www.Mcours.com
Site N°1 des Cours et Exercices Email: contact@mcours.com

Cours d'architecture et système

DEUG MIAS 2e année

Introduction

- composants indispensables d'un ordinateur?
  - ➥ **unité de calcul :** (micro)processeur
  - ➥ **mémoire :** RAM
  - ➥ **dispositif d'entrée/sortie :** clavier, écran
- composants quasi indispensables?
  - ➥ **dispositif de stockage de masse :** disques
  - ➥ **système d'exploitation :** DOS, UNIX

Codage de l'information

- **c'est nécessaire !**
- **exemple :** transmission d'un message
  - ➥ **amplification directe** (porte-voix)
  - ➥ **codage** (courrier, signaux de fumée, signal électrique, ...)
- **en informatique :** codage binaire

Codage des entiers naturels

- **numération décimale**
  - ➥ **10 symboles ordonnés :** 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
  - ➥ **chiffres arabes** ⇒ manipulation de droite à gauche
- **algorithme d'énumération :**

```
subtype chiffres is integer range 0..9 ;
  for milliers in chiffres loop
    for centaines in chiffres loop
      for dizaines in chiffres loop
        for unités in chiffres loop
          put(milliers, 1) ; put(centaines, 1) ;
          put(dizaines, 1) ; put(unités, 1) ;
  loop ; loop ; loop ; loop ;
```


Base quelconque

- **exécution de l'algorithme précédent avec**
  `subtype chiffres is integer range 0..1;`

0000	0001	0010	0011	0100	0101	0110	0111
0000	0001	0002	0003	0004	0005	0006	0007
1000	1001	1010	1011	1100	1101	1110	1111
0008	0009	0010	0011	0012	0013	0014	0015

- **écriture générale :** (avec $0 \leq a_i < b$)
  - ➥ $N = a_n a_{n-1} ... a_1 a_0 \ _{(b)} = a_n b^n + a_{n-1} b^{n-1} + ... + a_1 b^1 + a_0 b^0$
  - ➥ $N = a_n a_{n-1} ... a_1 a_0 \ _{(b)} = (...((a_n b + a_{n-1})b + a_{n-2})b + ...)b + a_0$

Remarque sur la base usuelle

- la base 10 nous paraît « naturelle », il n'en est rien
- **autres bases usuelles :**
  - ➥ **20** chez les celtes ➜ quatre-vingt
  - ➥ **12** une douzaine d'œufs, 12 pence pour 1 shilling
  - ➥ **60** les heures, minutes, secondes
- **algorithmes de calcul ?** ➜ les mêmes

Puissances d'une base

$$N = a_n b^n + a_{n-1} b^{n-1} + ... + a_1 b^1 + a_0 b^0 \text{ avec } 0 \leq a_i < b$$

➥ mettons b^k en facteur :

$$\begin{aligned} N = {} & (... + a_n b^{n-pk} + ...)(b^k)^p + ... + \\ & (a_{(i+1)k-1} b^{k-1} + ... + a_{ik+1} b^1 + a_{ik} b^0)(b^k)^i + ... + \\ & (a_{2k-1} b^{k-1} + ... + a_{k+1} b^1 + a_k b^0)(b^k)^1 + \\ & (a_{k-1} b^{k-1} + ... + a_1 b^1 + a_0 b^0)(b^k)^0 \end{aligned}$$

➥ posons $c = b^k$ et $A_i = (a_{(i+1)k-1} b^{k-1} + ... + a_{ik+1} b^1 + a_{ik} b^0)$,

$$N = A_p c^p + A_{p-1} c^{p-1} + ... + A_1 c^1 + A_0 c^0 \text{ avec } 0 \leq A_i < c$$

Binaire, octal, hexadécimal

- **octal** = base 8, chiffres 0 1 2 3 4 5 6 7
- **hexadécimal** = base 16, chiffres 0 1 2 3 4 5 6 7 8 9 A B C D E F

0000	0001	0010	0011	0100	0101	0110	0111
00	01	02	03	04	05	06	07
0	1	2	3	4	5	6	7
1000	1001	1010	1011	1100	1101	1110	1111
10	11	12	13	14	15	16	17
8	9	A	B	C	D	E	F

➥ compacité + conversion rapide avec le binaire
⇒ hexadécimal souvent utilisé pour représenter les nombres

Addition d'entiers (ici base 3)

- **table d'addition**

+	0	1	2
0	0	1	2
1	1	2	10
2	2	10	11

- **exemple :** $212_{(3)} + 120_{(3)} = ?$

Codage des entiers relatifs

- **idée initiale :** ajouter 1 bit en tête du nombre pour indiquer le signe
- **problèmes :**
  - ➥ 2 façons de coder 0 : $-0 = 10000000$ ou $+0 = 00000000$
  - ➥ test obligatoire du signe avant addition comme en algèbre
  - ➥ nombre de chiffres fixé

Comment soustraire sans soustraction

- **exemple en base 10 :**

$$3247 - 1452 = 3247 + (9999 - 1452 + 1) - 10000$$

  - ➥ **complément à 9 :** $9999 - 1452 = 8547$ ➜ pas de retenue
  - ➥ **complément à 10 :** $8547 + 1 = 8548$ ➜ ajout de 1
  - ➥ **additionner :** $3247 + 8548 = 11795$
  - ➥ **soustraire 10000 :** $11795 - 10000 = 1795$
    ➜ laisser tomber le 1 de poids fort

Complément à 1, complément à 2

- **application de la même méthode en base 2**
- **exemple :**

$$101101001 - 1100010 =$$
$$101101001 + (111111111 - 001100010 + 1) - 1000000000$$

  - ➥ **complément à 1 :** transforme les 1 en 0 et les 0 en 1
- **nombres négatifs** = complément à 2 de la valeur absolue

Multiplication en binaire

- **algorithme usuel**
- **basée sur les décalages :**

```
      101101
    × 11001
    ------
      101101
     101101...
    101101....
    ----------
    10001100101
```


Nombres à « virgule »

- **virgule fixe** = entiers
- **virgule flottante**
  - ➥ **extension directe** de l'écriture des entiers :

$$N = a_n a_{n-1} ... a_1 a_0, a_{-1} a_{-2} ... a_{-p\ (b)}$$
$$= a_n b^n + a_{n-1} b^{n-1} + ...$$
$$+ a_1 b^1 + a_0 b^0 + a_{-1} b^{-1} + a_{-2} b^{-2} + ... + a_{-p} b^{-p}$$

  - ➥ **exemple** en binaire :

$$10,11011_{(2)} = 2 + 0 + 0,5 + 0,25 + 0 + 0,0625 + 0,03125$$
$$= 2,84375$$

ANSI/IEEE Standard 754-1985

- **simple précision**

1 bit	8 bits	23 bits
signe	exposant	mantisse

- **double précision**

1 bit	11 bits	52 bits
signe	exposant	mantisse

- $nombre = (-1)^{signe} \times 1, mantisse \times 2^{(exposant - biais)}$
  - ➥ **biais simple précision** $= 127$
  - ➥ **biais double précision** $= 1023$

www.Mcours.com
Site N°1 des Cours et Exercices Email: contact@mcours.com

Cas particuliers IEEE 754

signe	exposant	mantisse	valeur
_	0...0	0...0	± 0
_	0...0	...1...	nombre dénormalisé
_	1...1	0...0	$\pm\infty$
1	1...1	10...0	indéterminé
_	1...1	_	NaN

Codage des caractères

- **puissance lexicographique :** nombre de caractères représentables par un code
  - ➥ **exemple :** ASCII sur 7 bits ⇒ puissance lexicographique de $2^7 = 128$
- **types de caractères**
  - ➥ **caractères éditables :** lettres (a, B, ...), chiffres (0, 1, ...), symboles (☛, ✎, ...), opérateurs mathématiques (+, -, ...), diacritiques (é, ç, ø, ...), ...
  - ➥ **caractères non éditables :** espace arrière, retour chariot, échappement, tabulation, sonnerie, ...

