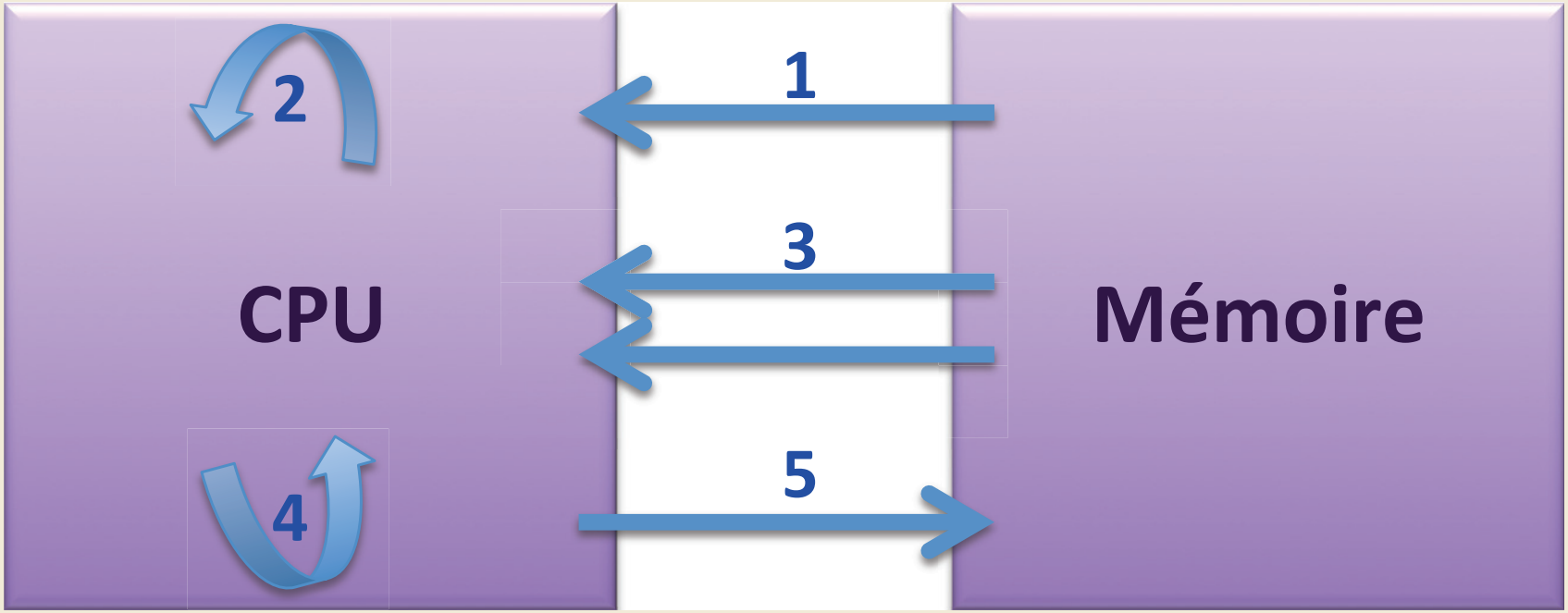
Etapes d'une opération de calcul



1. Charger une instruction
2. Décoder l'instruction
3. Charger les opérandes
4. Effectuer les calculs

Architecture générale d'un processeur

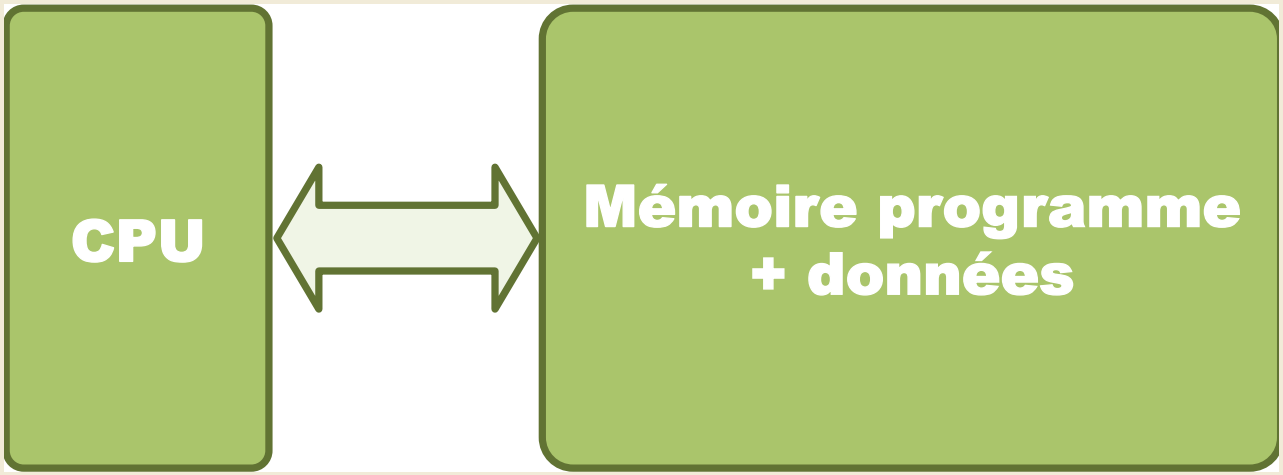
Etapes d'une opération de calcul



1. Charger une instruction
2. Décoder l'instruction
3. Charger les opérandes
4. Effectuer les calculs
5. Stocker le résultat

Architecture générale d'un processeur

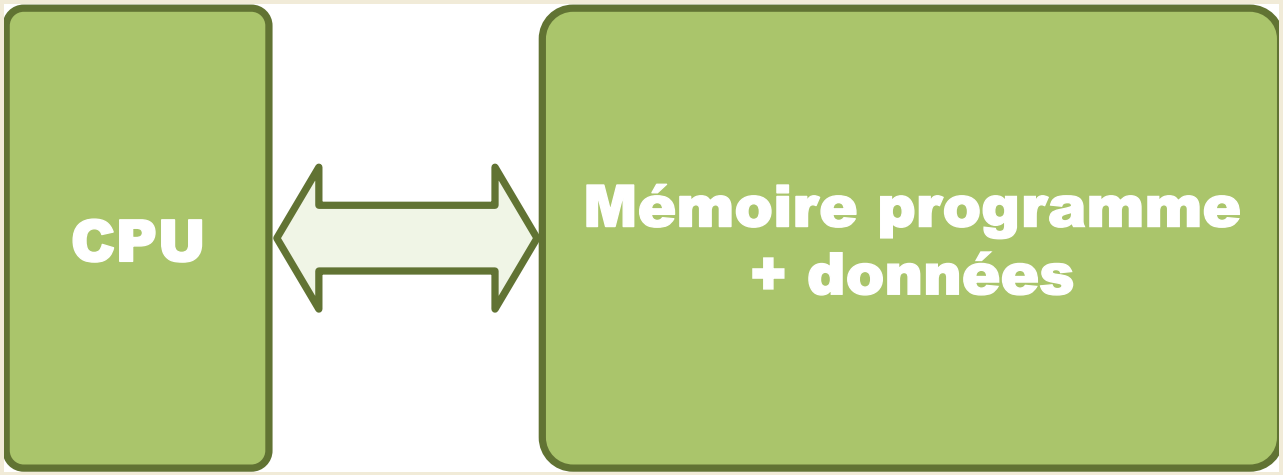
Architecture de Von Neuman



- Un seul chemin d'accès à la mémoire

Architecture générale d'un processeur

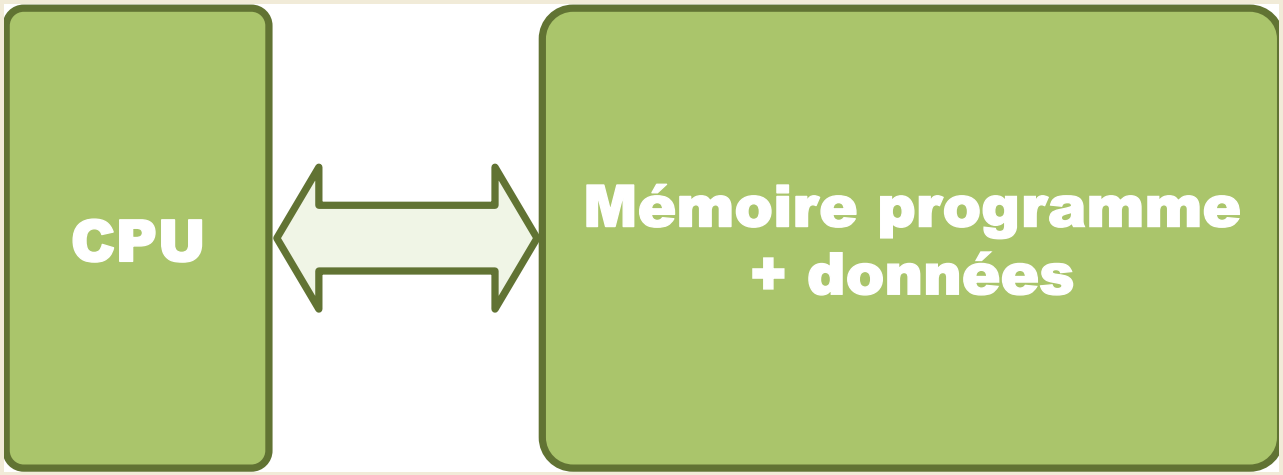
Architecture de Von Neuman



- Un seul chemin d'accès à la mémoire
- Architecture des processeurs d'usage général (Pentium, 68000)

Architecture générale d'un processeur

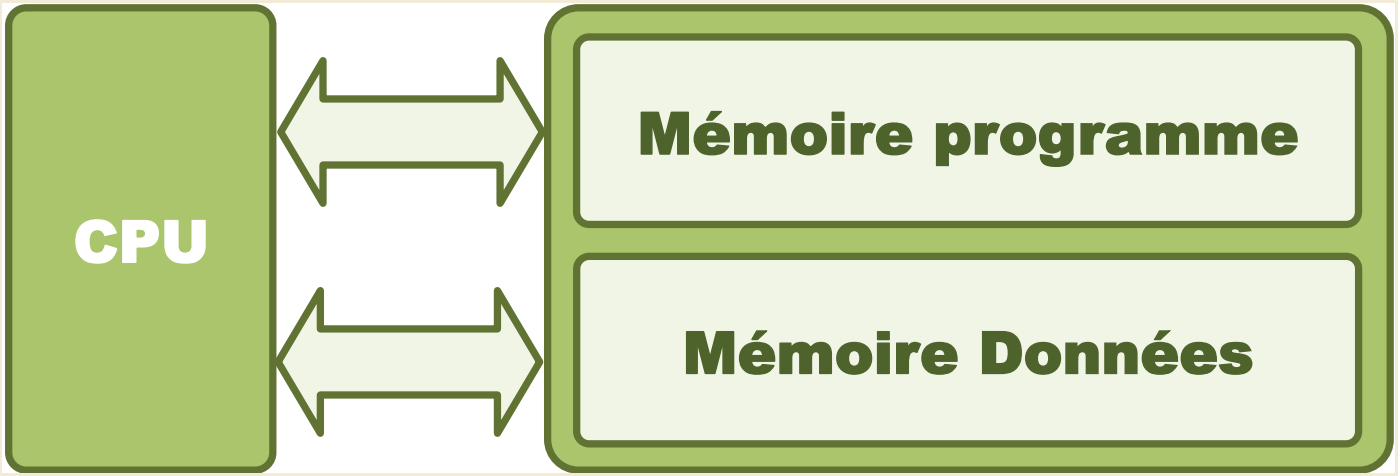
Architecture de Von Neuman



- Un seul chemin d'accès à la mémoire
- Architecture des processeurs d'usage général (Pentium, 68000)
- Pas de sécurisation matérielle du programme

Architecture générale d'un processeur

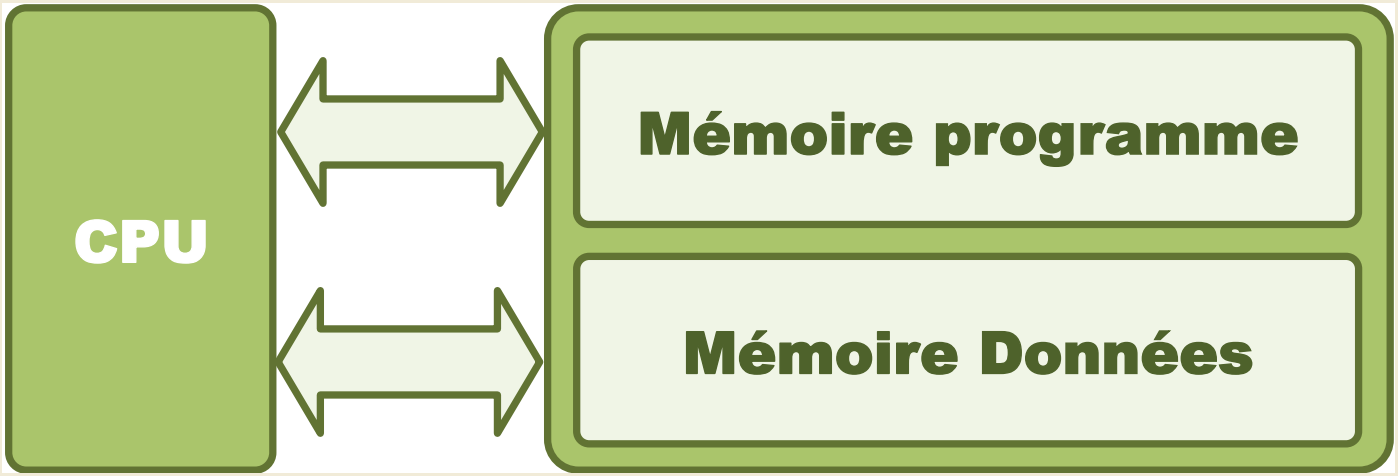
Architecture de Harvard



- Séparation des mémoires programme et données

Architecture générale d'un processeur

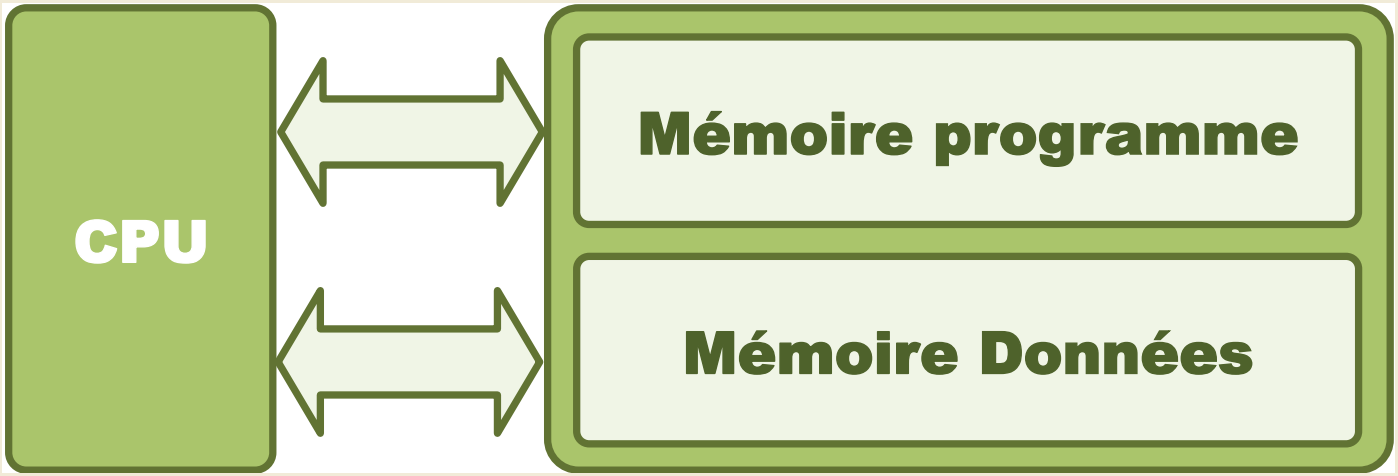
Architecture de Harvard



- Séparation des mémoires programme et données
- Moins de risque de corruption du programme

Architecture générale d'un processeur

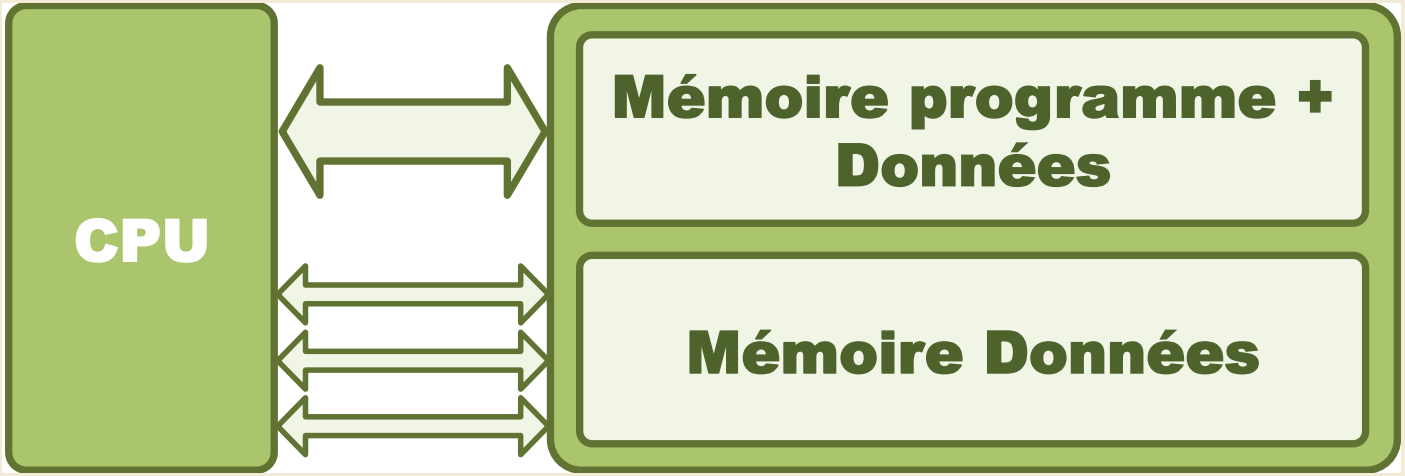
Architecture de Harvard



- Séparation des mémoires programme et données
- Moins de risque de corruption du programme
- Meilleure utilisation du CPU: Chargement du programme et des données en parallèle

