ARO2

L'ordinateur : Histoire et présentation générale

Basé sur le cours du prof. E. Sanchez
et le cours ASP du prof. M.Starkier

Romuald Mosqueron

Cours ASP

BRÈVE HISTOIRE DES ORDINATEURS

Un peu d'histoire

- Le besoin en traitement de l'information est une résultante de la croissance de la population avec la révolution industrielle (début du XIXème): recensement de la population, messages, données pour les assurances, etc…

- L'industrie des machines de bureau est donc née à la fin du XIXème: machines à écrire, systèmes d'archivage, systèmes de copie et de comptabilité, etc. Mais aussi la machine à calculer!

Informatique

- Petit Robert:
  Science du traitement de l'information; ensemble de techniques de la collecte, du tri, de la mise en mémoire, du stockage, de la transmission et de l'utilisation des informations traitées automatiquement à l'aide de programmes mis en oeuvre sur ordinateurs.

Système informatique

- Un système informatique est un ensemble de composants de type logiciel (*software*) et matériel (*hardware*), mis ensemble pour collaborer dans l'exécution d'une application
- Le principal composant matériel est l'ordinateur
- Un informaticien doit comprendre le fonctionnement de tous les composants d'un système, sans se limiter au logiciel. En effet, les caractéristiques du matériel agissent sur la justesse et la performance des programmes
- Une bonne connaissance du matériel permet d'éviter des erreurs et d'augmenter la performance, en optimisant les programmes

Définition de l'ordinateur

- [Larousse] Machine automatique de traitement de l'information, obéissant à des programmes formés par des suites d'opérations arithmétiques et logiques.
- [Wikipedia] Un ordinateur est une machine électronique qui fonctionne par la lecture séquentielle d'un ensemble d'instructions qui lui font exécuter des opérations logiques et arithmétiques sur des chiffres binaires.
- Computer:1640s, "one who calculates," agent noun from compute (v.). Meaning "calculating machine" (of any type) is from 1897;

L'ordinateur

- Un ordinateur est une machine capable de traiter des informations en suivant une liste d'instructions (un programme informatique)
- Le terme « ordinateur » est d'origine biblique (il se trouvait dans le Littré comme adjectif désignant « Dieu qui met de l'ordre dans le monde ») et a été proposé par le philologue Jacques Perret dans une lettre datée du 16 avril 1955 en réponse à une demande d'IBM France. Les dirigeants estimaient le mot « calculateur » (computer) bien trop restrictif en regard des possibilités de ces machines

fr.wikipedia.org

L'ordinateur humain

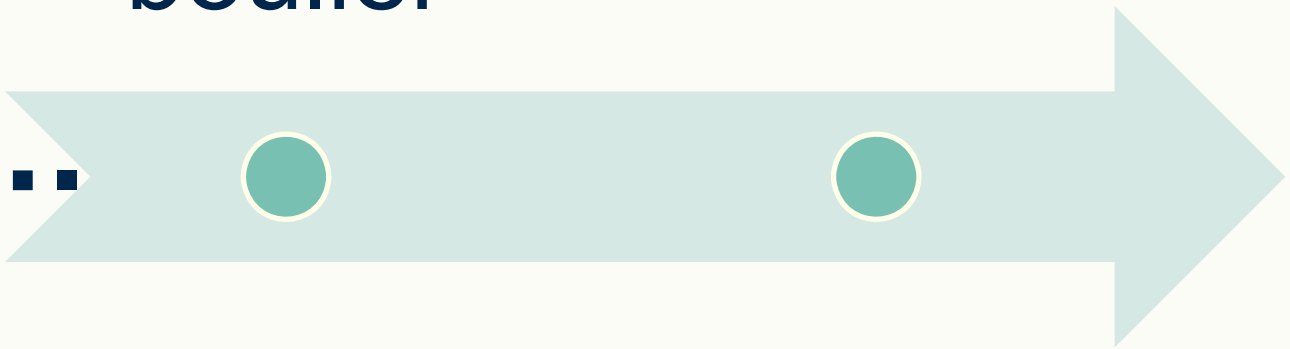
- Le premier ordinateur en tant qu'élément de traitement de l'information était donc l'homme (et, plus généralement, la femme)
- Le premier essai de traitement de l'information à grande échelle, avec des ordinateurs humains, est fait pour la réalisation de tables mathématiques, logarithmiques ou trigonométriques
- Les tables logarithmiques sont nées au XVIème et XVIIème.
- A la fin du XVIIIème il y avait d'autres tables, toutes faites par des ordinateurs humains: pour la navigation, pour les compagnies d'assurances, pour les ingénieurs civils, etc. Celles pour la navigation étaient d'une grande importance économique

Chronologie

Machines à calculer

~700
boulier

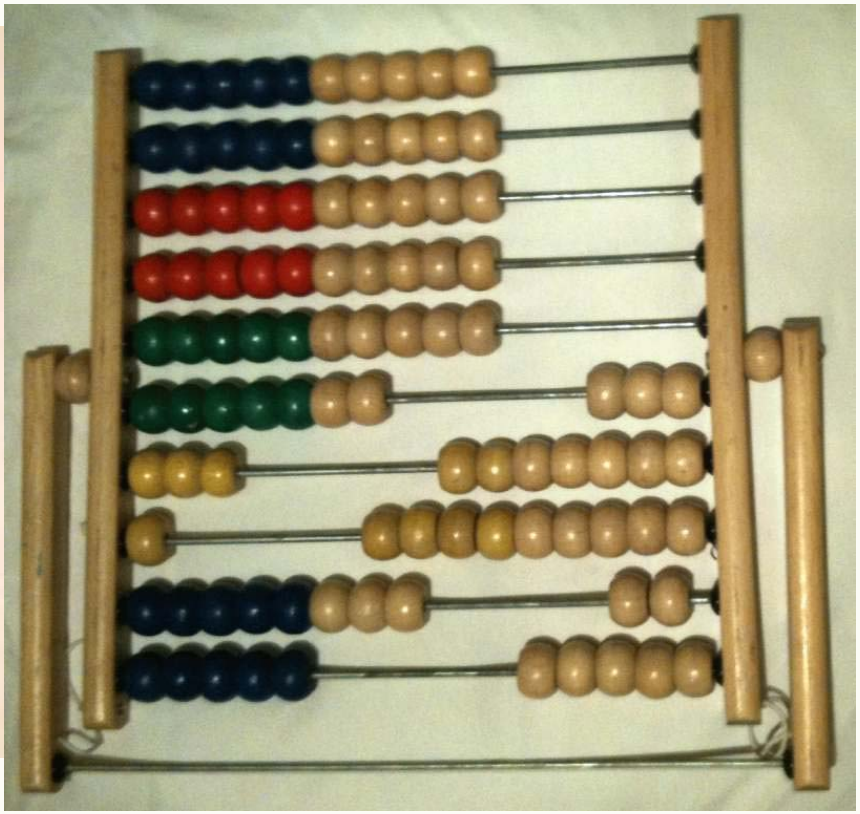
??....



Machines à calculer

Boulier ou abaque (~700)

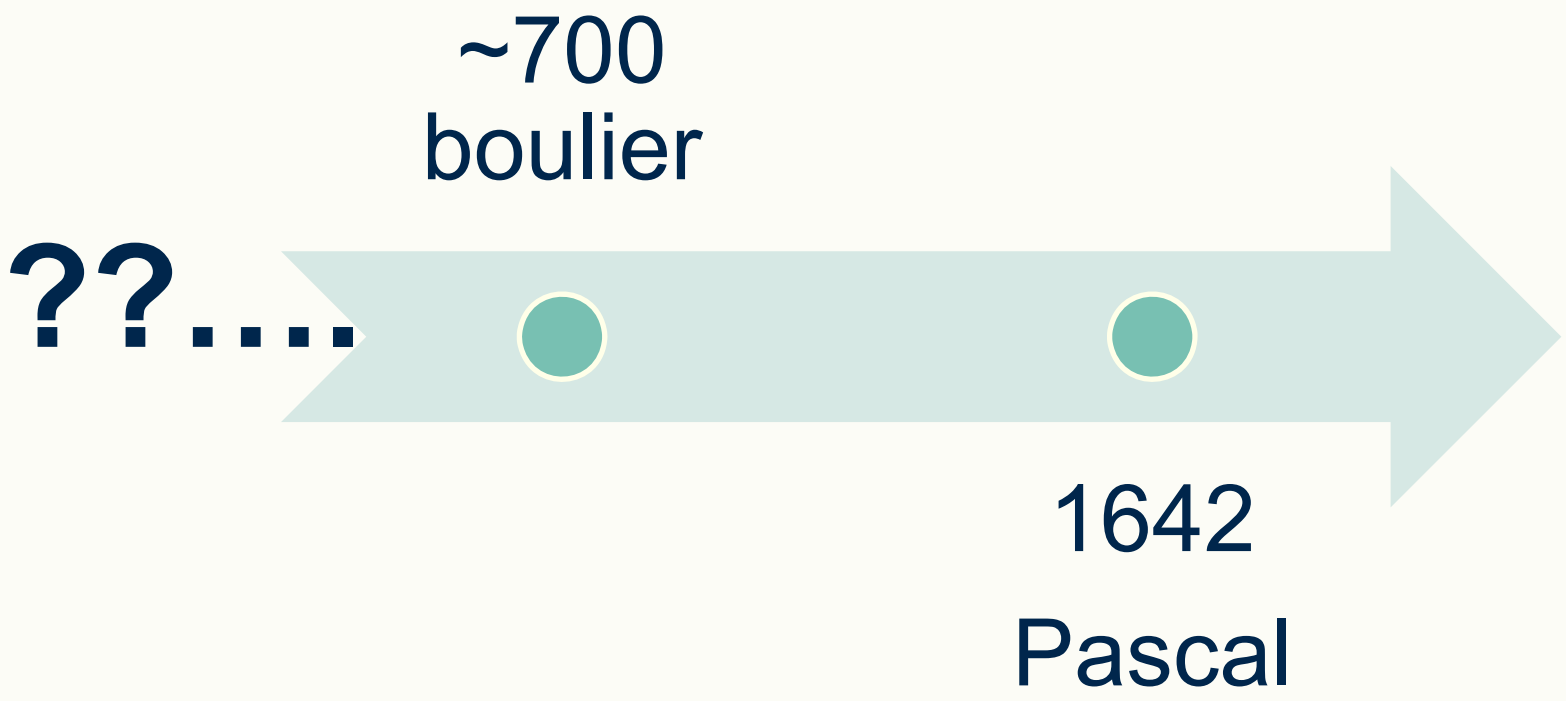
Asie



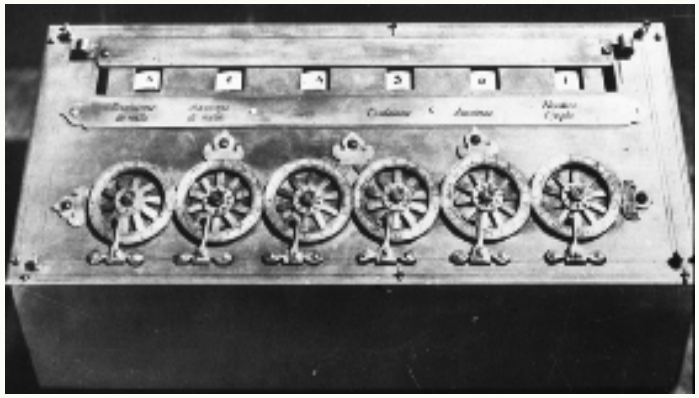
Europe

Chronologie

Machines à calculer



Machines à calculer



“Pascaline” (Pascal, 1642)

https://www.youtube.com/watch?v=3h71HAJWnVU&t=21s

Chronologie

Machines à calculer

Machines à calculer

"Reckoner" (Leibniz, 1673)

https://www.youtube.com/watch?v=aWDWiQHOCHw

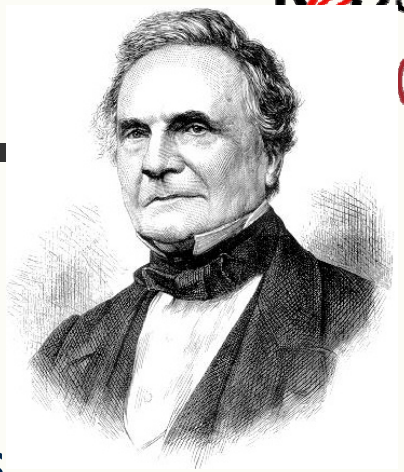
Chronologie

Machines à calculer

Traitement de l'information au 19ème

- En 1830 les banques construisent un bâtiment, le *Banker's Clearing House*. Il y avait dedans une trentaine d'employés qui s'échangeaient des chèques. A 16h toutes les transactions s'arrêtaient et chacun faisait son bilan. A 17h un inspecteur prenait place au milieu, recevait l'argent des banques débiteurs et le distribuait aux créanciers
- En 1842 est créée la *Railway Clearing House*. En 1870 elle emploie 1'300 employés et traite 5 millions de transactions/an
- La *Central Telegraph Office* centralise tous les messages entre les villes anglaises. Fondée en 1859, elle emploie 1'200 personnes en 1875 et 4'500 à la fin du siècle, pour traiter 150'000 télégrammes par jour
- *Prudential*, compagnie d'assurances fondée en 1856, gérait 20 ans après 2.5 millions de polices, avec un million de nouvelles polices par année et 300 employés

Charles Babbage



- Mathématicien anglais né en 1791
- Admis à l'Université de Cambridge en 1810
- Membre de la Royal Society, principale organisation scientifique anglaise, à l'âge de 25 ans
- En 1819, pendant l'un de ses voyages en France, il prend connaissance du plus grand projet de table de l'époque, commandé par Napoléon en 1790 pour calculer les taxes foncières et réaliser les passage des poids et mesures au système décimal. Le travail était divisé en trois parties:
  - douze mathématiciens pour choisir les formules
  - organisation des calculs et compilation des résultats
  - réalisation des calculs par 60-80 ordinateurs humains
- Les tables sont prêtes en 1801 mais elle ne sont jamais imprimées, en absence de fonds...

Difference Engine

(Charles Babbage, 1833)

© Science museum, Londres

https://www.youtube.com/watch?v=B8tmfcOg8l8

https://en.wikipedia.org/wiki/Difference_engine

Difference Engine (2)

- Une machine à différences utilise la méthode des différences de Newton pour calculer la valeur d'un polynôme pour x = 1, 2, 3, …
- Exemple: calculer $f(x) = 2x^2 + 3x + 5$, pour x=1, 2, 3…

Première différence (diff1)

f(1) = 2+3+5 = 10
f(2) = 8+6+5 = 19 = f(1) + 9
f(3) = 18+9+5= 32 = f(2) + 13 = f(2) + 9 + 4
f(4) = 32+12+5=49 = f(3) + 17 = f(3) + 13 + 4
f(5) = 50+15+5=70 = f(4) + 21 = f(4) + 17 + 4

deuxième différence (diff2)

f(x) = f(x-1) + diff1(x-1) + diff2

⇨ diff1(x) = diff1(x-1) + diff2

Analytical Engine

- En 1834, Babbage abandonne le projet de développement de la Difference Engine, pour une machine plus ambitieuse, “capable de réaliser tout calcul qu'un homme soit capable de spécifier pour elle”: c'est la Analytical Engine. Le gouvernement lui coupe le financement…
- Le principal concept de l'*Analytical Engine* était la séparation entre les unités de calcul et de stockage des nombres. Il fallait ordonnancer le transfert entre les deux unités: c'était la programmation

Ada Lovelace

La meilleure description de la machine est écrite à l'époque par Ada Lovelace (1815-1852): *Sketch of the Analytical Engine*.

Les biographes considèrent que la plupart des programmes qui y apparaissent sont l'œuvre de Babbage, mais elle y décrit une manière de calculer les nombres de Bernoulli est ces notes sont considérées comme le premier programme informatique au monde.

Ada Lovelace

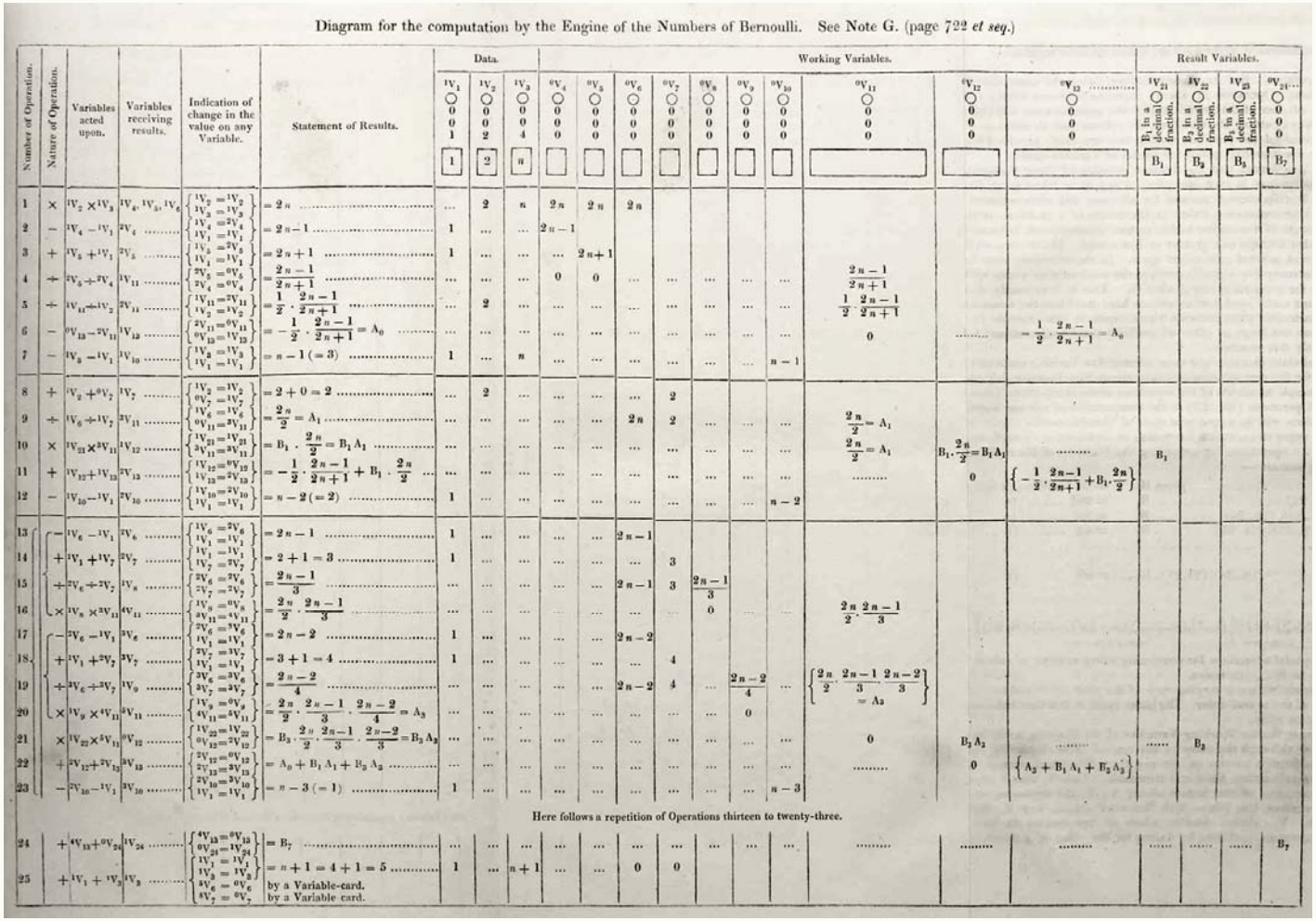


Diagram for the computation by the Engine of the Numbers of Bernoulli. See Note G. (page 722 *et seq.*)

