Chapitre 1 : Introduction à la technologie des ordinateurs

1. DEFINITIONS

1.1. L'informatique

L'informatique est la science du traitement automatique d l'information.
Comme les autres disciplines l'informatique présente des options à savoir :

- Architecture des ordinateurs
- Programmation
- Génie Logiciel
- Réseaux Informatiques
- Etc.

1.2. L'information

1.2.1. Représentation de l'information

L'information peut être représenter sous forme de :

-image -dessin
-son -texte

1.2.2. Définition

L'information est le support des connaissances humaine dans tous les domaines (scientifique, technique, économique et social).
Ensemble des données et des instructions.

Donnée :

Représente un ou plusieurs renseignements, elle peut être :

- Numérique (nombre, chiffre, ...)
- Logique (vrai, faux)
- Textuelle (alphabet, mot, phrase, texte, ...)
- Graphique (schéma, dessin, organigramme, ...)
- Son, image, vidéo, etc.
- Constante (celerite, π, charge d'électron ,...)
- Variable (température, position des planètes / soleil, ..)
- Etc.

Instruction :

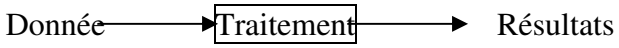
Ecrite en langage machine, l'instruction représente les opérations effectuées par un ordinateur (addition, division, tri, recherche, sauvegarde, affectation...).

1.3. Le traitement

1.3.1. Définition

Le traitement est l'ensemble des opérations effectuées sur des données afin de les rendre utilisables (résultats)
Exemples :

1.3.2. Schéma du traitement



1.4. Le système informatique

Un système informatique est défini comme étant composé de deux parties essentielles:

- Matériel (hardware)
- Logiciel (software)

Le matériel : imprimante, scanner..
Le logiciel est abstrait et il fait fonctionner le matériel

2. PREMIERE APPROCHE DU SYSTEME INFORMATIQUE

Pour présenter brièvement ce que représente un ordinateur, nous allons établir un parallèle avec la façon dont travaille, par exemple, un employé de bureau.

Pour travailler, l'employé qui est au centre des traitements, et que l'on peut donc considérer comme l'unité centrale de traitement, s'installe à son bureau où on lui remet dans un dossier le travail à faire (informations en entrée), tandis que dans un autre dossier il devra mettre le travail fait (informations en sortie).

Nous avons donc là un système de traitement à qui l'on remet des informations en entrée, qui exécute un traitement sur ces informations et qui restitue enfin des informations en sortie.

Pour exécuter son travail, l'employé peut avoir besoin de réaliser des calculs (opérations mathématiques, additions, comparaisons, …), il dispose pour cela d'une calculatrice (ou unité de calcul = Microprocesseur ou CPU = Central Processing Unit). Pour ne pas oublier ce qu'on lui demande, l'employé notera sur un brouillon les instructions qu'il a reçues et qui constituent son programme de travail, de même il va certainement être amené à noter quelques informations sur les données qu'il va traiter.

Ceci constitue une mémorisation des instructions à exécuter et des données à traiter. Cette information peut être actualisée et on peut donc considérer que l'on se trouve en présence d'une mémoire qui « vit » (où l'information naît, vit et meurt). On parle alors de mémoire vive = RAM = Random Acess Memory.

De plus, il se peut que pour réaliser la tâche demandée, l'employé ait besoin d'informations qui soient toujours les mêmes (le taux de T.V.A. par exemple) et qu'il mémorisera une fois pour toutes. Il s'agit là d'une mémoire qui ne « vit pas » et l'on parle alors de mémoire morte = ROM = Read Only Memory.

Certaines informations (telles que le catalogue des prix, les adresses des clients, …) sont trop volumineuses pour être mémorisées de tête, et l'employé aura alors à sa disposition des classeurs ou des bacs à fiches « dossiers » contenant des fichiers clients ou tarifs, qui constituent en fait une mémorisation externe ou auxiliaire = mémoire de masse = (Disque dur, CD-ROM, DVD-ROM, Graveur, Floppy « lecteur de disquettes », Lecteur ZIP/JAZ, etc…) de certaines données.

Commandes Mémoire vive : instructions + données

Unité de calculs Mémoire morte

Mémoire auxiliaire (Fichiers (

L'ordinateur tel que nous le rencontrons le plus souvent dans la vie courante se présente généralement sous l'aspect du micro-ordinateur. Faisons preuve de curiosité et « levons le capot ! ». Nous pouvons

d'ores et déjà distinguer un certain nombre d'éléments. Le plus imposant et le plus lourd est le bloc d'alimentation électrique, souvent accompagné d'un ventilateur de refroidissement.

Partant de ce bloc, des câbles électriques alimentent les divers composants parmi lesquels on distingue particulièrement un certain nombre de plaques, appelées cartes ou contrôleurs, recouvertes de puces électroniques et enfichées sur une carte plus grande, dite carte mère et qui est située au fond de la machine.

Cette carte mère comporte l'unité de traitement, plus communément désignée sous le terme de microprocesseur. Comme son nom l'indique, cette carte mère accueille des cartes filles ou d'extensions qui possèdent des fonctions diverses (afficher la vidéo, gérer le son, gérer un réseau coté terminal, etc …)

Entrées Sorties

Cette carte mère accueille également la mémoire vive et la mémoire morte.

Pour relier à l'intérieur du boîtier un certain nombre d'unités de stockage (mémoire auxiliaire), on utilise des nappes de câbles qui renvoient vers des modules situés à l'avant de la machine. Il est également possible de rencontrer ces diverses unités de stockage à l'extérieur du boîtier principal et vous pouvez déjà constater que l'on relie à ce boîtier l'écran de visualisation (ou moniteur vidéo) ainsi que le clavier, qui permet d'entrer les informations nécessaires au traitement, ou encore la souris.

3. Historique de l'ordinateur

3.1. Avancées technologiques

- Génération 0 : 17ème siècle à 1945 (Calculateurs mécaniques)
- Première génération : 1945 – 1955
    - Tubes à vide
    - Premiers calculateurs électroniques
    - Ex: ENIAC
- Seconde génération : 1955 – 1965
    - Transistors remplacent les tubes à vides
    - Premières séries commerciales d'ordinateurs
- Troisième génération : 1965 – 1980
    - Circuits intégrés : permettent de placer un nombre important de transistors sur une même puce de silicium
    - Début de la montée en puissance et de la miniaturisation
    - 1971 : Intel 4004
    - Première unité de calcul (sur 4 bits) intégrée entièrement sur une seule puce
    - Premier micro-processeur
- Quatrième génération : 1980 à aujourd'hui
    - VLSI : *Very Large Scale Integration*
    - Intégration de millions de transistors sur une
    - même puce
    - Toujours plus de puissance et de miniaturisation

- à un coût toujours moindre

- Cinquième génération : ??
  - ??

4. Données non numériques

Les données non numériques correspondent aux caractères alphanumériques,

- a.b.c…y,z (26)
- A B C…Z (26) ;
- 0,1,2…9 (10) ;
- opérations :+,-,*,/,<,>…= (15) ;
- ponctuation : . , ;? « { } [ ] ( ) _ ..(20) ;
- caractères spéciaux :$ & @...(10) ;
- signe invisible : retour a la ligne, espace..(5)

au total environ 112

Pour associer a chaque de ces caractères une suite distincte de 0 et 1 qui est le code, nous avons donc au minimum besoin de 7 bits car 2^7=128, c'est précisément ce que réalise la premier version du code ASCII(**American Standard Code for Information Interchange**)

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.asciitable.com

128	Ç	144	É	161	í	177	▒	193	┴	209	╤	225	ß	241	±
129	ü	145	æ	162	ó	178	▓	194	┬	210	╥	226	Γ	242	≥
130	é	146	Æ	163	ú	179	│	195	├	211	╙	227	π	243	≤
131	â	147	ô	164	ñ	180	┤	196	─	212	╘	228	Σ	244	⌠
132	ä	148	ö	165	Ñ	181	╡	197	┼	213	╒	229	σ	245	⌡
133	à	149	ò	166	ª	182	╢	198	╞	214	╓	230	µ	246	÷
134	å	150	û	167	º	183	╖	199	╟	215	╫	231	τ	247	≈
135	ç	151	ù	168	¿	184	╕	200	╚	216	╪	232	Φ	248	°
136	ê	152	ÿ	169	⌐	185	╣	201	╔	217	┘	233	Θ	249	∙
137	ë	153	Ö	170	¬	186	║	202	╩	218	┌	234	Ω	250	·
138	è	154	Ü	171	½	187	╗	203	╦	219	█	235	δ	251	√
139	ï	156	£	172	¼	188	╝	204	╠	220	▄	236	∞	252	ⁿ
140	î	157	¥	173	¡	189	╜	205	═	221	▌	237	ϕ	253	²
141	ì	158	₧	174	«	190	╛	206	╬	222	▐	238	ε	254	■
142	Ä	159	ƒ	175	»	191	┐	207	╧	223	▀	239	∩	255	
143	Å	160	á	176	░	192	└	208	╨	224	α	240	≡		