Architecture générale d'un processeur

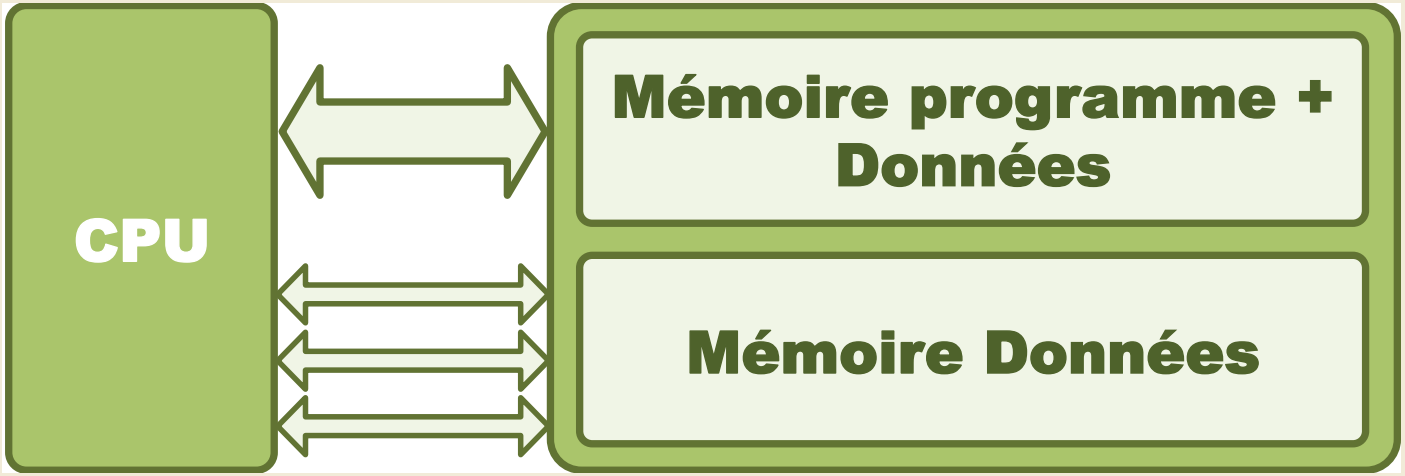
Accès mémoire multi-port



- Plusieurs bus de données

Architecture générale d'un processeur

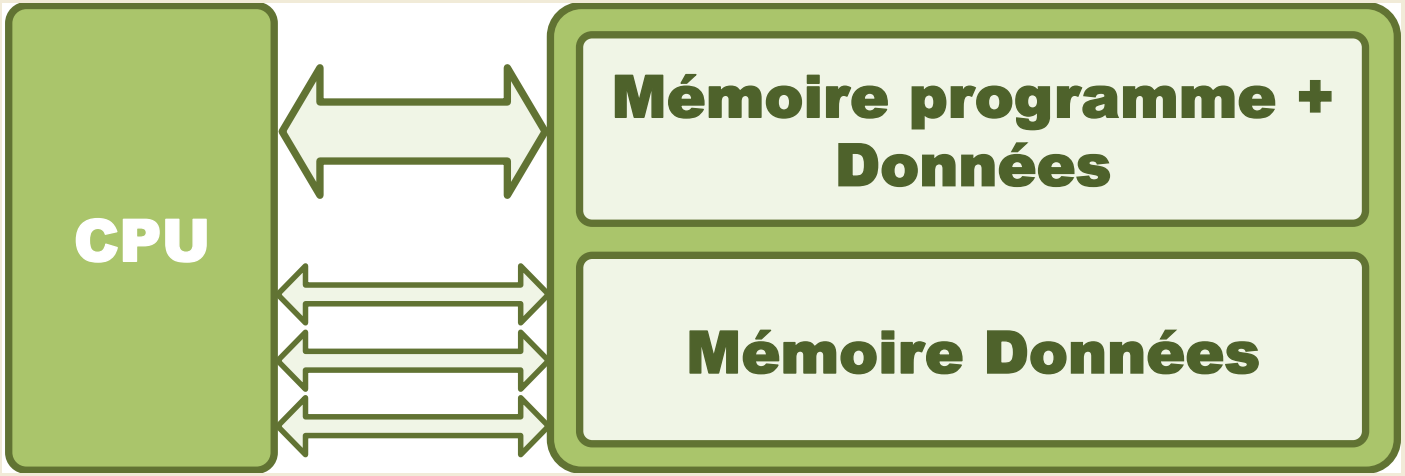
Accès mémoire multi-port



- Plusieurs bus de données
- Moins de risque de corruption du programme

Architecture générale d'un processeur

Accès mémoire multi-port

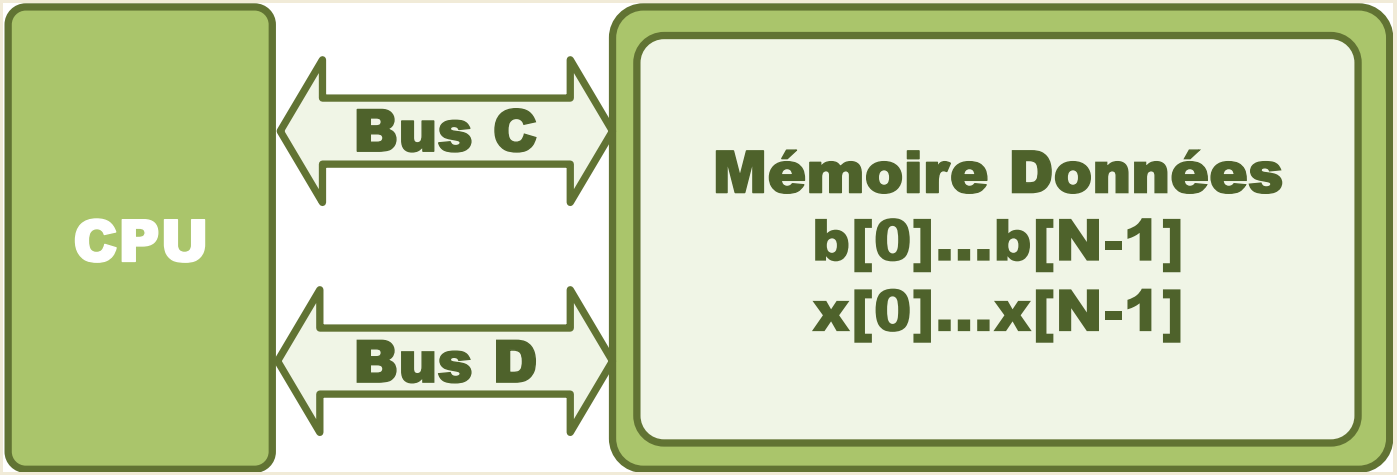


- Plusieurs bus de données
- Moins de risque de corruption du programme
- Meilleure utilisation du CPU: Chargement du programme et des données en parallèle

Architecture générale d'un processeur

Accès mémoire multi-port

Exemple

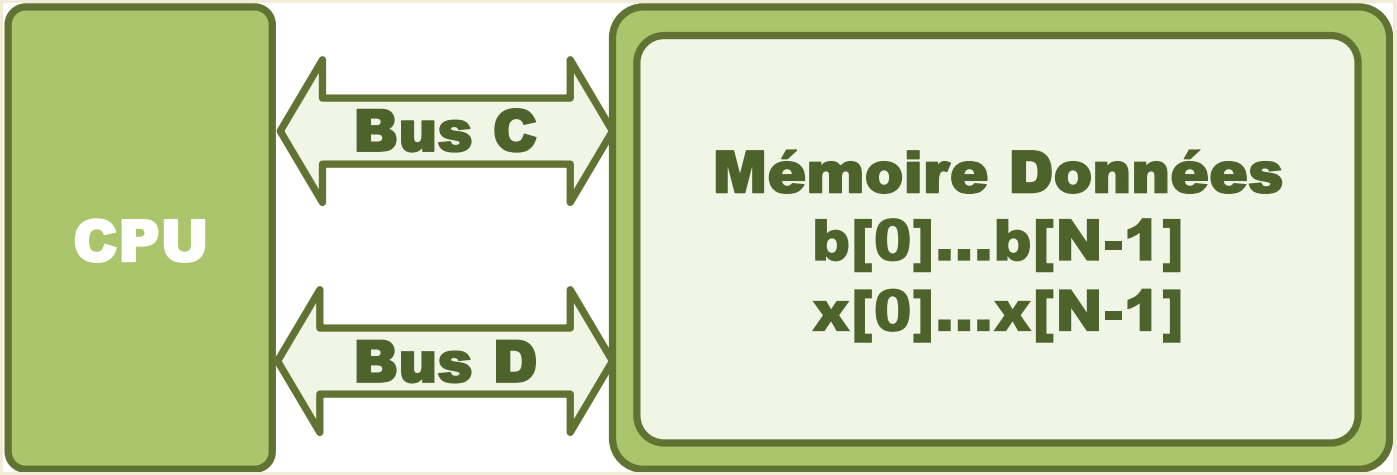


- Pour l'opération $y[n] \leftarrow b[n] * x[n]$

Architecture générale d'un processeur

Accès mémoire multi-port

Exemple

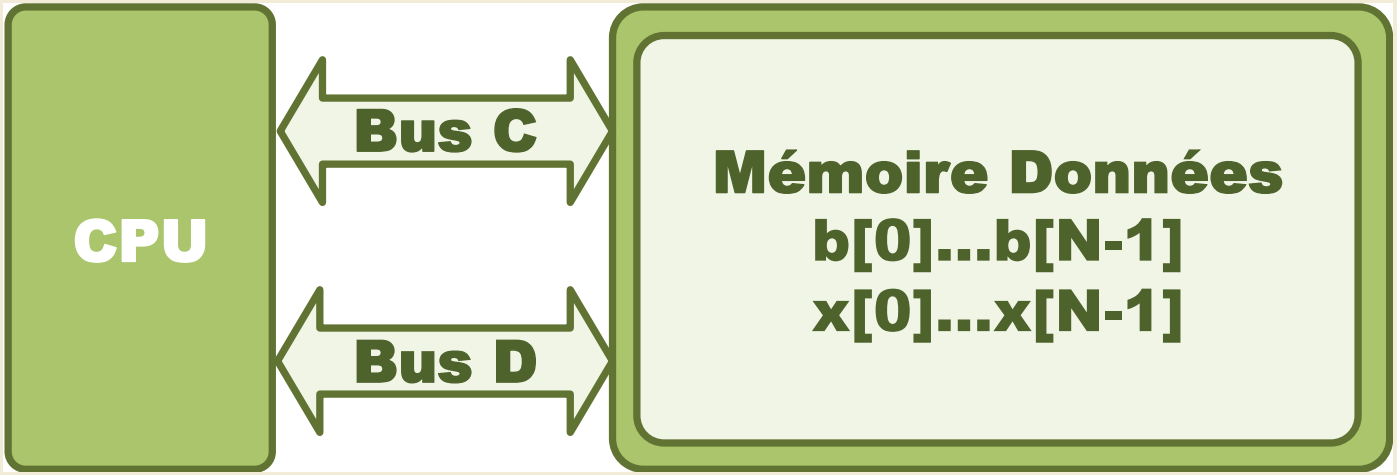


- Pour l'opération $y[n] \leftarrow b[n] * x[n]$
- On souhaite récupérer $b[n]$ et $x[n]$

Architecture générale d'un processeur

Accès mémoire multi-port

Exemple

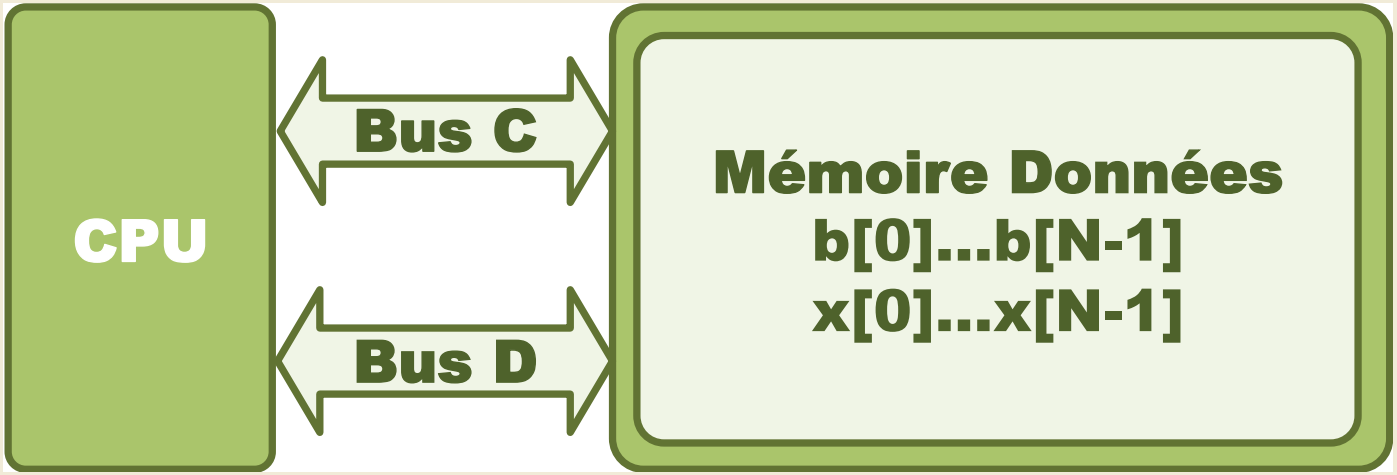


- Pour l'opération $y[n] \leftarrow b[n] * x[n]$
- On souhaite récupérer $b[n]$ et $x[n]$
- Chaque bus est capable de transmettre 1 donnée par cycle
- Mémoire à Simple Accès (SARAM) $\Rightarrow$ 2 cycles

Architecture générale d'un processeur

Accès mémoire multi-port

Exemple

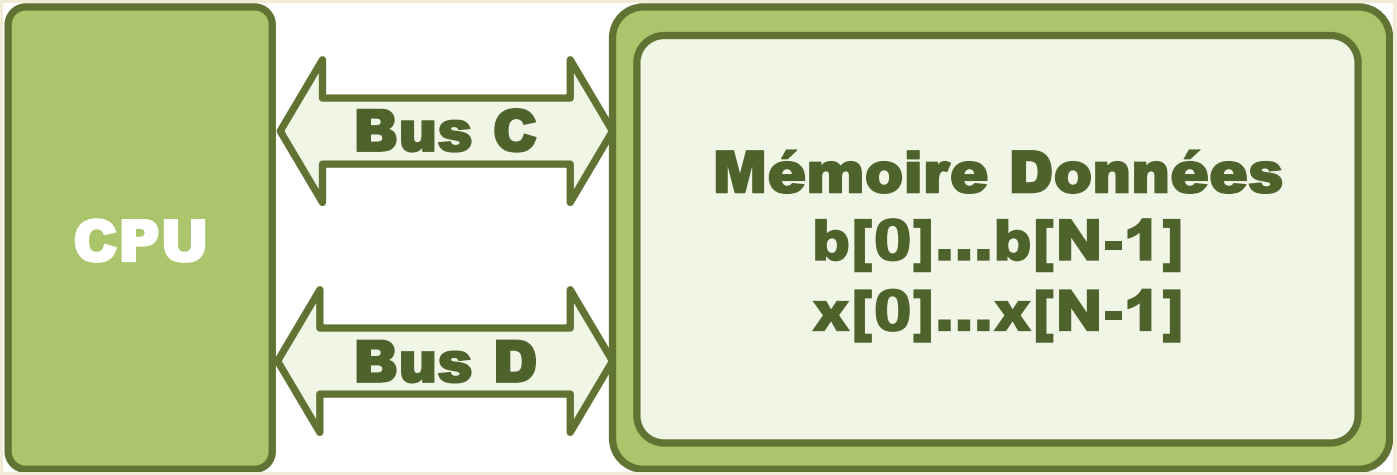


- Pour l'opération $y[n] \leftarrow b[n] * x[n]$
- On souhaite récupérer $b[n]$ et $x[n]$
- Chaque bus est capable de transmettre 1 donnée par cycle
- Mémoire à Simple Accès (SARAM) $\Rightarrow$ 2 cycles
- Mémoire à Double Accès (DARAM) $\Rightarrow$ 1 cycle

Architecture générale d'un processeur

Accès mémoire multi-port

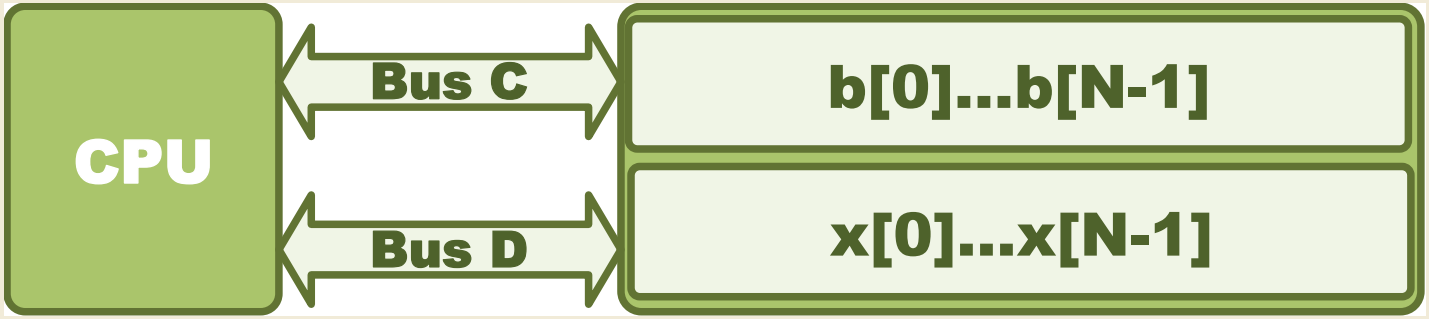
Exemple



- Pour l'opération $y[n] \leftarrow b[n] * x[n]$
- On souhaite récupérer $b[n]$ et $x[n]$
- Chaque bus est capable de transmettre 1 donnée par cycle
- Mémoire à Simple Accès (SARAM) $\Rightarrow$ 2 cycles
- Mémoire à Double Accès (DARAM) $\Rightarrow$ 1 cycle
- Mémoire externe $\Rightarrow$ temps de latence dépend de la qualité de la mémoire

Architecture générale d'un processeur

Accès mémoire multi-port

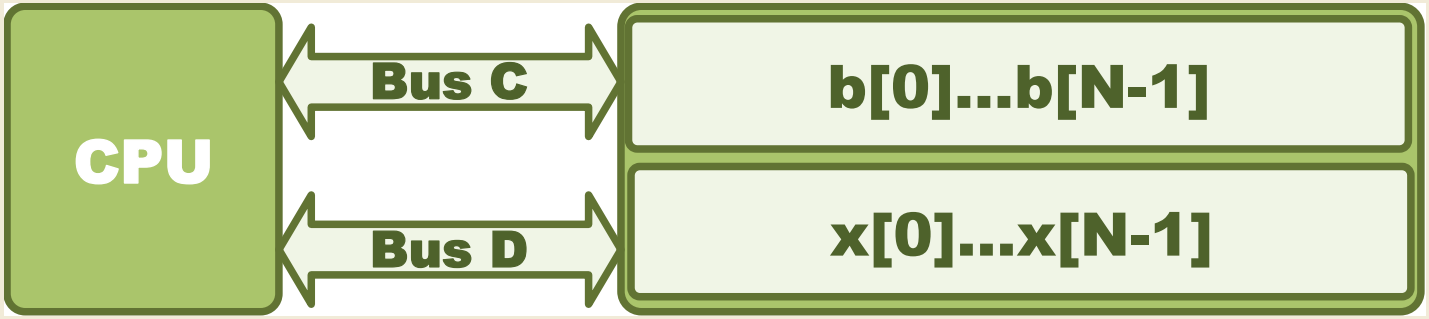


Multiplexage spatial (Mémoire multi-blocs)