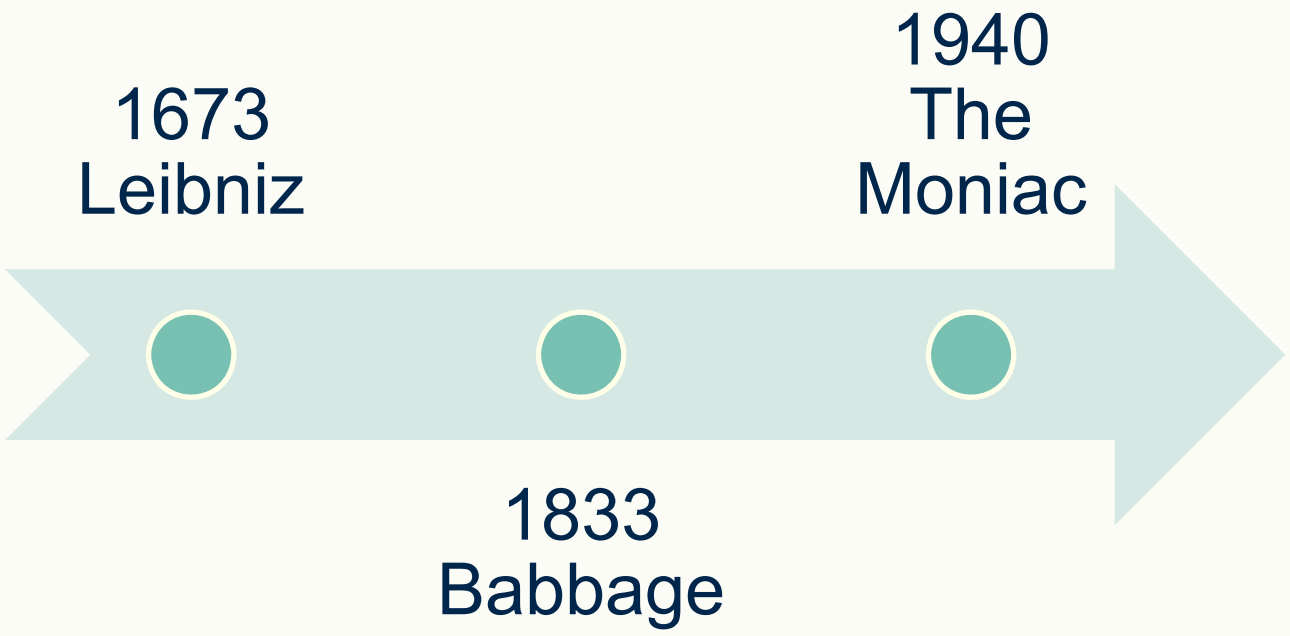
Number of Operation.	Nature of Operation.	Variables acted upon.	Variables receiving results.	Indication of change in the value on any Variable.	Statement of Results.	Data. $^1V_1$	Data. $^1V_2$	Data. $^1V_3$	Working Variables. $^0V_4$	$^0V_5$	$^0V_6$	$^0V_7$	$^0V_8$	$^0V_9$	$^0V_{10}$	$^0V_{11}$	$^0V_{12}$	$^0V_{13}$	Result Variables. $^1V_{21}$ B_1	$^1V_{22}$ B_3	$^1V_{23}$ B_5	$^0V_{24}$ ... B_7
1	×	$^1V_2 \times ^1V_3$	$^1V_4, ^1V_5, ^1V_6$	$^1V_2 = ^1V_2$, $^1V_3 = ^1V_3$	$= 2n$	...	2	n	$2n$	$2n$	$2n$											
2	−	$^1V_4 - ^1V_1$	$^2V_4$	$^1V_4 = ^2V_4$, $^1V_1 = ^1V_1$	$= 2n-1$	1	...	...	$2n-1$													
3	+	$^1V_5 + ^1V_1$	$^2V_5$	$^1V_5 = ^2V_5$, $^1V_1 = ^1V_1$	$= 2n+1$	1	...	...	...	$2n+1$												
4	÷	$^2V_5 \div ^2V_4$	$^1V_{11}$	$^2V_5 = ^0V_5$, $^2V_4 = ^0V_4$	$= \frac{2n-1}{2n+1}$	...	...	...	0	0	...	...	...	...	...	$\frac{2n-1}{2n+1}$						
5	+	$^1V_{11} + ^1V_2$	$^2V_{11}$	$^1V_{11} = ^2V_{11}$, $^1V_2 = ^1V_2$	$= \frac{1}{2} \cdot \frac{2n-1}{2n+1}$	...	2	...	...	...	...	...	...	...	...	$\frac{1}{2} \cdot \frac{2n-1}{2n+1}$						
6	−	$^0V_{13} - ^2V_{11}$	$^1V_{13}$	$^2V_{11} = ^0V_{11}$, $^0V_{13} = ^1V_{13}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} = A_0$	...	...	...	...	...	...	...	...	...	...	0	.....	$-\frac{1}{2} \cdot \frac{2n-1}{2n+1} = A_0$				
7	−	$^1V_3 - ^1V_1$	$^1V_{10}$	$^1V_3 = ^1V_3$, $^1V_1 = ^1V_1$	$= n-1\ (=3)$	1	...	n	...	...	...	...	...	...	$n-1$							
8	+	$^1V_2 + ^0V_7$	$^1V_7$	$^1V_2 = ^1V_2$, $^0V_7 = ^1V_7$	$= 2+0=2$	...	2	...	...	...	...	2										
9	÷	$^1V_6 \div ^1V_7$	$^2V_{11}$	$^1V_6 = ^2V_6$, $^1V_7 = ^2V_7$, $^2V_{11} = ^2V_{11}$	$= \frac{2n}{2} = A_1$	...	...	...	...	...	$2n$	2	...	...	...	$\frac{2n}{2} = A_1$						
10	×	$^1V_{21} \times ^2V_{11}$	$^1V_{12}$	$^1V_{21} = ^1V_{21}$, $^2V_{11} = ^2V_{11}$	$= B_1 \cdot \frac{2n}{2} = B_1 A_1$	...	...	...	...	...	...	...	...	...	...	$\frac{2n}{2} = A_1$	$B_1 \cdot \frac{2n}{2} = B_1 A_1$	.....	B_1			
11	+	$^1V_{12} + ^1V_{13}$	$^2V_{13}$	$^1V_{12} = ^0V_{12}$, $^1V_{13} = ^2V_{13}$	$= -\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}$ ...	...	...	...	...	...	...	...	...	...	...	.....	0	$\left\{-\frac{1}{2} \cdot \frac{2n-1}{2n+1} + B_1 \cdot \frac{2n}{2}\right\}$				
12	−	$^1V_{10} - ^1V_1$	$^2V_{10}$	$^1V_{10} = ^2V_{10}$, $^1V_1 = ^1V_1$	$= n-2\ (=2)$	1	...	...	...	...	...	...	...	...	$n-2$							
13	−	$^1V_6 - ^1V_1$	$^2V_6$	$^1V_6 = ^2V_6$, $^1V_1 = ^1V_1$	$= 2n-1$	1	...	...	...	...	$2n-1$											
14	+	$^1V_1 + ^1V_7$	$^2V_7$	$^1V_1 = ^1V_1$, $^1V_7 = ^2V_7$	$= 2+1=3$	1	...	...	...	...	...	3										
15	÷	$^2V_6 \div ^2V_7$	$^1V_8$	$^2V_6 = ^2V_6$, $^2V_7 = ^2V_7$	$= \frac{2n-1}{3}$	...	...	...	...	...	$2n-1$	3	$\frac{2n-1}{3}$									
16	×	$^1V_8 \times ^2V_{11}$	$^4V_{11}$	$^1V_8 = ^0V_8$, $^2V_{11} = ^4V_{11}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3}$	...	...	...	...	...	...	...	0	...	...	$\frac{2n}{2} \cdot \frac{2n-1}{3}$						
17	−	$^2V_6 - ^1V_1$	$^3V_6$	$^2V_6 = ^3V_6$, $^1V_1 = ^1V_1$	$= 2n-2$	1	...	...	...	...	$2n-2$											
18	+	$^2V_1 + ^2V_7$	$^3V_7$	$^2V_7 = ^3V_7$, $^1V_1 = ^1V_1$	$= 3+1=4$	1	...	...	...	...	...	4										
19	÷	$^3V_6 \div ^3V_7$	$^1V_9$	$^3V_6 = ^3V_6$, $^3V_7 = ^3V_7$	$= \frac{2n-2}{4}$	...	...	...	...	...	$2n-2$	4	...	$\frac{2n-2}{4}$	...	$\left\{\frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{3} = A_2\right\}$						
20	×	$^1V_9 \times ^4V_{11}$	$^5V_{11}$	$^1V_9 = ^0V_9$, $^4V_{11} = ^5V_{11}$	$= \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{4} = A_3$ ...	...	...	...	...	...	...	...	...	0	...							
21	×	$^1V_{22} \times ^5V_{11}$	$^0V_{12}$	$^1V_{22} = ^1V_{22}$, $^0V_{12} = ^2V_{12}$	$= B_3 \cdot \frac{2n}{2} \cdot \frac{2n-1}{3} \cdot \frac{2n-2}{3} = B_3 A_3$	...	...	...	...	...	...	...	...	...	...	0	$B_3 A_3$	.....	.....	B_3		
22	+	$^2V_{12} + ^2V_{13}$	$^3V_{13}$	$^2V_{12} = ^0V_{12}$, $^2V_{13} = ^3V_{13}$	$= A_0 + B_1 A_1 + B_3 A_3$	...	...	...	...	...	...	...	...	...	...	.....	0	$\left\{A_3 + B_1 A_1 + B_3 A_3\right\}$				
23	−	$^2V_{10} - ^1V_1$	$^3V_{10}$	$^2V_{10} = ^3V_{10}$, $^1V_1 = ^1V_1$	$= n-3\ (=1)$	1	...	...	...	...	...	...	...	...	$n-3$							
Here follows a repetition of Operations thirteen to twenty-three.																						
24	+	$^4V_{13} + ^0V_{24}$	$^1V_{24}$	$^4V_{13} = ^0V_{13}$, $^0V_{24} = ^1V_{24}$	$= B_7$	...	...	...	...	...	...	...	...	...	...	.....	.....	.....	.....	.....	.....	B_7
25	+	$^1V_1 + ^1V_3$	$^1V_3$	$^1V_1 = ^1V_1$, $^1V_3 = ^1V_3$, $^2V_6 = ^0V_6$, $^2V_7 = ^0V_7$	$= n+1 = 4+1 = 5$ ..... by a Variable-card. by a Variable card.	1	...	$n+1$	...	...	0	0										

Ordinateurs analogiques/numériques

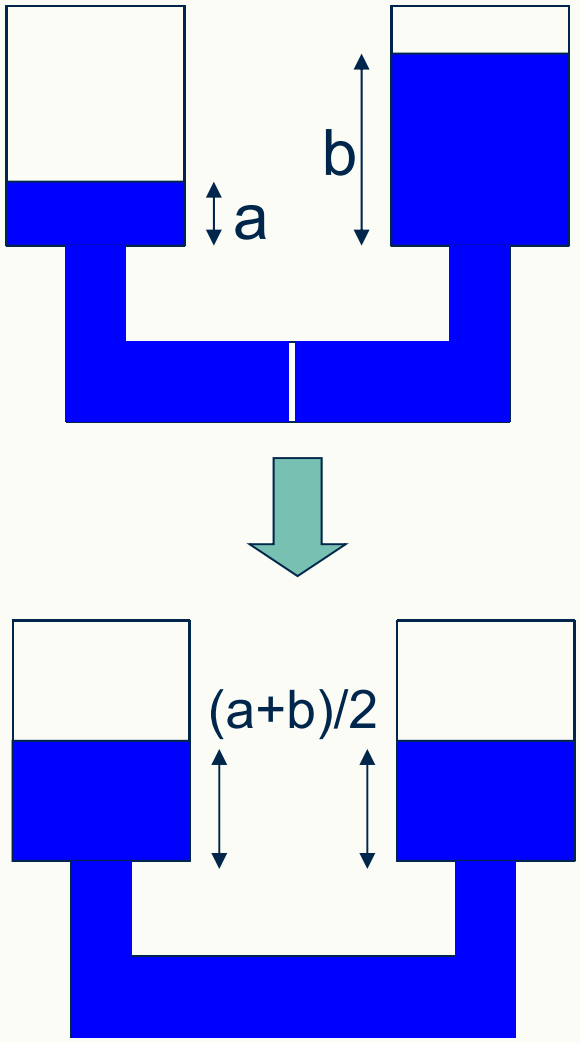
- Babbage travaille sur l'analytical engine jusqu'à sa mort. Non seulement elle était programmable, mais digital (numérique). Donc, conceptuellement, cette machine était plus avancée que les premiers ordinateurs des années 1940.
- Les ordinateurs analogiques utilisent des phénomènes électroniques, mécaniques ou hydrauliques (composants fluidiques) pour simuler UN problème à résoudre.

Chronologie

Machines à calculer



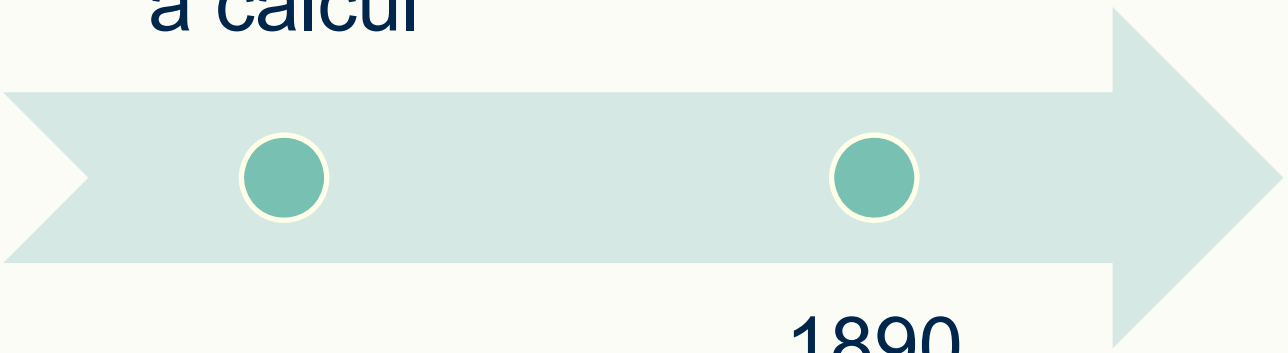
The MONIAC



Chronologie

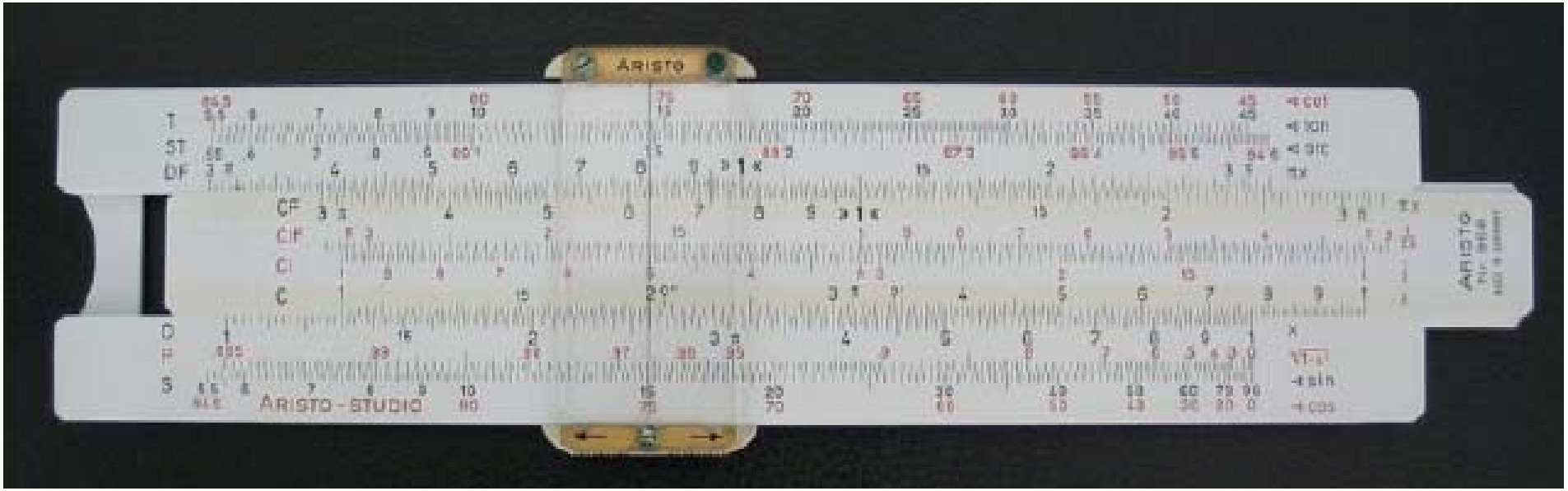
Ordinateurs

1650 Regles
à calcul



1890
Machine à
calculer

Règle à calcul



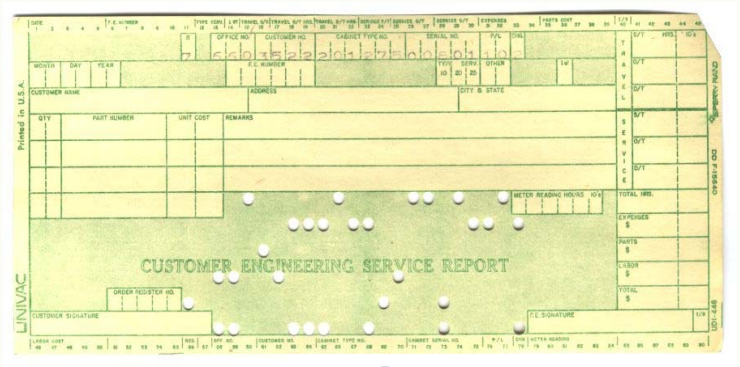
ou règle à calculer est un instrument de calcul qui permet, par simple déplacement longitudinal d'échelles graduées, d'effectuer des opérations arithmétiques de base, multiplication et division, mais pas les additions. Une règle à calcul peut aussi servir à exécuter des opérations plus complexes, telles que le calcul de racines carrées ou cubiques, des calculs logarithmiques ou bien trigonométriques. Inventé vers 1620-1630, elle a été utilisée jusqu'aux années 1970.

Le recensement aux USA

- Le premier grand bureau de traitement de l'information aux USA est créé en 1790: le *Bureau of the Census*. Il était essentiel pour déterminer le nombre de personnes qui pouvaient avoir un représentant au Congrès
- D'une vingtaine d'employés lors de sa création, il passa à 1'495 en 1880, où 21'000 pages de rapports furent pondues, qui demandèrent plus de 7 ans pour être traitées
- Pour le recensement de 1890 un concours est lancé pour choisir une machine. Le gagnant est Herman Hollerith, inventeur d'un système mécanique de traitement de cartes perforées. Chaque carte perforée contenait un certain nombre de données
- Le traitement est fait en seulement 2.5 ans et, avec le succès, Hollerith fonde une compagnie en 1896: Tabulating Machine Company, à l'origine d'IBM. C'était le début de la systématisation des bureaux, avec des systèmes d'archivage, des machines à écrire et à calculer, etc

Machine à calculer

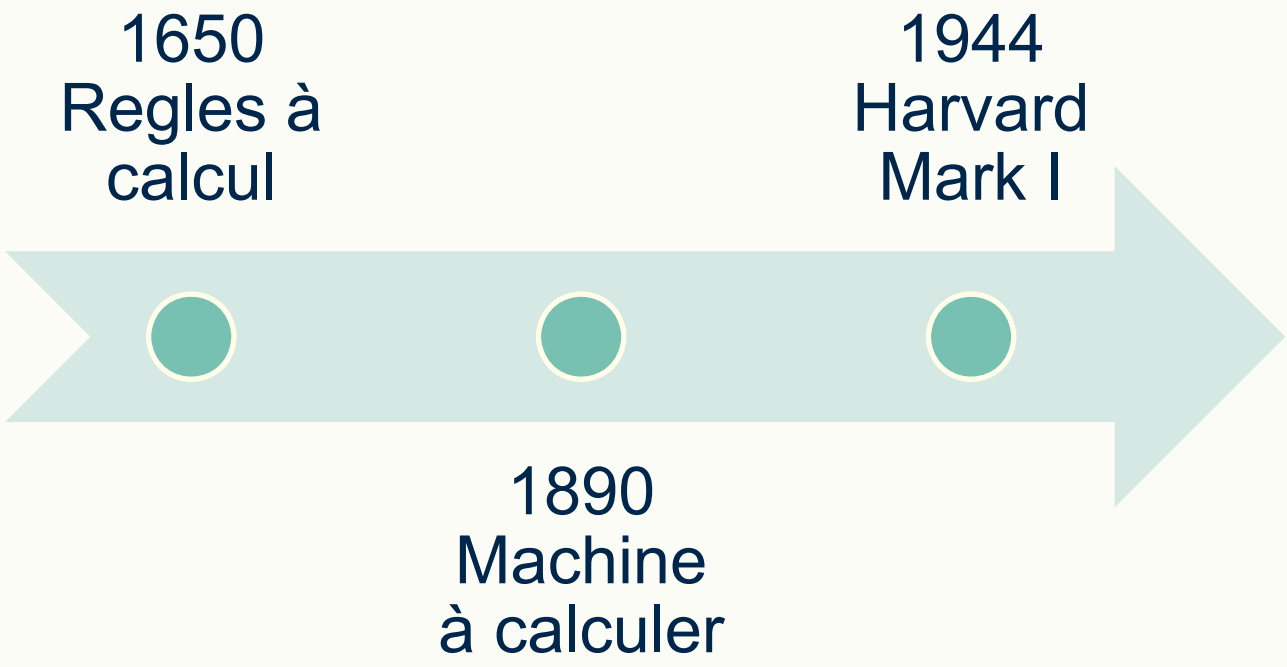
Système d'archivage



Carte perforée

Chronologie

Ordinateurs

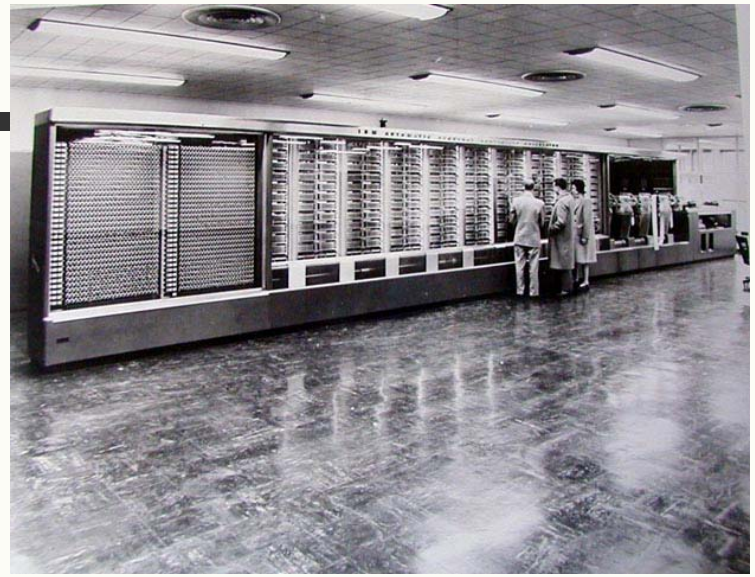


Computers

- “Computer” en anglais désignait un métier jusqu’aux années 1940. Il s’agissait d’une personne qui réalisait des calculs mathématiques.
- La trajectoire du comète Halley a été calculé par une équipe de “computers” en 1759
- Pendant la Seconde Guerre mondiale, des femmes mathématiciennes ont été engagées comme “computers” dans le projet Manhattan

Première machine de calcul automatique

- A la fin des années 30, Howard Hathaway Aiken, un chercheur de Harvard, propose un projet à IBM pour éliminer toute intervention humaine une fois qu'un processus de calcul était lancé. En 1941 Aiken entre à l'armée et la machine devient l'un des projets militaires d'IBM.
- Le premier test de la machine a lieu en janvier 1943. Elle pèse 5 tonnes, sur 51x2 pieds, et utilise un million de pièces et des centaines de kilomètres de câbles. Elle peut stocker 72 nombres, réaliser 3 additions par seconde, une multiplication en 6 secondes et le calcul d'un logarithme ou d'une fonction trigonométrique en une minute. Cette vitesse n'est pas énorme, même pour l'époque, mais elle a le mérite d'être la première machine de calcul réellement automatique, pouvant exécuter un programme introduit dans une bande de papier.

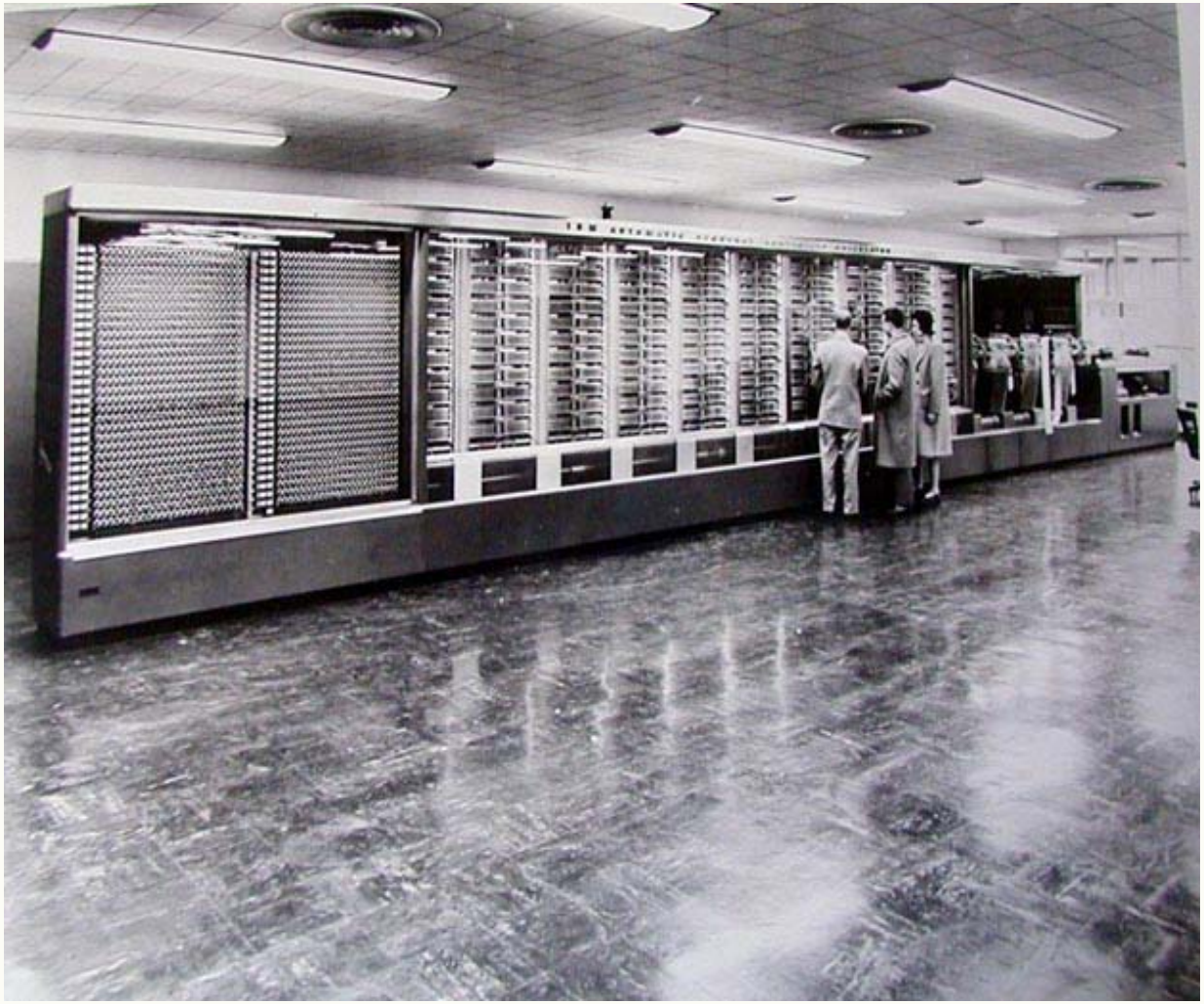


- En 1944 Aiken rentre de l'armée et la présentation officielle de la machine, appelée IBM Automatic Sequence Controlled Calculator ou Harvard Mark I, a lieu à Harvard le 7 août 1944.
- Le succès médiatique est énorme:
  - *American Weekly: Harvard Robot Superbrain*
  - *Popular Science Monthly: Robot Mathematicien Knows All The Answers*