Différents codages : codages historiques

- ➥ **ASCII** (American Standard Code for Information Interchange) sur 7 bits, l'un des plus utilisés, apparu en 1963
- ➥ **EBCDIC** sur 8 bits,vient des cartes perforées, gros systèmes IBM
- ➥ **TELETEL VIDEOTEX** sur 7 bits, minitel, ASCII et caractères graphiques
- ➥ **ANSI** (American National Standard Institute) sur 8 bits, utilisé par DOS et Windows 3, ASCII + page adaptée au pays (français = page n° 437)

Différents codages : codages modernes

- **codages ISO** (International Standard Organization)
  - ➥ **ISO 8859-1 ou latin1** sur 8 bits, ASCII + caractères européens (sauf œ), utilisé sur Windows 95, Unix, internet (HTML, courrier électronique et news)

```
 !"#$%&'()*+,-./0123456789:;<=>?@
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
abcdefghijklmnopqrstuvwxyz{|}
¡¢£¤¥¦§¨©ª«¬®¯°±²³´µ¶·¸¹º»¼½¾¿
ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚÛÜÝÞß
àáâãäåæçèéêëìíîïðñòóôõö÷øùúûüýþÿ
```

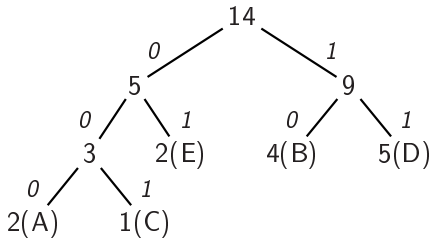
- **UNICODE :**
  - ➥ sur 16 bits (puissance lexicographique = 65 536)
  - ➥ 38 885 caractères codés pour 25 alphabets + nombres, symboles mathématiques, techniques, flèches, ...
  - ➥ couvre la plupart des langues écrites des 5 continents

Autres codages

- **correction d'erreur**
  - **bit de parité**
  - **autres :** code de Hamming, codes de bloc (VRC, LRC, code de Gray), codes cycliques (CRC)
- **code de Huffman**
  - code de longueur variable
  - compression de données
- **uuencode / uudecode**

Codage de Huffman

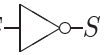
- **principe :** on recode les motifs les plus fréquents par des motifs plus courts
- **exemple :**
  - **code initial :** ABBCDBDDBAEDDE (14*8=112 bits)
  - **code final :** 00010100011110111110000001111101 (31 bits)
  - **arbre de codage :**



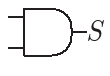
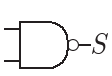
Circuits logiques

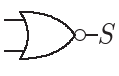
- **circuits combinatoires**
  - les sorties ne dépendent que des entrées
  - modélisés par l'algèbre de Boole
- **circuits séquentiels**
  - les sorties dépendent aussi du temps, il peut y avoir rétroaction
  - modélisés par les automates d'états finis

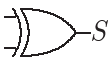
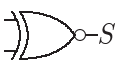
Fonctions logiques

nom	représentation	table de vérité
OUI	 $S = E$ $(S = E)$	E | S 0 | 0 1 | 1
NON	 $S = \neg E$ $(S = \overline{E})$	E | S 0 | 1 1 | 0

➥ **représentation :** symbole électronique, équation logique, (algèbre de Boole)

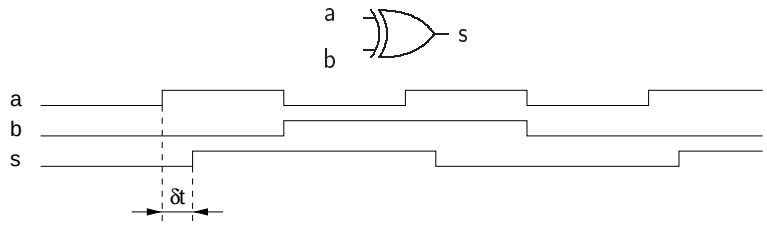
nom	représentation	table de vérité
ET [AND]	 $S = E_1 \wedge E_2$ $(S = E_1.E_2)$	E_1 | E_2 | S 0 | 0 | 0 1 | 0 | 0 0 | 1 | 0 1 | 1 | 1
NON-ET [NAND]	 $S = \neg(E_1 \wedge E_2)$ $(S = \overline{E_1.E_2})$	E_1 | E_2 | S 0 | 0 | 1 1 | 0 | 1 0 | 1 | 1 1 | 1 | 0

nom	représentation	table de vérité
OU [OR]	 $S = E_1 \vee E_2$ $(S = E_1 + E_2)$	E_1 | E_2 | S 0 | 0 | 0 1 | 0 | 1 0 | 1 | 1 1 | 1 | 1
NON-OU [NOR]	 $S = \neg(E_1 \vee E_2)$ $(S = \overline{E_1 + E_2})$	E_1 | E_2 | S 0 | 0 | 1 1 | 0 | 0 0 | 1 | 0 1 | 1 | 0

nom	représentation	table de vérité
OU exclusif [XOR]	 $S = \neg(E_1 \Leftrightarrow E_2)$ $(S = E_1 \oplus E_2)$	E_1 | E_2 | S 0 | 0 | 0 1 | 0 | 1 0 | 1 | 1 1 | 1 | 0
NON-OU exclusif [NXOR]	 $S = E_1 \Leftrightarrow E_2$ $(S = \overline{E_1 \oplus E_2})$	E_1 | E_2 | S 0 | 0 | 1 1 | 0 | 0 0 | 1 | 0 1 | 1 | 1

Circuits combinatoires

- **définition :** circuits logiques dont les sorties ne dépendent que des entrées
- **rôle du temps :**

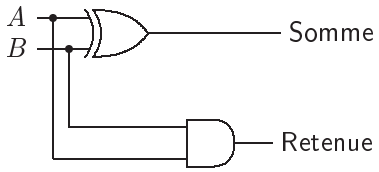


$\frac{1}{2}$ additionneur

- **somme de 2 bits**

A	B	R	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

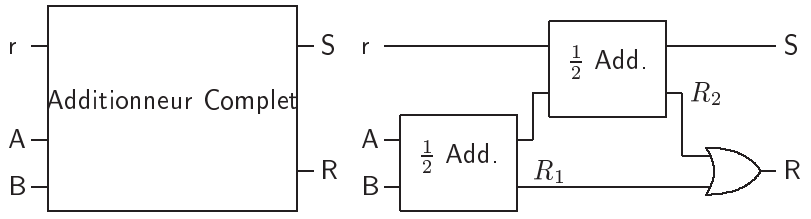
$$S = A \oplus B$$
$$R = A.B$$



Additionneur complet

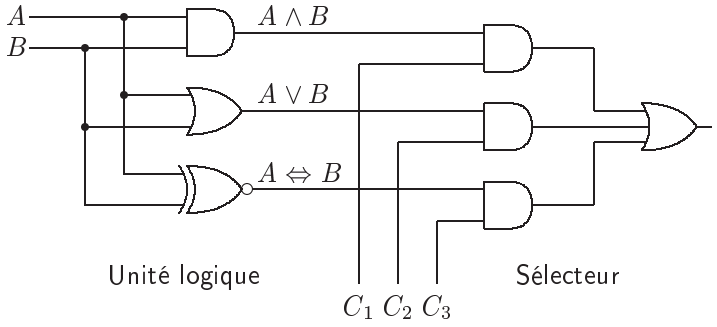
- somme de 3 nombres de 1 bit
- **but :** 2 nombres + 1 retenue
- **somme :** $S = (A + B) + r$
- **retenue :** vient de l'une des 2 sommes $\Rightarrow$ OU entre les deux retenues