- Découpage en blocs indépendants: Accès simultané à deux blocs distincts

Architecture générale d'un processeur

Accès mémoire multi-port

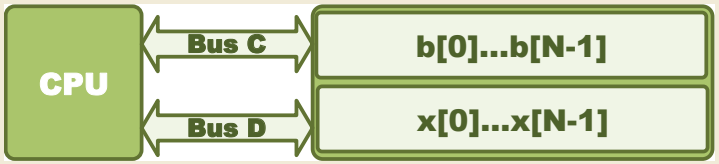


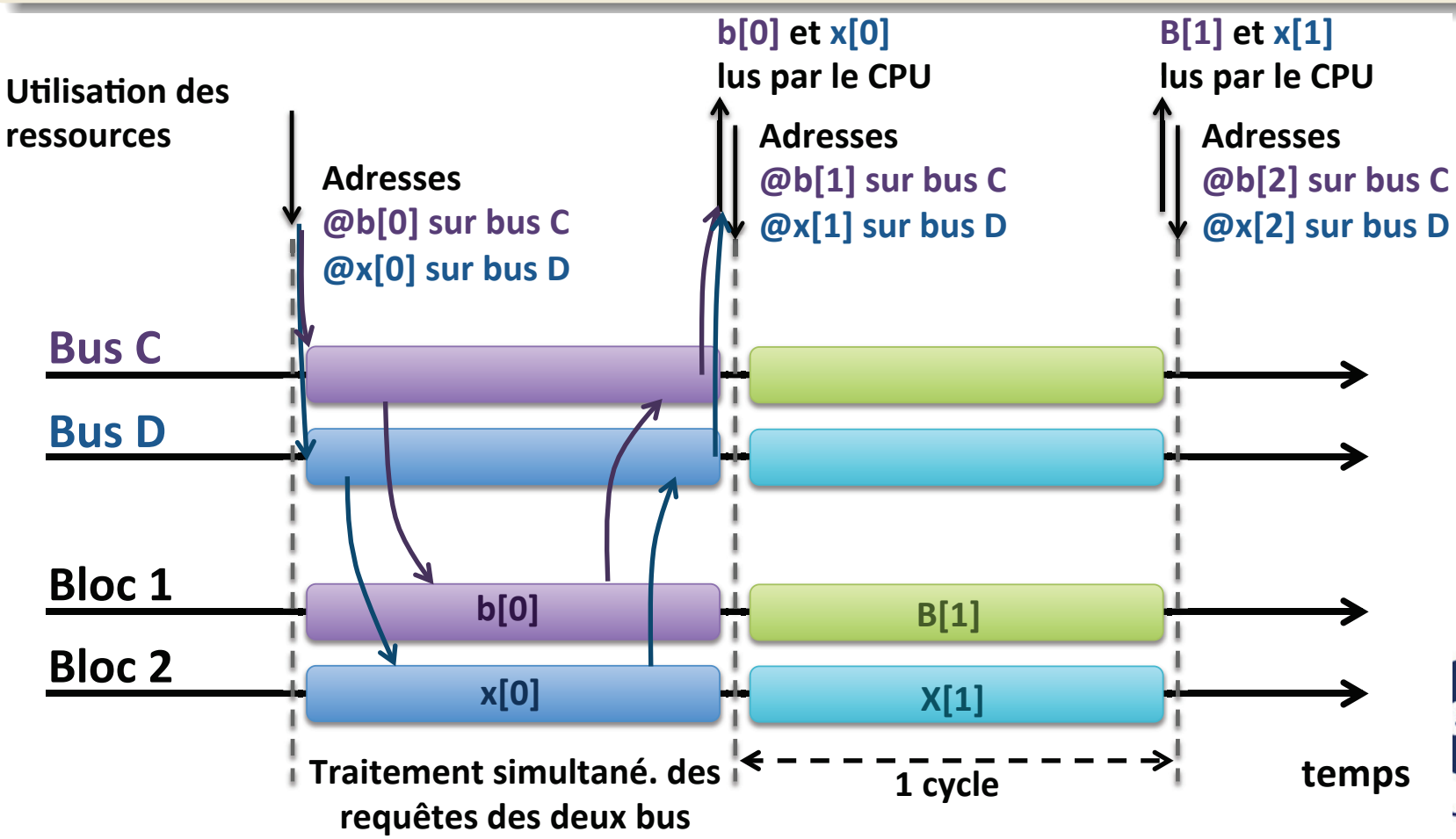
Multiplexage spatial (Mémoire multi-blocs)

- Découpage en blocs indépendants: Accès simultané à deux blocs distincts
- Soient $b[n]$ stockés dans le bloc 1, $x[n]$ stockés dans le bloc 2

Architecture générale d'un processeur

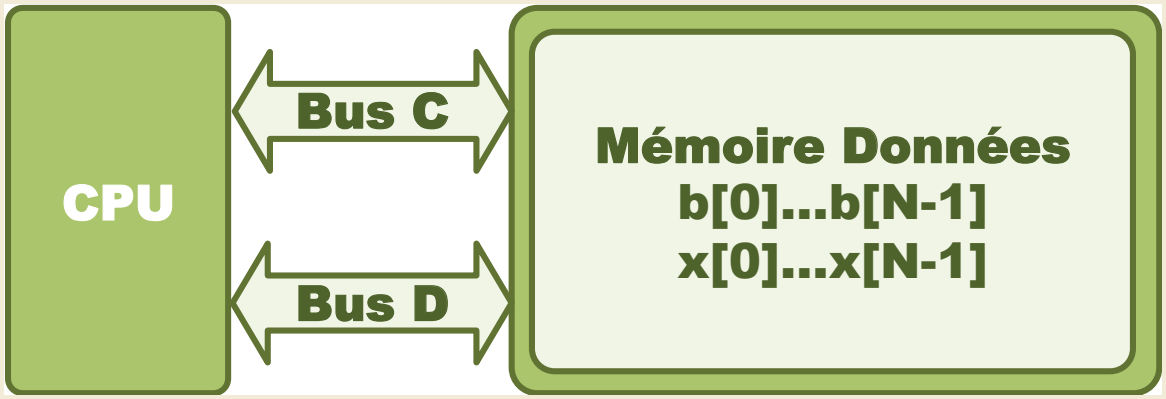
Accès mémoire multi-port





Architecture générale d'un processeur

Accès mémoire multi-port

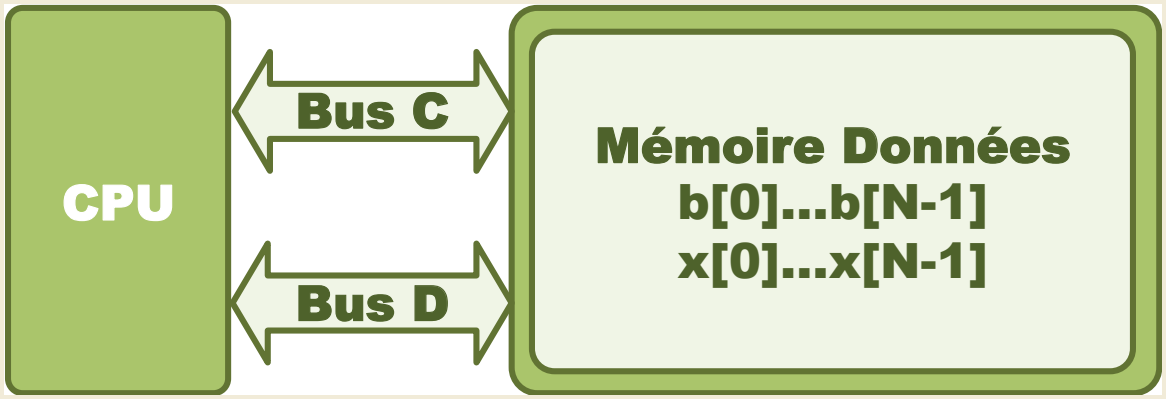


Multiplexage temporel (Mémoire multi-accès)

- Plusieurs accès à la même mémoire en un cycle

Architecture générale d'un processeur

Accès mémoire multi-port

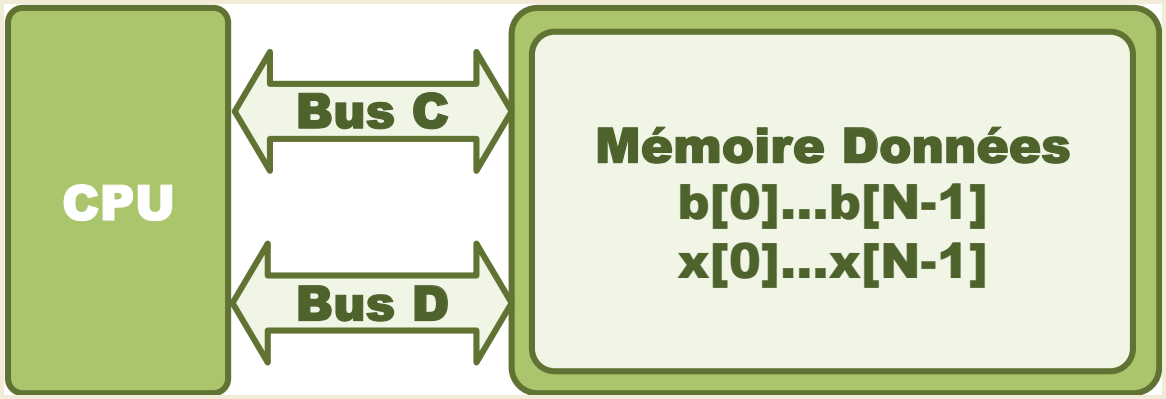


Multiplexage temporel (Mémoire multi-accès)

- Plusieurs accès à la même mémoire en un cycle
- Le CPU récupère cependant une donnée par cycle et par bus

Architecture générale d'un processeur

Accès mémoire multi-port

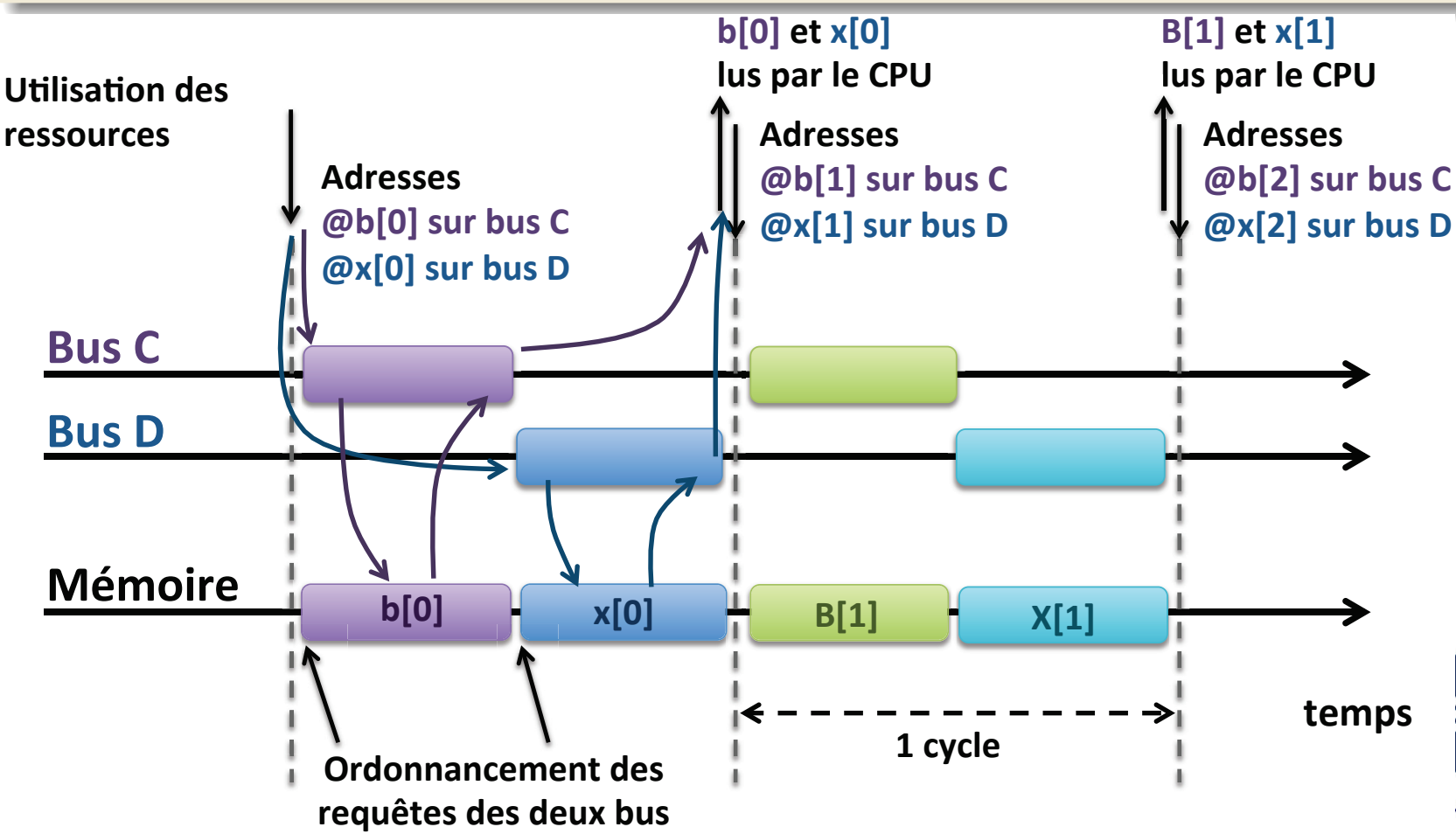


Multiplexage temporel (Mémoire multi-accès)

- Plusieurs accès à la même mémoire en un cycle
- Le CPU récupère cependant une donnée par cycle et par bus
- Si la fréquence du CPU est f_0, alors la fréquence du DARAM est $2f_0$

Architecture générale d'un processeur

Accès mémoire multi-port



Architecture d'un processeur DSP

Mémoire interne sur les processeurs de la famille C54x

Plusieurs types de RAM incluse sur le chip:

- Single access (SARAM) : un accès par cycle

Architecture d'un processeur DSP

Mémoire interne sur les processeurs de la famille C54x

Plusieurs types de RAM incluse sur le chip:

- Single access (SARAM) : un accès par cycle
- Dual access (DARAM) : deux accès par cycle

Architecture d'un processeur DSP

Mémoire interne sur les processeurs de la famille C54x

Plusieurs types de RAM incluse sur le chip:

- Single access (SARAM) : un accès par cycle
- Dual access (DARAM) : deux accès par cycle

DARAM découpée en blocs

- DARAM0: 0 x 0080 - 0 x 1FFF

Architecture d'un processeur DSP

Mémoire interne sur les processeurs de la famille C54x

Plusieurs types de RAM incluse sur le chip:

- Single access (SARAM) : un accès par cycle
- Dual access (DARAM) : deux accès par cycle

DARAM découpée en blocs

- DARAM0: 0 x 0080 - 0 x 1FFF
- DARAM1: 0 x 2000 - 0 x 3FFF

Architecture d'un processeur DSP

Mémoire interne sur les processeurs de la famille C54x

Plusieurs types de RAM incluse sur le chip:

- Single access (SARAM) : un accès par cycle
- Dual access (DARAM) : deux accès par cycle

DARAM découpée en blocs

- DARAM0: 0 x 0080 - 0 x 1FFF
- DARAM1: 0 x 2000 - 0 x 3FFF
- DARAM2: 0 x 4000 - 0 x 5FFF

Architecture d'un processeur DSP

Mémoire interne sur les processeurs de la famille C54x

Plusieurs types de RAM incluse sur le chip:

- Single access (SARAM) : un accès par cycle
- Dual access (DARAM) : deux accès par cycle

DARAM découpée en blocs

- DARAM0: 0 x 0080 - 0 x 1FFF
- DARAM1: 0 x 2000 - 0 x 3FFF
- DARAM2: 0 x 4000 - 0 x 5FFF
- DARAM3: 0 x 6000 - 0 x 7FFF

Architecture d'un processeur DSP

Mémoire interne sur les processeurs de la famille C54x

Plusieurs types de RAM incluse sur le chip:

- Single access (SARAM) : un accès par cycle
- Dual access (DARAM) : deux accès par cycle

DARAM découpée en blocs

- DARAM0: 0 x 0080 - 0 x 1FFF
- DARAM1: 0 x 2000 - 0 x 3FFF
- DARAM2: 0 x 4000 - 0 x 5FFF
- DARAM3: 0 x 6000 - 0 x 7FFF
- DARAM4: 0 x 8000 - 0 x 9FFF

Architecture d'un processeur DSP

Mémoire interne sur les processeurs de la famille C54x

Plusieurs types de RAM incluse sur le chip:

- Single access (SARAM) : un accès par cycle
- Dual access (DARAM) : deux accès par cycle

DARAM découpée en blocs

- DARAM0: 0 x 0080 - 0 x 1FFF
- DARAM1: 0 x 2000 - 0 x 3FFF
- DARAM2: 0 x 4000 - 0 x 5FFF
- DARAM3: 0 x 6000 - 0 x 7FFF
- DARAM4: 0 x 8000 - 0 x 9FFF
- DARAM5: 0 x A000 - 0 x BFFF

Architecture d'un processeur DSP

Mémoire interne sur les processeurs de la famille C54x

Plusieurs types de RAM incluse sur le chip:

- Single access (SARAM) : un accès par cycle
- Dual access (DARAM) : deux accès par cycle

DARAM découpée en blocs

- DARAM0: 0 x 0080 - 0 x 1FFF
- DARAM1: 0 x 2000 - 0 x 3FFF
- DARAM2: 0 x 4000 - 0 x 5FFF
- DARAM3: 0 x 6000 - 0 x 7FFF
- DARAM4: 0 x 8000 - 0 x 9FFF
- DARAM5: 0 x A000 - 0 x BFFF
- DARAM6: 0 x C000 - 0 x DFFF

Architecture d'un processeur DSP

Mémoire interne sur les processeurs de la famille C54x

Plusieurs types de RAM incluse sur le chip:

- Single access (SARAM) : un accès par cycle
- Dual access (DARAM) : deux accès par cycle

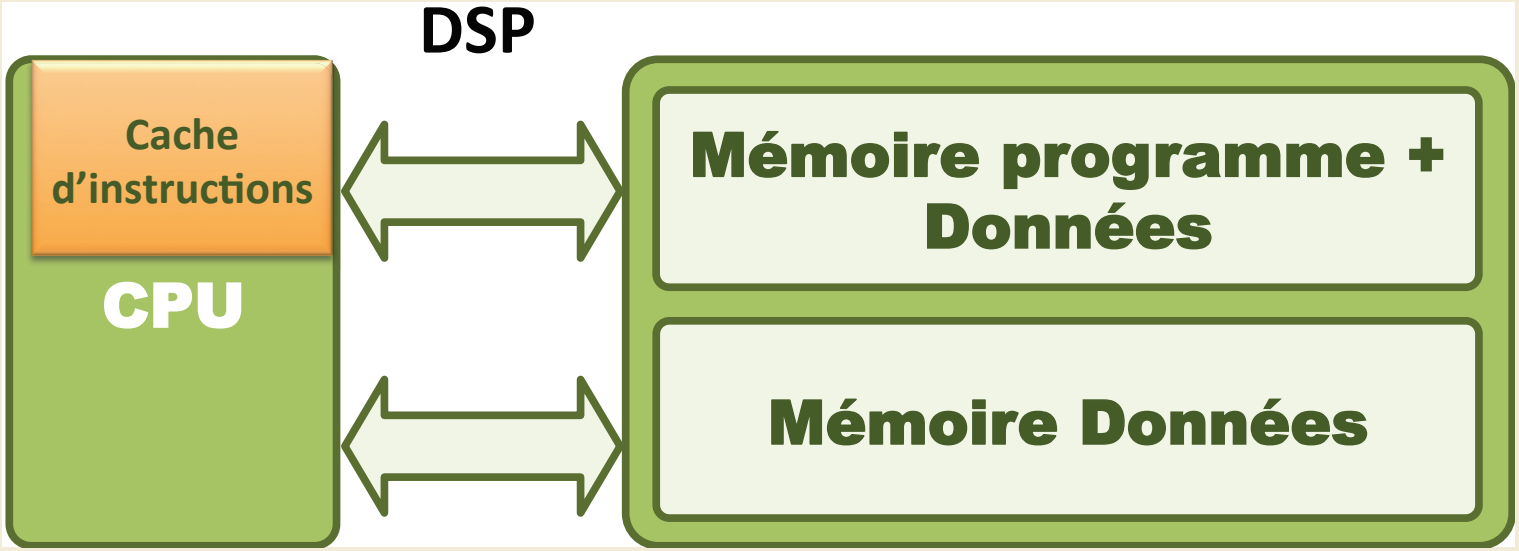
DARAM découpée en blocs

- DARAM0: 0 x 0080 - 0 x 1FFF
- DARAM1: 0 x 2000 - 0 x 3FFF
- DARAM2: 0 x 4000 - 0 x 5FFF
- DARAM3: 0 x 6000 - 0 x 7FFF
- DARAM4: 0 x 8000 - 0 x 9FFF
- DARAM5: 0 x A000 - 0 x BFFF
- DARAM6: 0 x C000 - 0 x DFFF
- DARAM7: 0 x E000 - 0 x FFFF