Harvard Mark I

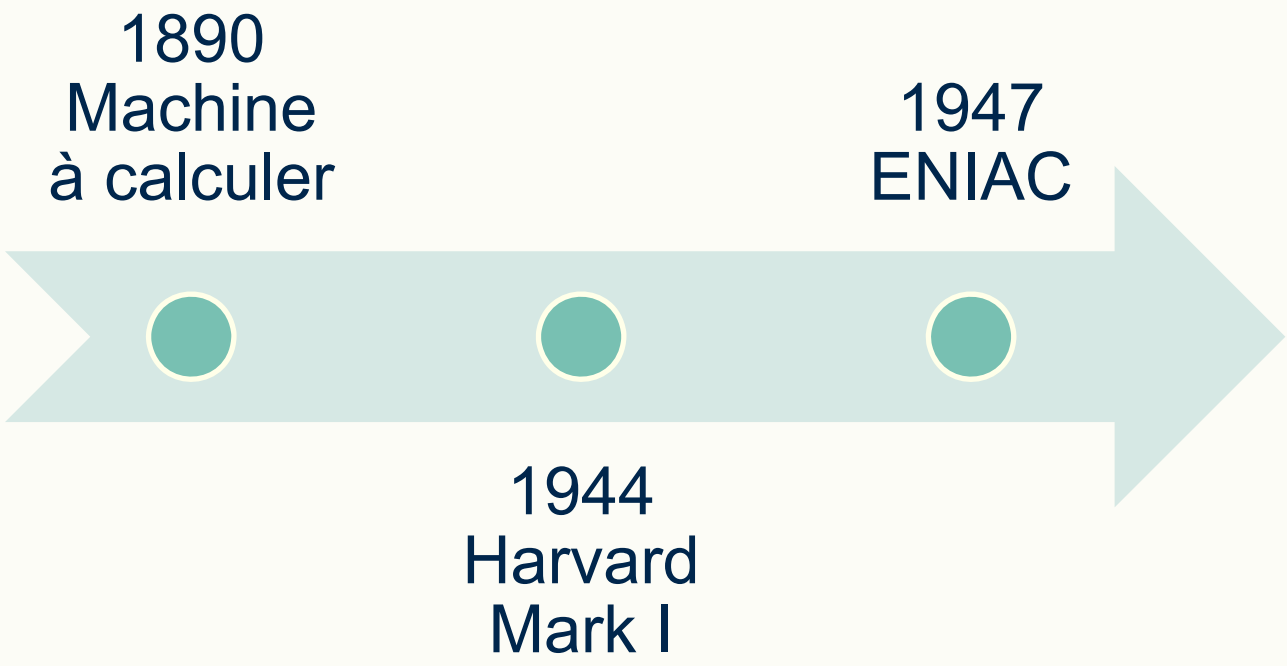
- 1944
- IBM Automatic
- Sequence Controlled Calculator

Site: histoire.info



Chronologie

Ordinateurs



L'ordinateur électronique

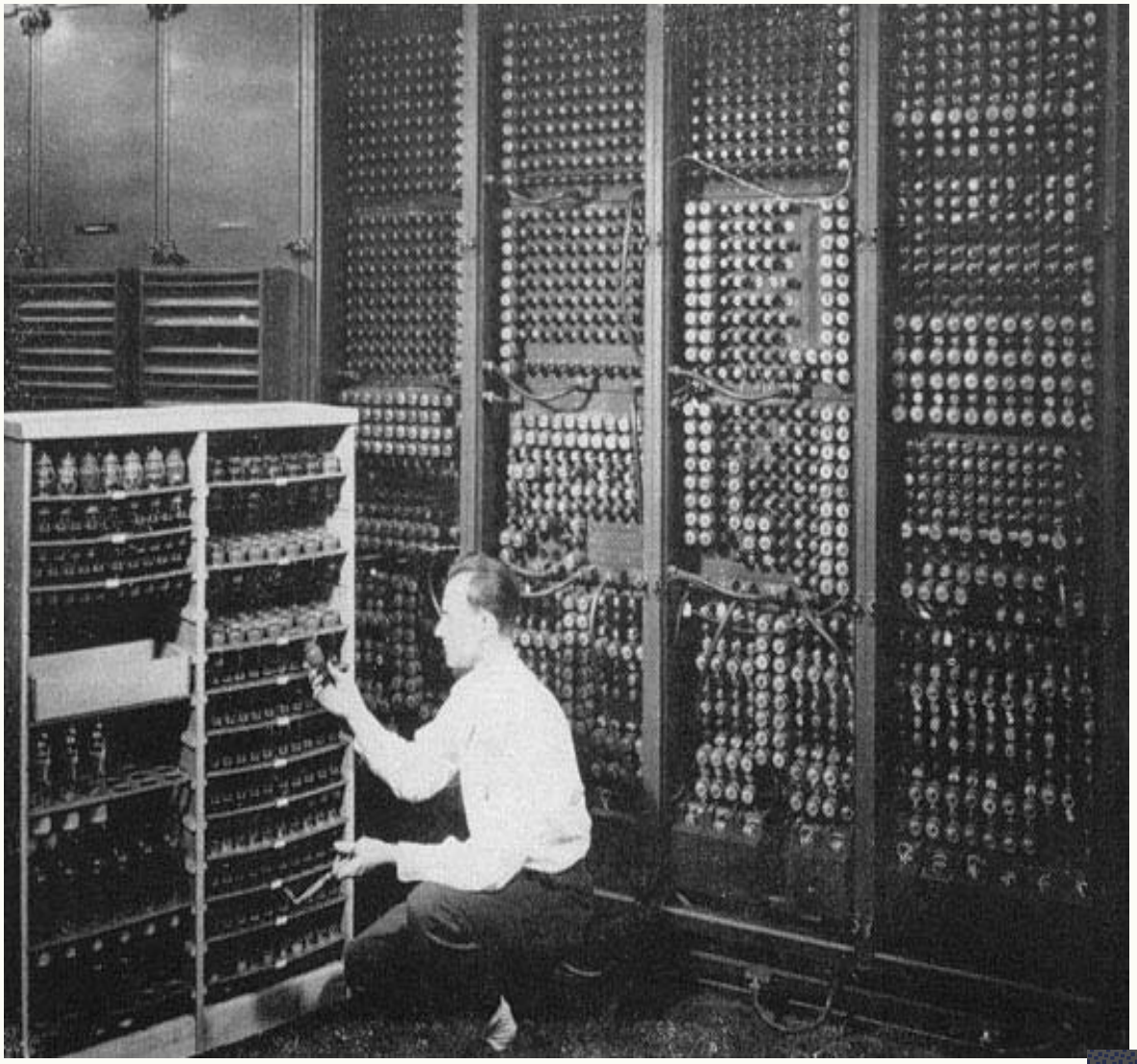
- Les deux grands projets scientifiques de la guerre, la bombe atomique et le radar, demandait de gros calculs.
- Une table de calcul balistique typique avait des données pour 3'000 trajectoires et le calcul complet pouvait prendre un mois à une équipe de 100 filles calculatrices humaines
- En 1943, John W. Mauchly, instructeur d'électronique propose un projet de construction d'un ordinateur électronique en utilisant 18'000 tubes, coûtant 400 mille dollars et appelée Electronic Numerical Integrator and Computer (ENIAC).

Electronic Numerical Integrator and Computer (ENIAC)

RéDS heig-vd

1947 - 1955

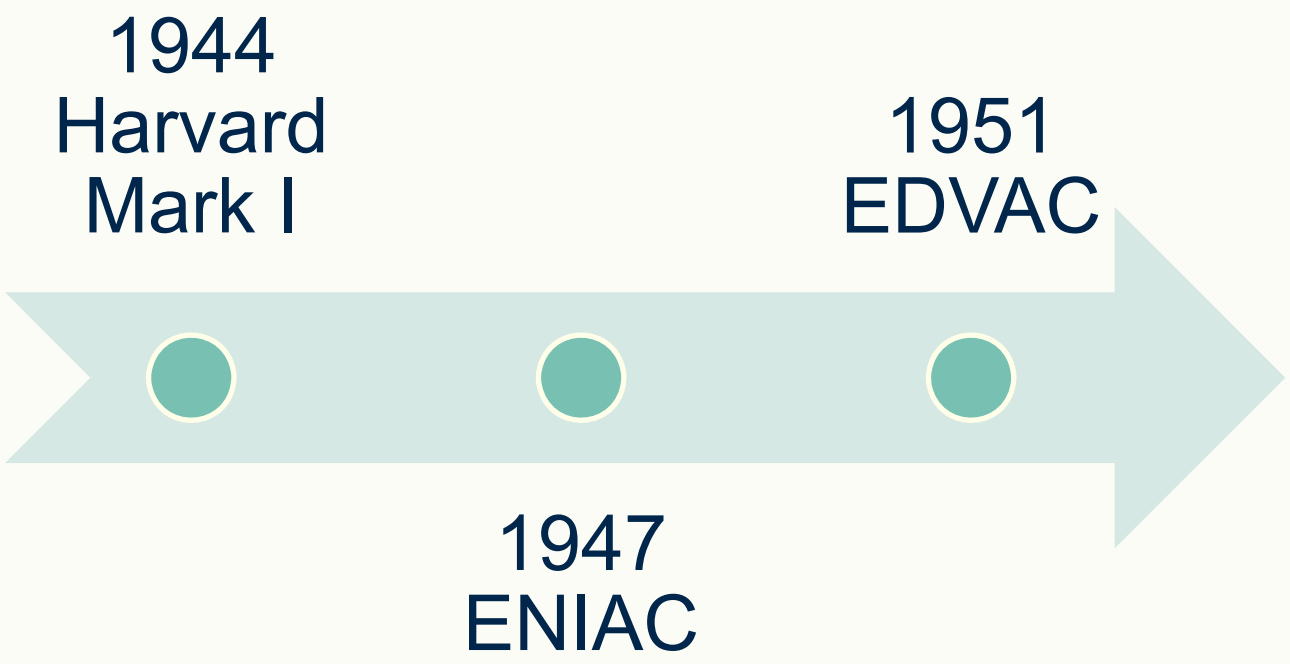
John W. Mauchly



- La principale critique du projet venait de la faible durée de vie des tubes: 3'000 heures, ce qui donnait une panne en moyenne chaque 10 minutes.
- Mais le plus gros problème venait de sa programmation: la vitesse des opérations (5'000 par seconde) empêchait une programmation par lectures des cartes perforées ou bande de papier. Il a été décidé alors de programmer par câblage, ce qui pouvait prendre plusieurs jours par programme
- Le troisième problème était la faible taille de la mémoire
- Un nouveau projet d'ordinateur à programme stocké est proposé: Electronic Discrete Variable Automatic Computer (EDVAC). La structure logique de la machine comprenait 5 parties, nommés par le mathématicien John von Neumann en suivant une métaphore biologique avec l'organisation du cerveau: contrôle central, mémoire, organes d'entrée sortie et unité arithmétique

Chronologie

Ordinateurs



Electronic Discrete Variable Automatic Computer (EDVAC)

1951 – 1960

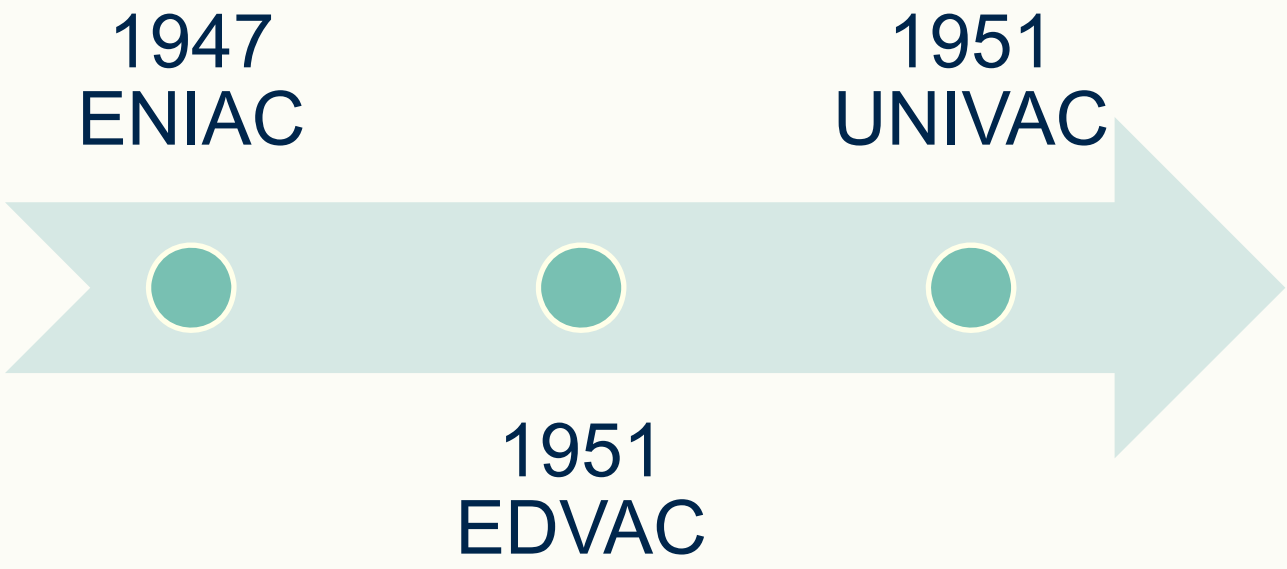
J. Presper Eckert

John William Mauchly

- L'EDVAC est terminé 6 semaines après la fin de la guerre. Et von Neumann est invité partout pour présenter son approche.
- L'inauguration officielle a lieu le 16 février 1946, avec une grande couverture médiatique qui présente l'EDVAC comme un cerveau électronique capable de faire 5'000 opérations par seconde, mille fois plus que la Mark I

Chronologie

Ordinateurs



UNIVAC (Universal Automatic Computer).

1951

J. Presper Eckert

John Mauchly

L'ordinateur comme machine de bureau

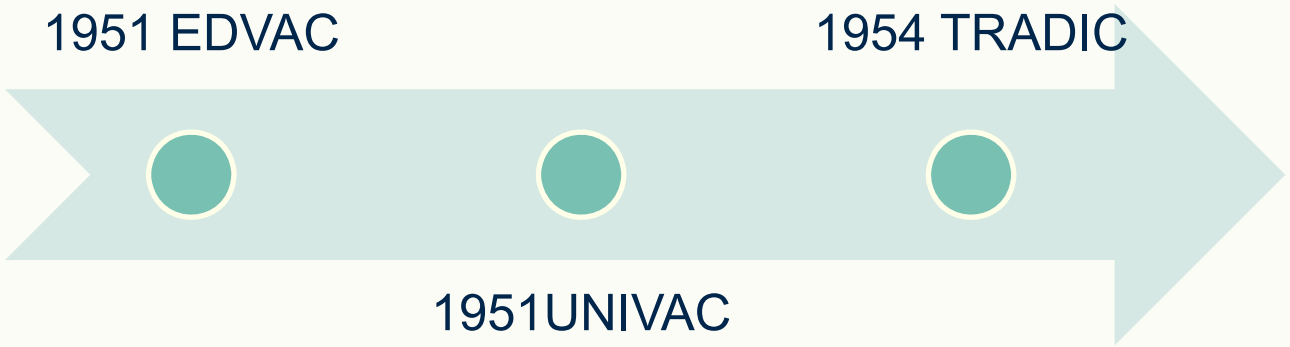
- Au début des années 50, une trentaine de compagnies aux USA et une dizaine en Angleterre entrent au marché de l'ordinateur, vendu comme machine de traitement de données plus que comme outil mathématique pour des applications scientifiques
- En 1946, le bureau du recensement commande une machine appelée UNIVAC (Universal Automatic Computer). La caractéristique la plus ambitieuse est le remplacement des cartes perforées par des bandes magnétiques pour le stockage des données
- En 1949 il y a 6 commandes, pour un total de 1.2 millions de dollars. UNIVAC a commencé à marcher au début de 1951. Le logiciel faisant défaut, ils engagent une programmeure du *Harvard Computer Laboratory*: Grace Murray Hopper
- En 1952, CBS accepte d'utiliser UNIVAC pour prédire les résultats de l'élection présidentielle.

IBM

- Entre temps, les ingénieurs d'IBM travaillaient à l'introduction de l'électronique dans leurs machines de bureau. Le premier modèle électronique est le 601, multiplicateur qui fait 100 multiplications par minute en 1946. Son successeur est utilisé pour calculer des trajectoires de missiles. On l'appela "*calculator*", pour éviter que le nom "*computer*" évoque une idée d'humain remplacé et mis au chômage…
- Le premier modèle d'ordinateur non scientifique est le IBM 702 appelé Electronic Data Processing Machine

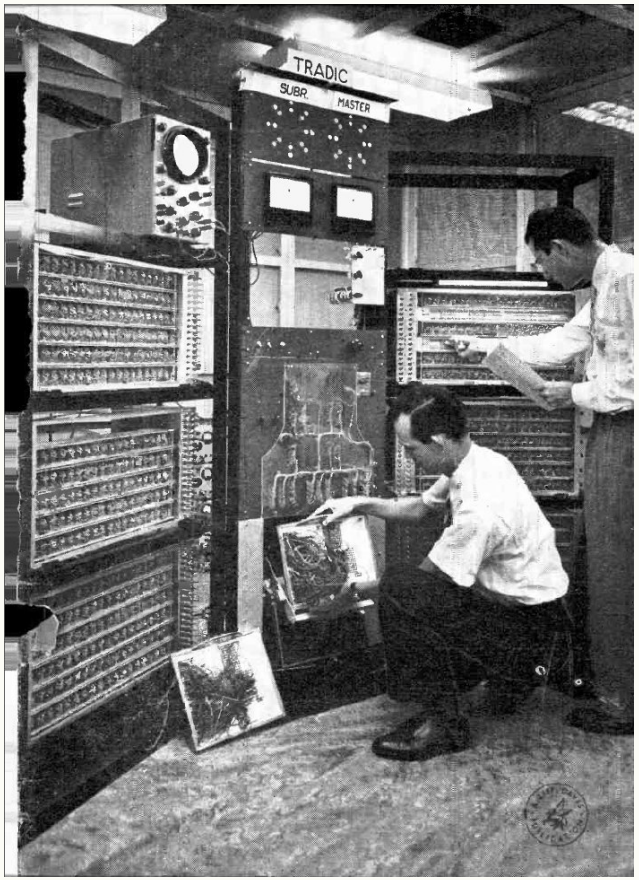
Chronologie

Ordinateurs



TRADIC (TRAnsistor DIgital Computer).

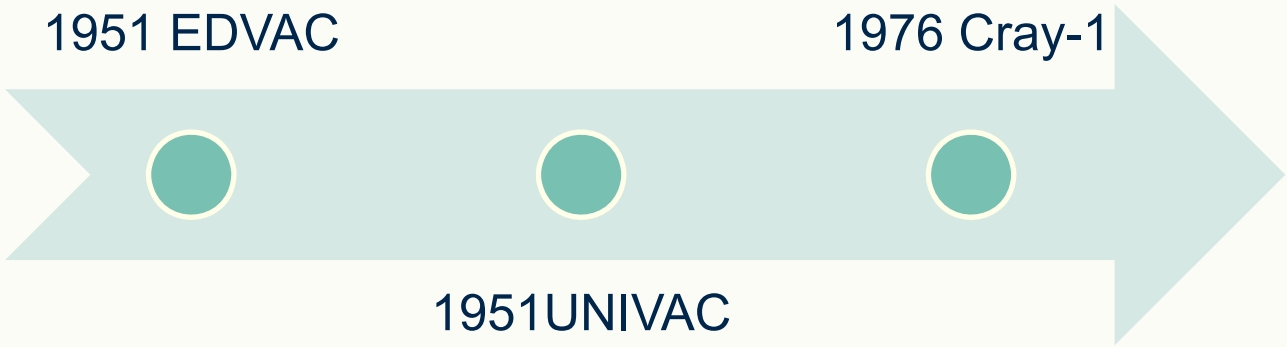
- **TRADIC : premier ordinateur à transistors**



Il comportait 700 transistors et 10000 diodes.
Il a été mis en service en 1954 dans les laboratoires Bell à la demande de l'US Air Force

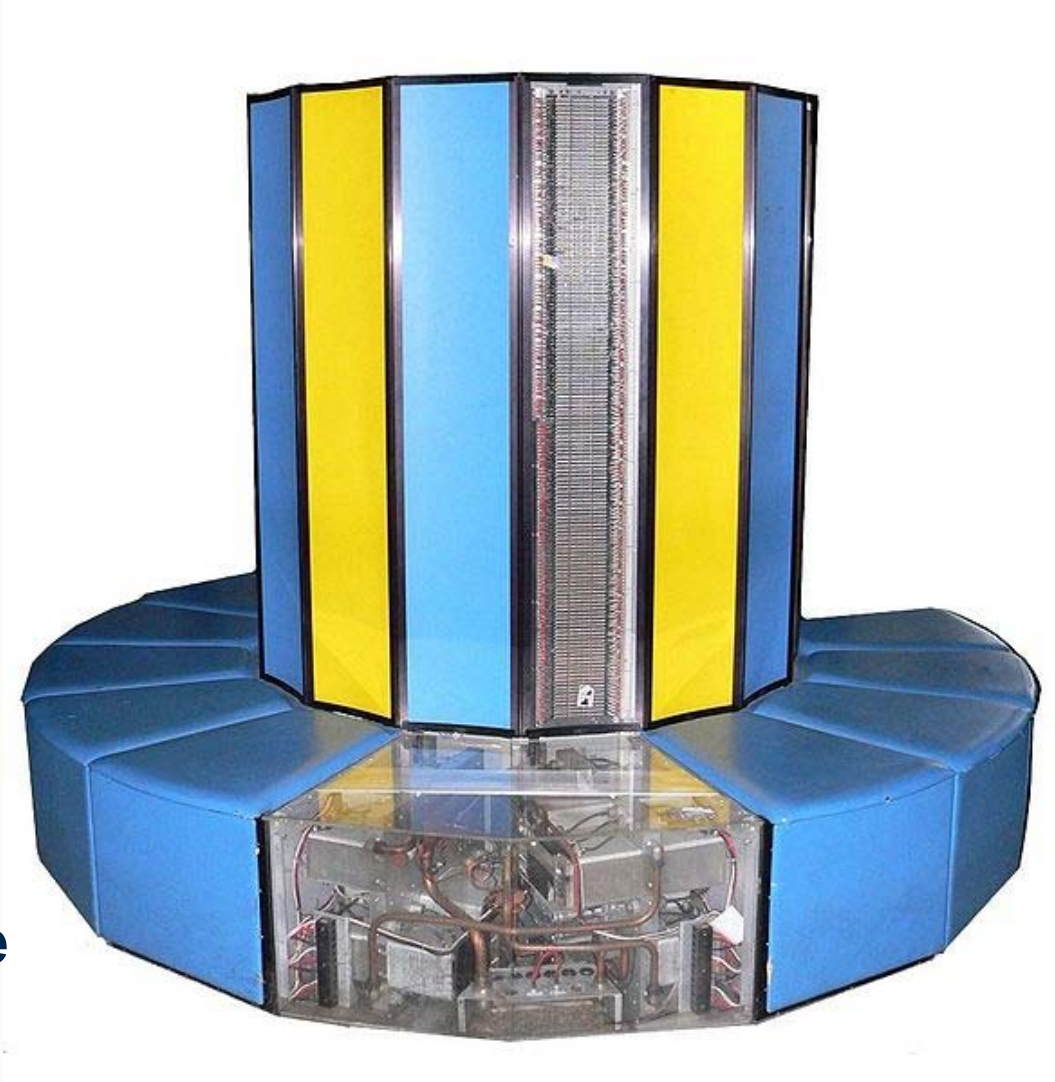
Chronologie

Ordinateurs



Cray-1

- 1976
- Seymour Cray



- A voir: Le musée Bolo à l'EPFL

Historique

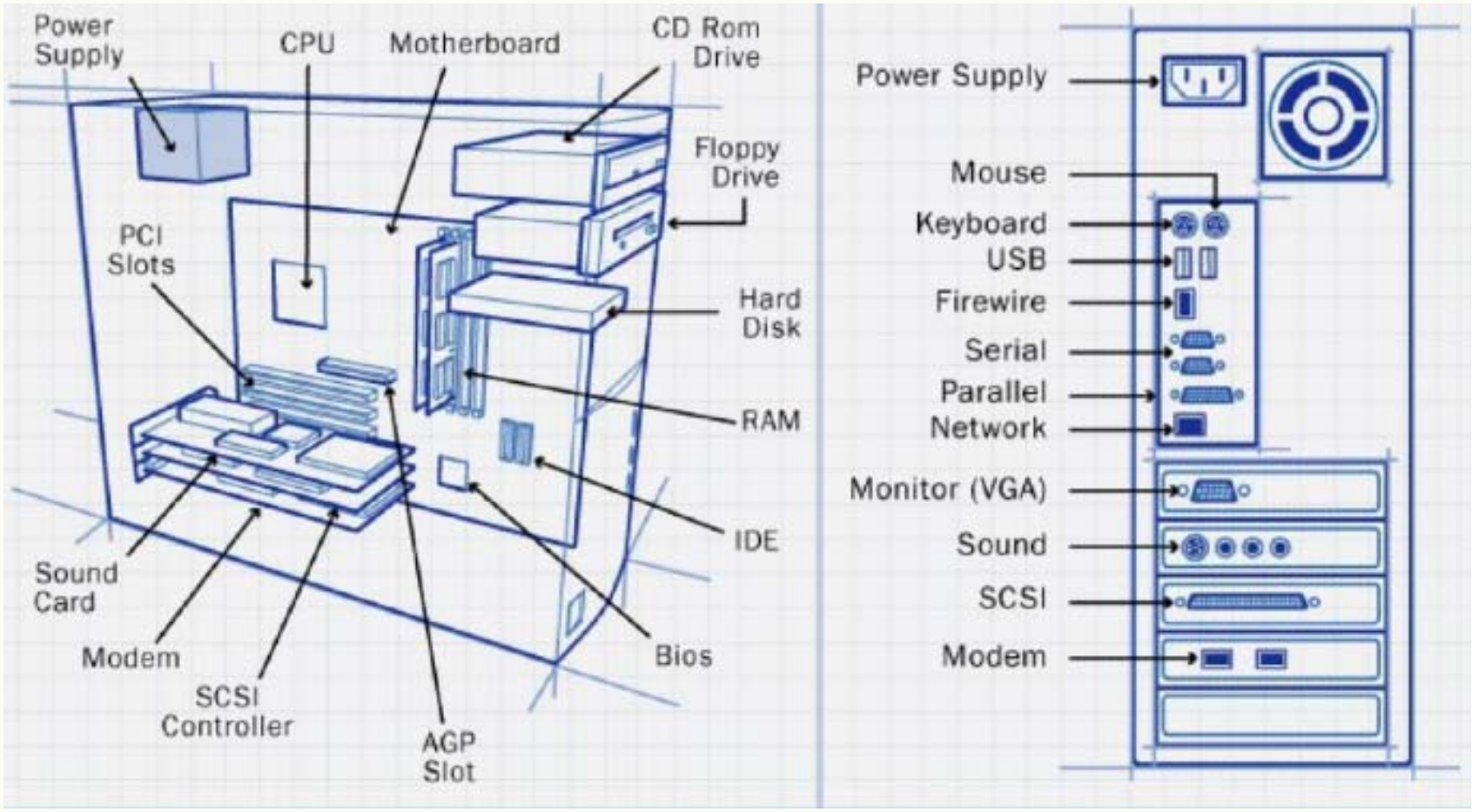
Année	Nom	Taille m3	Puis-sance W	Perfor-mances (add/s)	Mém. Ko	Prix $	Rapport Perfor-/Prix
1946	ENIAC	60	174’000	5000	≈ 0,1	486’000	1
1964	IBM S360	1,699	10'000	500'000	64	1'000'000	263
1965	PDP-8	0,227	500	330'000	4	16'000	10'855
1976	Cray-1	1,642	60'000	166'000'000	32'768	4'000'000	2'842
1981	IBM PC	0,0283	150	240'000	256	3'000	42'105
1991	HP 9000	0,0566	500	50'000'000	16'384	7'400	3'556'188
2000	PC	0,0250	150	≈1'000'000'000	256'000	1'000	≈526'315'789

