Cours
Architectures Avancées

Prof. OUADOU M.

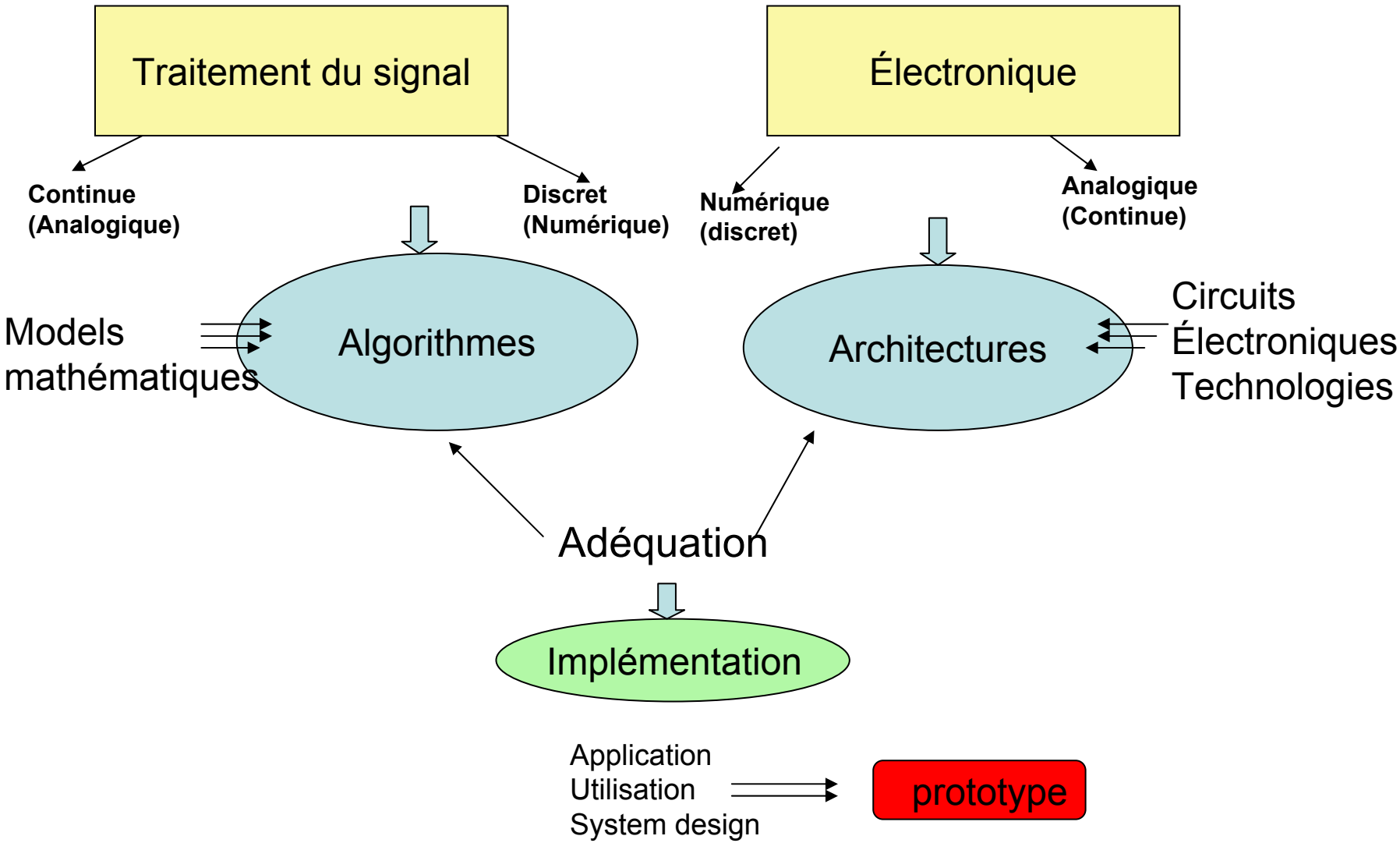
Département de Physique
Faculté des Sciences Rabat

Année 2006-2007

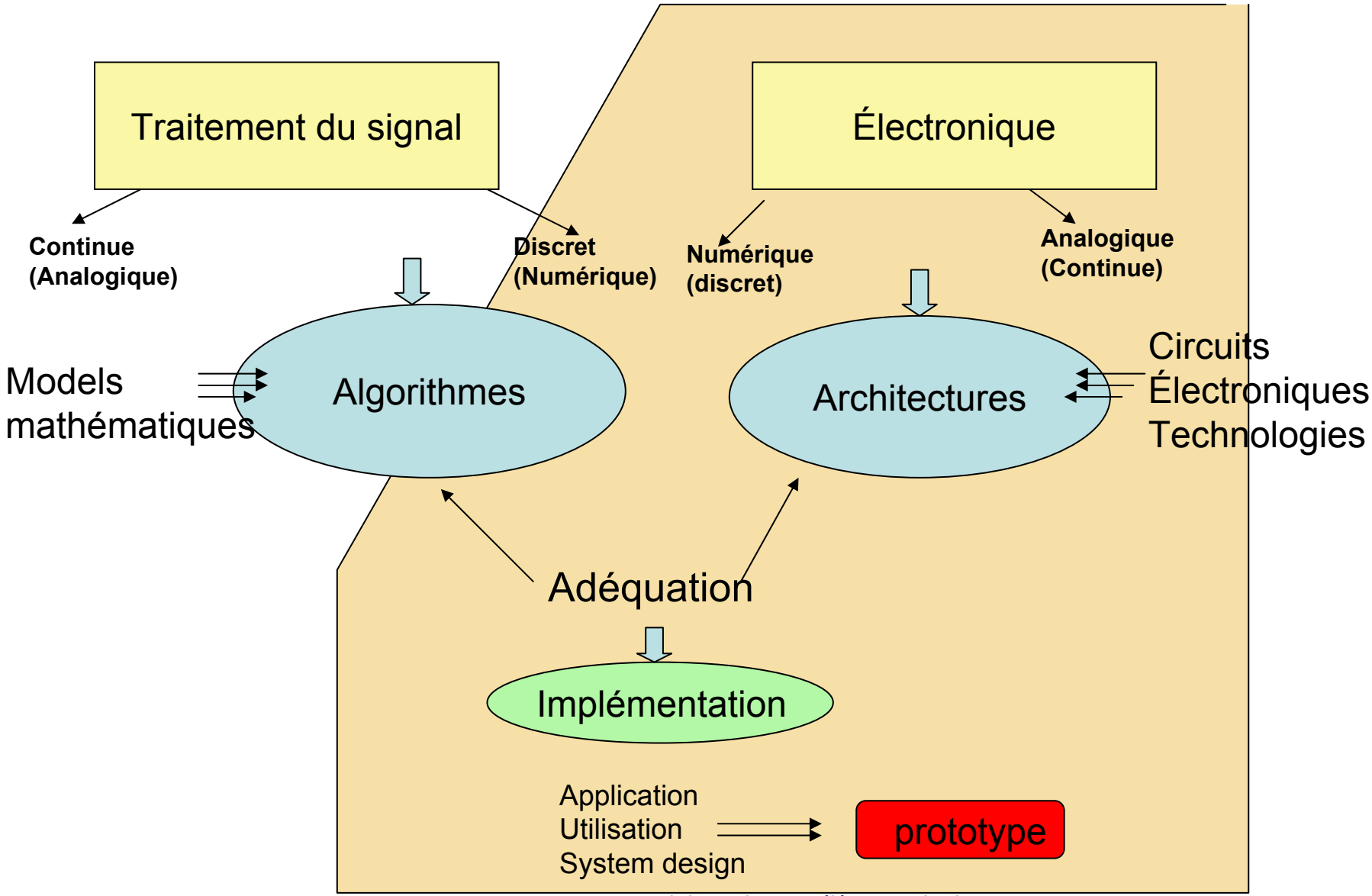
Sommaire

- **Rappels**
  - **Électricité**
  - **Électronique Analogique**
  - **Électronique Numérique**
- **Architecture des processeurs**
  - **Les microprocesseurs**
  - **L'architecture du 6811 de Motorola**
  - **Les algorithmes de traitement du signal et de l'image**
  - **Les architectures des processeurs DSP**
  - **La familles des DSP TMS 320**
    - * outils de développements
    - * programmation
    - * méthodologie d'implémentation
  - **Applications**

LE DOMAINE


Traitement du signal
Électronique
Continue
(Analogique)
Discret
(Numérique)
Numérique
(discret)
Analogique
(Continue)
Models
mathématiques
Algorithmes
Architectures
Circuits
Électroniques
Technologies
Adéquation
Implémentation
Application
Utilisation
System design
prototype

LE DOMAINE


Traitement du signal
Électronique
Continue
(Analogique)
Discret
(Numérique)
Numérique
(discret)
Analogique
(Continue)
Models
mathématiques
Algorithmes
Architectures
Circuits
Électroniques
Technologies
Adéquation
Implémentation
Application
Utilisation
System design
prototype

Domaines d'application

- **Calcul général**

  - Ordinateurs et micro-ordinateurs
  - Calculatrices

- **Traitement du signal**

  - Télécommunications (réseaux et GSM)
  - Traitement des signaux sonores, de la parole et la musique
  - Signaux sismiques
  - Radars (poursuite de cible)
  - Navigation

- **Traitement du signal**

  - TV , vidéo et photo numérique
  - Traitement et filtrage de l'image
  - Compression de l'image (pour le stockage et la transmission)
  - Détection d'objets contenus dans une image
  - Reconnaissance de forme et du visage et de l'individu
  - Reconnaissance des caractères

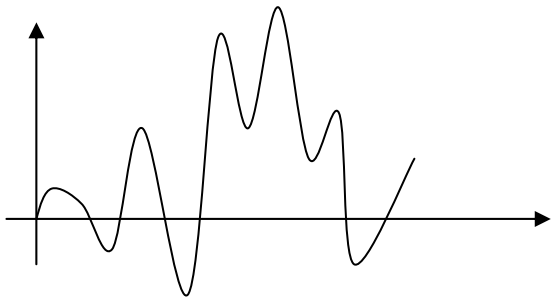
Domaines d'application

- **Automatique**
  - Identification
  - Contrôle adaptatif
  - Surveillance des processus industriels
- **Instrumentation**
  - **Oscilloscopes**
  - Microscope électronique
  - Appareils de mésure
- **Electronique domestique**
- **Systèmes de surveillance**
- **Avions, navires, auto**
- **…** etc.

Rappels

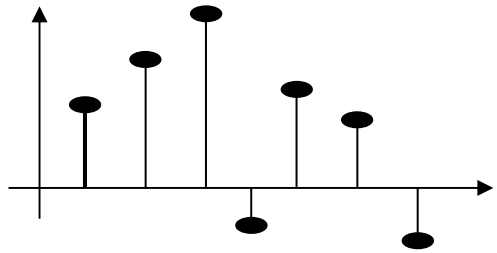
Signal analogique

continue



Signal numérique

discret



Signal ana.

Processeur de traitement analogique

Signal ana.

Signal ana.

A/D

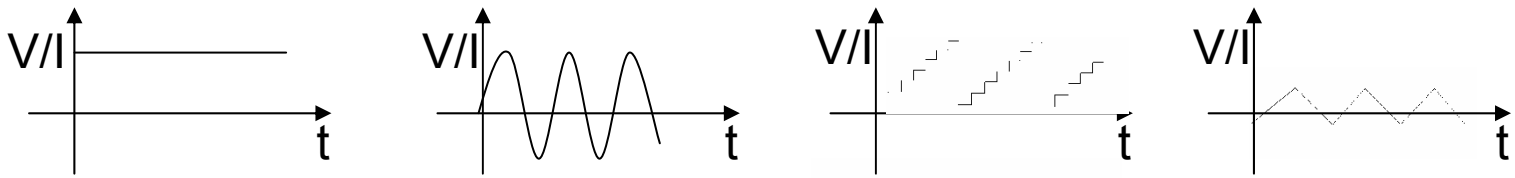
Processeur de traitement Numérique

D/A

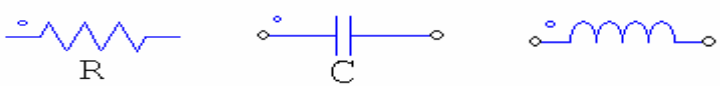
Signal an

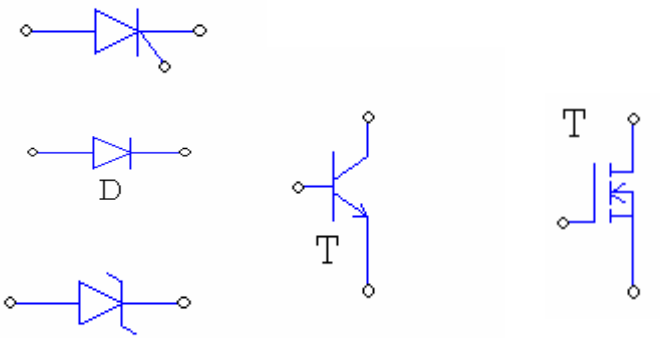
Rappels

Courants et tensions :

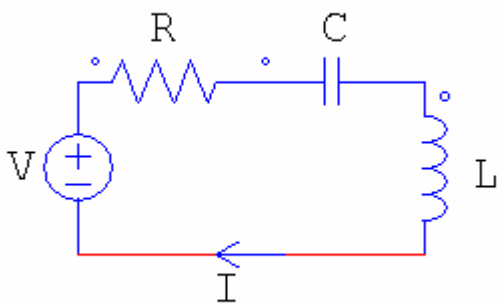


Les éléments :





Les circuits :



$$V=R*I + I*Z_C + I*Z_L$$

Rappels

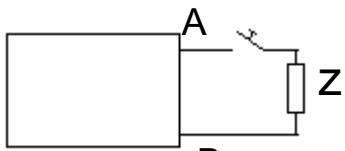
Électronique analogique

Loi d' Hom : $U / I = R$

Lois de Kirchov : $\sum I_k = 0$ $\sum R_k I_k = \sum U_k$

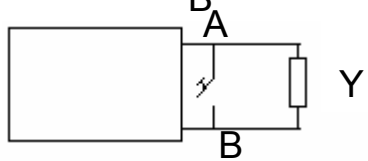
Théorèmes :

Thevenin



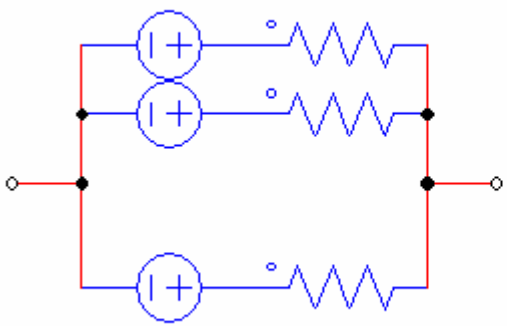
$I=V_{ab} / (Z+Z_{ab})$

Norton

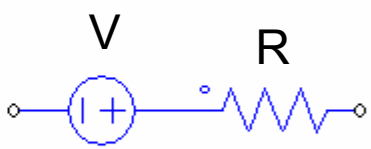


$V_y=I_{ab} / (Y+Y_{ab})$

Millman



≈



$V=(1/ \sum (1/R_i)) \sum Vi/R_i$

Outils mathématiques :

Transformation de Laplace **Représentation de Bode**

Rappels

Électronique analogique

L'outil mathématique principal de calcul :

La transformation de Laplace

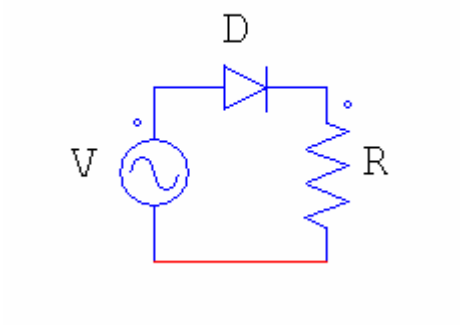
L'outil principal de représentation graphique:

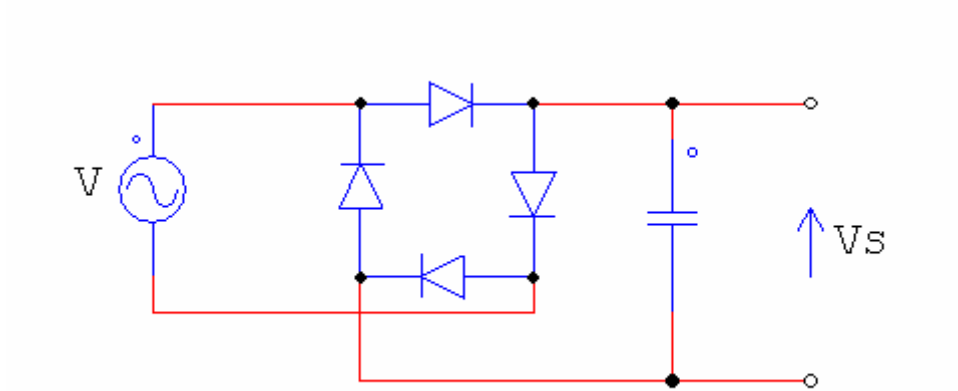
La représentation de Bode

Rappels

Circuits à base de diodes :

Redressement

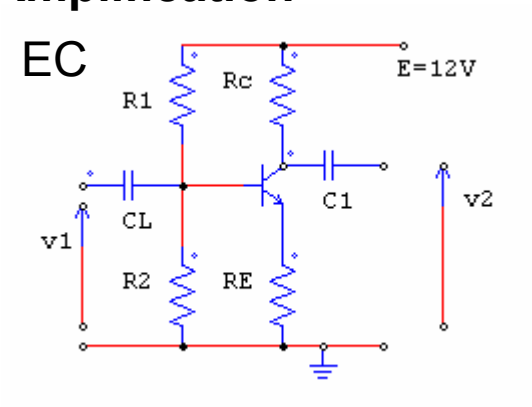




Circuits à base de transistors :

Amplification

EC

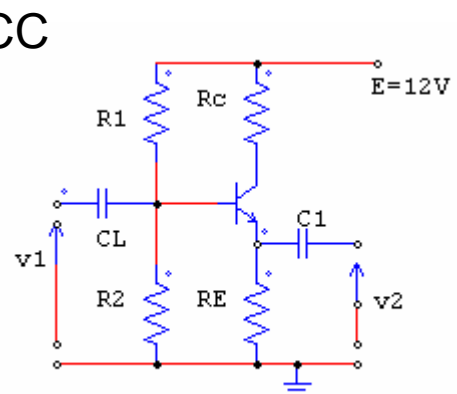


Vs=Kv * Ve

Is=Ki * Ie

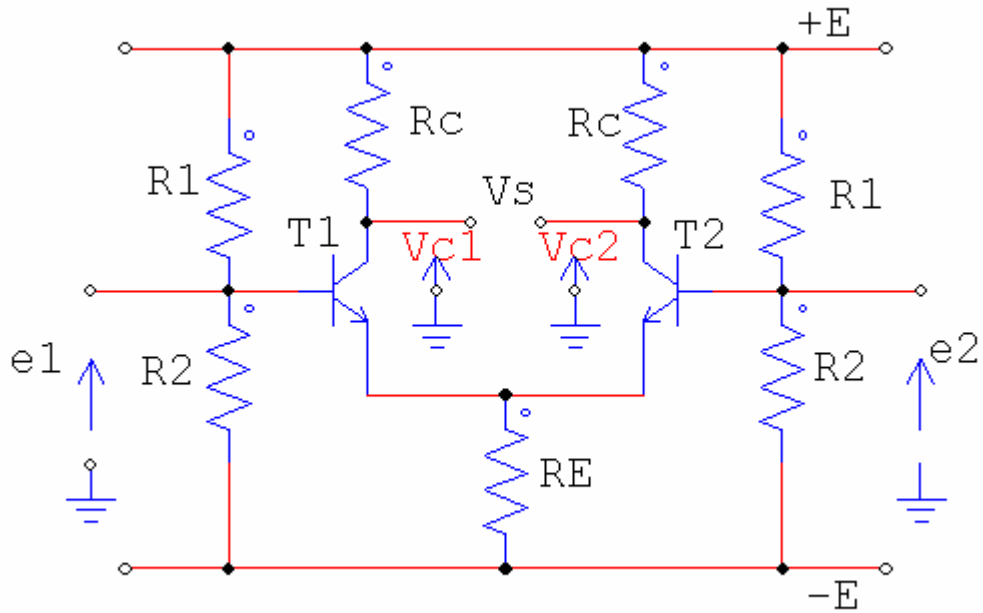
Ze, Zs

CC

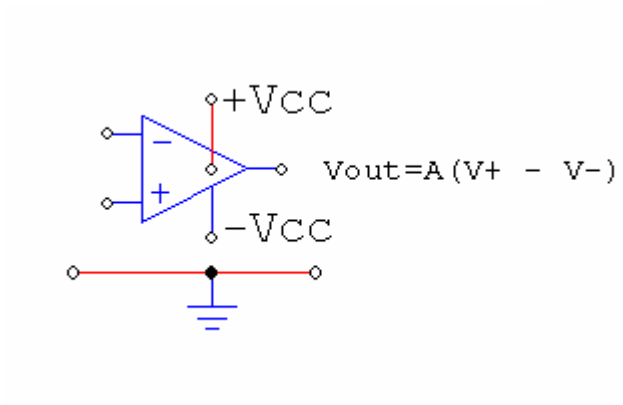


Rappels

Les amplificateurs différentielles



Les amplificateurs opérationnels



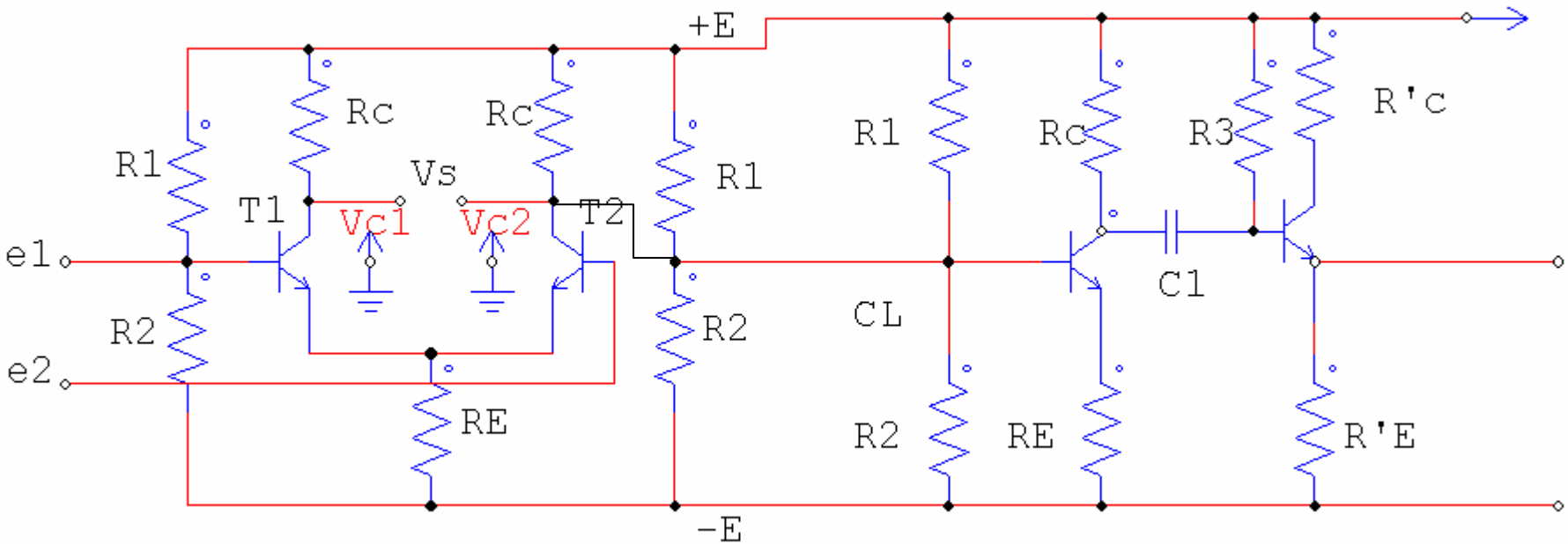
Gain

Impédance

Amplification AC et DC

Dérives

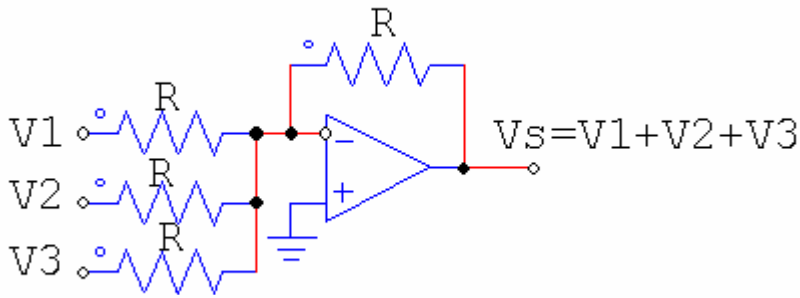
L'amplificateur opérationnel



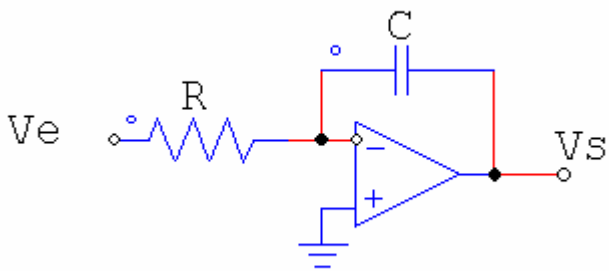
Rappels

Réaction négative

Sommateur

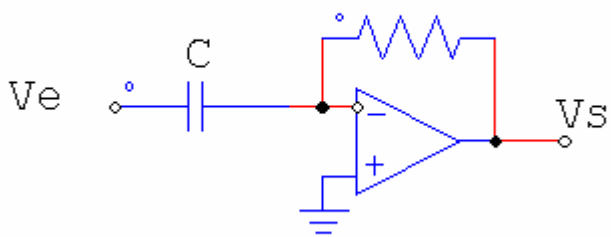


Intégrateur



$$Vs = \int Ve\, dt$$

Différentiateur



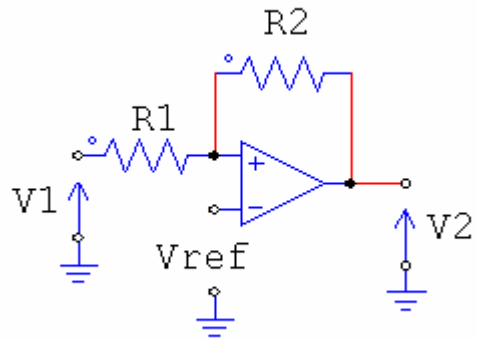
$$Vs = dVe/dt$$

Filtres

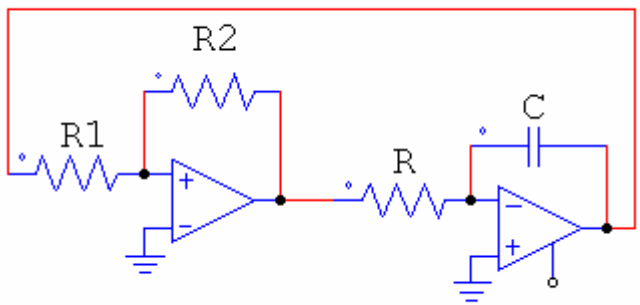
Rappels

Réaction positive

Trigger de Schmitt



Générateurs de signaux



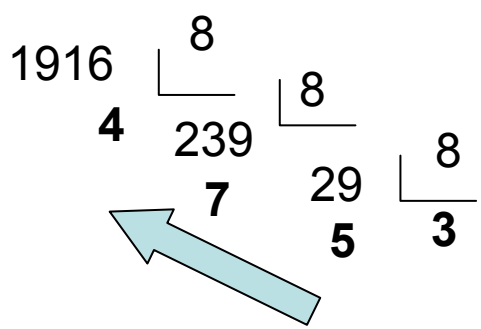
Filtres

La numération et représentation des nombres

La numération est la science qui traite de la dénomination et de la représentation graphique des nombres.

Le problème qui se pose est de représenter tous les entiers naturels et les décimaux à l'aide d'un ensemble fini de symboles (souvent des chiffres) rassemblés selon des règles (le code).

$$(2563)_{10}=2x10^3 + 5x\ 10^2 + 6x10^1 + 3x10^0$$



$(1916)_{10} = (3574)_8$

Codage binaire des nombres entiers

$$A = (N)_b = a_n \ldots a_i \ldots a_1 a_0 = a_n b^n + \ldots a_i b^i + \ldots a1b^1 + a_0 b^0$$

$$= \sum a_i \, . \, b^i$$

Où b est la base

La base 2: **b=2**

$a_i \in \{ 0 , 1 \}$

$A \in \{ 0 , 2^n - 1 \}$

Ex. passage de la base 10 à la base 2

Codage binaire des nombres décimaux

Codage binaire des nombres entiers relatifs

Définitions :

Chiffre binaire ou **bit** (Binary digit) : la plus petite unité d'information
Binaire de valeur 0 ou 1

Octet (**byte**) : nombre binaire de 8 bits

Mot (**word**) : élément d'information mémorisé ou traité d'un seul bloc.
(16 , 32 , 64 … etc.)

Le traitement de l'information dans les calculateurs (processeurs) s'effectue
Sur des mots de 8 bits 16, 32, … etc.
Il faut aussi représenter les nombres positives et les nombres négatives.

Pour 8 bits :

C'est le bit le plus fort qui représente le signe du mot traité.

$$A = - a_{n-1} \times 2^{n-1} + \sum_{i=0}^{N-2} a_i \times 2^i \; ; \qquad a_i \in \{ 0 , 1 \}$$

$$A \in [-2^{n-1} , + 2^{n-1} - 1]$$

$$A = - a_{n-1} \times 2^{n-1} + \sum_{i=0}^{6} a_i \times 2^i$$

a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
-	+	+	+	+	+	+	+
-128	+64	+32	+16	+8	+4	+2	+1

Les valeurs max et min ?

Deux représentations des nombres :

* Virgule fixe

Avec 8 bits, 28 nombres = 256 combinaisons différentes

La précision = 1 / 256

a7	a6	a5	a4	a3 ,	a2	a1	a0

Avec plusieurs bits on a plus de précision

++ rapidité de calcul

-- dynamique limitée (…)

* Virgule flottante

Tout nombre est représenté sous forme :

$$N = M \cdot b^E$$

Ex: b_{10}, b_2

SM	SE	e3	e2	e1	e0	.m9	m8	m7	m6	m5	m4	m3	m2	m1	m0

Exposant (4) — Mantisse(10) (en virgule fixe)

++ dynamique large

++ plus de précision

-- plus de temps de calcul

Ex: representer (+2.5)10 sur 2 octets (16 bits)

Le code décimal code binaire (DCB ou BCD)

Théorèmes fondamentaux de l'algèbre de BOOLE

BOOLE 1847

Algèbre qui s'applique à des fonctions logiques de variables logiques.
(variables Booléennes)

Toute fonction logique peut être réalisée à l'aide d'un petit nombre de fonctions logiques de base appelées **opérateurs logiques** ou **portes** (**gates**).

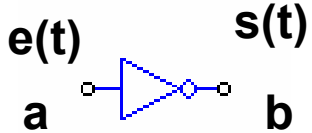
Principaux sont: **NON, ET, OU + XOR, NAND, NOR**

Les théorèmes:

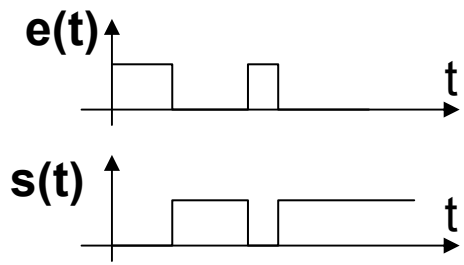
Théorèmes des constantes	$a + 0 = a$ $a + 1 = 1$	$a \times 0 = 0$ $a \times 1 = 1$
Idempotence	$a + a = a$	$a \times a = a$
Complémentation	$a + \overline{a} = 1$	$a \times \overline{a} = 0$
Commutativité	$a + b = b + a$	$a \times b = b \times a$
Distributivité	$a + (b\ c) = (a + b)(a + c)$ $a\ (b + c) = (a\ b) + (a\ c)$	
Associativité	$a + (b + c) = (a + b) + c = a + b + c$ $a\ (b\ c) = (a\ b)\ c = a\ b\ c$	
Autres relations	$a = \overline{\overline{a}}$ $a + (\overline{a}\ b) = a + b$ $(a + b)(a + \overline{b}) = a$	$a + (a\ b) = a$ $a\ (a + b) = a$

Les circuits logiques élémentaires (représentation des fonctions binaires)

La complémentation (inversion ou négation ou pas) : NON (NO)

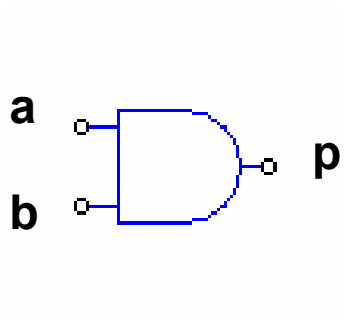


a	b
0	1
1	0

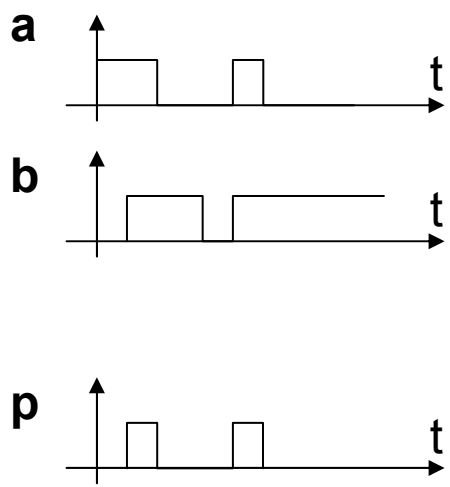


Les circuits logiques élémentaires

La produit (intersection ou multiplication logique) : ET (AND)

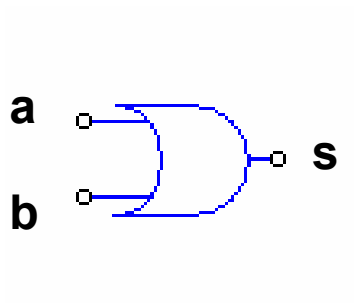


a	b	p
0	0	0
0	1	0
1	0	0
1	1	1

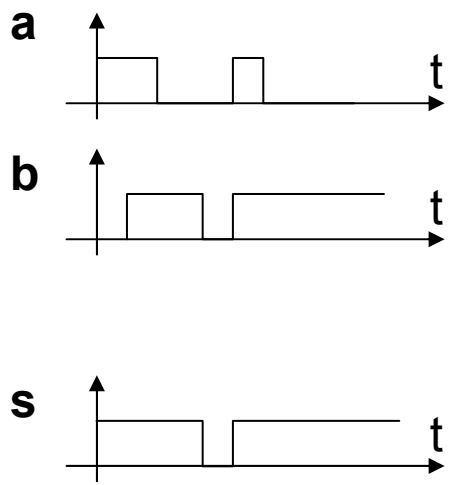


Les circuits logiques élémentaires

La produel (réunion ou addition logique) : OU (OR)

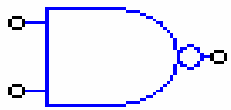


a	b	s
0	0	0
0	1	1
1	0	1
1	1	1



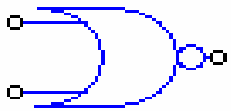
Les combinaisons :

Non Et



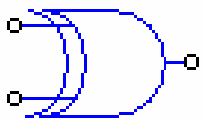
NAND

Non Ou



NOR

Ou exclusif



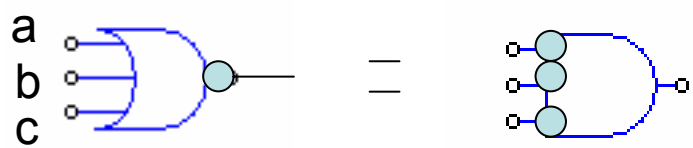
XOR

$$a \oplus b = a\,\overline{b} + \overline{a}\,b$$

Les théorèmes de Morgan:

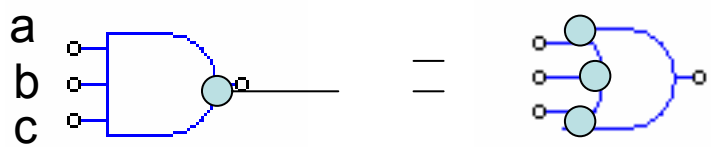
Le complément d'un produel est égal au produit des compléments des facteurs qui le composent.

$$\overline{a + b + c + \ldots + q} = \bar{a}\,\bar{b}\,\bar{c}\,\bar{d}\,\ldots\bar{q}$$



Le complément d'un produit est égal au produel des compléments des facteurs qui le composent.

$$\overline{a\,b\,c\,d\,\ldots q} = \bar{a} + \bar{b} + \bar{c} + \ldots + \bar{q}$$

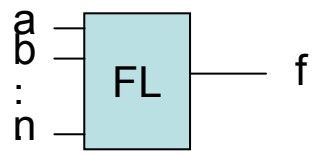


Les paramètres de base d'un élément logique

1 – **La fonction logique réalisée**

2 – **Le coefficient de liaison en entrée** : (nombre maximal d'entrées logiques de l'élément) (dépend de la technologie)

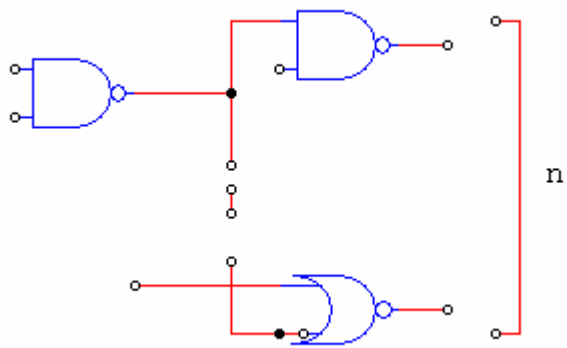
pour TTL n = 8



3 – **Le coefficient de liaison en sortie** : nombre maximal de branchement de la sortie

pour TTL (n= 4 - 10)

pour circuits spéciales n=30

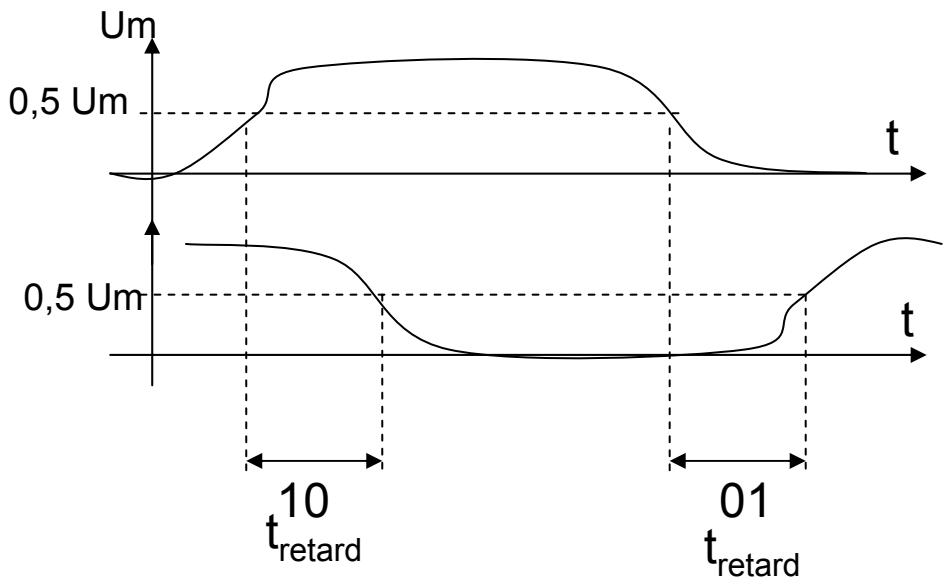


Les paramètres de base d'un élément logique

4 – La rapidité de fonctionnement

temps de réponse moyen:

$$t_{rep.\ moy.} = (t^{10}_{ret} + t^{01}_{ret}) / 2$$



* Le temps de transition

$$t = \max \{ t^{10}_{ret} , t^{01}_{ret} \}$$

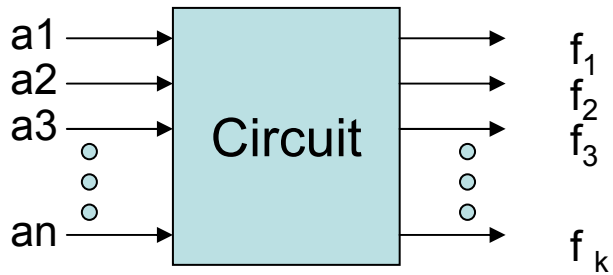
* La fréquence maximale

$$f_{max} = 1 / 2t$$

Les circuits numériques

Les circuits logiques 2 états d'équilibre = 2 niveaux de tension ou courant.

Les circuits logiques sont des dispositifs qui exécutent des opérations sur des variables logiques, transportent et traitent des signaux logiques.



- **Les circuits combinatoires** : circuits idéalisés ou le temps de propagation des signaux n'est pas pris en considération. Les signaux de sortie ne dépendent que des signaux d'entrée, appliqués à l'instant considéré.

- Les circuits séquentiels : circuits où il faut tenir compte du temps de la propagation des signaux et de la mémoire du circuit .

Les signaux de sortie dépendent même des signaux d'entrée appliqués ultérieurement.

Les circuits numériques

La fonction logique d'un circuit peut se définir soit par :

1- tableau de correspondance = table de vérité

2- diagrammes de temps (temporaire)

3- expressions algébriques

4- schéma

Les circuits combinatoires

La synthèse d'un circuit combinatoire:

La synthèse d'un circuit destiné à réaliser une fonction binaire donnée comprend trois étapes:

- Construire la table de vérité de la fonction logique.

- Écriture de l'expression de la fonction binaire.

- Simplification de l'expression en vue d'obtenir un circuit économique ou un circuit à temps de traversée minimal.

- Passage de l'écriture symbolique de l'expression simplifiée au schéma électronique du circuit.

Les circuits numériques

Les formes canoniques des fonctions binaires:

Toute fonction binaire peut s'exprimer:

-Soit par un produel de produits.

- Soit par un produit de produel.

en faisant intervenir toutes les variables directs ou complémentées

D'une façon general:

$$F = P_0+P_1+P_2+\ldots\ldots+P_N$$

$$F = S_0S_1S_2\ldots\ldots S_N$$

Forme canonique disjonctive

Forme canonique conjonctive

C'est deux expressions sont les deux formes canoniques de la fonction.

Avec: P= minterme S= maxterme

Simplification et minimisation des fonctions booléennes

Les formes canoniques des fonctions représentent des façons relativement compliquées d'écriture.