Schéma de l'additionneur



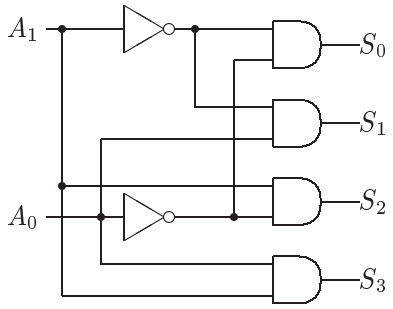
➥ on peut ensuite composer des additionneurs complets 1 bit pour réaliser des additionneurs de nombres codés sur plusieurs bits

Sélecteur et unité logique 1 bit



Décodeur d'adresses

- **utilisation** lors des accès mémoire pour sélectionner la cellule correspondant à l'adresse donnée
- **fonction logique :** convertisseur base 2 vers base 1

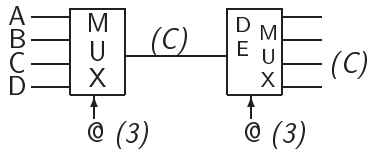


Unité arithmétique et logique

- **unité 1 bit :** composition éléments détaillés précédemment
  - ➥ **décodeur de commandes**
  - ➥ **unité logique**
  - ➥ **unité arithmétique**
- **unité 4 ou 8 bits :** composition d'unités 1 bit

Multiplexeur, démultiplexeur

- **but :** faire transiter sur un même fil des informations provenant de plusieurs sources



- **multiplexeur :** aiguillage de plusieurs entrées sur un même fil de sortie
- **démultiplexeur :** aiguillage d'une entrée sur plusieurs fils de sortie

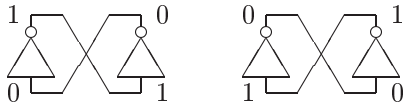
www.Mcours.com
Site N°1 des Cours et Exercices Email: contact@mcours.com

Circuits séquentiels

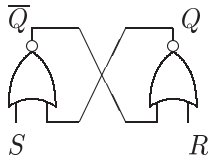
- les sorties dépendent des entrées *et du temps*
  - ➥ **dépendance des sorties par rapport aux entrées précédentes**
- **utilisation :** éléments de mémorisation, horloges

Bistable RS

- **principe :** circuit à deux états stables ≡ deux inverseurs en opposition



- **bistable RS :** impulsion sur $R \Rightarrow Q = 0, \overline{Q} = 1$ et impulsion sur $S \Rightarrow Q = 1, \overline{Q} = 0$



  - ➥ **attention :** impulsion simultannée sur R et S $\Rightarrow Q$ indéterminé

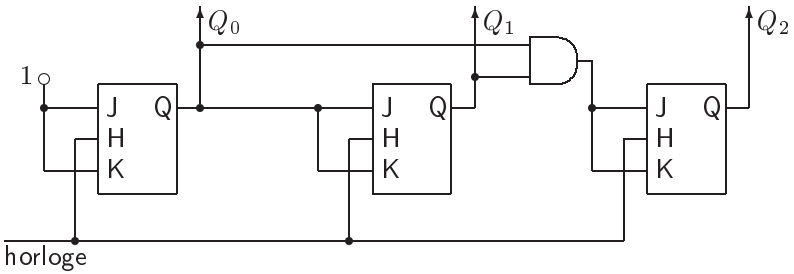
Bascule JK

- élément de base des cellules de mémorisation
  - ➥ basée sur le bistable RS
  - ➥ utilise le signal d'horloge
- **entrées :**
  - ➥ **Clear** initialisation à 0
  - ➥ **Preset** initialisation a 1
  - ➥ **J** passage à 1
  - ➥ **K** passage à 0
  - ➥ **Horloge** signal régulier

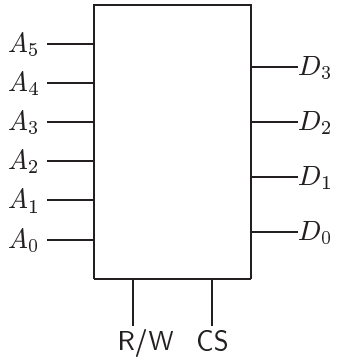
Table de transition de la bascule JK

C	P	J	K	H	$Q(t+1)$	$\overline{Q(t+1)}$
0	1	_	_	_	0	1
1	0	_	_	_	1	0
1	1	0	0	↑	$Q(t)$	$\overline{Q(t)}$
1	1	0	1	↑	0	1
1	1	1	0	↑	1	0
1	1	1	1	↑	$\overline{Q(t)}$	$Q(t)$
1	1	_	_	_	$Q(t)$	$\overline{Q(t)}$

Compteur binaire



Boîtier mémoire



A_i : broches d'adresse
D_i : broches de données
CS : Chip Select
R/W : lecture/écriture
+ broches d'alimentation électrique

mémoire de 64 (2^6) mots de 4 bits

Bus - définition

- **moyen de communication**
  - lignes de données
  - lignes d'adresses
  - lignes de contrôle
- **caractéristiques**
  - nombre de lignes
  - fréquence

Bus d'extension

- **rôle :** faire le lien entre le processeur et le reste de l'ordinateur
  - mémoire, périphériques d'entrée-sortie
  - doit disposer de connecteurs d'extension
  - présence d'un contrôleur

Bus ISA – Industry Standard Architecture

- **caractéristiques :**
  - **largeur :** 8 ou 16 bits
  - **fréquence :** 8 à 12 MHz
  - **débit :** 5 Mo/s
- standard des PC dès 1984, développé par Intel
- **inconvénients :**
  - faible débit
  - largeur de 16 bits

Bus MCA – Micro Channel Architecture

- **caractéristiques :**
  - **largeur :** 32 bits
  - **fréquence :** 10 MHz
  - **débit :** 32 Mo/s
- créé en 1987 par IBM
  - capable d'éviter les conflits mémoire de 2 composants
- **inconvénients :**
  - fréquence inférieure à celle des proc. modernes
  - incompatible ISA
  - IBM demande une redevance

Bus EISA – Extended Industry Standard Architecture

- **caractéristiques :**
  - **largeur :** 32 bits
  - **fréquence :** 20 MHz
  - **débit :** 32 Mo/s
- créé en 1988, compatible ISA
- **inconvénients :**
  - relativement coûteux
  - fréquence limitée

Bus local VESA – Video Electronics Standard Association

- **bus local :** uniquement entre processeur, mémoire et vidéo
- **caractéristiques :**
  - **largeur :** 32 ou 64 bits
  - **fréquence :** 33 MHz
  - **débit :** 132 Mo/s
- créé en 1992 pour accélérer transfert vidéo
- **inconvénients :**
  - 3 connecteurs maxi
  - fréquence limitée et dépendant du processeur

www.Mcours.com
Site N°1 des Cours et Exercices Email: contact@mcours.com

Bus PCI – Peripheral Components Interconnect

- **caractéristiques :**
  - **largeur :** 64 bits (32 possible)
  - **fréquence :** 66 MHz
  - **débit :** 132 Mo/s
- créé en 1992
  - compatible ISA et EISA
  - Plug and Play
  - 5 connecteurs mais possibilité de plusieurs contrôleurs