Voir poster:

Révolution matériel PC de 1977 – 1997 en A07

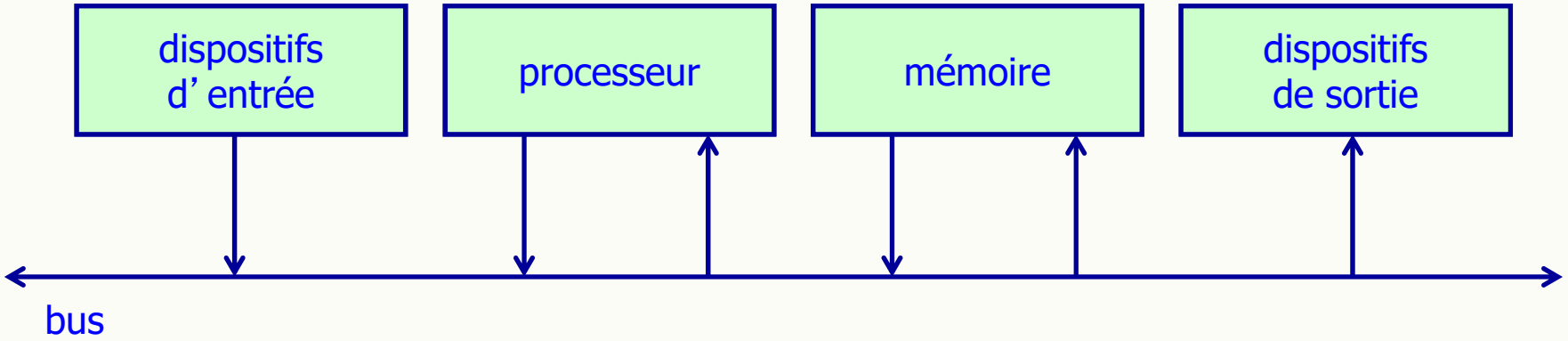
ARCHITECTURE GÉNÉRALE D'UN ORDINATEUR

L'ordinateur



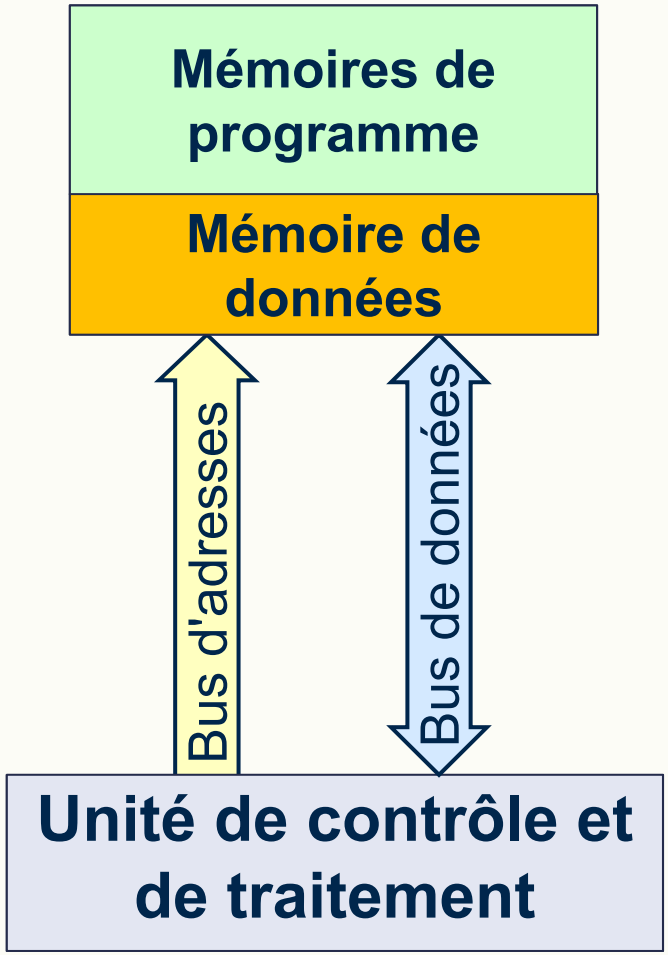
Architecture générale

- **On peut voir 4 grandes parties dans un ordinateur:**



- Un ordinateur est une machine électronique composée de plusieurs parties interconnectées par des fils (bus)
- A tout moment, tout fil dans l'ordinateur se trouve à un voltage haut ou bas. La valeur réelle n'intéresse pas: c'est seulement un 1 ou un 0

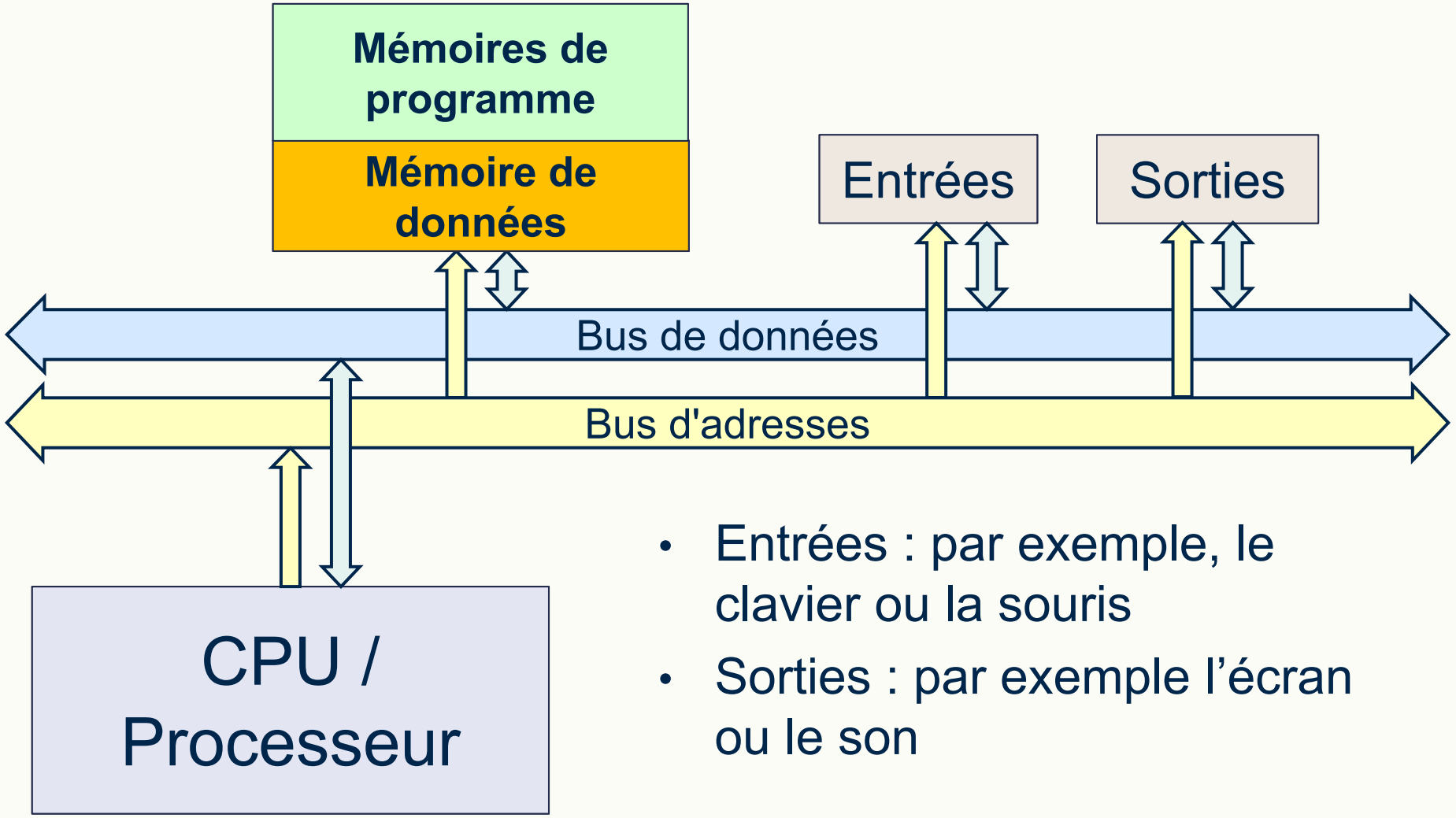
Architecture Von Neumann



- Les instructions et les données sont regroupées dans la même mémoire à des adresses différentes
- Les instructions et les données empruntent le même bus

Aussi appelé « processeur », en anglais : CPU
Central Processing Unit

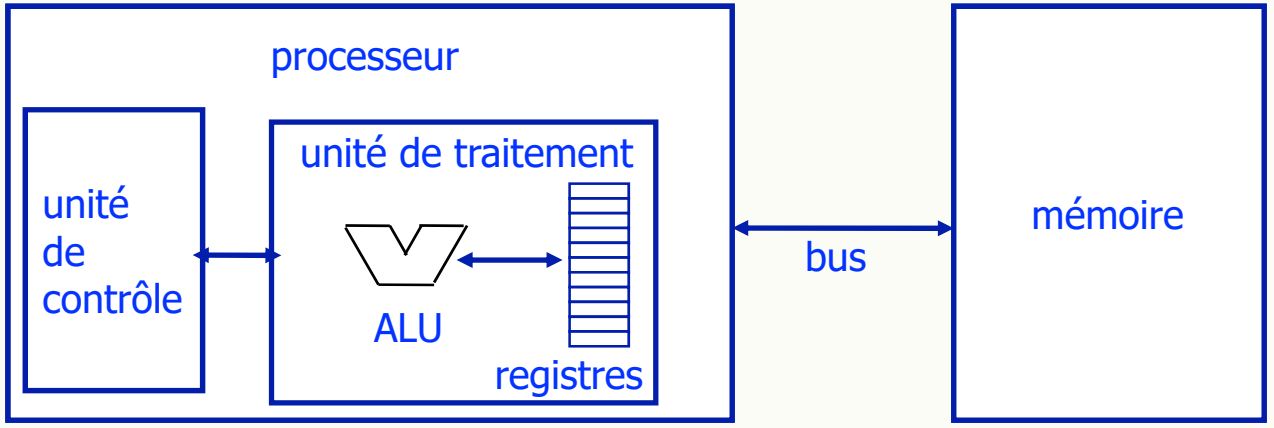
Architecture Von Neumann avec entrées /sorties



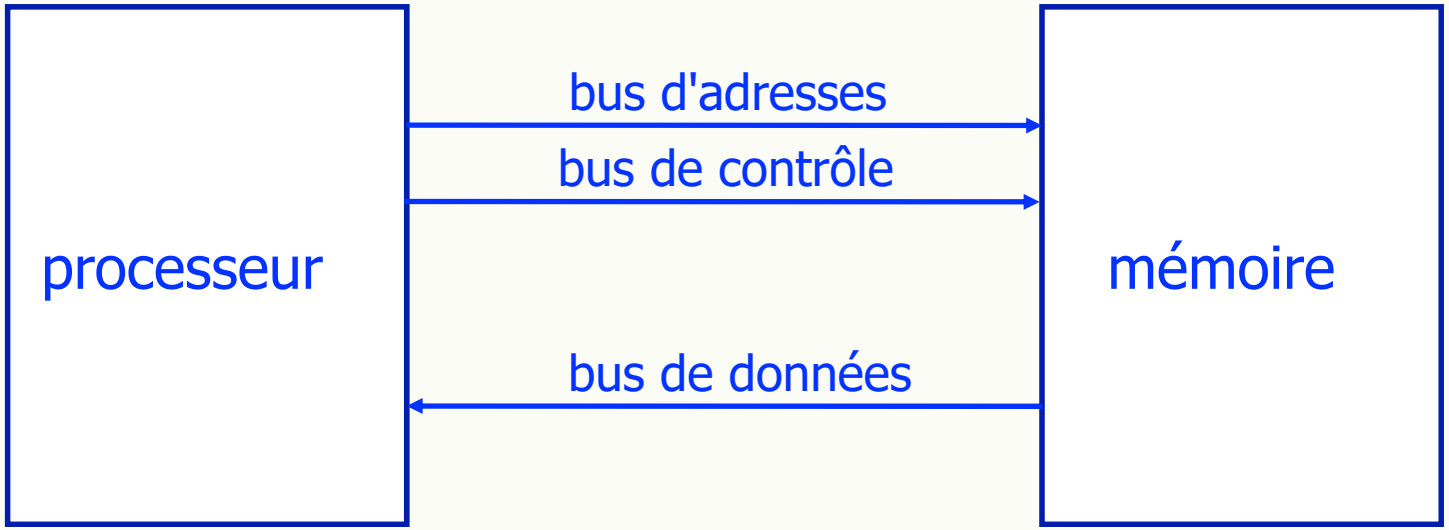
- Entrées : par exemple, le clavier ou la souris
- Sorties : par exemple l'écran ou le son

Architecture de von Neumann

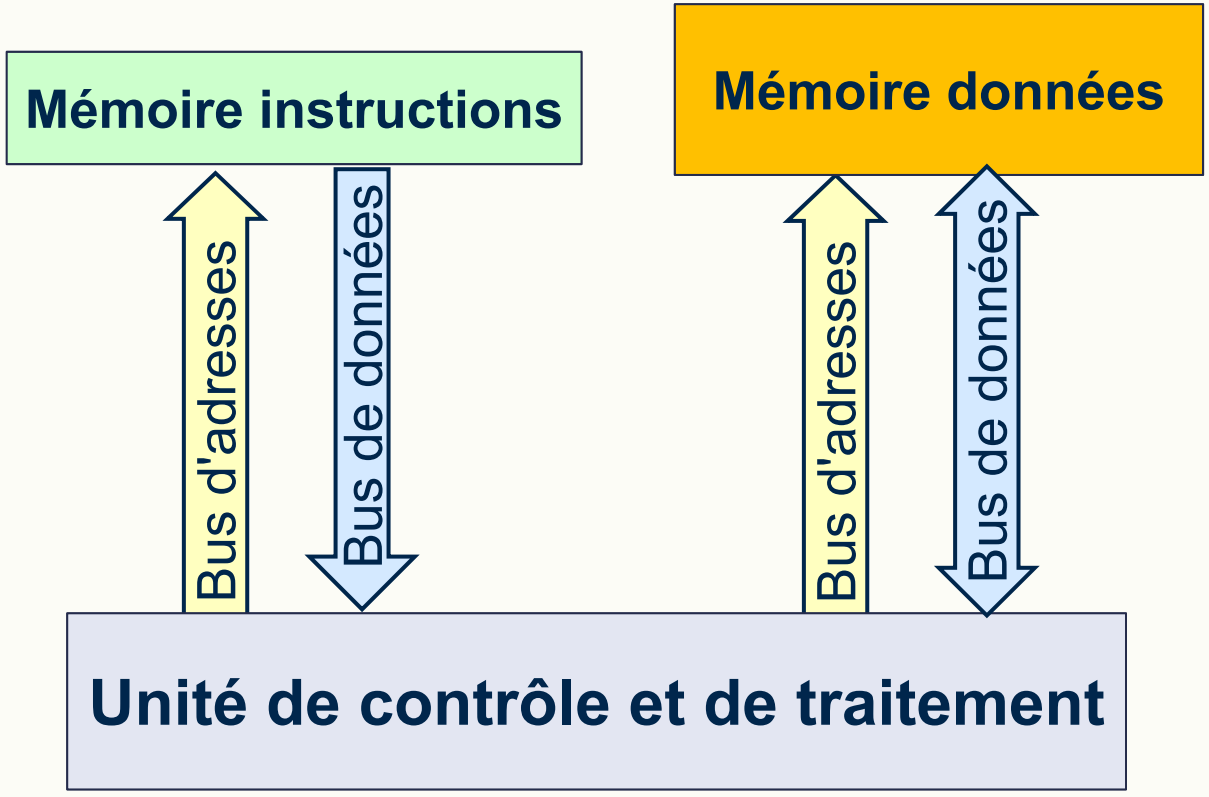
- Le processeur est composé de deux parties: l'unité de traitement, qui contient l'ensemble d'opérateurs arithmétiques et logiques, groupés autour d'une ou plusieurs ALUs (Arithmetic and Logic Unit); et l'unité de contrôle qui coordonne les différentes activités du processeur
- Les données à traiter et les instructions sont stockées dans la même mémoire



- Pour lire une donnée ou une instruction, le processeur doit envoyer une adresse à la mémoire par le biais du bus d'adresses
- La mémoire répond en envoyant l'information demandée par le biais du bus de données

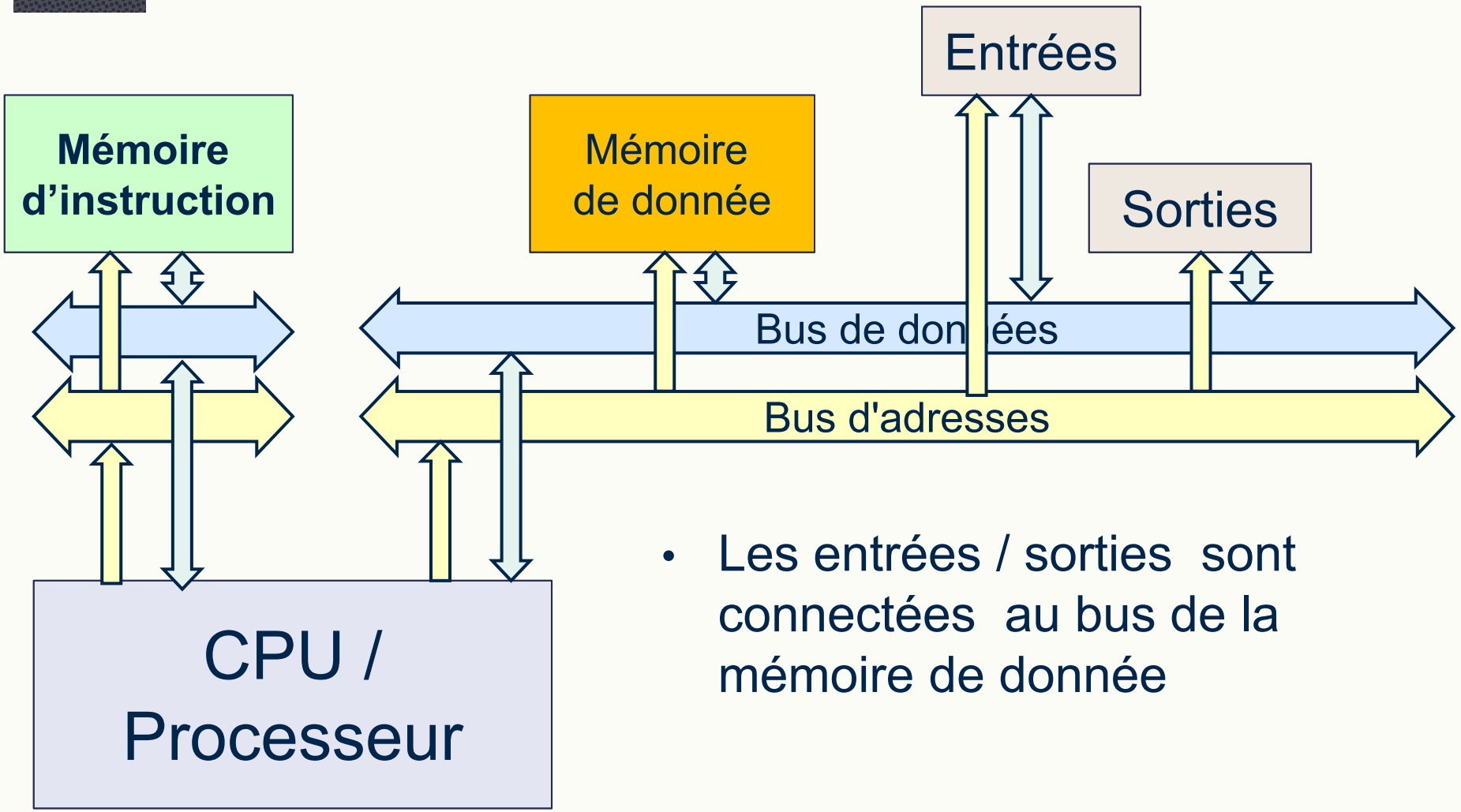


Architecture Harvard



- Une mémoire pour les instructions et une pour les données
- Avantage : accès simultané aux instructions et aux données