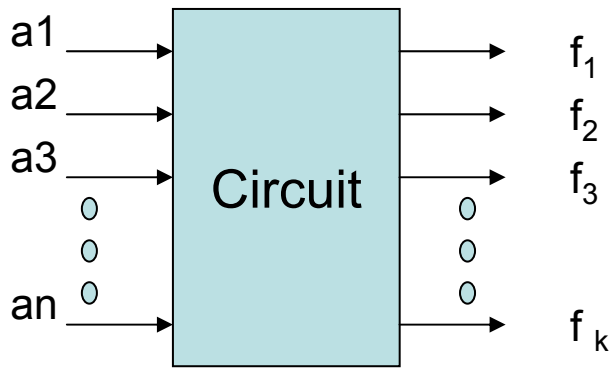
Simplifier une fonction booléennes consiste à mettre en œuvre des méthodes qui permettent d'écrire la fonction ou de réaliser le circuit correspondant sous sa forme la plus simple, tout en conservant les caractéristiques de la fonction.

Simplification par développement

Simplification par tables de Karnaugh

Exemples des circuits combinatoires

Fonction de transcodage



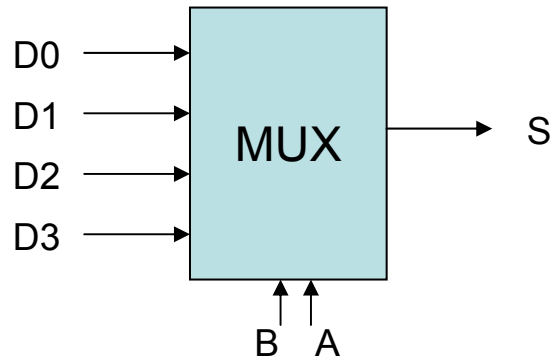
Le circuit est un circuit combinatoire si à chacun des combinaisons des variables a1, a2, …an correspond une combinaison et une seul des fonctions f1, f2, …fn.

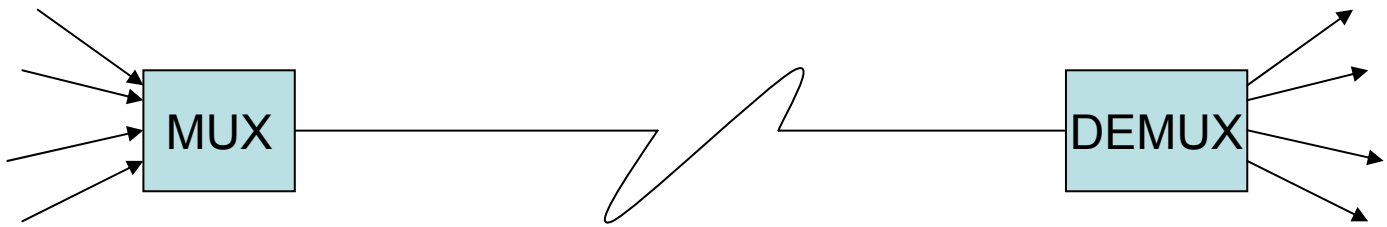
k=1, circuit logique

k>1, Additionneurs, multiplieurs, décodeurs, multiplexeurs …

Exemples des circuits combinatoires

Multiplexeurs et démultiplexeurs





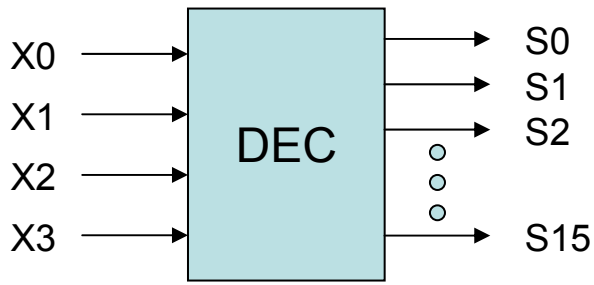
Conversion série parallèle

Conversion parallèle série

Exemples des circuits combinatoires

Décodeurs codeurs et transcodeurs

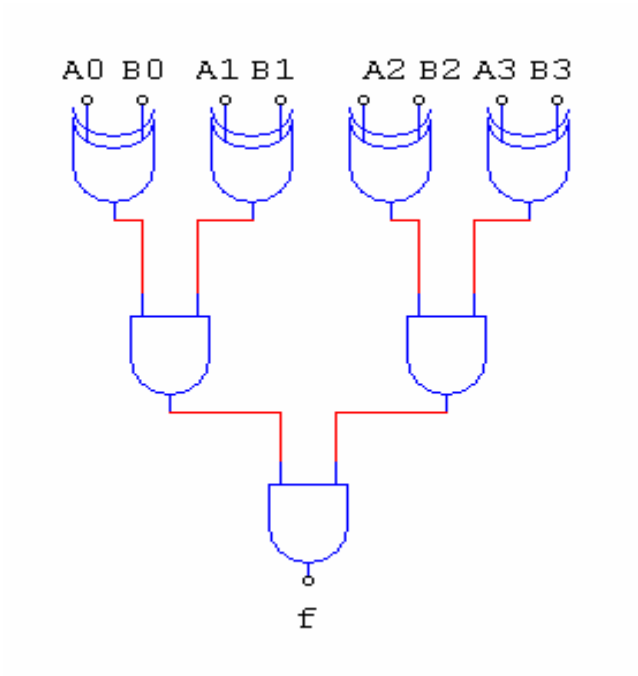
Le décodeurs fait correspondre à un code en entrée (sur n lignes) une seule Sortie active parmi les 2n sorties possibles



Exemples des circuits combinatoires

Les comparateurs

Détection de l'égalité de deux nombres



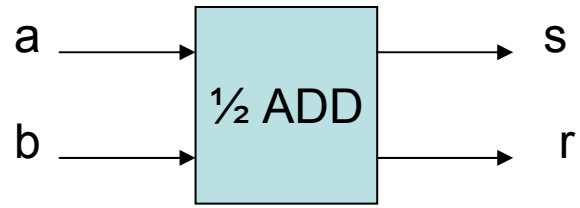
ALU

Décodeurs d'adresses

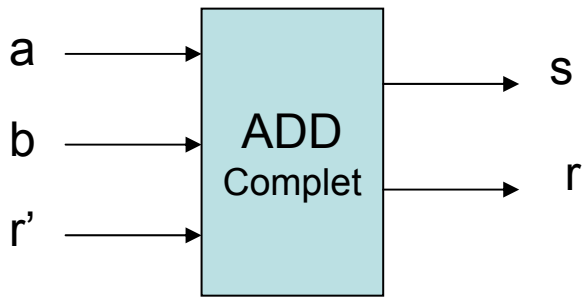
Exemples des circuits combinatoires

L'additionneur binaire

Le demi additionneur

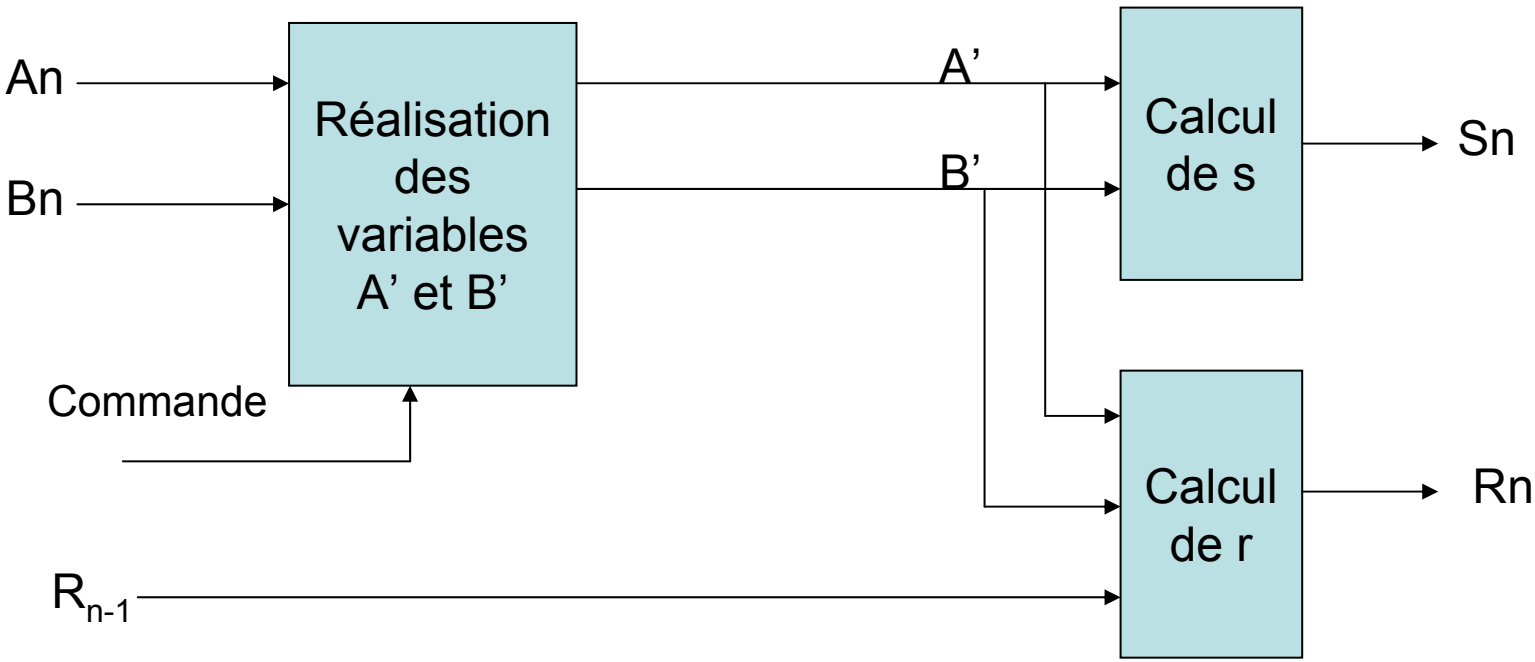


L'additionneur complet



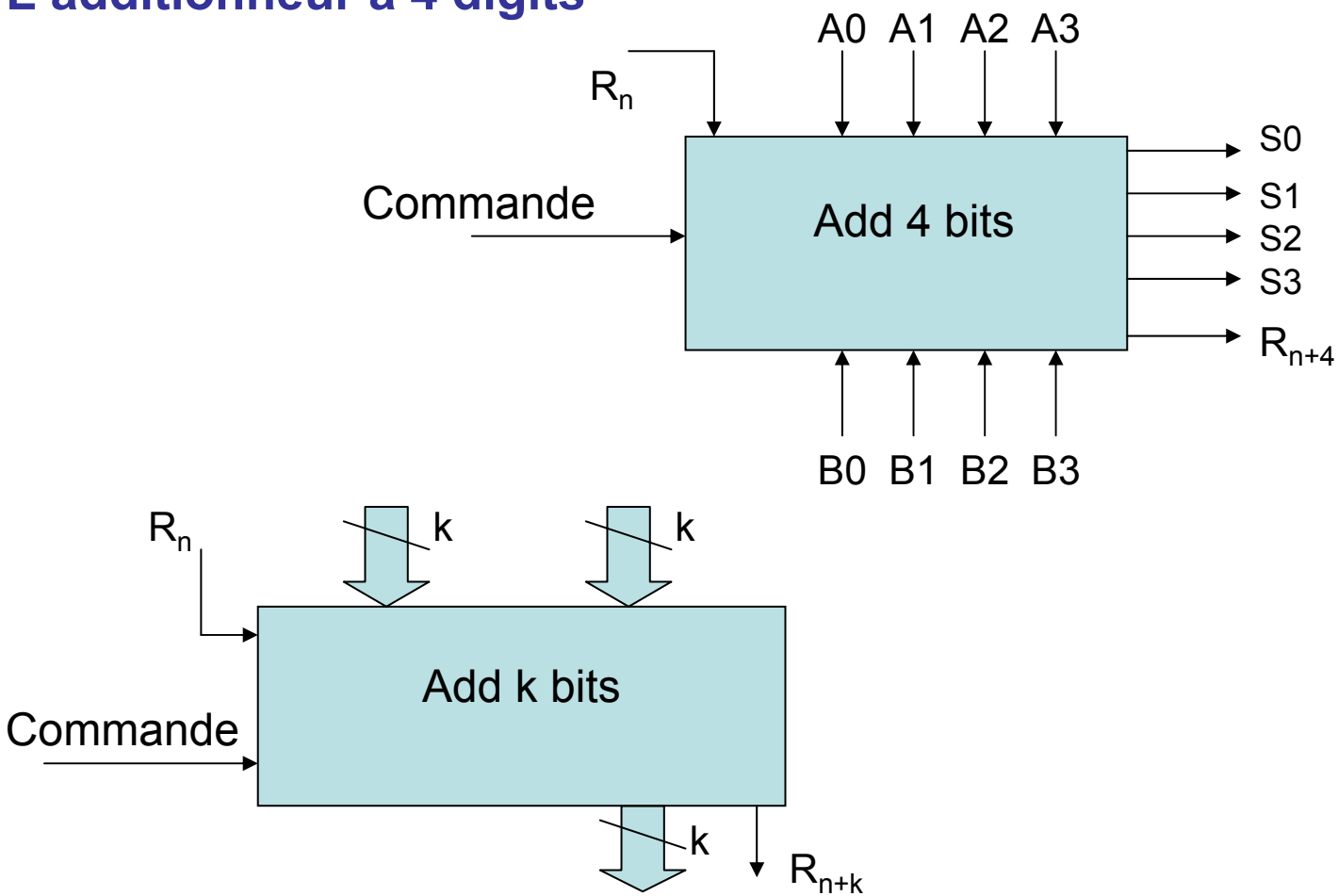
Exemples des circuits combinatoires

L'addition - soustraction



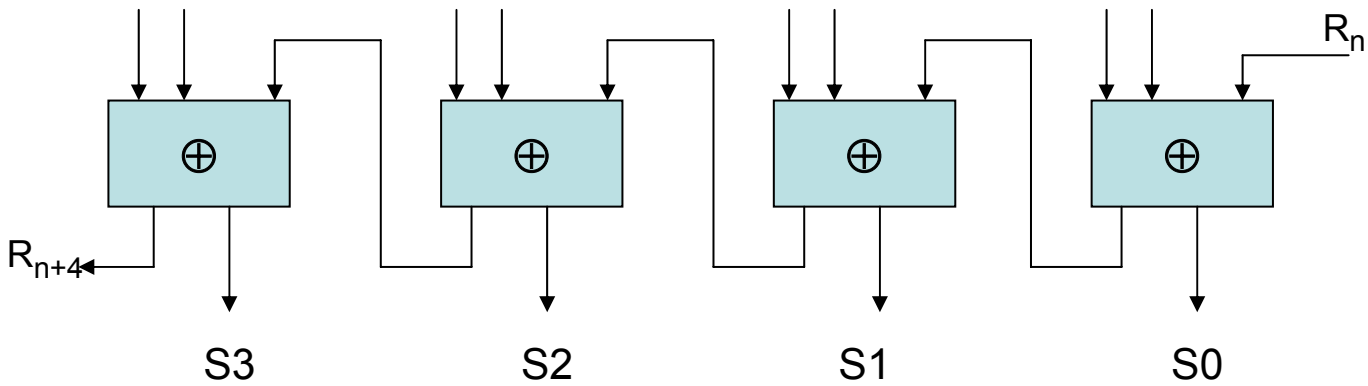
Exemples des circuits combinatoires

L'additionneur à 4 digits



L'additionneur

Propagation de la retenue



$t = 4 \times t_{add}$

SN74LS83

L'additionneur

La retenue anticipé

Les cas :

1 – pas de retenue Ri = 0

2 – La retenue propagée à travers l'étage

Ri = Ri-1

Termes Pi = ai + bi = 1

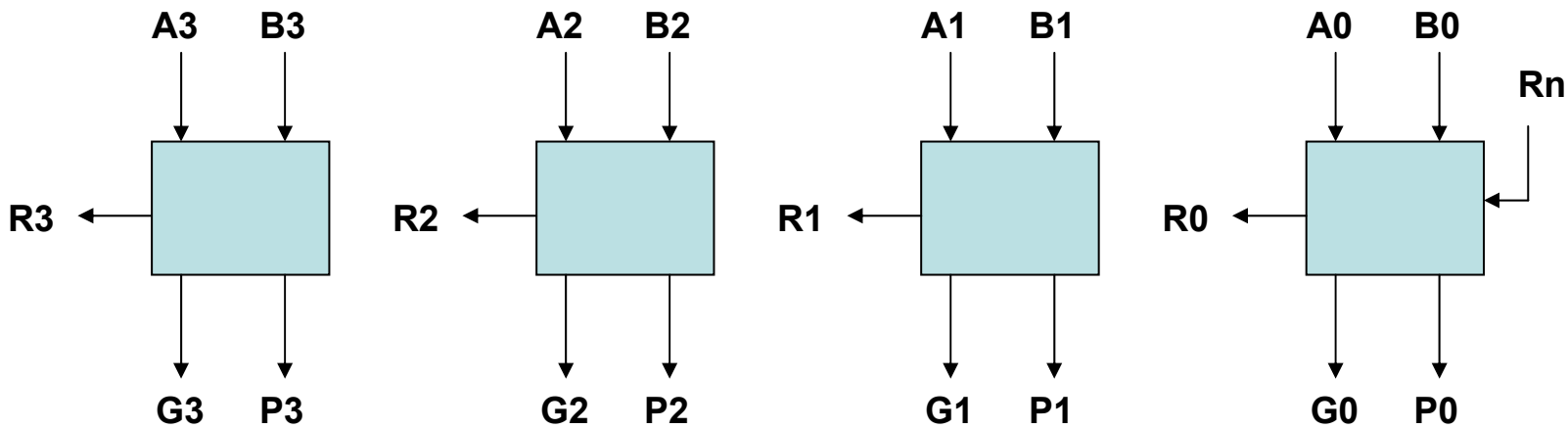
3 – La retenue y est indépendamment de Ri Ri = 1

Termes de génération Gi = ai bi = 1

A_i	B_i	R_{i-1}	Si	Ri	N° de cas
0	0	0	0	0	1
0	0	1	1	0	
0	1	0	1	0	2
0	1	1	0	1	
1	0	0	1	0	
1	0	1	0	1	
1	1	0	0	1	3
1	1	1	1	1	

L'additionneur

La retenue anticipé



Les expressions :

$$R_0=G_0 + R_nP_0 \qquad R_1=G_1 + R_0P_1 \qquad R_2= G_2 + R_1P_2 \qquad R_3= G_3 + R_2P_3 = R_{n+4}$$

$$Rn+4 = R3 = G3 + P3G2 + P3P2G1 + P3P2P1G0 + P3P2P1P0Rn$$

3 couches de portes

$t_{add} = 3 \times t_{porte}$

SN74LS83A
SN74AS181A
SN74AS182

Les circuits numériques

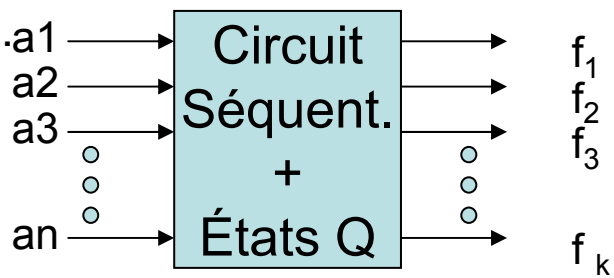
Les circuits séquentiels

Les circuits combinatoires n'ont pas de rétroactions et sont des circuits idéaux sans délai.

Les sorties ne dépendent que des entrées au même instant et l'étude repose sur **l'algèbre de Boole**.

Les circuits séquentiels possèdent des **rétroactions**.

Le circuit se rappelle des **Entrées**
et des **États** précédents



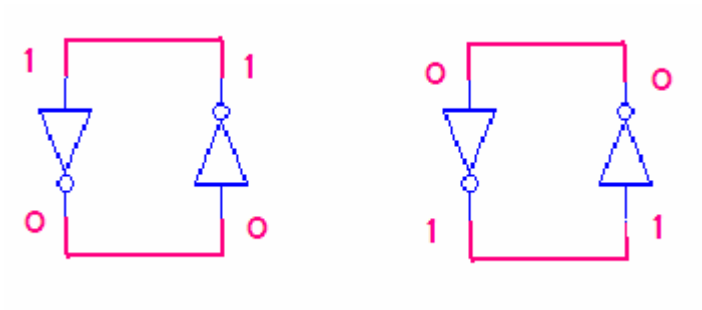
Il y a une mémoire du passé

L'étude des circuits séquentiels repose sur **la théorie des automates finis**.

Les bistables (les basculeurs ou flip-flops)

Deux inverseurs en opposition.

Deux états stables



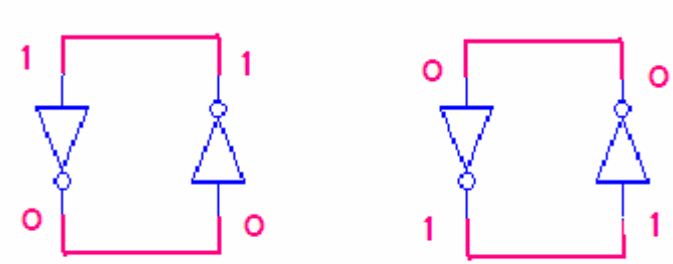
Plusieurs types :

R-S , D , T , J-K

Les bistables (les basculeurs ou flip-flops)

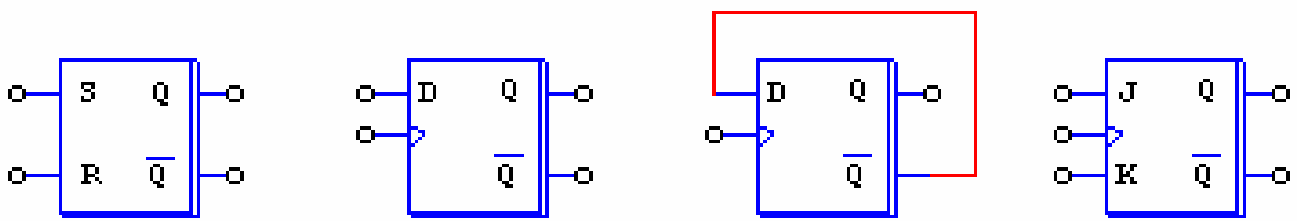
Deux inverseurs en opposition.

Deux états stables



Plusieurs types : sans et avec horloge de synchronisation

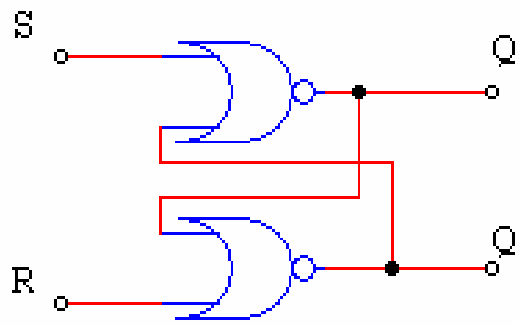
R-S , D , T , J-K

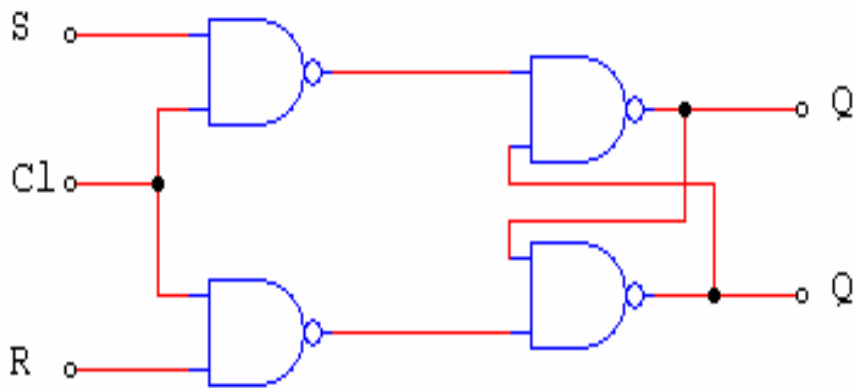


Les bistables :

Le bistable R-S.

R	S	Q	Q^+	Action
0	0	0	0	$Q^+=Q$
0	0	1	1	$Q^+=Q$
0	1	0	1	Mise a 1
0	1	1	1	Mise a 1
1	0	0	0	Effacement
1	0	1	0	Effacement
1	1	0	?	Indetermine
1	1	1	?	Indetermine



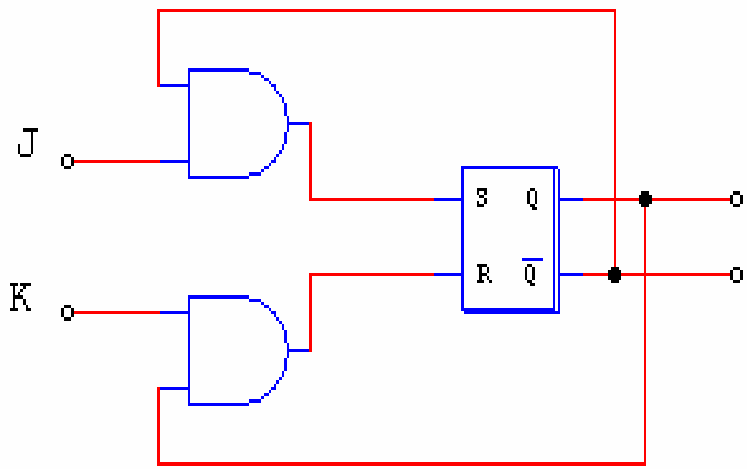


R-S synchonise

Les bistables :

Le bistable J-K.

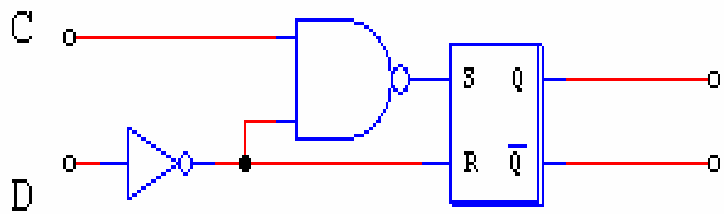
J	K	Q	Q^+	Action
0	0	0	0	$Q^+=Q$
0	0	1	1	$Q^+=Q$
0	1	0	1	Mise à 1
0	1	1	1	Mise à 1
1	0	0	0	Effacement
1	0	1	0	Effacement
1	1	0	1	Complément
1	1	1	0	Complément

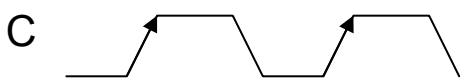


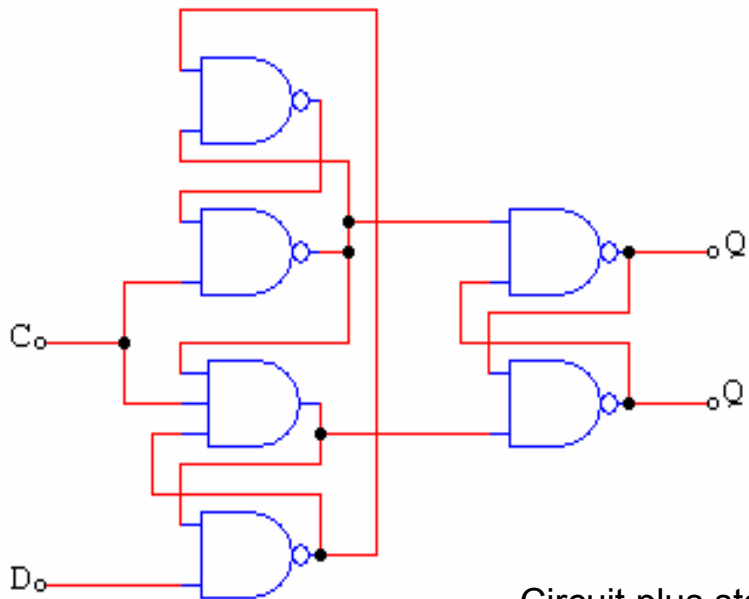
Les bistables :

Le bistable D.

D	C	Q	Q^+	Action
0	0	0	0	$Q^+=Q$
0	0	1	1	$Q^+=Q$
0	1	0	0	Mise à 0
0	1	1	0	Mise à 0
1	0	0	0	$Q^+=Q$
1	0	1	1	$Q^+=Q$
1	1	0	1	Mise à 1
1	1	1	1	Mise à 1





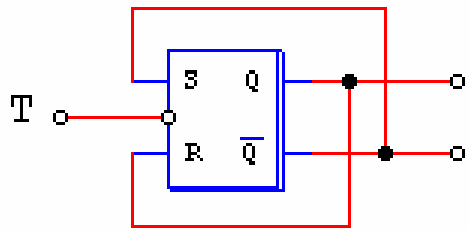


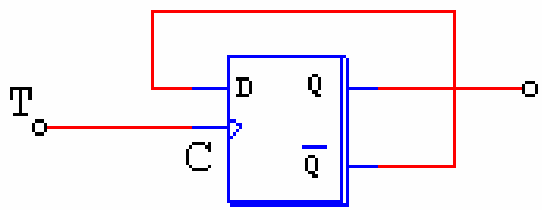
Circuit plus stable

Les bistables :

Le bistable T.

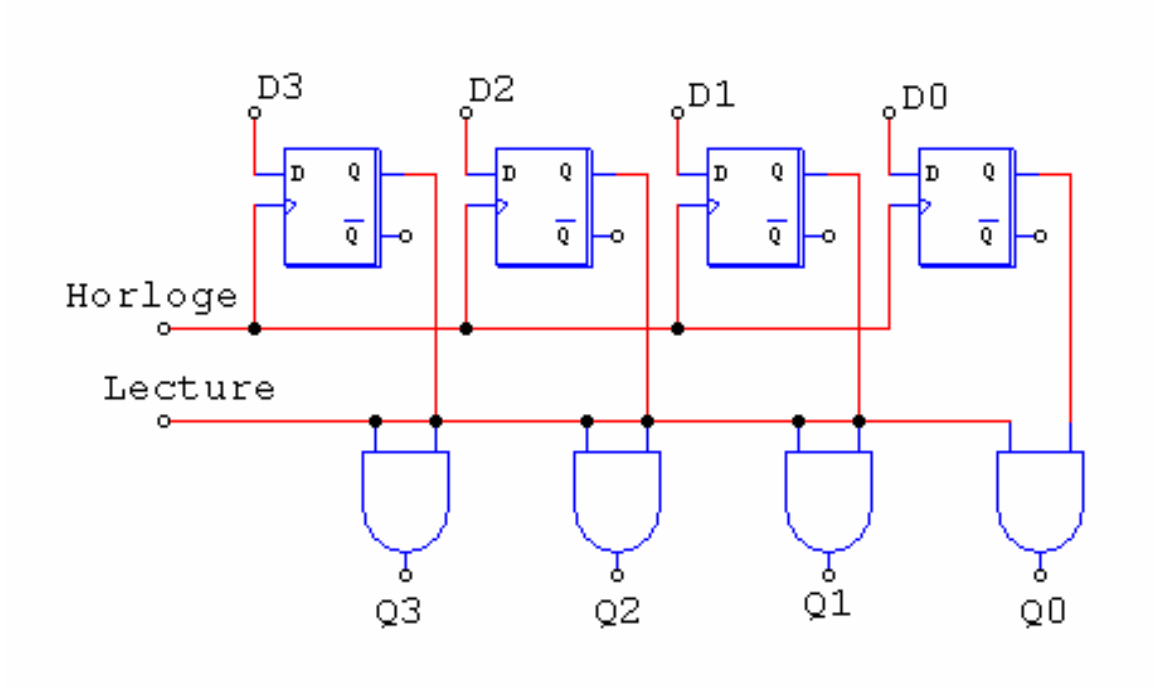
T	Q	Q^+	Action
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0





Les applications

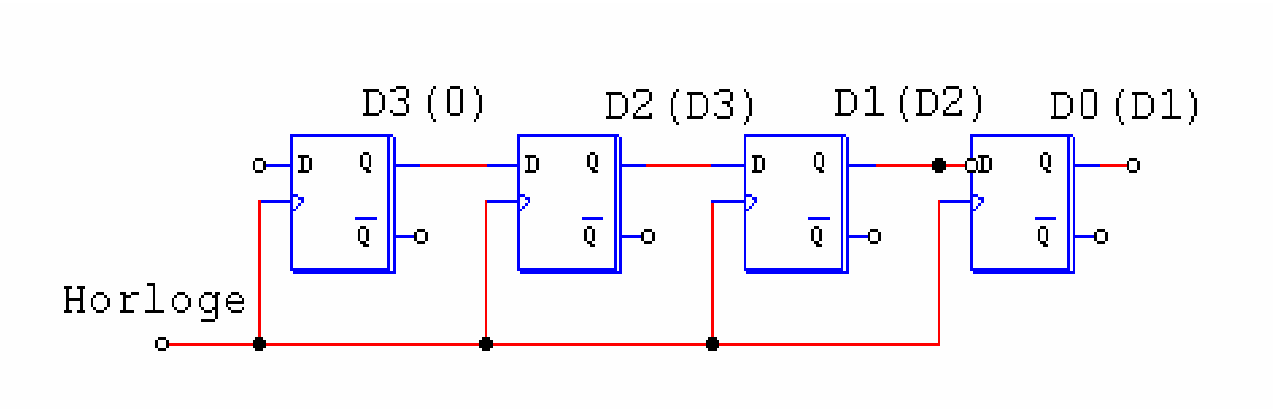
Le registre



Registre à 4 bits

Les applications

Le registre à décalage

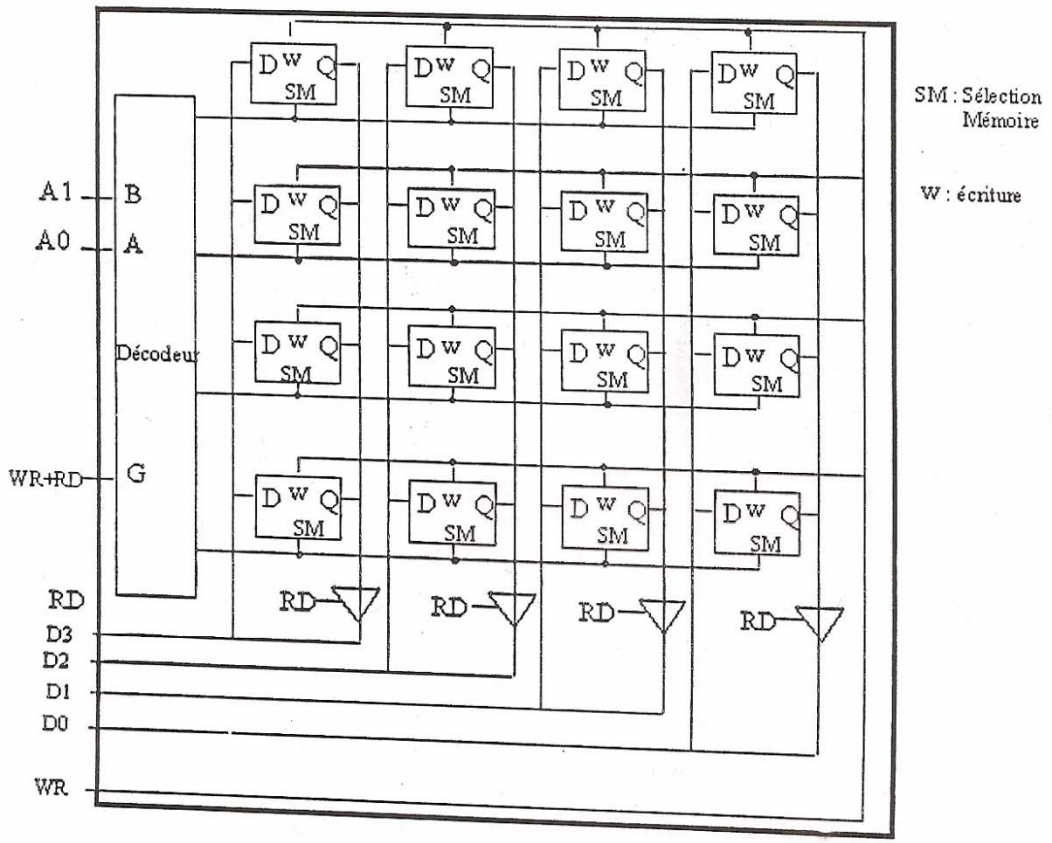


Diviseur sur 2

Convertisseur série/parallèle

Les applications

Mémoires rapides

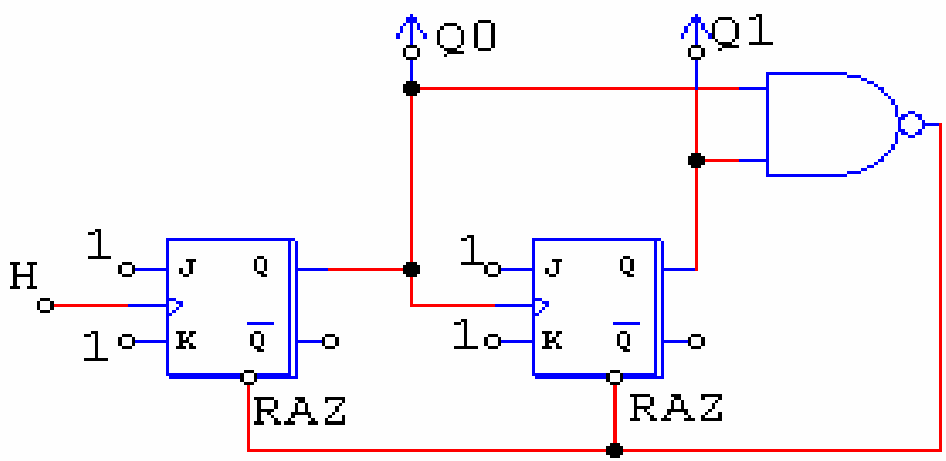


Les compteurs

Les compteurs réalisent la fonction de dénombrement des événements

Deux types : asynchrone et synchrone

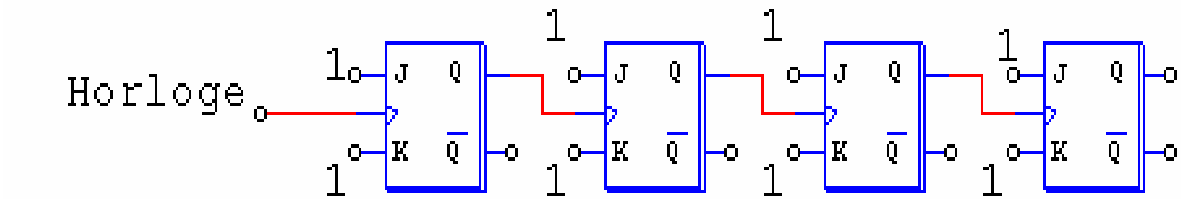
Asynchrone

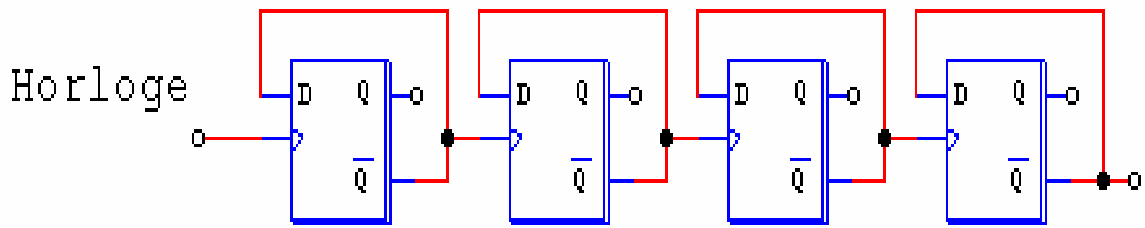


Modulo - n

Les compteurs

Asynchrone





+ circuit plus simple

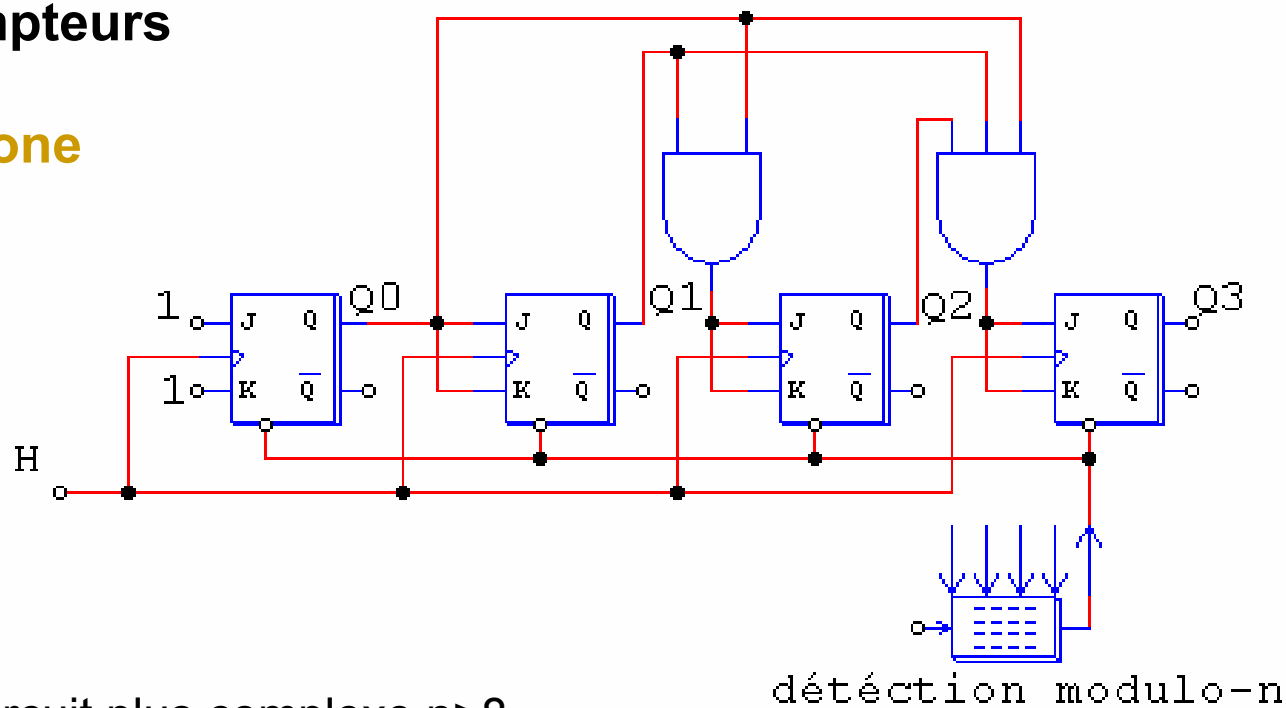
- Temps de comptage plus long

- Problème d'aléas

Les applications

Les compteurs

synchrone



- circuit plus complexe n>8

+ Rapidité de comptage

+ pas de problème d'aléas

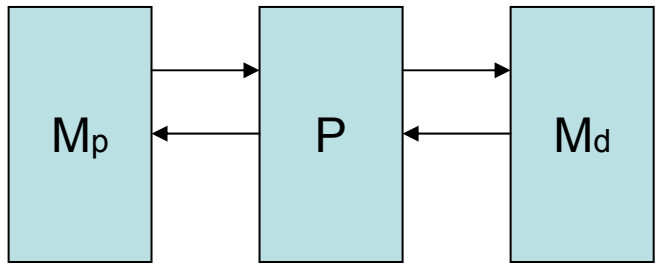
SN74LS190/191

Les architecture des processeurs

Historique :

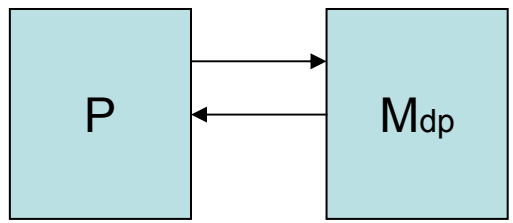
Hardward

1944 Mark
1946 ENIAC



- Organisation complexe

Von Neuman 1945



Devenu un standard pour les architectures a utilisations génerales

ALU + shifter (add, sub, shift) opérations plus faciles à réaliser

Instr. Compl. (x , /) sont réalisées par une série de shift/ADD ou SUB

dans ROM

CISC

Les architectures des processeurs

Principes de fonctionnement

L'architecture de type Von Neuman (1945)

Caractéristiques :

- L'information est codée en binaire et représentée par un ensemble de mots.
- Les instructions et les données sont codées de la même façon et sont dans la même mémoire.
- Les mots dans la mémoire se distinguent par leurs adresses.
- L'utilisation d'un seul bus pour le transfert des données.