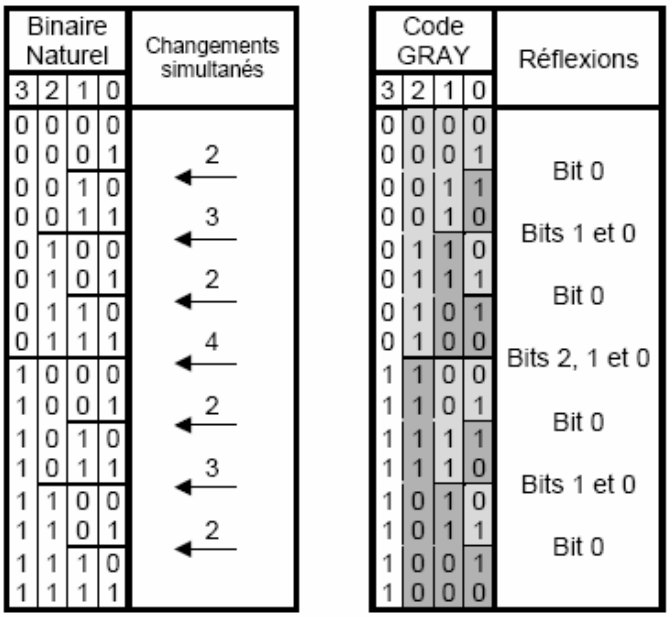
Source: www.LookupTables.com

L'opération permettant le passage d'un code A à un code B est appelée transcodage.
Parmi les plus connus, on cite les codes :

- BCD(6bits),
- UNICODE-ISO/IEC 10646(16 bits puis 32)
- code ASCII (7 bits 128 caractères) code utilisé par la majorité des ordinateurs > code DOS.
- code ANIS (255 caractères) code utilisé sous Windows et basé sur ASCII, les 128 premiers caractères sont les mêmes que pour ASCII, les autres constituent une extension pour Windows.
- Code UNICODE-ISO/IEC 10646(16 bits 65535 caractères puis 32bits) code universel pour les signes particuliers (langues étrangères comme le chinois)

Le code GRAY.

Le code GRAY pallie efficacement à l'un des plus gros problèmes de l'électronique : la non-simultanéité.
Ainsi, si l'on récapitule les 16 premiers nombres du binaire naturel, on se rend compte que des changements simultanés apparaissent à foison ! Le code **GRAY** présenté à coté est lui par contre tout à fait libre de ces défauts.



Comme présenté par le tableau, le code GRAY est réalisé par des réflexions successives. Cela lui permet de ne faire varier pour passer d'un nombre à l'autre qu'un seul bit. Ce système de comptage est utilisé pour certains calculs en binaire que nous étudierons plus loin. Il sert aussi à certains systèmes de positionnement par systèmes optiques.

5. Systèmes de numération

Les systèmes informatiques actuels étant construits à l'aide de circuits intégrés (composants électroniques qui rassemblent sur une puce de silicium quelques millions de transistors), ils ne peuvent fonctionner que selon une logique à deux états telle que, de façon schématique, le courant passe ou ne passe pas dans le transistor. Ces deux états logiques, conventionnellement notés 0 et 1, déterminent une logique dite binaire qui correspond à deux niveaux électriques que nous donnerons, pour simplifier, comme étant équivalents à 0 volt et à + 5 volts.

ETAT 0 : Le courant ne passe pas => Interrupteur Ouvert
ETAT 1 : Le courant passe => Interrupteur Fermé

Toute information à traiter devra donc pouvoir être représentée sous une forme assimilable par la machine, et donc sous une forme binaire. Que ce soit en interne dans la machine, ainsi que nous venons de la signaler, mais également sur les « fils » permettant de faire circuler l'information entre tous les composants d'un ordinateur.
Le passage d'une information, d'un langage compréhensible par l'homme à un langage compréhensible par le système informatique, s'appelle codage ou codification. Nous verrons qu'il existe de nombreuses possibilités de codage de l'information, BINAIRE, HEXADICEMAL, BCD, ASCII, … Mais étudions d'abord les éléments de base du langage binaire.

5.1. Principe d'une base

La base est le nombre qui sert à définir un système de numération. La base du système décimal est dix alors que celle du système octal est huit.
Quelque soit la base numérique employée, elle suit la relation suivante :

$$\sum_{i=0}^{i=n}(b_i a^i) = b_i a^n + \ldots\ldots + b_5 a^5 + b_4 a^4 + b_3 a^3 + b_2 a^2 + b_1 a^1 + b_0 a^0$$

ou : $\mathbf{b}_i$: chiffre de la base de rang i
et : $\mathbf{a}^i$: puissance de la base a d'exposant de rang i

Exemple : base 10

$1986 = (1 \times 10^3) + (9 \times 10^2) + (8 \times 10^1) + (6 \times 10^0)$

Le système décimal est celui dans lequel nous avons le plus l'habitude d'écrire.
Chaque chiffre peut avoir 10 valeurs différentes : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, de ce fait, le système décimal a pour **base** 10.
Tout nombre écrit dans le système décimal vérifie la relation suivante :

$745 = 7 \times 100 + 4 \times 10 + 5 \times 1$
$745 = 7 \times 10 \times 10 + 4 \times 10 + 5 \times 1$
$745 = 7 \times 10^2 + 4 \times 10^1 + 5 \times 10^0$

Chaque chiffre du nombre est à multiplier par une puissance de 10 : c'est ce que l'on nomme le **poids du chiffre**.

L'exposant de cette puissance est nul pour le chiffre situé le plus à droite et s'accroît d'une unité pour chaque passage à un chiffre vers la gauche.

$12\,435 = 1 \times 10^4 + 2 \times 10^3 + 4 \times 10^2 + 3 \times 10^1 + 5 \times 10^0$.

Cette façon d'écrire les nombres est appelée **système de numération de position**.

Dans notre système conventionnel, nous utilisons les puissances de 10 pour **pondérer** la valeur des chiffres selon leur position, cependant il est possible d'imaginer d'autres systèmes de nombres ayant comme base un nombre entier différent.

5.2.Le système octal

Le système octal utilise un système de numération ayant comme base 8 (octal => latin octo = huit).
Il faut noter que dans ce système nous n'aurons plus 10 symboles mais 8 seulement :0, 1, 2, 3, 4, 5, 6, 7Ainsi, un nombre exprimé en base 8 pourra se présenter de la manière suivante :

$(745)_8$

Lorsque l'on écrit un nombre, il faudra bien préciser la base dans laquelle on l'exprime pour lever les éventuelles indéterminations (745 existe aussi en base 10).Ainsi le nombre sera mis entre parenthèses (745 dans notre exemple) et indicé d'un nombre représentant sa base (8 est mis en indice).

Cette base obéira aux même règles que la base 10, vue précédemment, ainsi on peut décomposer $(745)_8$ de la façon suivante :
$(745)_8 = 7 \times 8^2 + 4 \times 8^1 + 5 \times 8^0$
$(745)_8 = 7 \times 64 + 4 \times 8 + 5 \times 1$

$(745)_8 = 448 + 32 + 5$

Nous venons de voir que : **$(745)_8 = (485)_{10}$**

5.3.Le système binaire

Dans le système binaire , chaque chiffre peut avoir 2 valeurs différentes : 0, 1.
De ce fait, le système a pour base 2.
Tout nombre écrit dans ce système vérifie la relation suivante :

$(10\ 110)_2 = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$
$(10\ 110)_2 = 1 \times 16 + 0 \times 8 + 1 \times 4 + 1 \times 2 + 0 \times 1$
donc : $(10110)_2 = (22)_{10}$.

Tous les systèmes de numération de position obéissent aux règles que nous venons de voir.

Tableau récapitulatif

Système décimale	Système octal	Système binaire
0	0	0
1	1	1
2	2	10
3	3	11
4	4	100
5	5	101
6	6	110
7	7	111
8	10	1000
9	11	1001
10	12	1010
11	13	1011
12	14	1100
13	15	1101
14	16	1110
15	17	1111
16	20	10000
17	21	10001
-	-	-
-	-	-
63	77	111111
64	100	1000000
65	101	1000001

5.4.Le système hexadécimal

Le système hexadécimal utilise les 16 symboles suivant :
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

De ce fait, le système a pour base 16.
Un nombre exprimé en base 16 pourra se présenter de la manière suivante : $(5AF)_{16}$
La correspondance entre base 2, base 10 et base 16 est indiquée dans le tableau ci-après

Base 10	Base 16	Base 2
0	0	0
1	1	1
2	2	10
3	3	11
4	4	100
5	5	101
6	6	110
7	7	111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Le nombre $(5AF)_{16}$ peut se décomposer comme suit :

$(5AF)_{16} = 5 \times 16^2 + A \times 16^1 + F \times 16^0$

En remplaçant A et F par leur équivalent en base 10, on obtient :

$(5AF)_{16} = 5 \times 16^2 + 10 \times 16^1 + 15 \times 16^0$
$(5AF)_{16} = 5 \times 256 + 10 \times 16 + 15 \times 1$

donc = $(5AF)_{16} = (1455)_{10}$

6. Conversion d'un nombre de base quelconque en nombre décimal

En exposant les principes des **systèmes de numération** de position, nous avons déjà vu comment convertir les nombres de base 8, base 2 et base 16 en nombres décimaux.