Architecture Harvard avec entrées /sorties



- Les entrées / sorties sont connectées au bus de la mémoire de donnée

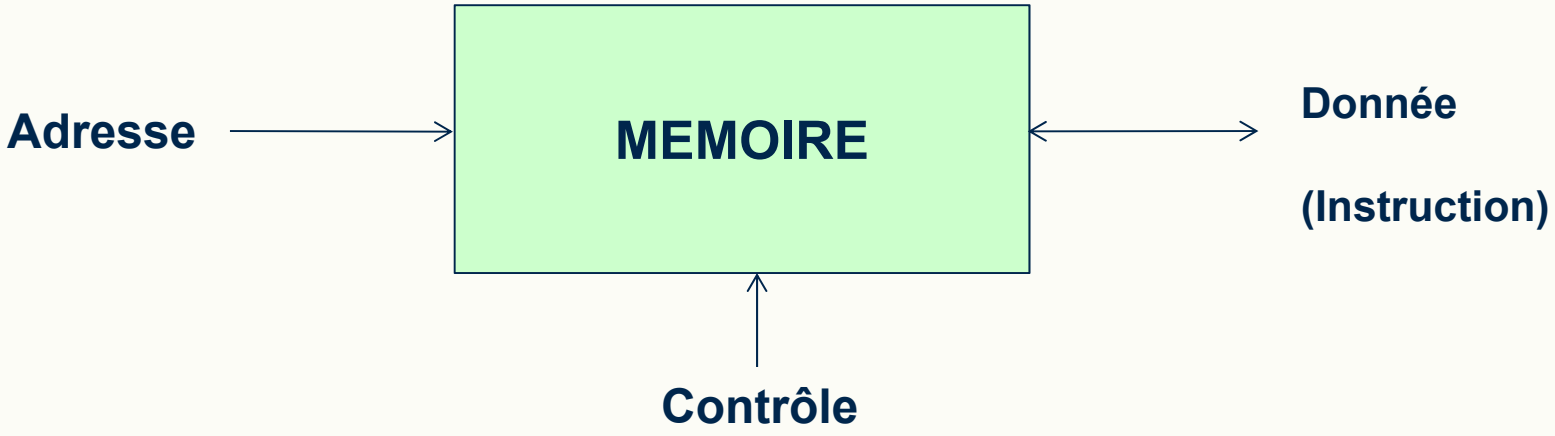
ORDINATEUR, MÉMOIRE ET PROCESSEUR

Ordinateur et mémoire

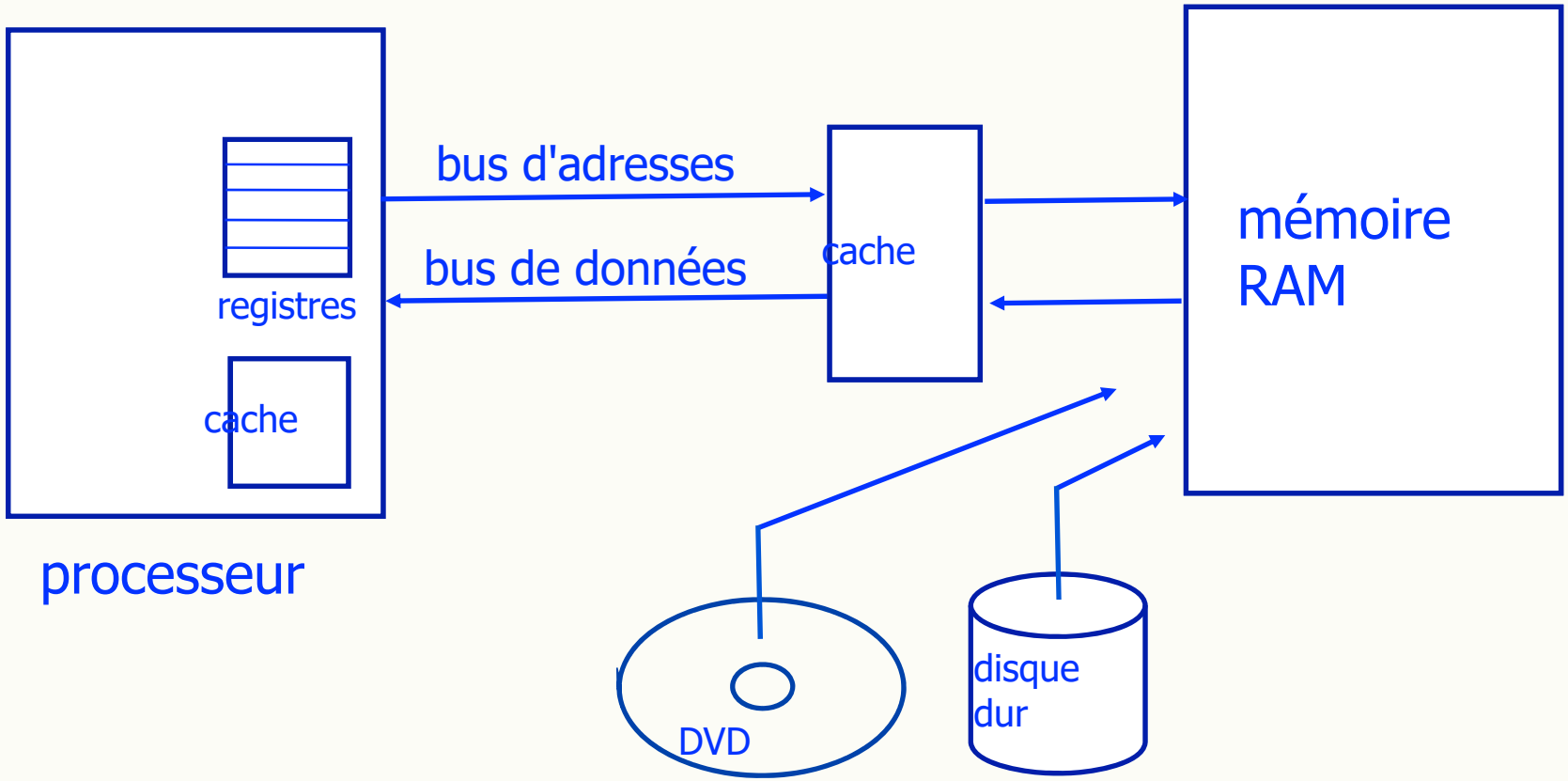
- Un programme est une suite d'instructions stockés dans une mémoire
- Les opérations arithmétiques et logiques sont exécutées [en fonction du programme] sur des données stockées en mémoire
- Les instructions ou les données sont stockées sous forme de mots binaire

La mémoire

- La mémoire est un dispositif [électronique] servant à stocker des mots binaires
- Chaque mot à une adresse unique

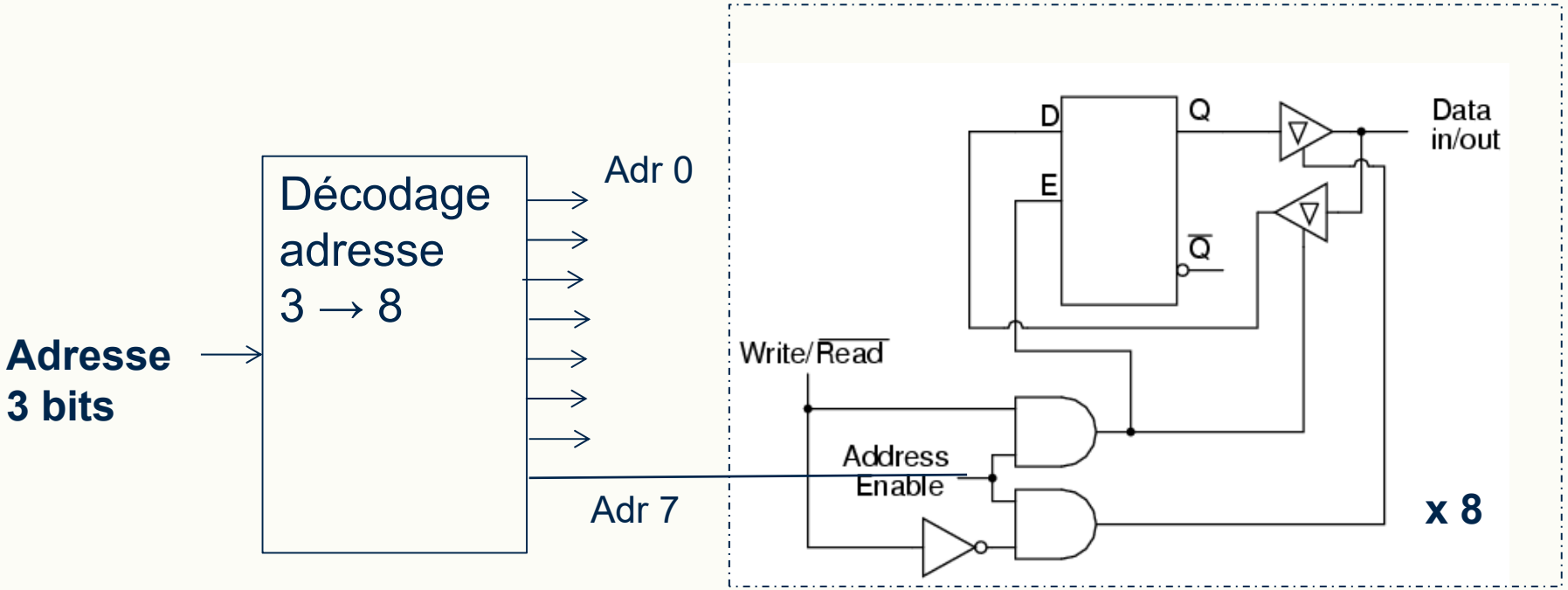


Hiérarchie des mémoires



Cellule de base de mémoire

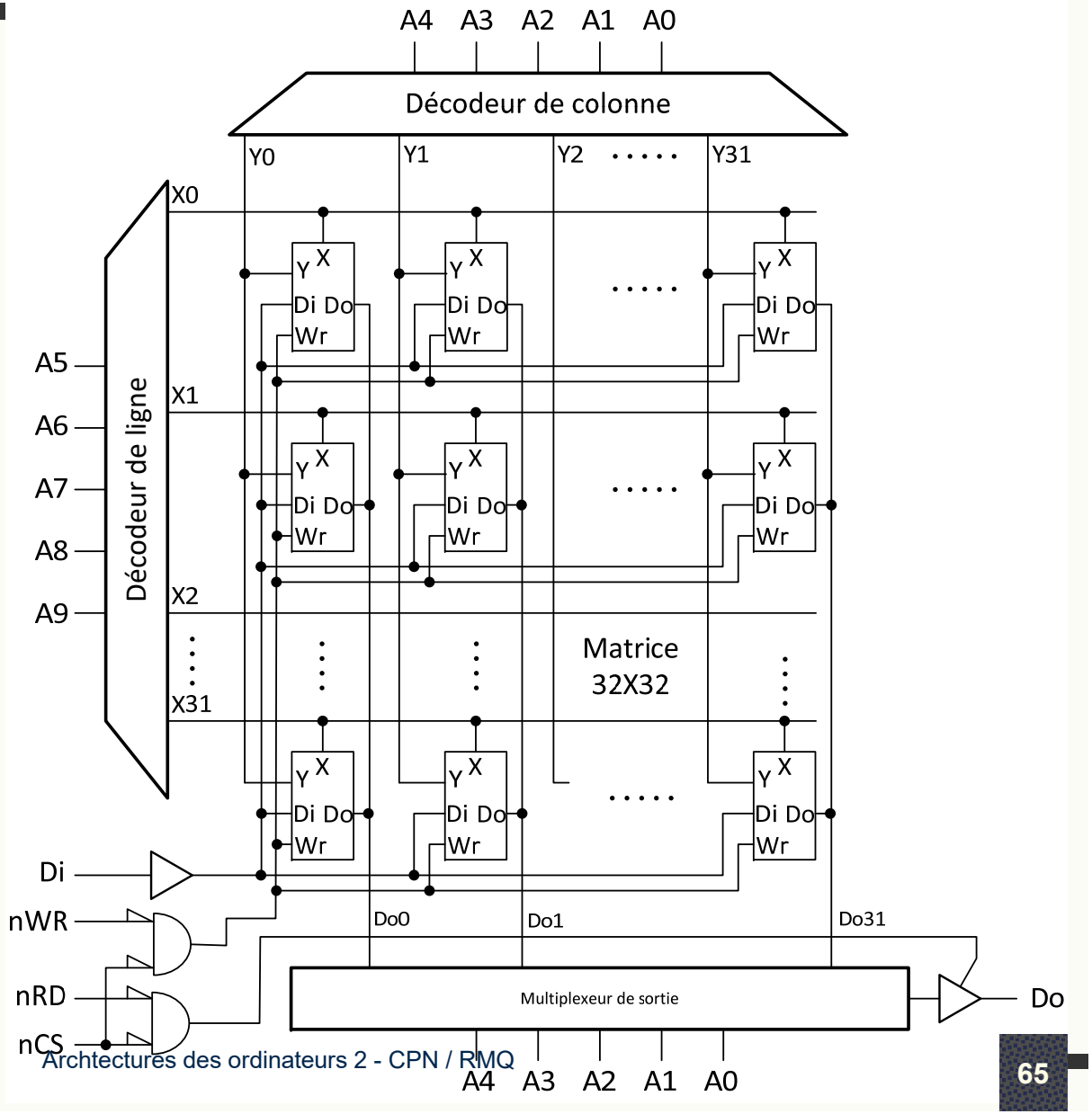
- Exemple de mémoire 8 x 1 bit : une cellule de base peut être réalisée avec un registre :



Structure d'une RAM statique

Construction matricielle, meilleure utilisation de la surface d'une puce (IC: circuit intégré)

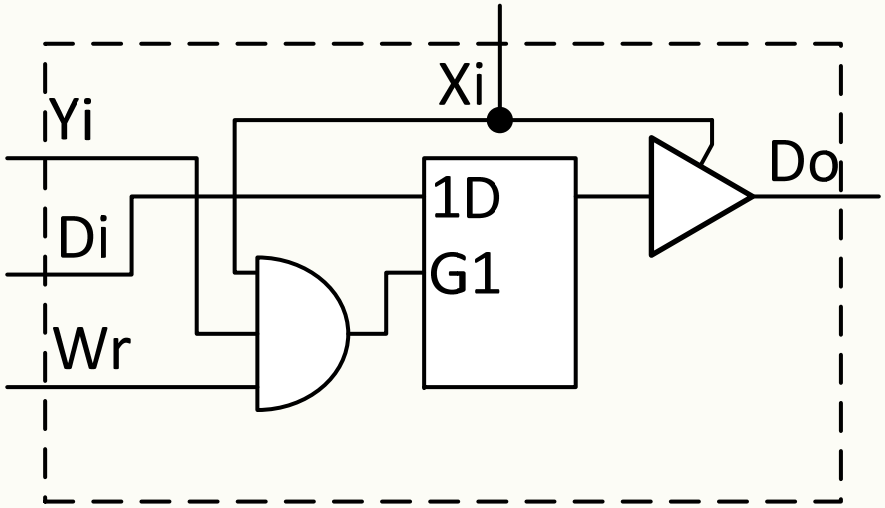
RAM 1024 bits
matrice 32x32



Structure d'une RAM statique

- L'information est mémorisée dans un latch

- Structure d'une cellule
  - mémorisation de Di si Wr • Yi • Xi actif
  - lecture latch si Xi actif



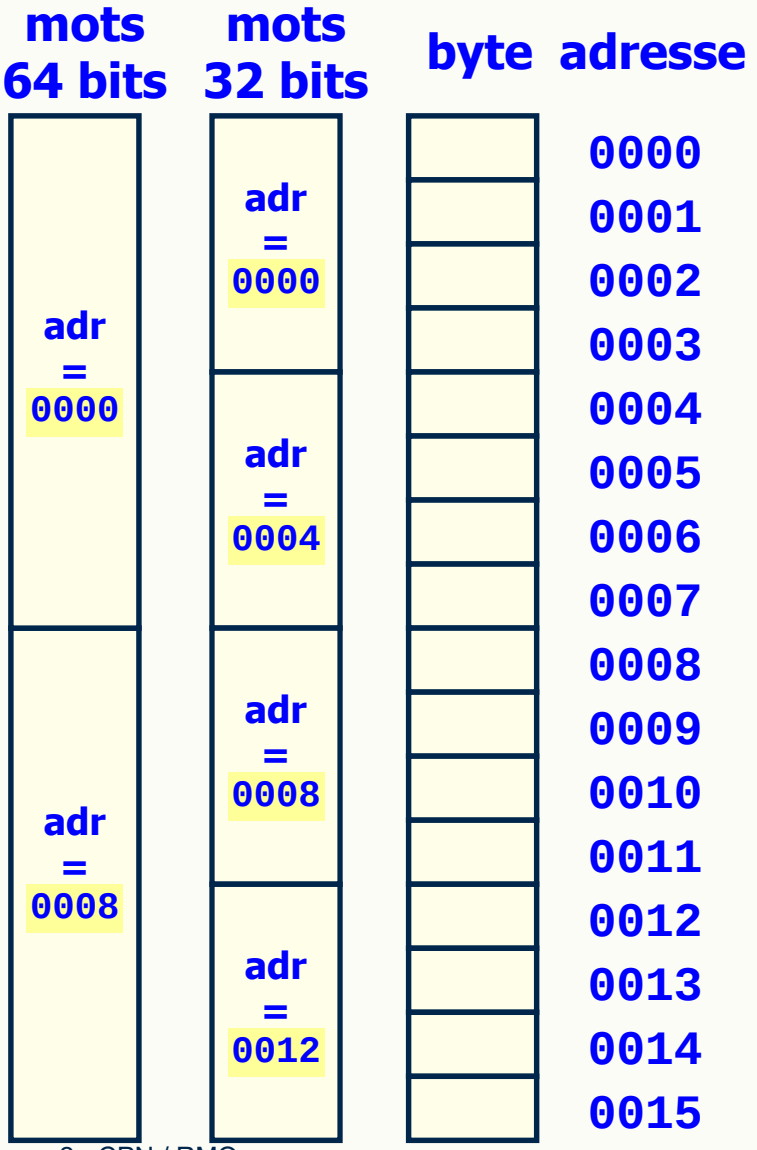
Mots et taille mémoire

- Une mémoire est organisée par mots binaire en général multiples de 8 :
  - octet / byte (mots de 8 bits)
  - mots de 16, 32, 64, 128 .....

- La taille d'une mémoire est donnée en byte
  - Kilobyte : 1 KB = 2^{10} bytes = 1024 bytes
  - Megabyte : 1 MB = 2^{20} bytes = 1024 KB
    1'048'576 bytes
  - Gigabyte : 1 GB = 2^{30} bytes = 1024 MB
    1'048'576 KB
    1'073'741'824 MB

Adressage mémoire

- Une mémoire d'ordinateur est toujours adressé par bytes
- Si une donnée contient plus d'un byte, l'adresse de la donnée correspond à celle du premier byte
- Les mots sont toujours alignés dans la mémoire
  Adr = N x nb_bytes /mot
  (N est entier)



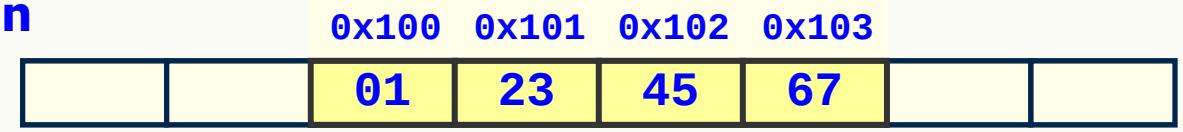
Rappel : Little endian / big endian

- big endian:
  - le byte de poids fort est mis à l'adresse inférieure (le mot commence par le byte de poids fort)
- little endian:
  - le byte de poids faible est mis à l'adresse inférieure (le mot commence par le byte de poids faible)

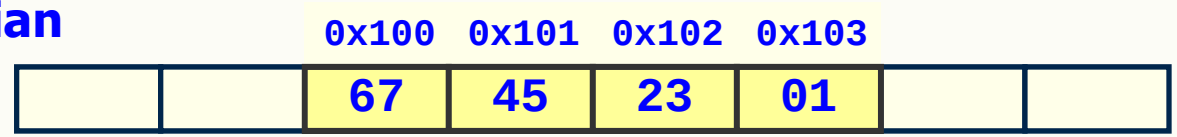
Rappel: Little endian / big endian

- Exemple:
  - Une variable toto est stockée à l'adresse 0x0100 (c'est-à-dire &toto=0x0100)
  - La valeur de la variable toto est 0x01234567

big endian



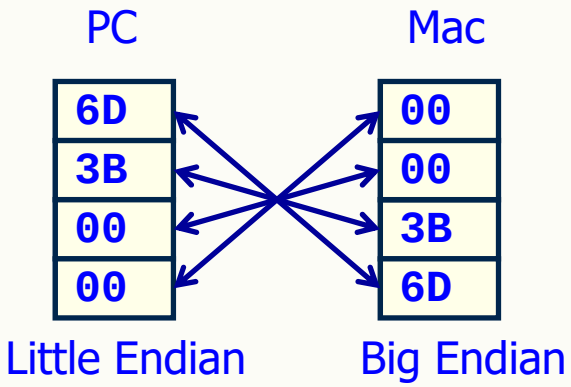
little endian



Exemple, sur un ordinateur 32 bits

```
int toto = 15213;
```

décimal: **15213**
binaire: **0011 1011 0110 1101**
hexadécimal: **3 B 6 D**

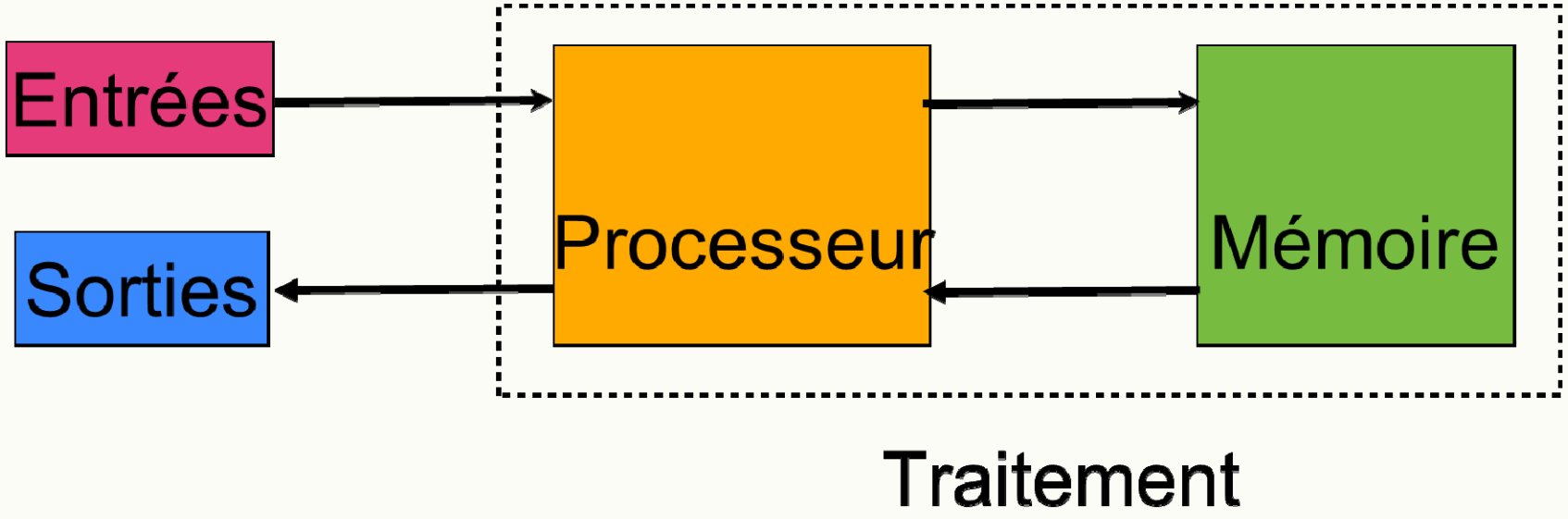


Ordinateur et processeur

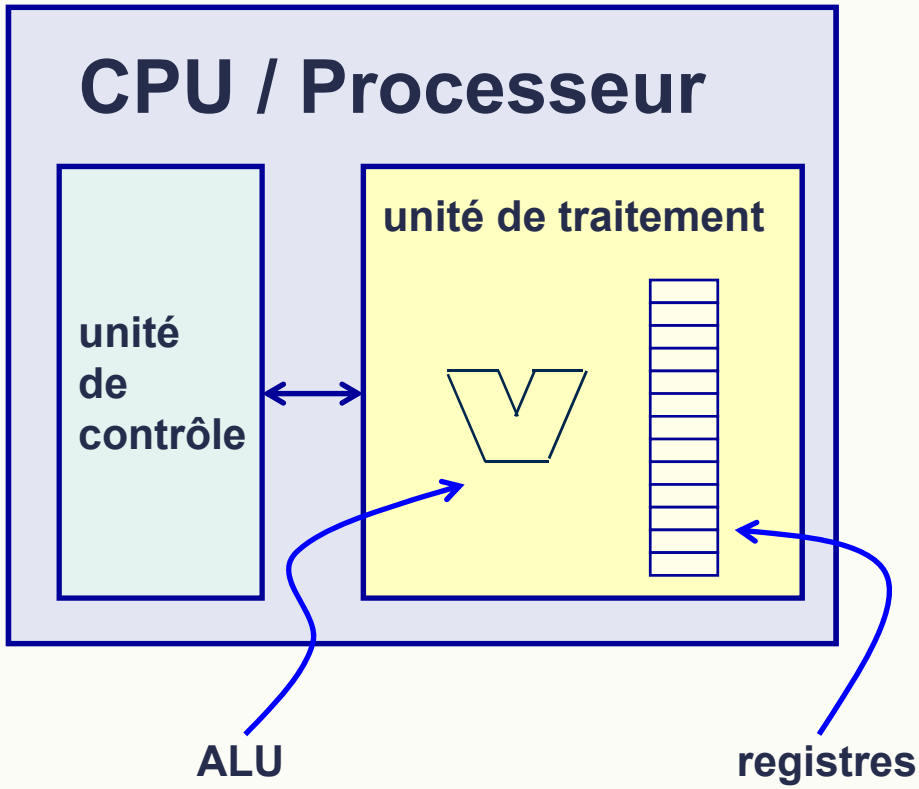
- **Processeur: définitions**
  - **Un processeur (ou unité centrale de traitement, UCT, en anglais central processing unit, CPU) est un composant présent dans de nombreux dispositifs électroniques (informatique) qui exécute les instructions machine des programmes informatiques.**
  - **C'est la pièce maîtresse de l'ordinateur. Son but est de contrôler et de traiter l'ensemble des requêtes émises par l'utilisateur ou par les pièces de l'ordinateur afin que chacune puisse exécuter sa tâche lorsque nécessaire.**

Ordinateur et processeur

- Le processeur dirige le traitement de l'information et réalise ce traitement.
  Diriger, c'est décider quelle tâche exécuter et dans quel ordre, contrôler toutes les autres parties