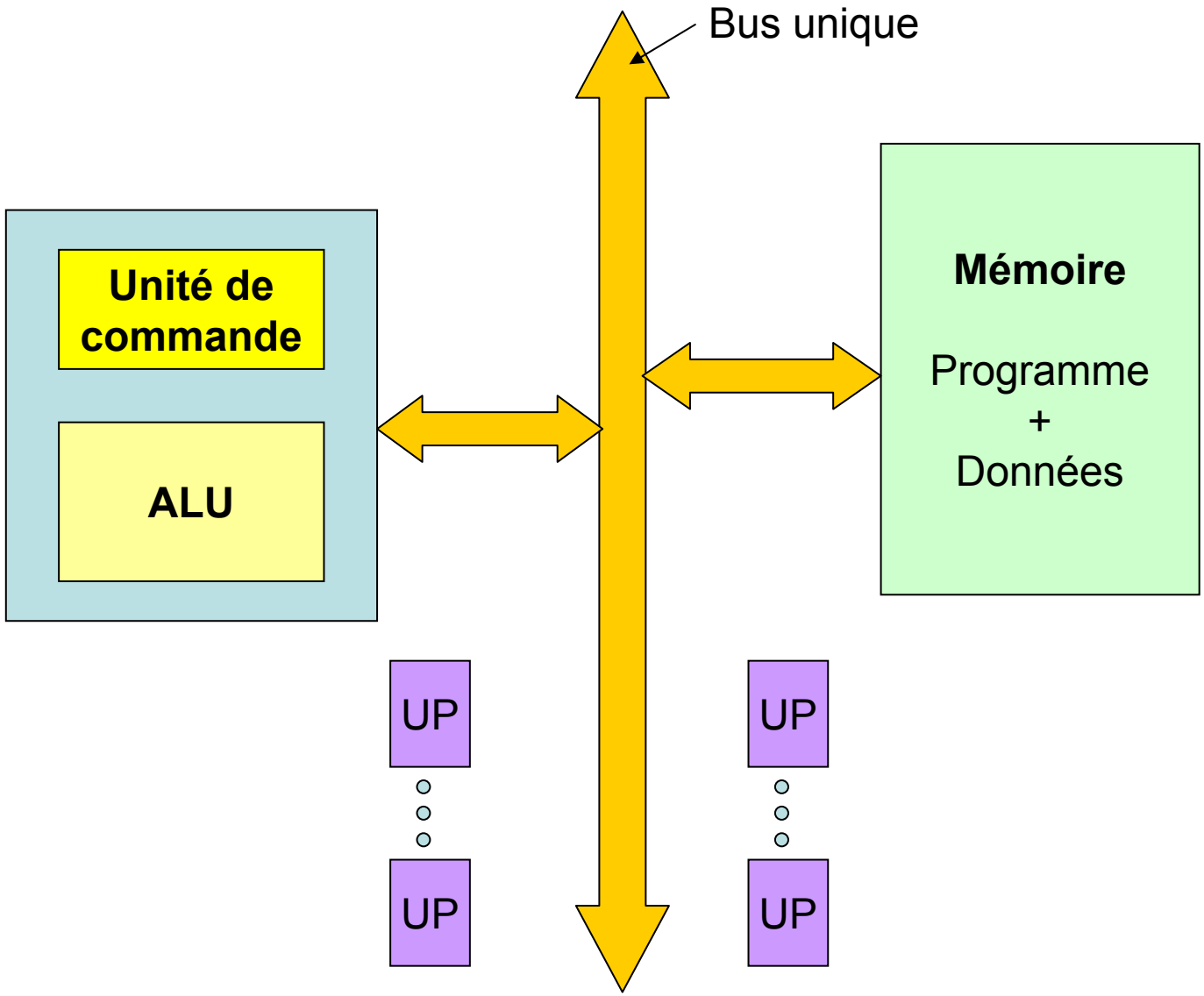
Les architectures des processeurs
Principes de fonctionnement

L'architecture de type Von Neumann (1945)

Caractéristiques :

- L'algorithme pour résoudre le problème est réalisé sous forme de programme.
- Le programme est une suite des instructions.
- L'exécution séquentielle des instructions.
    * Extraction de l'instruction;
    * Décodage de l'instruction;
    * extraction des opérandes;
    * Exécution de l'opération;
    * Stockage du résultat.

L'architecture Von Neumann



Les architectures des processeurs
Principes de fonctionnement

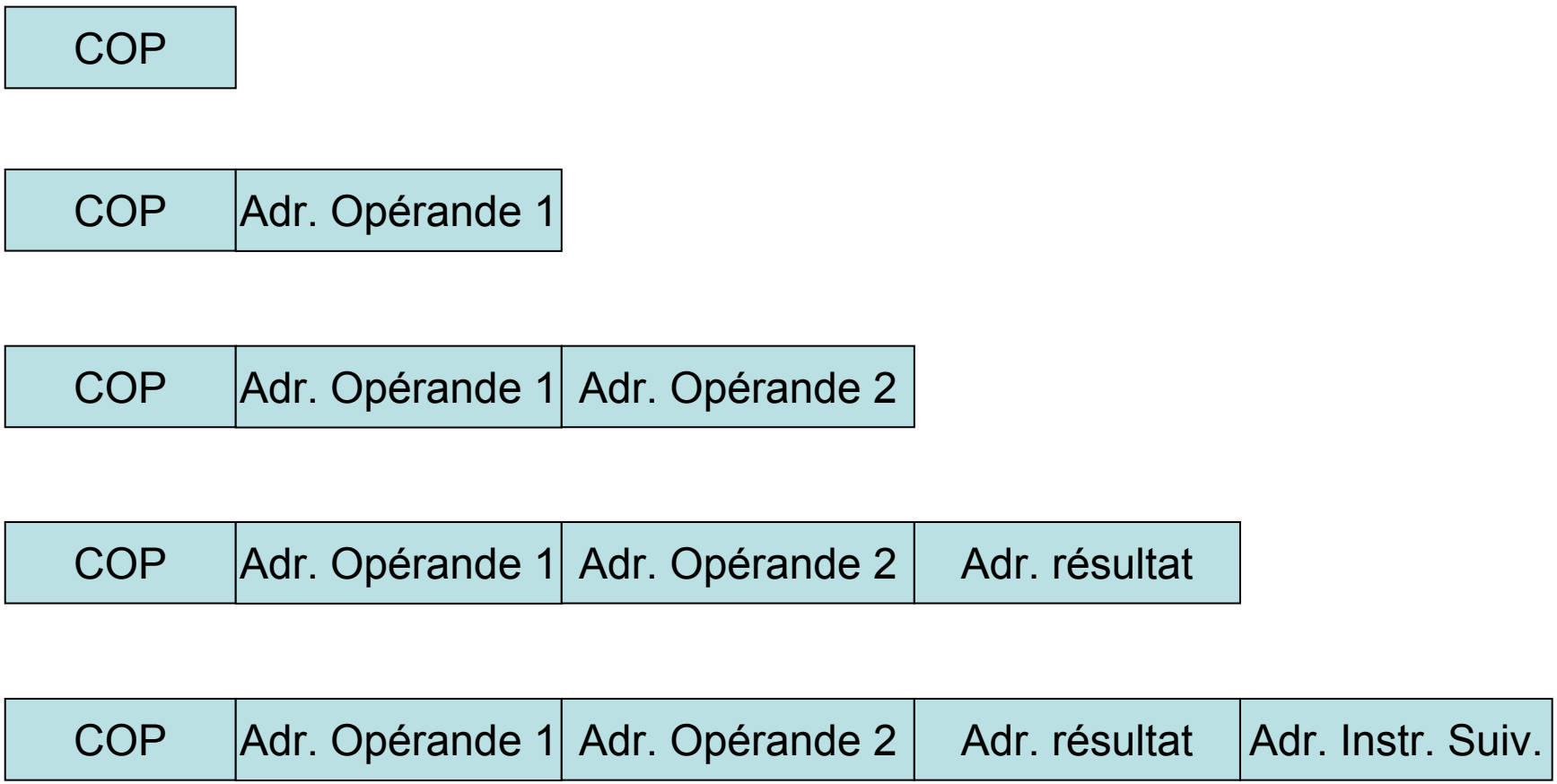
Définitions :

Instruction: ensemble de micro- opérations réalisées à un instant donné effectuant une opération spécifique.

Micro- opération: une des actions réalisée au niveau du processeur lors de l'exécution d'une instruction.

Les architectures des processeurs
Principes de fonctionnement

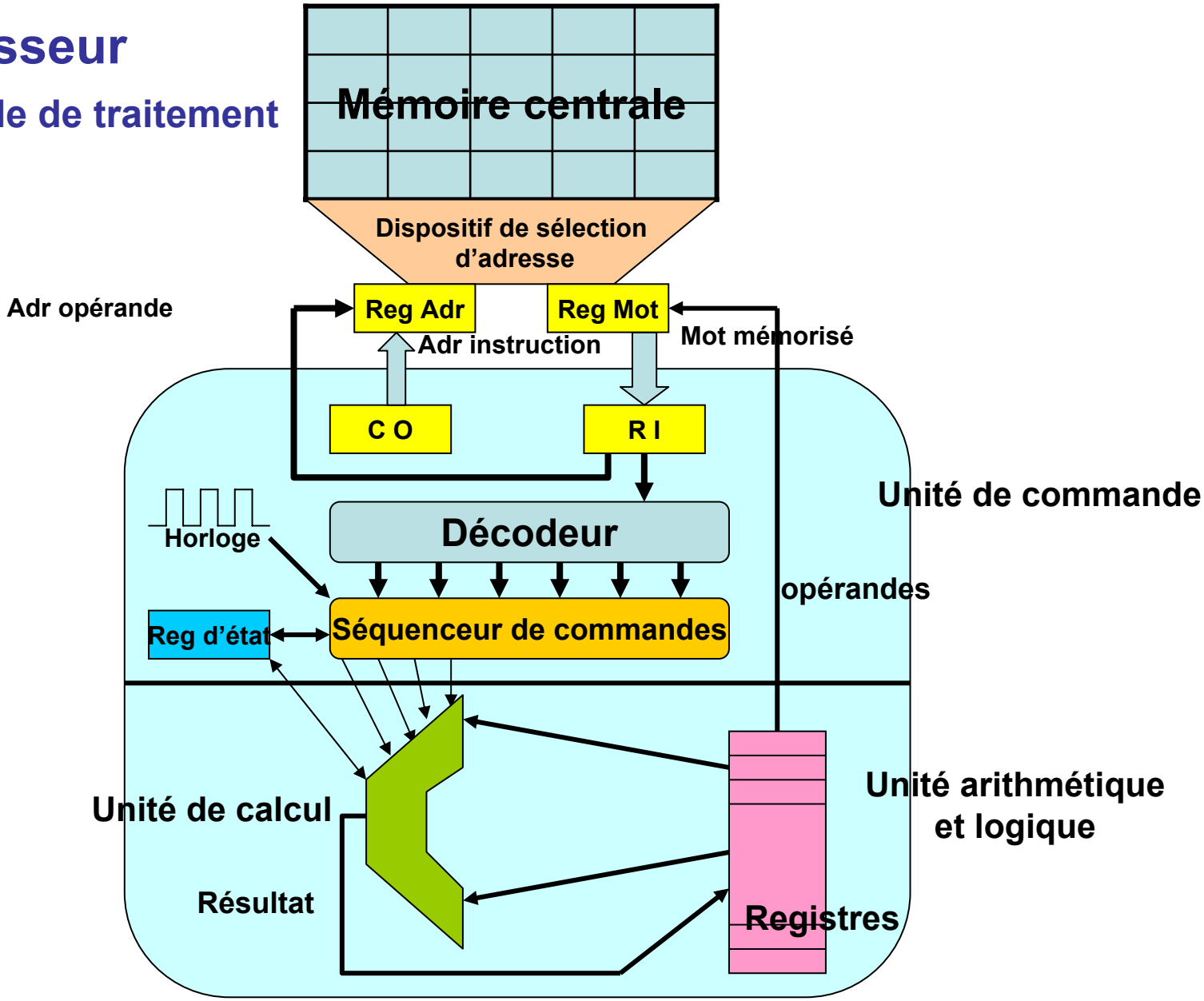
Plusieurs catégories d'architectures Von Neumann selon la manière d'adressage spécifiée dans l'instruction.



Le processeur

Unité centrale de traitement

CPU



Composition d'un processeur:

Mémoire centrale (programmes et données)

Unité centrale de traitement (exécution des programmes)

Unités d'E/S (échange d'informations avec les unités périphériques)

L'unité centrale de traitement

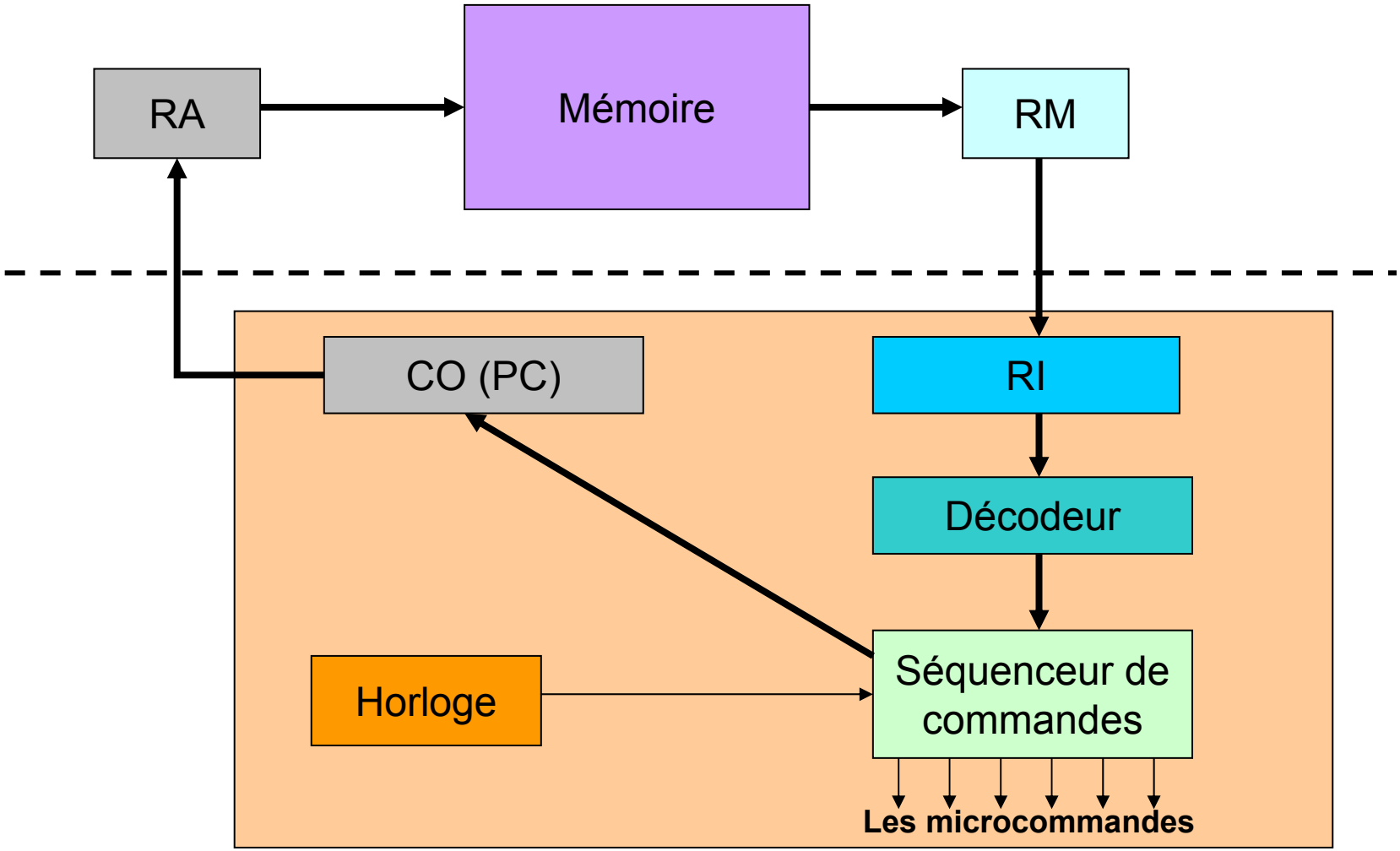
- L'unité de commande s'occupe de gérer l'exécution d'un programme.

A – Deux registre importants :

1 – Le registre d'instruction (RI) : contient l'instruction en cours d'exécution.

2 – Le compteur ordinal (PC) : contient toujours l'adresse de la prochaine instruction à exécuter. Il est automatiquement incrémenté (taille).

Le cycle de l'exécution des instructions

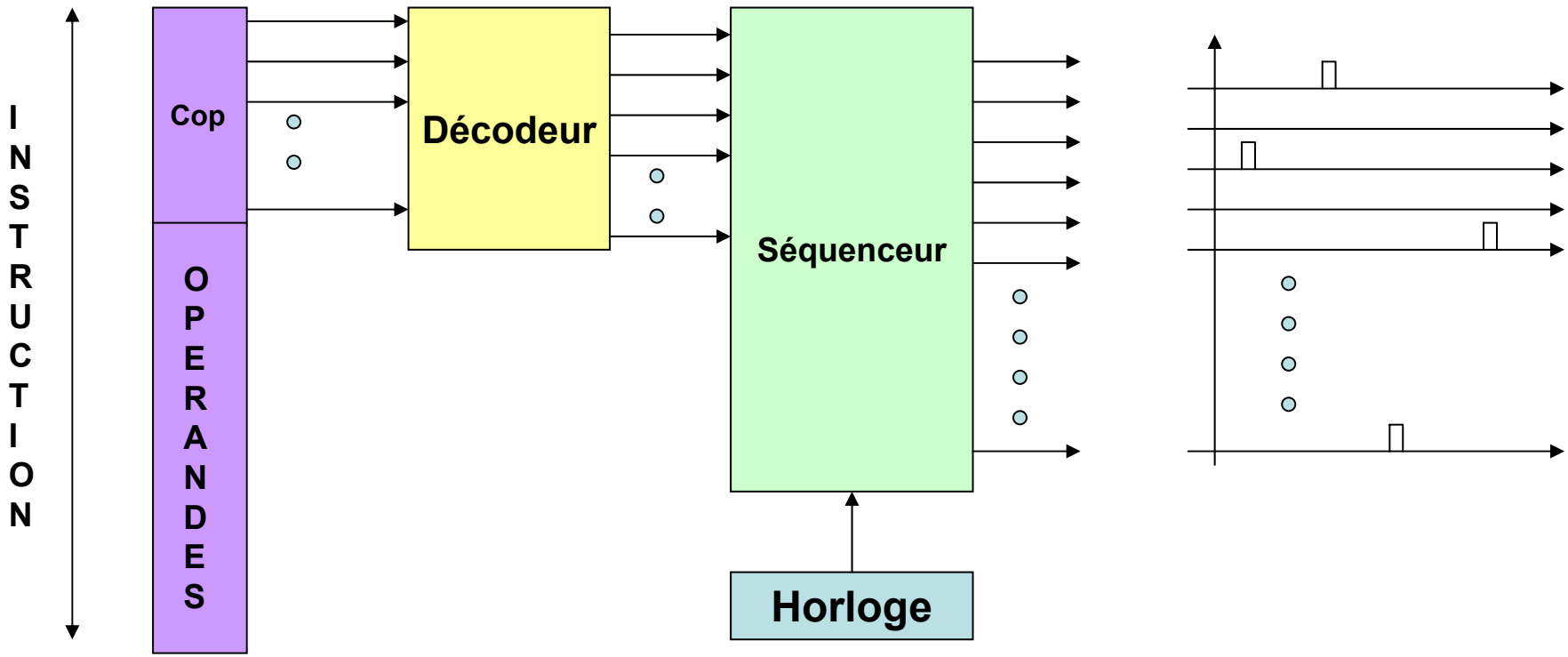


B – Décodeur et séquenceur de commandes :

1 – Le décodeur est un dispositif de décodage des instructions. C'est un circuit combinatoire qui permet à partir du champ du code opération de l'instruction de générer les différents signaux nécessaires à l'entrée du séquenceur.

2 – Le séquenceur de commande est un circuit séquentiel qui active les Circuits nécessaires à l'exécution de l'instruction en cours. Cette unité a besoin des signaux d'une horloge pour enchaîner les commandes

- ***Séquenceur câblé*** : réalisé entièrement à base de logique combinatoire et de la logique séquentielle



+ Plus rapide **- Complexité de réalisation**

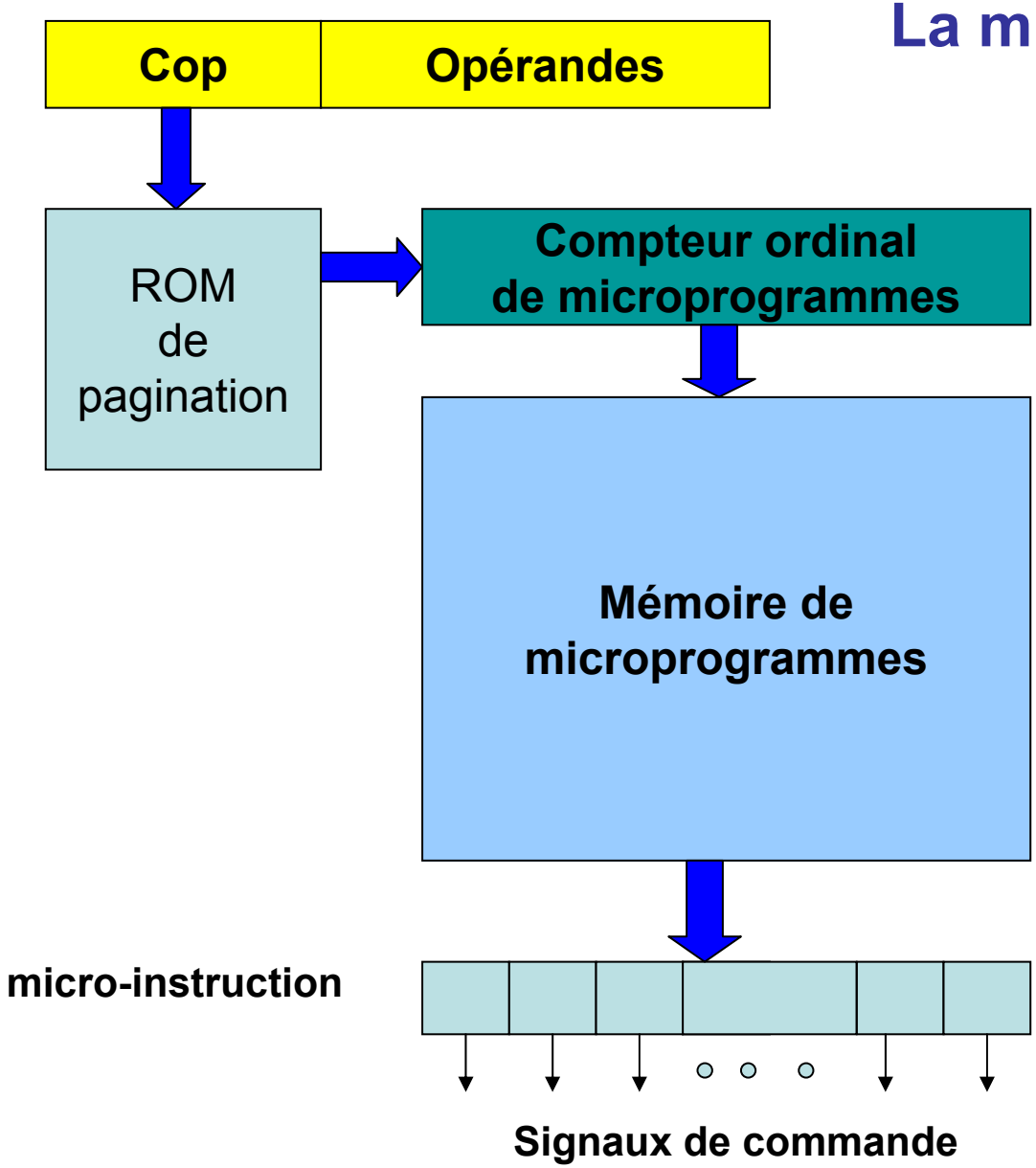
- ***Séquenceur micro programmé*** : réalisé entièrement à base d'une mémoire ROM de pagination et de microprogrammation.

Nécessite un compteur ordinal pour la lecture séquentielle des micro commandes à partir de la mémoire de microprogrammes.

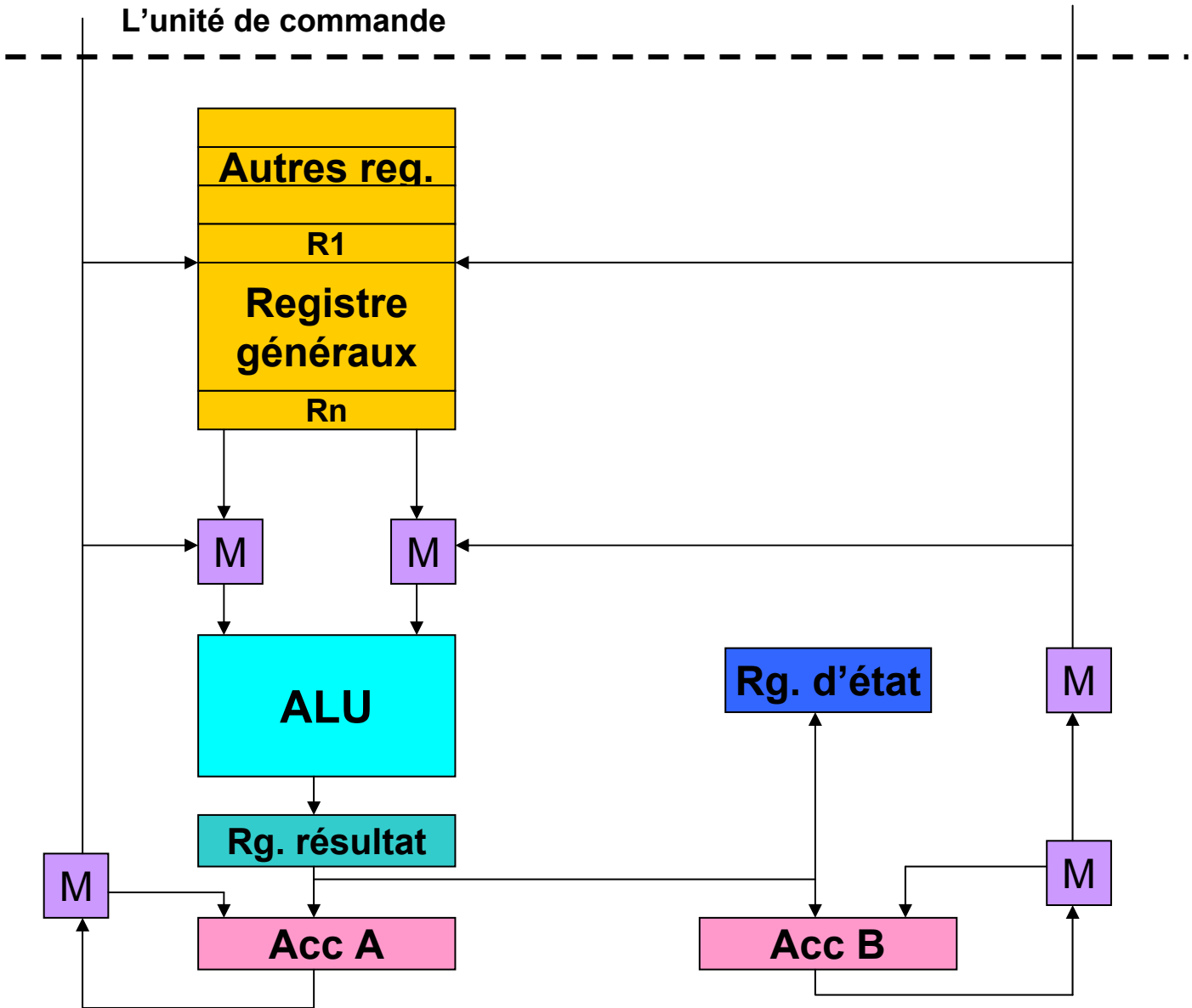
+ simple à réaliser

- Moins rapide

La micro programmation



L'unité arithmétique et logique



Les registres

- Les registres arithmétiques (Acc)

- Les registres de base et d'index

- Les registres banalisés

- Le registre d'état (PSW)

- Autres registres comme : Rgs à décalage (shift reg)
  Rgs pour op. à VF

Le registre d'état (PSW)

Indique l'état du système après opération arithmétique ou logique

Les bits (drapeaux, flags) : indiquent l'état d'une condition particulière dans le CPU

C : Carry flag , Retenue

V : Overflow flag , Dépassement de capacité

Z : Zero

N : Negative

I : Interrupt mask

Les modes d'adressage

Pour faciliter la programmation, les fabricants offrent toute une gamme de méthodes pour adresser les opérandes. Le format des instructions prévoit un champ dans le Cop dont les bits indiquent le mode choisi.

• Adressage direct	**Ad. effective**
• Adressage indirect	**Ad. De l'Ad** (plusieurs niveaux)
• Adressage immédiat	**l'opérande lui même**
• Adressage implicite	**Indiqué dans le Cop**
• Adressage indexé	**Ad=Champ Ad + Rgx**
• Adressage basé	**Ad=Champ Ad + Rgb**
• Adressage relatif	**Ad=Champ Ad + CO**
• Une combinaison des modes	

Le registre pointeur de pile
SP – Stack pointer

La pile est une zone de stockage organisé en LIFO.

Le SP est un registre contenant le niveaux de remplissage de la pile.

2 opérations fondamentales :

PUSH **PULL**

Le registre pointeur de pile SP – Stack pointer

* La pile est une structure dynamique.

* La pile conserve l'ordre de l'exécution des événements.

* En arithmétique la pile garde les op. et les rés. Int.

* En appel à des sous-routines la pile garde l'adresse de retour.

* En traitement des interruptions la pile garde l'état du Processeur.

* En appel aux procédures la pile est utilisé pour la Passation des paramètres.

Les mémoires

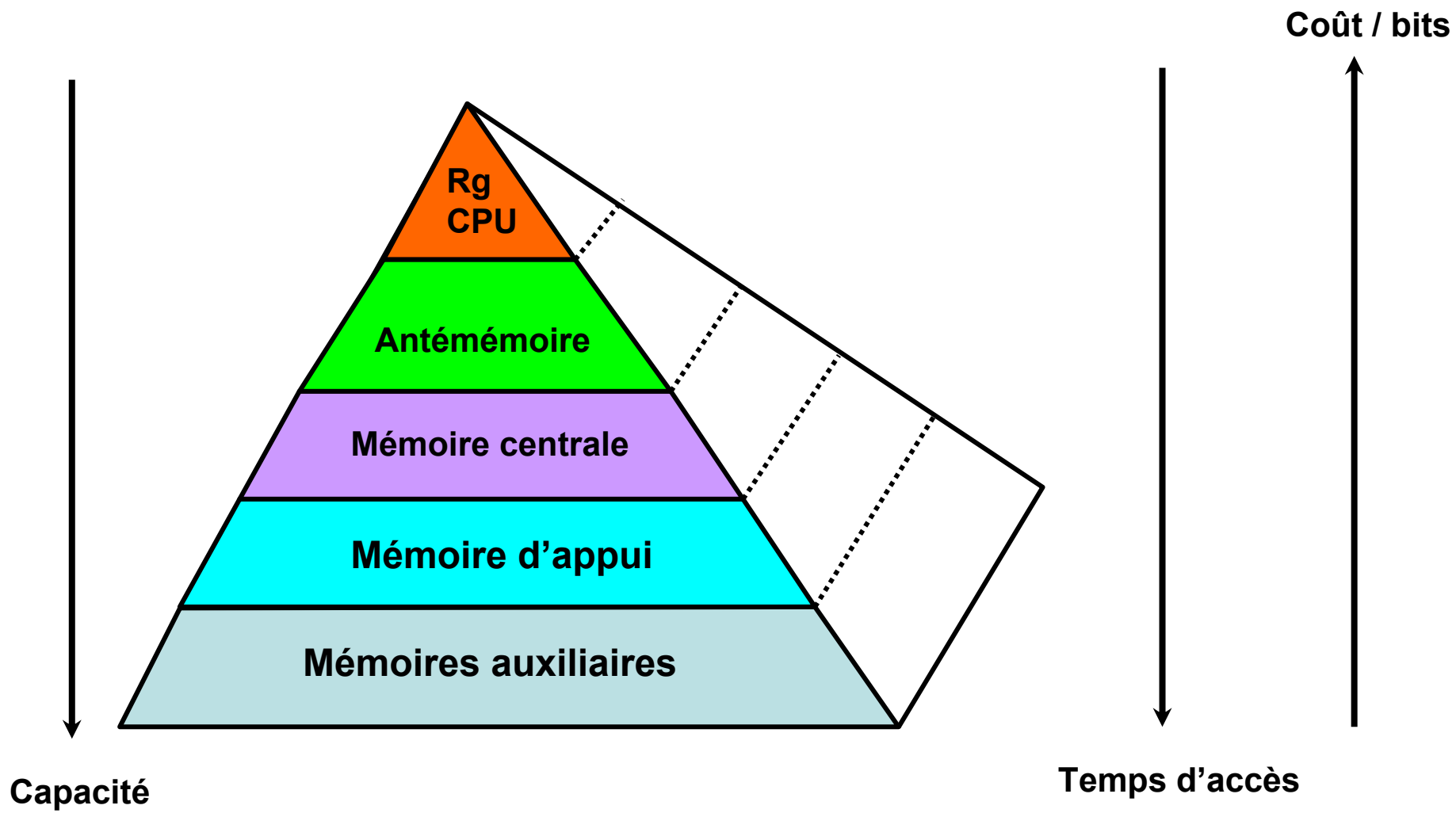
Dans un ordinateur deux caractéristiques essentielles :

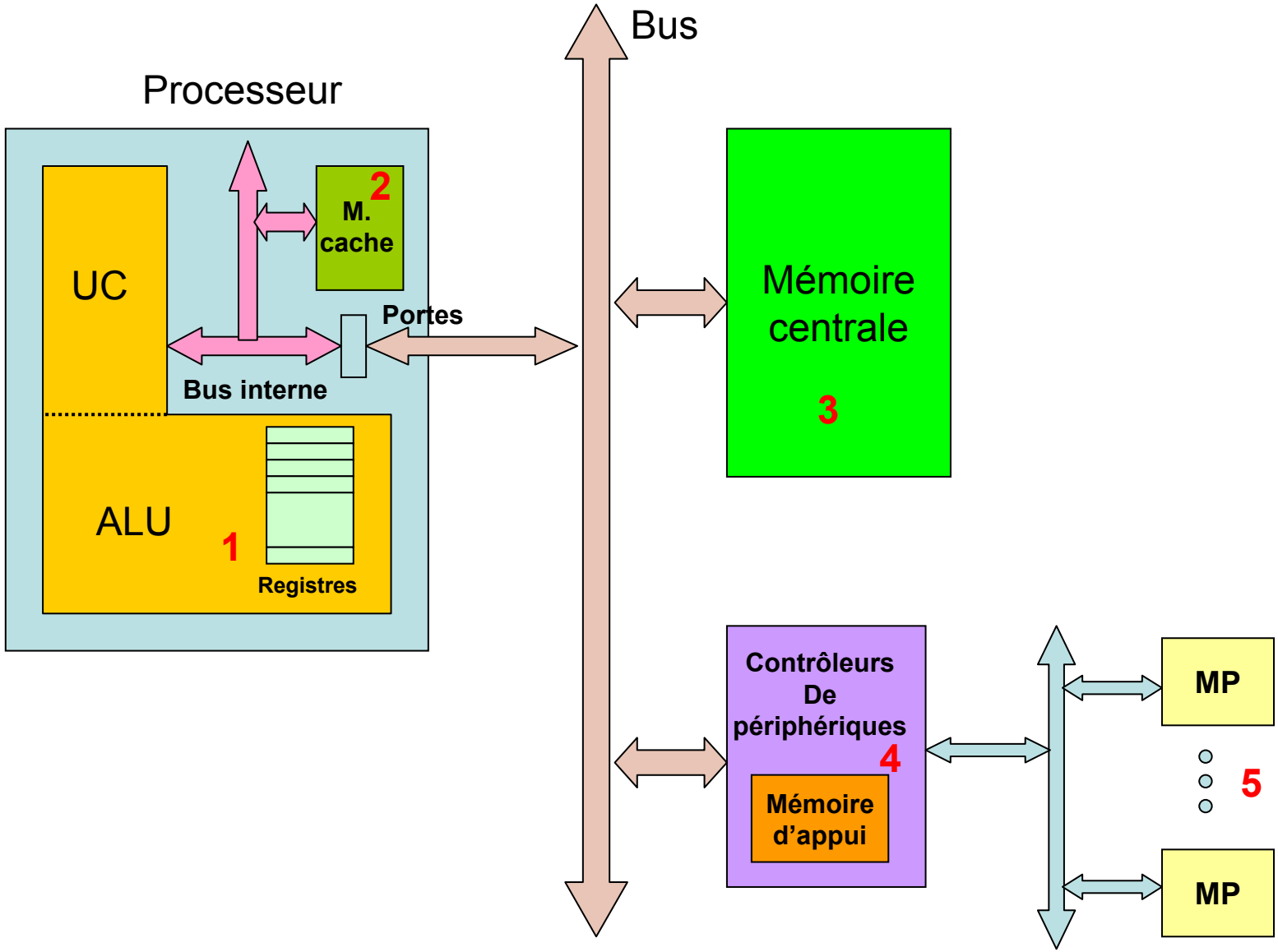
La vitesse de traitement **Capacité de mémorisation**

Une mémoire **: dispositif capable d'enregistrer, de concerver et de restituer des informations codées en binaire.**

La mémoire se caractérise par sa capacité, son le temps d'accès et son coût par bit

Hiérarchie des niveaux de mémoires




Bus
Processeur
UC
2
M.
cache
Portes
Bus interne
ALU
1
Registres
Mémoire
centrale
3
Contrôleurs
De
périphériques
4
Mémoire
d'appui
MP
5
MP