6.1. Conversion d'un nombre décimal en nombre binaire

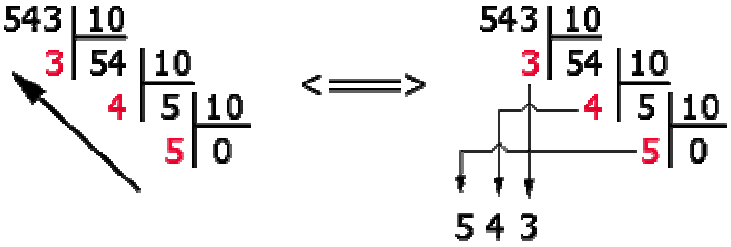
Pour expliquer ce type de conversion, on peut revenir sur le système décimal. Si nous divisons le nombre $(543)_{10}$ par 10, nous obtenons comme quotient 54 et 3 comme reste. Cela signifie que ce nombre équivaut à :

(54 x 10) + 3

Le reste 3 est le chiffre indiquant le nombre d'unités.

En redivisant ce quotient (54) par 10, nous obtenons 5 comme deuxième quotient et 4 comme reste. Ce reste donne le deuxième chiffre du nombre, donc celui des dizaines.

Enfin, si l'on divise ce deuxième quotient par 10, nous obtenons O et il restera 5 qui représentera le chiffre des centaines.



Résumer du principe de conversion

En divisant successivement un nombre par la base (10) et en ne **conservant que les restes**, on a réussi à exprimer le nombre par des chiffres inférieurs de 10. Mais attention, il faut **lire les restes de bas en haut**.

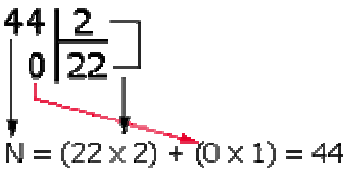
6.2. Conversion binaire

Maintenant si nous divisons un nombre décimal par 2, le quotient indique le nombre de fois que 2 est contenu dans ce nombre et le reste indique le chiffre des unités dans l'expression du nombre binaire.

Soit N le nombre, Q1 le quotient et R1 le reste, nous avons :

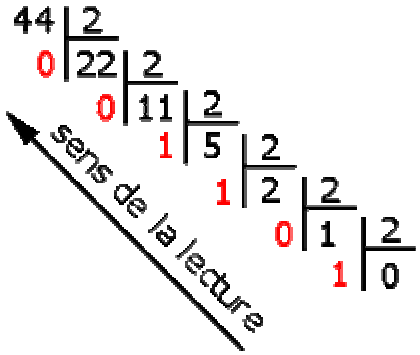
$N = (Q1 \times 2) + (R1 \times 1)$
$N = (Q1 \times 2^1) + (R1 \times 2^0)$

Exemple :



soit :
$N = (22 \times 2) + (0 \times 1) = 44$

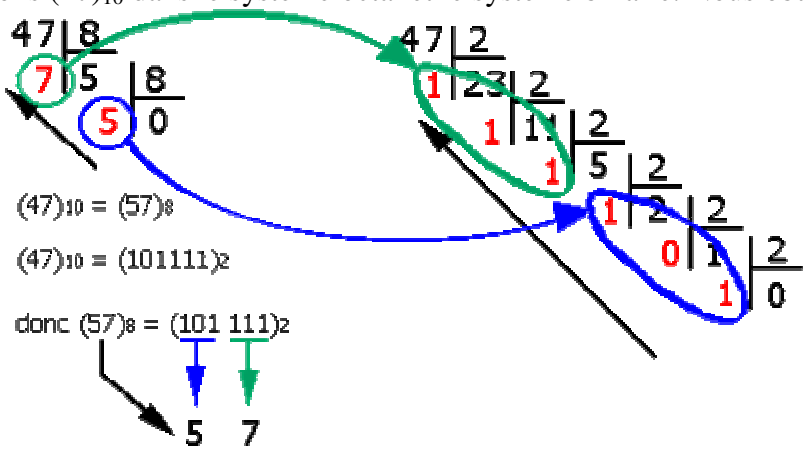
Pour obtenir l'expression binaire d'un nombre exprimé en décimal, il suffit de **diviser successivement ce nombre par 2** jusqu'à ce que le quotient obtenu soit égal à O. Comme pour la conversion dans le système décimal les restes de ces divisions lus de bas en haut représentent le nombre binaire.



$(44)_{10} = (101100)_2$

6.3. Relation entre les nombres binaires et les nombres octaux

Exprimons $(47)_{10}$ dans le système octal et le système binaire. Nous obtenons :



Nous pouvons remarquer qu'après 3 divisions en binaire nous avons le même quotient qu'après une seule en octal. De plus le premier reste en octal obtenu peut être mis en relation directe avec les trois premiers restes en binaire :

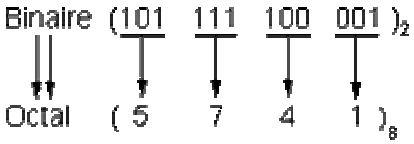
$(111)_2 = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
$(111)_2 = 1 \times 4 + 1 \times 2 + 1 \times 1$
$(111)_2 = (7)_8$

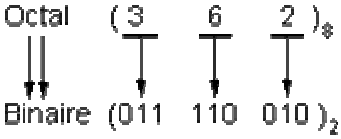
et il en est de même pour le caractère octal suivant :

$(101)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
$(101)_2 = 1 \times 4 + 0 \times 2 + 1 \times 1$
$(101)_2 = (5)_8$

Cette propriété d'équivalence entre chaque chiffre octal et chaque groupe de **3 chiffres binaires** permet de passer facilement d'un système à base 8 à un système à base 2 et vice versa.

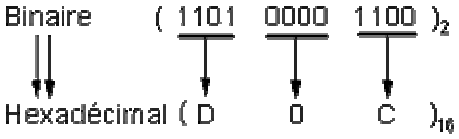
Exemple de conversion binaire octal et octal binaire :

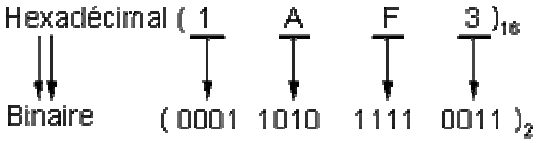




6.4.Relation entre les nombres binaires et les nombres hexadécimaux

La propriété d'équivalence que nous venons de voir entre le binaire et l'octal existe entre l'hexadécimal et le binaire. La seule différence est qu'il faut exprimer chaque caractère hexadécimal à l'aide de **4 informations** binaires.





6.5.OPERATIONS BINAIRES

Les opérations sur les nombres binaires s'effectuent de la même façon que sur les nombres décimaux.
Toutefois il ne faut pas oublier que les seuls symboles utilisés sont le 1 et le 0, nous aurons ainsi les opérations fondamentales suivantes.

6.5.1. Addition

0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0 et « on retient 1 »

6.5.1.1. Soustraction

0 – 0 = 0
0 – 1 = 1 et « on retient 1 »
1 – 1 = 0 et 1 – 0 = 1

6.5.1.2. Multiplication

Les multiplications binaires s'effectuent selon le principe des multiplications décimales. On multiplie donc le multiplicande par chacun des bits du multiplicateur. On décale les résultats intermédiaires obtenus et on effectue ensuite l'addition de ces résultats partiels.

0 * 0 = 0
0 * 1 = 0
1 * 0 = 0
1 * 1 = 1

6.5.1.3. Division

Nous avons vu que la multiplication était basée sur une succession d'additions. Inversement la division va être basée sur une succession de soustractions et s'emploie de la même façon qu'une division décimale ordinaire.
Soit 11002 à diviser par 1002

1100 / 100 = 11

Le résultat qui se lit de haut en bas « descendant » est donc 112
Soit 1011002 à diviser par 1002

101100 / 100 = 10112

Le résultat qui se lit en « descendant » est donc 10112

6.5.1.4. complément à 1

6.5.1.5. complément à 2

6.5.1.5.1. les porte logiques

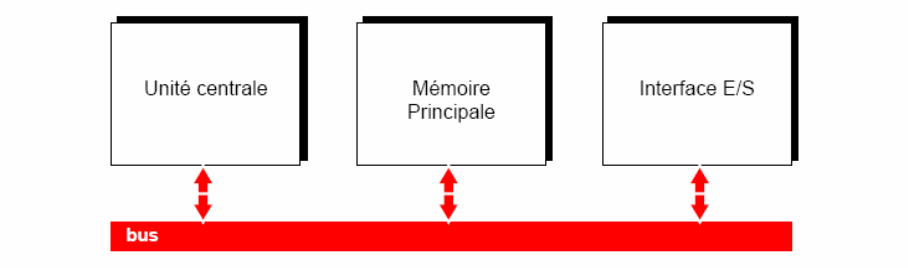
7. Architecture de base

7.1. Modèle de Von Neumann

Pour traiter une information, un microprocesseur seul ne suffit pas, il faut l'insérer au sein d'un système minimum de traitement programmé de l'information. John Von Neumann est à l'origine d'un modèle de machine universelle de traitement programmé de l'information (1946). Cette architecture sert de base à la plupart des systèmes à microprocesseur actuel. Elle est composée des éléments suivants :

- une unité centrale
- une mémoire principale
- des interfaces d'entrées/sorties

Les différents organes du système sont reliés par des voies de communication appelées **bus**.