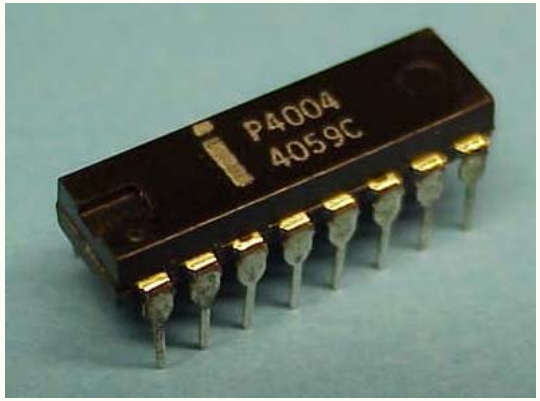


Architecture du processeur



- Unité de contrôle:
  - Déroulement du programme
  - Contrôle unité de traitement
- Unité de traitement:
  - L'ALU (Arithmetic and Logic Unit)
  - Les registres (éléments mémoire)

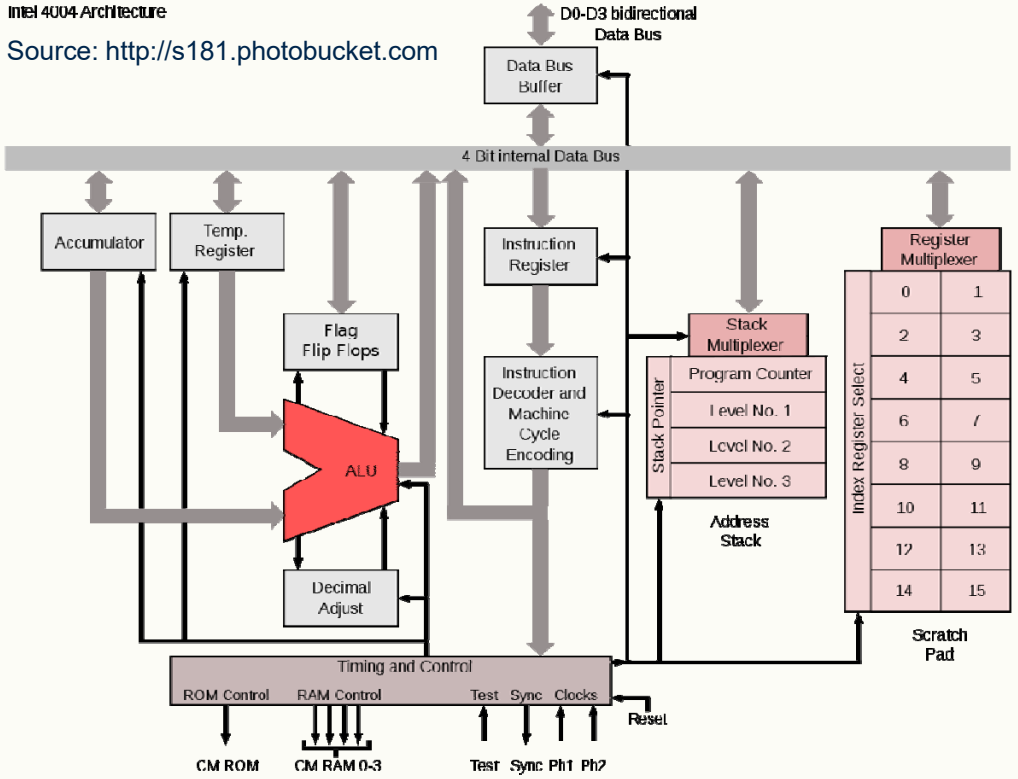
Processeurs Intel



- 1971
- 4-bits microprocessor 4004
- 2300 transistors, 10 microns

Intel 4004 Architecture

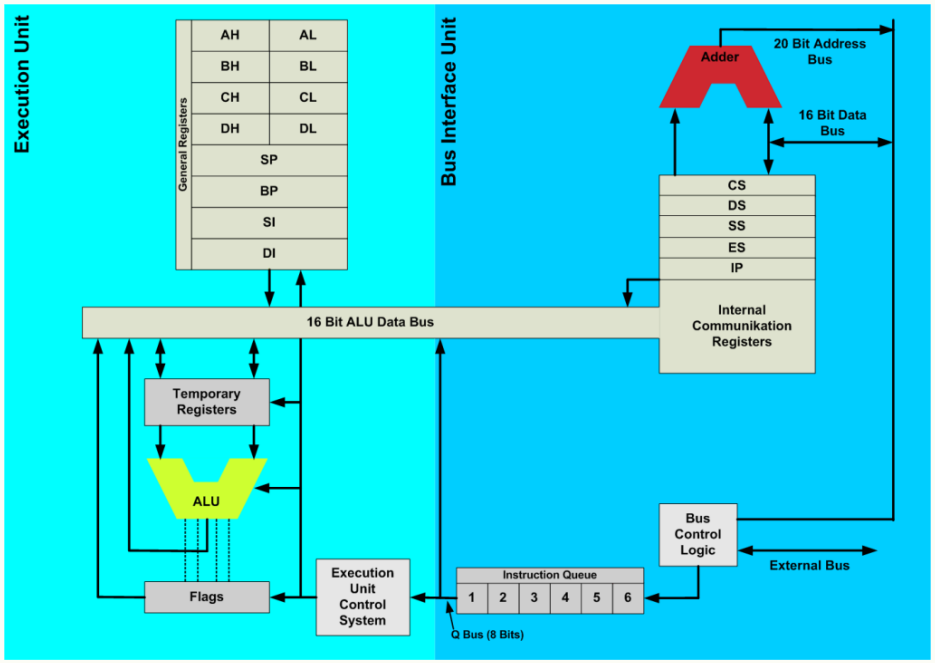
Source: http://s181.photobucket.com



- 1972
- 8-bits 8008
- puis 8080

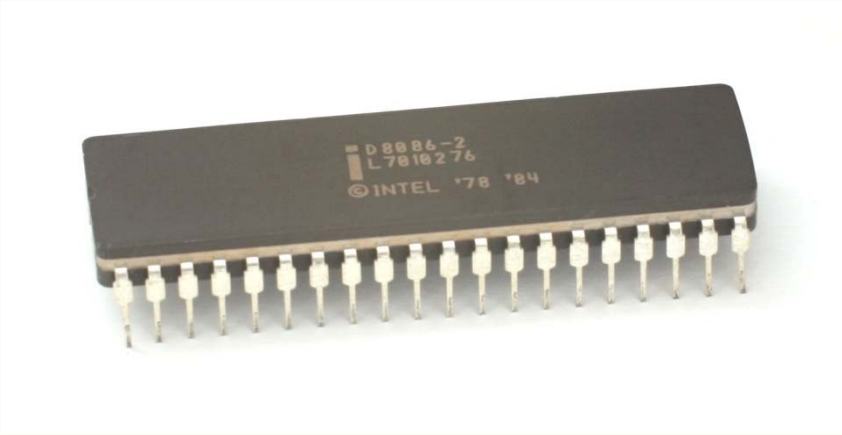
Intel

Processeurs Intel



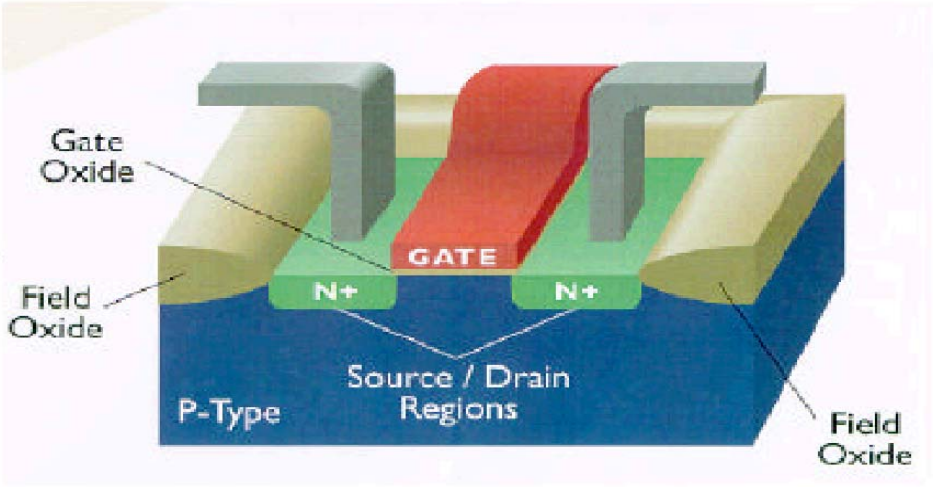
Source: http://code.google.com/p/fpga-x86-processor/wiki/Architecture

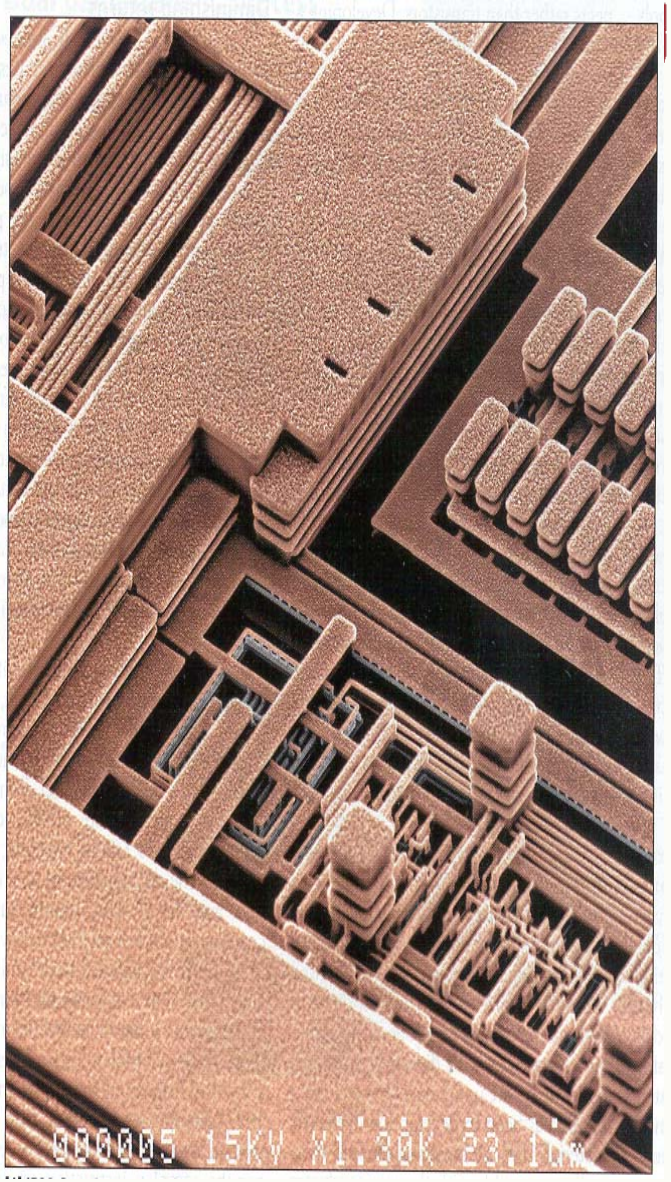
- 16-bits microprocessor
- 8086
- 1er de la série x86 !
- 1978, 360$
- 29’000 transistors
- 3 microns



Intel

- Cette évolution a été permise par l'évolution de la technologie, qui a permis une miniaturisation croissante des dispositifs électroniques (le transistor)

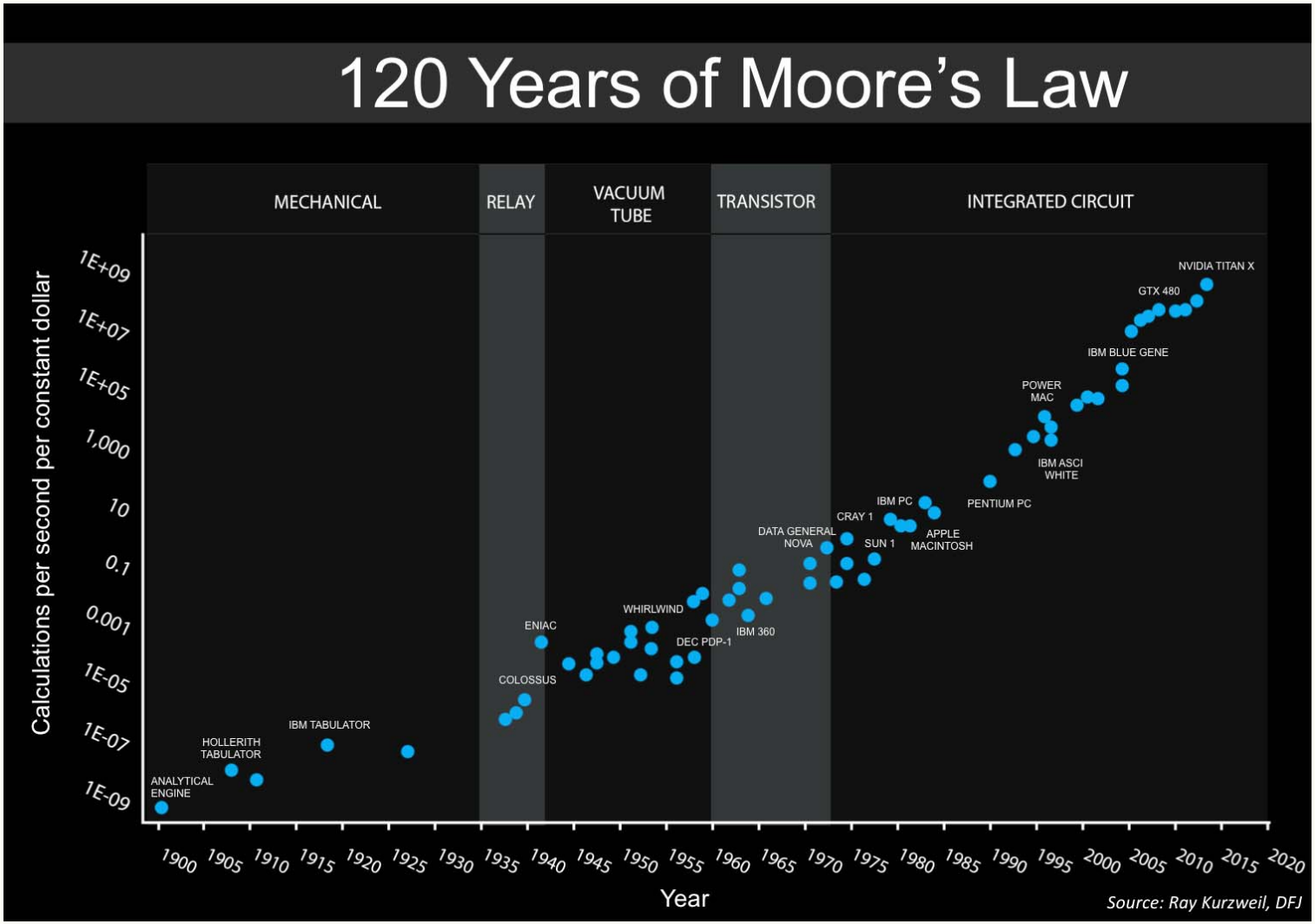




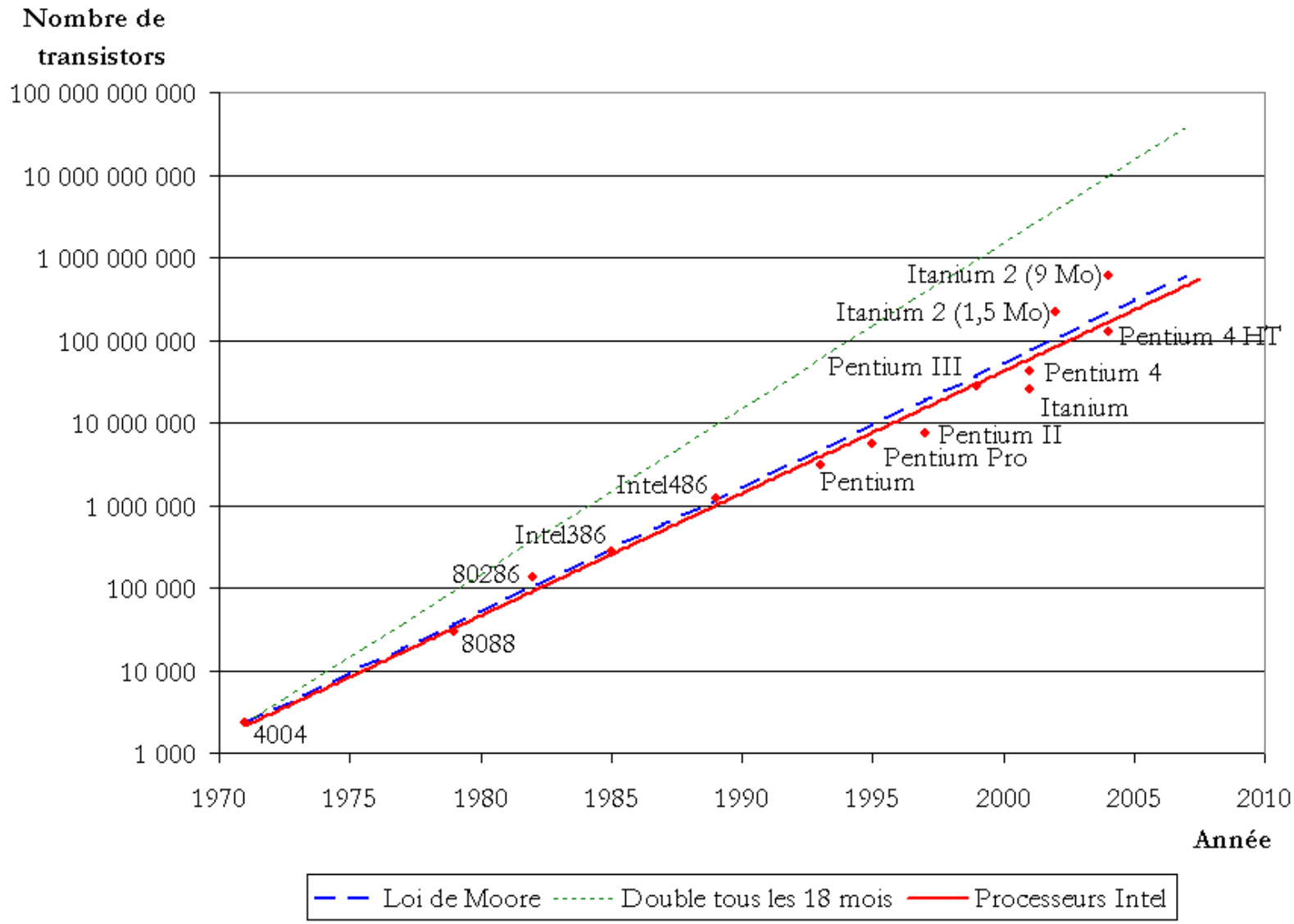
Loi de Moore

- En avril 1965, 6 ans après l'invention du circuit intégré, Gordon Moore, co-fondateur d'Intel 3 ans plus tard, a fait une prédiction connue plus tard comme *la loi de Moore*: le nombre de transistors d'un circuit intégré doublera chaque année
- A ce moment, l'équipe de Moore travaillait à la conception d'un circuit à ... 60 transistors
- En 1975, Moore a fait passer à 2 ans la durée du cycle
- Dès la fin des années 80, la durée du cycle est de 18 mois. Et la loi est appliquée à tous les paramètres de la technologie, notamment vitesse et performance

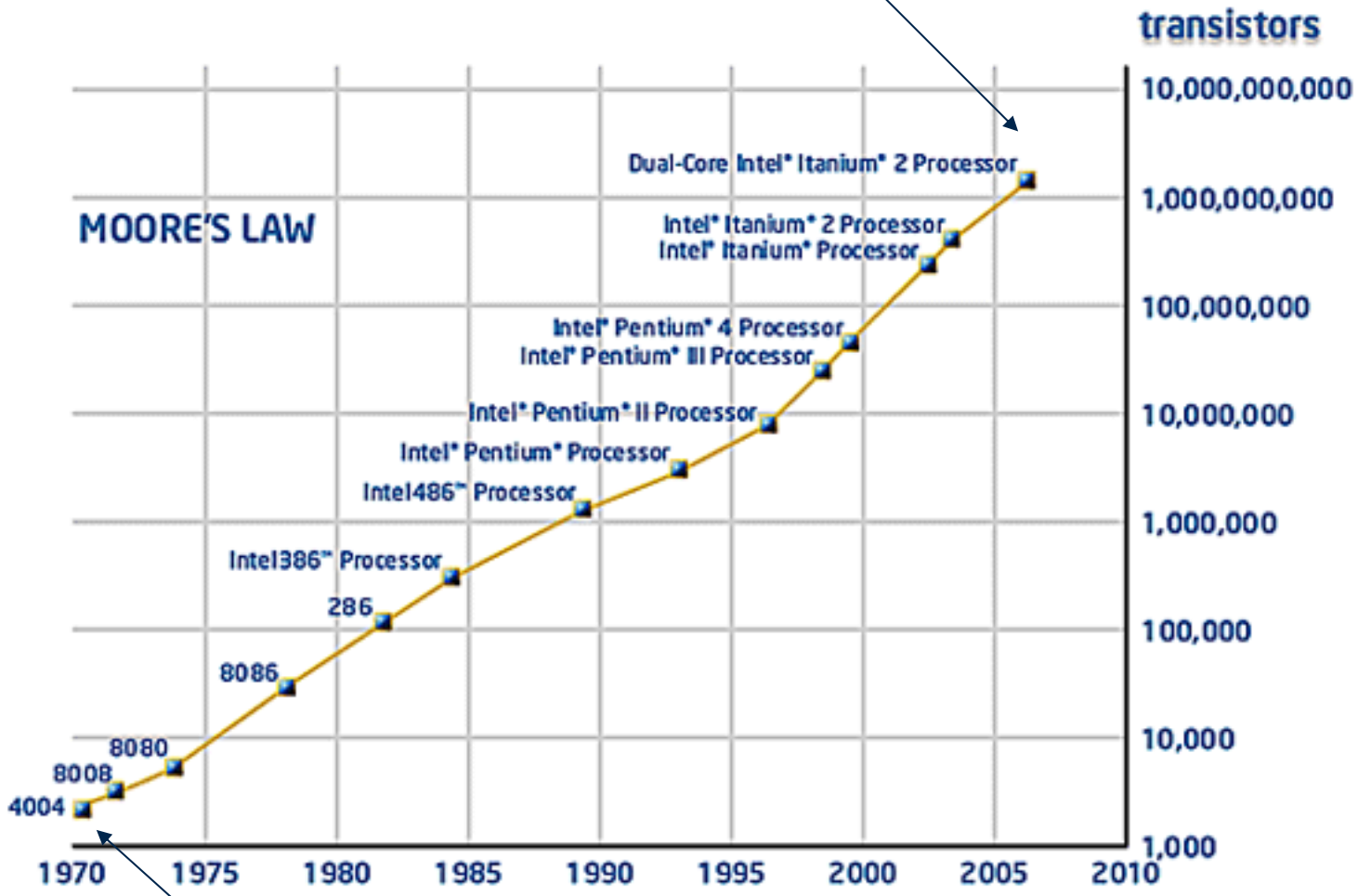
Loi de Moore


120 Years of Moore's Law
MECHANICAL
RELAY
VACUUM TUBE
TRANSISTOR
INTEGRATED CIRCUIT
Calculations per second per constant dollar
1E+09
1E+07
1E+05
1,000
10
0.1
0.001
1E-05
1E-07
1E-09
ANALYTICAL ENGINE
HOLLERITH TABULATOR
IBM TABULATOR
COLOSSUS
ENIAC
WHIRLWIND
DEC PDP-1
IBM 360
DATA GENERAL NOVA
CRAY 1
SUN 1
IBM PC
APPLE MACINTOSH
PENTIUM PC
IBM ASCI WHITE
POWER MAC
IBM BLUE GENE
GTX 480
NVIDIA TITAN X
1900 1905 1910 1915 1920 1925 1930 1935 1940 1945 1950 1955 1960 1965 1970 1975 1980 1985 1990 1995 2000 2005 2010 2015 2020
Year
Source: Ray Kurzweil, DFJ