Classification par mode d'accès

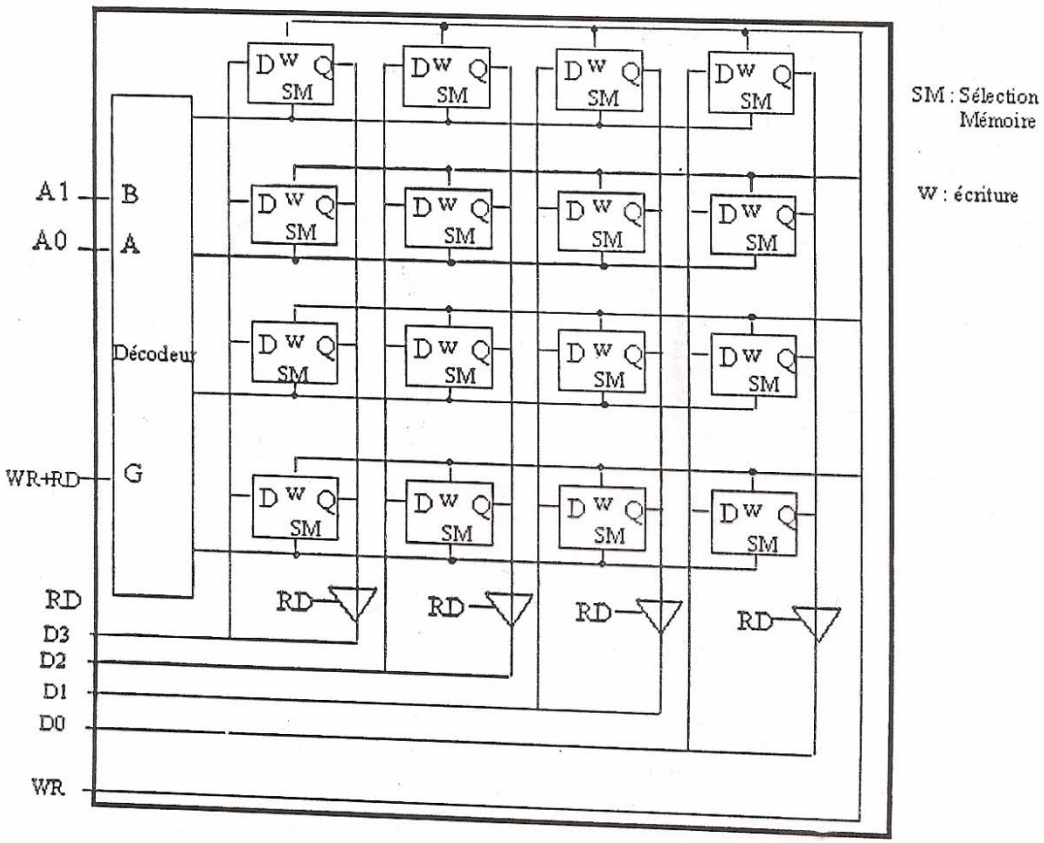
Mémoires à accès séquentiel

Mémoires à accès semi séquentiel

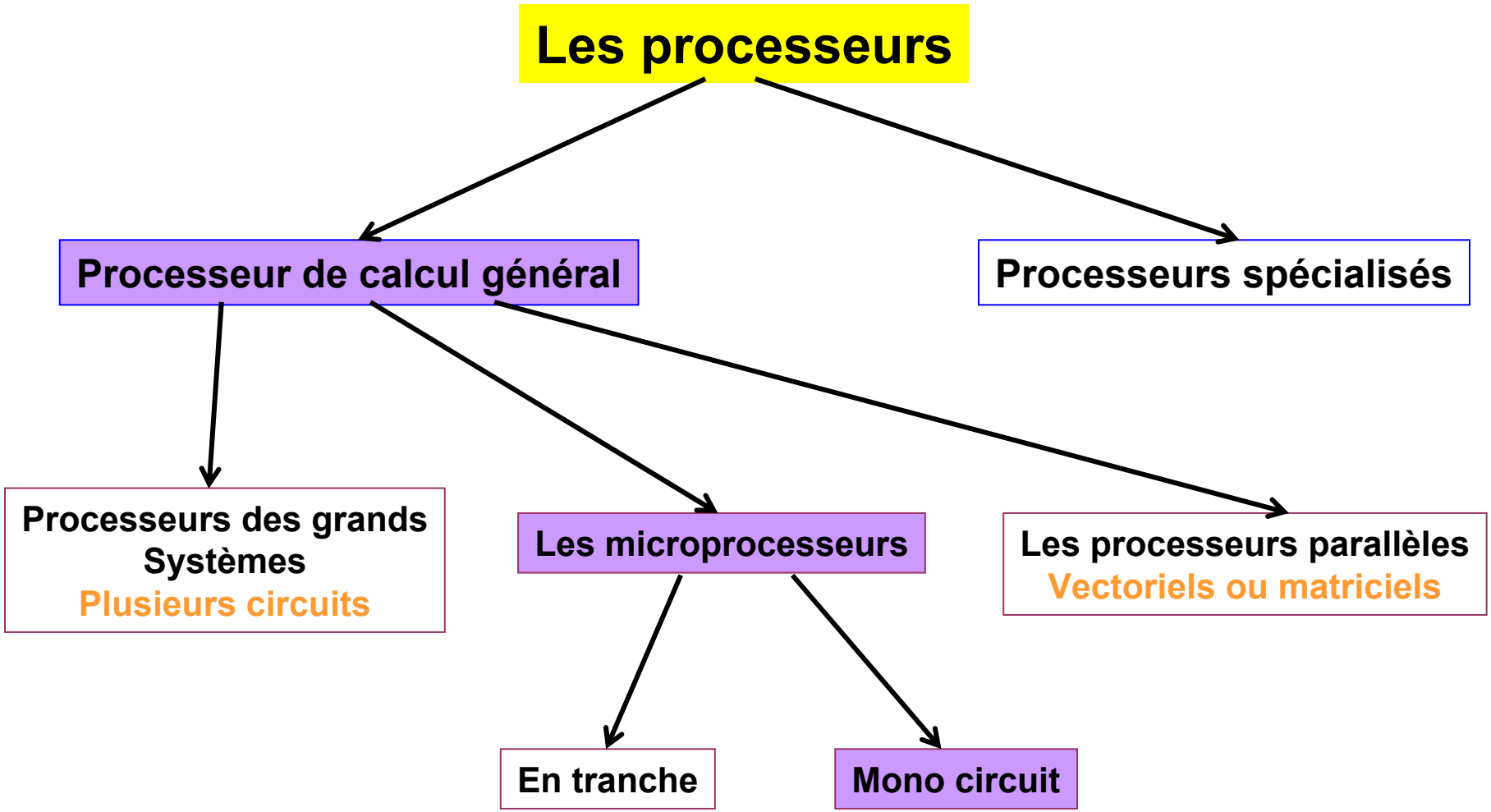
Mémoires à accès aléatoire

Mémoires à accès par le contenu : Mémoire associative

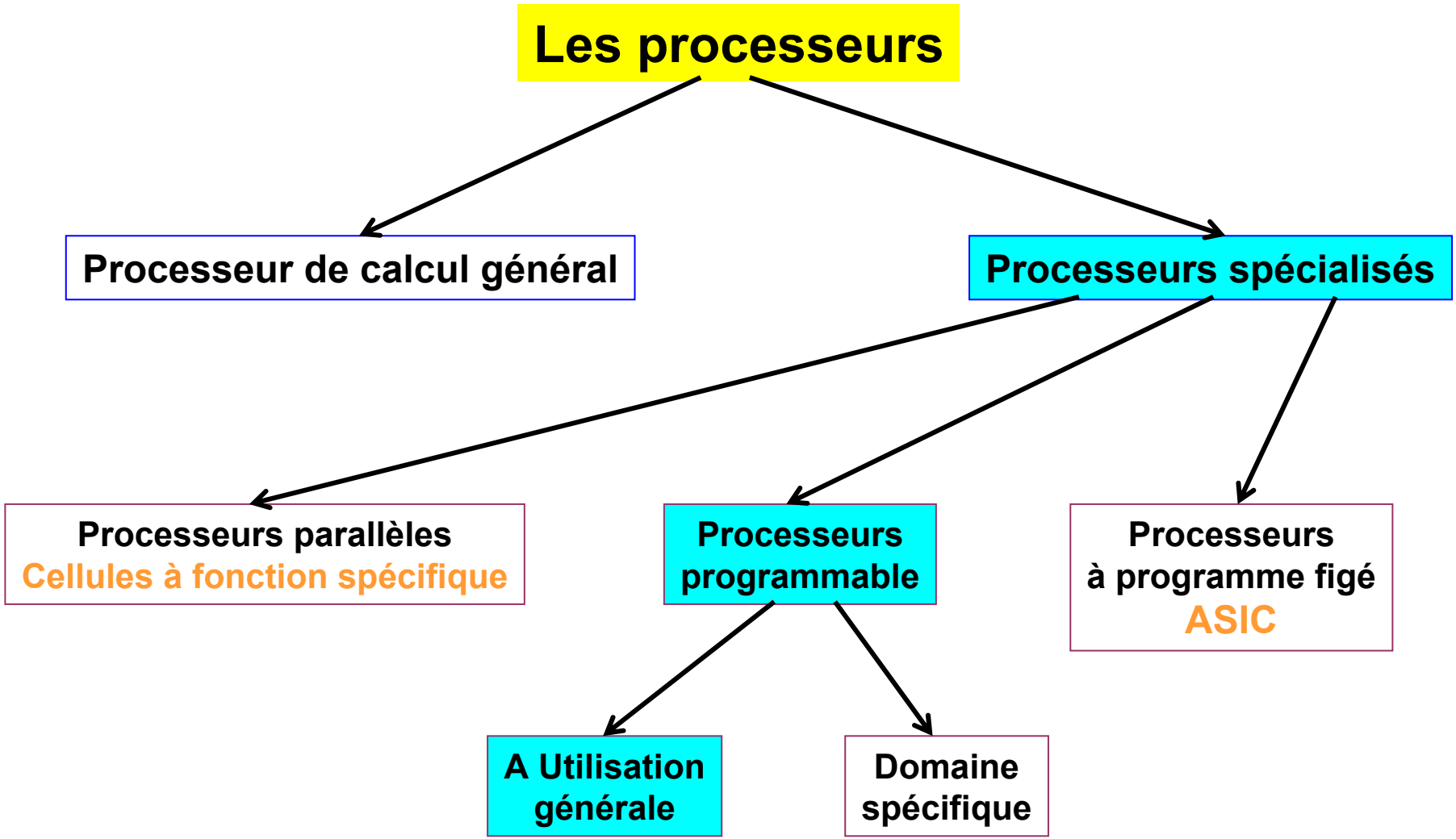
Fonction de mémorisation : bascule et registre



Classification des processeurs



Classification des processeurs



Définitions

Microprocesseur

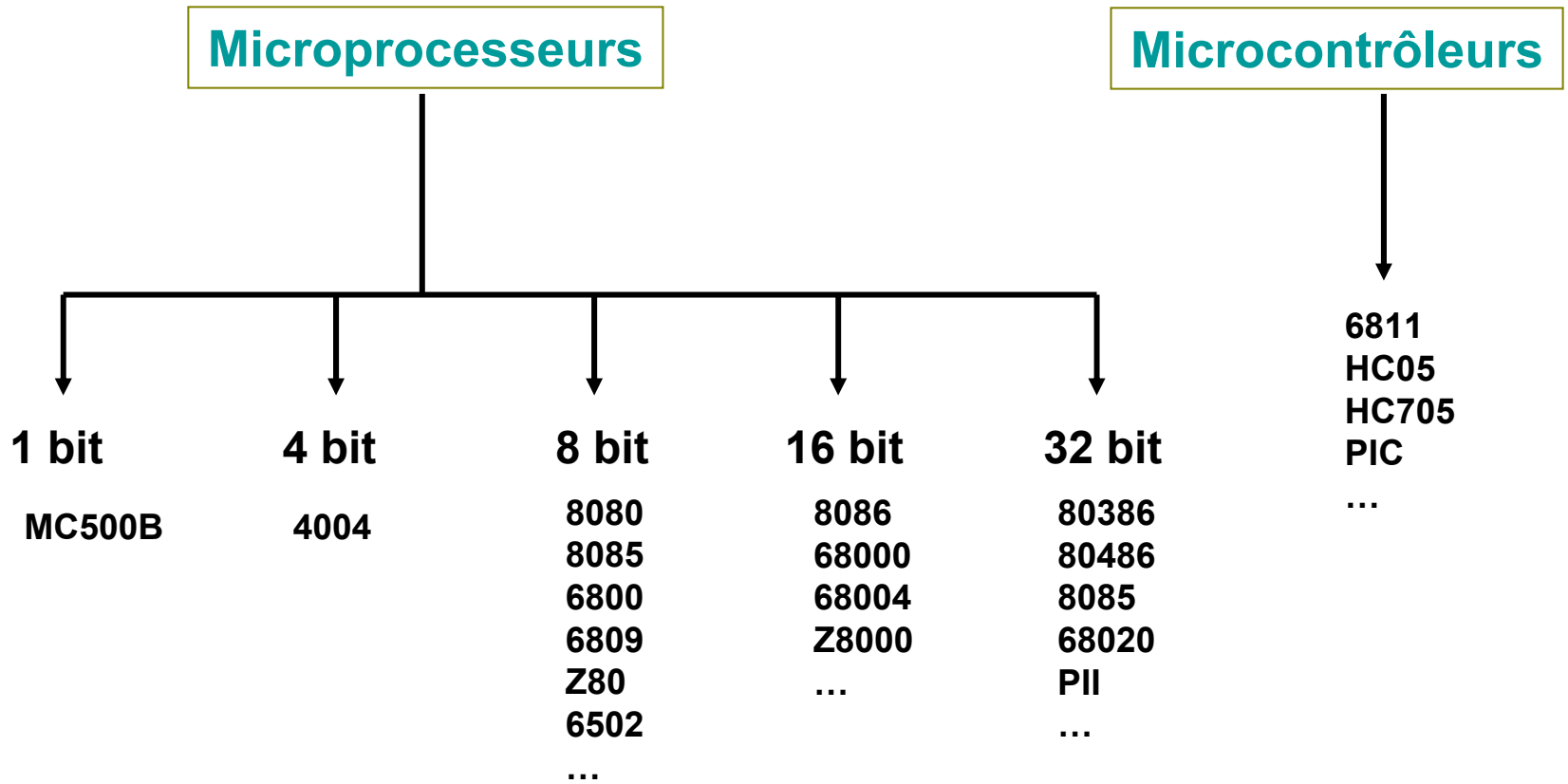
Microprocesseur en tranche

Système microprocesseur

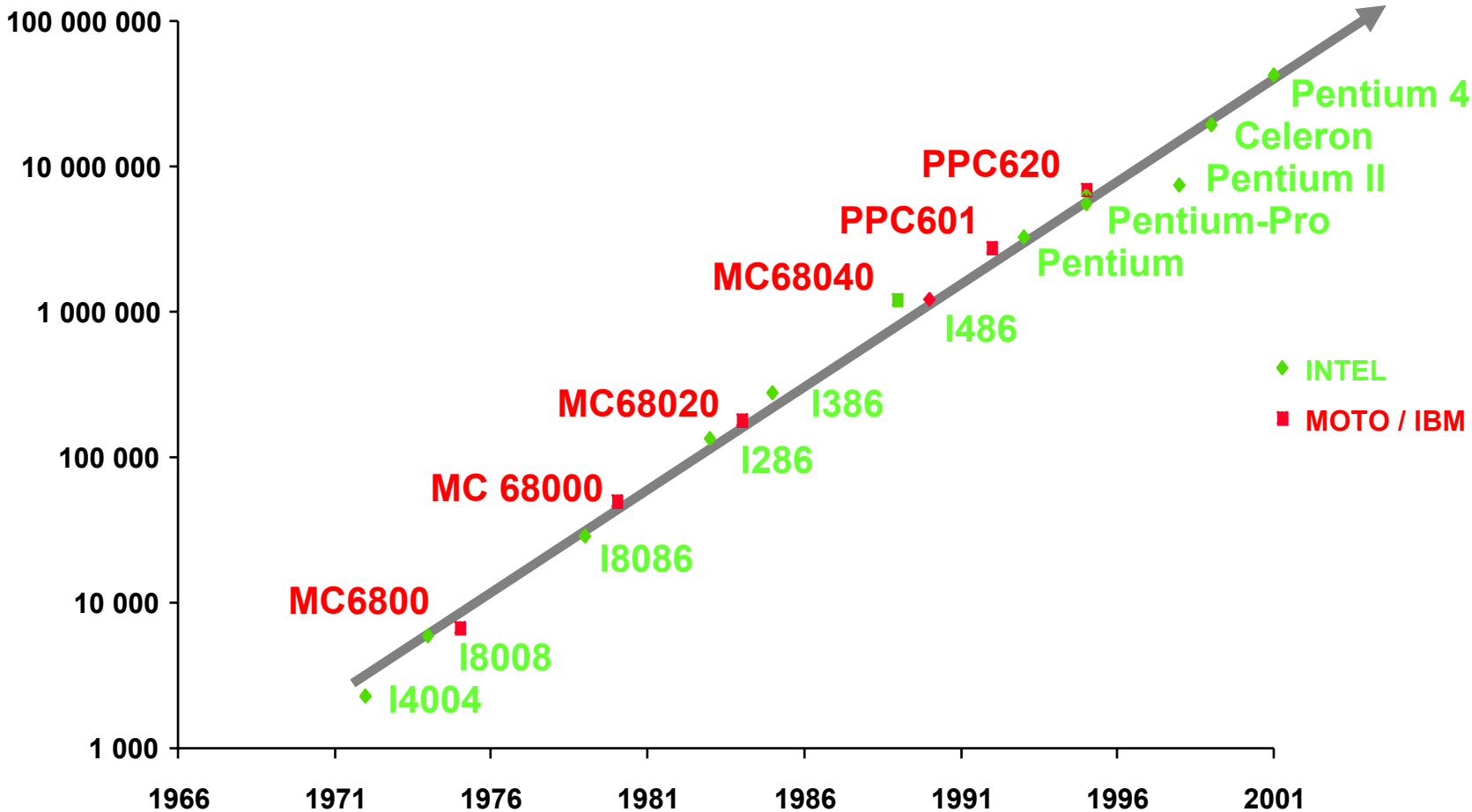
Microcontrôleur

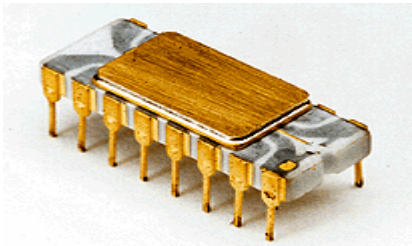
Famille microprocesseur

Types des microprocesseurs



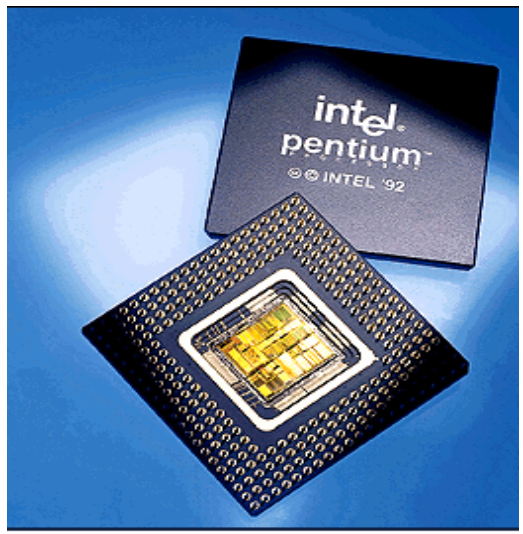
Évolution de la complexité

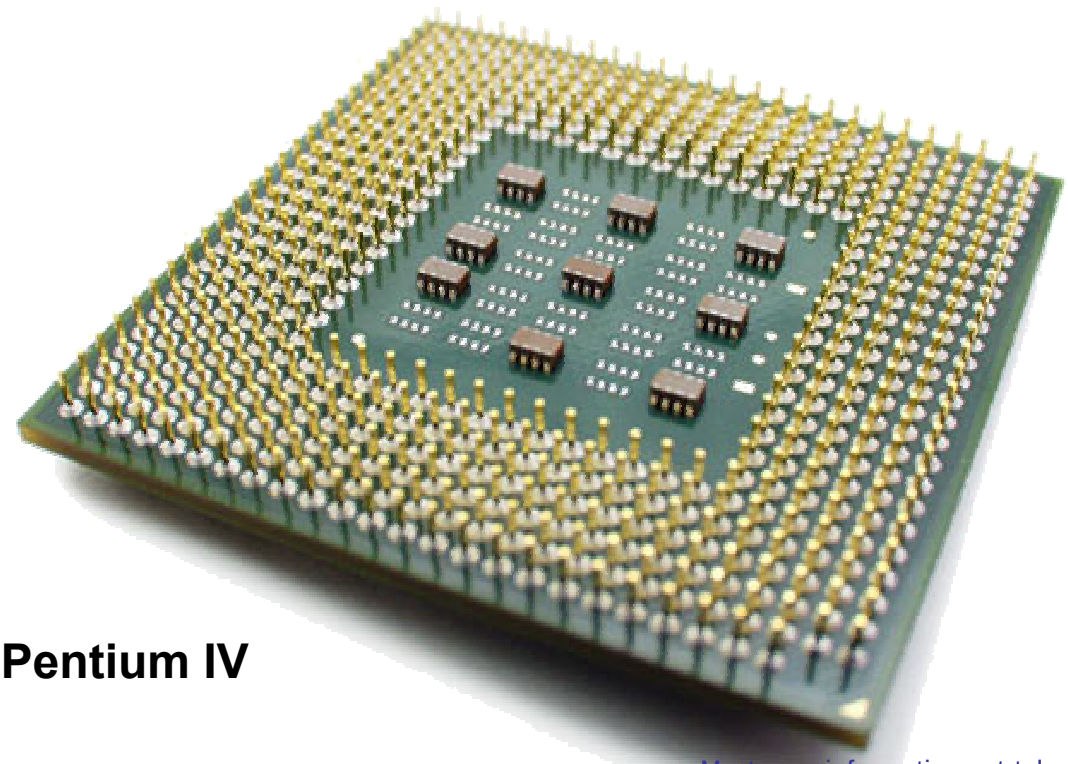




Intel4004

Pentium I

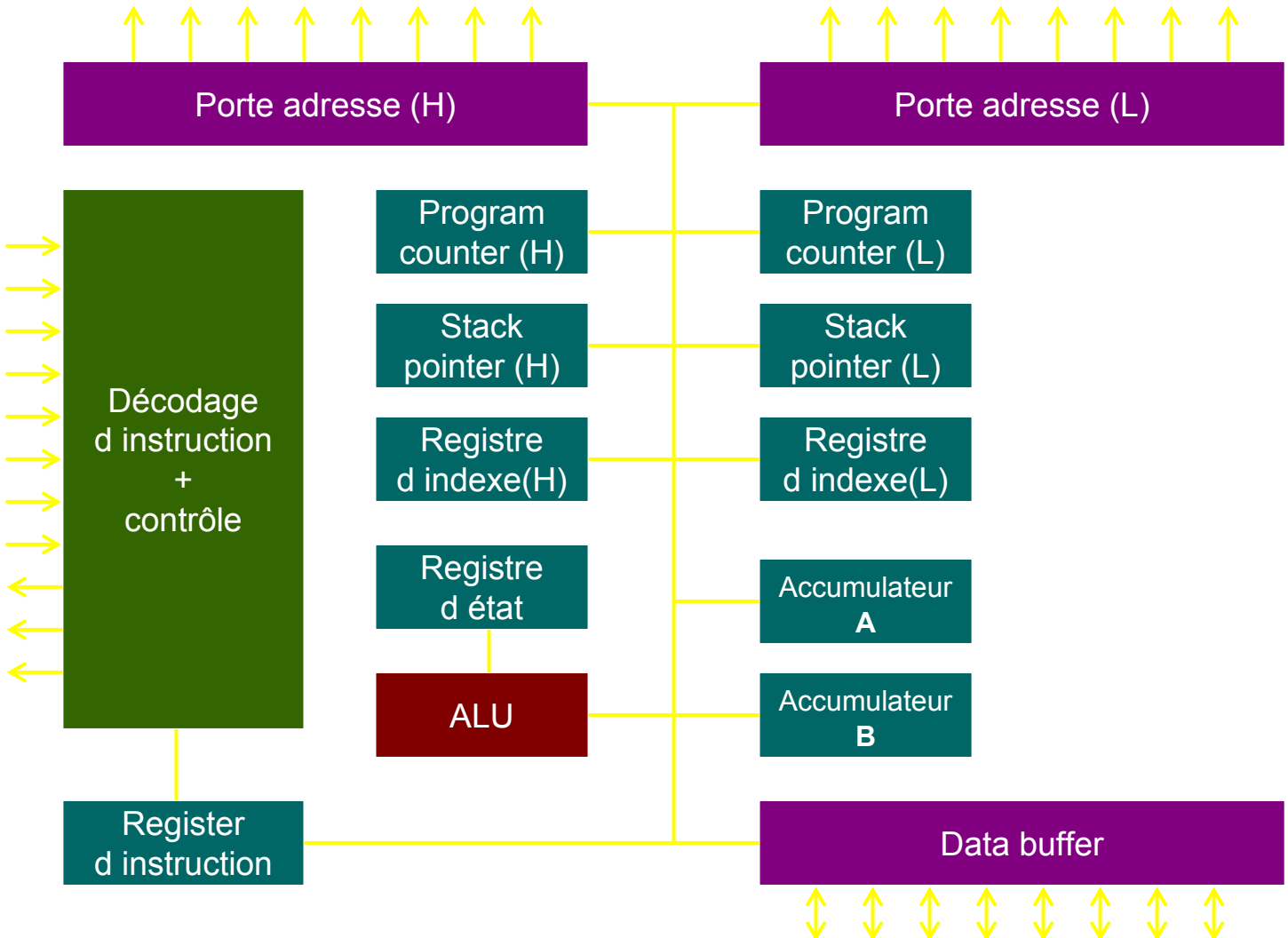




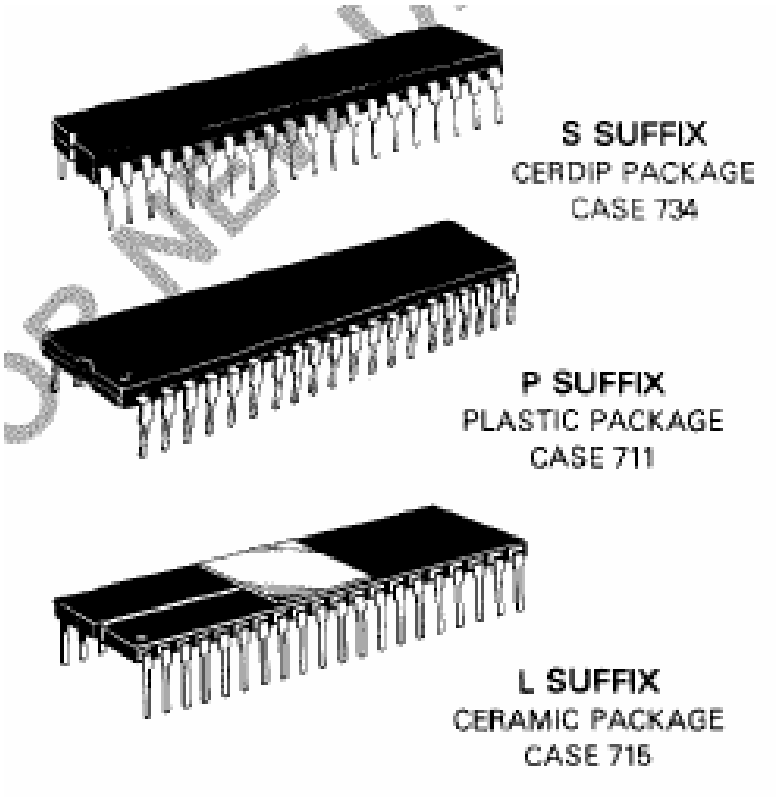
Pentium IV

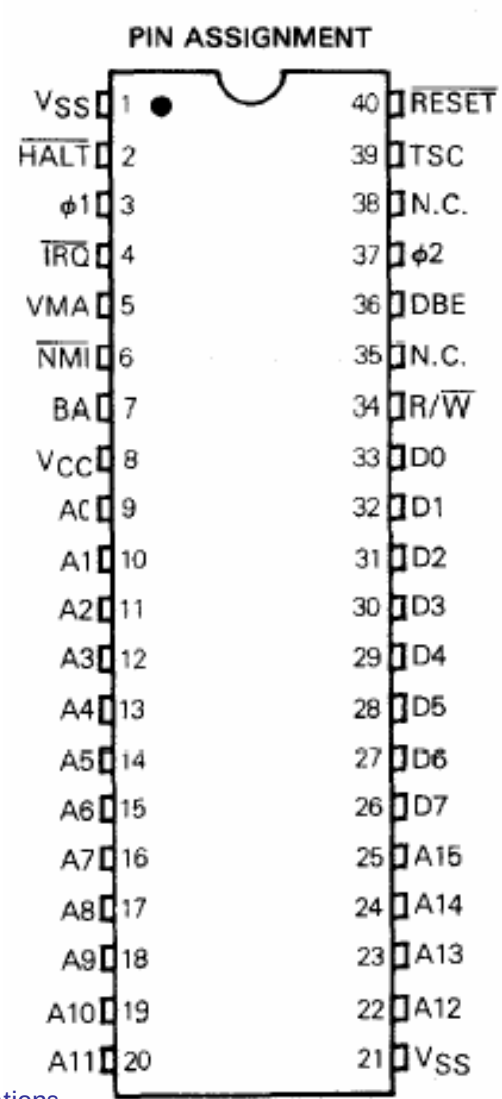
La famille du microprocesseur MC6800 de MOTOROLA

MCM6800	Le microprocesseur
MCM6810	128x8-bits RAM
MCM6830	1024x8-bits ROM
MCM6821	PIA
MCM6850	ACIA


Porte adresse (H)
Porte adresse (L)
Décodage
d instruction
+
contrôle
Program
counter (H)
Program
counter (L)
Stack
pointer (H)
Stack
pointer (L)
Registre
d indexe(H)
Registre
d indexe(L)
Registre
d état
Accumulateur
A
ALU
Accumulateur
B
Register
d instruction
Data buffer

Les pins de MC6800



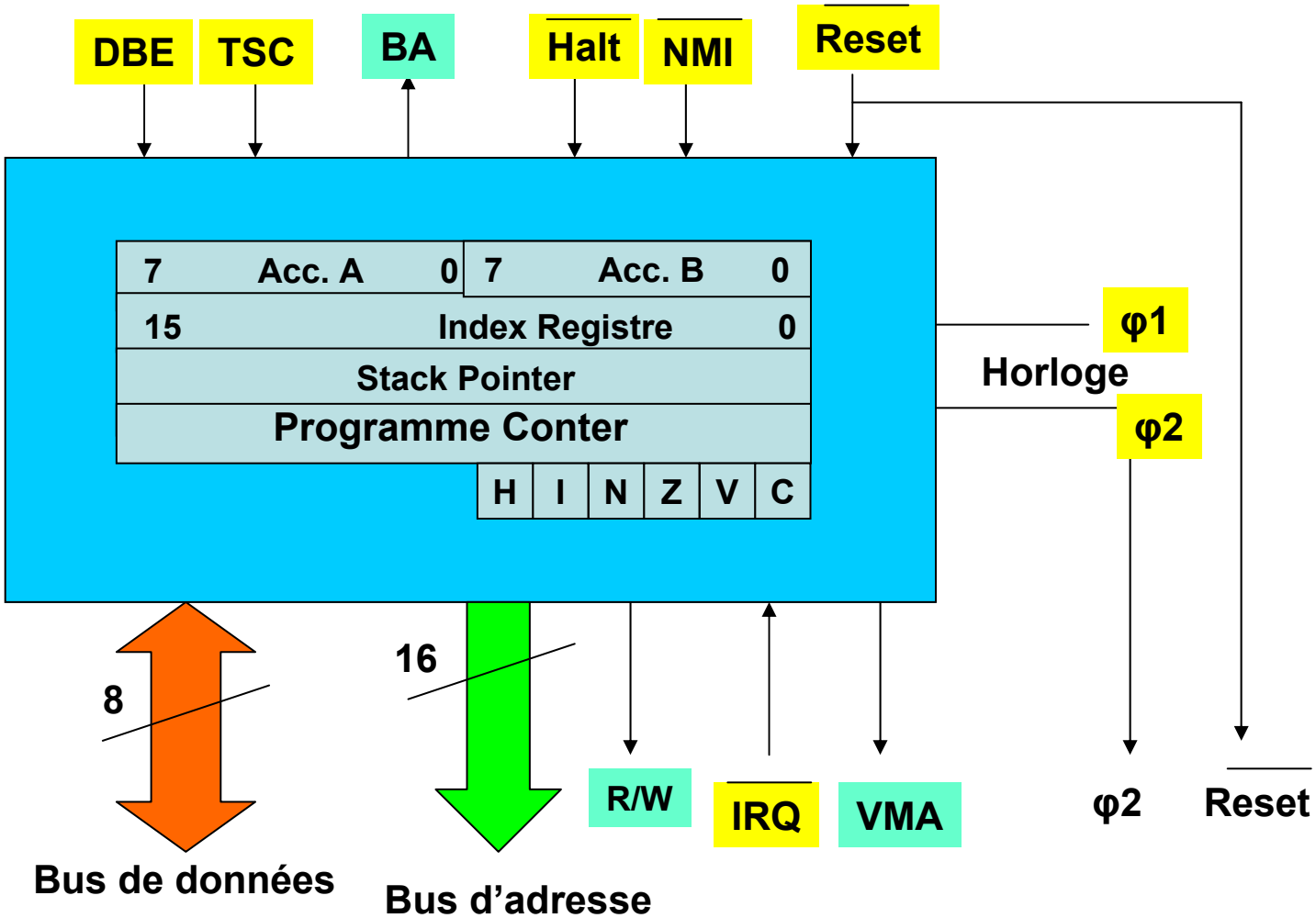


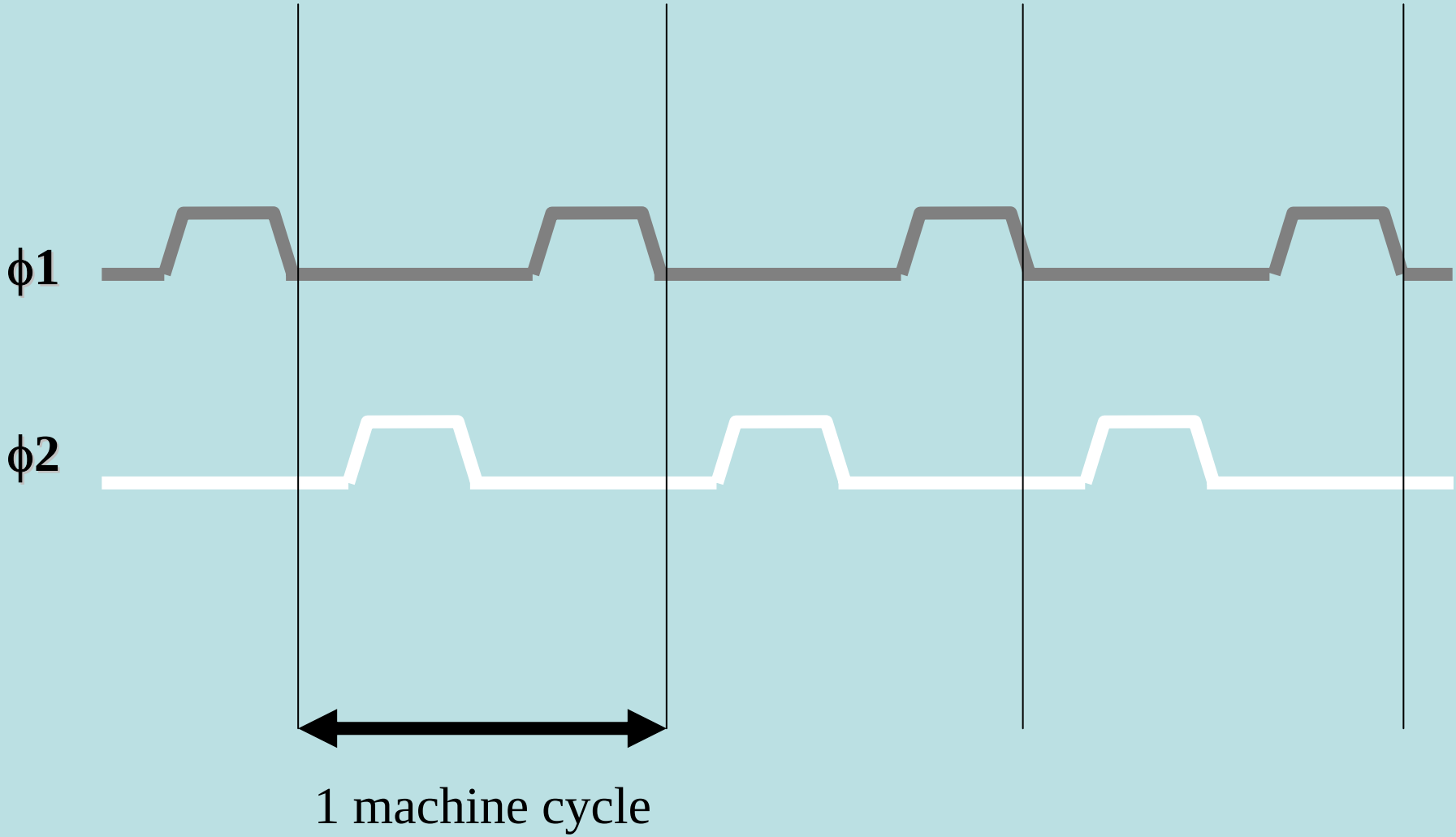
Les pins de MC6800

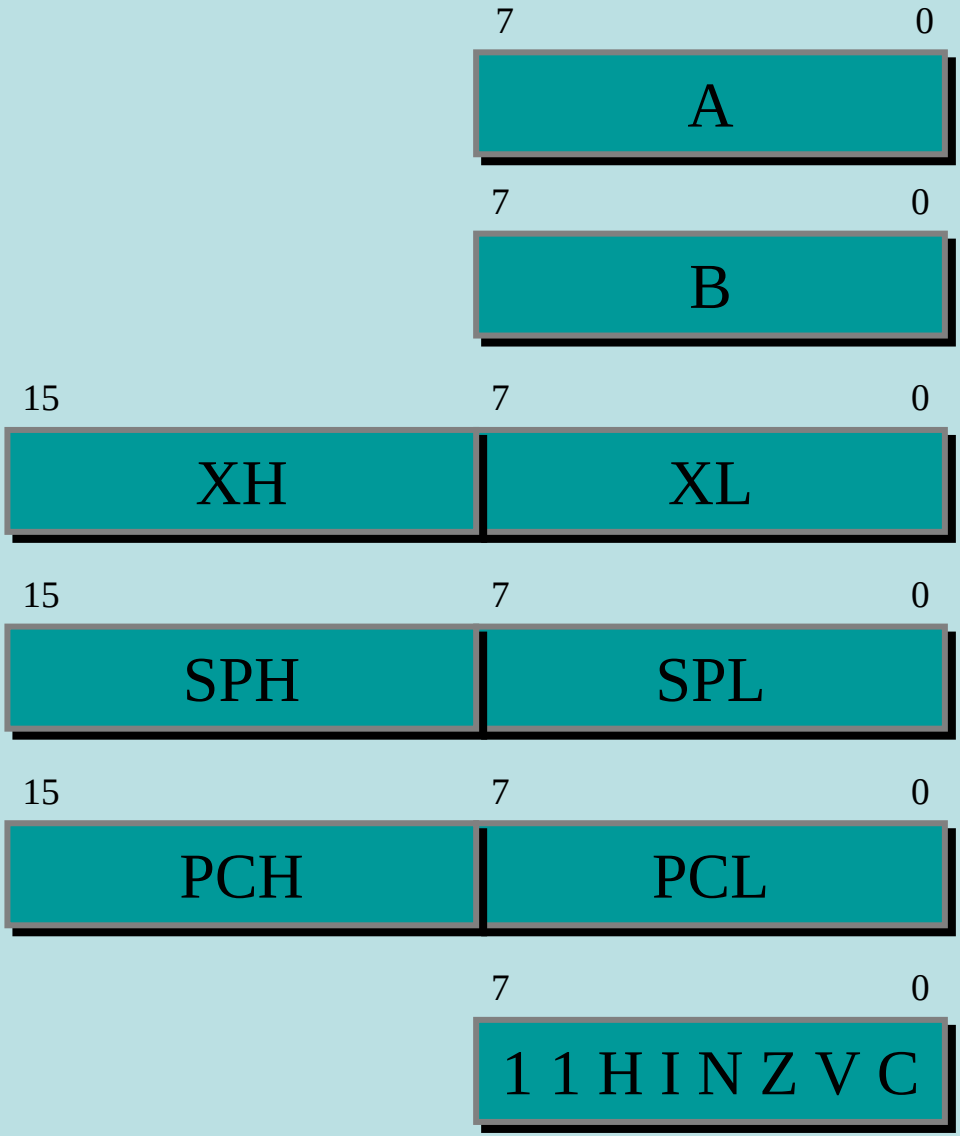
- Nom des signaux et type de pin

Signal name	Function	Data type
A0 – A15	Address lines	Output
D0 – D7	Data lines	Bidirectional
R/W	Read and Write lines	Output
VMA	Valid Memory Address	Output
DBE	Data Bus Enable	Input
RESET	Reset Line	Input
HALT	Halt line	Input
BA	Bus Available	Output
TSC	3-State Control	Output
IRQ	Interrupt Request	Input
NMI	Non- Maskable Interrupt	Input
ϕ1, ϕ2	Phase 1 and 2 Clocks	Input
V_{CC}, V_{SS}	Power and Ground	Input

Le microprocesseur MC6800 de MOTOROLA




ϕ1
ϕ2
1 machine cycle



Accumulator A

Accumulator B

Index Register (X)

Stack Pointer (SP)

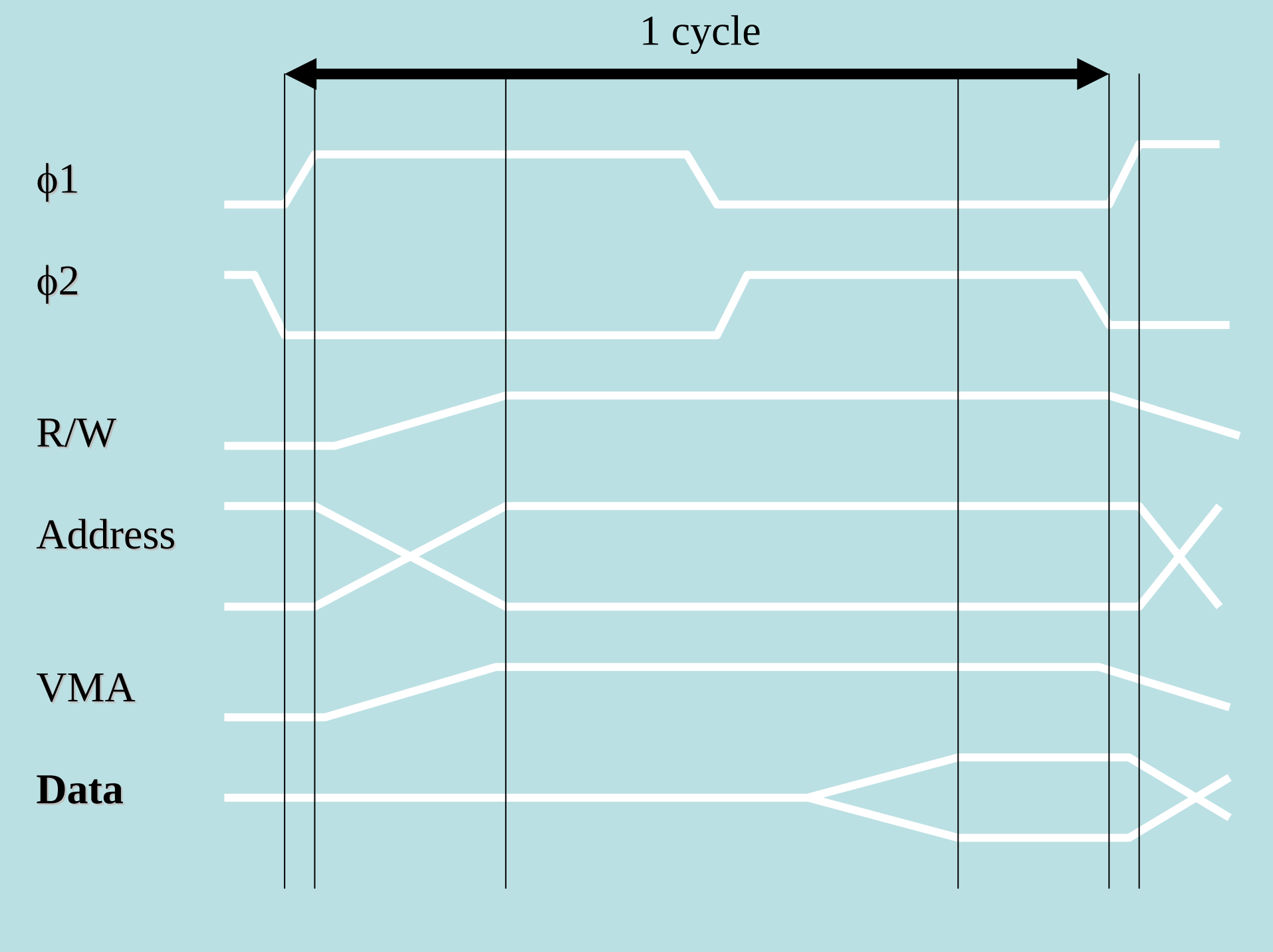
Program Counter (PC)

Condition Code Register (CCR)