7.2.L'unité centrale

Elle est composée par le microprocesseur qui est chargé d'interpréter et d'exécuter les instructions d'un programme, de lire ou de sauvegarder les résultats dans la mémoire et de communiquer avec les unités d'échange. Toutes les activités du microprocesseur sont cadencées par une horloge.

On caractérise le microprocesseur par :

- sa fréquence d'horloge : en MHz ou GHz
- le nombre d'instructions par secondes qu'il est capable d'exécuter : en MIPS
- la taille des données qu'il est capable de traiter : en bits

7.3. La mémoire principale

Elle contient les instructions du ou des programmes en cours d'exécution et les données associées à ce programme. Physiquement, elle se décompose souvent en :

- une mémoire morte (**ROM** = Read Only Memory) chargée de stocker le programme. C'est une mémoire à lecture seule.
- une mémoire vive (**RAM** = Random Access Memory) chargée de stocker les données intermédiaires ou les résultats de calculs. On peut lire ou écrire des données dedans, ces données sont perdues à la mise hors tension.

Remarque :

Les disques durs, disquettes, CDROM, etc… sont des périphériques de stockage et sont considérés comme des mémoires secondaires.

7.4. Les interfaces d'entrées/sorties

Elles permettent d'assurer la communication entre le microprocesseur et les périphériques. (capteur, clavier, moniteur ou afficheur, imprimante, modem, etc…).

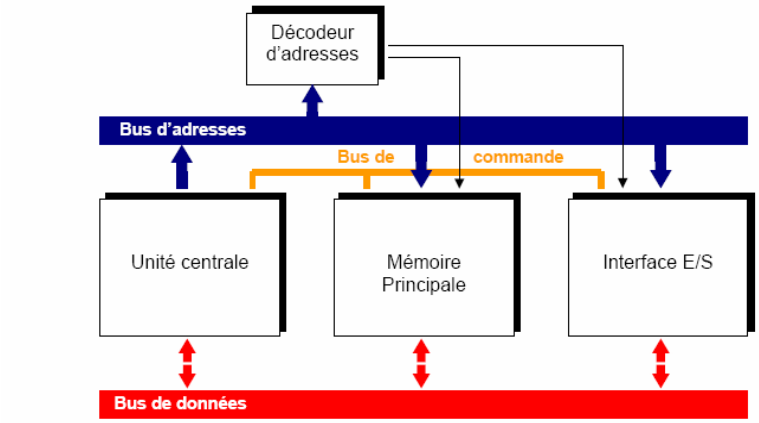
7.5. Les bus

Un bus est un ensemble de fils qui assure la transmission du même type d'information. On retrouve trois types de bus véhiculant des informations en parallèle dans un système de traitement programmé de l'information :

- **un bus de données** : bidirectionnel qui assure le transfert des informations entre le microprocesseur et son environnement, et inversement. Son nombre de lignes est égal à la capacité de traitement du microprocesseur.
- **un bus d'adresses**: unidirectionnel qui permet la sélection des informations à traiter dans un espace mémoire (ou espace adressable) qui peut avoir 2^n emplacements, avec n = nombre de conducteurs du bus d'adresses.
- **un bus de commande:** constitué par quelques conducteurs qui assurent la synchronisation des flux d'informations sur les bus des données et des adresses.

7.6.Décodage d'adresses

La multiplication des périphériques autour du microprocesseur oblige la présence d'un **décodeur d'adresse** chargé d'aiguiller les données présentes sur le bus de données.
En effet, le microprocesseur peut communiquer avec les différentes mémoires et les différents boîtiers d'interface. Ceux-ci sont tous reliés sur le même bus de données et afin d'éviter des conflits, un seul composant doit être sélectionné à la fois.
Lorsqu'on réalise un système microprogrammé, on attribue donc à chaque périphérique une zone d'adresse et une fonction « décodage d'adresse » est donc nécessaire afin de fournir les signaux de sélection de chacun des composants.



Remarque : lorsqu'un composant n'est pas sélectionné, ses sorties sont mises à l'état « haute impédance » afin de ne pas perturber les données circulant sur le bus. (Elle présente une impédance de sortie très élevée = circuit ouvert).

8. Les mémoires

Une mémoire est un circuit à semi-conducteur permettant d'enregistrer, de conserver et de restituer des informations (instructions et variables). C'est cette capacité de mémorisation qui explique la polyvalence des systèmes numériques et leur adaptabilité à de nombreuses situations. Les informations peuvent être écrites ou lues. Il y a écriture lorsqu'on enregistre

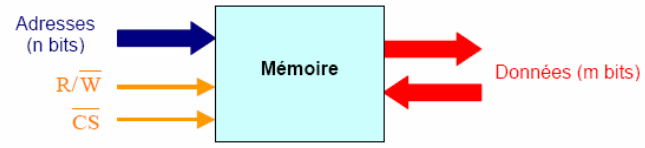
des informations en mémoire, lecture lorsqu'on récupère des informations précédemment enregistrées.

8.1.Organisation d'une mémoire

Une mémoire peut être représentée comme une armoire de rangement constituée de différents tiroirs. Chaque tiroir représente alors une case mémoire qui peut contenir un seul élément : des **données**. Le nombre de cases mémoires pouvant être très élevé, il est alors nécessaire de pouvoir les identifier par un numéro. Ce numéro est appelé **adresse**. Chaque donnée devient alors accessible grâce à son adresse

Adresse	Case mémoire
7 = 111	
6 = 110	
5 = 101	
4 = 100	
3 = 011	
2 = 010	
1 = 001	
0 = 000	0001 1010

Avec une adresse de *n* bits il est possible de référencer au plus 2^n cases mémoire. Chaque case est remplie par un mot de données (sa longueur *m* est toujours une puissance de 2). Le nombre de fils d'adresses d'un boîtier mémoire définit donc le nombre de cases mémoire que comprend le boîtier. Le nombre de fils de données définit la taille des données que l'on peut sauvegarder dans chaque case mémoire.
En plus du bus d'adresses et du bus de données, un boîtier mémoire comprend une entrée de commande qui permet de définir le type d'action que l'on effectue avec la mémoire (lecture/écriture) et une entrée de sélection qui permet de mettre les entrées/sorties du boîtier en haute impédance.
On peut donc schématiser un circuit mémoire par la figure suivante où l'on peut distinguer :



- les entrées d'adresses
- les entrées de données
- les sorties de données
- les entrées de commandes :
  - une entrée de sélection de lecture ou d'écriture. (R/W)
  - une entrée de sélection du circuit. (CS)

Une opération de lecture ou d'écriture de la mémoire suit toujours le même cycle :

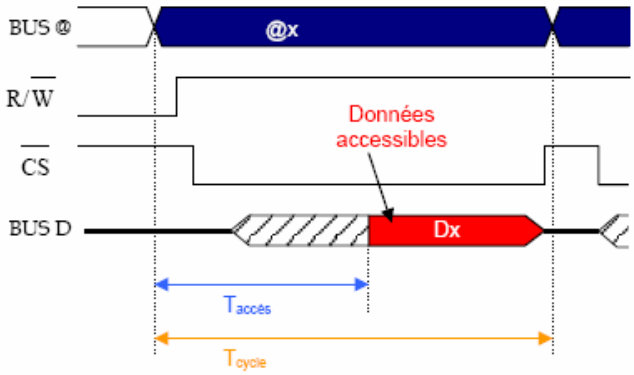
1. sélection de l'adresse
2. choix de l'opération à effectuer (R/W)
3. sélection de la mémoire (CS = 0)
4. lecture ou écriture la donnée

Remarque :
Les entrées et sorties de données sont très souvent regroupées sur des bornes bidirectionnelles.

8.2. Caractéristiques d'une mémoire

- **La capacité :** c'est le nombre total de bits que contient la mémoire. Elle s'exprime aussi souvent en octet.
- **Le format des données :** c'est le nombre de bits que l'on peut mémoriser par case mémoire.
- On dit aussi que c'est la largeur du mot mémorisable.
- **Le temps d'accès :** c'est le temps qui s'écoule entre l'instant où a été lancée une opération de lecture/écriture en mémoire et l'instant où la première information est disponible sur le bus de données.
- **Le temps de cycle :** il représente l'intervalle minimum qui doit séparer deux demandes successives de lecture ou d'écriture.
- **Le débit :** c'est le nombre maximum d'informations lues ou écrites par seconde.
- **Volatilité :** elle caractérise la permanence des informations dans la mémoire. L'information stockée est volatile si elle risque d'être altérée par un défaut d'alimentation électrique et non volatile dans le cas contraire.