Loi de Moore

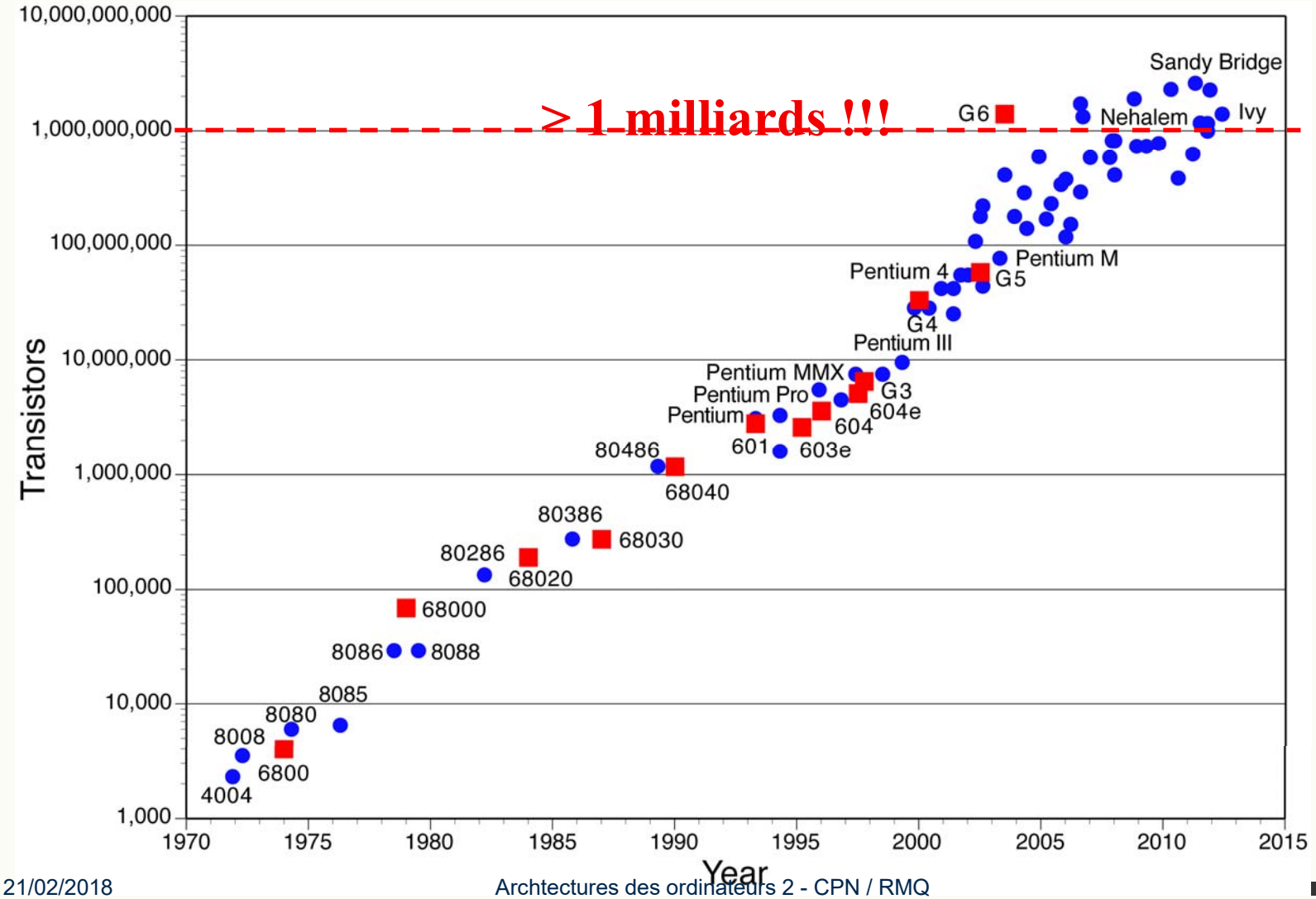

Nombre de transistors
100 000 000 000
10 000 000 000
1 000 000 000
100 000 000
10 000 000
1 000 000
100 000
10 000
1 000
1970
1975
1980
1985
1990
1995
2000
2005
2010
Année
4004
8088
80286
Intel386
Intel486
Pentium
Pentium Pro
Pentium II
Itanium
Pentium 4
Pentium 4 HT
Pentium III
Itanium 2 (1,5 Mo)
Itanium 2 (9 Mo)
Loi de Moore
Double tous les 18 mois
Processeurs Intel

~2‘000‘000‘000 transistors @ 3GHz



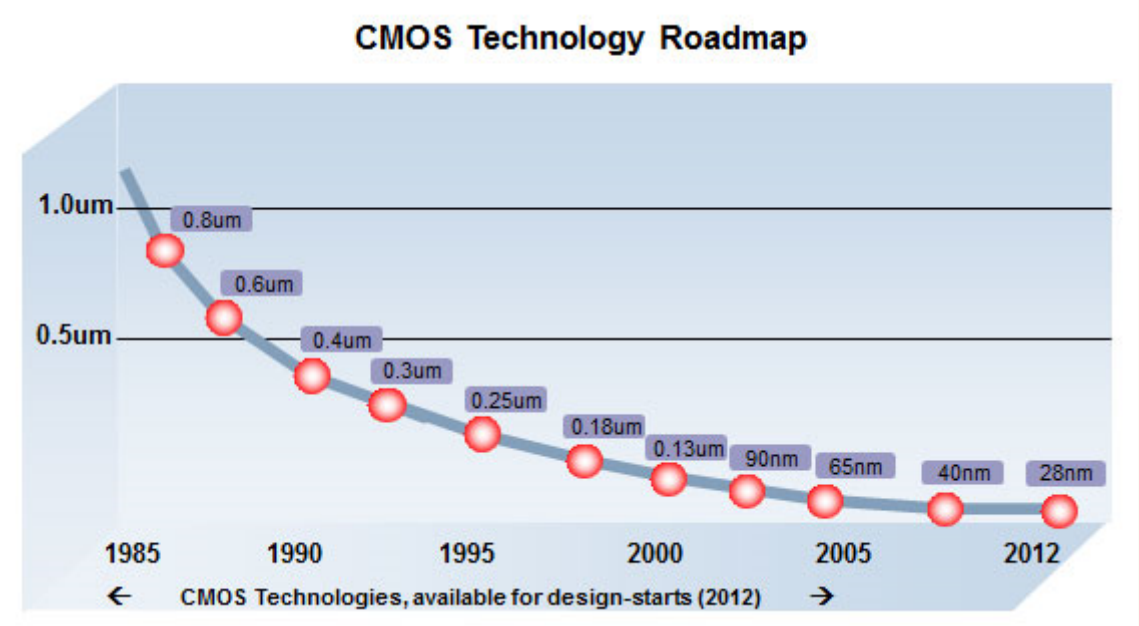
~2‘000 transistors @ 100KHz

Technologie: évolution nombre de transistors



Technologie: évolution CMOS

- **Roadmap Toshiba,**



http://www.toshiba-components.com/asic/Technology.html

Evolution des processeurs

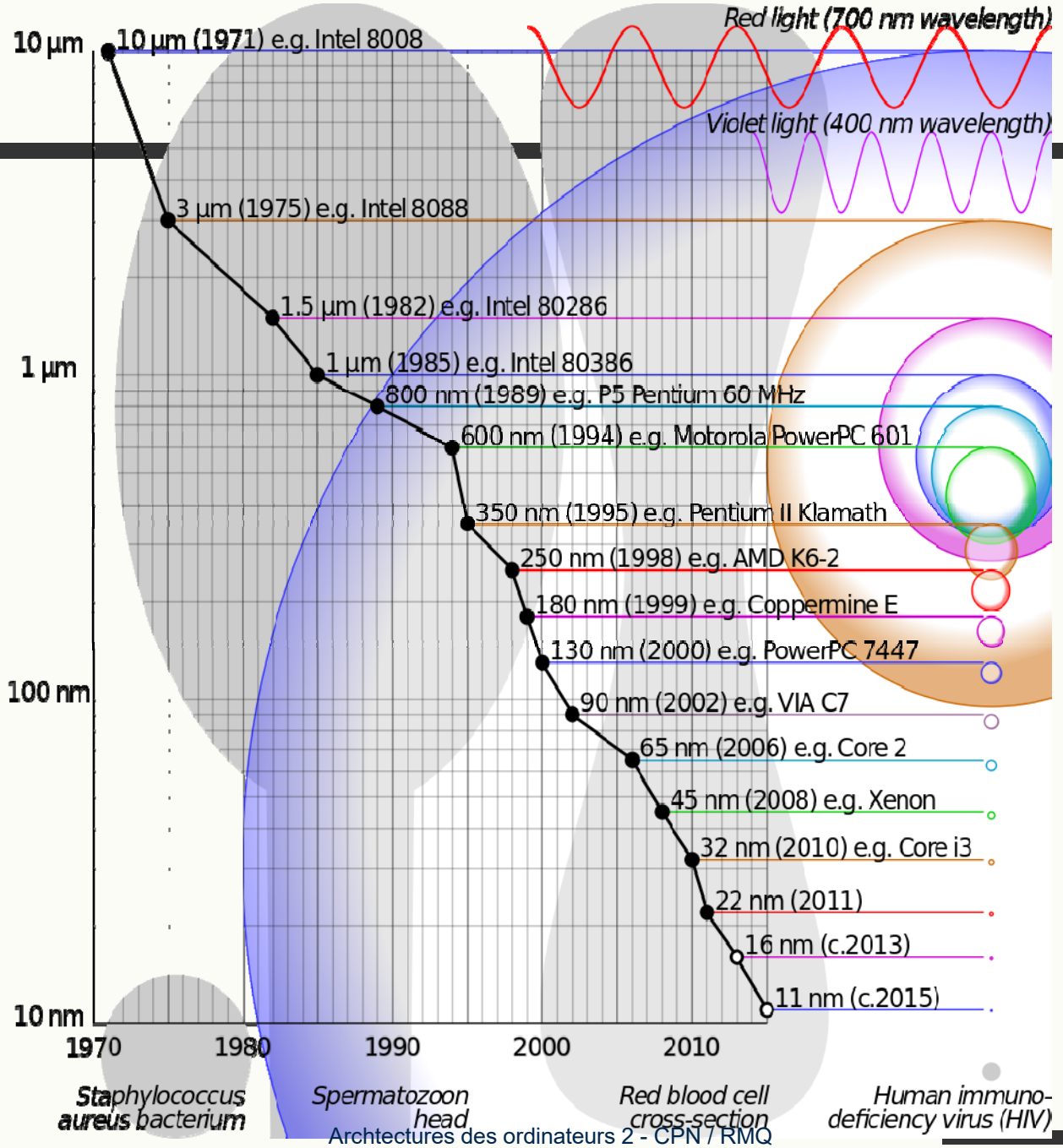
Date	Description	Transistors
1971	Intel 4004, CPU 4 bits, 740 KHz Adressage 640 Bytes, 0.06 Mips	2'300 10 µm
1979	Motorola 68000, CPU 32 bits, 8 MHz Adressage 16 MB, 0.7 Mips	68'000 3.5 µm
1990	Motorola 68040, CPU 32 bits, 50MHz, Adressage 4 GB, MMU, FPU, 44Mips	1.2 million 0.8 µm
1999	Pentium III, CPU 32 bits, 500MHz Adressage 4 GB, MMU, L1 cache 16KB	9.5 million 0.25 µm
2005	Pentium D, CPU 64 bits, Dual cores, 2.8GHz	290 million 0.09 µm
2008	Core i7 (Bloomfield), CPU 64 bits, 4 cores, 2.8GHz	781 million 0.045 µm

- **Ce qui est vrai, c'est qu'Intel introduit un nouveau processus de fabrication tous les 2 ans:**
  - **2000: 0.13 µm**
  - **2002: 90 nm**
  - **2006: 65 nm**
  - **2008: 45 nm**
  - **2010: 32 nm**
  - **2012: 22 nm**
  - **2014: 14nm**
  - **2016: 10nm**
  - **2018: 7nm**

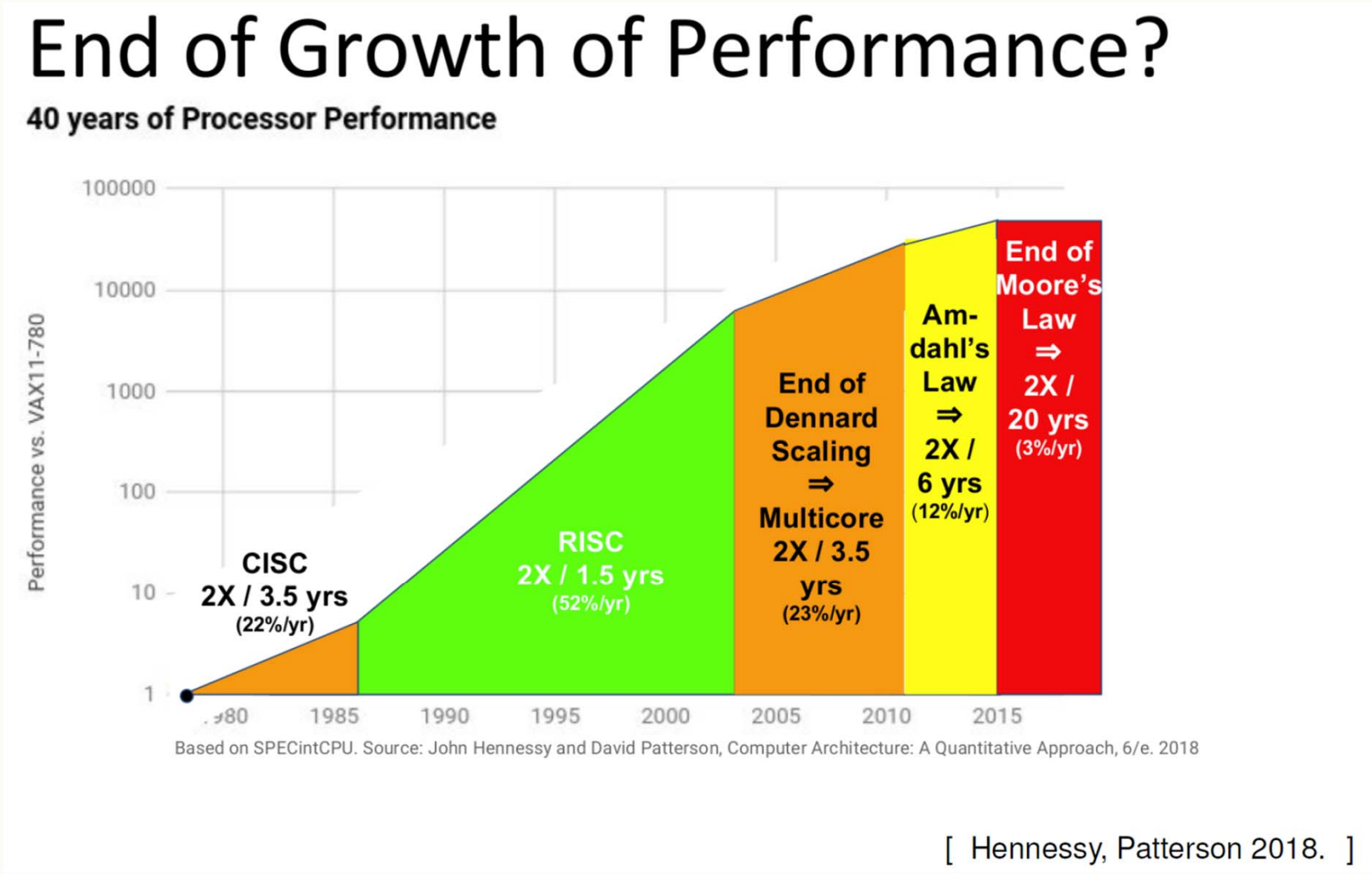
Sandy bridge

Ivy bridge de Intel
(3D transistors)

Processeurs Core M de Intel
(Q4 2014)

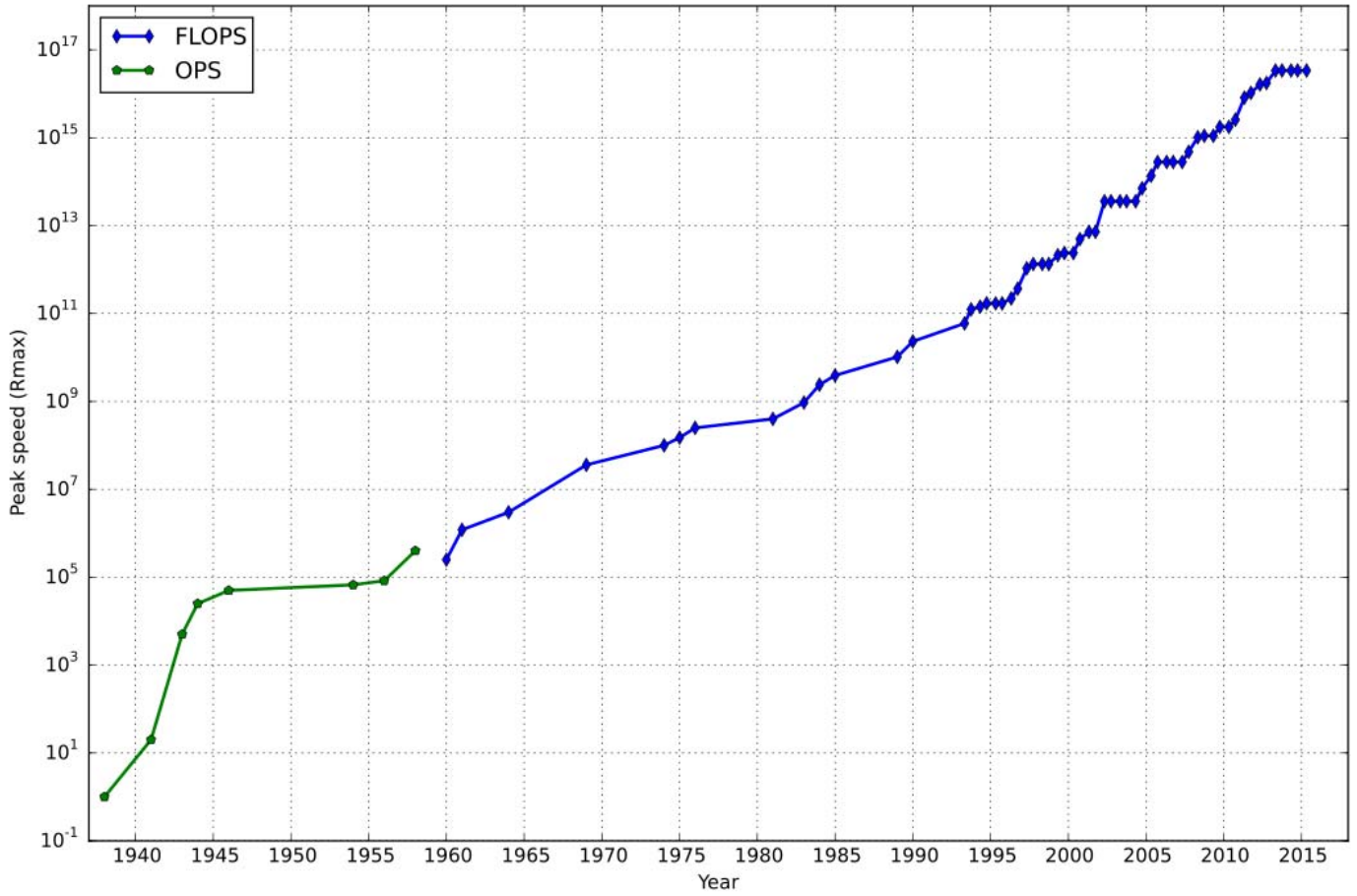

10 µm
1 µm
100 nm
10 nm
1970
1980
1990
2000
2010
10 µm (1971) e.g. Intel 8008
3 µm (1975) e.g. Intel 8088
1.5 µm (1982) e.g. Intel 80286
1 µm (1985) e.g. Intel 80386
800 nm (1989) e.g. P5 Pentium 60 MHz
600 nm (1994) e.g. Motorola PowerPC 601
350 nm (1995) e.g. Pentium II Klamath
250 nm (1998) e.g. AMD K6-2
180 nm (1999) e.g. Coppermine E
130 nm (2000) e.g. PowerPC 7447
90 nm (2002) e.g. VIA C7
65 nm (2006) e.g. Core 2
45 nm (2008) e.g. Xenon
32 nm (2010) e.g. Core i3
22 nm (2011)
16 nm (c.2013)
11 nm (c.2015)
Red light (700 nm wavelength)
Violet light (400 nm wavelength)
Staphylococcus aureus bacterium
Spermatozoon head
Red blood cell cross-section
Human immuno-deficiency virus (HIV)

Loi de Moore



[Hennessy, Patterson 2018.]

Loi de Moore


FLOPS
OPS
Peak speed (Rmax)
10^{17}
10^{15}
10^{13}
10^{11}
10^{9}
10^{7}
10^{5}
10^{3}
10^{1}
10^{-1}
1940
1945
1950
1955
1960
1965
1970
1975
1980
1985
1990
1995
2000
2005
2010
2015
Year

EXECUTION D'UN PROGRAMME

Programme

- Un programme est une suite d'instructions stockés dans une mémoire
- Les opérations arithmétiques et logiques sont exécutées [en fonction du programme] sur des données stockées en mémoire
- Les instructions ou les données sont stockées sous forme de mots binaire

La programmation


Recette de biscuits Shabby

Bonjour les gourmandes !
à la demande generale voici ma recette personnel de biscuits Shabby Evidemment cela reste entre nous, bien entendu ^ ^

250g de farine
125g de sucre
125g de beurre
1 oeuf
1 cs d'eau de rose
1 pincée de sel

Melanger le sucre et la farine, le sel puis l'oeuf et l'eau de rose
Ajouter le beurre en pommade et pétrir rapidement
Etaler la pâte afin d'obtenir une epaisseur de 3/4mm
Decouper les fleurs avec votre emporte pièce (moitié des fleurs avec le trou au milieu)
mettre au four pré-chauffer 10 mn à 180°
une fois les biscuits refroidis décorer le dessus de la fleur au glacage rose
Une fois sec fourrer les biscuits à la confiture de rose

Voilà les filles,
à vous de faire vos variantes (fraise, etc) et régalez-vous !

G.