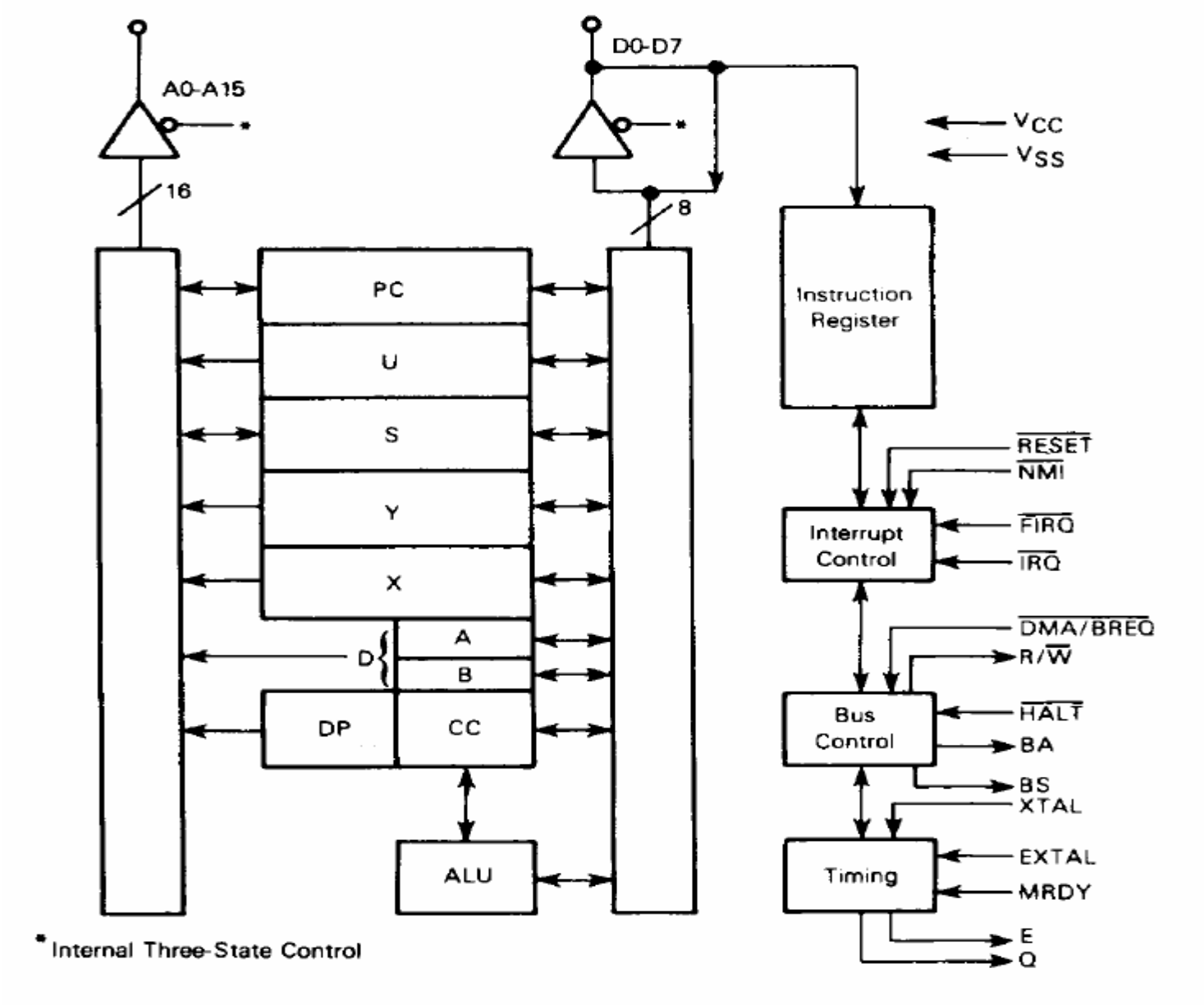

IRQ Mask
Zero
Carry
1
1
H
I
N
Z
V
C
7
6
5
4
3
2
1
0
Half carry
Negative
Overflow

La table d’instruction

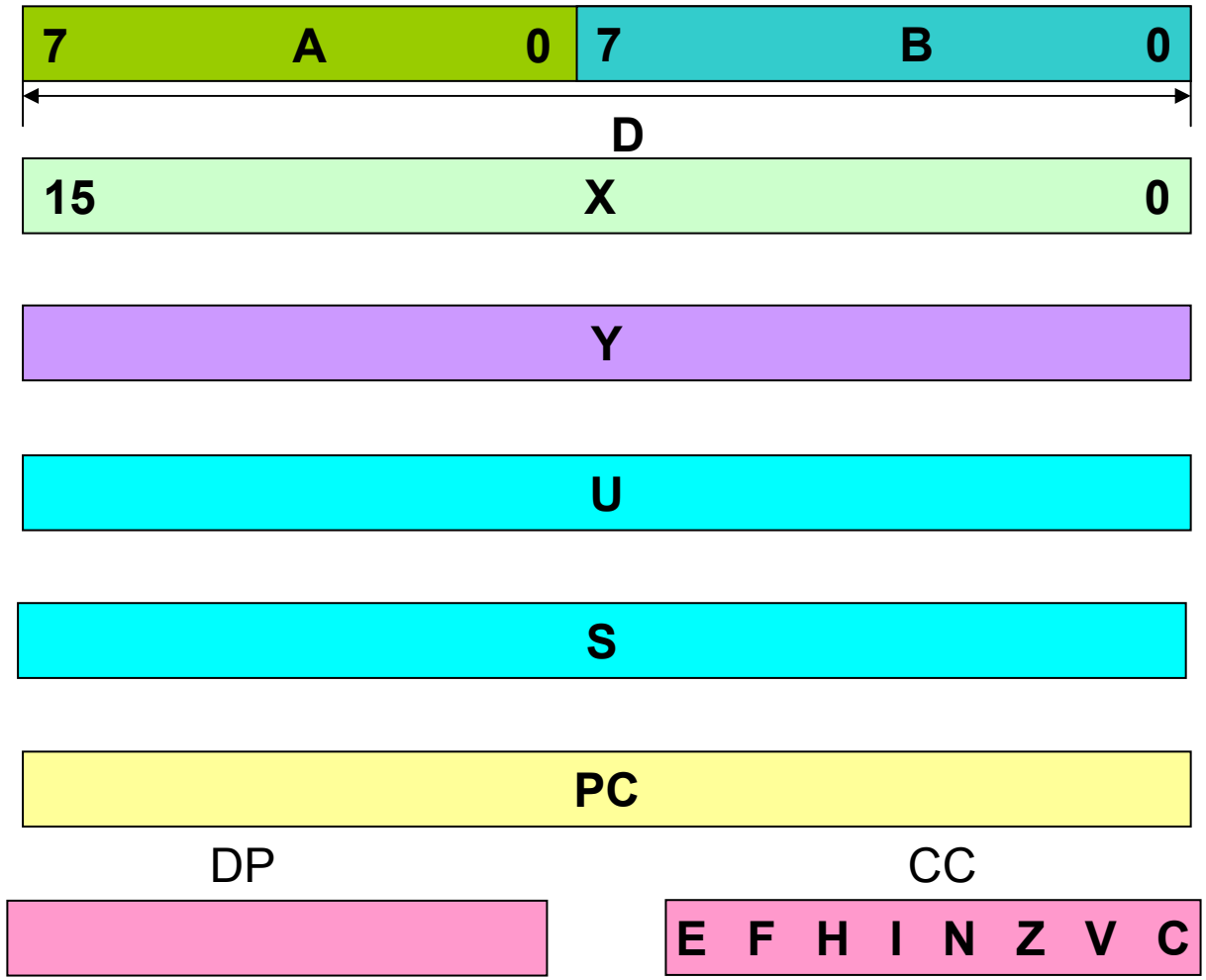
Voir MC6800.PDF


1 cycle
ϕ1
ϕ2
R/W
Address
VMA
Data

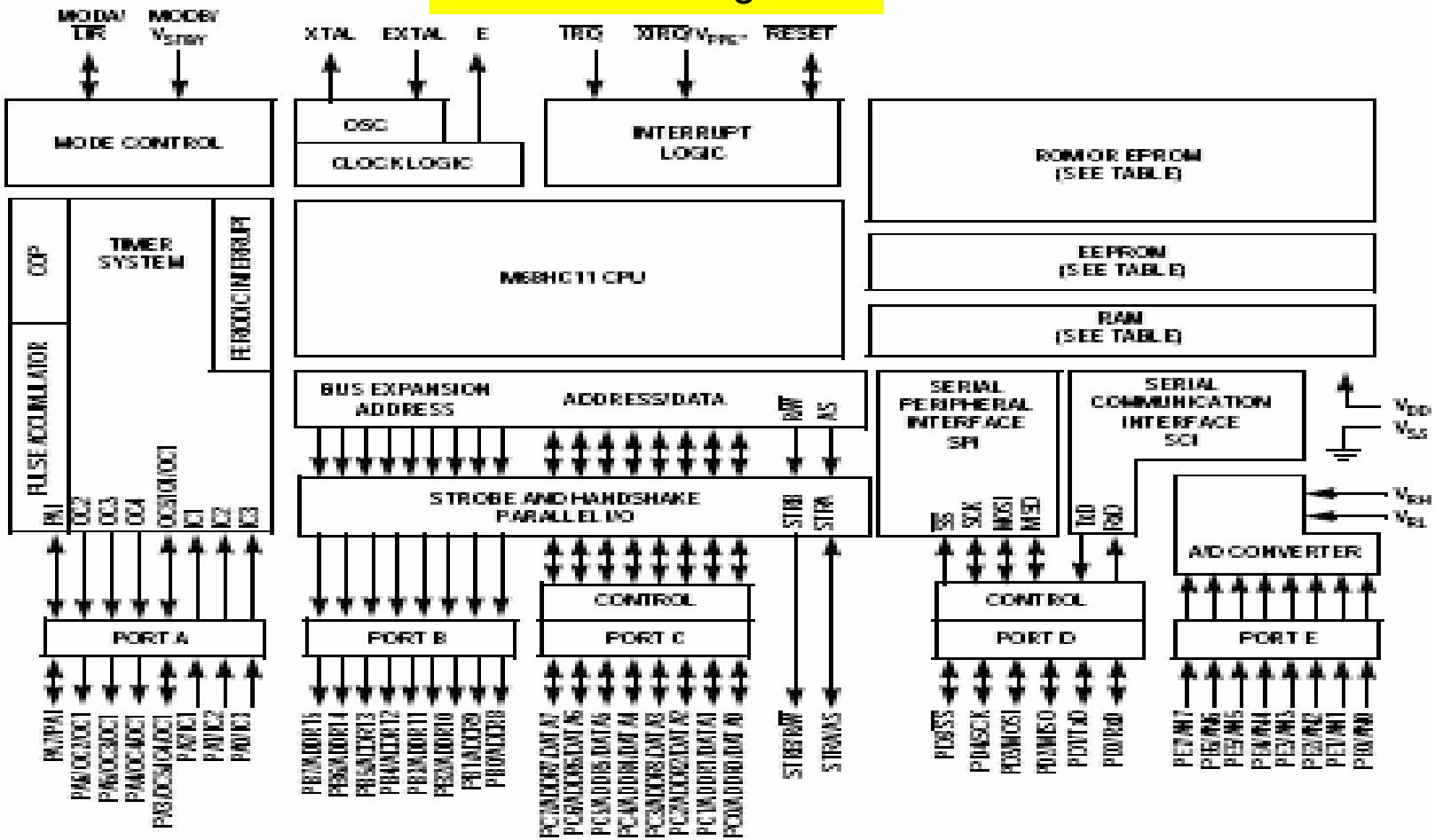
6809 Block Diagramm



Le 6809



6811 Block Diagramm

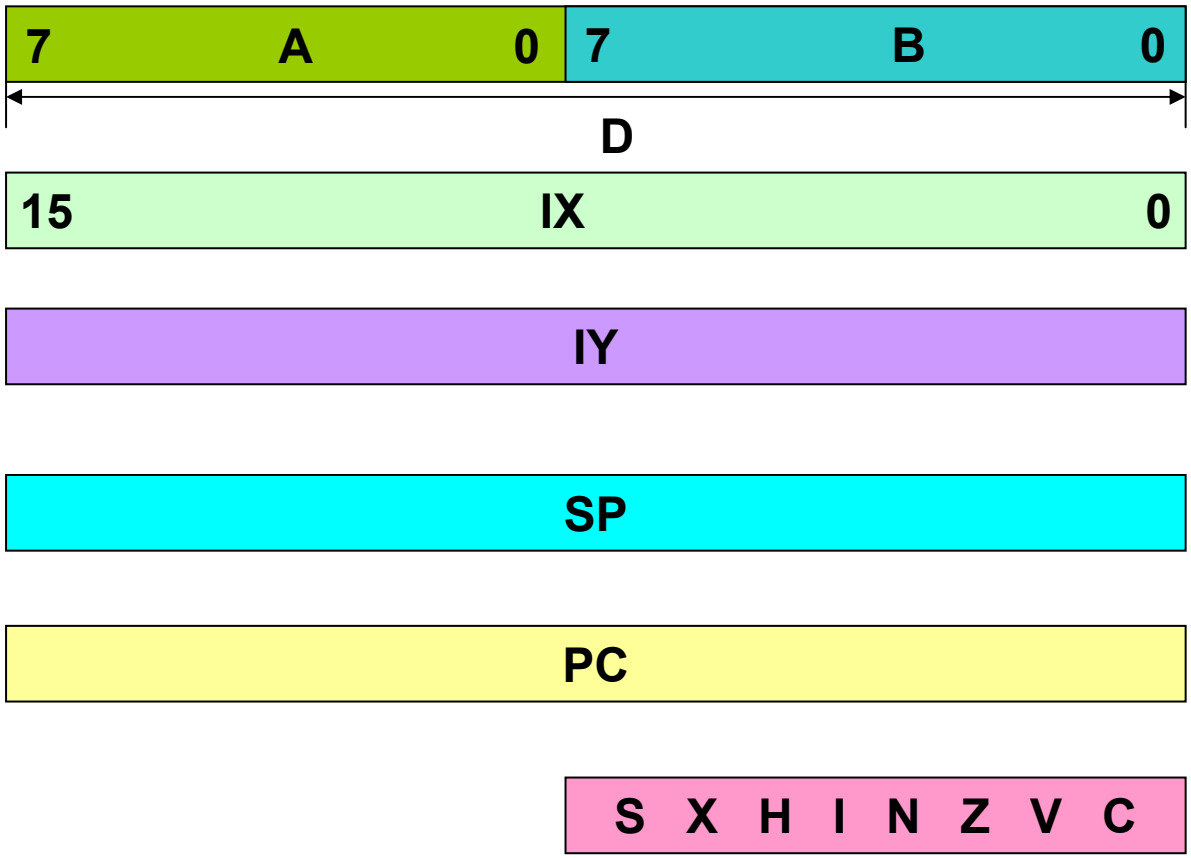


DEVICE	RAM	ROM	EPROM	EEPROM
MC68HC11E0	512	—	—	—
MC68HC11E1	512	—	—	512
MC68HC11E8	512	12 K	—	512
MC68HC711E8	512	—	12 K	512
MC68HC11E20	768	20 K	—	512
MC68HC711E20	768	—	20 K	512
MC68HC811E2	256	—	—	2048

* V_{PPE} applies only to devices with EPROM/OTPROM.

Figure 1-1. M68HC11 E-Series Block Diagram

Le 6811



La conception des circuits à microprocesseurs

La programmation

Quelques liens

http://www.abcelectronique.com/

http://www.alldatasheet.com/

http://www.intel.com/

http://perso.orange.fr/xcotton/electron/constructeurs.htm

http://www.histoire-informatique.org/idx/

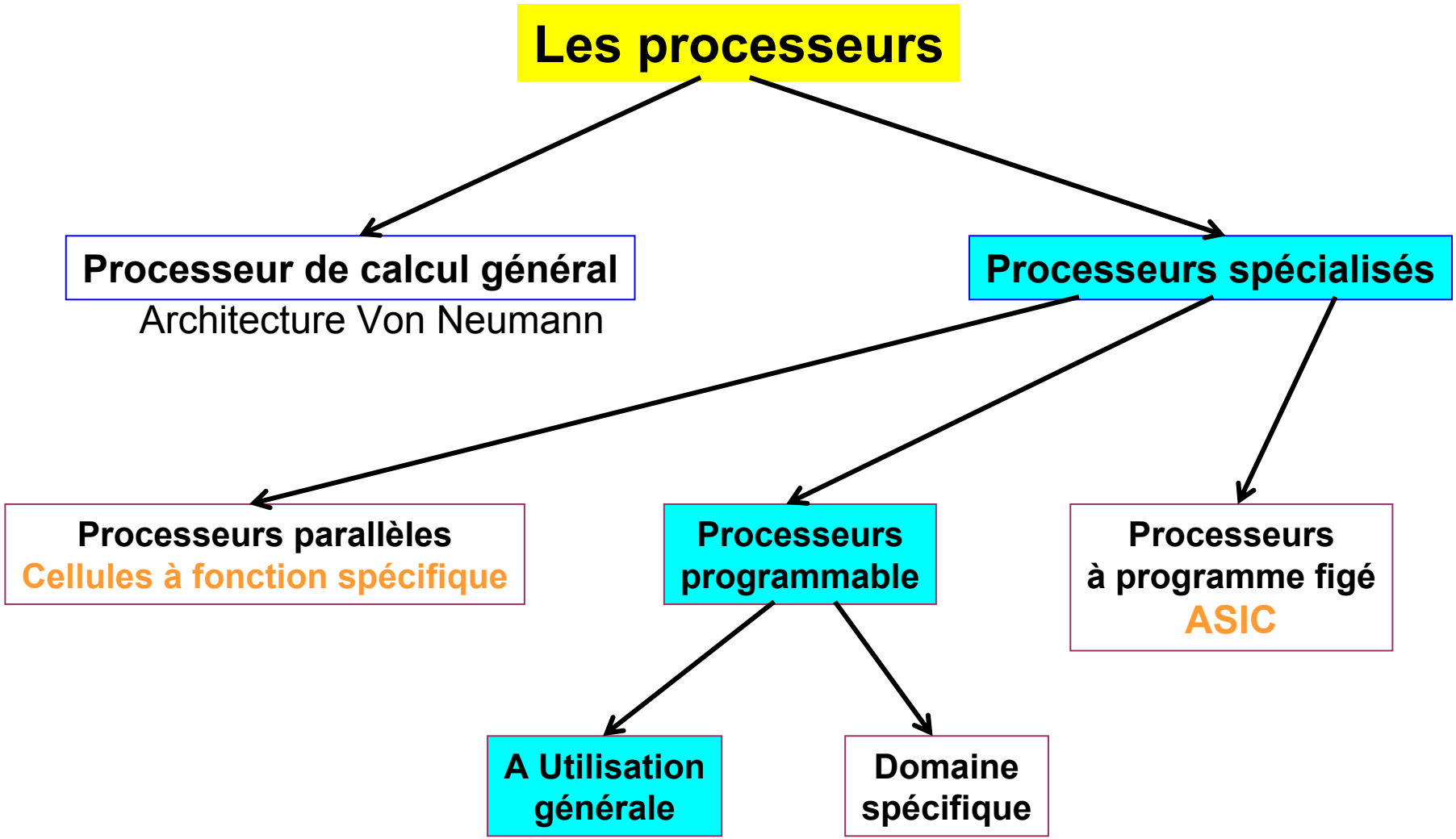
http://para.maxim-ic.com/

http://www.abcelectronique.com/

http://fanelectronique.free.fr/

http://www.mon-ordi.com/
http://www.ldlc.fr/
http://cours.sofad.qc.ca/microinfo/page_princ.htm

Classification des processeurs



Les algorithmes DSP

- **Filtre FIR de longueur N**

$$Y(n) = h[0]\ x[n] + h[1]\ x[n-1] + \ldots\ldots\ldots + h[N-1]\ x[n-(N-1)]$$

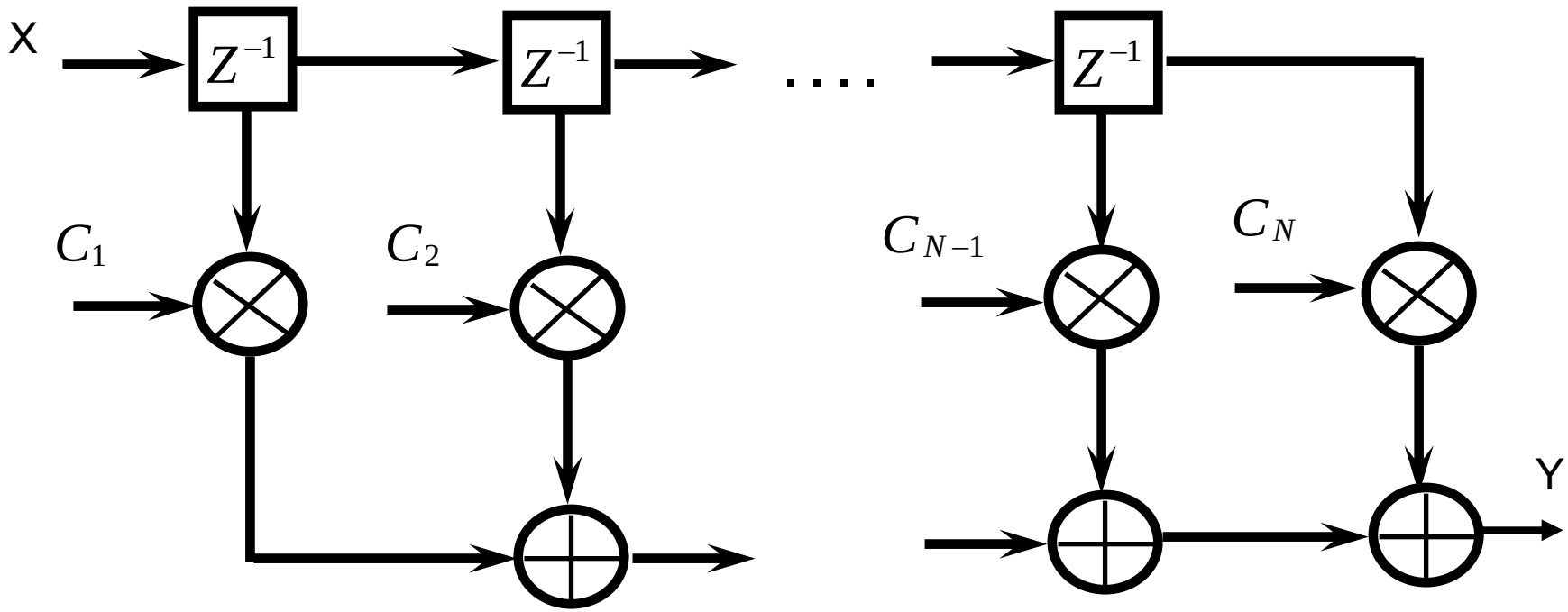
N muladd par échantillon

- **Convolution par une matrice**

$$New[l][c] = \sum \sum a[i][j]\ old[l-i][c-j]$$

N * P muladd par pixel

FINITE-IMPULSE RESPONSE (FIR) FILTER



Filtre FIR sur simple processeur à utilisation génerale

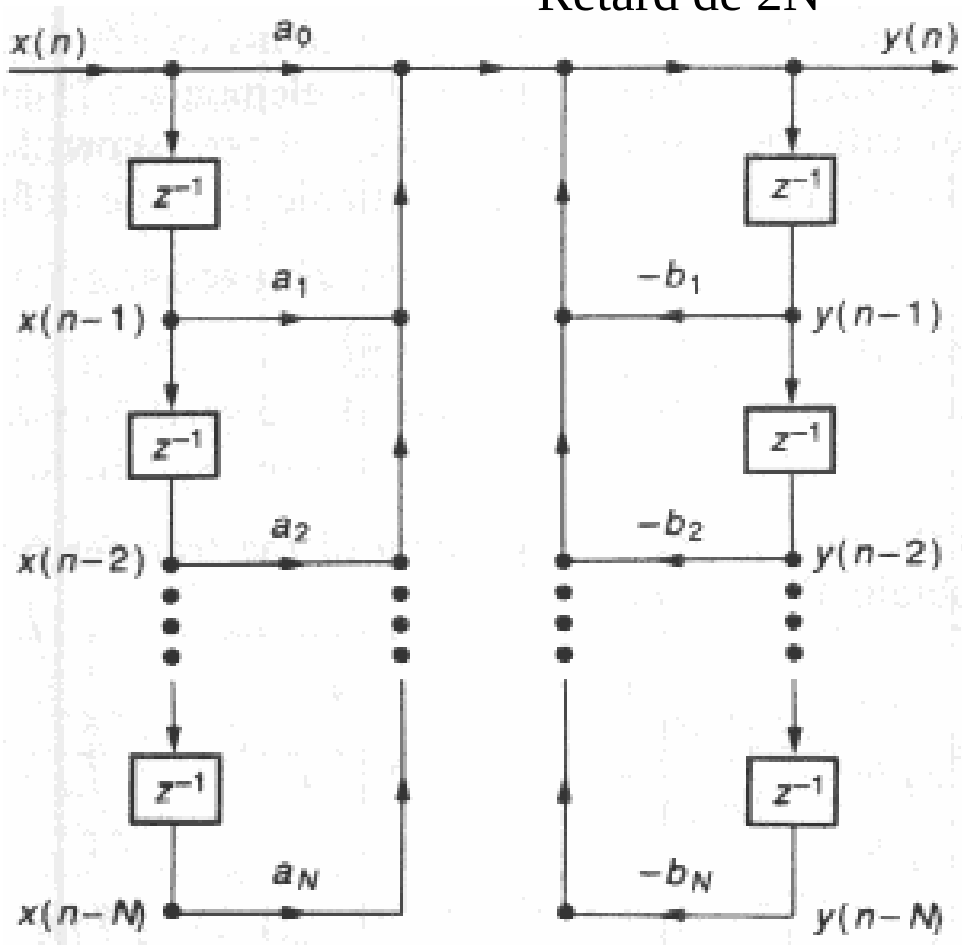
```
loop:
    lw      x0, 0(r0)
    lw      y0, 0(r1)
    mul     a, x0,y0    ; s'il existe
    add     y0,a,b
    sw      y0,(r2)
    inc     r0
    inc     r1
    inc     r2
    dec ctr
    tst ctr
    jnz loop
```

- Les problems: Les Bus / memory bandwidth bottleneck, control code overhead

Structure des filtres IIR

- Forme directe I :

N est l ’ordre du filtre
Retard de 2N



Les coefficients a et b sont ceux des transformées en Z

Besoins des applications

- **Filtre FIR**

  64 coefficients sur un signal a 1Mhz

  64 Mega – muladd / s

- **Convolution en temps réel**

  Image de 512x512 pixels
  Matrice de convolution 5x7
  Vitesse de 25 images/s

  224 Mega – muladd / s

- **Asservissement rapide**

  Vout = [matrice] Vin
  20 entrées , 20 sorties
  Boucle d asservissement de 50 micro

  8 Mega – muladd / s

Caractéristiques des algorithmes DSP

- **Nature des calculs mis en oeuvre**

  1) Calcul répétitif et volumineux régulier et non régulier

  2) Traitement sur des vecteurs

  3) Code petit

Caractéristiques des algorithmes DSP

- **Flux des données**

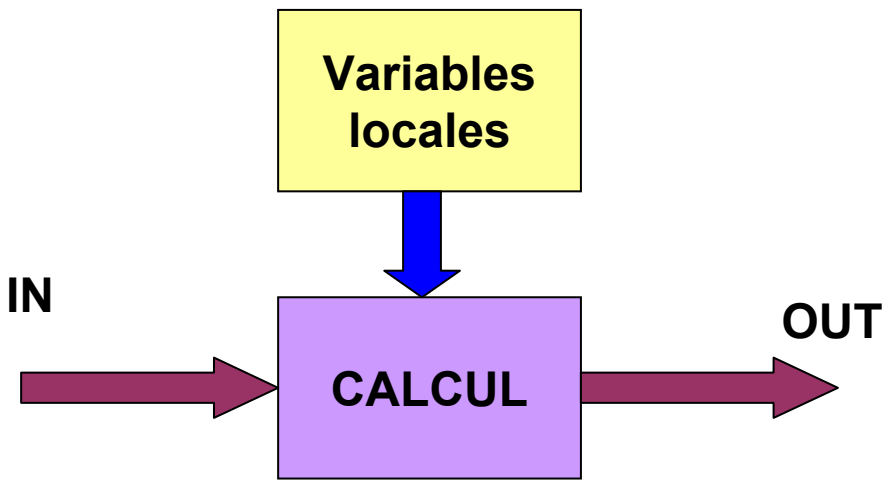
  1) Flot stable d'information

  2) Variables et constantes locales

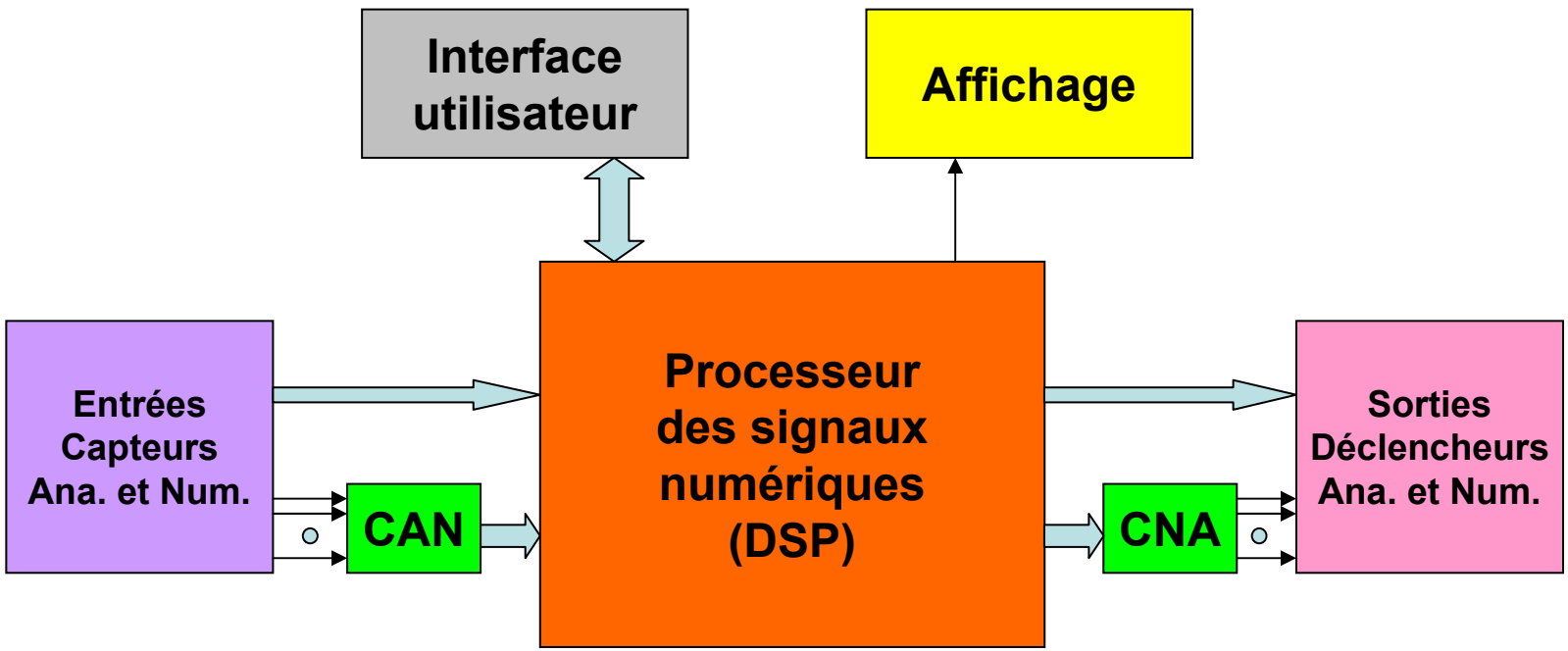
  3) Systèmes réactif

     Stimuli → calculs → réactions

  4) **Temps Réel** → Temps de réaction borné



Système DSP type



Les DSP vs. MPU à utilisation générale

- Le paramètre “MIPS/MFLOPS” des DSP est la vitesse de la Multiplication-Accumulation (**MAC**).
  - Les DSP sont comparés par ceux qui peuvent garder jusqu’à 100 % du temps du multiplieur.

- L’architecture des processeurs DSP sont conçu à base de 4 algorithmes:
  - Les filtre IIR (Inifinite Impulse Response filters)
  - Les filtre FIR (Finite Impulse Response filters)
  - FFT
  - Convolution

- Dans un DSP, les algorithmes sont importants:
  - La compatibilité binaire est sans importance.

- Le logiciel à niveau élevé n'est pas (encore) important dans les DSP.
  - Les concepteurs écrivent toujours en langage assembleur pour un produit pour minimiser la surface de la mémoire ROM sur le circuit DSP et pour avoir le maximum de vitesse de calcul.

Caractéristiques commune des Processeurs DSP

- Architecture à mémoire Multi-accès
  - Architecture Harvard avancée pour permettre la séparation programme et double accès à l'espace mémoire données.
- Modes d'adressages Spécialisés
  - circulaire, adressage par bit- reversed.
- Multiplication-Accumulation rapide (MAC)
  - Haute gestion performante des interruptions.
  - Une ou block d'instructions en structure pipeline pour arriver à une instruction par cycle.
- Périphériques On-chip et interfaces E/S
  - direct memory access Multicanaux (DMA).
  - Ports d'E/S haute performance série et parallèle.
  - Timer haute résolution….

  ….

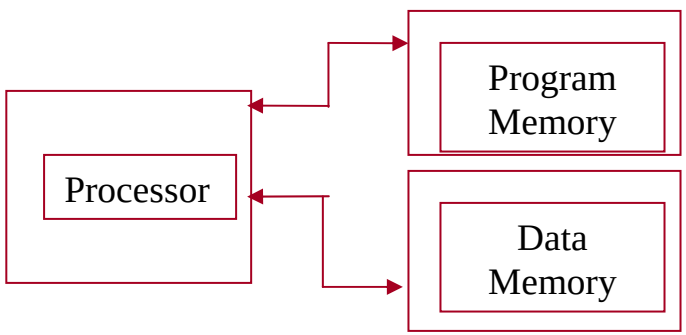
Comparaison des Architectures Mémoires

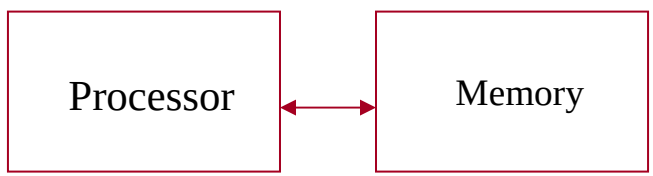
Processeur DSP

- Architecture Harvard
- 2-4 accès-memoires/cycle
- Pas de caches-on-chip SRAM

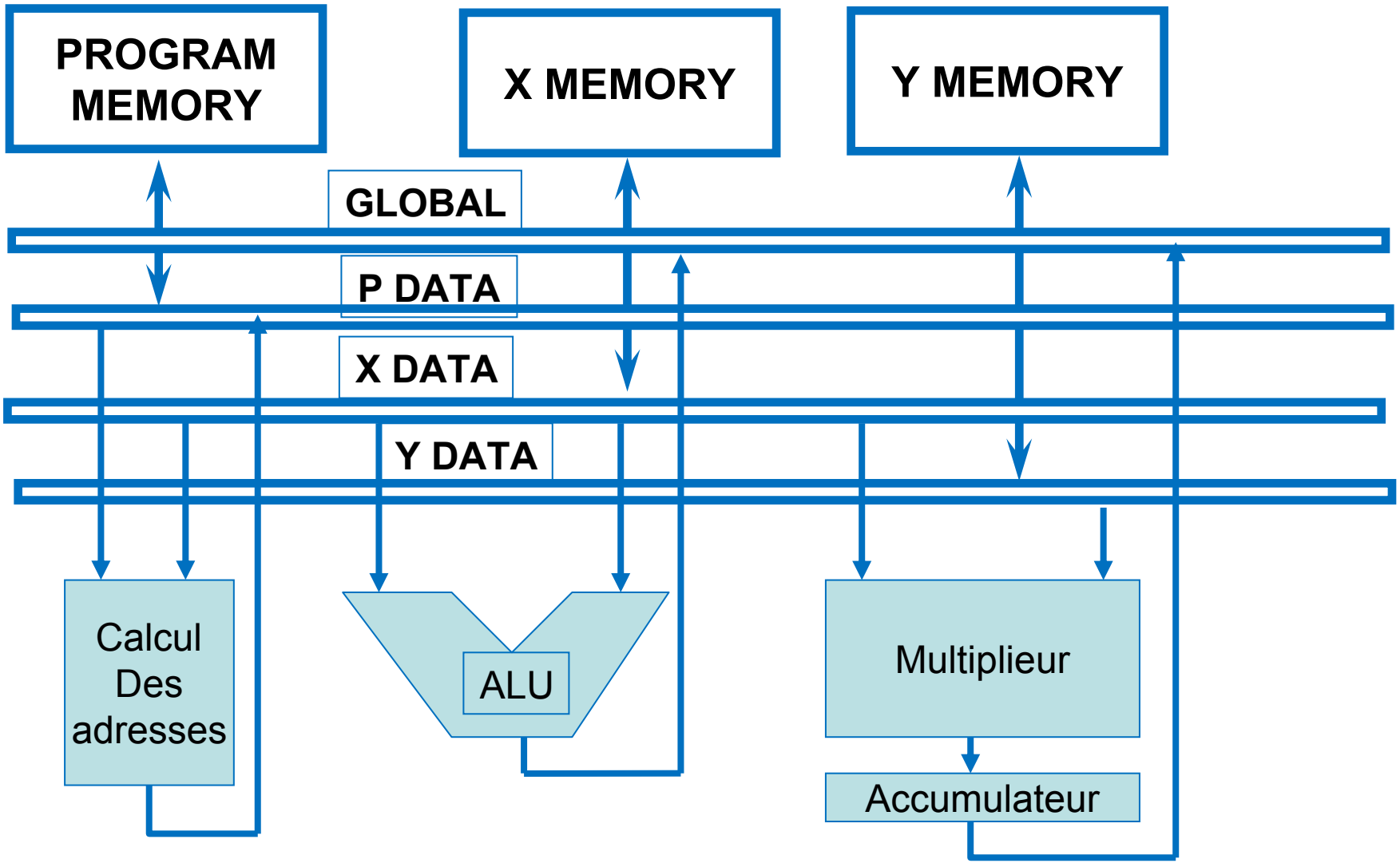
Processeur à utilisation générale

- Architecture Von Neumann
- 1 access/cycle
- Utilisation de caches





L'ARCHITECTURE HARVARD-MEMOIRE dans les DSP



Les types des processeurs DSP

- **16-BIT Virgule Fixe**
  - TI TMS320C2X, C5X
  - MOTOROLA 56000
  - AT&T DSP16
  - ANALOG DEVICES ADSP2100
- **32-BIT Virgule Flottante**
  - TI TMS320C4X
  - MOTOROLA 96000
  - AT&T DSP32C
  - ANALOG DEVICES ADSP21000
- **Les multiprocesseurs DSP**
  - TMS320C80
  - TMS320C6000

Le marché des DSP

- **TI** : TMS320C1x, C2x, C24x, C25x, C28x, C3x, C4x
  C5x,C54x, C55x, C6x, C67x, C64x, DM64x, C8x
- **ADI** : ADSP210x, 218x, 2106x, 2116x, TigerSharc
- **Motorola** : 5600x, 563xx, 9600x
- **AT&T** : DSP32C
- **ZSP** : LSI401Z, LSI402Z, LSI403Z
- **MXIC** : MAX93011
- **STM** : ST100

DSP et chemin de donnes (Data Path): Arithmétique

- Les DSPs s'occupe des nombres représentant le monde réel
  => "réels"/ fractions
- Les DSPs s'occupe des nombres pour les adresses:
  => entiers
- Les DSP doivent supporter la virgule fixe et les entiers

radix
point

$-1 \leq x < 1$

radix
point

$-2^{N-1} \leq x < 2^{N-1}$

DSP et chemin de donnes : La précision

- La dimension des mots affecte la précision surtout dans le cas des nombres en virgule fixe.
- Les DSP peuvent avoir 16-bit, 20-bit, 24-bit, ou 32-bit pour mot de donnée.
- Le prix des DSP virgule flottante est 2X - 4X vs. ceux du virgule fixe, en plus ils sont moins rapide.
- Le programmeur DSP doit poursuivre les valeurs à l'intérieur du programme surtout pour la v. fixe.
- Possibilité d'utilisation du "Blocked Floating Point": un seul exposant pour un groupe de fractions.
- La virgule flottante permet un développement simplifié.

DSP et chemin de donnes : L' overflow

- Les signaux DSP sont issus des signaux analogiques : Modulo arithmétique.
- Pour éviter la divergence lors des calculs le mode saturation est utilisé.
- Ce mode consiste en la mise au max valeur positive ($2^{N-1}-1$) ou min valeur négative (-2^{N-1}) si l' overflow est survenu.
- Plusieurs algorithmes peuvent être développés avec ce model.

DSP et chemin de donnes : Le multiplieur

- Hardware spécialisé réalisant toutes les opérations arithmétiques principales en 1 cycle.

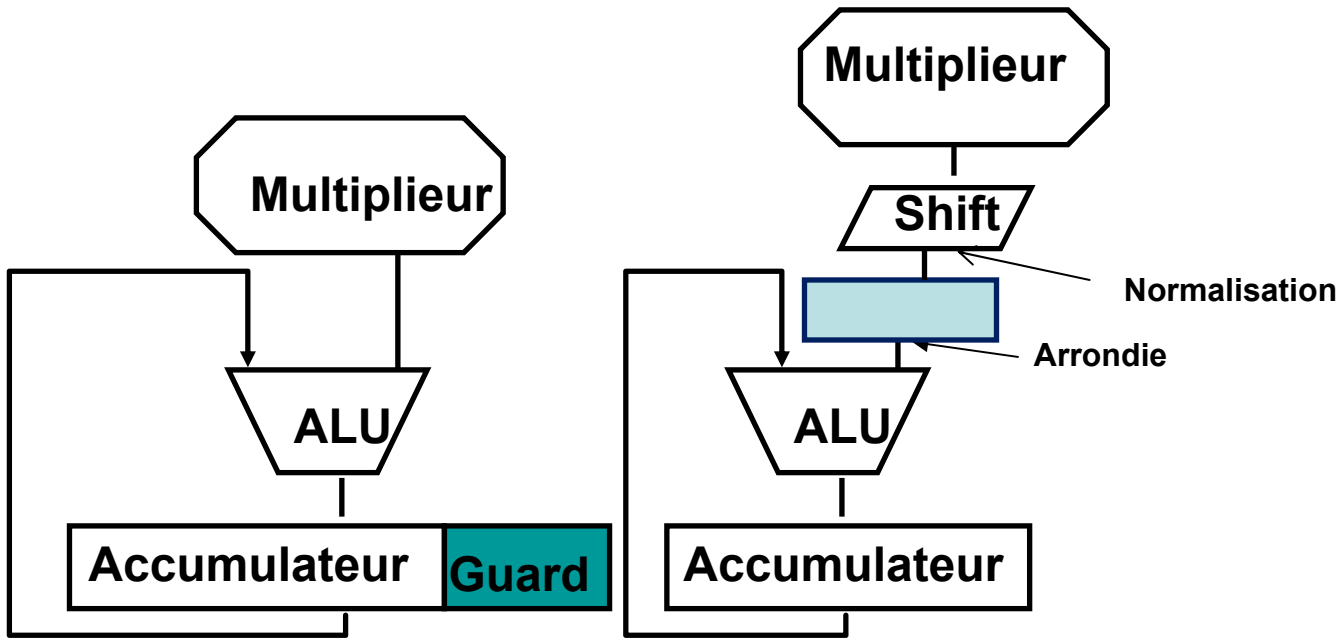
- 50% des instructions peuvent intégrer le l'opération multiplieur.
- => un seul cycle de délai pour le multiplieur.