Exemple : Chronogramme d'un cycle de lecture



8.3. Différents types de mémoire

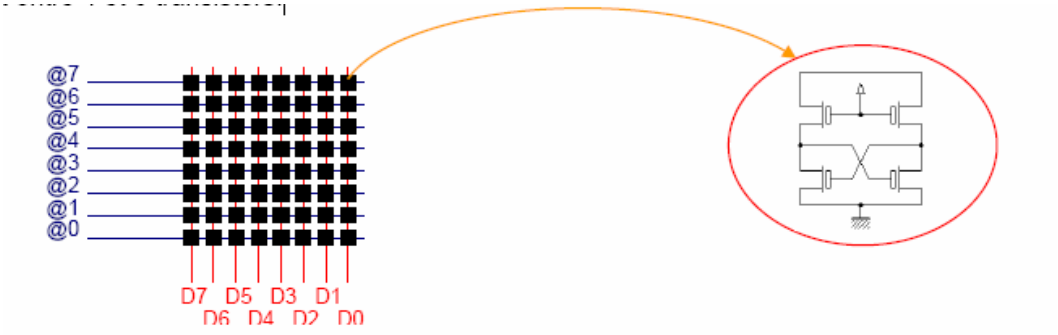
8.3.1. Les mémoires vives (RAM)

Une mémoire vive sert au stockage temporaire de données. Elle doit avoir un temps de cycle très court pour ne pas ralentir le microprocesseur. Les mémoires vives sont en général volatiles : elles perdent leurs informations en cas de coupure d'alimentation. Certaines d'entre elles, ayant une faible consommation, peuvent être rendues non volatiles par l'adjonction d'une batterie. Il existe deux grandes familles de mémoires RAM (Random Acces Memory : mémoire à accès aléatoire) :

- Les RAM statiques
- Les RAM dynamiques

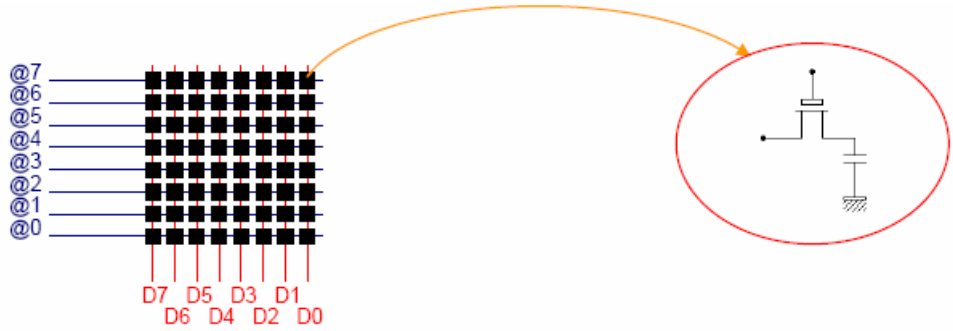
8.3.1.1. Les RAM statiques

Le bit mémoire d'une RAM statique (SRAM) est composé d'une bascule. Chaque bascule contient entre 4 et 6 transistors.



8.3.1.2. Les RAM dynamiques

Dans les RAM dynamiques (DRAM), l'information est mémorisée sous la forme d'une charge électrique stockée dans un condensateur (capacité grille substrat d'un transistor MOS).



Avantages :
Cette technique permet une plus grande densité d'intégration, car un point mémoire nécessite environ quatre fois moins de transistors que dans une mémoire statique. Sa consommation s'en retrouve donc aussi très réduite.

Inconvénients :
La présence de courants de fuite dans le condensateur contribue à sa décharge. Ainsi, l'information est perdue si on ne la régénère pas périodiquement (charge du condensateur). Les RAM dynamiques doivent donc être rafraîchies régulièrement pour entretenir la mémorisation : il s'agit de lire l'information et de la recharger. Ce rafraîchissement indispensable a plusieurs conséquences :

- il complique la gestion des mémoires dynamiques car il faut tenir compte des actions de rafraîchissement qui sont prioritaires.
- la durée de ces actions augmente le temps d'accès aux informations.

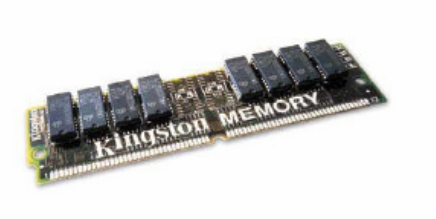
D'autre part, la lecture de l'information est destructive. En effet, elle se fait par décharge de la capacité du point mémoire lorsque celle-ci est chargée. Donc toute lecture doit être suivie d'une réécriture.

8.3.1.3. Conclusions

En général les mémoires dynamiques, qui offrent une plus grande densité d'information et un coût par bit plus faible, sont utilisées pour la mémoire centrale, alors que les mémoires statiques, plus rapides, sont utilisées lorsque le facteur vitesse est critique, notamment pour des mémoires de petite taille comme les caches et les registres.

Remarques :

Voici un historique de quelques DRAM qui ont ou sont utilisées dans les PC :



• **La DRAM FPM** (Fast Page Mode, 1987) : Elle permet d'accéder plus rapidement à des données en introduisant la notion de page mémoire. (33 à 50 Mhz)

• **La DRAM EDO** (Extended Data Out, 1995) : Les composants de cette mémoire permettent de conserver plus longtemps l'information, on peut donc ainsi espacer les cycles de rafraîchissement. Elle apporte aussi la possibilité d'anticiper sur le prochain cycle mémoire. (33 à 50 Mhz)

• **La DRAM BEDO** (Bursted EDO) : On n'adresse plus chaque unité de mémoire individuellement lorsqu'il faut y lire ou y écrire des données. On se contente de transmettre l'adresse de départ du processus de lecture/écriture et la longueur du bloc de données (Burst). Ce procédé permet de gagner beaucoup de temps, notamment avec les grands paquets de données tels qu'on en manipule avec les applications modernes. (66 Mhz)

• **La Synchronous DRAM** (SDRAM, 1997) : La mémoire SDRAM a pour particularité de se synchroniser sur une horloge. Les mémoires FPM, EDO étaient des mémoires asynchrones et elle induisaient des temps d'attentes lors de la synchronisation. Elle se compose en interne de deux bancs de mémoire et des données peuvent être lues alternativement sur l'un puis sur l'autre de ces bancs grâce à un procédé d'entrelacement spécial. Le protocole d'attente devient donc tout à fait inutile. Cela lui permet de supporter des fréquences plus élevées qu'avant (100 Mhz).

• **La DDR-I ou DDR-SDRAM** (Double Data Rate Synchronous DRAM, 2000) : La DDR-SDRAM permet de recevoir ou d'envoyer des données lors du front montant et du front descendant de l'horloge. (133 à 200 MHz)

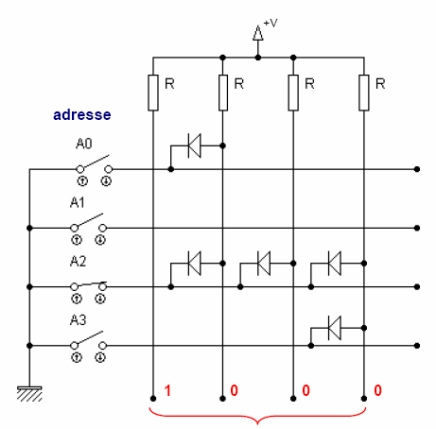
8.3.2. Les mémoires mortes (ROM)

Pour certaines applications, il est nécessaire de pouvoir conserver des informations de façon permanente même lorsque l'alimentation électrique est interrompue. On utilise alors des mémoires mortes ou mémoires à lecture seule (ROM : Read Only Memory). Ces mémoires sont non volatiles. Ces mémoires, contrairement aux RAM, ne peuvent être que lue. L'inscription en mémoire des données restent possible mais est appelée programmation. Suivant le type de ROM, la méthode de programmation changera. Il existe donc plusieurs types de ROM :

- ROM
- PROM
- EPROM
- EEPROM
- FLASH EPROM.

8.3.2.1. LA ROM

Elle est programmée par le fabricant et son contenu ne peut plus être ni modifié, ni effacé par l'utilisateur.



- **Structure :**

Cette mémoire est composée d'une matrice dont la programmation s'effectue en reliant les lignes aux colonnes par des diodes. L'adresse permet de sélectionner une ligne de la matrice et les données sont alors reçues sur les colonnes (le nombre de colonnes fixant la taille des mots mémoire).

- **Programmation :**

L'utilisateur doit fournir au constructeur un masque indiquant les emplacements des diode dans matrice.