Architecture d'un processeur DSP

Cache

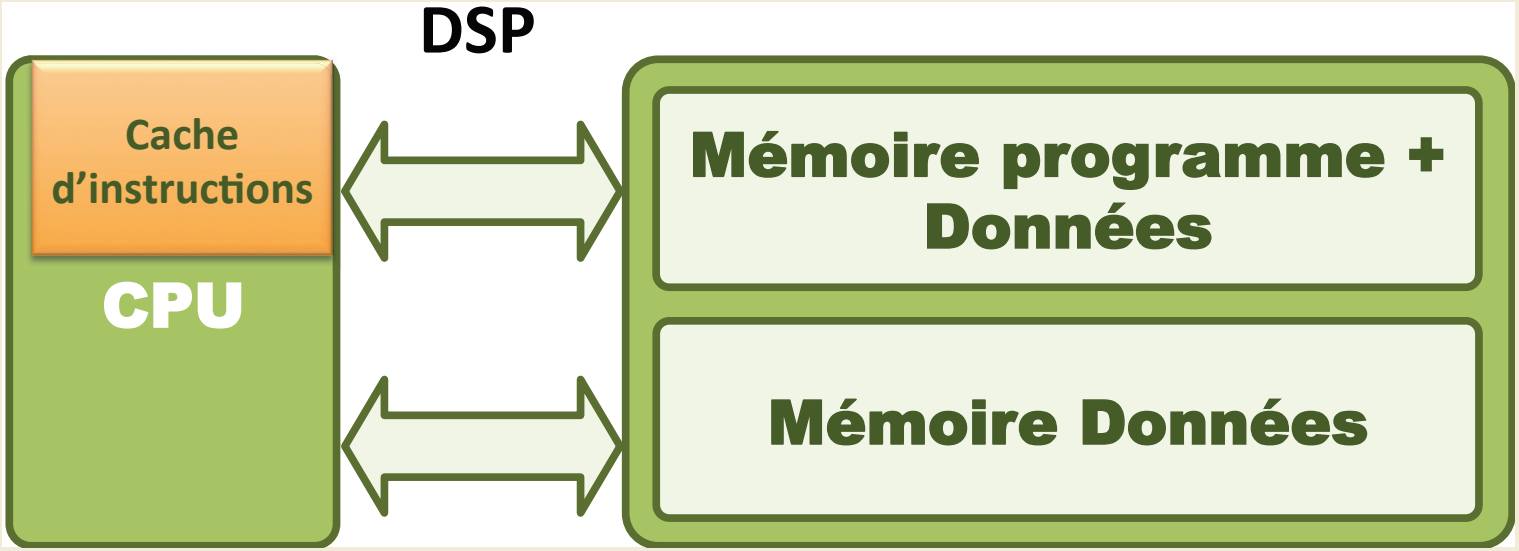
Cache = mémoire associative rapide



- Cache d'instructions

Cache

Cache = mémoire associative rapide

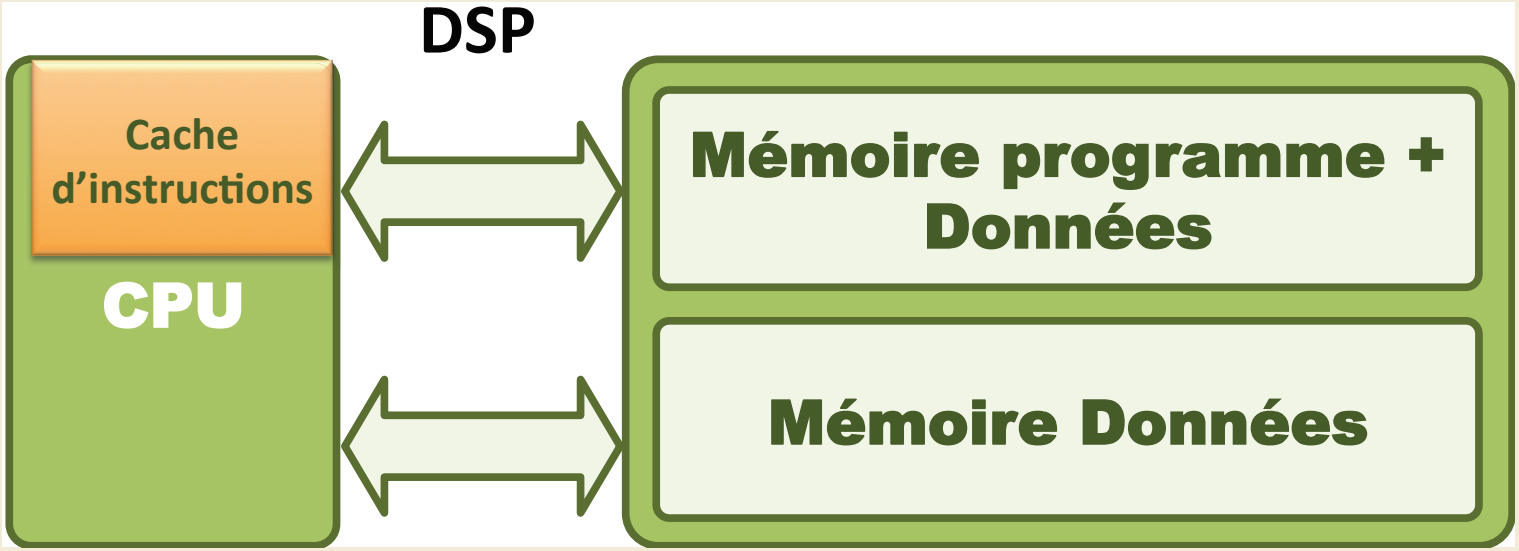


- Cache d'instructions
- Contient les dernières instructions exécutées

Architecture d'un processeur DSP

Cache

Cache = mémoire associative rapide

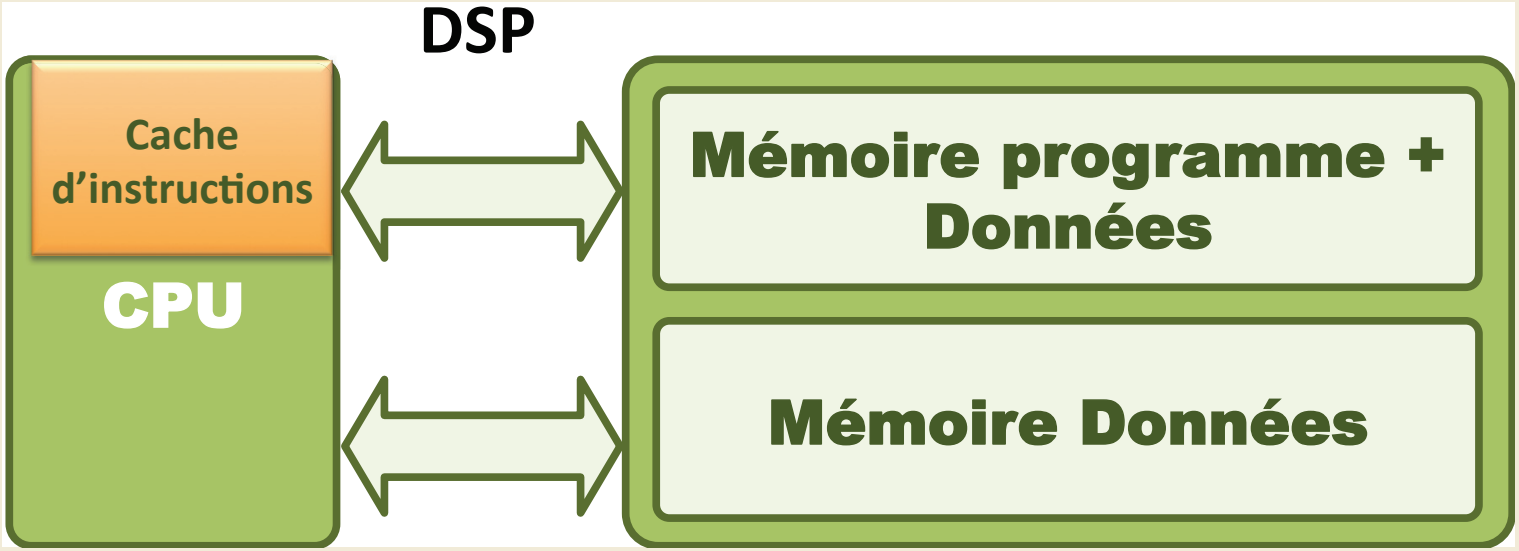


- Cache d'instructions
- Contient les dernières instructions exécutées
- Utile en cas de boucle

Architecture d'un processeur DSP

Cache

Cache = mémoire associative rapide

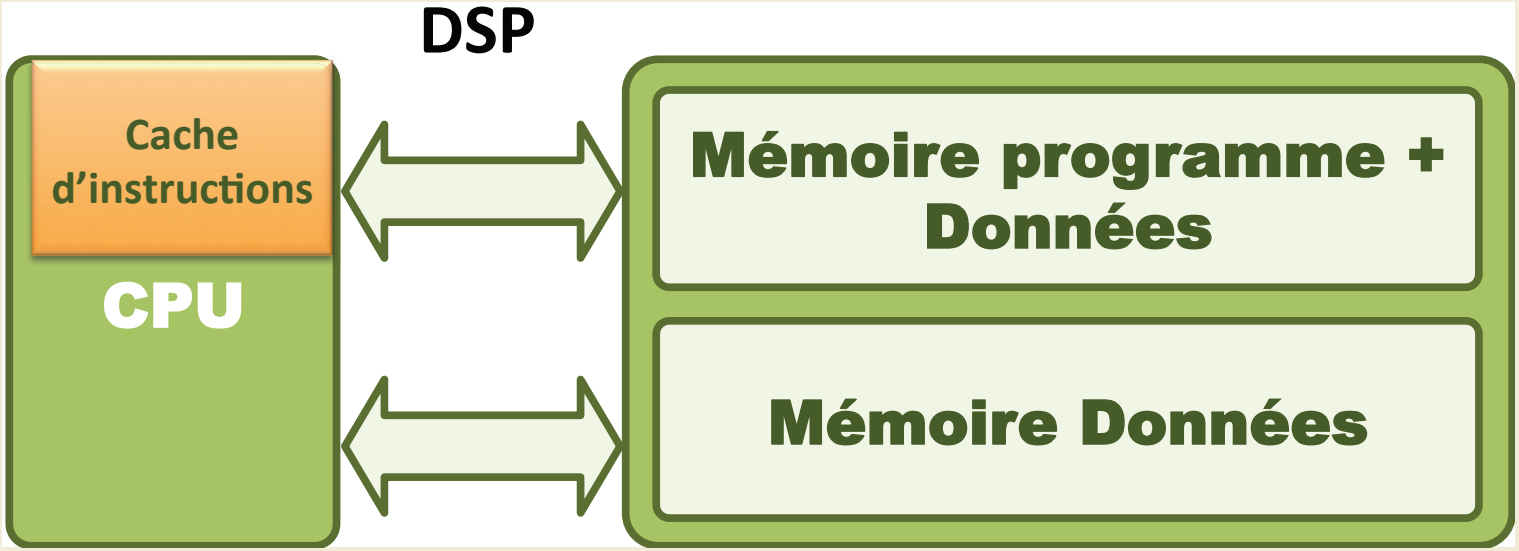


- Cache d'instructions
- Contient les dernières instructions exécutées
- Utile en cas de boucle
- Accès aux instructions sans accès à la mémoire programme

Architecture d'un processeur DSP

Cache

Cache = mémoire associative rapide



- Cache d'instructions
- Contient les dernières instructions exécutées
- Utile en cas de boucle
- Accès aux instructions sans accès à la mémoire programme
- Libère le bus pour des données

Architecture d'un processeur DSP

Cache

Cache pour les données:

- Cache pour les données

Architecture d'un processeur DSP

Cache

Cache pour les données:

- Cache pour les données
- Peu utilisé sur DSP classiques

Architecture d'un processeur DSP

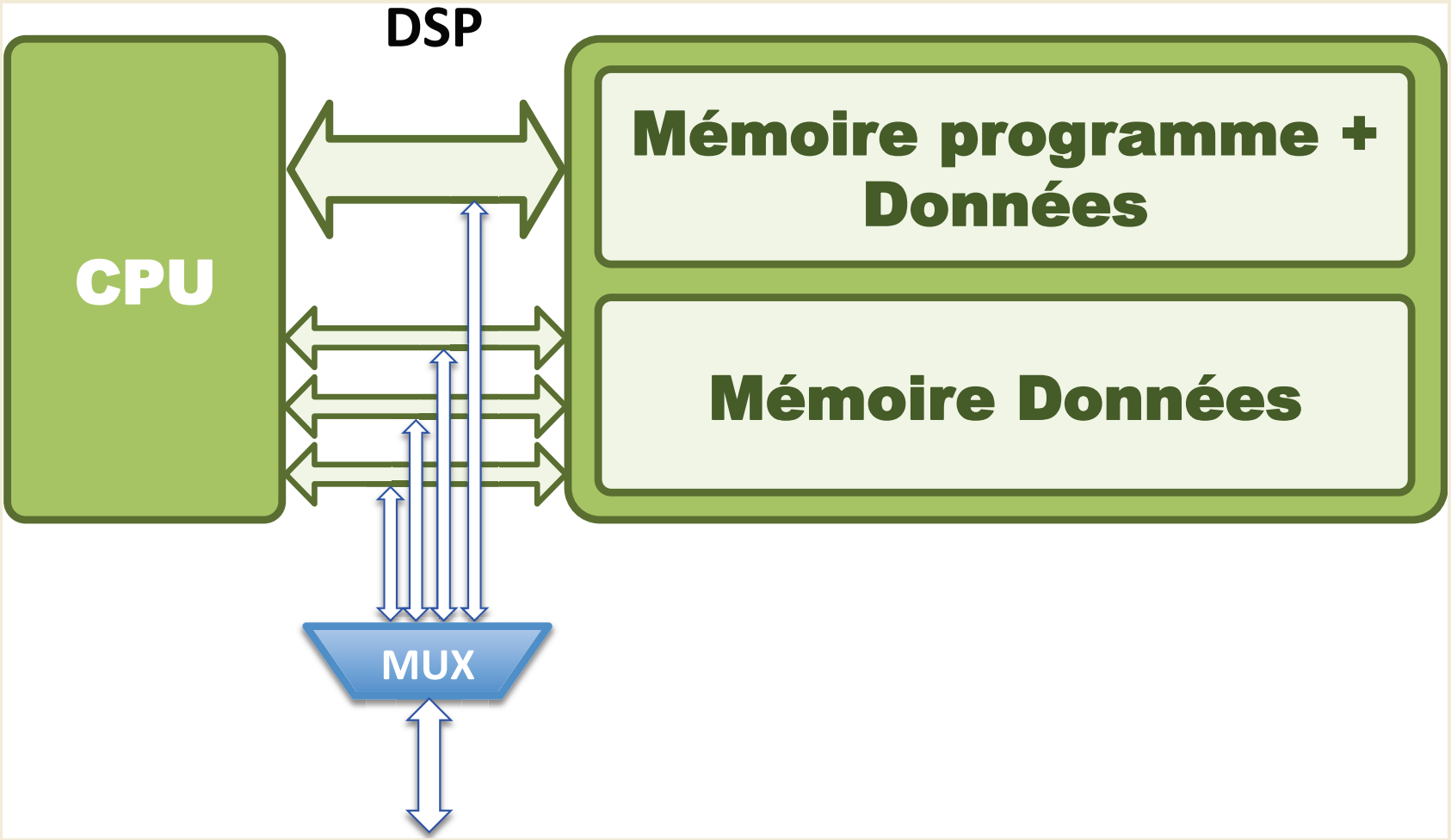
Cache

Cache pour les données:

- Cache pour les données
- Peu utilisé sur DSP classiques
- De plus en plus utilisé sur les DSP hautes-performances

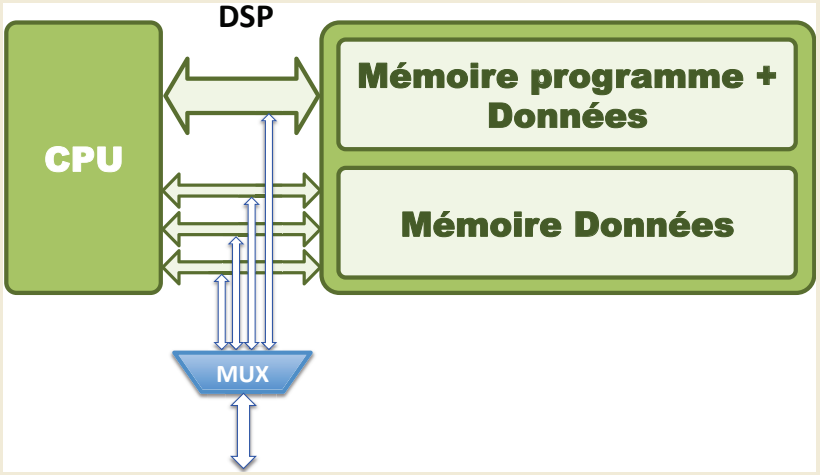
Architecture d'un processeur DSP

Bus externe dans une architecture Harvard



Architecture d'un processeur DSP

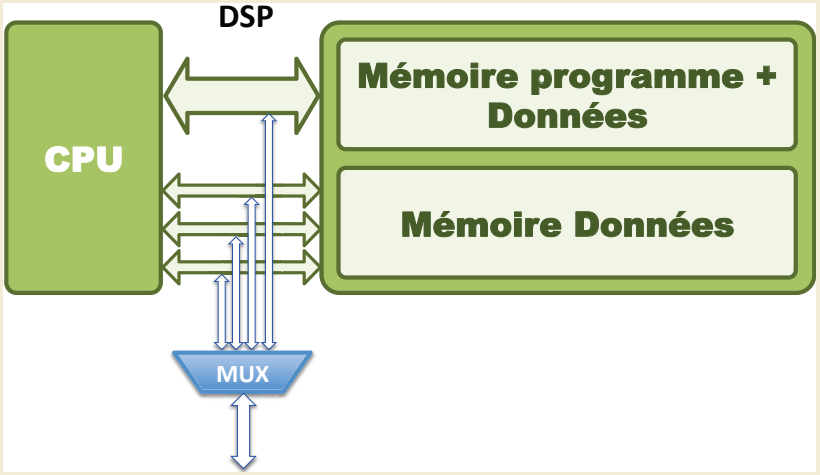
Bus externe dans une architecture Harvard



- Transfert entre les bus internes et externe par multiplexage temporel

Architecture d'un processeur DSP

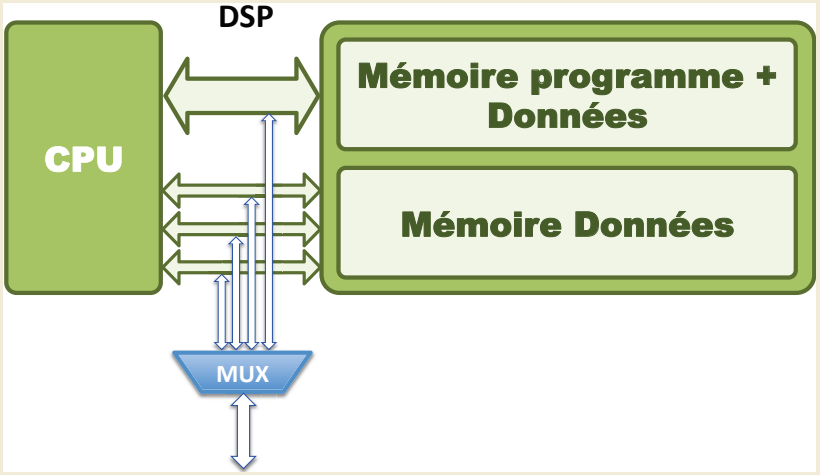
Bus externe dans une architecture Harvard



- Transfert entre les bus internes et externe par multiplexage temporel
- Limitation du nombre de broches