- n-bit multiplieur donne 2n-bit de produit.

DSP et chemin de donnes : Accumulateur

Pour éviter l' overflow de l'accumulateur on utilise :

- Option 1: l'accumulateur est plus large que le produit: "guard bits"
  - Ex - Motorola DSP: 24b x 24b => 48b pour le produit, Accumulateur à 56b.
- Option 2: décalage à gauche pour normaliser et arrondir le produit avant l'addition.



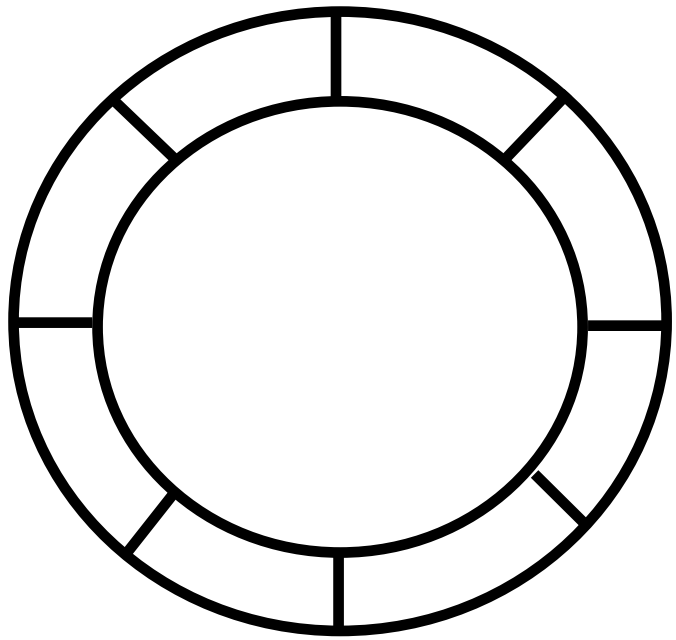
L’adressage circulaire (C. buffers)

Les instructions gèrent trois éléments:

- **les buffers d’adresses**
- **le buffer de dimension (size)**
- **l’incrémentation**

Tient compte du cycling à travers:

- **des éléments de délai**
- **des coefficients en mémoire de données.**



Comparaison de modes d'adressage

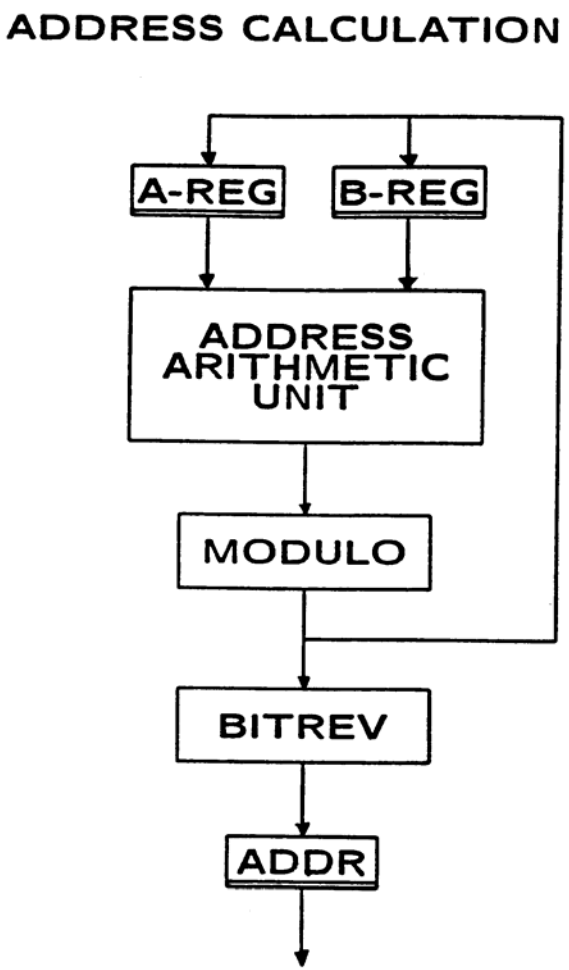
Processeur DSP

- Unité dédiée de génération d'adresse
- modes d'adressage spécialisés:
  - Auto incrémentation
  - Modulo (circulaire)
  - Bit- reversed (pour FFT)
- support immédiat de données

Processeur à utilisation générale

- En général pas de d'unité de génération d'adresse
- modes d'adressage à utilisation générale

Unit de calcul d'adresse pour les DSPs



- Support de l'arithmétique modulo et bit reversal.
- généralement dupliqué pour calculer multiple adresses par cycle.

Instructions DSP et exécution

- Peut être spécifié multiple opérations dans une seul instruction.
- Support complet de l'opération Multiplication-Accumulation (MAC).
- Possibilité de déplacements parallèle vers la mémoire.
- Avoir un support de boucle spéciale pour réduire l'effet des instructions de branchement.
- Boucler une instruction ou une séquence d'instruction.
- Exécution conditionnel des instructions pour réduire les effets des branchements conditionnels.

Les Processeurs de Texas Instruments

Les premières familles des processeurs DSP développées par TI (et qui assure jusqu'à maintenant le support technique) sont au nombre de 5.

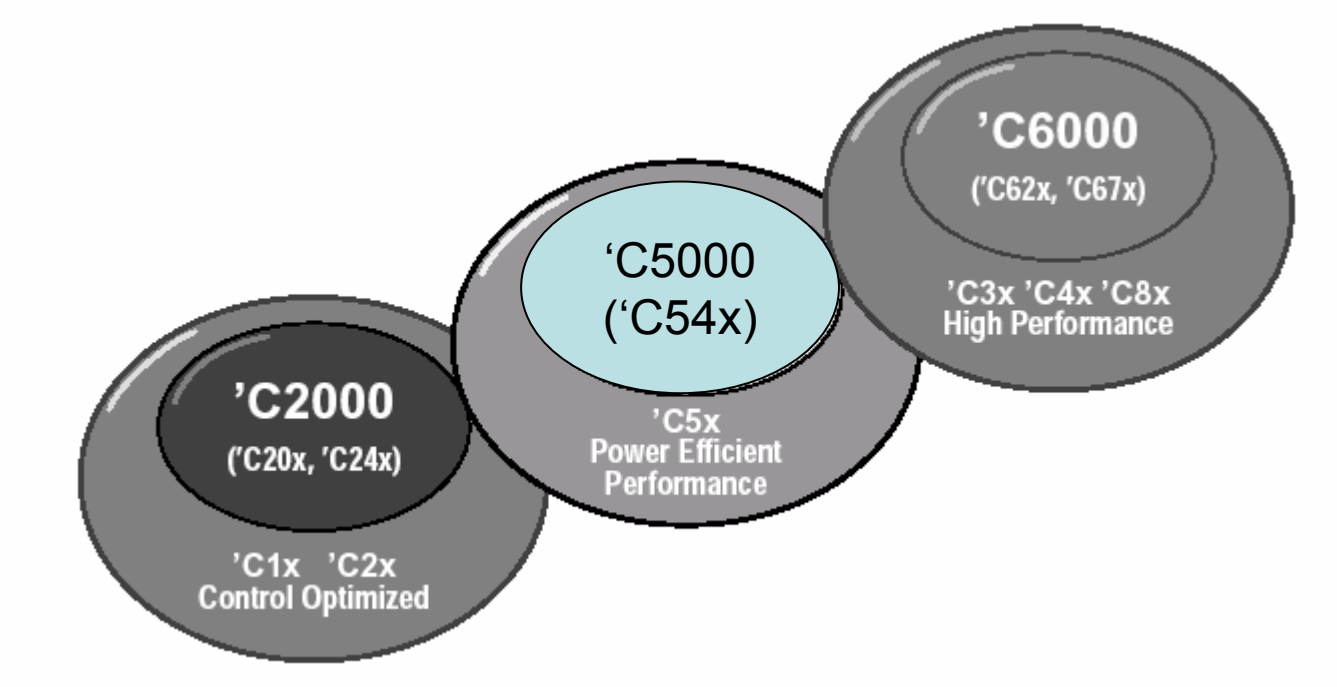
TMS32010, pas de C (NMOS non CMOS)
TMS320C30,
TMS320C40,
TMS320C50,
MS320C80.

Le développement de nouvelles versions pour ces familles est arrêté.

Actuellement, TI développe les nouveaux processeurs DSP en trois familles :

TMS320C2000 (connue comme TMS320C20)
TMS320C5000 (connue comme TMS320C54)
TMS320C6000

Panorama et historique des processeurs DSP de TI



Les Processeurs DSP virgule fixe Conventionnels

TMS32010, (1982),

TMS320C20, (1985),

TMS320C25

TMS320C203, (1995), en réponse au besoin des constructeurs du Disk-Driver, (40 MIPS) (80MHz), $5.00,

TMS320C24, pour contrôle de mouvement,

TMS320C50,

Les Processeurs DSP virgule fixe Conventionnels

TMS320C50,

TMS320C54, version plus petite du C50 pour applications sans fil (wireless basestations *and* handsets), table d'instructions non compatible avec le C50.

- Il réalise en un seul cycle :
  - 2 lectures données du block 1
  - 1 écriture donnée vers le block 2
  - 1 extraction d'une instruction du block 3.
- Intègre une instruction spécial pour le décodage Viterbi,

Une variété du C54x, La famille C54xx, possède 8 Mmots de mémoire adressable atteint par l'addition pointer de pages.

Le TMS320C5416 possède 128K mots de mémoire on chip SRAM et supporte 160 MHz d'horloge.

Applications : Voice over Internet Protocol (VoIP),
serveurs de communications,
computer telephony
customer premise equipment.

Les Processeurs DSP virgule fixe Conventionnels

TMS320C55.

Le C55 est de la famille C5000 avec conception consommation ‘lower power’ par rapport au C54.

Le TMS320C5509 DSP est destiné pour ‘portable handheld Internet appliances’.
Il possède un nombre étendu de périphériques on-board.
Fréquence: 144/200 MHz (jusqu’à 288/400 MIPS)
Mémoire On-chip: 128 Kmots de RAM et 32 Kmots de ROM
Interfaces: USB 1.1 port, I2C, Mémoire Stick, MMC, SD, trois ports série
Convertisseur ADC: on chip de 10-bit

Le TMS320C5502 est le moins cher de la famille C5000 pour les systèmes personnels à $9.95/unité en quantités de 10,000,
Fréquence: 200 MHz (jusqu’à 400 MIPS)
Mémoire On chip: 32 Kmots DARAM et 16 Kmots ROM
Interfaces: UART, I2C, trois ports série.

Les Processeurs DSP virgule flottante conventionnels

Les deux premiers processeurs à virgule flottante de TI sont TMS320C30 (1988)
Et le TMS320C40 processors.
Ces deux processeurs sont très similaires. La différence de base est que le C40
Possède des caractéristiques de communications qui lui permette de fonctionner
plus facilement dans un environnement parallèle.

TMS320C30 (1996)

TMS320C31 ($20)

TMS320C32 ($10)

Le **TMS320VC33** (1999, $5) possède 1-Mbits de RAM et réalise 120 MFLOPS.
La version 150-MHz de C33 coûte $8.
La dernière version du C33 est SM320VC33-EP (2002)

The TMS320C40 = C30 + traitement parallèle

C44

Les processors DSP non conventionnels

La famille TMS320C80,

Le C80 contient 4 DSPs virgule fixe + processeur RISC sur le même chip

-- cher, consomme beaucoup d'énergie, et les moyens de développements sont pauvres.

La famille C6x (C6000),

Le C6000 est un processeur DSP organisé en VLIW et RISC
Avec 8 unités fonctionnelles : 6 adders/ shifters et 2 multiplieurs.

Trois membres de La famille C6x (C6000) :

Les processors DSP non conventionnels

TMS320C62x, 16-bits virgule fixe

TMS320C6211: 150 MHz (1200 RISC MIPS) pour $25 (en quantité de 25K);64 kbits de mémoire on chip (32 kbits data; 32 kbits program) plus L2 cache (512 kbits)
TMS320C6201: 167 MHz (1333 RISC MIPS) et 200 MHz (1600 RISC MIPS);
1 Mbit de mémoire on chip (512 kbits données; 512 kbits programme);
la version low- power C6201B à 200 MHz consomme 1.94 W.
TMS320C6202: 250 MHz (2000 RISC MIPS).
TMS320C6203: 250 MHz (2000 RISC MIPS) and 300 MHz (2400 RISC MIPS);
7 Mbits on-chip memory (3 Mb programme; 4 Mb données); utilisé dans les systèmes de communication numériques. En 1999, était utilisé sur des systèmes de communication troisième génération sans fil (wireless data networks) et dans les modems banks (24 modems V.90 sur single chip).

TMS320C64x, 16-bits virgule fixe

TMS320C67x, 32-bits virgule flottante avec multiplication en un cycle et 3 cycles de délai.

Le processeur C67x

TMS320C67x, 32-bits virgule flottante avec multiplication en un cycle et 3 cycles de délai.

Pin compatible avec le 'C62x

A 100-MHz, le 'C6711 délivre 600 MFLOPS à seulement $20

La version 150-MHz délivre 900 MFLOPS

La famille 'C67x offre, pour les versions proches la possibilité d'atteindre to 3 GFLOPS et plus.

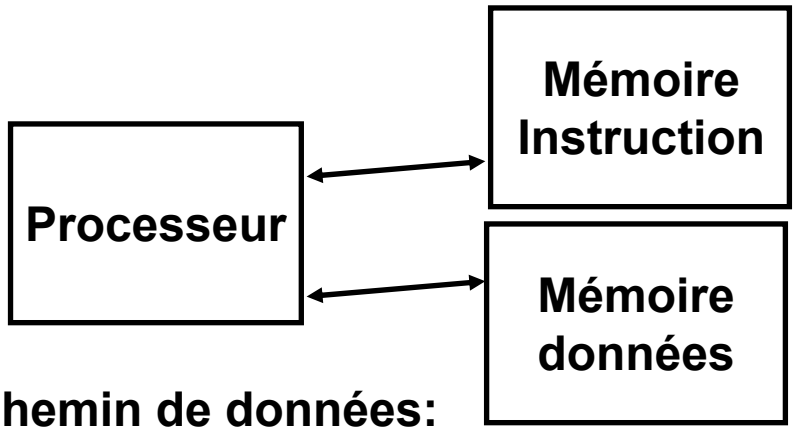
Les applications: beamforming, stations de base, réalité virtuel 3-D, graphiques, reconnaissance de la parole, radar/sonar, instrumentation de précision, et traitement d'images médicales.

Les problèmes avec les outils de TI

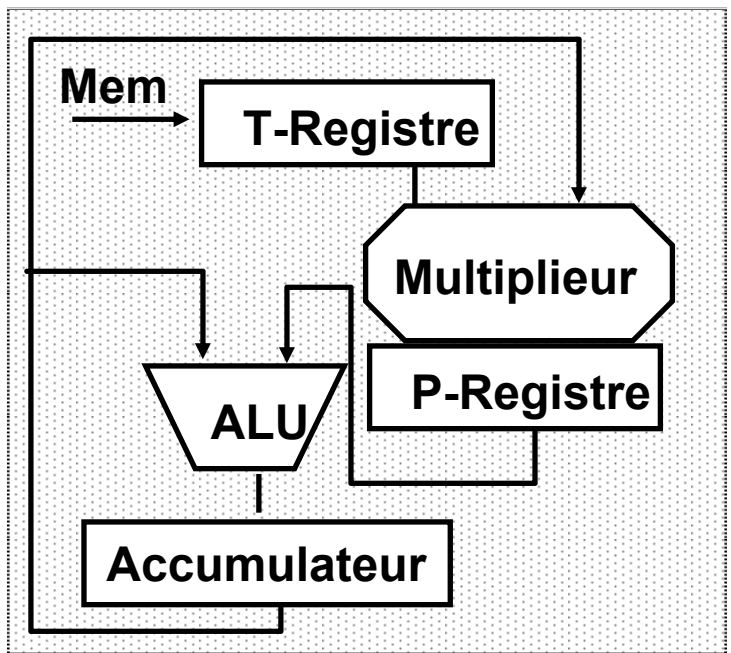
- Pas de translateurs de code entre C5x and C20x et entre C54x et C6x
- Pas de simulateurs et débuggeurs pour le publique, à l'exception du C31.
- Les compilateurs C sont très pauvres pour les processeurs DSP traditionnels virgule fixe (C2x/C5x/C54x),
- Compilateurs C relativement pauvre pour les processeurs C6000

La première génération DSP
Le TMS32010 de Texas Instruments - 1982

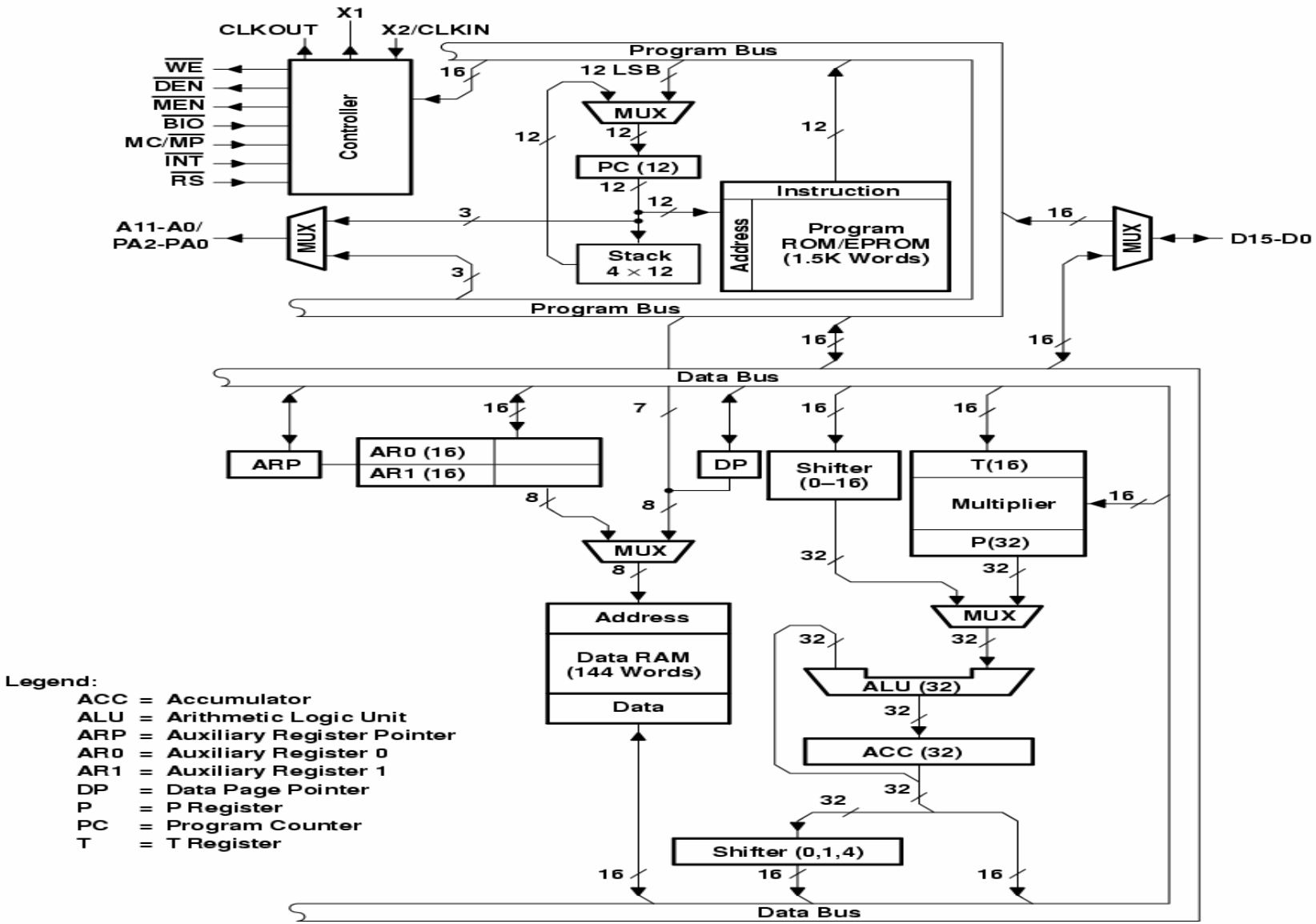
- 16-bit virgule fixe
- “Architecture Harvard”
  - mémoires séparées pour instruction et données
- Accumulateur
- Table d’ instruction spécialisé
  - Charger et Accumuler
- 390 ns temps d’Multiplication-Accumulation (MAC);



Chemin de données:



TMS32010 BLOCK DIAGRAM



Caractéristiques

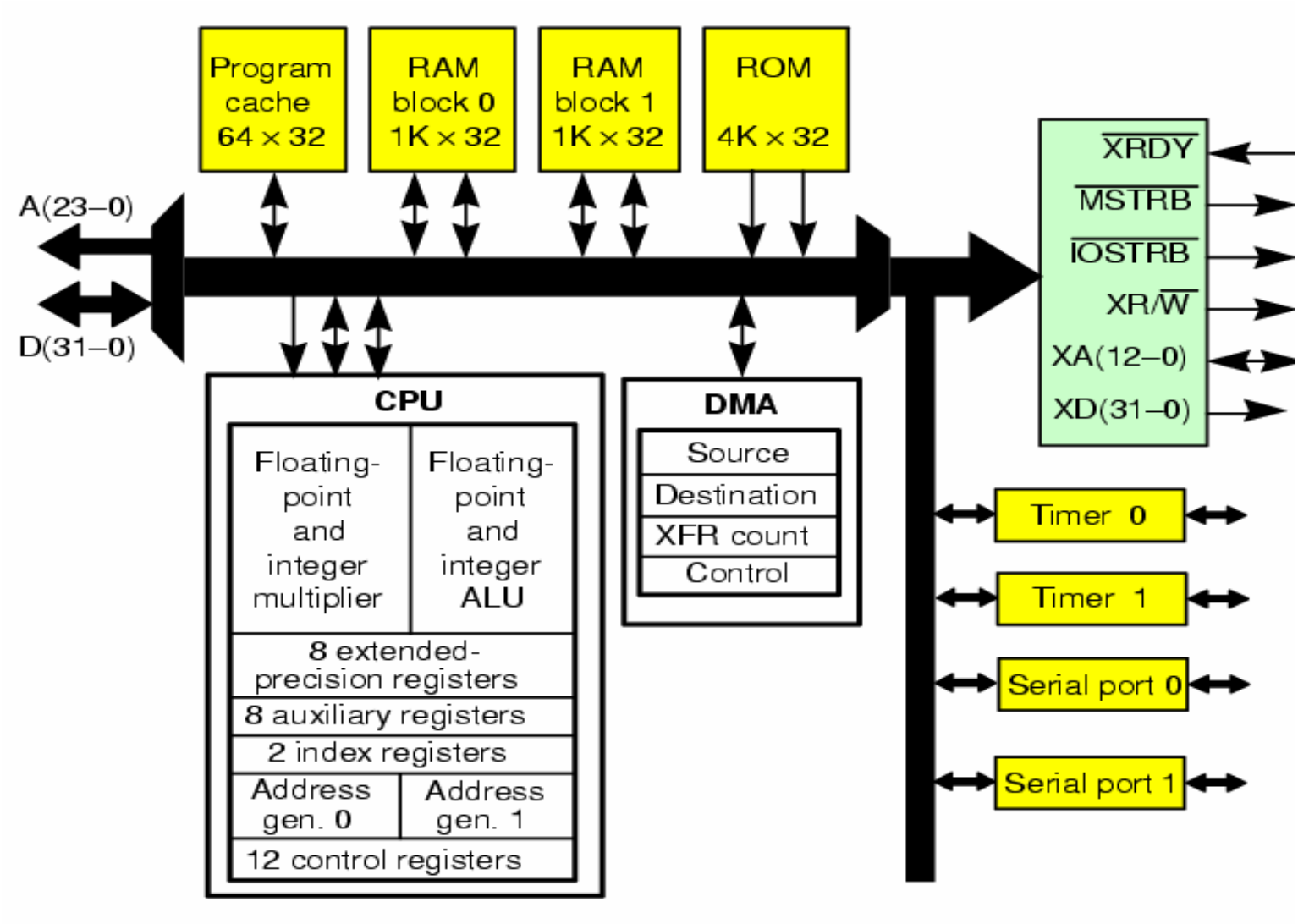
- 200 ns de cycle d'instruction (5 MIPS)
- 144 mots (16 bit) on-chip data RAM
- 1.5K mots (16 bit) on-chip program ROM
- Extensible jusqu'à 4K mots de mémoire programme à l'extérieur accessible par la même vitesse.
- Mots de 16-bit pour instruction et données
- Instruction 32-bit ALU/accumulateur en un seul cycle
- Multiplication 16 x 16-bit en un seul cycle en 200 ns
- Deux cycles MAC (5 MOPS)
- De zéro à 15-bit registre à décalage
- 8 canaux d'entrées et 8 canaux de sorties

Les DSP de troisième génération
TMS320C30 - 1988

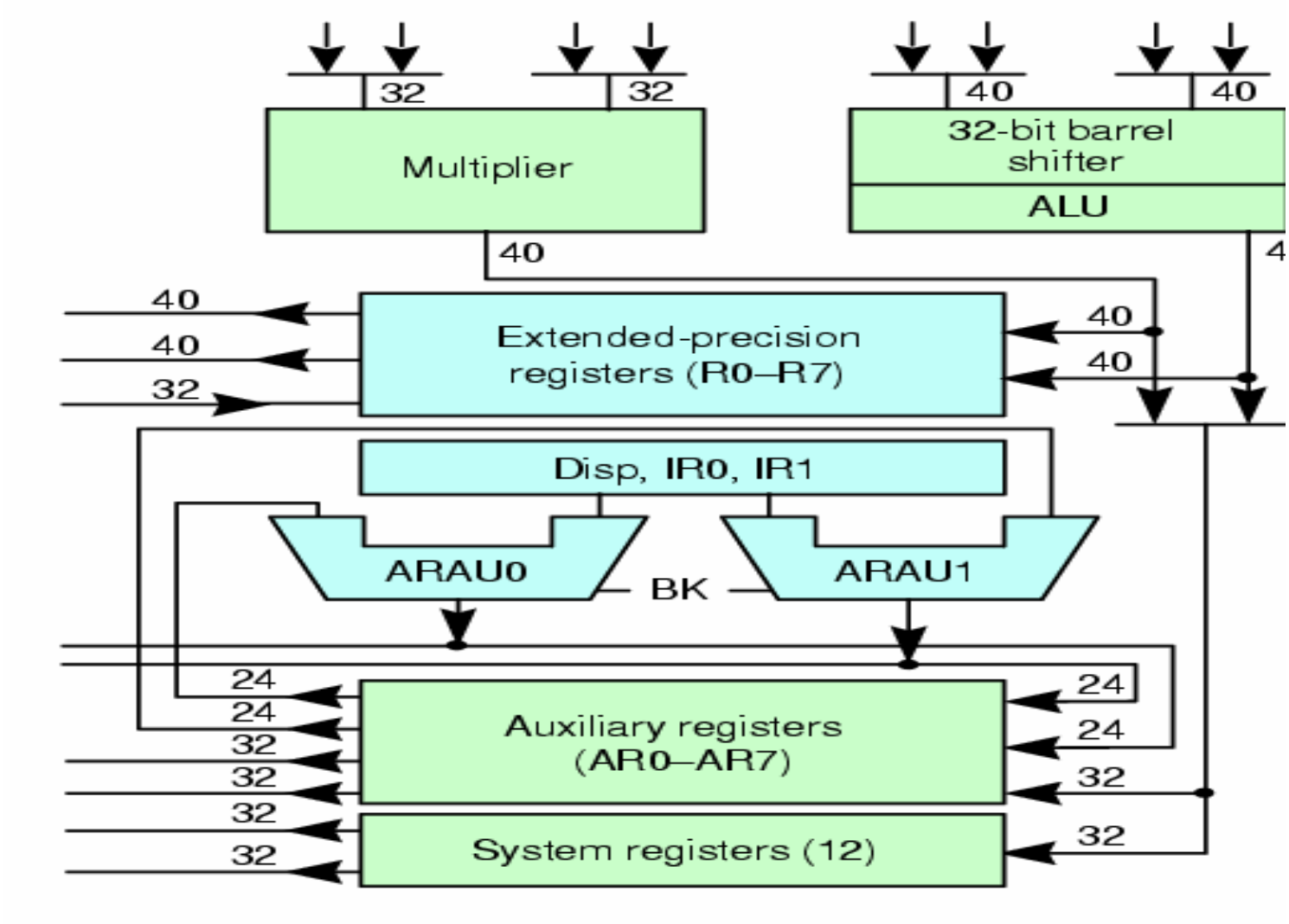
Caractéristiques du TMS320C30

- **Temps d'exécution d'instruction 60 ns en un seul-cycle**
  - **33.3 MFLOPS**
  - **16.7 MIPS**
- **Block mémoire ROM de 4K x 32-bit double-accès en un seul cycle on-chip**
- **Deux blocks RAM 1K x 32-bit double-accès en un seul cycle on-chip**
- **cache de 64 x 32-bit instruction**
- **Mots de 32-bit pour instructions et données, et 24-bit pour les adresses (16Mbytes)**
- **Multiplieur virgule-flottante/entier de 40/32-bit et ALU**
- **Registre à décalage de 32-bit**
- **8 registres à précision étendue (accumulateurs)**
- **2 générateurs d'adresses avec 8 registres auxiliaires et 2 registre auxiliaires pour l'unit arithmetique**
- **Contrôleur de Direct Memory Access (DMA) on-chip pour E/S et opération CPU parallèle**
- **Instruction ALU et multiplieur parallèle**
- **Capacité Block répétition**
- **Instructions 'Interlocked' pour support multitraitement**
- **2 ports série pour transferts 8/16/32-bit**
- **2 temporisateurs 32-bit**

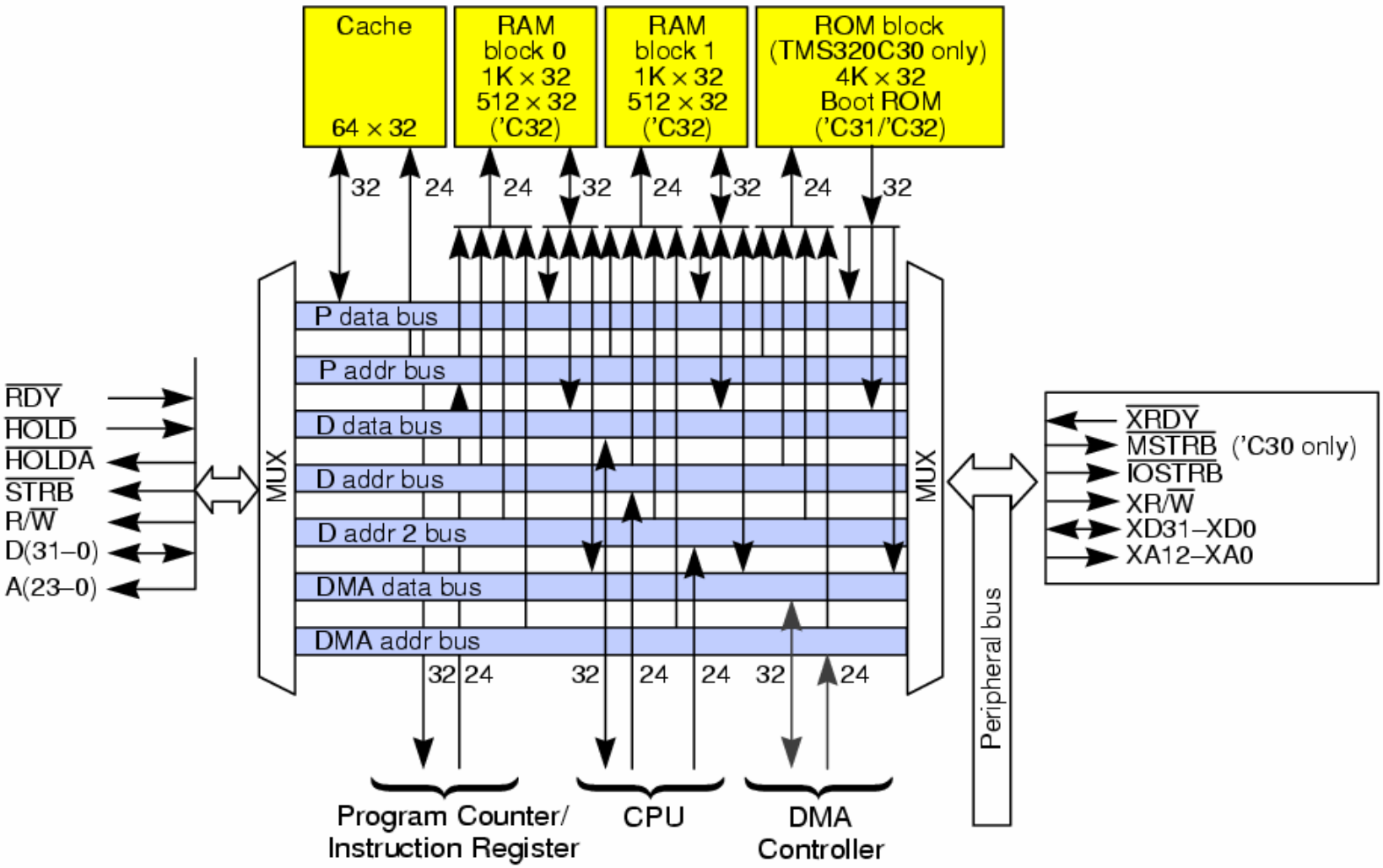
DIAGRAM PAR BLOCK TMS320C30



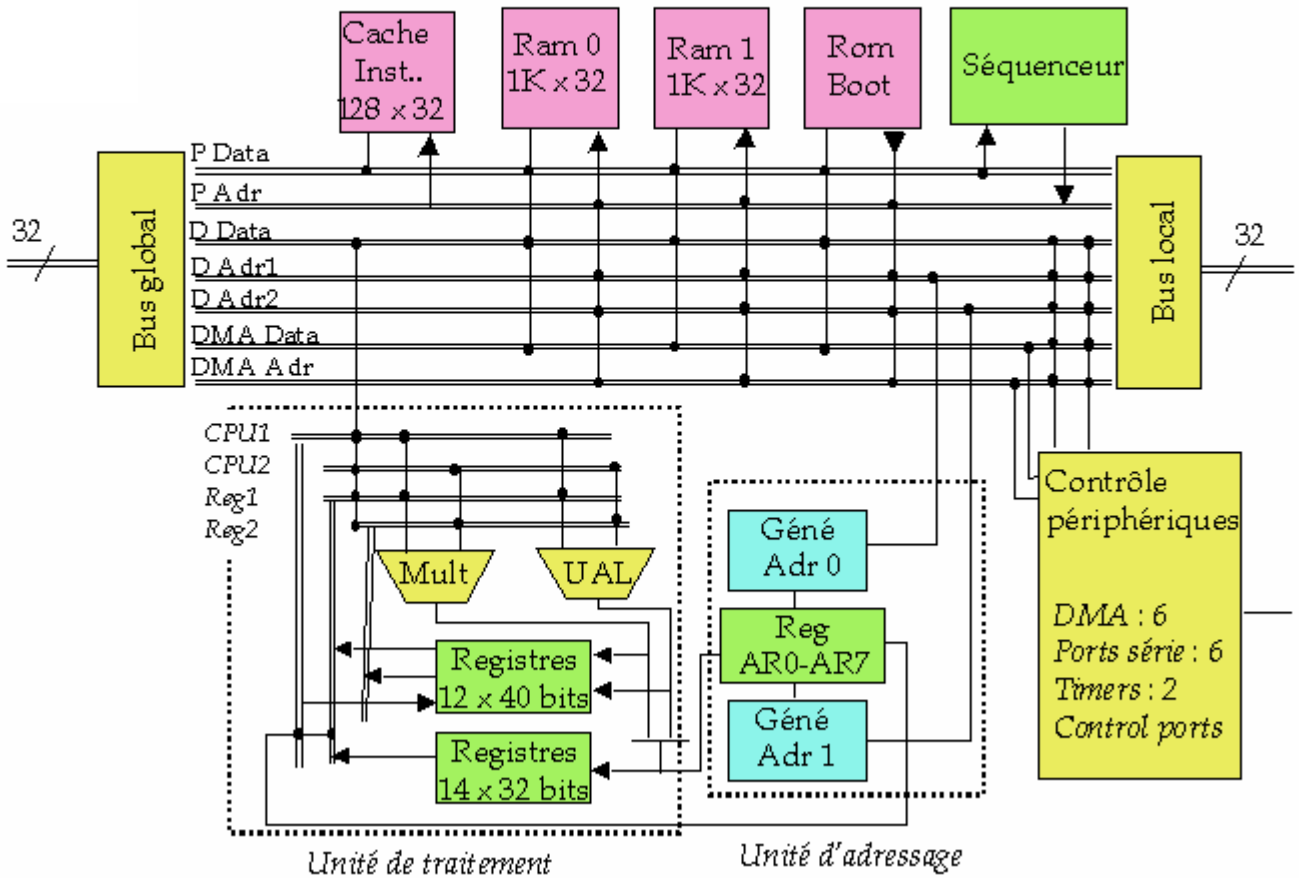
TMS320C3x CPU BLOCK DIAGRAM



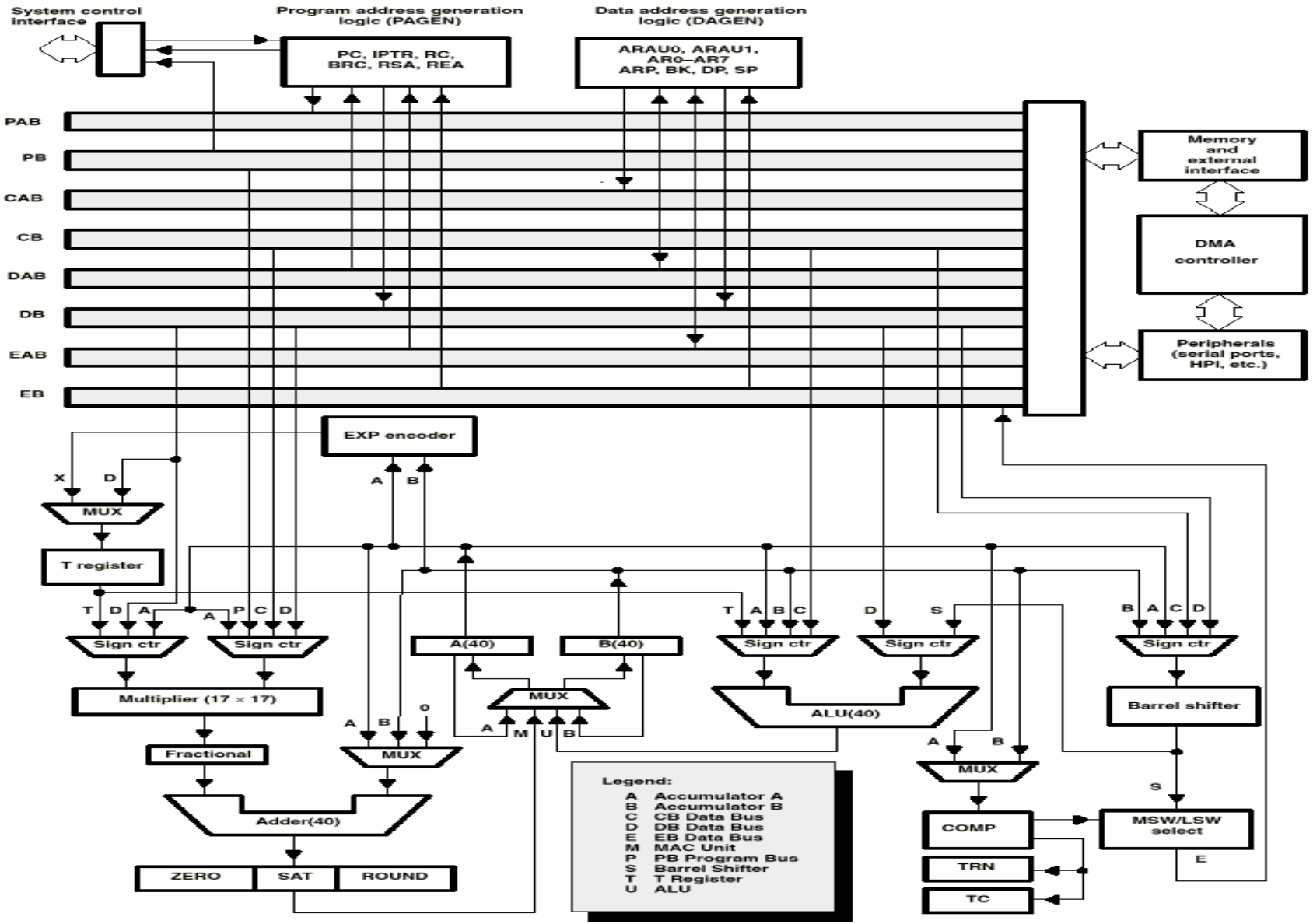
Block diagramme mémoire du TMS320C3x



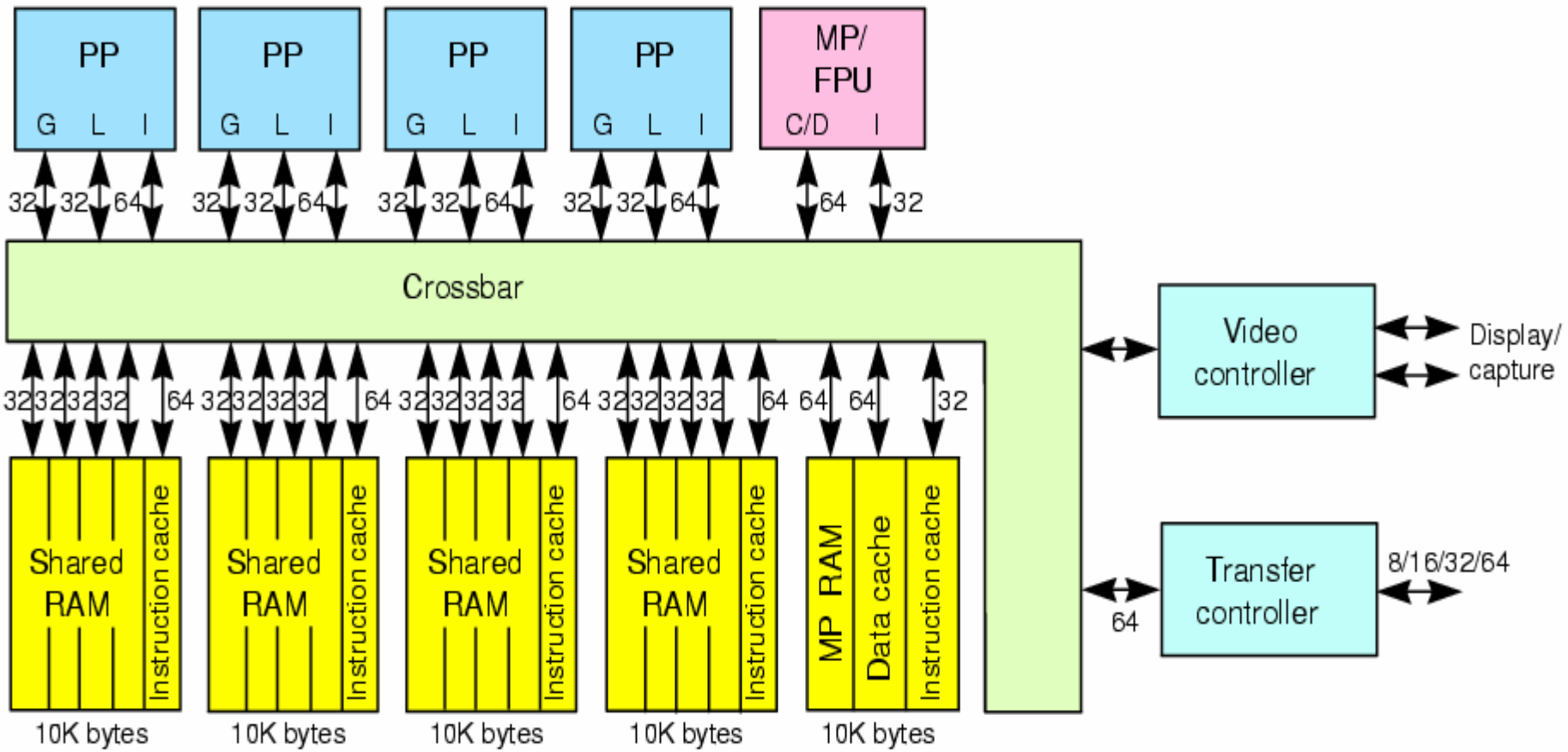
TMS320c40


Cache Inst.. 128 x 32
Ram 0 1K x 32
Ram 1 1K x 32
Rom Boot
Séquenceur
Bus global
32
P Data
P Adr
D Data
D Adr1
D Adr2
DMA Data
DMA Adr
Bus local
32
CPU1
CPU2
Reg1
Reg2
Mult
UAL
Registres 12 x 40 bits
Registres 14 x 32 bits
Géné Adr 0
Reg AR0-AR7
Géné Adr 1
Contrôle périphériques
DMA : 6
Ports série : 6
Timers : 2
Control ports
Unité de traitement
Unité d'adressage