- **Avantages :**
  - Densité élevée
  - Non volatile
  - Mémoire rapide
- **Inconvénients :**
  - Écriture impossible
  - Modification impossible (toute erreur est fatale).
  - Délai de fabrication (3 à 6 semaines)
  - Obligation de grandes quantités en raison du coût élevé qu'entraîne la production du masque et le
  - processus de fabrication.

8.3.2.2. La PROM

C'est une ROM qui peut être programmée une seule fois par l'utilisateur (Programmable ROM). La programmation est réalisée à partir d'un programmateur spécifique.

- **Structure :**

Les liaisons à diodes de la ROM sont remplacées par des fusibles pouvant être détruits ou des jonctions pouvant être court-circuitées.

- **Programmation :**

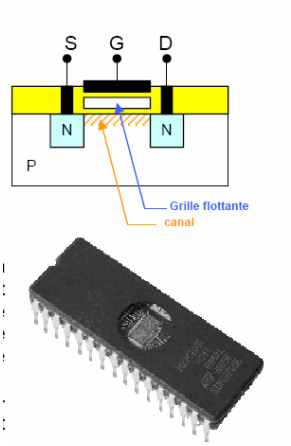
Les PROM à fusible sont livrées avec toutes les lignes connectées aux colonnes (0 en chaque point mémoire). Le processus de programmation consiste donc à programmer les emplacements des ''1'' en générant des impulsions de courants par l'intermédiaire du programmateur ; les fusibles situés aux points mémoires sélectionnés se retrouvant donc détruits.

Le principe est identique dans les PROM à jonctions sauf que les lignes et les colonnes sont déconnectées (1 en chaque point mémoire). Le processus de programmation consiste donc à programmer les emplacements des ''0'' en générant des impulsions de courants par l'intermédiaire du programmateur ; les jonctions situées aux points mémoires sélectionnés se retrouvant court-circuitées
par effet d'avalanche.

- **Avantages :**
  - idem ROM
  - Claquage en quelques minutes
  - Coût relativement faible
- **Inconvénients :**
  - Modification impossible (toute erreur est fatale).

8.3.2.3. L'EPROM ou UV-EPROM

Pour faciliter la mise au point d'un programme ou tout simplement permettre une erreur de programmation, il est intéressant de pouvoir reprogrammer une PROM. La technique de claquage utilisée dans celles-ci ne le permet évidemment pas.
L'EPROM (Erasable Programmable ROM) est une PROM qui peut être effacée.



- **Structure**

Dans une EPROM, le point mémoire est réalisé à partir d'un transistor FAMOS (Floating gate Avalanche injection Metal Oxyde Silicium). Ce transistor MOS a été introduit par Intel en 1971 et a la particularité de posséder une grille flottante.

- **Programmation**

La programmation consiste à piéger des charges dans la grille flottante. Pour cela, il faut tout d'abord appliquer une très forte tension entre Grille et Source. Si l'on applique ensuite une tension entre D et S, la canal devient conducteur. Mais comme la tension Grille-Source est très importante, les électrons sont déviés du canal vers la grille flottante et capturés par celle-ci. Cette charge se maintient une dizaine d'années en condition normale.
L'exposition d'une vingtaine de minutes à un rayonnement ultraviolet permet d'annuler la charge stockée dans la grille flottante. Cet effacement est reproductible plus d'un millier de fois. Les boîtiers des EPROM se caractérisent donc par la présence d'une petite fenêtre transparente en quartz qui assure le passage des UV. Afin d'éviter toute perte accidentelle de l'information, il faut obturer la fenêtre d'effacement lors de l'utilisation.

- **Avantages :**
  - Reprogrammable et non Volatile
- **Inconvénients :**
  - Impossible de sélectionner une seule cellule à effacer
  - Impossible d'effacer la mémoire in-situ.
  - l'écriture est beaucoup plus lente que sur une RAM. (environ 1000x)

8.3.2.4. L'EEPROM

L'EEPROM (Electically EPROM) est une mémoire programmable et effaçable électriquement.
Elle répond ainsi à l'inconvénient principal de l'EPROM et peut être programmée in situ.

- **Structure**

Dans une EEPROM, le point mémoire est réalisé à partir d'un transistor SAMOS reprenant le même principe que le FAMOS sauf que l'épaisseur entre les deux grilles est beaucoup plus faible.

- **Programmation**

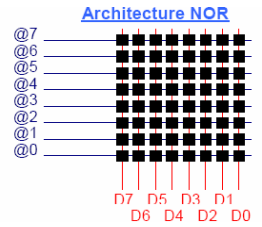
Une forte tension électrique appliquée entre grille et source conduit à la programmation de la mémoire. Une forte tension inverse provoquera la libération des électrons et donc l'effacement de la mémoire.

- **Avantages :**
  - Comportement d'une RAM non Volatile.
  - Programmation et effacement mot par mot possible.
- **Inconvénients :**
  - Très lente pour une utilisation en RAM.
  - Coût de réalisation.

8.3.2.5. La FLASH EPROM

La mémoire Flash s'apparente à la technologie de l'EEPROM. Elle est programmable et effaçable électriquement comme les EEPROM.

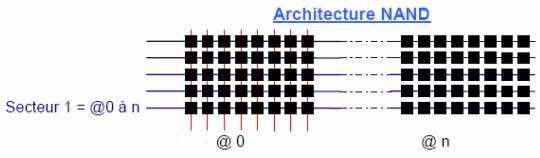
- **Structure**



Il existe deux technologie différentes qui se différencient par l'organisation de leurs réseaux mémoire : l'architecture NOR et NAND. L'architecture NOR propose un assemblage des cellules élémentaires de mémorisation en parallèle avec les lignes de sélection comme dans une EEPROM classique. L'architecture NAND propose un assemblage en série de ces mêmes cellules avec les lignes de sélection. D'un point de vue pratique, la différence majeure entre NOR et NAND tient à leurs interfaces. Alors qu'une NOR dispose de bus d'adresses et de données dédiés, la NAND est dotée d'une interface d'E/S indirecte.
Par contre, la structure NAND autorise une implantation plus dense grâce à une taille de cellule approximativement 40 % plus petite que la structure NOR.

- **Programmation**

Si NOR et NAND exploitent toutes deux le même principe de stockage de charges dans la grille flottante d'un transistor, l'organisation de leur réseau mémoire n'offre pas la même souplesse d'utilisation. Les Flash NOR autorise un adressage aléatoire qui permet de la programmer octet par octet alors que la Flash NAND autorise un accès séquentiel aux données et permettra seulement une programmation par secteur comme sur un disque dur.



- **Avantages**

Flash NOR :

- Comportement d'une RAM non Volatile.
- Programmation et effacement mot par mot possible.
- Temps d'accès faible.

Flash NAND :

- Comportement d'une RAM non Volatile.
- Forte densité d'intégration donc coût réduit.
- Rapidité de l'écriture/lecture par paquet
- Consommation réduite.

- **Inconvénients**

Flash NOR :

- Lenteur de l'écriture/lecture par paquet.
- coût.

Flash NAND :

- Ecriture/lecture par octet impossible.
- Interface E/S indirecte

La Flash EPROM a connu un essor très important ces dernières années avec le boom de la téléphonie portable et des appareils multimédia (PDA, appareil photo numérique, lecteur MP3, etc...).

8.4. Critères de choix d'une mémoire

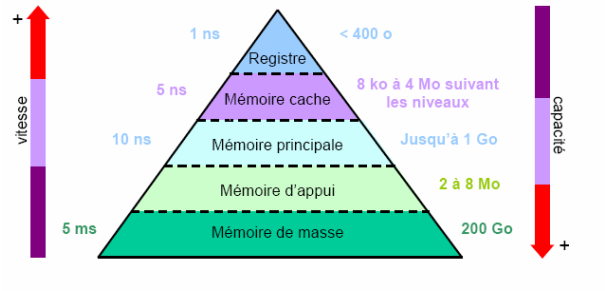
Les principaux critères à retenir sont :

- capacité
- vitesse
- consommation

- coût

8.5. Notion de hiérarchie mémoire

Une mémoire idéale serait une mémoire de grande capacité, capable de stocker un maximum d'informations et possédant un temps d'accès très faible afin de pouvoir travailler rapidement sur ces informations. Mais il se trouve que les mémoires de grande capacité sont souvent très lentes et que les mémoires rapides sont très chères. Et pourtant, la vitesse d'accès à la mémoire conditionne dans une large mesure les performances d'un système. En effet, c'est là que se trouve le *goulot d'étranglement* entre un microprocesseur capable de traiter des informations très rapidement et une mémoire beaucoup plus lente (ex : processeur actuel à 3Ghz et mémoire à 400MHz). Or, on n'a jamais besoin de toutes les informations au même moment. Afin d'obtenir le meilleur compromis coût-performance, on définie donc une hiérarchie mémoire. On utilise des mémoires de faible capacité mais très rapide pour stocker les informations dont le microprocesseur se sert le plus et on utilise des mémoires de capacité importante mais beaucoup plus lente pour stocker les informations dont le microprocesseur se sert le moins. Ainsi, plus on s'éloigne du microprocesseur et plus la capacité et le temps d'accès des mémoires vont augmenter.