Architecture d'un processeur DSP

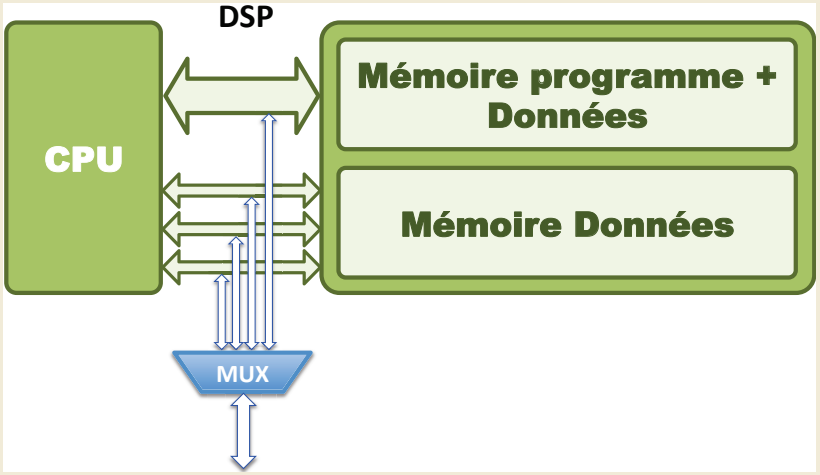
Bus externe dans une architecture Harvard



- Transfert entre les bus internes et externe par multiplexage temporel
- Limitation du nombre de broches
- Réduction des coûts

Architecture d'un processeur DSP

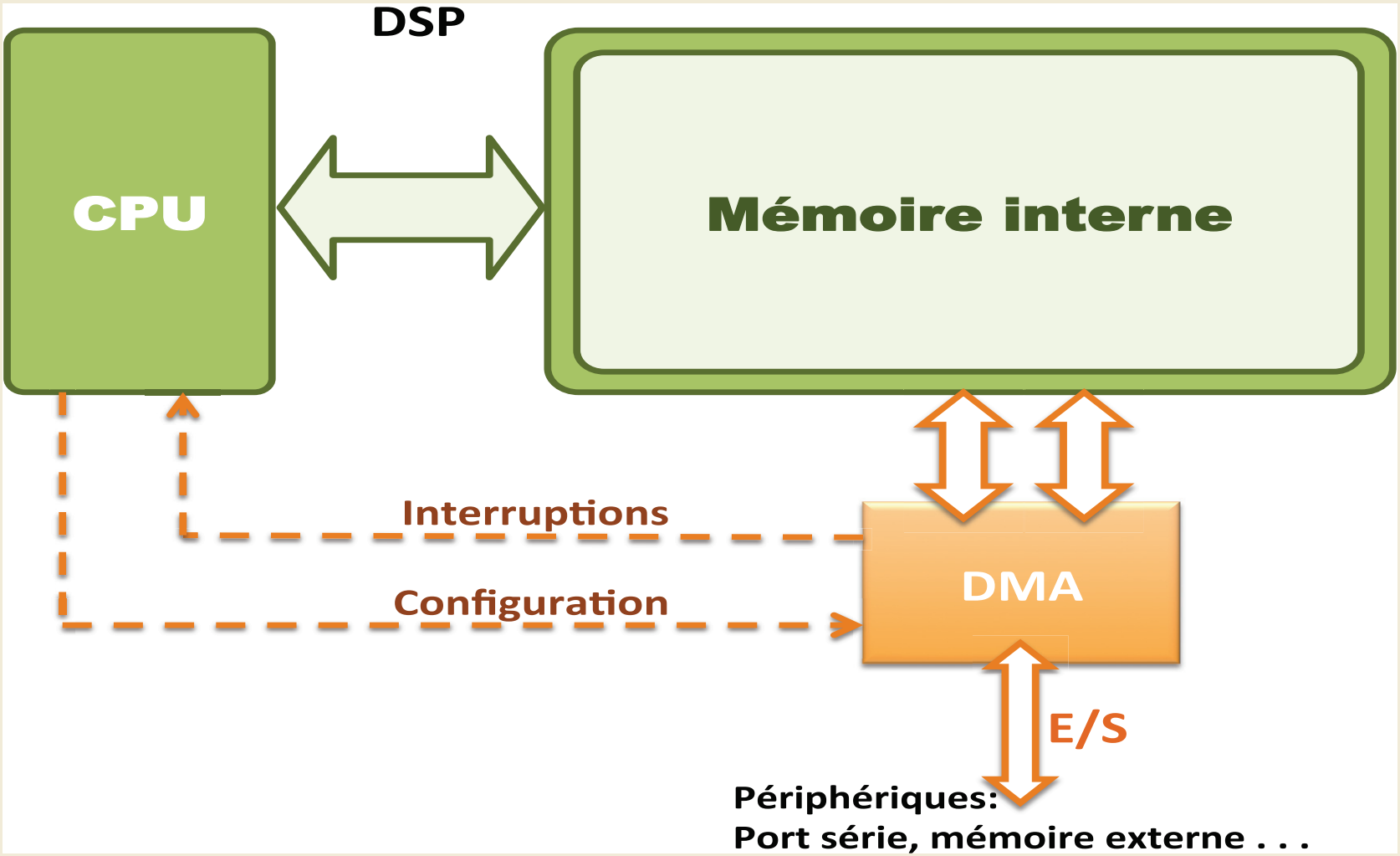
Bus externe dans une architecture Harvard



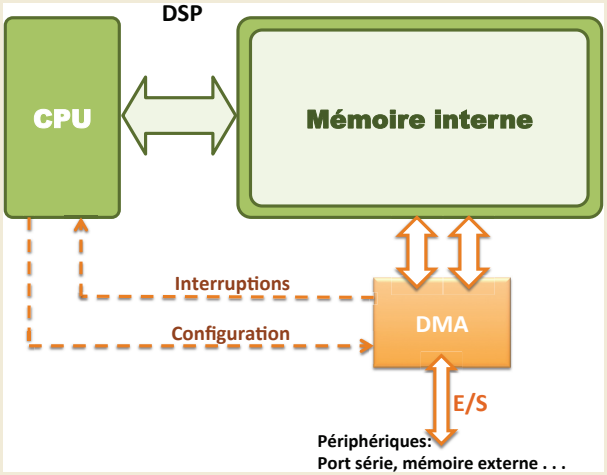
- Transfert entre les bus internes et externe par multiplexage temporel
- Limitation du nombre de broches
- Réduction des coûts
- Diminution des performances lors des accès au bus externe

Architecture d'un processeur DSP

Accès Direct à la Mémoire (DMA- Direct Memory Access)

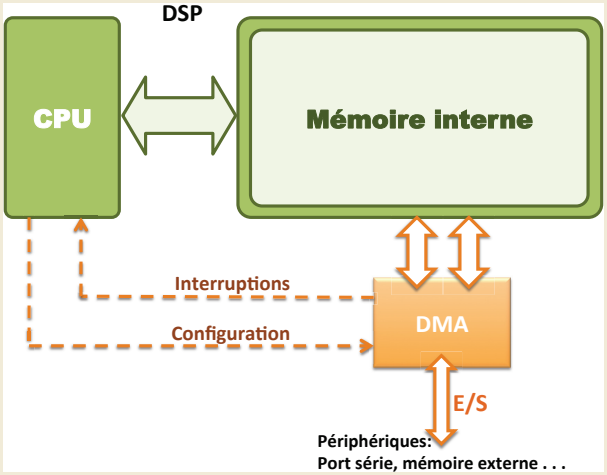


Accès Direct à la Mémoire (DMA- Direct Memory Access)



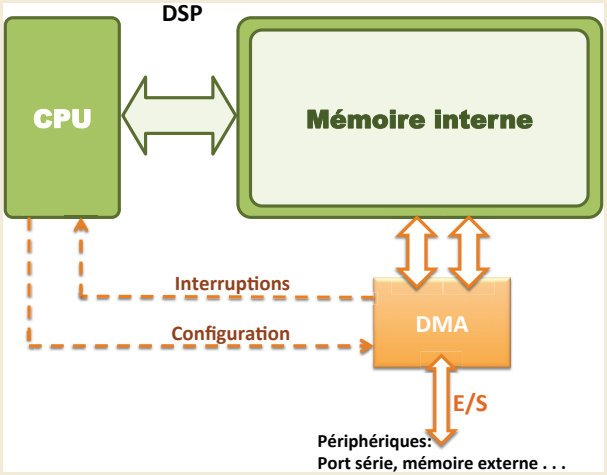
- Mouvements de données

Accès Direct à la Mémoire (DMA- Direct Memory Access)



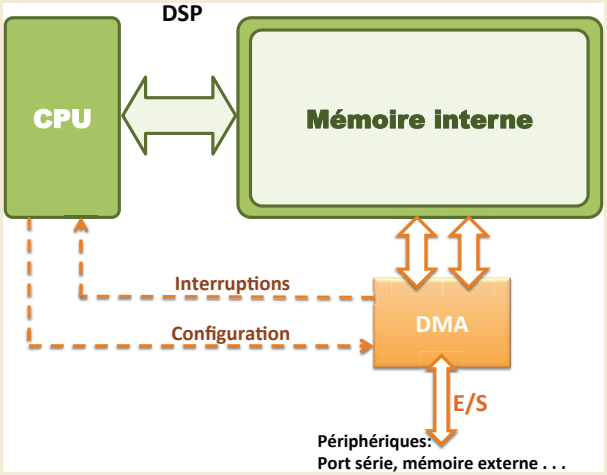
- Mouvements de données
  - E/S direct entre périphérique et mémoire interne

Accès Direct à la Mémoire (DMA- Direct Memory Access)



- Mouvements de données
  - E/S direct entre périphérique et mémoire interne
  - Déplacements au sein de la mémoire interne

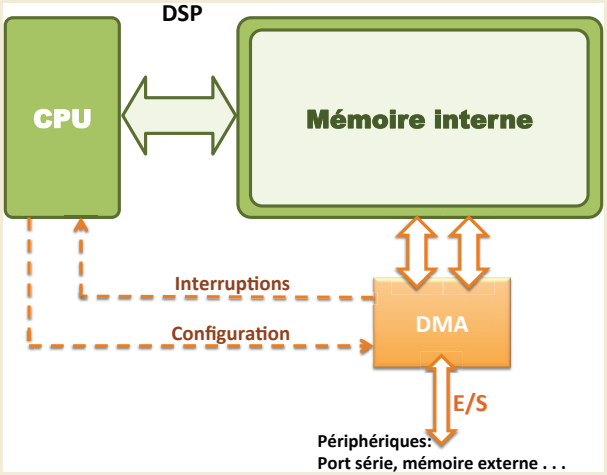
Accès Direct à la Mémoire (DMA- Direct Memory Access)



- Mouvements de données
  - E/S direct entre périphérique et mémoire interne
  - Déplacements au sein de la mémoire interne
- Indépendant du CPU

Architecture d'un processeur DSP

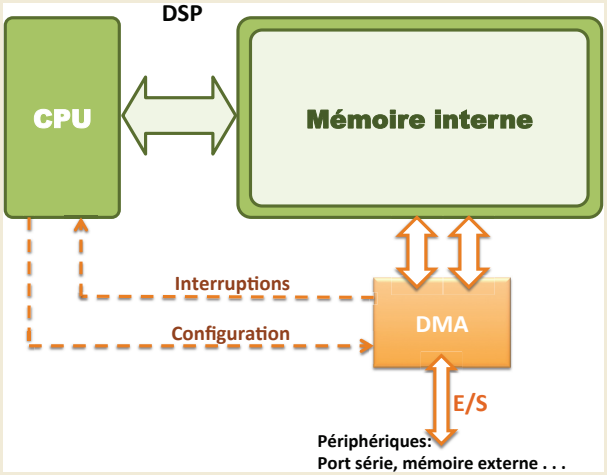
Accès Direct à la Mémoire (DMA- Direct Memory Access)



- Mouvements de données
  - E/S direct entre périphérique et mémoire interne
  - Déplacements au sein de la mémoire interne
- Indépendant du CPU
  - Configuration par le CPU une fois pour toutes

Architecture d'un processeur DSP

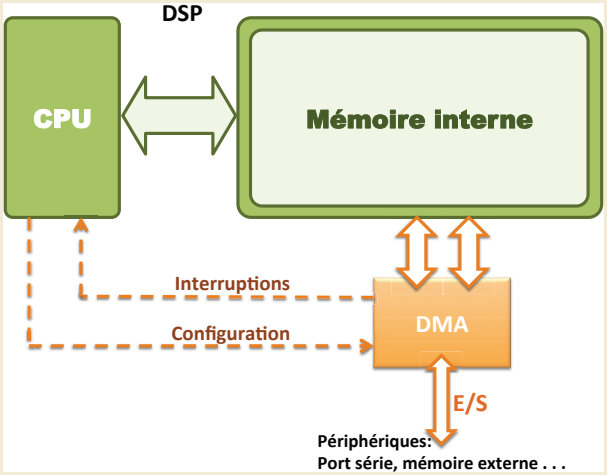
Accès Direct à la Mémoire (DMA- Direct Memory Access)



- Mouvements de données
  - E/S direct entre périphérique et mémoire interne
  - Déplacements au sein de la mémoire interne
- Indépendant du CPU
  - Configuration par le CPU une fois pour toutes
  - CPU prévenu du déroulement par interruptions

Architecture d'un processeur DSP

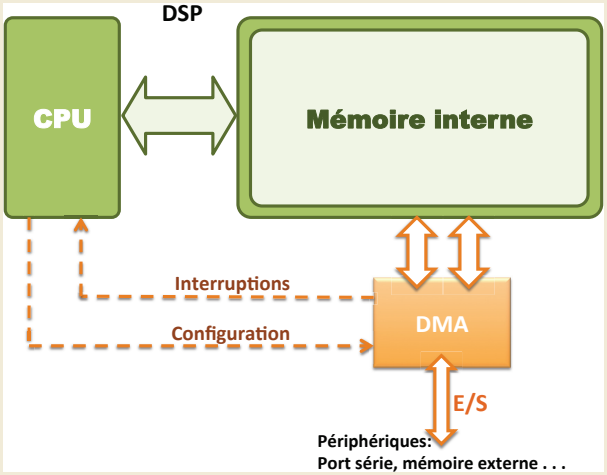
Accès Direct à la Mémoire (DMA- Direct Memory Access)



- Mouvements de données
  - E/S direct entre périphérique et mémoire interne
  - Déplacements au sein de la mémoire interne
- Indépendant du CPU
  - Configuration par le CPU une fois pour toutes
  - CPU prévenu du déroulement par interruptions
    - Buffer plein (en réception) / Buffer vide (en émission)

Architecture d'un processeur DSP

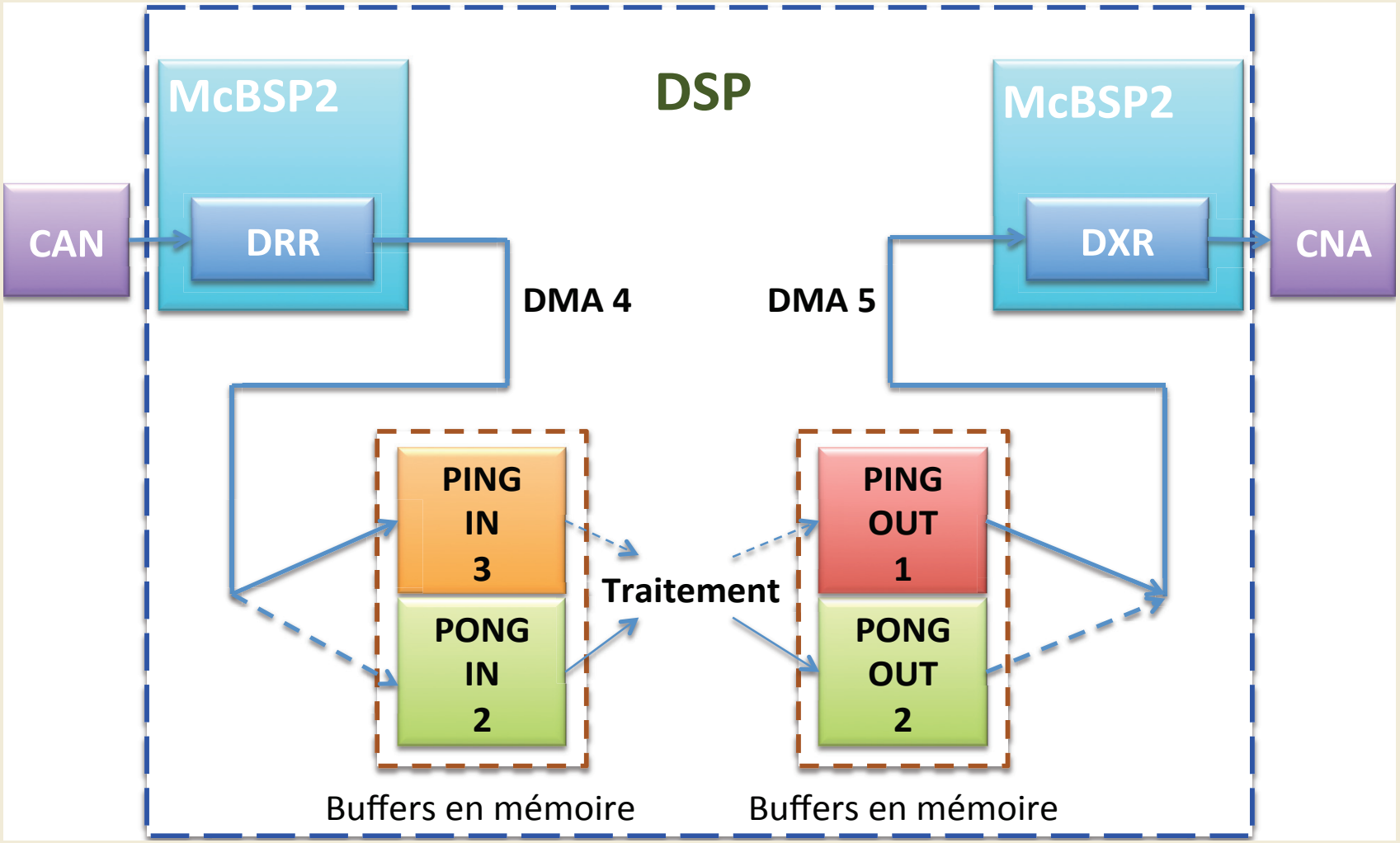
Accès Direct à la Mémoire (DMA- Direct Memory Access)



- Mouvements de données
  - E/S direct entre périphérique et mémoire interne
  - Déplacements au sein de la mémoire interne
- Indépendant du CPU
  - Configuration par le CPU une fois pour toutes
  - CPU prévenu du déroulement par interruptions
    - Buffer plein (en réception) / Buffer vide (en émission)
    - Buffer à demi-plein