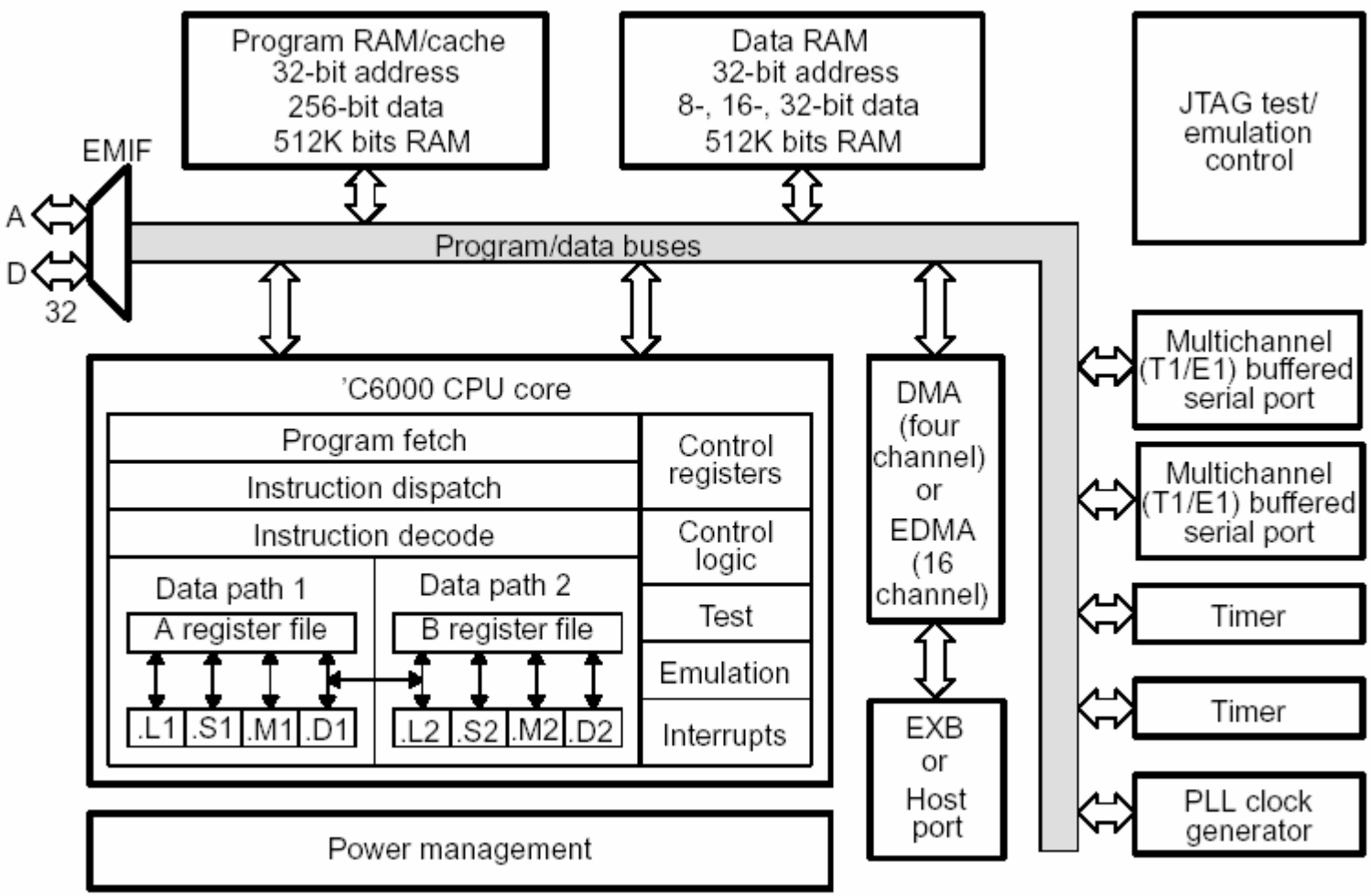
Block diagramme fonctionnel du DSP TMS320C54x



Texas Instruments TMS320C80
MIMD MULTIPROCESSOR DSP (1996)



Block Diagramme du TMS320C62x/C67x



Les unités fonctionnelles et opérations réalisées

Functional Unit	Fixed-Point Operations	Floating-Point Operations
.L unit (.L1,.L2)	32/40-bit arithmetic and compare operations Leftmost 1 or 0 bit counting for 32 bits Normalization count for 32 and 40 bits 32-bit logical operations	Arithmetic operations Conversion operations: DP → SP, INT → DP, INT → SP
.S unit (.S1, .S2)	32-bit arithmetic operations 32/40-bit shifts and 32-bit bit-field operations 32-bit logical operations Branches Constant generation Register transfers to/from the control register file (.S2 only)	Compare reciprocal and reciprocal square-root operations Absolute value operations SP to DP conversion operations
.M unit (.M1, .M2)	16 × 16 bit multiply operations	32 × 32 bit multiply operations Floating-point multiply operations
.D unit (.D1, .D2)	32-bit add, subtract, linear and circular address calculation Loads and stores with a 5-bit constant offset Loads and stores with a 15-bit constant offset (.D2 only)	Load double word with a 5-bit constant offset

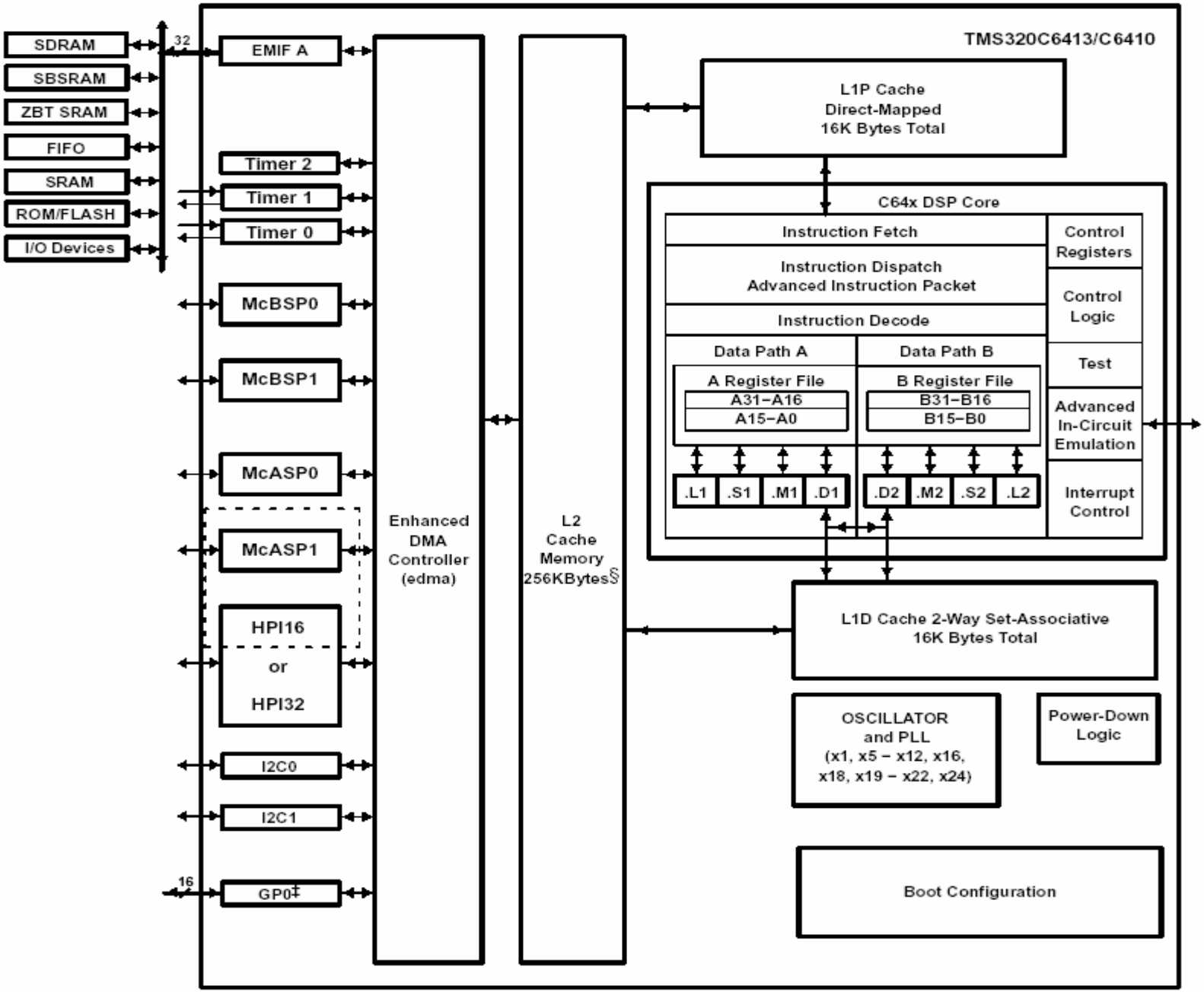
Instruction virgule fixe des unités fonctionnelles

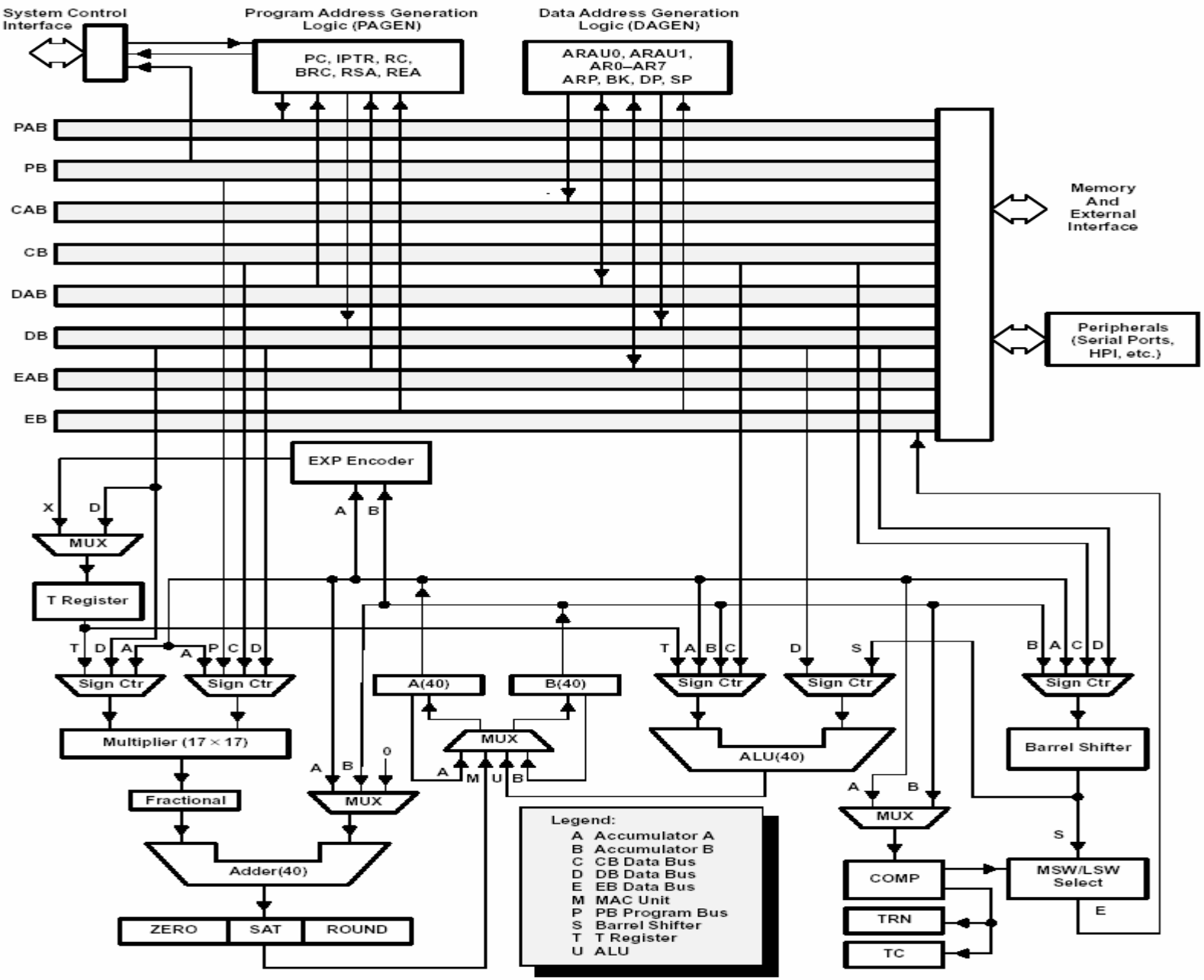
.L Unit	.M Unit	.S Unit		.D Unit	
ABS	MPY	ADD	SET	ADD	STB (15-bit offset)‡
ADD	MPYU	ADDK	SHL	ADDAB	STH (15-bit offset)‡
ADDU	MPYUS	ADD2	SHR	ADDAH	STW (15-bit offset)‡
AND	MPYSU	AND	SHRU	ADDAW	SUB
CMPEQ	MPYH	B disp	SHRL	LDB	SUBAB
CMPGT	MPYHU	B IRP†	SUB	LDBU	SUBAH
CMPGTU	MPYHUS	B NRP†	SUBU	LDH	SUBAW
CMPLT	MPYHSU	B reg	SUB2	LDHU	ZERO
CMPLTU	MPYHL	CLR	XOR	LDW	
LMBD	MPYHLU	EXT	ZERO	LDB (15-bit offset)‡	
MV	MPYHULS	EXTU		LDBU (15-bit offset)‡	
NEG	MPYHSLU	MV		LDH (15-bit offset)‡	
NORM	MPYLH	MVC†		LDHU (15-bit offset)‡	
NOT	MPYLHU	MVK		LDW (15-bit offset)‡	
OR	MPYLUHS	MVKH		MV	
SADD	MPYLSHU	MVKLH		STB	
SAT	SMPY	NEG		STH	
SSUB	SMPYHL	NOT		STW	
SUB	SMPYLH	OR			
SUBU	SMPYH				
SUBC					
XOR					
ZERO					

† S2 only
‡ D2 only

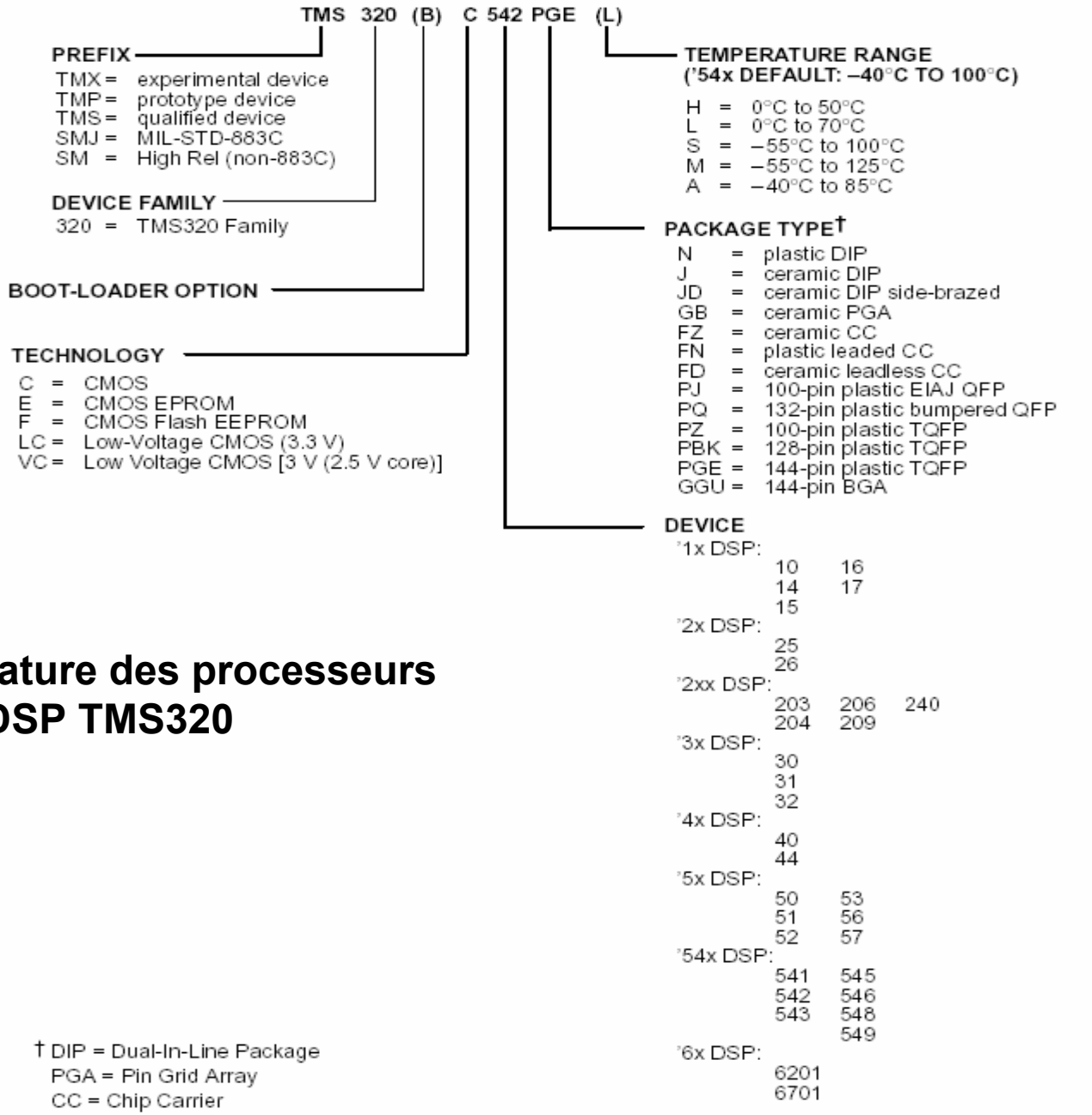
Les instructions virgule flottante des unités fonctionnelles

.L Unit	.M Unit	.S Unit	.D Unit
ADDDP	MPYDP	ABSDP	ADDAD
ADDSP	MPYI	ABSSP	LDDW
DPINT	MPYID	CMPEQDP	
DPSP	MPYSP	CMPEQSP	
INTDP		CMPGTDP	
INTDPU		CMPGTSP	
INTSP		CMPLTDP	
INTSPU		CMPLTSP	
SPINT		RCPDP	
SPTRUNC		RCPSP	
SUBDP		RSQRDP	
SUBSP		RSQRSP	
		SPDP	


TMS320C6413/C6410
SDRAM
SBSRAM
ZBT SRAM
FIFO
SRAM
ROM/FLASH
I/O Devices
32
EMIF A
Timer 2
Timer 1
Timer 0
McBSP0
McBSP1
McASP0
McASP1
HPI16
or
HPI32
I2C0
I2C1
16
GP0‡
Enhanced DMA Controller (edma)
L2 Cache Memory 256KBytes§
L1P Cache Direct-Mapped 16K Bytes Total
C64x DSP Core
Instruction Fetch
Instruction Dispatch
Advanced Instruction Packet
Instruction Decode
Data Path A
Data Path B
A Register File
A31–A16
A15–A0
B Register File
B31–B16
B15–B0
.L1
.S1
.M1
.D1
.D2
.M2
.S2
.L2
Control Registers
Control Logic
Test
Advanced In-Circuit Emulation
Interrupt Control
L1D Cache 2-Way Set-Associative 16K Bytes Total
OSCILLATOR and PLL (x1, x5 – x12, x16, x18, x19 – x22, x24)
Power-Down Logic
Boot Configuration


System Control Interface
Program Address Generation Logic (PAGEN)
Data Address Generation Logic (DAGEN)
PC, IPTR, RC, BRC, RSA, REA
ARAU0, ARAU1, AR0–AR7 ARP, BK, DP, SP
PAB
PB
CAB
CB
DAB
DB
EAB
EB
Memory And External Interface
Peripherals (Serial Ports, HPI, etc.)
EXP Encoder
X
D
MUX
T Register
T D A
A P C D
Sign Ctr
Sign Ctr
Multiplier (17 × 17)
Fractional
A B 0
MUX
Adder(40)
ZERO
SAT
ROUND
A(40)
B(40)
MUX
A M U B
T A B C
D S
Sign Ctr
Sign Ctr
ALU(40)
B A C D
Sign Ctr
Barrel Shifter
A B
MUX
COMP
TRN
TC
S
MSW/LSW Select
E
Legend:
A Accumulator A
B Accumulator B
C CB Data Bus
D DB Data Bus
E EB Data Bus
M MAC Unit
P PB Program Bus
S Barrel Shifter
T T Register
U ALU

La nomenclature des processeurs DSP TMS320



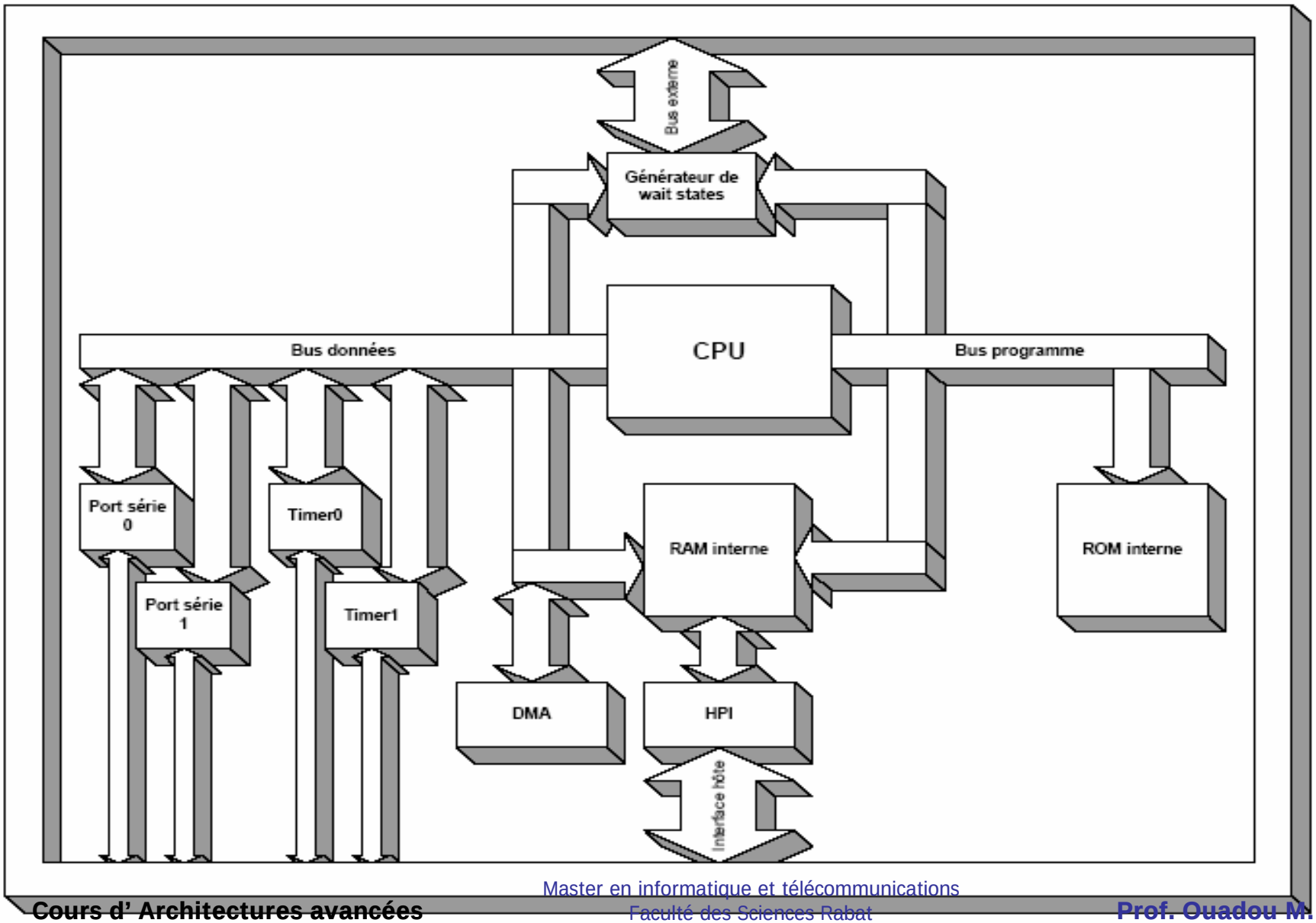
† DIP = Dual-In-Line Package
PGA = Pin Grid Array
CC = Chip Carrier
QFP = Quad Flat Package
TQFP = Thin Quad Flat Package

La famille TMS320C54x : Description générale

	C541	C542	C5402	C543	C545	C546	C548	C549	C5410	C5420
ROM:	28 K	2 K	4 K	2 K	48 K	48 K	2 K	16 K	16 K	0
•**Program**	20 K	2 K	4 K	2 K	32 K	32 K	2 K	16 K	16 K	0
•**Program/Data**	8 K	0	4 K	0	16 K	16 K	0	16	0	0
DARAM	5 K	10 K	16 K	10 K	6 K	8 K	8 K	8 K	8 K	32 K
SARAM	0	0	0	0	0	0	24 K	24 K	56 K	168 K

- **Tous les DSP de la famille C54x ont le même CPU interne**
- **Diffèrent par la taille de la mémoire interne et par les périphériques intégrés**

Architecture du TMS320C541



Architecture des BUS

Architecture de Havard modifiée :

- 1 bus (mémoire programme)
- 3 bus (mémoire données)
- 4 bus d'adresse

Programme et Données : 16 bits

Adressage : 16 bits

64 Kmots mémoire programme

64 Kmots mémoire données

64 Kmots entrées/sorties

Architecture des BUS

• **BUS Programme**

PB : véhicule les instructions et coefficients stockés en mémoire programme
PAB : bus d'adresse de la mémoire programme

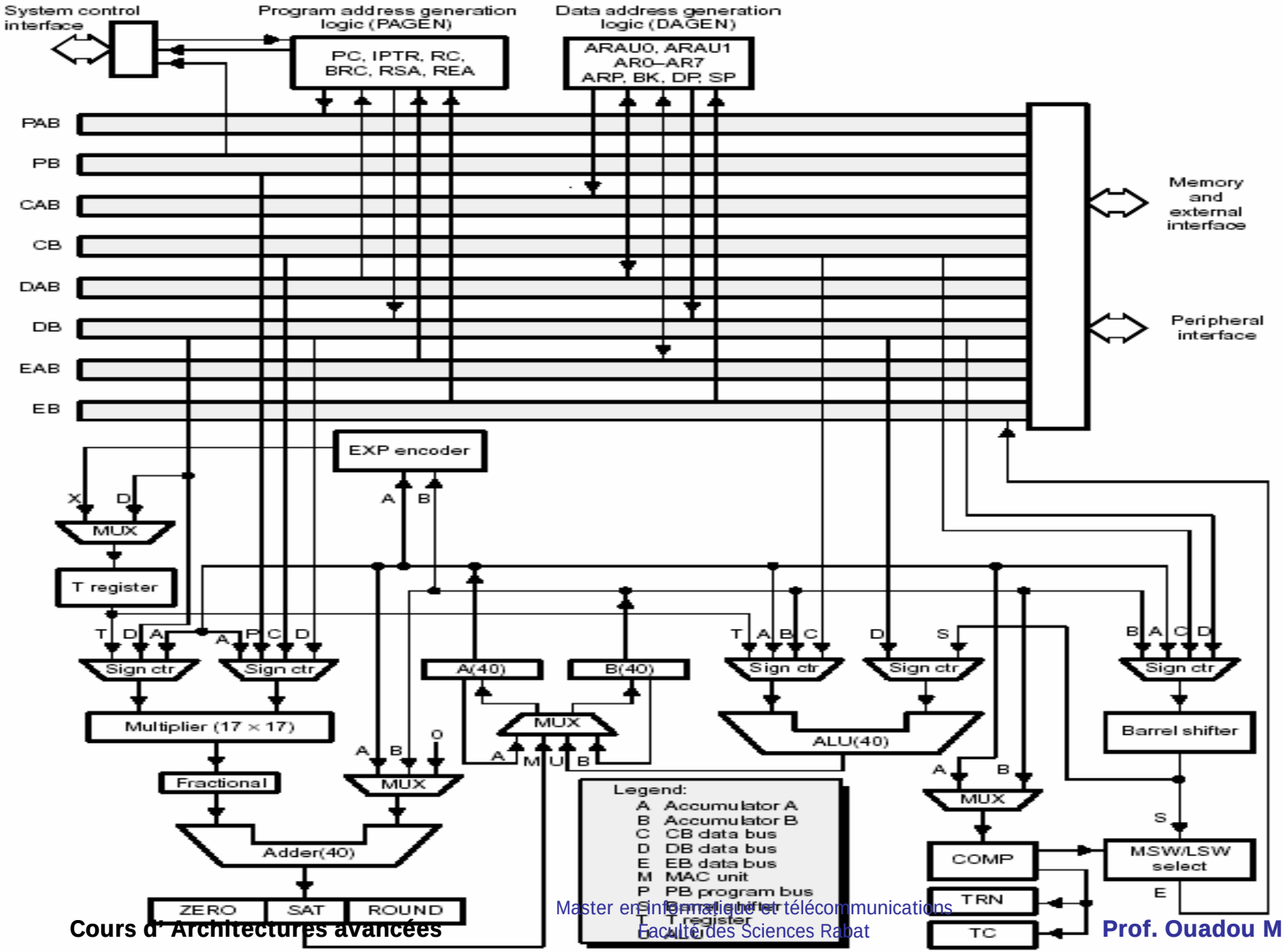
• **BUS Données**

CB, DB et EB : bus d'accès aux données
CAB, DAB et EAB : bus d'adresse associés

Bus sur 16 bits (adresses et données)
2 unités de calcul pour l'adressage par registre auxiliaire

☞ **Adressage possible de 2 données pendant un cycle**

L'unité centrale de calcul (CPU)



L’unité centrale de calcul (CPU)

- 1 ALU sur 40 bits
- 2 ACCU de 40 bits (A et B)
- 1 registre à décalage de 16 bits (T)
- 1 multiplieur 17x17 bits et 1 additionneur sur 40 bits
- 1 unité de comparaison, sélection et stockage (CSSU)
- 1 unité de génération d’adresse de données
- 1 unité de génération d’adresse de programme.

Les accumulateurs A et B

- Destination pour l'ALU et pour le multiplieur/additionneur.

- 3 champs :
  - Un ACCU bas (AL ou BL) contenant les 16 bits de poids faible
  - Un ACCU haut (AH ou BH) contenant les 16 bits de poids fort
  - Un champ de 8 bits de garde (AG ou BG) : éviter les saturations

- A et B sont adaptés au traitement d'instructions particulières (filtrage, algorithme de Viterbi).

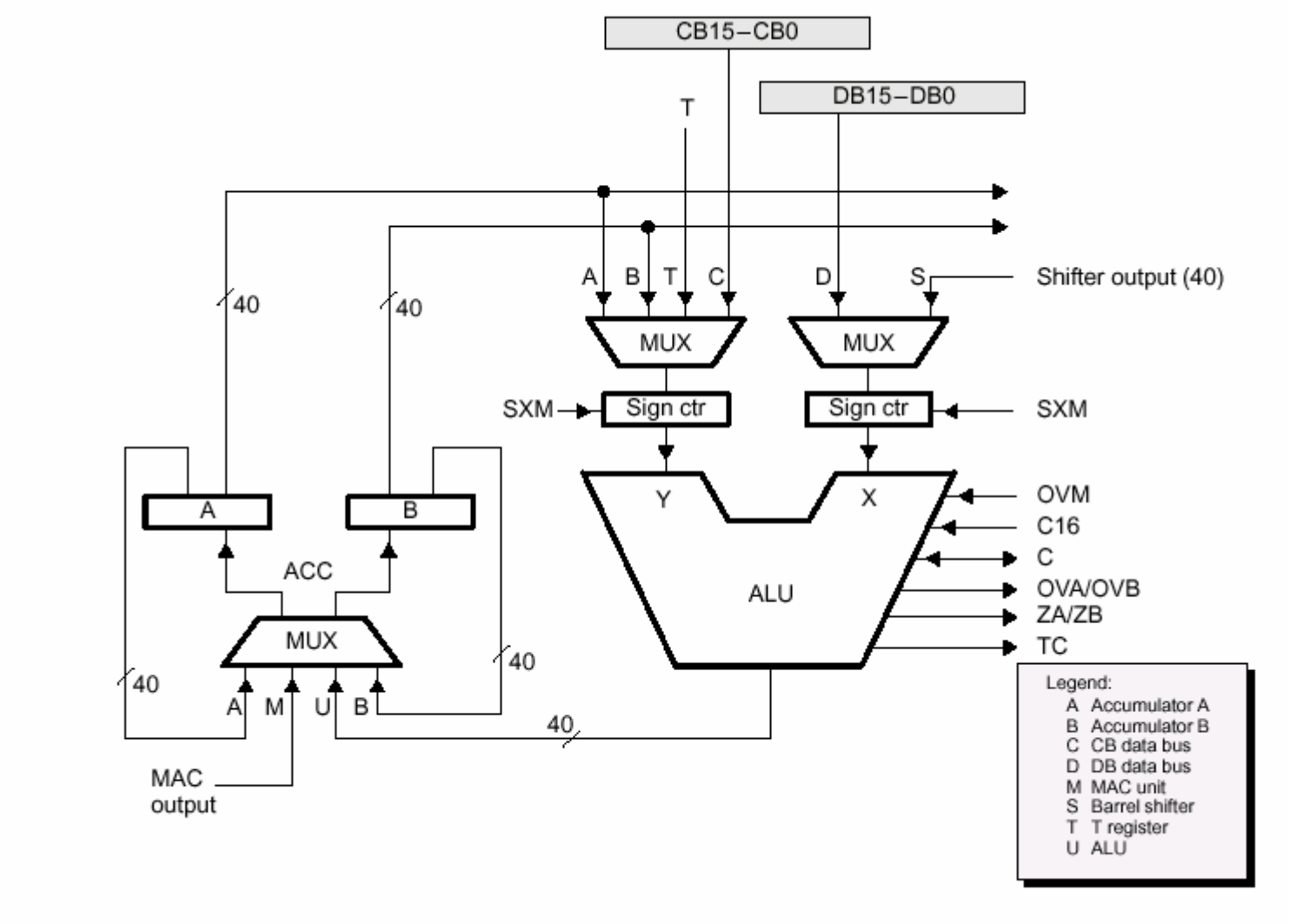
Registre A

AG	AH	AL
39-32	31-16	15-0

Registre B

BG	BH	BL
39-32	31-16	15-0

Unité Arithmétique et Logique (ALU)



Unité Arithmétique et Logique (ALU)

- Calculs arithmétiques et logiques sur 40 bits
- Deux accumulateurs (A et B)
- Opérations effectuées généralement en un seul cycle et stockées dans un des deux accumulateurs
- L'ALU travaille soit sur des mots de 32 bits (C16 = 0), soit avec 2 mots de 16 bits (C16 = 1)
- Une entrée de l'ALU peut provenir de l'ACCU A ou B

Unité Arithmétique et Logique (ALU)
Table d'instructions

Addition et Soustraction:

Multiplication

Multiplication Fractionnaire

Le bit FRCT (Fractional mode) du registre de contrôle ST1 à 1

Rotation

Décalage Logique

Décalage Arithmétique

Opérations Logiques

Unité Arithmétique et Logique (ALU)
Table d'instructions

Extension de signe automatique

ajuster la valeur du bit SXM (Signe Extension Mode) du registre de contrôle ST1

Arrondi Automatique

Saturation Automatique

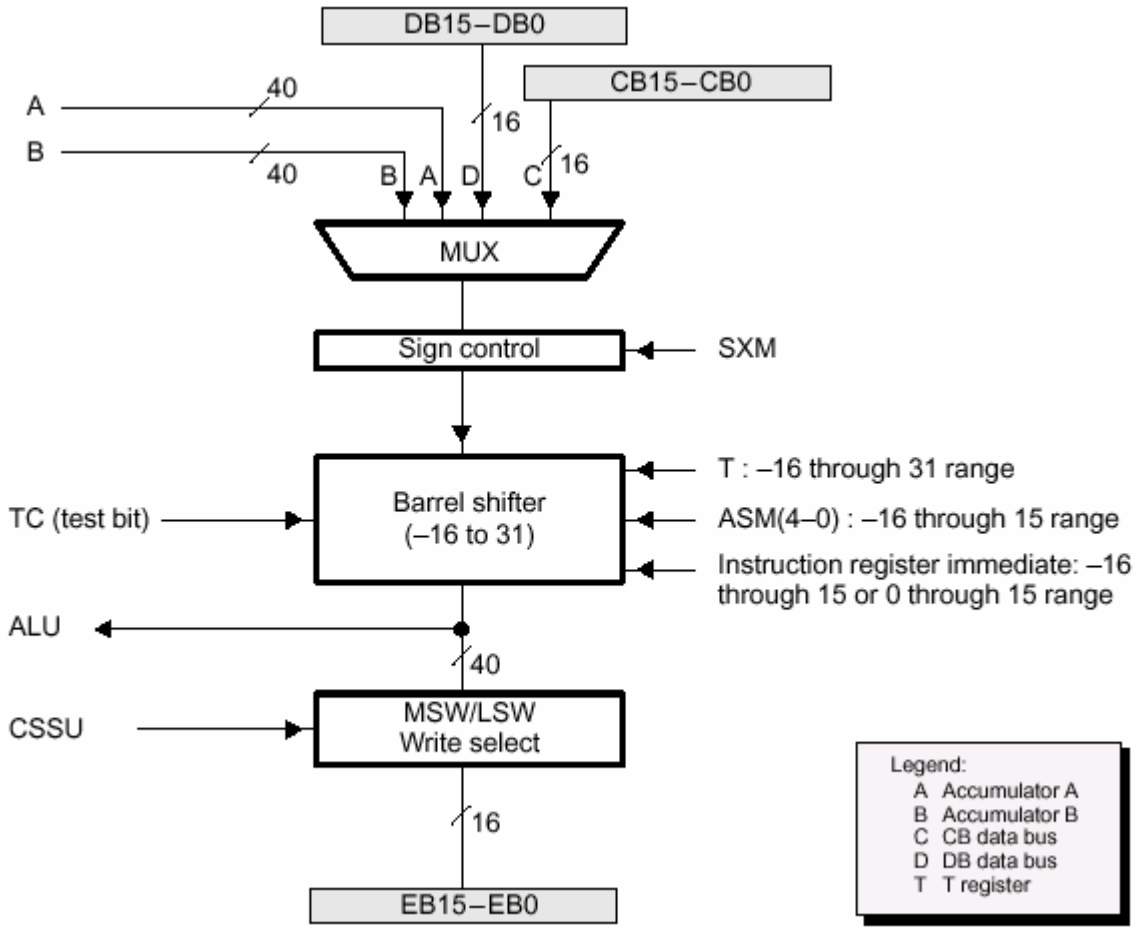
Pour cela, le bit OVM (Overflow Mode) du registre de contrôle ST1 doit être placé à 1

Exemple :

Le nombre 0900000000H sera saturé à 007FFFFFFFH (+2 147 483 647d).

Le nombre FE00000000H sera saturé à FF80000000H (-2 147 483 648d).