- **Les registres** sont les éléments de mémoire les plus rapides. Ils sont situés au niveau du processeur et servent au stockage des opérandes et des résultats intermédiaires.
- **La mémoire cache** est une mémoire rapide de faible capacité destinée à accélérer l'accès à la mémoire centrale en stockant les données les plus utilisées.
- **La mémoire principale** est l'organe principal de rangement des informations. Elle contient les programmes (instructions et données) et est plus lente que les deux mémoires précédentes.
- **La mémoire d'appui** sert de mémoire intermédiaire entre la mémoire centrale et les mémoires de masse. Elle joue le même rôle que la mémoire cache.
- **La mémoire de masse** est une mémoire périphérique de grande capacité utilisée pour le stockage permanent ou la sauvegarde des informations. Elle utilise pour cela des supports magnétiques (disque dur, ZIP) ou optiques (CDROM, DVDROM).

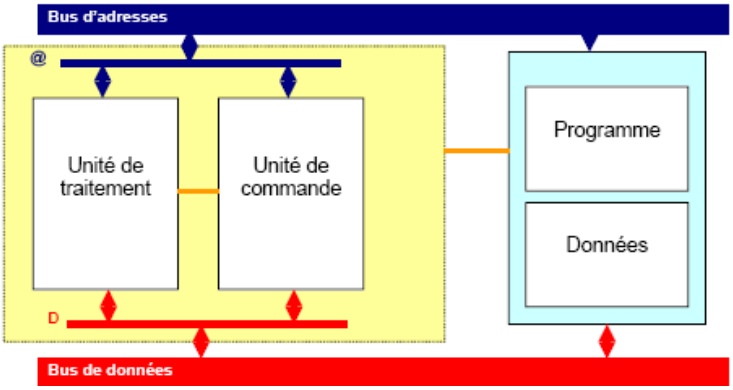
9. Microprocesseur

Un microprocesseur (noté **CPU**, pour *Central Processing Unit*) est un circuit électronique cadencé au rythme d'une horloge interne, grâce à un cristal de quartz qui, soumis à un courant électrique, envoie des impulsions, appelées « **top** ». La **fréquence d'horloge** (appelée également **cycle**, correspondant au nombre d'impulsions par seconde, s'exprime en Hertz (Hz).Il est doté des facultés d'interprétation et d'exécution des instructions d'un programme.
Il est chargé d'organiser les tâches précisées par le programme et d'assurer leur exécution. Il doit aussi prendre en compte les informations extérieures au système et assurer leur traitement. C'est le cerveau du système.
A l'heure actuelle, un microprocesseur regroupe sur quelques millimètre carré des fonctionnalités toujours plus complexes. Leur puissance continue de s'accroître et leur encombrement

diminue régulièrement respectant toujours, pour le moment, la fameuse *loi de Moore* (1).

9.1. Architecture de base d'un microprocesseur

Un microprocesseur est construit autour de deux éléments principaux :
 Une unité de commande
 Une unité de traitement
associés à des registres chargées de stocker les différentes informations à traiter. Ces trois éléments sont reliés entre eux par des bus interne permettant les échanges d'informations.



Remarques :
Il existe deux types de registres :
 les registres d'usage général permettent à l'unité de traitement de manipuler des données à vitesse élevée. Ils sont connectés au bus données interne au microprocesseur.
 les registres d'adresses (pointeurs) connectés sur le bus adresses.

9.1.1. L'unité de commande

Elle permet de séquencer le déroulement des instructions. Elle effectue la recherche en mémoire de l'instruction. Comme chaque instruction est codée sous forme binaire, elle en assure le décodage pour enfin réaliser son exécution puis effectue la préparation de l'instruction suivante. Pour cela, elle est composée par :
 le compteur de programme constitué par un registre dont le contenu est initialisé avec l'adresse de la première instruction du programme. Il contient toujours l'adresse de l'instruction à exécuter.
le registre d'instruction et le décodeur d'instruction : chacune des instructions à exécuter est rangée dans le registre instruction puis est décodée par le décodeur d'instruction.
 Bloc logique de commande (ou séquenceur) : Il organise l'exécution des instructions au rythme d'une horloge. Il élabore tous les signaux de synchronisation internes ou externes (bus de commande) du microprocesseur en fonction des divers signaux de commande provenant du décodeur d'instruction ou du registre d'état par exemple. Il s'agit d'un automate réalisé soit de façon câblée (obsolète), soit de façon micro-programmée, on parle alors de micromicroprocesseur.

9.1.1.1. L'unité de traitement

C'est le coeur du microprocesseur. Elle regroupe les circuits qui assurent les traitements nécessaires à l'exécution des instructions :
 L'Unité Arithmétique et Logique (UAL) est un circuit complexe qui assure les fonctions logiques (ET, OU, Comparaison, Décalage , etc…) ou arithmétique (Addition, soustraction).
 Le registre d'état est généralement composé de 8 bits à considérer individuellement. Chacun de ces bits est un indicateur dont l'état dépend du résultat de la dernière opération effectuée par l'UAL. On les appelle *indicateur d'état* ou *flag* ou *drapeaux*. Dans un programme le résultat du test de leur état conditionne souvent le déroulement de la suite du programme. On

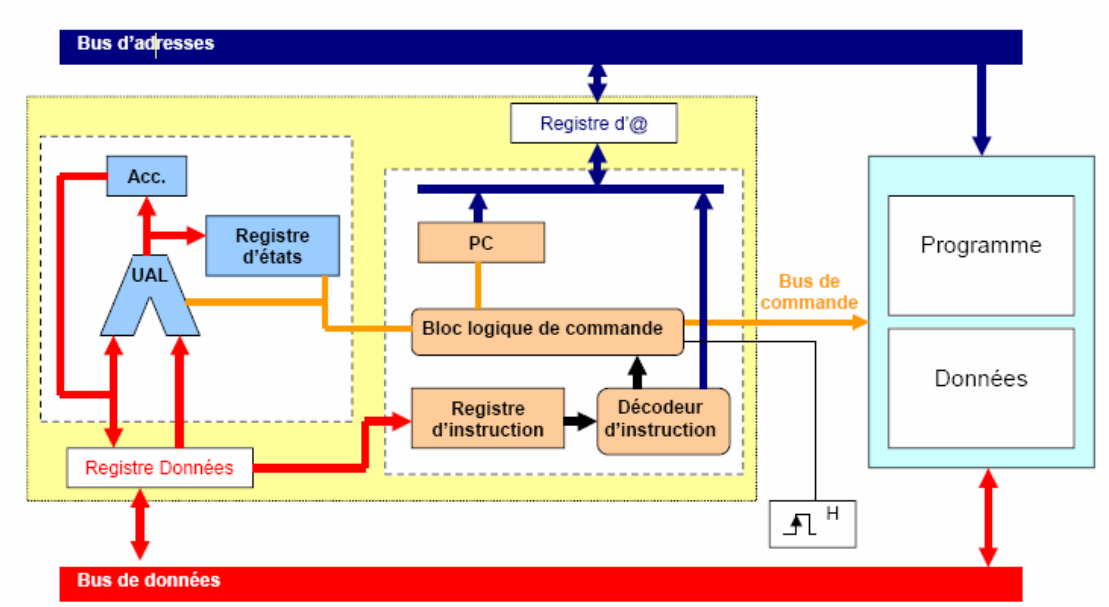
peut citer par exemple les indicateurs de :
o retenue **(carry : C) 1 si retenu ; 0 si non**
o retenue intermédiaire **(Auxiliary-Carry : AC)**

o signe **(Sign : S) 0 si positif et 1 si négatif**
o débordement **(overflow : OV ou V)**
o zéro **(Z) 1 si résultat nul ; 0 si non**
o parité **(Parity : P)**

C	AC	S	Z	P			

 Les accumulateurs sont des registres de travail qui servent à stocker une opérande au début d'une opération arithmétique et le résultat à la fin de l'opération.