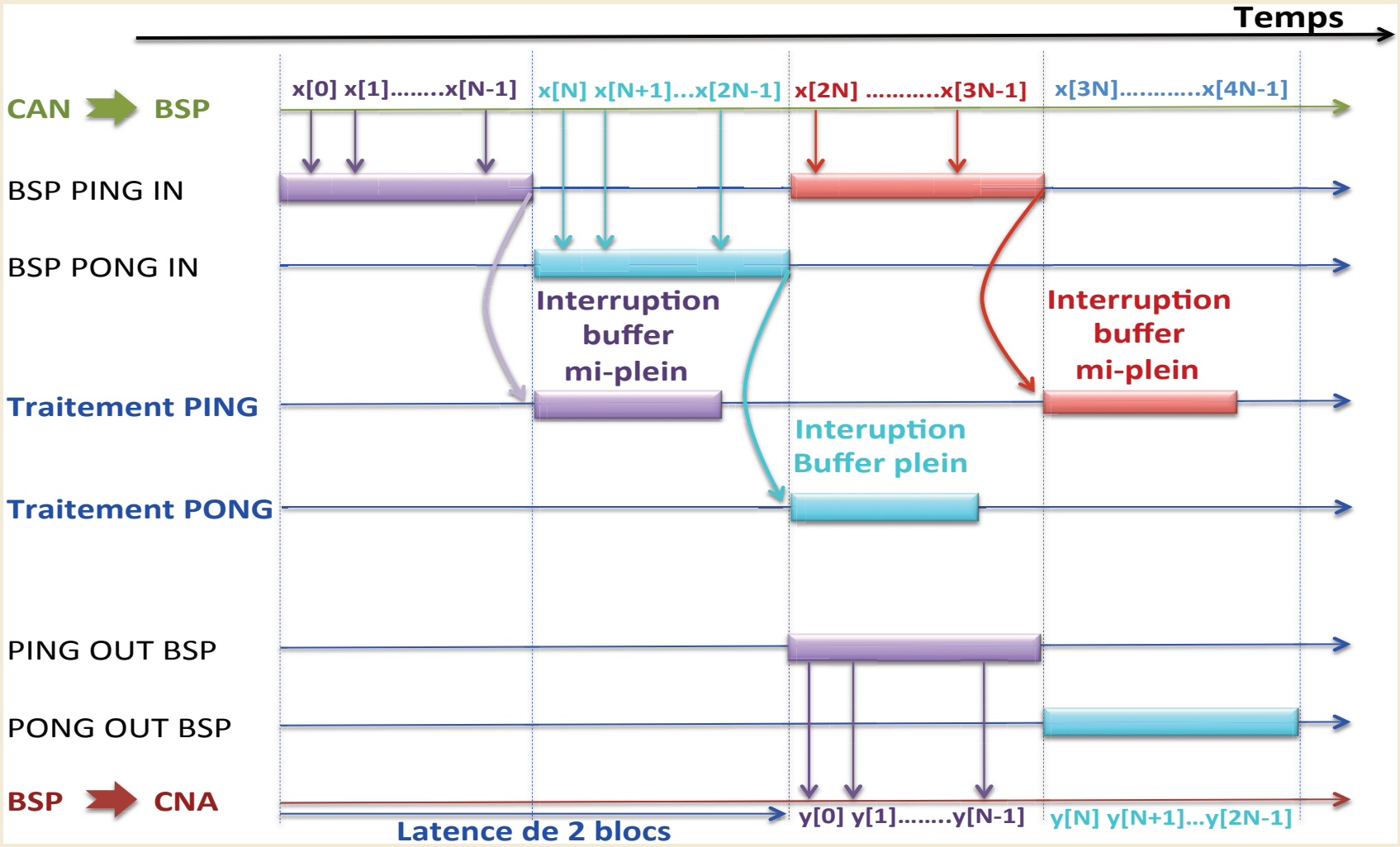
Architecture d'un processeur DSP

DMA et buffer ping-pong pour le traitement par blocs



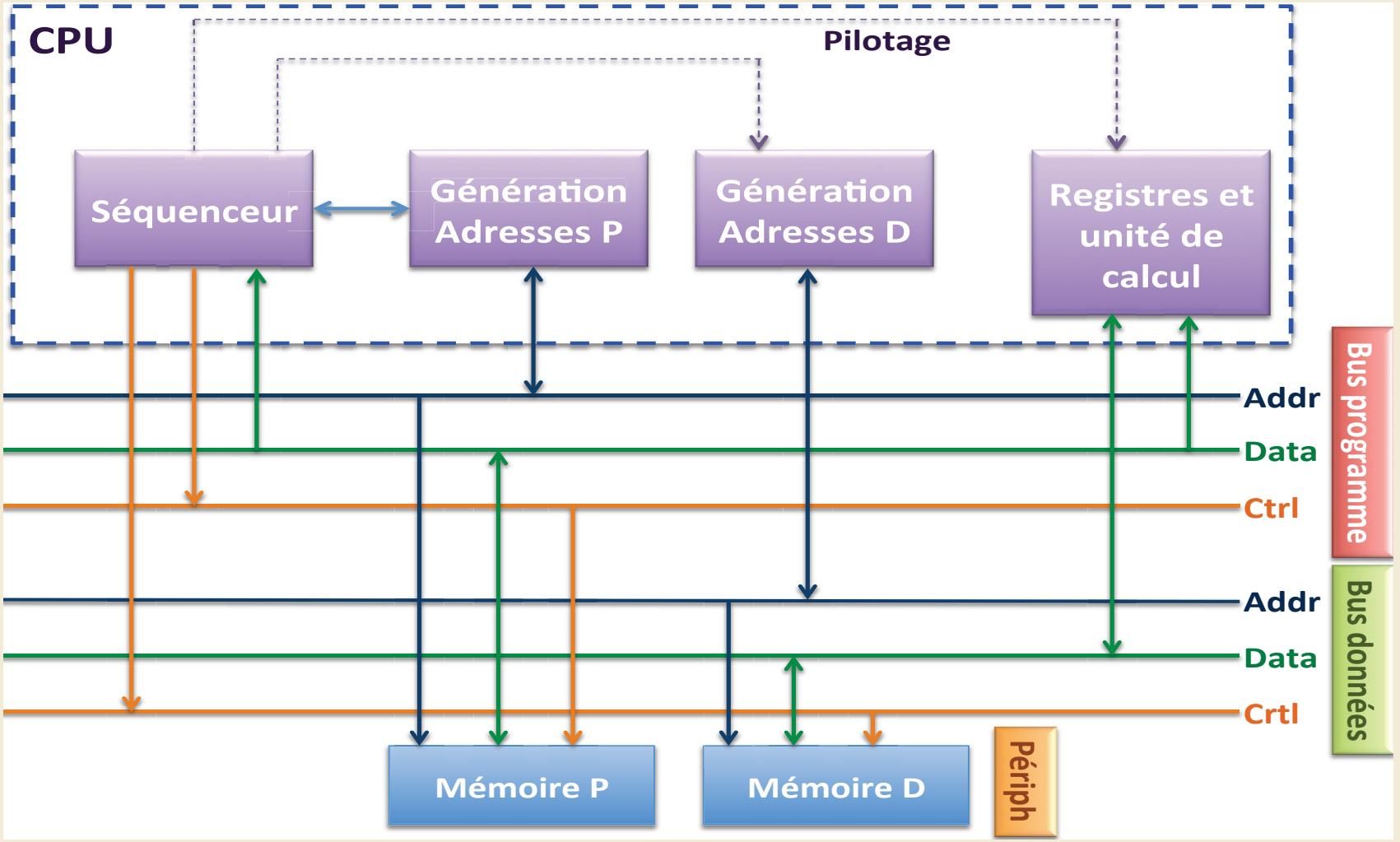
Architecture d'un processeur DSP

Chronogramme des échanges de données



Architecture d'un processeur DSP

Architecture interne du CPU: Schéma de principe des échanges



Architecture d'un processeur DSP

Architecture interne du CPU

- Séquenceur

Architecture interne du CPU

- Séquenceur
  - Décodage des instructions

Architecture interne du CPU

- Séquenceur
  - Décodage des instructions
  - Pilotage des autres unités

Architecture d'un processeur DSP

Architecture interne du CPU

- Séquenceur
  - Décodage des instructions
  - Pilotage des autres unités
- Unité de génération d'adresse programme

Architecture d'un processeur DSP

Architecture interne du CPU

- Séquenceur
  - Décodage des instructions
  - Pilotage des autres unités
- Unité de génération d'adresse programme
  - Compteur de programme (PC)

Architecture d'un processeur DSP

Architecture interne du CPU

- Séquenceur
  - Décodage des instructions
  - Pilotage des autres unités
- Unité de génération d'adresse programme
  - Compteur de programme (PC)
  - Gestion matérielle des boucles

Architecture d'un processeur DSP

Architecture interne du CPU

- Séquenceur
  - Décodage des instructions
  - Pilotage des autres unités
- Unité de génération d'adresse programme
  - Compteur de programme (PC)
  - Gestion matérielle des boucles
- Unité de génération d'adresses données

Architecture d'un processeur DSP

Architecture interne du CPU

- Séquenceur
  - Décodage des instructions
  - Pilotage des autres unités
- Unité de génération d'adresse programme
  - Compteur de programme (PC)
  - Gestion matérielle des boucles
- Unité de génération d'adresses données
  - Adressage indirect efficace

Architecture d'un processeur DSP

Architecture interne du CPU

- Séquenceur
  - Décodage des instructions
  - Pilotage des autres unités
- Unité de génération d'adresse programme
  - Compteur de programme (PC)
  - Gestion matérielle des boucles
- Unité de génération d'adresses données
  - Adressage indirect efficace
  - Adressage circulaire, bit-reverse

Architecture d'un processeur DSP

Architecture interne du CPU

- Séquenceur
  - Décodage des instructions
  - Pilotage des autres unités
- Unité de génération d'adresse programme
  - Compteur de programme (PC)
  - Gestion matérielle des boucles
- Unité de génération d'adresses données
  - Adressage indirect efficace
  - Adressage circulaire, bit-reverse
- Unités de traitement

Architecture d'un processeur DSP

Architecture interne du CPU

- Séquenceur
  - Décodage des instructions
  - Pilotage des autres unités
- Unité de génération d'adresse programme
  - Compteur de programme (PC)
  - Gestion matérielle des boucles
- Unité de génération d'adresses données
  - Adressage indirect efficace
  - Adressage circulaire, bit-reverse
- Unités de traitement
  - Effectuent les calculs

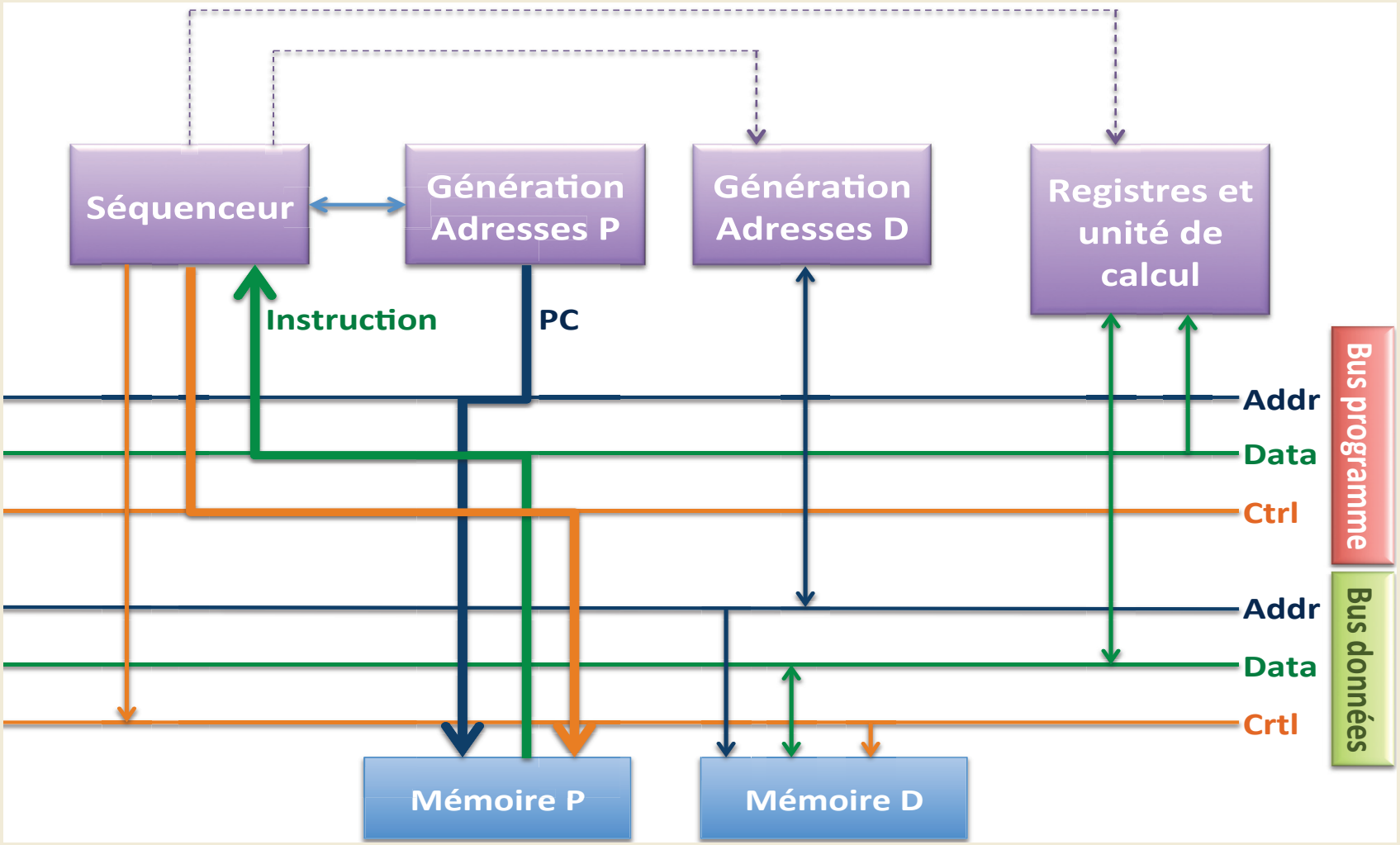
Architecture d'un processeur DSP

Architecture interne du CPU

- Séquenceur
  - Décodage des instructions
  - Pilotage des autres unités
- Unité de génération d'adresse programme
  - Compteur de programme (PC)
  - Gestion matérielle des boucles
- Unité de génération d'adresses données
  - Adressage indirect efficace
  - Adressage circulaire, bit-reverse
- Unités de traitement
  - Effectuent les calculs
  - Registres pour résultats intermédiaires

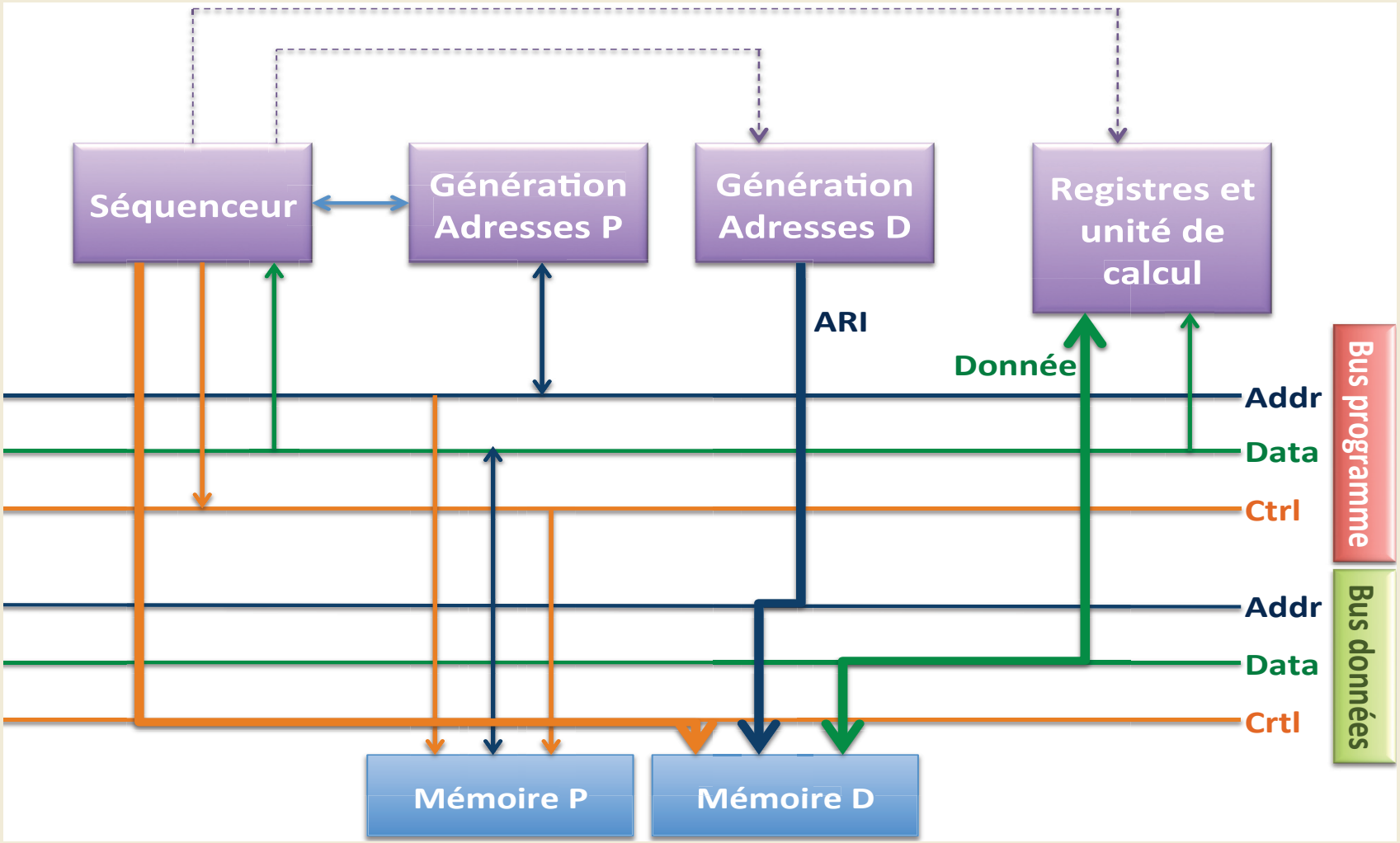
Architecture d'un processeur DSP

Fetch: (lecture des instructions)



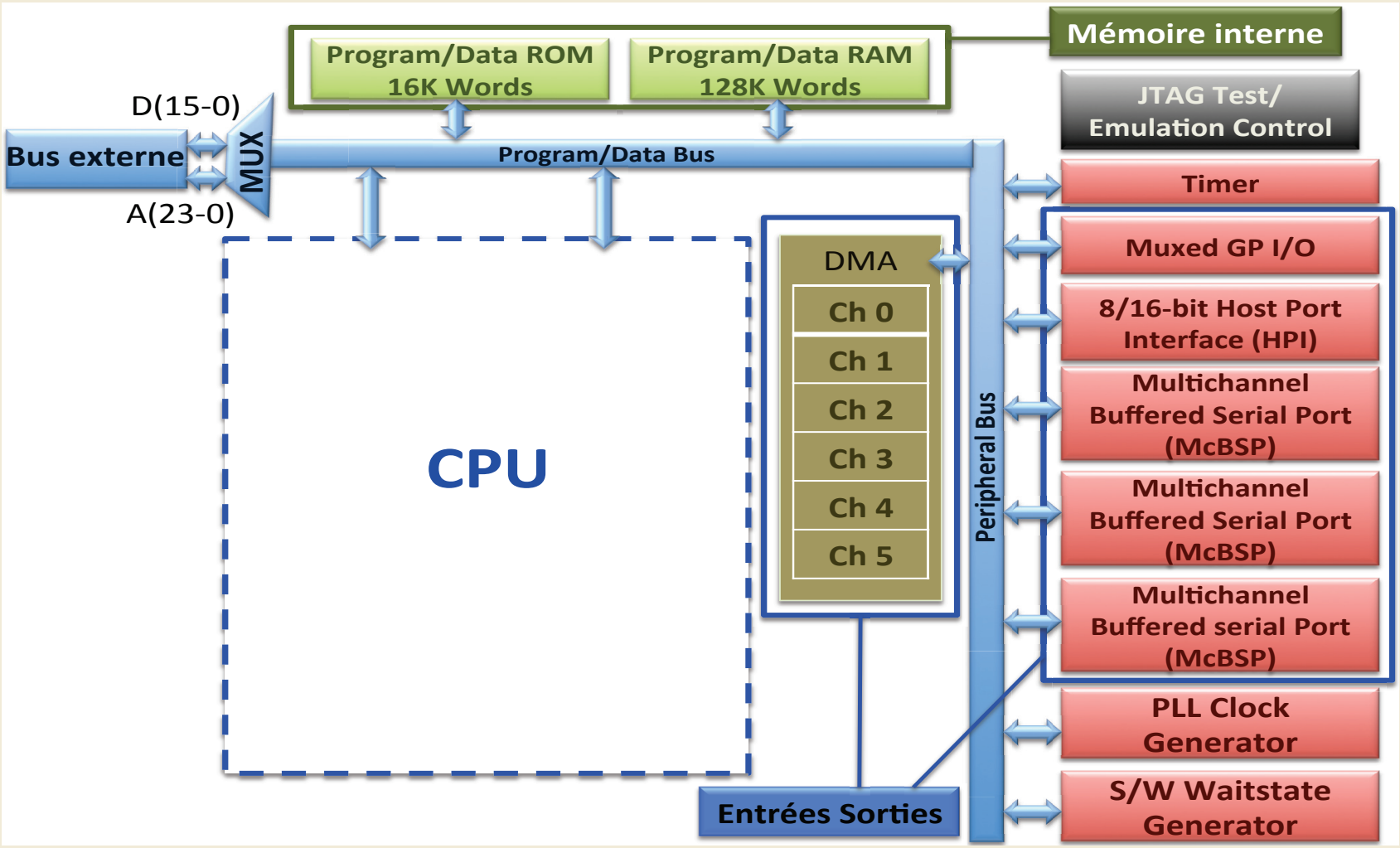
Architecture d'un processeur DSP

Read/Write: (lecture/écriture donnée)