Bus PCMCIA – Personnal Computer Memory Card Industry Association

- **caractéristiques :**
  - **largeur :** 16 bits
  - **fréquence :** 33 MHz
- pour les portables
  - seulement 26 lignes d'adresse ⇒ 64 Mo
  - connecteurs de petite taille
  - connexion et déconnexion sans couper l'alimentation

Bus AGP – Advanced Graphic Port

- **bus local :** uniquement entre processeur, mémoire et vidéo
- **caractéristiques :**
  - **largeur :** 32 bits
  - **fréquence :** 66 MHz
  - **débit :** 528 Mo/s
- créé en 1997 par Intel
  - accès direct à la mémoire pour stocker textures (DIrect Memory Execute)
  - pas encore de support logiciel pour DIME

Bus SCSI (Small Computer System Interface)

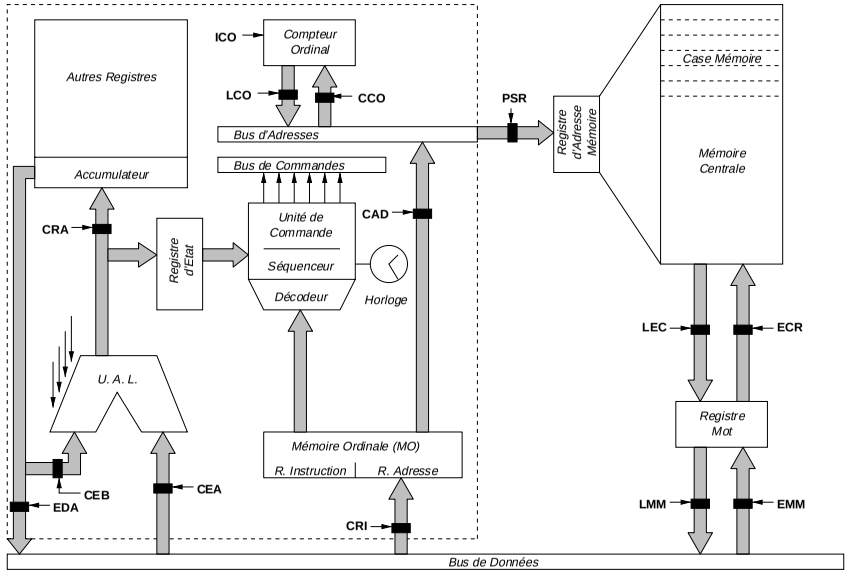
- bus d'entrées/sorties parallèles
  - jusqu'à 8 périphériques (dont contrôleur)
  - **débit :** de 5 Mo/s pour le SCSI-1 à 40 Mo/s pour l'Ultra Wide SCSI
  - Plug and Play
- « coprocesseur » qui soulage le processeur de la gestion des transferts
- limite des bus parallèles ⇒ bus série (SCSI-3)

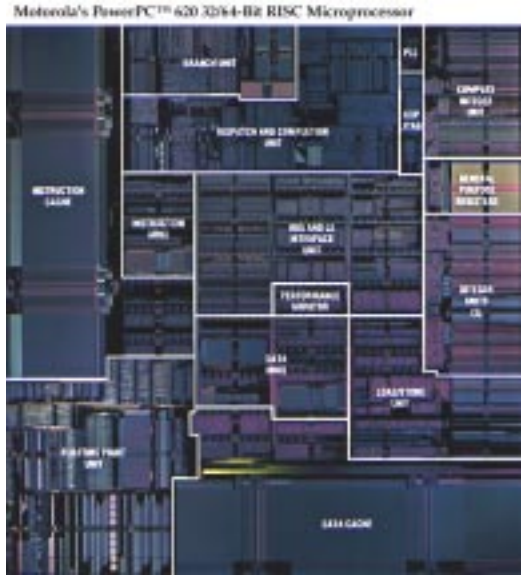
Bus série

- **USB – Universal Serial Bus**
  - **débit :** 1,5 Mo/s
  - jusque 127 périphériques
  - Hot Plug and Play
  - utilisation pour les périphériques courants
- **IEEE 1394 - Fire Wire**
  - **débit :**12,5 Mo/s, 25 Mo/s ou 50 Mo/s
  - jusque 63 périphériques
  - Hot Plug and Play
  - utilisation pour le multimédia

Microcommandes

- **LCO :** Lecture du Compteur Ordinal
- **CCO :** Chargement du Compteur Ordinal
- **PSR :** Pointage Sur Registre
- **LEC :** LECture
- **ECR :** ECRiture
- **LMM :** Lecture Mot Mémoire
- **EMM :** Ecriture Mot Mémoire
- **CRI :** Chargement Registre Instruction
- **CAD :** Chargement ADresse
- **CRA :** Chargement Registre Accumulateur
- **CEA :** Chargement Entrée A
- **CEB :** Chargement Entrée B
- **EDA :** Envoi de Données Accumulateur
- **ICO :** Incrémentation du Compteur Ordinal
- **NOP :** No OPération





Fonctionnement du microprocesseur

- plusieurs couches d'interprétation
  - portes logiques
  - micro-instructions
  - langage machine
  - assembleur
  - langage de haut niveau (scheme, ada, ...)

Évolution des microprocesseurs

- **8088** (1979)
  - utilisé dans le premier IBM PC
  - architecture interne 16 bits
  - bus 8 bits
  - adressage de 1 Mo maximum
  - fréquences d'horloge : 5/8 MHz
  - 0,029 millions de transistors

- **80286** (1982)
  - utilisé dans le PC AT (Advance Technology)
  - vrai microprocesseur 16 bits
  - capable de travailler avec de la *mémoire virtuelle*
  - introduit le concept de mode protégé (*multitâche*)
  - adressage de 16 Mo de mémoire physique et de 1 Go de mémoire virtuelle par tâche
  - fréquences d'horloge : 6/10/12 MHz
  - 0,134 millions de transistors

- **i386** (1988)
  - microprocesseur 32 bits
  - adressage de 4 Go de mémoire physique et 64 To de mémoire virtuelle
  - *coprocesseur arithmétique* 80387 externe
  - *cache* de 16 octets
  - faible voltage et gestion d'économies d'énergie
  - fréquences d'horloge : 16/20/25/33 MHz
  - 0,275 millions de transistors
- **i386SX**
  - version réduite du i386
  - bus externe 16 bits

- **i486** (1991)
  - i386 + coprocesseur arithmétique interne
  - optimisations
  - plusieurs versions :

nom	fréquence (MHz)	M trans.	cache	copro.
i486SX	16/20/25/33	0,9	8 Ko	non
i486DX	25/33/50	1,2	8 Ko	interne
i486DX2	50/66	1,2	8 Ko	interne
i486DX4	75/100	1,6	16 Ko	interne

- **Pentium** (1993)
  - bus interne de 64 bits et externe de 32 bits
  - un i486DX et un i486SX : technologie *superscalaire* (2 instructions par cycle d'horloge)
  - unités de calcul *pipelinées*
  - *prédiction de branchements*
  - cache : 8 Ko pour instructions et 8 Ko pour données
  - fréquences : 75/90/100/120/133/150/166/200 MHz, rapports vitesse externe / vitesse interne fractionnaire (1,5/2/2,5/3)
  - 3,1 millions de transistors