9.1.2. Schéma fonctionnel

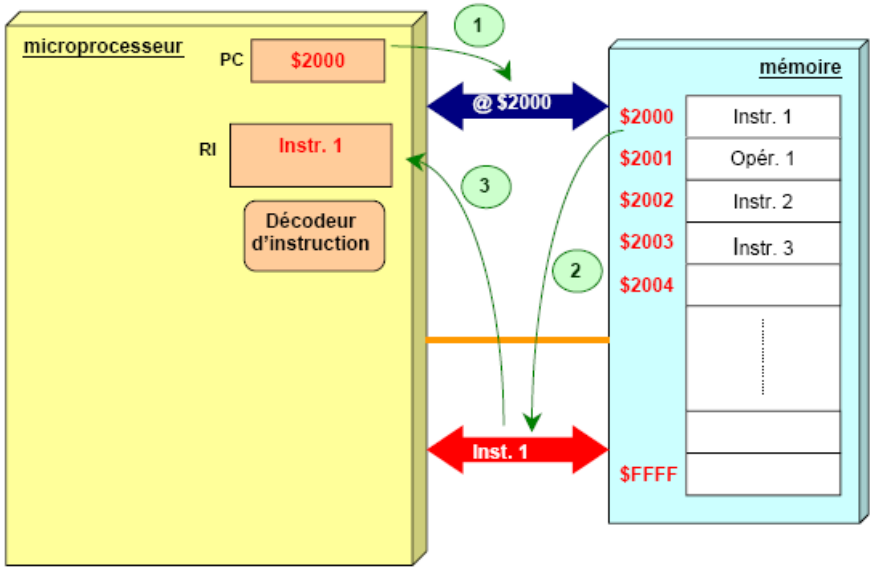


Cycle d'exécution d'une instruction

Le microprocesseur ne comprend qu'un certain nombre d'instructions qui sont codées en binaire. Le traitement d'une instruction peut être décomposé en trois phases.

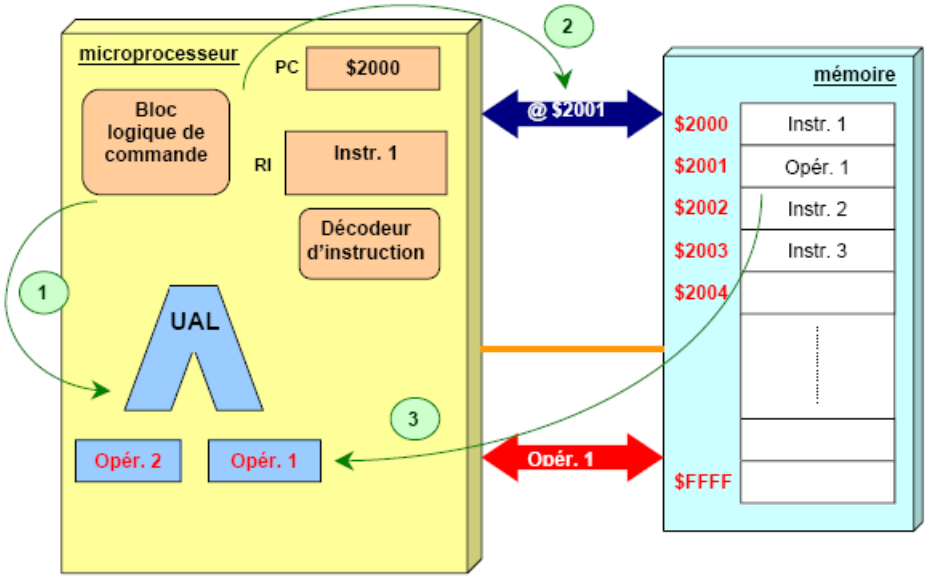
Phase 1: ***Recherche de l'instruction à traiter***

1. Le PC contient l'adresse de l'instruction suivante du programme. Cette valeur est placée sur le bus d'adresses par l'unité de commande qui émet un ordre de lecture.
2. Au bout d'un certain temps (temps d'accès à la mémoire), le contenu de la case mémoire sélectionnée est disponible sur le bus des données.
3. L'instruction est stockée dans le registre instruction du processeur.



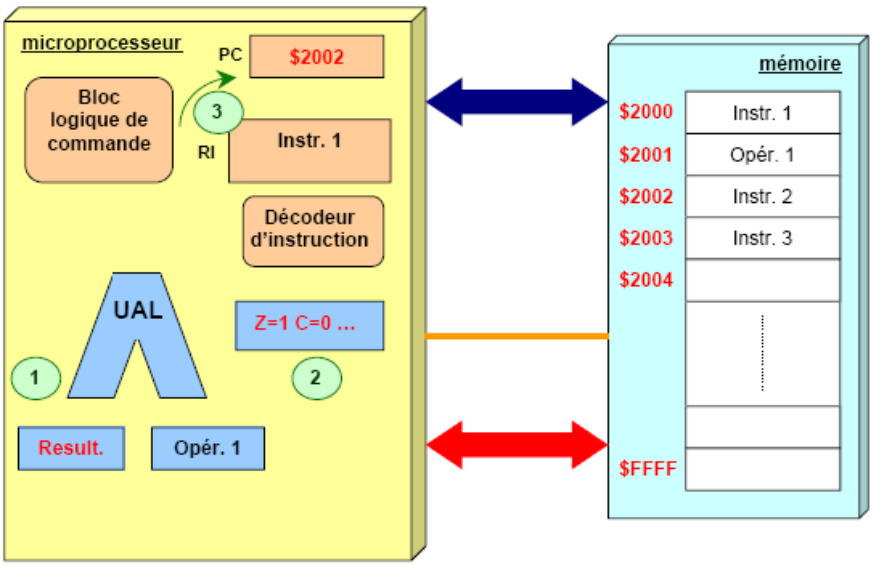
Phase 2 : Décodage de l'instruction et recherche de l'opérande

Le registre d'instruction contient maintenant le premier mot de l'instruction qui peut être codée sur plusieurs mots. Ce premier mot contient le code opératoire qui définit la nature de l'opération à effectuer (addition, rotation,...) et le nombre de mots de l'instruction.
1. L'unité de commande transforme l'instruction en une suite de commandes élémentaires nécessaires au traitement de l'instruction.
2. Si l'instruction nécessite une donnée en provenance de la mémoire, l'unité de commande récupère sa valeur sur le bus de données.
3. L'opérande est stocké dans un registre.



Phase 3 : Exécution de l'instruction

1. Le micro-programme réalisant l'instruction est exécuté.
2. Les drapeaux sont positionnés (*registre d'état*).
3. L'unité de commande positionne le PC pour l'instruction suivante.

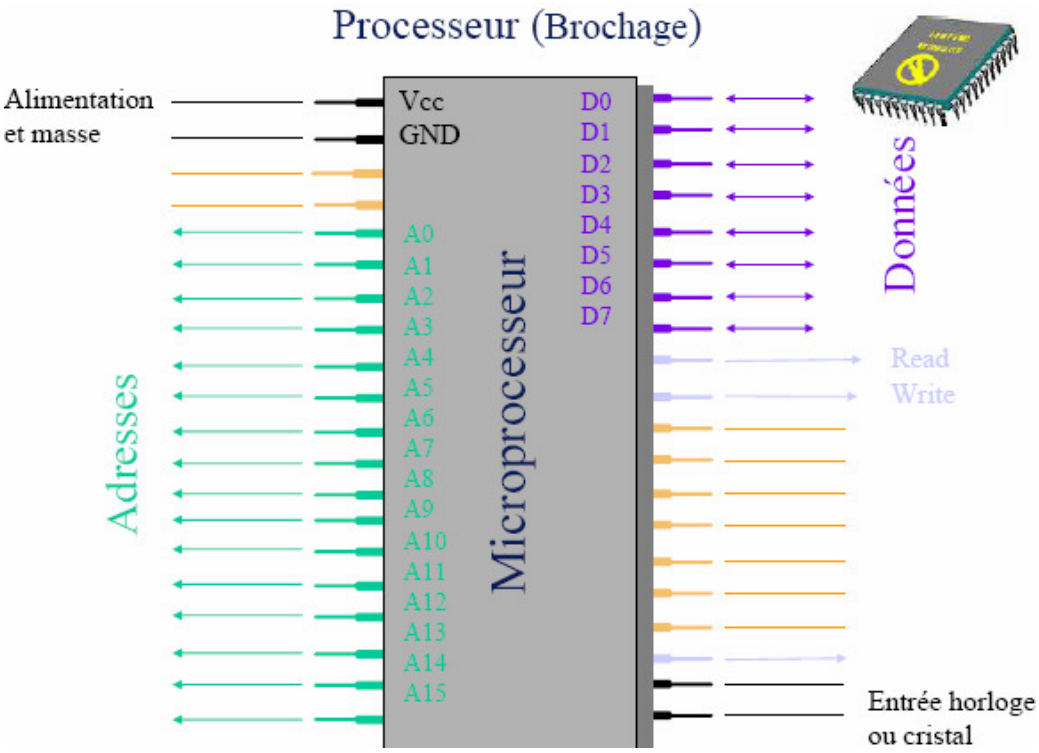


Les étapes d'exécution d'une instruction
(Chargement/Décodage/Exécution)

Début

Charger	la prochaine instruction à exécuter depuis la mémoire dans le registre instruction (RI),	Fetch
Modifier	le Compteur Ordinal pour qu'il pointe sur la prochaine instruction à exécuter,	
Décoder	l'instruction qui vient d'être chargée,	Décodage
Charger	les données éventuelles dans les registres internes,	Exécution
Exécuter	l'instruction,	

Fin


Processeur (Brochage)
Alimentation
et masse
Vcc
GND
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
Microprocesseur
D0
D1
D2
D3
D4
D5
D6
D7
Adresses
Données
Read
Write
Entrée horloge
ou cristal

.