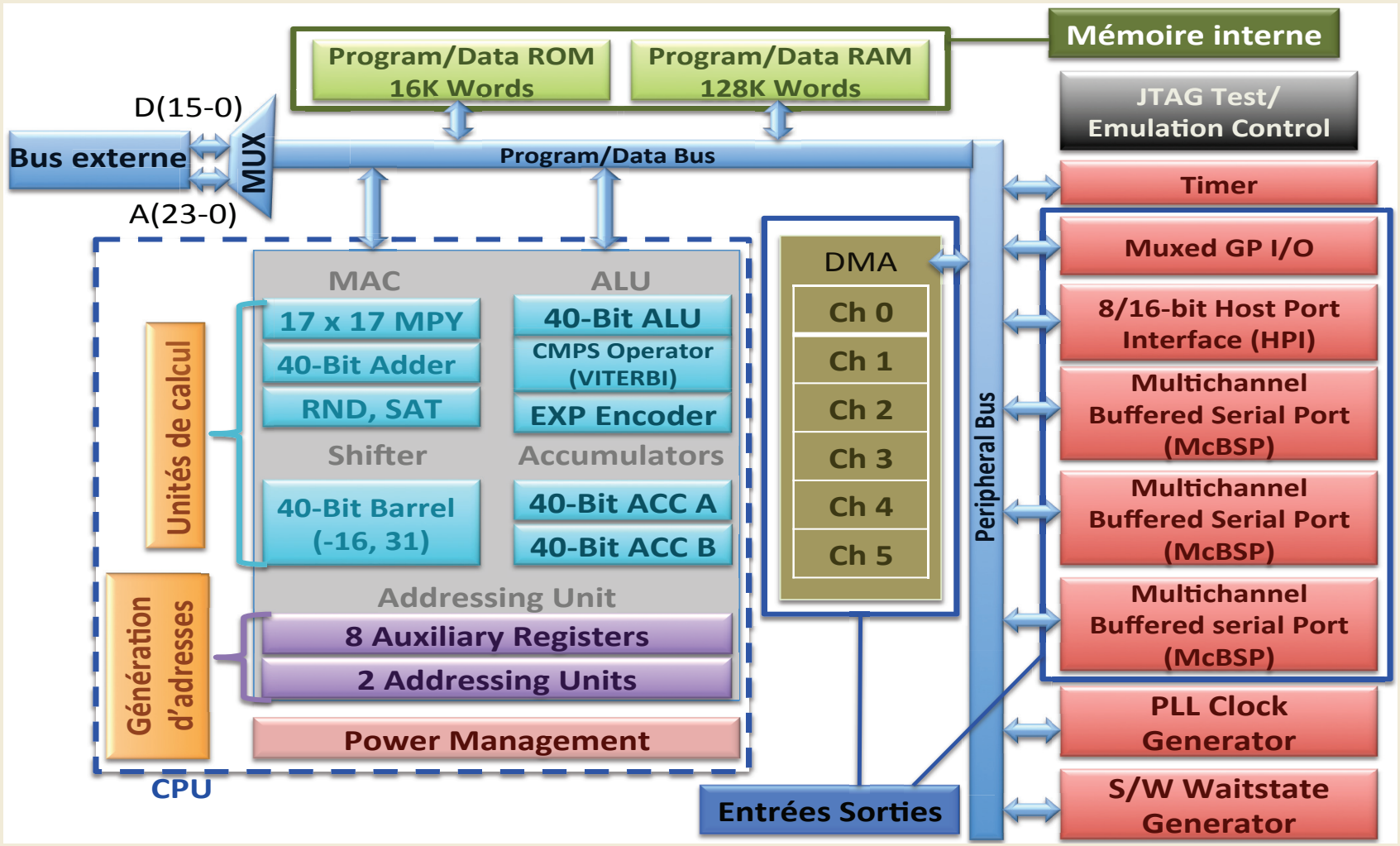
Architecture d'un processeur DSP

Diagramme du processeur TMS320C5416



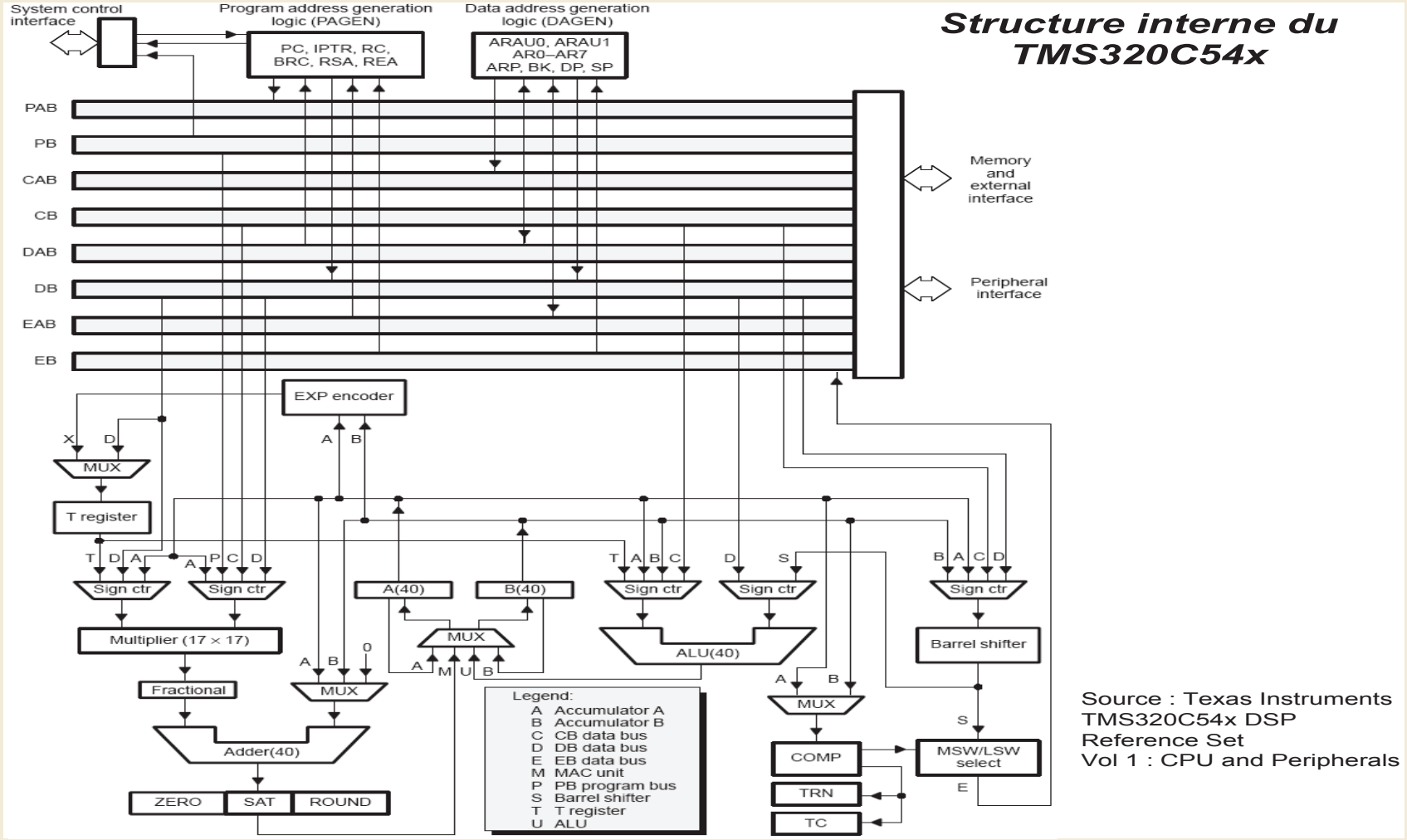
Architecture d'un processeur DSP

Diagramme du processeur TMS320C5416



Architecture d'un processeur DSP

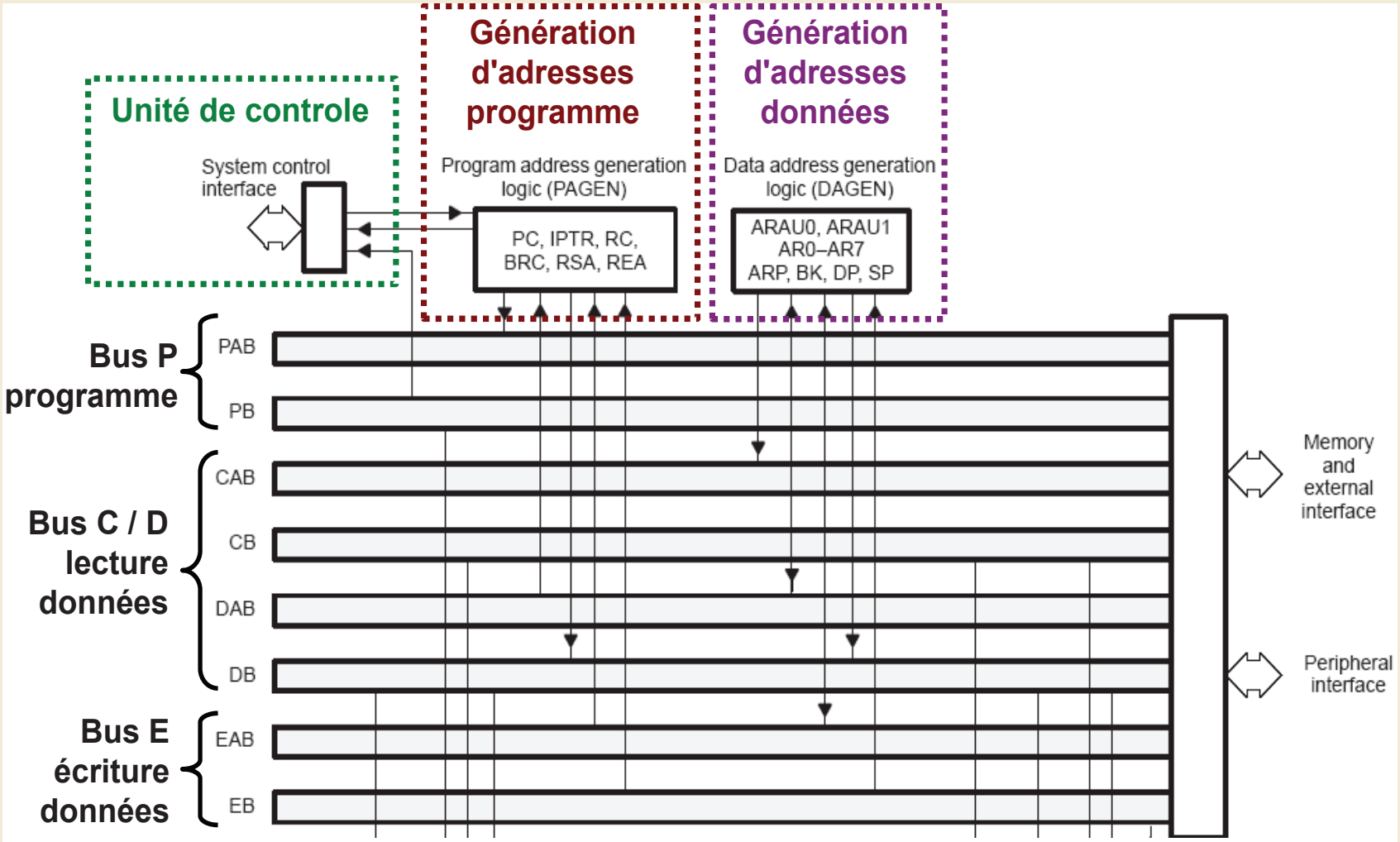
Diagramme du processeur TMS320C5416



Source : Texas Instruments
TMS320C54x DSP
Reference Set
Vol 1 : CPU and Peripherals