- **Pentium Pro** (1995)
  - microprocesseur *RISC* qui interprète les instructions i386 (découpage plus fin, *réordonnancement*, registres banalisés)
  - cache niveau 2 interne (à la même vitesse que le processeur), taille 256 ou 512 Ko
  - fréquences : 150/200 MHz
  - 5,5 millions de transistors
- **Pentium MMX** (1996)
  - ajout du jeu d'instructions multimédia MMX (57 instructions *SIMD* et registres 64 bits)
  - doublement de la taille des caches, diverses optimisations
  - fréquences : 133/166/200/233/266 MHz
  - 4,5 millions de transistors

- **pentium II** (1997)
  - Pentium Pro + MMX + *exécution spéculative*
  - 512 Ko cache N.2 ($\frac{1}{2}$ vitesse du processeur) dans le même boîtier
  - fréquences : 233/266/300/333/350/400/450 MHz
  - 7,5 millions de transistors
- **variantes**
  - **Celeron** (1998) (→ vitesse cache = vitesse proc.)
    - 266/300 MHz | pas de cache N.2 | 7,5M transistors
    - 300/333/366/400 MHz | 128Ko cache N.2 | 19M transistors
  - **Xeon** (1998) (→ vitesse cache = $\frac{1}{2}$ vitesse proc.)
    - 400/450 MHz | 512Ko/1Mo/2Mo cache N.2
    - jusqu'à 8 proc. sur la même carte mère (4 pour le pentium pro et 2 pour le pentium II)

- **pentium III** (1999)
  - pentium II + 70 instructions SIMD
  - fréquences : 450/500 MHZ (bus : 100 MHz)
  - numéro de série intégré
- **Autres architectures**
  - **AMD, Cyrix, IBM, ... :** clônes d'Intel
  - **Motorola :** 680x0
  - **Motorola, Apple, IBM, ... :** PowerPC
  - **IBM :** RS6000, RS10000
  - **Silicon Graphics :** MIPS
  - **DEC (Compaq) :** Alpha
  - **Sun, ... :** Sparc
  - **Hewlett-Packard :** PA-RISC
  - **ARM :** StrongArm

Assembleur

- jeu d'instruction de base = compormis entre
  - simplicité
  - universalité
  - efficacité
  - confort du programmeur
- nombre d'instructions typiquement compris entre 50 et 250
  - **RISC :** petit nombre d'instructions élémentaires d'exécution rapide (nécessité d'un excellent compilateur)
  - **CISC :** riche jeu d'instructions de tailles variables

Généralités

- classement des instructions :
  - transfert de données
  - opérations arithmétiques et logiques
  - contrôle de séquence
  - entrées/sorties
  - manipulations diverses

Z80 : architecture

- amélioration du 8080 d'Intel
- largeur du bus interne : 8 bits

Registres

- accumulateur 8 bits : A
- registres généraux 8 bits : B, C, D, E, H et L
  - utilisables par paires (16 bits) BC, DE et HL (Hi Low)
- registre indicateur 8 bits : F (flags)
- compteur ordinal 16 bits : PC
- pointeur de pile 16 bits : SP
- registres d'index 16 bits : IX et IY
- une copie des registres A, B, C, D, E, F, H et L : A', B', C', D', E', F', H' et L' qu'on peut échanger rapidement

Transferts de données

- transferts entre registres et mémoire : instruction `LD destination, source`
- la richesse du jeu d'instruction vient des nombreux modes d'adressage :
  - par registre
  - implicite
  - immédiat
  - immédiat étendu
  - étendu
  - indirect par registre
  - indexé
  - relatif
  - (page zéro)
  - (par bit)

Adressage par registre

- la donnée est spécifiée par le nom du registre qui la contient
- **exemple :** `LD A, B`

Adressage implicite

- concerne les instructions travaillant sur l'accumulateur
- **exemple :** `SUB B`

Adressage immédiat

- la donnée est contenue dans l'instruction
- **exemples :**
  - `LD A, 32`
  - `LD B, 20H`

Adressage immédiat étendu

- la donnée (sur 16 bits) est contenue dans l'instruction
- **exemples :**
  - `LD BC, 2815`
  - `LD HL, 0AFFH`

Adressage étendu

- la donnée est spécifiée par son emplacement mémoire donné dans l'instruction
- **exemple :**
  - `LD (1203H), A`
  - `LD BC, (1203H)`

Adressage indirect par registre

- la donnée est spécifiée par son emplacement mémoire lu dans l'une des paires de registres HL ou SP
- **exemple :** `LD (HL), A`

Adressage indexé

- 2 registres 16 bits d'usage spécial : IX et IY
- adressage indirect par registre + déplacement sur 8 bits en complément à 2
- **exemples :**
  - `LD A, (IX + 02H)`
  - `LD (IY + FFH), A`

Adressage relatif

- spécialisé pour les branchements, spécifie un déplacement (codé en complément à 2) par rapport à l'adresse courante
- **exemple :** `JR 09H`

Échange entre registres

- instruction `EX` :
  - `EX AF, AF'`
  - `EX DE, HL`
  - `EX (SP), HL`
  - `EX (SP), IX`
  - `EX (SP), IY`
- instruction `EXX`
  - permutte les registres BC,DE et HL avec BC', DE' et HL'

Opérations arithmétiques et logiques

- en général, pour les opérations 8 bits :
  - le premier opérande est le contenu de l'accumulateur
  - le résultat est stocké dans l'accumulateur
  - le registre F est modifié selon le résultat
- contenu du registre indicateur F (`SZHPNC`) :
  - `S` : bit de signe du résultat
  - `Z` : 1 si résultat nul
  - `H` : demi-retenue
  - `P` : parité/dépassement de capacité en complément à 2
  - `N` : 1 si soustraction, 0 si addition
  - `C` : retenue

www.Mcours.com
Site N°1 des Cours et Exercices Email: contact@mcours.com

Incrémentation / décrémentation

- `INC source` : ajoute 1 à la source
- `DEC source` : retranche 1 à la source
- modes d'adressage possibles pour la source :
  - par registre (8 ou 16 bits)
  - indirect par registre
  - indexé

Opérations arithmétiques 8 bits

- `ADD A, source` : addition
- `ADC A, source` : addition avec retenue entrante
- `SUB source` : soustraction
- `SBC A, source` : soustraction avec retenue entrante
- modes d'adressage possibles pour la source :
  - par registre
  - indirect par registre
  - indexé
  - immédiat
- opérations en adressage implicite :
  - `CPL` : complément à 1
  - `NEG` : complément à 2

Opérations logiques

- `AND source`, `XOR source`, `OR source`
  - ➥ modes d'adressage comme pour les opérations arithmétiques
- `CP source` : comparaison avec l'accumulateur
  - ➥ ne modifie ni l'accumulateur, ni la source
  - ➥ place les indicateurs (registre F)

Opérations arithmétiques 16 bits

- `ADD dest, src`
  - ➥

dest	sources possibles
HL	BC, DE, HL, SP
IX	BC, DE, SP, IX
IY	BC, DE, SP, IY

- `ADC HL, src` et `SBC HL, src`
  - ➥ src : BC, DE, HL, ou SP

Contrôle de séquence

- exécution d'un programme :
  1. lire et charger (PC) dans un registre interne ; incrémenter PC
  2. décoder le premier octet de l'instruction, en déduire si d'autres octets doivent être lus, sinon, aller à l'étape 4
  3. continuer à lire (PC) et à incrémenter PC jusqu'à lecture complète de l'instruction
  4. exécuter l'instruction et retour à l'étape 1
- transférer le contrôle de séquence = modifier PC

Branchements inconditionnels

- `JP nn`, `JP (HL)`, `JP (IX)` et `JP (IY)`
  - ➥ branchement à une adresse immédiate ou indirecte par registre
- `JR e`
  - ➥ branchement relatif