Registre à décalage



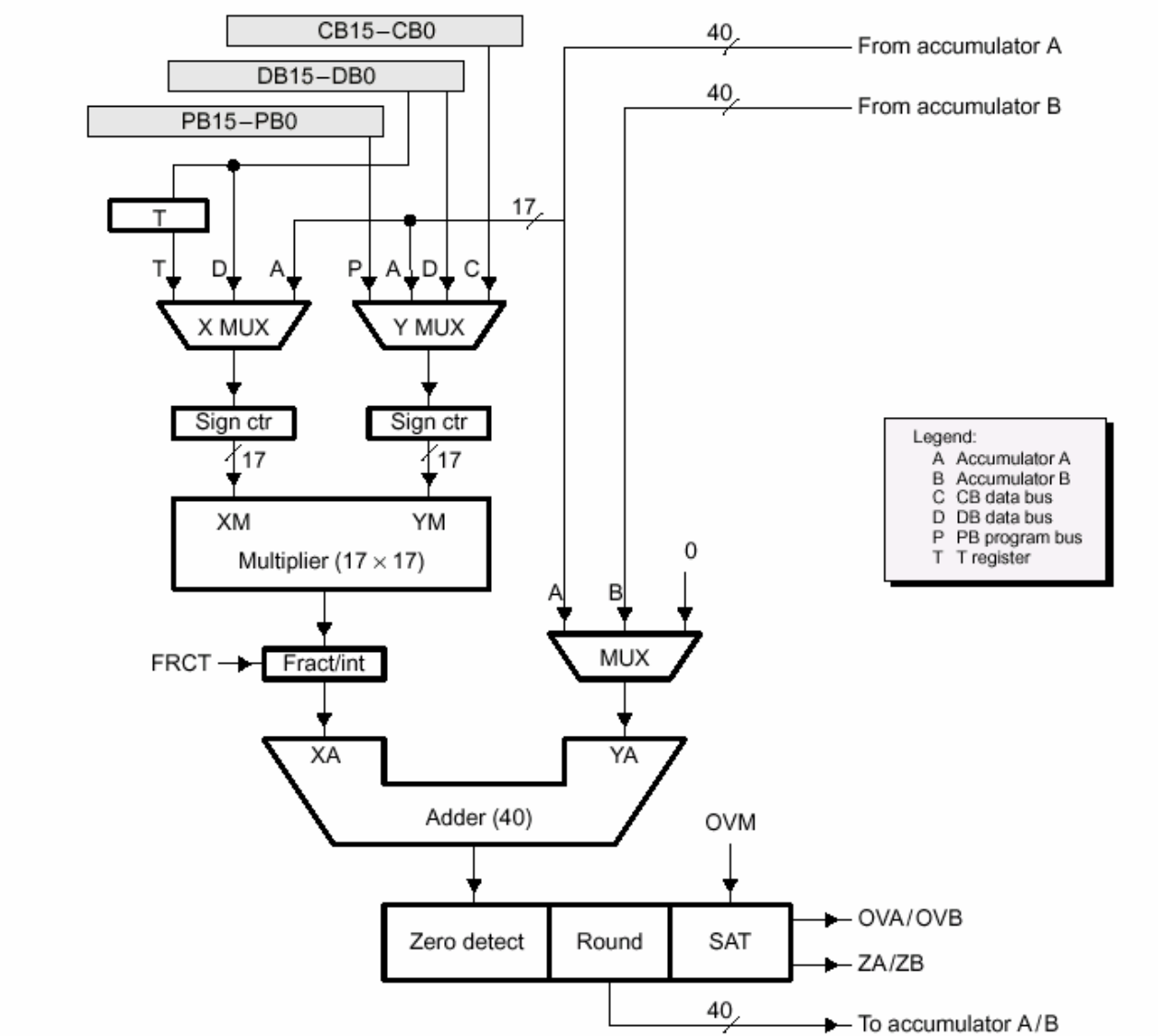
Registre à décalage

• Registre de 40 bits servant à **cadrer les données** en provenance de la mémoire **ou la valeur de l'ACCU** avant une opération dans l'ALU.

• Décalage à droite ou à gauche.

• Valeur du décalage spécifiée par :
  • champ ASM du registre d'état ST1
  • Registre T
  • Valeur immédiate

Unité de Multiplieur/Additionneur



Le Multiplieur/Additionneur

- Non couplé à l'ALU
- Possède un additionneur de 40 bits dédié

☞ **Exécution d'une instruction MAC en un seul cycle**

- Utilise des mots de 17 bits :

16 bits + 1 bit de signe si multiplication signée
16 bits + RAZ du bit de poids fort si multiplication non signée

Registres d'état et de contrôle

Registre ST0

15 - 13	12	11	10	9	8 - 0
ARP	TC	C	OVA	OVB	DP

Registre ST1

15	14	13	12	11	10	9	8	7	6	5	4 - 0
BRAF	CPL	XF	HM	INTM	0	OVM	SXM	C16	FRCT	CMPT	ASM

Registre PMST

15 – 7	6	5	4	3	2	1	0
IPTR	MP/MC	OVLY	AVIS	DROM	CLKOFF	SMUL	SST

Processor Mode Status Register

Registre ST0

Bit 15-13	ARP	Spécifie le Registre Auxiliaire par défaut en mode comparatif (par défaut : AR0)
Bit 12	TC	Sert à stocker le résultat d'une opération logique
Bit 11	C	Bit de la retenue
Bit 10	OVA	Bit de saturation (Overflow) pour le registre A
Bit 9	OVB	Bit de saturation (Overflow) pour le registre B
Bit 8-0	DB	Pointeur de la page de la mémoire des données

Registre ST1

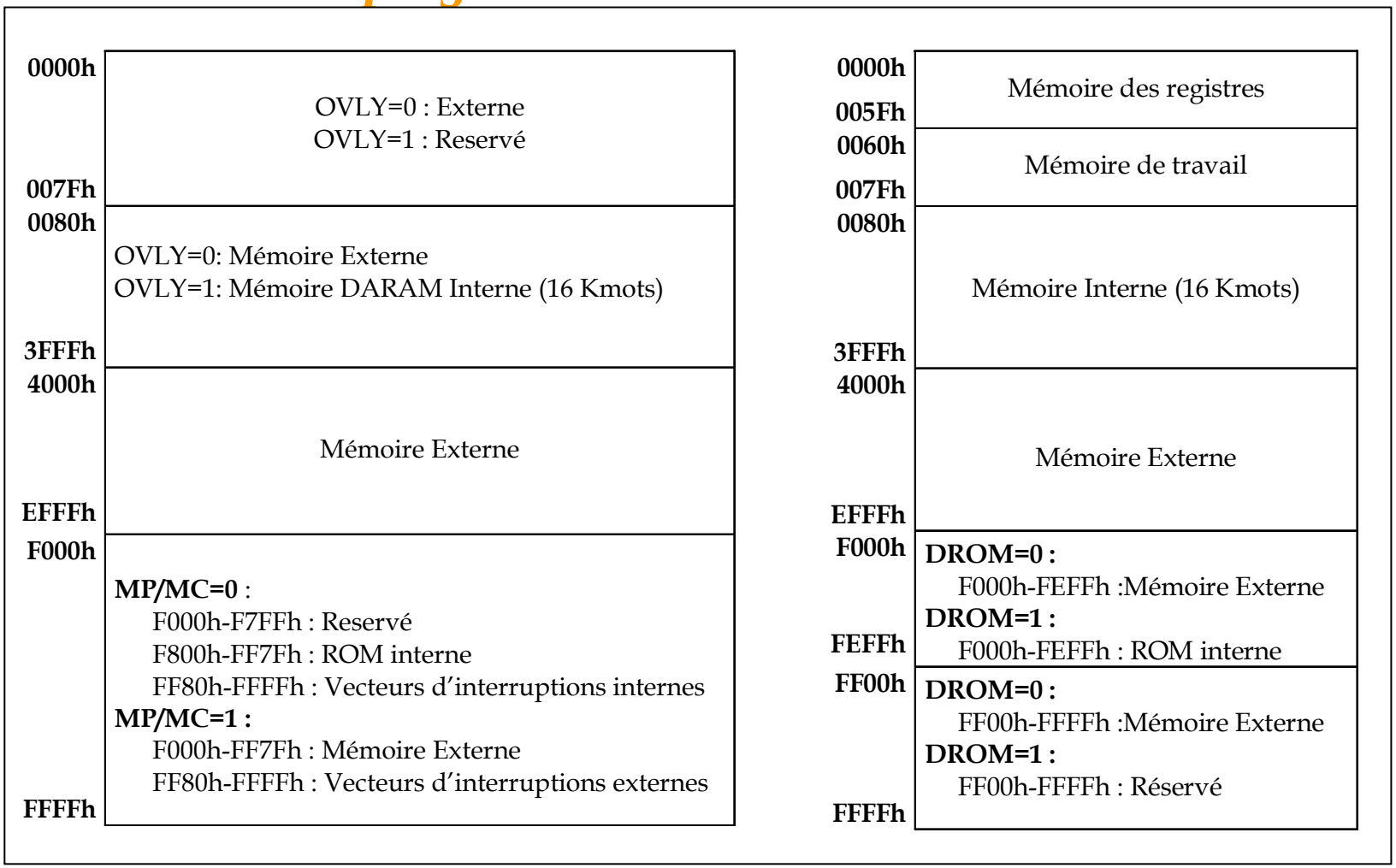
Bit 15	BRAF	Indique si l'exécution d'une boucle est en cours lors d'une répétition
Bit 14	CPL	Indique le registre pointeur de la page des données DP ou SP (par défaut : DP)
Bit 13	XF	Change l'état de la broche XF du DSP qui peut être visualiser sur un oscilloscope
Bit 12	HM	HM Type de fonctionnement du DSP en mode HOLD
Bit 11	INTM	Bit d'autorisation des interruptions non masquées (à 0 : autorisation ; à 1 : masquage)
Bit 10	0	Lu toujours 0
Bit 9	OVM	Bit pour l'activation mode de saturation
Bit 8	SXM	Bit d'activation du mode d'extension du signe
Bit 7	C16	C16=1: l 'ALU fonctionne en mode double précision sur 40 bits C16=0 : l' ALU travaille en mode dual sur deux mots de 16 bits
Bit 6	FRCT	Bit de Validation du mode fractionnaire
Bit 5	CMPT	Définit le mode de comparaison pour le registre spécifié par ARP
Bit 4-0	ASM	Spécifie le nombre de décalage (-16 à 15) de l'accumulateur

Registre PMST

Bit 15-7	IPTR	Sert de pointeur pour les vecteurs d'interruptions et définit la page de 128 mots dans laquelle est déclarée la table des vecteurs d'interruptions (par défaut : IPTR=1FFh et la table des vecteurs d'interruptions commence à l'adresse 1FFh*80h=FF80h)
Bit 6	MP/MC	Valide la ROM interne du DSP lorsque ce bit est à 1
Bit 5	OVLY	OVLY = 1 : Autorise le mappage de la mémoire RAM à double accès (DARAM) à la fois en espace programme et en escape données OVLY = 0 : utilisation d'une mémoire externe (voir organisation de la mémoire)
Bit 4	AVIS	Autorise l'apparition sur les broches du bus d'adresse du DSP des signaux d'adresse du programme interne
Bit 3	DROM	DROM=0 : la mémoire F000h-FFFFh (C542) dépend de la broche MP/MC DROM=1 : la mémoire F000h-FFFFh (C542) sera la ROM interne et ne dépend pas de la broche MP/MC
Bit 2	CLKOFF	CLKOFF = 0 : l'horloge sort sur la broche CLKOUT CLKOFF = 1 : désactivation de la sortie de l'horloge (haut impédance de CLKOUT)
Bit 1	SMUL	Autorise ou non la saturation lors d'une multiplication
Bit 0	SST	Autorise ou non la saturation des sauvées en provenance de l'accumulateur

Organisation mémoire

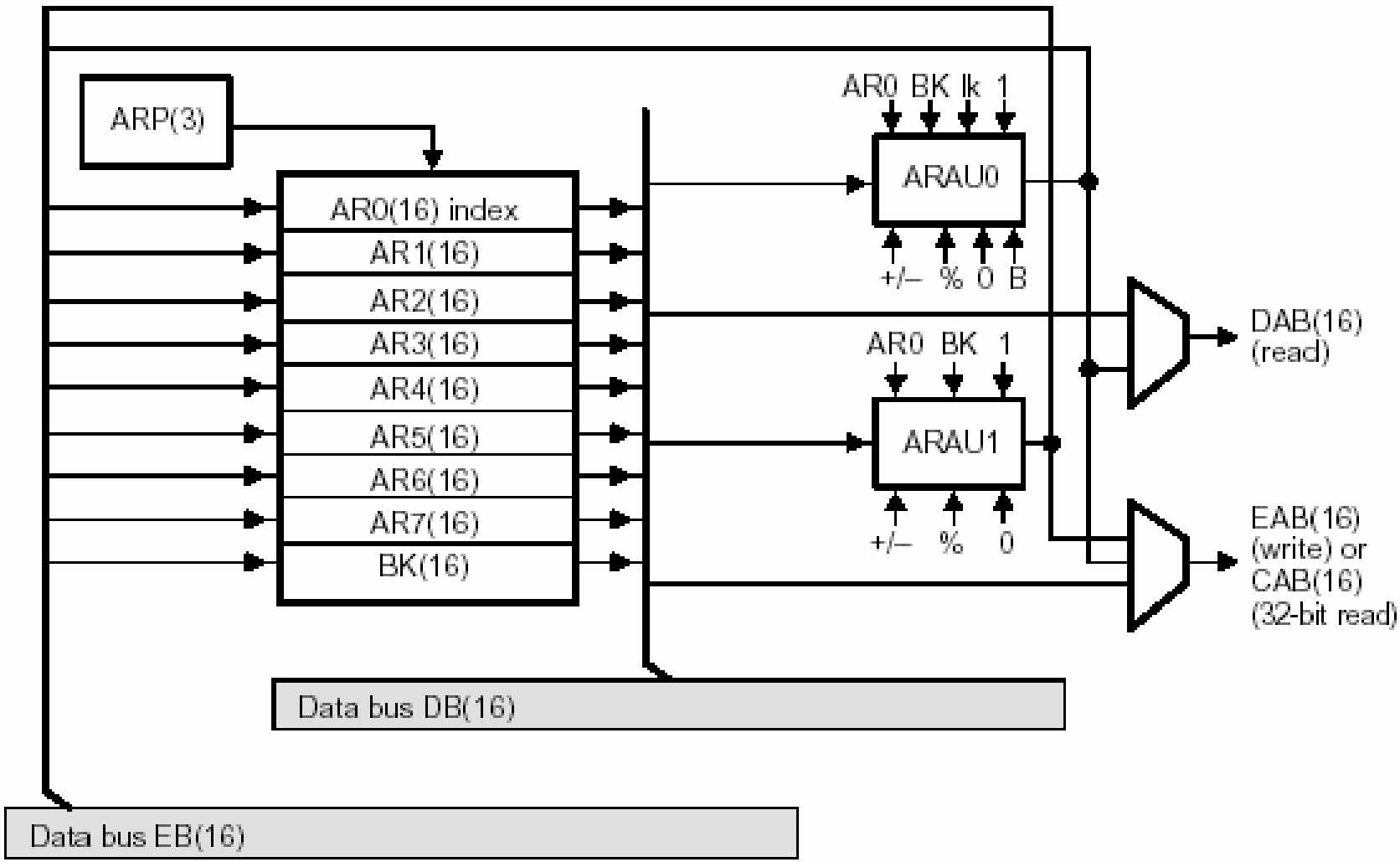
Elle diffère selon la version du circuit; ex : TMS320C541



Registres du CPU mappés en Page 0

Adresse en hexadécimal	Registre	Description
0	IMR	Registre de masque des interruptions
1	IFR	Registre d'état des interruptions
2-5	-	Réservé au test
6	ST0	Registre d'état 0
7	ST1	Registre d'état 1
8	AL	A(15-0) accumulateur Bas
9	AH	A(31-16) accumulateur Haut
A	AG	A(39-32) accumulateur de Garde
B	BL	B(15-0) accumulateur Bas
C	BH	B(31-16) accumulateur Haut
D	BG	B(39-32) accumulateur de Garde
E	T	Registre Tampon
F	TRN	Registre de Transition
10	AR0	Registre Auxiliaire 0
11	AR1	Registre Auxiliaire 1
12	AR2	Registre Auxiliaire 2
13	AR3	Registre Auxiliaire 3
14	AR4	Registre Auxiliaire 4
15	AR5	Registre Auxiliaire 5
16	AR6	Registre Auxiliaire 6
17	AR7	Registre Auxiliaire 7
18	SP	Pointeur de Pile
19	BK	Registre de Buffeur Circulaire
1A	BRC	Compteur de Répétition de Bloc
1B	RSA	Adresse de Début de Bloc
1C	REA	Adresse de Fin de Bloc
1D	PMST	Registre d'Etat des Modes du Processeur
1E	XPC	Registre d'extension de la mémoire programme (C5402, C548, C549, C5410, C5420)
1E-1F	-	Réservé

Registres Auxiliaires: AR0 – AR7



Modes d'Adressage

Adressage immédiat

Adressage absolu:

Adressage Direct :

Adressage Indirect

•Adressage par l'Accumulateur :

•Adressage Circulaire :

Adressage Bit-reverse

Interruptions

- Interruptions HARDWARE :
  - Quatre d'usage général INT0 à INT3 permettant un fonctionnement asynchrone du DSP,
  - Une broche RESET pour le chargement du programme source,
  - Une broche d'interruption non masquable NMI (NonMaskable Interrupt).

- Interruptions SOFTWARE :
  - Le TIMER génère l'interruption TINT
  - Le port série BSP0 génère les interruptions BRINT0 et BXINT0
  - Le port série BSP1 génère les interruptions BRINT1 et BXINT1
  - Le port parallèle HPI (Host Port Interface) génère l'interruption HPINT qui doit rester toujours active pendant le développement d'un programme sur le ''TMS3205402 DSK Kit'' de Texas Instrument.
  - L'accès direct à la mémoire par un périphérie peut utiliser une interruption de type DMA. Cinq interruptions DMAC (Direct Memory Acces Channel) sont prévues sur le C5402.

Interruptions

- **Registre de masque IMR :**
  - Toutes les interruptions, à part le RESET et le NMI, peuvent être masquées individuellement au moyen d'un bit dédié dans le registre de masquage d'interruption IMR (Interrupt Mask Register) :

15 - 14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Réservé	DMAC5	DMAC4	BXINT1 Or DMAC3	BRINT1 Or DMAC2	HPINT	INT3	TINT1 Or DMAC1	DMAC0	BXINT0	BRINT0	TINT0	INT2	INT1	INT0

  - L'ordre des bits dans le registre IMR correspond à l'ordre de priorité de ces interruptions. Un 1 logique dans un bit IMR valide l'interruption correspondante.
  - Exemple : instruction en code algébrique qui permet de valider l'interruption du HPI et du TIMER est la suivante :
  - *IMR = #208h*

Interruptions

- **Bit de mode d'interruption INTM :**
- **Le bit INTM (INTerrupt Mode) du registre d'état ST0, permet d'autoriser ou d'interdire globalement les interruptions non masquées. Les instructions d'autorisation et d'interdiction en code algébrique sont :**

- ***INTM = 0 ; Les interruptions non masquées sont autorisées***
- ***INTM = 1 ; Les interruptions non masquées sont interdites***

- **Le bit INTM est mis automatiquement à 1 lorsqu'une interruption est en service. Ce qui interdit les interruptions pendant l'exécution d'une autre. Lorsqu'on sort d'une routine d'interruption le bit INTM est remis à 0.**

Périphériques disponibles

- Ils dépendent de la version du circuit et comprennent :

- Un timer
- Un ou deux port série synchrone
- Un ou plusieurs ports bufférisés
- Un port à multiplexage temporel (TDM)
- Un port d'interface hôte (HPI)
- Une broche XF (eXternal Flag)
- Une broche BIO testée par logiciel
- Un contrôleur DMA

Implantation d'un filtre FIR en format fixe

$$y(n) = \sum_{k=0}^{N-1} h(k) \cdot x(n-k)$$

Données comprises entre +1 et -1 → Format fixe Q_{15} .

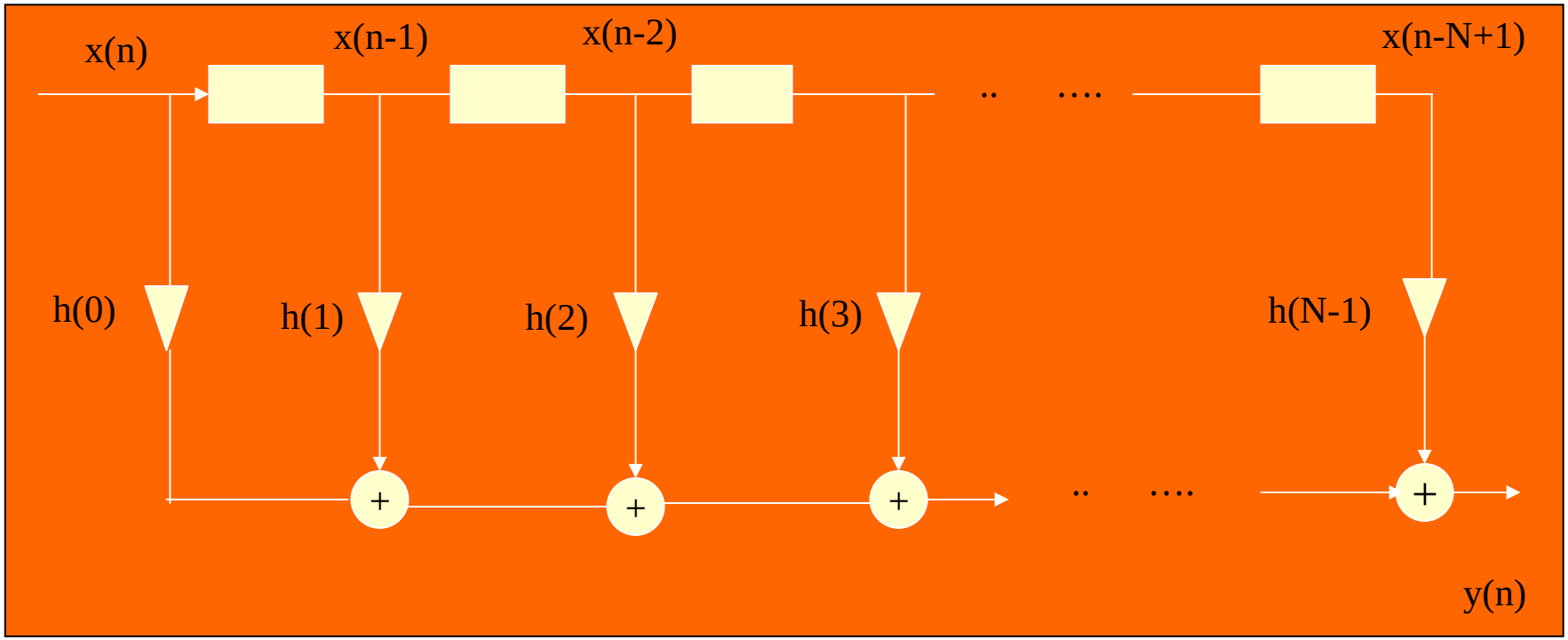
Produit format Q_{15} → Produit format Q_{30}

Filtre à gain unitaire → Si valeur absolue de l'entrée <1 alors
Valeur absolue de sortie <1 (format Q_{15})

y(n) correspond aux 16 bits de poids fort de l'ACCU après décalage de 1 bit à gauche

Implantation d'un filtre FIR en format fixe

Structure du filtre



Réalisation du filtre à l'aide d'un registre à décalage

Implantation d'un filtre FIR en format fixe

Opérations à effectuer pour calculer y(n) :

- Décaler les x(n-i) d'une case vers la droite dans la ligne à retard
- Lire le nouvel échantillon x(n), le stocker à la 1ère place de la ligne à retard
- Initialiser à zéro un ACCU pour le stockage de la somme des produits.
- Multiplier chaque échantillon x(n-i) par le coefficient correspondant et ajouter le produit à l'ACCU.
- Extraire de l'ACCU la partie correspondant à y(n) sur 16 bits.
- Envoyer y(n) vers le CNA

Implantation d'un filtre FIR en format fixe

Autre possibilité: Effectuer le décalage dès qu'une donnée a été utilisée.

⇒ Commencer les calculs par les données les plus anciennes

adresses	contenu
(adr_coef) j	h(N-1)
j+1	h(N-2)
j+2	h(N-3)
…	…
j+N-1	h(0)

Mémoire programme

adresses	contenu
k	x(n)
k+1	x(n-1)
k+2	x(n-2)
…	…
Adr_fin_datk+N-1	x(n-N+1)
k+N	Recopie de x(n-N+1)

Mémoire données

Implantation d'un filtre FIR en format fixe

L'instruction MACD

Syntaxe	MACD Smem,pmad,src
Opération	src=src + Smem x pmad
	T = Smem
	(Smem + 1) = Smem

• Produit de 2 valeurs aux adresses Smem en mémoire données et pmad en mémoire programme.

• Ajoute le résultat à l'ACCU (src : A ou B) .

• Ecrit la donnée Smem dans le registre T.

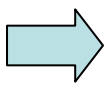
• Recopie la donnée de l'adresse Smem à l'adresse Smem + 1.

Implantation d'un filtre FIR en format fixe

L'instruction MACD

Association de MACD avec l'instruction de répétition RPTZ
RPTZ initialise à zéro un ACCU avant de commencer les calculs

• Stocker les coefficients h(i) en mémoire programme; l'adresse pmad est incrémentée à chaque répétition



Commencer les calculs par la donnée la plus ancienne

• Stocker les données en mémoire de données (adressage indirect pour pointer les données).

• Charger l'adresse du coefficient le plus ancien dans un registre auxiliaire.

Implantation d'un filtre FIR en format fixe

L'instruction MACD

Ecriture du programme en assembleur :

Initialisation :

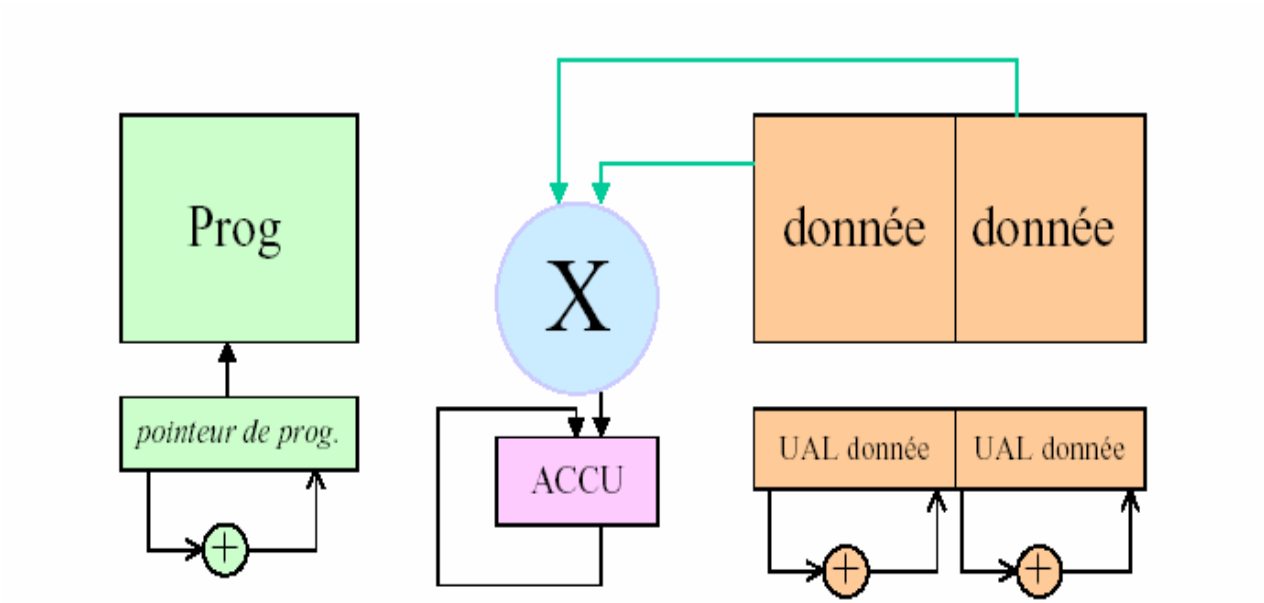
```
ST      #adr_fin_dat,AR1        ; valeur immédiate dans AR1
SSBX    FRCT                    ; mode fractionnaire, décalage de
                                ; 1 bit à gauche de la sortie du
                                ; multiplieur
```

Boucle de filtrage :

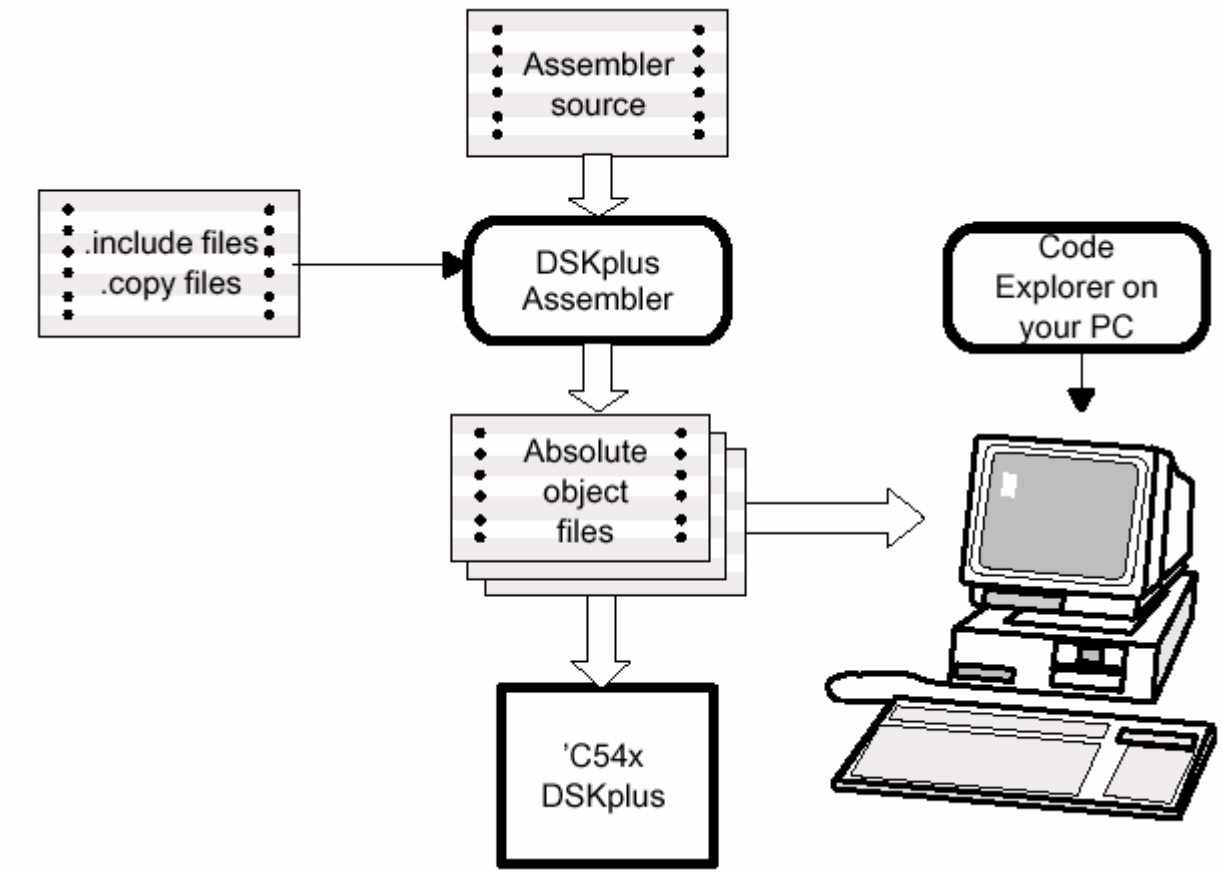
```
RPTZ    A,#N-1
MACD    *AR1-,adr_coef ,A
```

Le parallélisme lors de l'exécution

$$y(n) = \sum_{k=0}^{N} a(k)x(n-k)$$



Chaîne de développement des programmes sur DSP Starter Kit



La programmation des Registres Auxiliaires: AR0 – AR7

Syntaxe de L'opérande	Fonction	Description
*ARx	Adresse = ARx	ARx contient l'adresse de la donnée
*ARx-	Adresse = ARx, puis ARx = ARx - 1	Décrémentation de ARx après l'accès
*ARx+	Adresse = ARx, puis ARx = ARx + 1	Incrémentation de ARx après l'accès
*+ARx	Adresse = ARx + 1, puis ARx = ARx + 1	Incrémentation de ARx avant l'accès
*ARx-0	Adresse = ARx, puis ARx = ARx – AR0	AR0 est soustrait de ARx après l'accès
*ARx+0	Adresse = ARx, puis ARx = ARx + AR0	AR0 est ajouté à ARx après l'accès
*ARx-0B	Adresse = ARx, puis ARx = B(ARx – AR0)	AR0 est soustrait de ARx après l'accès avec propagation de la retenue à droite
*ARx+0B	Adresse = ARx, puis ARx = B(ARx + AR0)	AR0 est ajouté à ARx après l'accès avec propagation de la retenue à droite
*ARx-%	Adresse = ARx, puis ARx = Cir(ARx – 1)	ARx est décrémenté modulo BK après l'accès
*ARx+%	Adresse = ARx, puis ARx = Cir(ARx + 1)	ARx est incrémenté modulo BK après l'accès
*ARx-0%	Adresse = ARx, puis ARx = Cir(ARx – AR0)	AR0 est soustrait de ARx modulo BK après l'accès
*ARx+0%	Adresse = ARx, puis ARx = Cir(ARx + AR0)	AR0 est ajouté à ARx modulo BK après l'accès
*ARx(lk)	Adresse = ARx + lk, ARx = ARx	ARx + lk sert à l'accès et ARx est inchangé
*+ARx(lk)	Adresse = ARx + lk, puis ARx = ARx + lk	ARx + lk sert à l'accès et lk est ajouté à ARx
*+ARx(lk)%	Adresse = Cir(ARx + lk), ARx = Cir(ARx + lk)	ARx + lk modulo BK sert à l'accès et ARx est prend le resultat de ARx + lk modulo BK
*(lk)	Adresse = lk	lk sert d'adresse absolue

Champ Entrée/Sorties

- Les ports sont implantés à l'extérieur du DSP et ont un Espace de 64 kMot

- Accessible par l' intermédiaire des instructions de lecture ou d' écriture de port d'Entrée /Sortie :

  @InDATA = **PORT**(10h) ; Lecture du port 10h et transfert dans la mémoire InDATA
  PORT(20h) = @OutDATA ; Ecriture dans le port 20h du contenu de la mémoire OutDATA

Problème: Les périphériques ne sont pas aussi rapide que le DSP

Solution: L'accès au champ E/S peut être allongé de 1 à 7 cycles d'attente (Wait-State). Le registre SWWSR (SoftWaire Wait State Register) permet de spécifier le nombre de Wait-State pour chaque champ interne ou externe au DSP

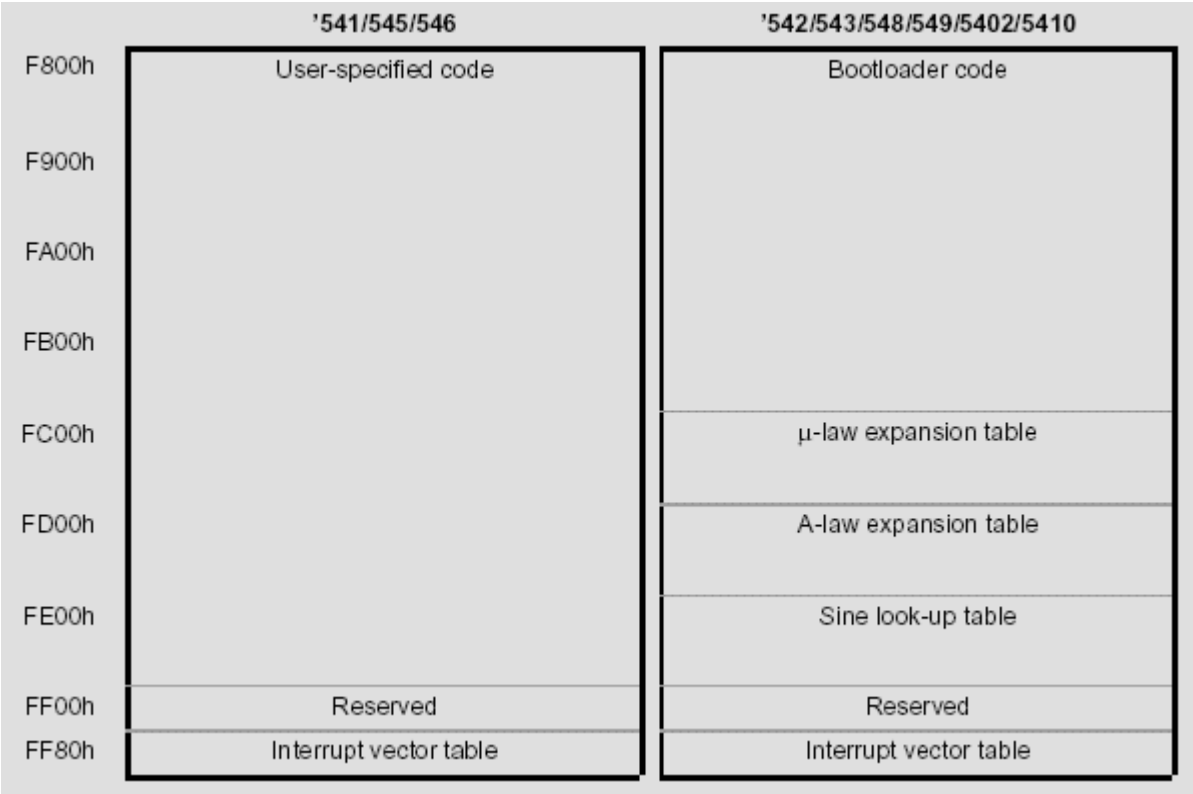
15	14 13 12	11 10 9	8 7 6	5 4 3	2 1 0
Réservé	E/S (0-7) 0000h-FFFFh	Données (0-7) 8000h-FFFFh	Données (0-7) 0000h-7FFFh	Programme (0-7) 8000h-FFFFh	Programme (0-7) 0000h-7FFFh
R	R/W	R/W	R/W	R/W	R/W

```
SWWSR = #0111111000111000b   ; 7 Wait-State pour le champ E/S,
                             ; zone mémoire de données externe (8000h-FFFFh)
                             ; et zone mémoire programme externe (8000h-FFFFh).
                             ; 0 Wait-State pour la zone mémoire de données (0000h-7FFFh interne et
                             ; externe) et la zone mémoire programme (0000h-7FFFh interne et externe)
```

Remarque :

Le cycle T_a de lecture ou d'écriture d'un champ programmé en N Wait-State, demandera $T_a = N + 1$ cycles d'horloge du DSP, avec N de 0 à 7.

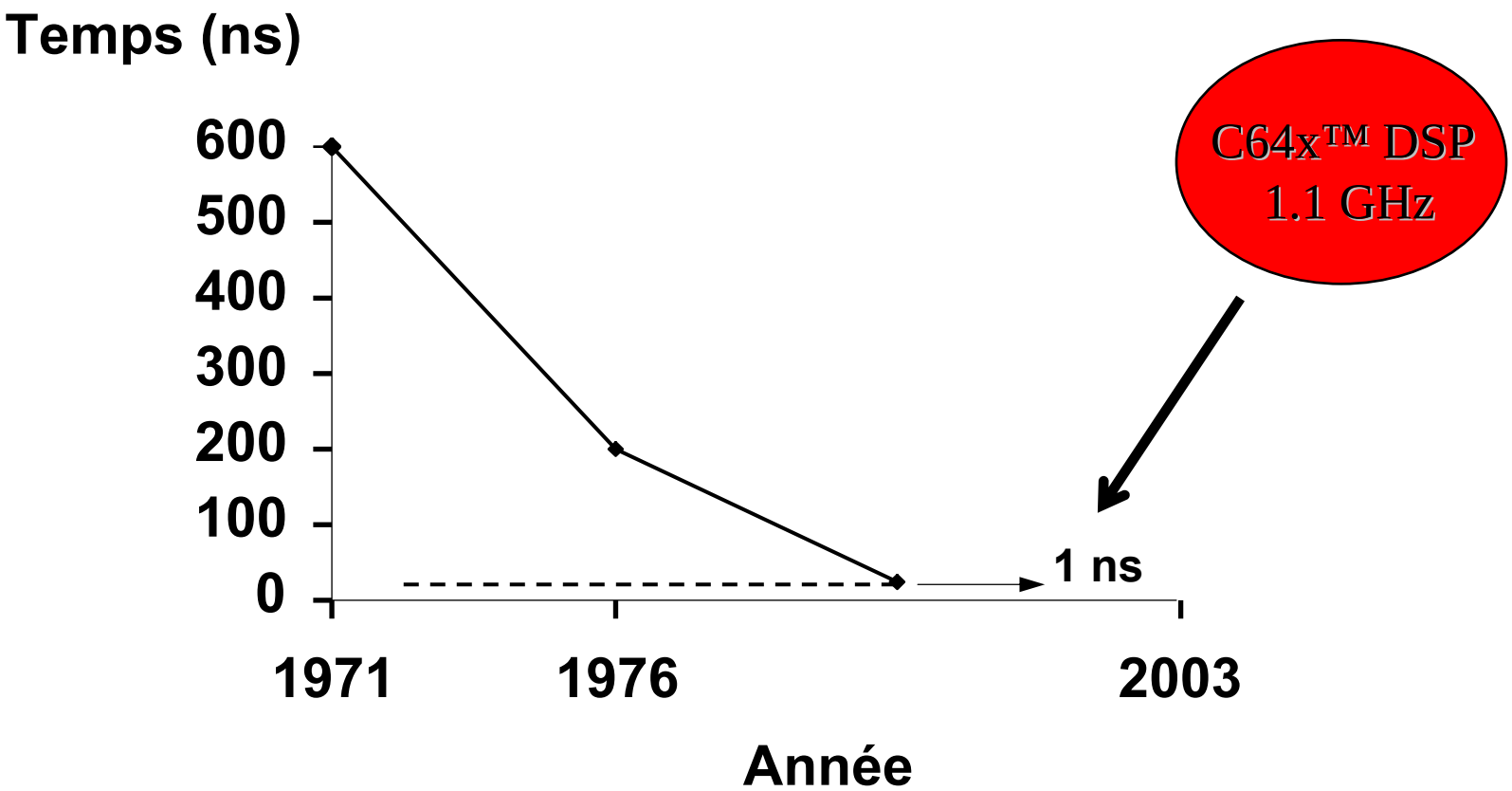
Organisation mémoire
De la famille 54



Les algorithmes les plus utilisés en traitement du signal

Algorithm	Equation
Finite Impulse Response Filter	$y(n)=\sum_{k=0}^{M} a_k\, x(n-k)$
Infinite Impulse Response Filter	$y(n)=\sum_{k=0}^{M} a_k\, x(n-k)+\sum_{k=1}^{N} b_k\, y(n-k)$
Convolution	$y(n)=\sum_{k=0}^{N} x(k)h(n-k)$
Discrete Fourier Transform	$X(k)=\sum_{n=0}^{N-1} x(n)\exp[-j(2\pi/N)nk]$
Discrete Cosine Transform	$F(u)=\sum_{x=0}^{N-1} c(u).f(x).\cos\left[\frac{\pi}{2N}u(2x+1)\right]$

La multiplication est l'opération centrale en DSP



La Multiplication câblée dans le temps