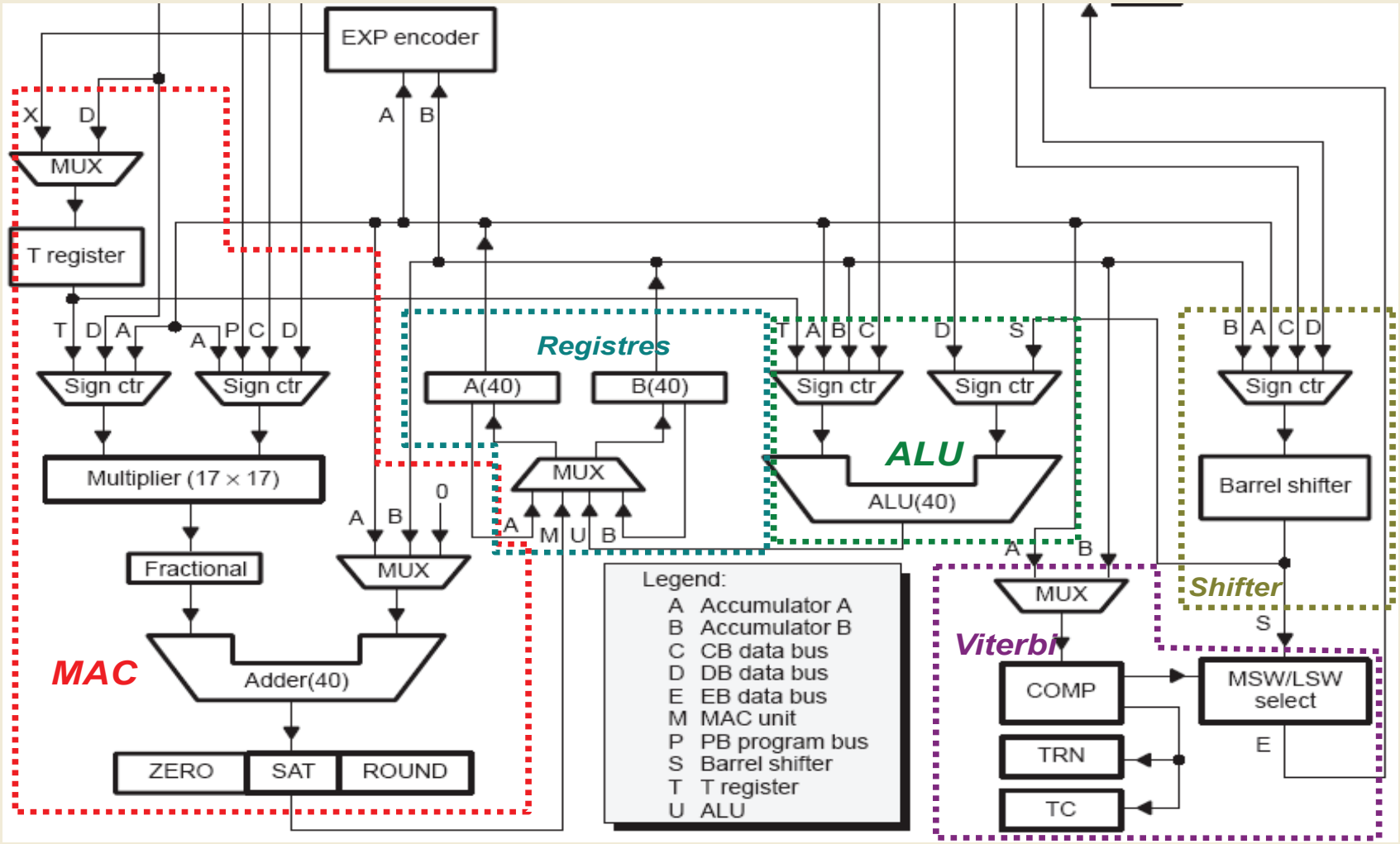
Architecture d'un processeur DSP

Diagramme du processeur TMS320C5416



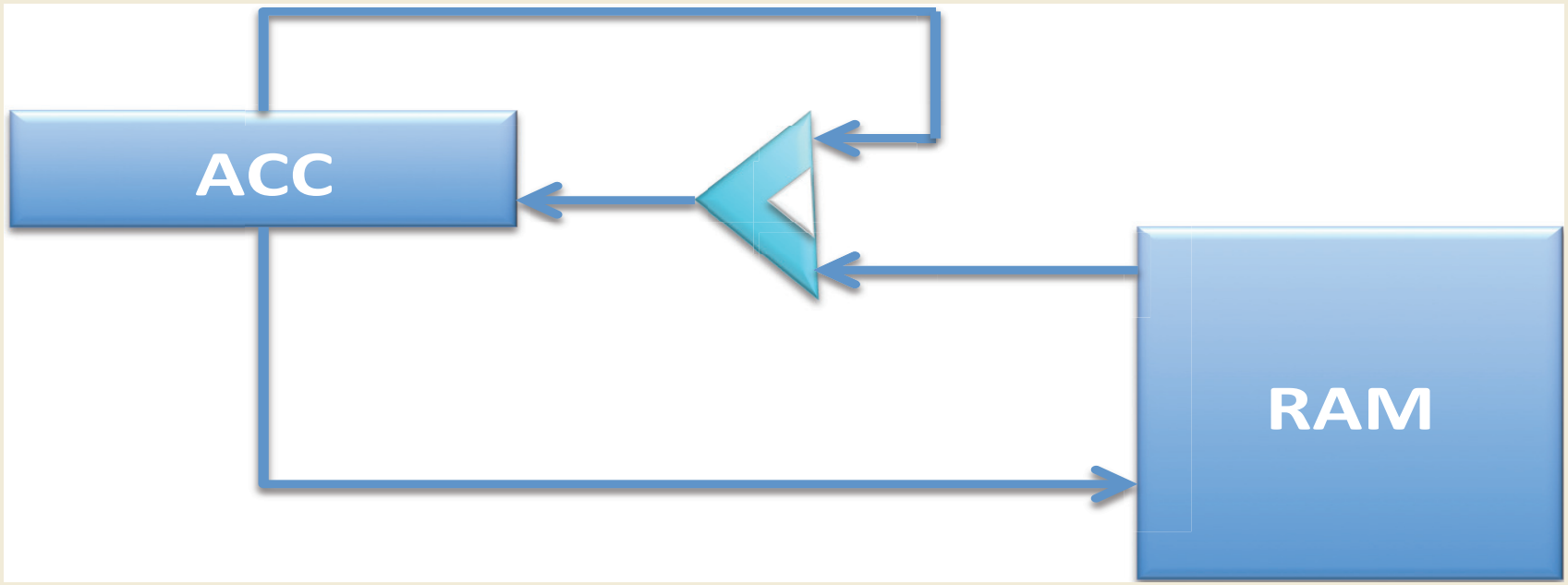
Architecture d'un processeur DSP

Diagramme du processeur TMS320C5416



Architectures des unités de traitement

Couplage accès mémoire et calcul



Types d'instructions

Exemple: Calcul d'un filtre FIR

- $$y[n] = \Sigma_{i=0}^{N-1} b[i]x[n-i]$$

Types d'instructions

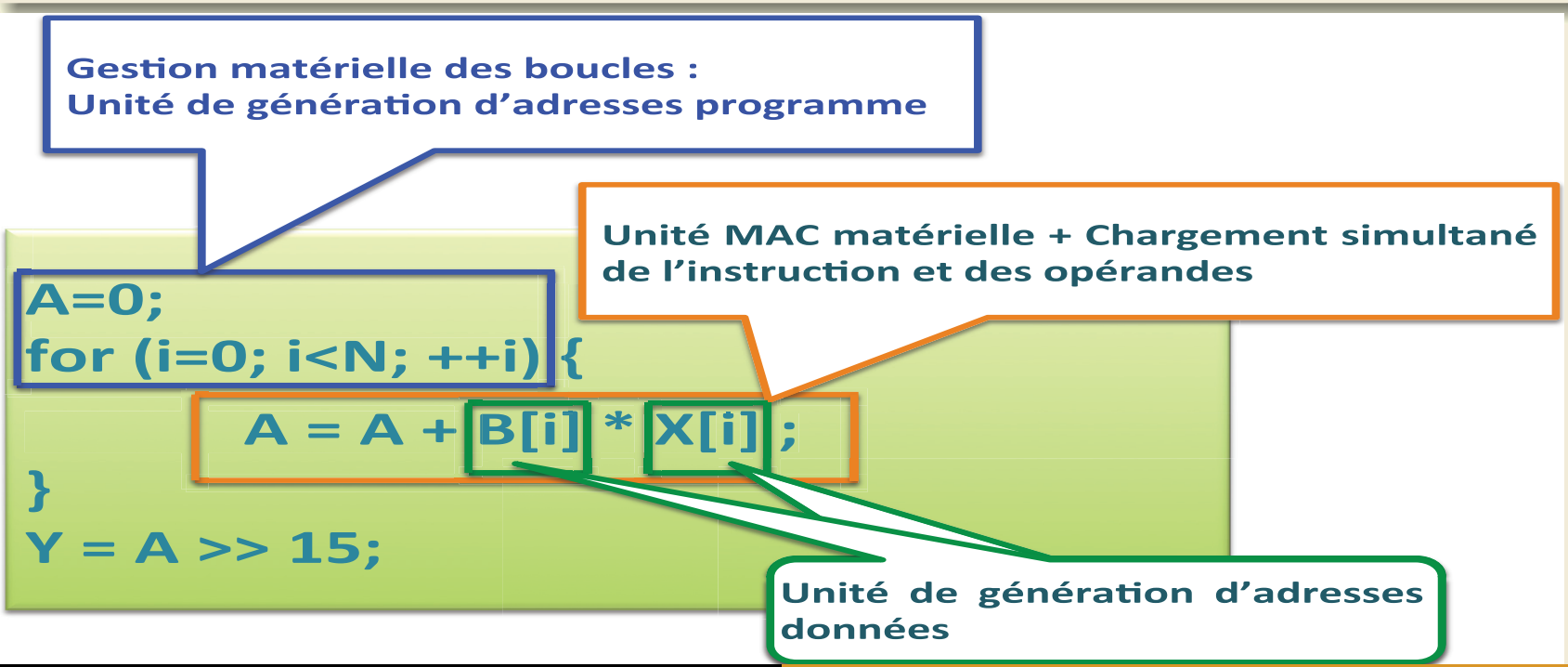
Exemple: Calcul d'un filtre FIR

- $$y[n] = \Sigma_{i=0}^{N-1} b[i]x[n-i]$$
- $y[n] \rightarrow Y$, $b[i] \rightarrow B[i]$, $x[i] \rightarrow X[i]$

Types d'instructions

Exemple: Calcul d'un filtre FIR

- $$y[n] = \Sigma_{i=0}^{N-1} b[i]x[n-i]$$
- $y[n] \rightarrow Y$, $b[i] \rightarrow B[i]$, $x[i] \rightarrow X[i]$



Utilisation de l'adressage indirect

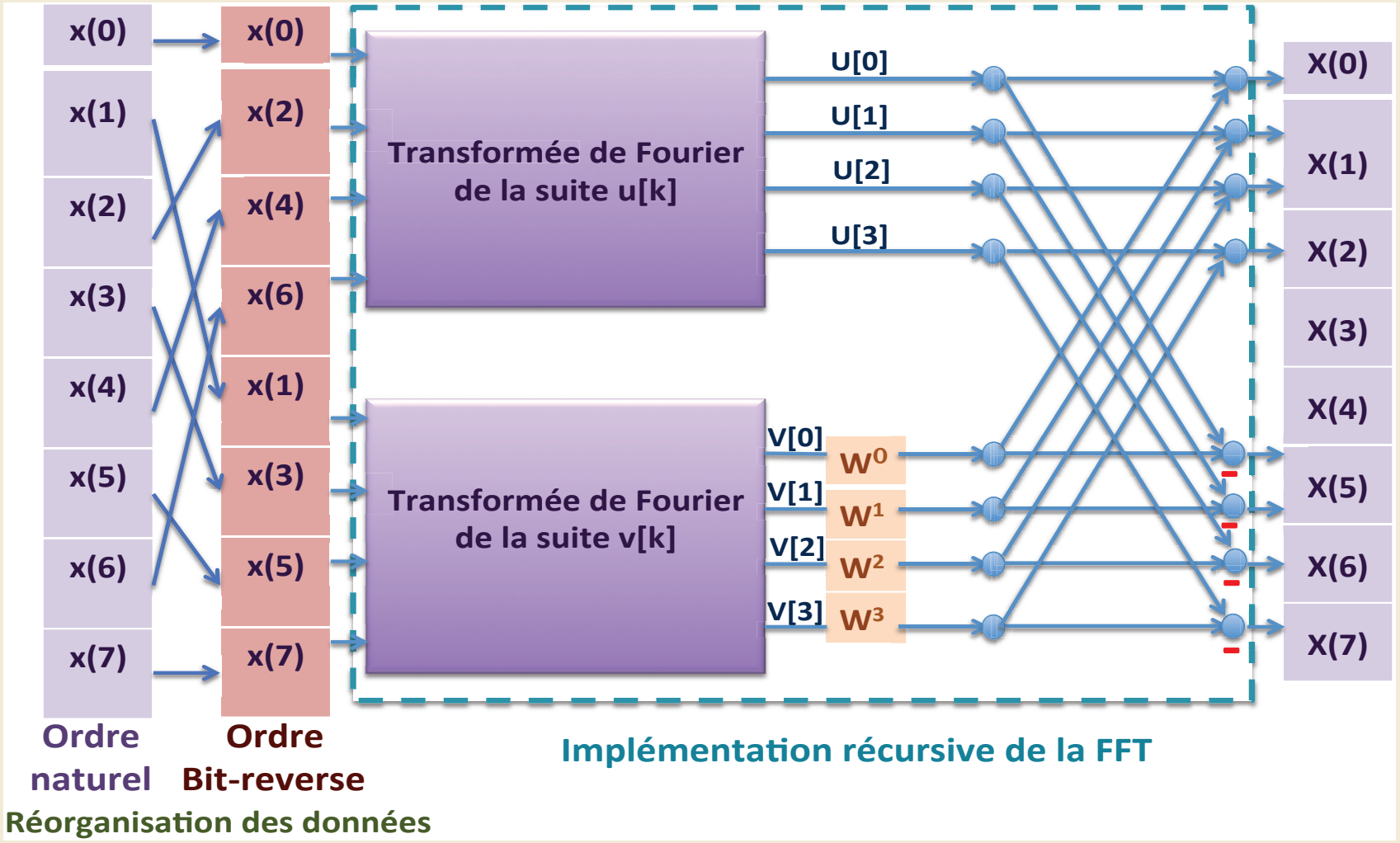
Produit scalaire entre deux vecteurs

```
long A;
int X[100],Y[100];
int *px, *py;
px=X;
py=Y;
A=0;
for (i=0; i<100; ++i)
A = A + (long)(*px++)*(*py++);
```

X[i] Y[i]

Adressage indirect bit-reverse

Ordre des données pour la FFT



Pipelining

- Instructions segmentées en étages

Architecture d'un processeur DSP

Pipelining

- Instructions segmentées en étages : Chacune à un étage différent

Architecture d'un processeur DSP

Pipelining

- Instructions segmentées en étages : Chacune à un étage différent
- Exécution entrelacée de plusieurs instructions

Architecture d'un processeur DSP

Pipelining

- Instructions segmentées en étages : Chacune à un étage différent
- Exécution entrelacée de plusieurs instructions Etages plus simples donc plus rapides

Architecture d'un processeur DSP

Pipelining

- Instructions segmentées en étages : Chacune à un étage différent
- Exécution entrelacée de plusieurs instructions Etages plus simples donc plus rapides
- Augmentation de la fréquence d'horloge

Architecture d'un processeur DSP

Pipelining

- Instructions segmentées en étages : Chacune à un étage différent
- Exécution entrelacée de plusieurs instructions Etages plus simples donc plus rapides
- Augmentation de la fréquence d'horloge
- Géré par le séquenceur

Pipelining

Exemple du TMS320C54xx : Pipeline à 6 étages

- Prefetch (P): Incrémentation du PC (Program Counter)

Pipelining

Exemple du TMS320C54xx : Pipeline à 6 étages

- Prefetch (P): Incrémentation du PC (Program Counter)
- Fetch (F): Lecture de l'instruction en mémoire

Pipelining

Exemple du TMS320C54xx : Pipeline à 6 étages

- Prefetch (P): Incrémentation du PC (Program Counter)
- Fetch (F): Lecture de l'instruction en mémoire
- Decode (D): Décodage de l'instruction

Pipelining

Exemple du TMS320C54xx : Pipeline à 6 étages

- Prefetch (P): Incrémentation du PC (Program Counter)
- Fetch (F): Lecture de l'instruction en mémoire
- Decode (D): Décodage de l'instruction
- Access (A): Calcul des adresses des opérandes

Pipelining

Exemple du TMS320C54xx : Pipeline à 6 étages

- Prefetch (P): Incrémentation du PC (Program Counter)
- Fetch (F): Lecture de l'instruction en mémoire
- Decode (D): Décodage de l'instruction
- Access (A): Calcul des adresses des opérandes
- Read (R)

Pipelining

Exemple du TMS320C54xx : Pipeline à 6 étages

- Prefetch (P): Incrémentation du PC (Program Counter)
- Fetch (F): Lecture de l'instruction en mémoire
- Decode (D): Décodage de l'instruction
- Access (A): Calcul des adresses des opérandes
- Read (R)
  - Lecture des opérandes en mémoire

Pipelining

Exemple du TMS320C54xx : Pipeline à 6 étages

- Prefetch (P): Incrémentation du PC (Program Counter)
- Fetch (F): Lecture de l'instruction en mémoire
- Decode (D): Décodage de l'instruction
- Access (A): Calcul des adresses des opérandes
- Read (R)
  - Lecture des opérandes en mémoire
  - Calcul de l'adresse du résultat

Pipelining

Exemple du TMS320C54xx : Pipeline à 6 étages

- Prefetch (P): Incrémentation du PC (Program Counter)
- Fetch (F): Lecture de l'instruction en mémoire
- Decode (D): Décodage de l'instruction
- Access (A): Calcul des adresses des opérandes
- Read (R)
  - Lecture des opérandes en mémoire
  - Calcul de l'adresse du résultat
- Execute (X)

Pipelining

Exemple du TMS320C54xx : Pipeline à 6 étages

- Prefetch (P): Incrémentation du PC (Program Counter)
- Fetch (F): Lecture de l'instruction en mémoire
- Decode (D): Décodage de l'instruction
- Access (A): Calcul des adresses des opérandes
- Read (R)
  - Lecture des opérandes en mémoire
  - Calcul de l'adresse du résultat
- Execute (X)
  - Exécution du calcul

Pipelining

Exemple du TMS320C54xx : Pipeline à 6 étages

- Prefetch (P): Incrémentation du PC (Program Counter)
- Fetch (F): Lecture de l'instruction en mémoire
- Decode (D): Décodage de l'instruction
- Access (A): Calcul des adresses des opérandes
- Read (R)
  - Lecture des opérandes en mémoire
  - Calcul de l'adresse du résultat
- Execute (X)
  - Exécution du calcul
  - Ecriture en mémoire

Pipelining

Séquentiel vs pipeline

durée	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8
Instruction	Instruction1				Instruction2			
Fetch	F_1				F_2			
Decode		D_1				D_2		
Read			R_1				R_2	
Execute				X_1				X_2
Exécution séquentielle								

Pipelining

Séquentiel vs pipeline

durée	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8
Instruction	Instruction1				Instruction2			
Fetch	F_1				F_2			
Decode		D_1				D_2		
Read			R_1				R_2	
Execute				X_1				X_2
Exécution séquentielle								
heightFetch	F_1	F_2	F_3	F_4	F_5			
Decode	**	D_1	D_2	D_3	D_4	D_5		
Read	**		R_1	R_2	R_3	R_4	R_5	
Execute	**			X_1	X_2	X_3	X_4	X_5
Exécution avec pipeline: entrelacement des instructions								
** Amorçage du pipline								

Architecture d'un processeur DSP

Pipelining

Principe

Découper une opération en tâches élémentaires à effectuer en parallèle

Architecture d'un processeur DSP

Pipelining

Principe

Découper une opération en tâches élémentaires à effectuer en parallèle

Avantages

Gain en vitesse d'exécution

Architecture d'un processeur DSP

Pipelining

Principe

Découper une opération en tâches élémentaires à effectuer en parallèle

Avantages

Gain en vitesse d'exécution

Inconvénients

Electronique et programmation plus complexe

Architecture d'un processeur DSP

Pipelining

Principe

Découper une opération en tâches élémentaires à effectuer en parallèle

Avantages

Gain en vitesse d'exécution

Inconvénients

Electronique et programmation plus complexe

Un retard peut se produire si:

Architecture d'un processeur DSP

Pipelining

Principe

Découper une opération en tâches élémentaires à effectuer en parallèle

Avantages

Gain en vitesse d'exécution

Inconvénients

Electronique et programmation plus complexe

Un retard peut se produire si:

- S'il existe un conflit de ressources

Architecture d'un processeur DSP

Pipelining

Principe

Découper une opération en tâches élémentaires à effectuer en parallèle

Avantages

Gain en vitesse d'exécution

Inconvénients

Electronique et programmation plus complexe

Un retard peut se produire si:

- S'il existe un conflit de ressources
- En cas de rupture de séquence (interruption)

Architecture d'un processeur DSP

Représentation des nombres

Les valeurs traitées (coefficients, échantillons...) sont représentées sous 2 formes:

- Virgule fixe

Architecture d'un processeur DSP

Représentation des nombres

Les valeurs traitées (coefficients, échantillons...) sont représentées sous 2 formes:

- Virgule fixe
- Virgule flottante

Représentation des nombres

- Virgule fixe: Complément à 2 (C_2)

Représentation des nombres

- Virgule fixe: Complément à 2 (C_2)

Nombre	Codage		
3	0	1	1
2	0	1	0
1	0	0	1
0	0	0	0
-1	1	1	1
-2	1	1	0
-3	1	0	1
-4	1	0	0
	Signe		

Représentation des nombres

- Virgule fixe: Complément à 2 (C_2)

Nombre	Codage		
3	0	1	1
2	0	1	0
1	0	0	1
0	0	0	0
-1	1	1	1
-2	1	1	0
-3	1	0	1
-4	1	0	0
	Signe		

- Nombre positif ($2^{N-1}-1$): Codé comme un binaire normal

Représentation des nombres

- Virgule fixe: Complément à 2 (C_2)

Nombre	Codage		
3	0	1	1
2	0	1	0
1	0	0	1
0	0	0	0
-1	1	1	1
-2	1	1	0
-3	1	0	1
-4	1	0	0
	Signe		

- Nombre positif ($2^{N-1}-1$): Codé comme un binaire normal
- Nombre négatif (-2^{N-1}): Inversion des bits puis ajout de 1

Représentation des nombres

- Virgule fixe: Codage des entiers

Représentation des nombres

- Virgule fixe: Codage des entiers

Exemple: Sur 8 bits

$$x = -b_7 * 2^7 + \Sigma_{i=0}^{6} b_i * 2^i$$

Poids	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
27	0	0	0	1	1	0	1	1
0	0	0	0	0	0	0	0	0
-4	1	1	1	1	1	1	0	0

	Bit signe

Architecture d'un processeur DSP

Représentation des nombres

- Virgule fixe: Codage des réels $\rightsquigarrow$ fomat Q_k

Architecture d'un processeur DSP

Représentation des nombres

- Virgule fixe: Codage des réels $\rightsquigarrow$ fomat Q_k

Exemple: $\underbrace{2}$, $\underbrace{3125}$

- Partie entière sur $n-k$ bits en $C2$

Architecture d'un processeur DSP

Représentation des nombres

- Virgule fixe: Codage des réels $\rightsquigarrow$ fomat Q_k

Exemple: $\underbrace{2}$, $\underbrace{3125}$

- Partie entière sur $n-k$ bits en $C2$
- Partie fractionnaire sur k bits

Représentation des nombres

- Virgule fixe: Codage des réels $\rightsquigarrow$ fomat Q_k
- Pour coder le réel x, trouver les b_i tels que:

$$x = -b_n * 2^{n-k} + \Sigma_{i=-k}^{n-k-1} b_{i+k} * 2^i$$

Architecture d'un processeur DSP

Représentation des nombres

- Virgule fixe: Codage des réels $\rightsquigarrow$ fomat Q_k
- Pour coder le réel x, trouver les b_i tels que:

$$x = -b_n * 2^{n-k} + \Sigma_{i=-k}^{n-k-1} b_{i+k} * 2^i$$

Exemple: $\underbrace{2}$, $\underbrace{3125}$

- Ex: format Q_5 sur 8 bits (n=8, k=5)

Représentation des nombres

- Virgule fixe: Codage des réels $\rightsquigarrow$ fomat Q_k
- Pour coder le réel x, trouver les b_i tels que:

$$x = -b_n * 2^{n-k} + \Sigma_{i=-k}^{n-k-1} b_{i+k} * 2^i$$

Exemple: $\underbrace{2}$, $\underbrace{3125}$

- Ex: format Q_5 sur 8 bits (n=8, k=5)
- Partie entière: 3 bits / fractionnaire: 5 bits

Représentation des nombres

- Virgule fixe: Codage des réels $\rightsquigarrow$ fomat Q_k

Représentation des nombres

- Virgule fixe: Codage des réels $\rightsquigarrow$ fomat Q_k

Exemple $\underbrace{2}$, $\underbrace{3125}$

- Correspond à la représentation C2 de l'entier y tel que: $y = round(2^k x)$

Représentation des nombres

- Virgule fixe: Codage des réels $\rightsquigarrow$ fomat Q_k

Exemple $\underbrace{2}$, $\underbrace{3125}$

- Correspond à la représentation C2 de l'entier y tel que: $y = round(2^k x)$
- $y = round(2^5 * 2,3125) = 74 = (01001010)_b$

Poids	-2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}
2,3125	0	1	0	0	1	0	1	0

	Bit signe

Mesure de performances

- Benchmark:

Architecture d'un processeur DSP

Mesure de performances

- Benchmark:
  - Mesurer le temps que met le DSP pour exécuter des programmes "standard" de TNS

Architecture d'un processeur DSP

Mesure de performances

- Benchmark:
  - Mesurer le temps que met le DSP pour exécuter des programmes "standard" de TNS
- Comment les choisir ?

Architecture d'un processeur DSP

Mesure de performances

- Benchmark:
  - Mesurer le temps que met le DSP pour exécuter des programmes "standard" de TNS
- Comment les choisir ?
- Dépendent des domaines

Architecture d'un processeur DSP

Mesure de performances

- Benchmark:
  - Mesurer le temps que met le DSP pour exécuter des programmes "standard" de TNS
- Comment les choisir ?
- Dépendent des domaines