Branchements conditionnels

- `JP cond, nn` : conditions testées sur le registre F

`NZ`	$Z = 0$	résultat non nul
`Z`	$Z = 1$	résulat nul
`NC`	$C = 0$	pas de retenue
`C`	$C = 1$	retenue
`PO`	$P = 0$	parité impaire ou pas de dépassement de capacité
`PE`	$P = 1$	parité paire ou dépassement de capacité
`P`	$S = 0$	résultat positif
`M`	$S = 1$	résultat négatif

Branchements conditionnels (suite)

- `JR cond, e` :
  - ➥ conditions possibles : `NZ`, `Z`, `NC` ou `C`
- `DJNZ e` :
  - ➥ décrémente B
  - ➥ si B est non nul, fait le branchement relatif

Mécanisme de pile

- analogie avec une liste de scheme
- deux fonctions de manipulation :
  - ➥ `PUSH rr` : ajoute le contenu du registre rr (AF, BC, DE, HL, IX ou IY) au sommet de la pile et enlève 2 à SP
  - ➥ `POP rr` : charge dans rr la valeur (SP) et ajoute 2 à SP
- utilité : appel de sous-programmes

Appel de sous-programmes

- `CALL nn` : appel du sous-programme à l'adresse ; équivaut à `nn`
  - ➥ `PUSH PC`
  - ➥ `LD PC, nn`
- `RET` : retour à l'appelant ; équivaut à
  - ➥ `POP PC`
- il existe les variantes conditionnelles :
  - ➥ `CALL cond, nn`
  - ➥ `RET cond`

Autres instructions

- ❑ opérations sur les bits, décalages
- ❑ calcul en binaire codé décimal
- ❑ opérations de transfert de blocs
- ❑ opérations de recherche par blocs
- ❑ opérations d'entrée/sortie
- ❑ gestion des interruptions

Système d'exploitation

- ❑ un ordinateur sans logiciel ne sert à rien
- ❑ types de logiciels :
  - ➥ logiciels d'application
  - ➥ logiciels utilitaires (aident à développer les applications)
  - ➥ système d'exploitation
- ❑ rôles du système d'exploitation :
  - ➥ interface entre le matériel et l'utilisateur
  - ➥ gestion des tâches de bas niveau
  - ➥ gestion des ressources

Historique

- ❑ **préhistoire**
  - ➥ au début, pas de logiciel, programmation en binaire
  - ➥ arrivée de l'assembleur, utilisation self service
  - ➥ opérateur
  - ➥ *moniteur* qui supprime les manipulations manuelles
- ❑ **systèmes par lots**
  - ➥ fin des années 50
  - ➥ machines à transistors et bandes magnétiques
  - ➥ gestion des entrées/sorties par un ordinateur auxiliaire

Spooling

- milieu des années 60
- amélioration techniques : disques magnétiques, interruptions
  - ➥ indépendance des entrées/sorties
- *Simultaneous Peripheral Operation On-Line*
- ordre d'exécution choisi par le moniteur
  - ➥ **ordonnanceur**

Multi-programmation

- aggrandissement de la différence de vitesse entre processeur et périphériques d'entrée/sortie
  - ➥ temps morts de plus en plus génants
- plusieurs programmes chargés en mémoire simultannément
  - ➥ **allocateur :** gestion instantannée du CPU en tenant compte de la plannification et des priorités données par l'ordonnanceur
  - ➥ multi-tâches *coopératif* ou *préemptif*
- possibilité de systèmes multi-utilisateurs

Problèmes

- comment partager le CPU entre plusieurs programmes et sauvegarder le contexte d'exécution de chacun ?
- gestion de la mémoire ?
  - ➥ partage
  - ➥ taille limitée
- gestion des entrées/sorties ?
  - ➥ ne pas mélanger les données
- protection des programmes et des données d'éventuelles erreurs ?

Virtualisation de la machine

- **buts**
  - ➥ indépendance par rapport au matériel
  - ➥ simplicité de programmation
- **moyens**
  - ➥ machine à deux états

    superviseur ou noyau

    utilisateur ou protégé
  - ➥ notion de *processus* = programme en exécution

Structure d'un système

- **monolitiques :** un gros bloc en mode superviseur
  - ➥ performances maximales mais fermé, peu protable, difficile à maintenir, peu sûr
  - ➥ MS-DOS, Windows 9x, vieux UNIX, Linux 1.x, MacOS
- **modulaires :** on peut ajouter ou enlever des composants à tout moment
  - ➥ très performant, plus ouvert et flexible, peu sûr
  - ➥ Linux 2.x, FreeBSD, HP-UX, Solaris, RiscOS

Structure d'un système - bis

- **micronoyau :** seul le minimum tourne en mode superviseur, un serveur en mode utilisateur fait le reste
  - ➥ excellente portabilité, sûreté, baisse de performances
  - ➥ Digital Unix, Mk-Linux, BSD, Rhapsody
- **multiserveurs :** un micronoyau + plusieurs petits serveurs
  - ➥ sûreté de fonctionnement maximale, portabilité, dégradation des performances, problèmes de sécurité
  - ➥ Windows NT
  - ➥ BeOS, mais avec des serveurs en mode superviseur pour des raisons de performance

Processus

- permettent de gérer le multitâche
- pas de zone mémoire réservée ⇒ programmes relogeables
- gestion du temps : ordonnancement
- gestion des ressources : problème de l'interblocage

Ordonnancement

- chaque processus a un descripteur
- états possibles d'un processus :
  - ➥ en exécution
  - ➥ bloqué (en attente d'un événement)
  - ➥ prêt (en attente du processeur)
  - ➥ suspendu
- ordonnancement coopératif ou préemptif
  - ➥ quantum de temps de l'ordre de la milliseconde pour les systèmes temps réel à la dizaine de millisecondes pour les systèmes grand public
- listes de priorités et/ou tourniquet

Interblocage

- accès concurrent à des ressources partagées, problème général de la concurrence ou du parallélisme
  - ➥ structures de données adaptées : sémaphores, moniteurs
  - ➥ algorithmes de détection, reprise ou évitement
- voir cours de licence et maîtrise

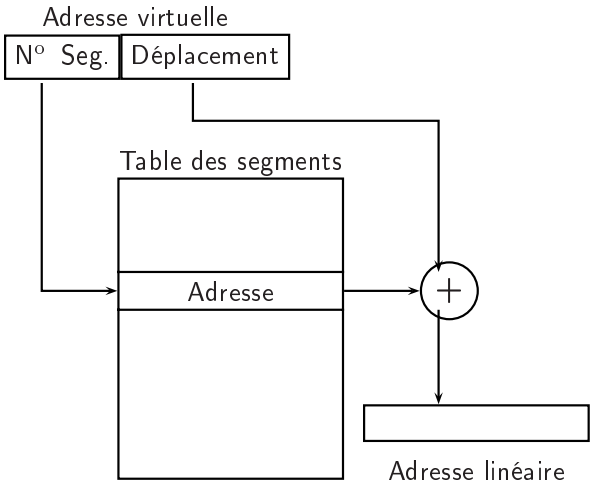
Gestion de la mémoire

- partage et protection entre les processus
  - ➥ segmentation
- mémoire virtuelle
  - ➥ pagination

Partage de la mémoire

- partitions de la mémoire en morceaux de tailles fixes
  - ➥ gaspillage de place en raison de la différence de taille entre morceaux de mémoire et programmes
- partitions de tailles variables (au cours du temps)
  - ➥ apparition de trous lors du fonctionnement $\Rightarrow$ retassement par *translation dynamique*
- protection entre processus
  - ➥ registres bornes + déplacement
- segmentation : division des programmes en segments
  - ➥ meilleure utilisation d'une mémoire fragmentée
  - ➥ une table de segments par processus
  - ➥ adresse = n° de segment + décalage

Segmentation : calcul d'adresse



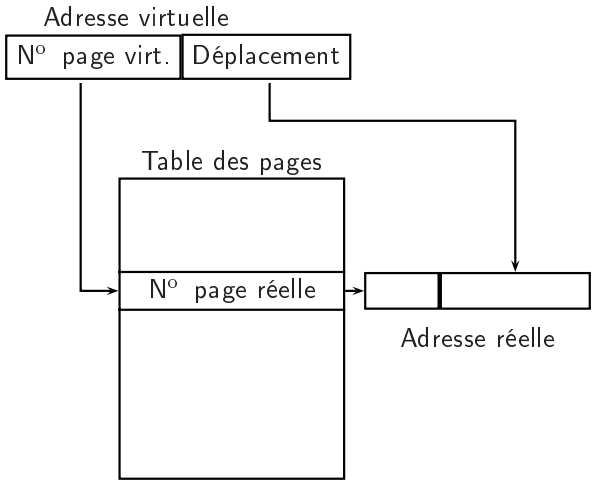
Mémoire virtuelle

- traîtement séparé de 2 types d'adresses
  - ➥ **virtuelles :** référencées dans un programme
  - ➥ **réelles :** mémoire physique
- le programme voit une mémoire de 2^n mots (4 Go sur Pentium) indépendante de la localisation du programme en mémoire physique
- utilisation de la mémoire de masse comme extension de la mémoire centrale

Pagination

- découpage des 2 espaces d'adressage en morceaux de la même taille (entre 512 et 4096 mots) : les pages
- table des pages pour faire la correspondance adresse virtuelle / adresse physique
  - ➥ bit de présence ou absence de la page en mémoire physique
  - ➥ emplacement sur le disque
  - ➥ emplacement éventuel en mémoire physique
  - ➥ bit de modification
- quelle page remplacer quand on a besoin de place?
  - ➥ algorithme LRU
  - ➥ pages résidentes pour des processus privilégiés
  - ➥ pages partagées par plusieurs processus

Pagination : calcul d'adresse



Hiérarchie mémoire

- **mémoire centrale**
  - ➥ faible capacité (quelques dizaines de Mo)
  - ➥ rapide : temps d'accès 50 ns, taux de transfert 500 Mo/s
  - ➥ volatile
  - ➥ diverses technologies : DRAM, SRAM, VRAM
- **stockage de masse**
  - ➥ grande capacité (quelques Go)
  - ➥ lente : temps d'accès 10ms, taux de transfert 10 Mo/s
  - ➥ disques durs
  - ➥ supports amovibles (disquettes, CD-ROMs, bandes, ...)

Types de mémoire centrale

- **ROM (Read Only Memory) :** mémoire non volatile
  - ➥ **PROM :** Programable ROM
  - ➥ **EPROM :** Erasable PROM (pour le BIOS)
- **SRAM (Static Random Acces Memory) :** bascules
  - ➥ mémoire cache, rapide (10 ns) et chère
  - ➥ taille typique = 256 ou 512 Ko

Types de mémoire centrale - bis

- **DRAM (Dynamic RAM) :** condensateurs, rafraîchissement
  - ➥ bon marché, plus lente que la SRAM (50ns)
  - ➥ organisation matricielle avec accès par lignes

Type	accès	débit
Fast Page Mode (FPM)	5-3-3-3	176 Mo/s
Extended Data Output RAM (EDO RAM)	5-2-2-2	264 Mo/s
Synchronous Dynamic RAM (SDRam)	5-1-1-1	528 Mo/s

  - ➥ SIMM (Single Inline Memory Module) de 32 bits (36 avec parité) ou DIMM (Dual IMM) de 64 bits (72 avec parité)
- **VRAM (Vidéo RAM) :** accès simultanné en lecture et en écriture

Mémoire cache

- **localité des références**
  - ➥ boucles, procédures
  - ➥ structure de tableau
- **idée de base :**
  - ➥ une petite mémoire rapide
  - ➥ y garder les données accédées fréquemment

Fonctionnement du cache

- même principe de fonctionnement que la pagination
- **opération de base :**
  - ➥ examiner le cache
  - ➥ si donnée présente, y accéder depuis la mémoire rapide
  - ➥ sinon, lire la donnée depuis la mémoire lente et copier un bloc de mots contenant la donnée considérée dans la mémoire cache
- **paramètres :**
  - ➥ taille du bloc
  - ➥ stratégie de recherche dans le cache
  - ➥ stratégie de remplacement des blocs

Stratégie de recherche dans le cache

- **il faut un accès rapide aux données**
- deux stratégies principales
  - **accès associatif**
  - **accès direct**
  - autres : combinaisons de ces deux stratégies
- **exemple :**
  - **mémoire principale :** 32K mots de 12 bits
  - **mémoire cache :** 512 mots

Mémoire associative

- **fonctionnement :**
  - stockage de couples (adresse, donnée)
  - lors d'un accès, chaque cellule compare son adresse avec celle demandée et retourne la donnée associée s'il y a correspondance
- **avantages :**
  - souplesse, rapidité
- **inconvénients :**
  - coût (logique donc surface)

Accès direct

- **principe :**
  - découpage de l'adresse en 2 parties : l'index (9 bits) et l'étiquette (6 bits)
  - cache = RAM adressée par l'index, stockage de couples (étiquette, donnée)
  - lors d'un accès, comparaison des étiquettes de l'adresse demandée et de la cellule stockée à l'index demandé
- **inconvénient :**
  - cas de deux mots accédés fréquemment et de même index
- **extension :** remplacer par blocs
  - découpage de l'index en deux parties : numéro du bloc (6 bits) et rang au sein du bloc (3 bits) avec étiquette commune à toutes les cellules d'un bloc

Stratégies de remplacement

- que faire quand le cache est plein ?
  - remplacer un bloc déjà présent par le nouveau
- **stratégies :**
  - **aléatoire**
  - **FIFO :** First In First Out (file d'attente)
  - **LRU :** Least Recent Used

www.Mcours.com
Site N°1 des Cours et Exercices Email: contact@mcours.com

Écriture dans le cache

- que se passe-t-il quand on veut écrire une donnée à une adresse présente dans le cache?
- **écriture à travers le cache**
  - mise à jour de la mémoire principale en même temps que le cache
  - simple
  - assure la cohérence de la mémoire (important pour les accès DMA, Direct Memory Access, qui court-circuitent le microprocesseur)

Écriture dans le cache - bis

- **écriture retardée**
  - mise à jour du cache et marquage de la cellule modifiée
  - écriture dans la mémoire principale lors du remplacement du bloc dans lequel se trouve la cellule modifiée
  - efficace, on peut écrire plusieurs fois au même endroit sans passer par la mémoire principale
  - entre 10 et 30% des accès mémoire sont des écritures

Systèmes de fichiers

- gestion efficace de l'espace disque
- choix et implantation d'un système de catalogue
- indépendance par rapport au matériel
- rapidité
- protection contre les erreurs et les accès non autorisés

Les disques magnétiques

- **structure**
  - plusieurs plateaux tournant à haute vitesse (3600 tr/min)
  - divisés en pistes concentriques (cylindres)
  - divisées en secteurs
  - bloc (unité d'allocation) = un ensemble de secteurs contigus
- **exemples :**

taille (Mo)	cylindres	têtes	secteurs
4 335	8960	15	63
	527	255	63
408	899	15	62

Formatage et fragmentation

- **formatage**
  - détermination de la taille des blocs
  - table d'allocation des fichiers
    - –MS-DOS : chaînage des blocs d'un fichier
    - –UNIX : une table de blocs par fichier
- **fragmentation :** quand les blocs ne sont pas contigus
- **différences suivant les systèmes**
  - **MS-DOS :** allocation des blocs aux fichiers dans l'ordre de disponibilité ⇒ fragmentation
  - **UNIX :** allocation d'ensembles contigus de blocs en fonction de la taille du fichier ⇒ peu de fragmentation

Organisation des fichiers

- chaque fichier a un descripteur
  - table des descripteurs
- présentation sous forme d'une arborescence
  - un répertoire est un fichier
  - un lien est un fichier
- informations contenues dans le descripteur
  - nom symbolique
  - taille du fichier
  - éventuellement table des blocs
  - informations de protection (droits de lecture, écriture ou exécution pour les différents utilisateurs)
  - dates de création, dernière modification ou dernier accès

Gestion avancée

- performances
  ⇒ cache disque
- masquage de la structure des disques
  ⇒ une arborescence unifiée (montage à la UNIX)
- partage de fichiers en réseau
  ⇒ serveurs de fichiers et systèmes de fichiers répartis
- sécurité des données
  ⇒ sauvegardes, redondance des informations

Entrées/Sorties

(on note aussi E/S ou In/Out ou I/O)

- but
- moyen
- périphériques standards
- cartes filles - cartes propriétaires
- sorties parallèles et séries
- IRQ et DMA
- connecteur SCSI et USB

But

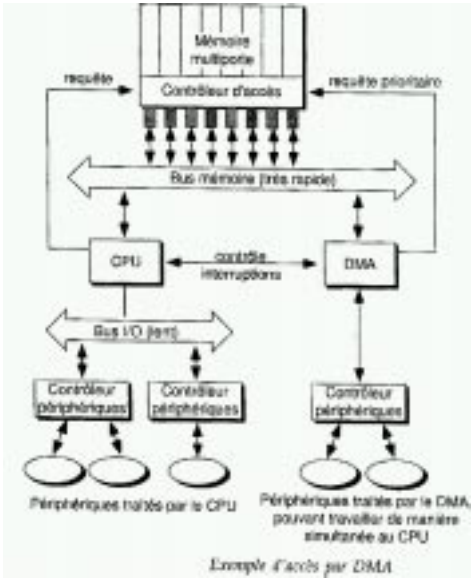
- dispositif de communication avec l'extérieur (de l'ordinateur)
  - l'humain
  - des capteurs et des machines
  - d'autres ordinateurs
- c'est à dire, d'une façon générale
  - recevoir des données (numériques)
  - fournir des résultats (numériques)

Périphériques standard

- entrées :
  - clavier
  - souris
  - manette de jeu
  - scanner
  - micro
  - caméra vidéo
- sorties :
  - écran
  - imprimante
  - enceinte acoustique
- entrées + sorties :
  - disques (internes ou externes)
  - modem/carte réseau

Controleur de périphériques

- liaison bus/périphérique
- situé sur :
  - la carte mère
  - une carte (fille) insérée dans un slot d'extension
  - elles possèdent de plus en plus leur processeur spécialisé
- communique avec le processeur par :
  - interruptions (IRQ)
  - accès direct à la mémoire (DMA)
  - canal d'entrée/sortie
  - problème de conflit matériel
  - solution Plug & Play



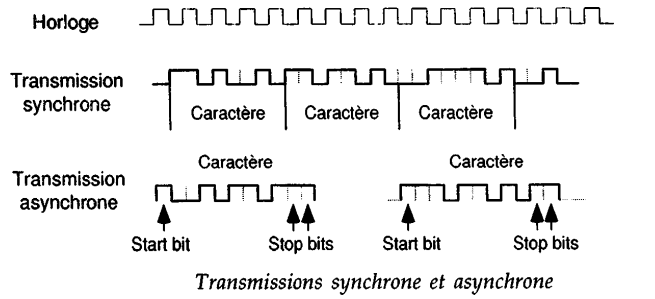
Exemple d'accès par DMA

Controleur de périphériques - bis

- moyen :
  - connecteur interne
  - connecteur externe
    - spécifique
    - série et parallèle
    - SCSI ou USB

Série et parallèle

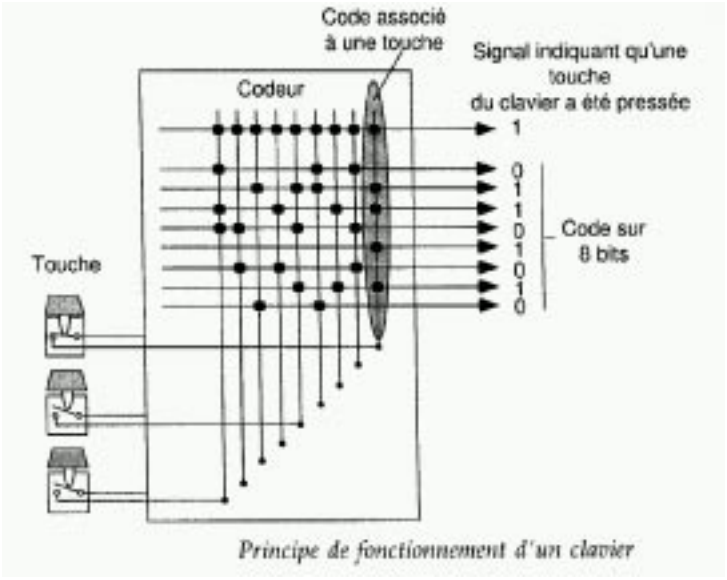
- parallèle (actuellement bi-directionnel)
  - connecteur DB25 femelle
  - 8 broches pour les données aller
  - 8 broches pour les données retour
  - broches de contrôle et d'alimentation
- série
  - connecteur DB25 ou DB9 male
  - 1 broche pour les données aller
  - 1 broche pour les données retour
  - broches de contrôle et d'alimentation
- transmission synchrone
- transmission asynchrone



Transmissions synchrone et asynchrone

SCSI ou USB

- SCSI
  - plutôt réservé aux professionnels (coût)
  - jusqu'à 16 phériphériques chaînés
- USB
  - sur tout les ordinateurs actuels
  - des périphériques de plus en plus nombreux
  - jusqu'a 127 (avec un concentrateur)
  - les périphériques peuvent être alimentés par le connecteur
  - les périphériques peuvent être (dé)connectés alimentés
  - Plug & play



Principe de fonctionnement d'un clavier

Carte vidéo et écran

- **définition :**
  - bande passante = vitesse d'un pixel × 1.5 (surcharge de travail)
  - vitesse d'un pixel = nombre de pixels horizontaux × nombre de pixels verticaux × taux de rafraîchissement
- exemple :
  - vitesse d'un pixel = 1024 (H) × 768 (V) × 72 = 56,623,104 Hz
  - bande passante = 56,623,104 × 1.5 = 84,934,656 Hz
- processeur d'acceleration 2D / 3D

Multimédia

www.Mcours.com
Site N°1 des Cours et Exercices Email: contact@mcours.com

- carte son
  - digitalisation
  - synthèse
  - processeur son 3D
- appareil photo numérique
  - les résolutions atteignent 1280×1024 pixels
- caméra
  - pour visio-conférence
  - webcam
  - prix abordable

Imprimantes

- NB ou couleur
- différentes technologies :
  - aiguilles
  - marguerite
  - laser
  - jet d'encre
  - thermique
- processeur interne
  - langages Postscript, ESC/P2, PCL5, ...