Recette
programme informatique
séquence d'instructions

Décodage &
Exécution

- Un algorithme est un ensemble d'instructions qui, exécutées dans la bonne séquence, résolvent un problème dans un temps fini. Une recette de cuisine est un exemple d'algorithme
- Un programme est un algorithme écrit pour exécution dans un ordinateur

Logiciel (exemple en C)

```
int data = 0x123456;
int result = 0;
int mask = 1;
int count = 0;
int temp;

while (count < 32) {
  temp = data & mask;
  result = result + temp;
  data = data >> 1;
  count = count + 1;
};

/* result = 9 */
```

variables

data:	0x123456
result:	0
mask:	1
count:	0
temp:	undef

opérations et affectations

contrôle

Compilation et assemblage de l'exemple précédent en C

```
int data = 0x123456;
int result = 0;
int mask = 1;
int count = 0;
int temp;

while (count < 32) {
  temp = data & mask;
  result = result + temp;
  data = data >> 1;
  count = count + 1;
};
```

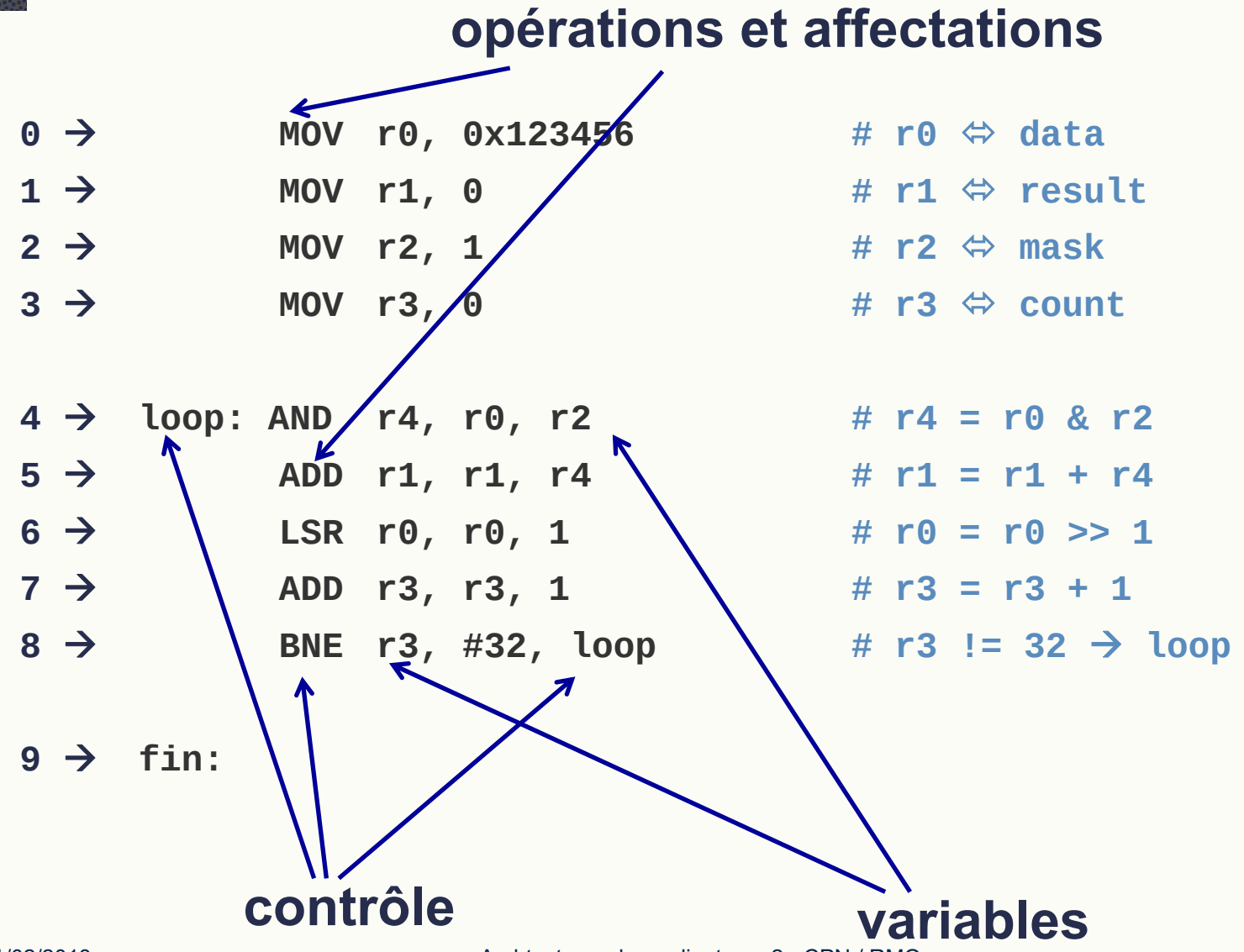
```
   MOV r0, #0x123456
   MOV r1, 0
   MOV r2, #1
   MOV r3, 0

loop:
   AND r4, r0, r2
   ADD r1, r1, r4
   LSR r0, r0, #1
   ADD r3, r3, #1
   BNE r3, #32, loop
fin:
```

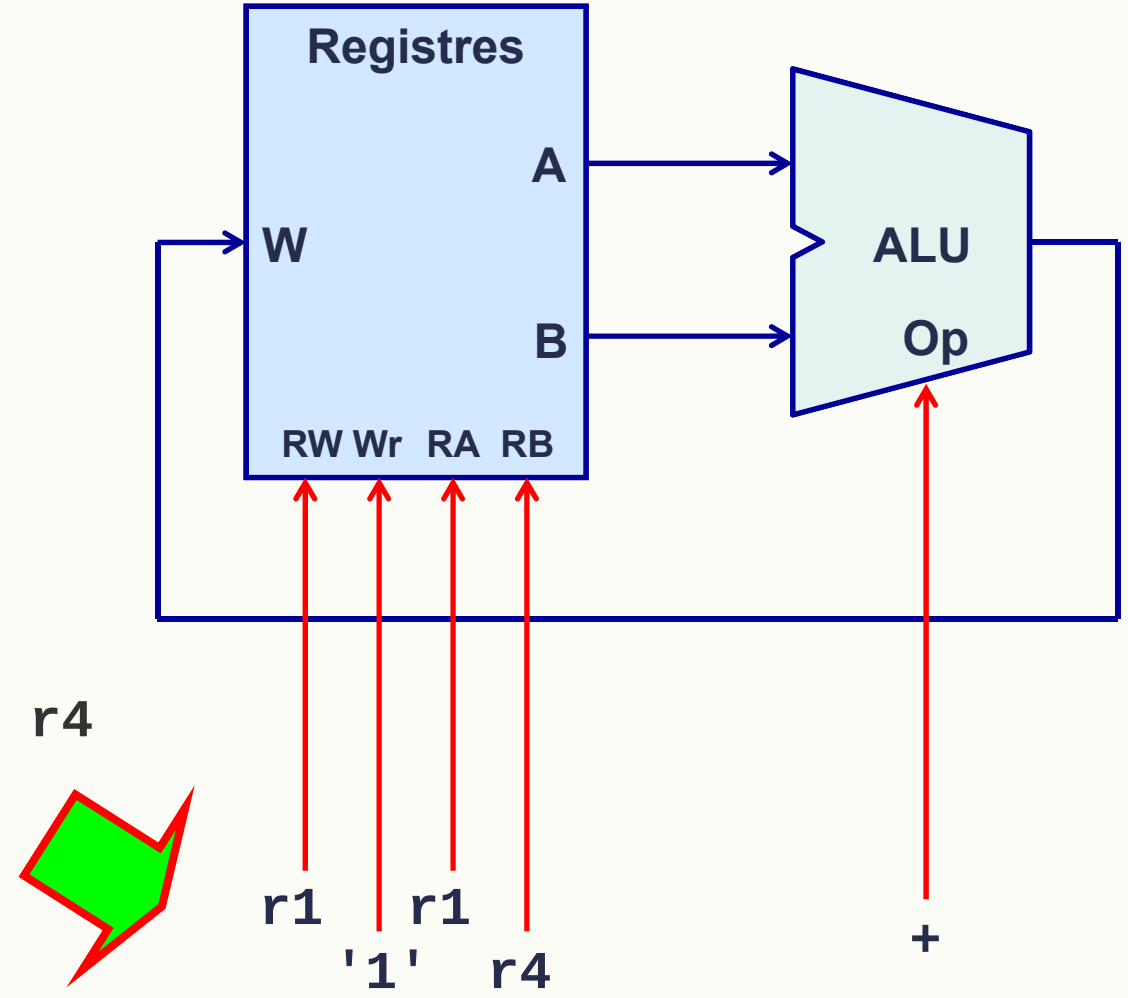
```
1010 1111
1011 1111
0100 0000
0011 0010

1010 1111
1011 1111
0101 0000
0011 0110
0010 0111
```

Code assembleur après compilation de l'exemple précédent en C

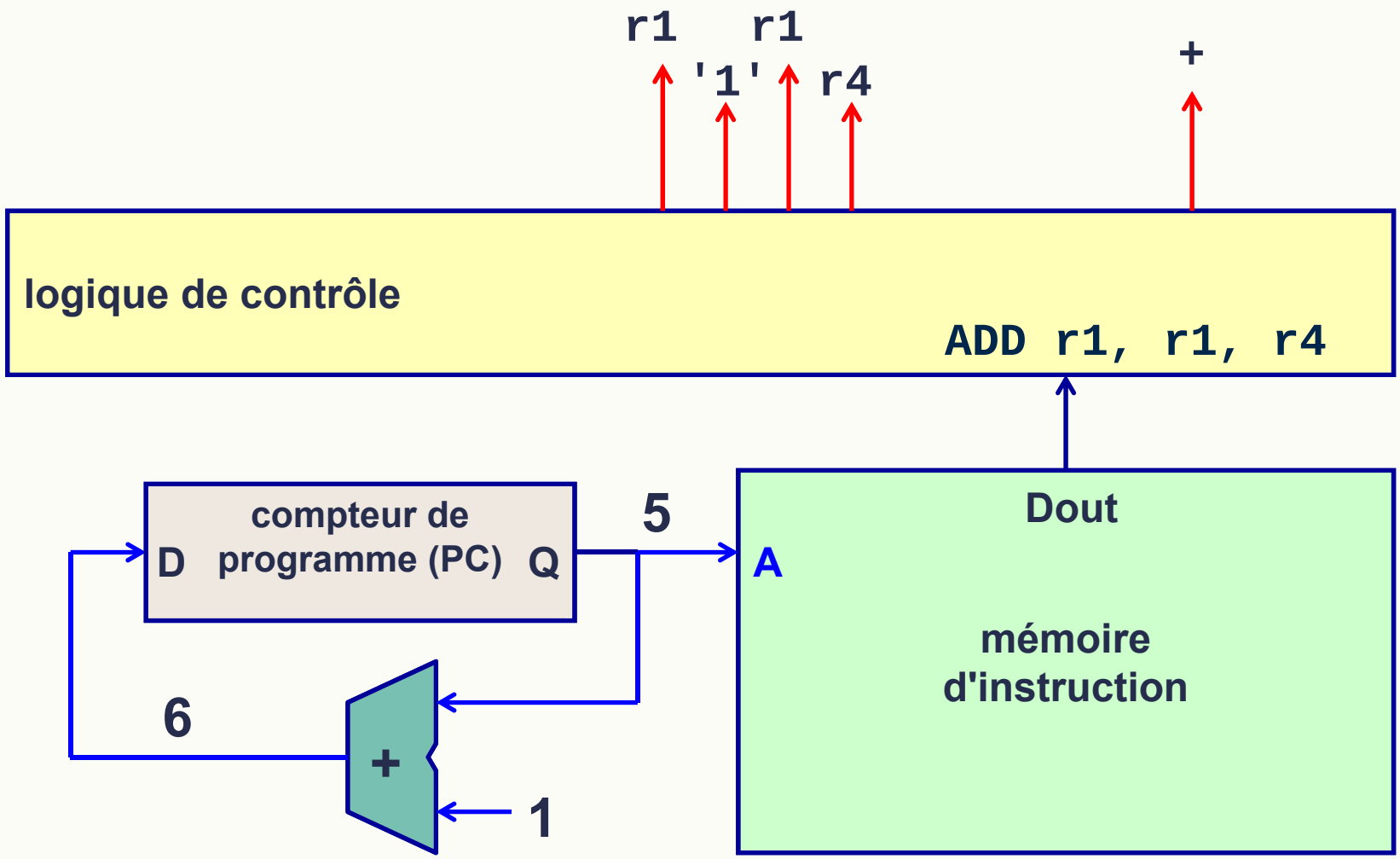


Opération arithmétique: addition : 5 – ADD r1, r1, r4

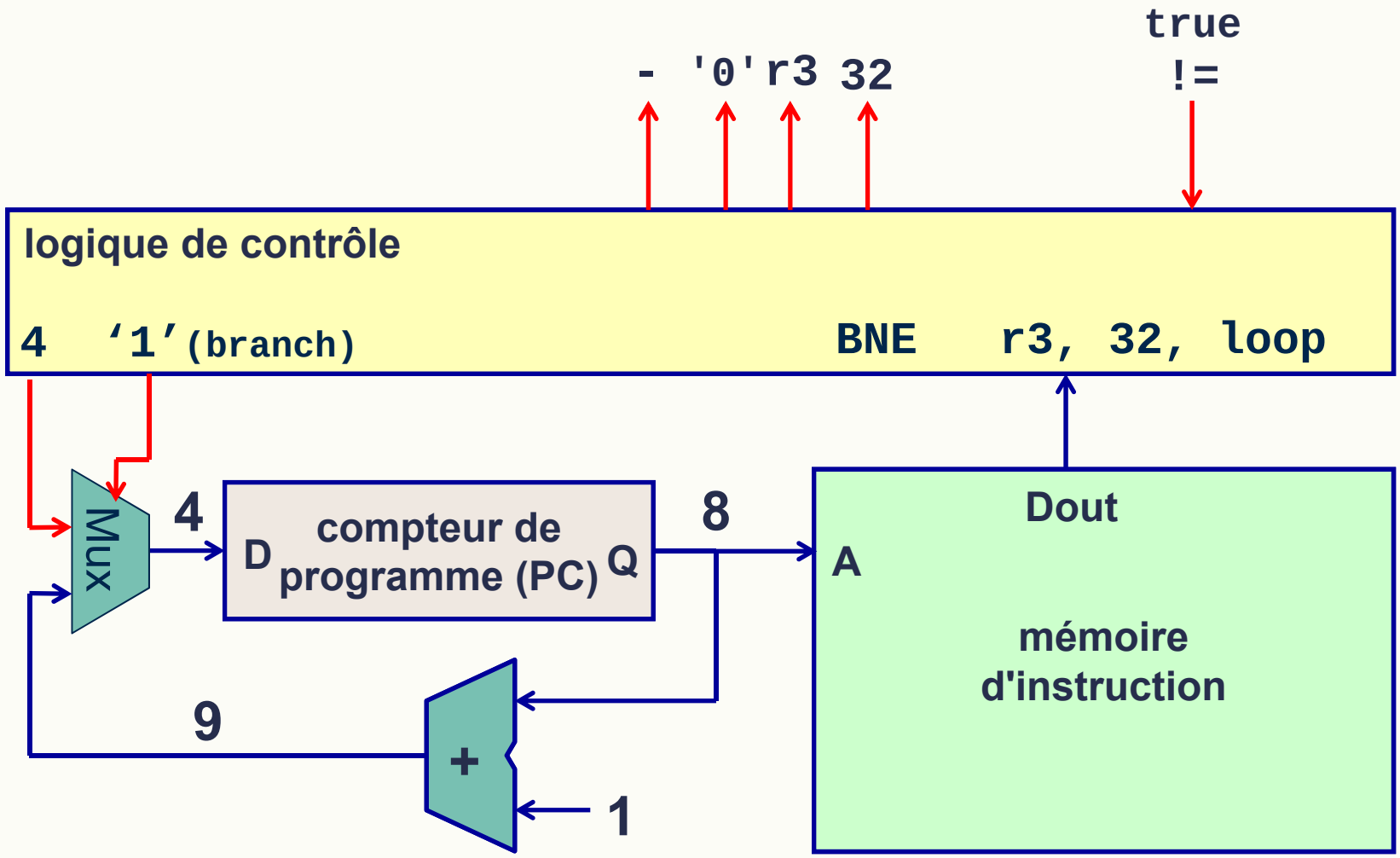



```
5 → ADD r1, r1, r4
```

Opération arithmétique: addition : 5 – ADD r1, r1, r4



Opération de contrôle : saut conditionnel : 8 – BNE r3, 32, loop



CODE D'INSTRUCTION ET TRAITEMENT

Langage machine

- Les processeurs doivent reconnaître des instructions codifiées sous la forme de groupes de bits
- L'ensemble des instructions reconnues par un processeur et son système de codage forment ce qu'on appelle le langage machine du processeur
- Il y a trois grands types d'instruction:
  - transfert de données
  - opérations arithmétiques/logiques
  - contrôle

Types d'instruction

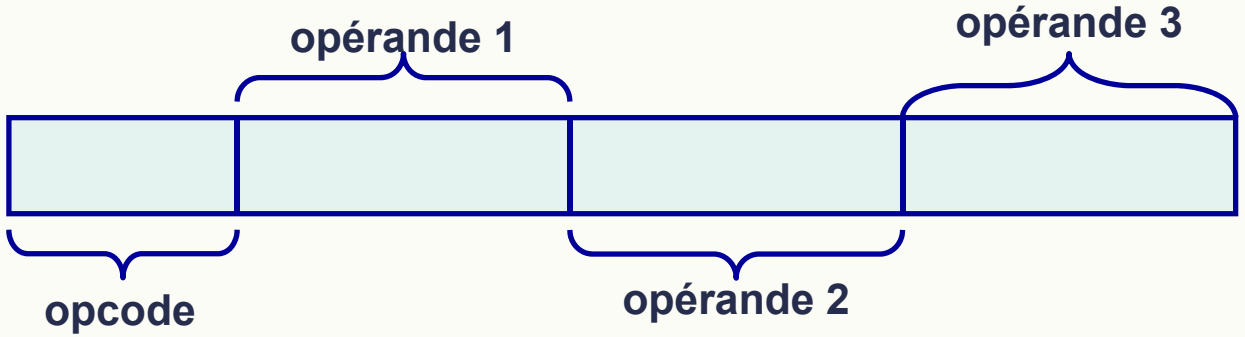
- Les instructions d'un CPU peuvent être classées en trois catégories :
  - instructions de calcul/traitement (arithmétique et logique)
  - instructions de transfert de donnée
    - **interne ⇔ interne, interne ⇔ externe**
  - instructions de branchement (saut)

Jeu d'instructions

- Chaque processeur à un jeu d'instructions, qui comporte l'ensemble d'opérations que le processeur peut exécuter.
- Chaque instruction est identifiée avec un code, appellé opcode (e.g., 00010, 00011) et avec un mnémonique (e.g., ADD, SUB).
- Une suite d'instructions d'un processeur donne lieu à un programme en langage machine

Format d'une instruction (1)

- Le code contient plusieurs *champs*
  - Le champ «opcode» est l'identificateur de l'instruction
  - Les champs opérandes spécifient la destination et les deux opérandes pour le traitement
    - Le nombre d'opérandes peut varier de 1 à 3.
    - Ci-dessous exemple avec le cas général
- Processeur ARM :
  code d'instruction 16 bits (ARM Thumb)



Format d'une instruction (2)

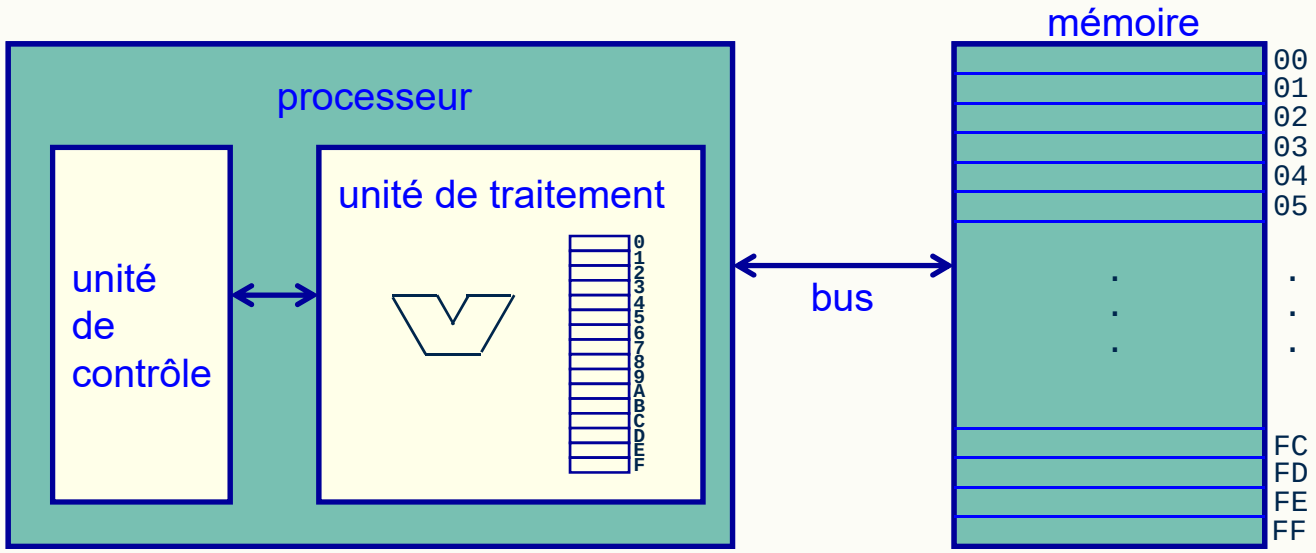
- **Instructions à 3 opérandes**
  Exemple : ADD r1, r2, r3 => r1 = r2 + r3

- **Instructions à 2 opérandes**
  Exemple : ADD r1, r2 => r1 = r1 + r2

- **Instructions à 1 opérande**
  Exemple : INC r1 => r1 = r1 + 1

Exemple

- Supposons un processeur traitant des données à 8 bits, avec 16 registres internes et une mémoire externe avec 256 positions ou cellules de stockage
- Les registres et les positions de mémoire reçoivent des adresses à partir de 0

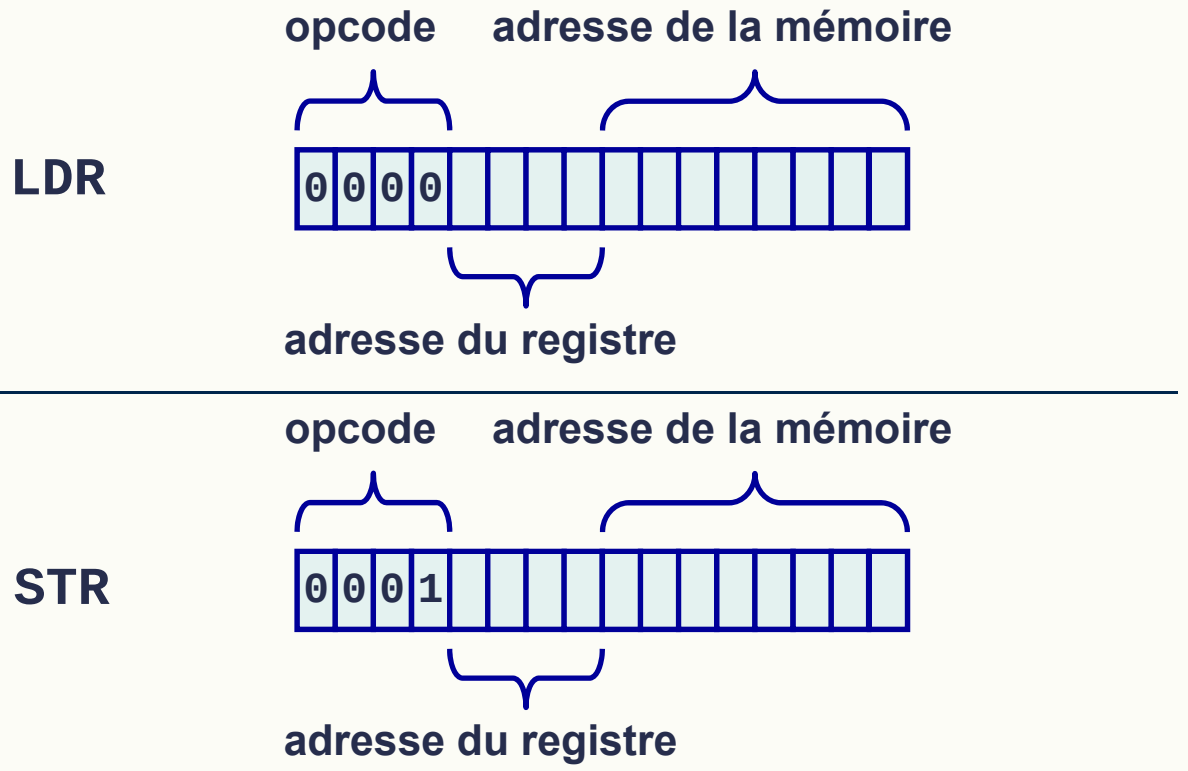


Division : exemple de code assembleur

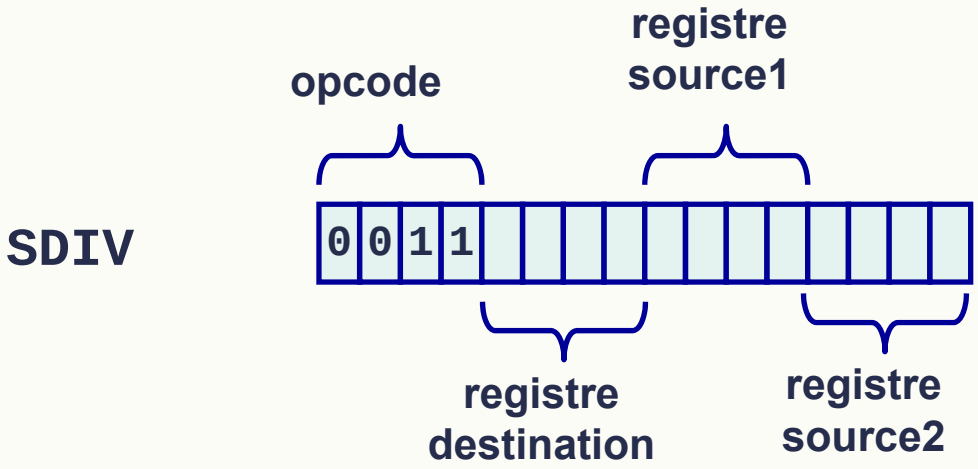
- Division de deux données stockées en mémoire :
  - 1: charger un registre avec la première donnée: LDR
  - 2: charger un autre registre avec la deuxième donnée: LDR
  - 3: si la deuxième valeur est zéro, aller au pas 6: BEQ
  - 4: diviser les contenus des deux registres et stocker le résultat dans troisième registre : SDIV
  - 5: sauver le résultat dans une position de mémoire: STR
  - 6: arrêter

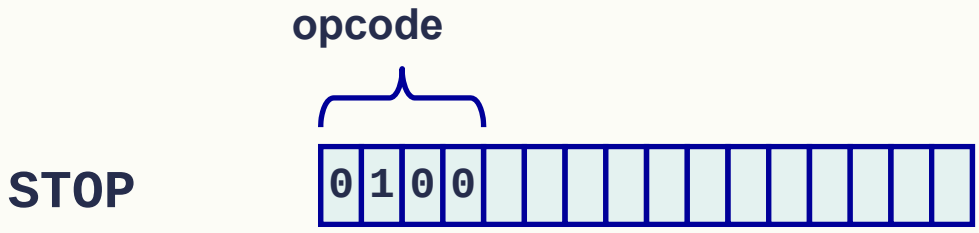
```
1 LDR   R0, M[0]      R0 ← M[0]
2 LDR   R1, M[1]      R1 ← M[1]
3 BEQ   zéro, 6       si 0 aller à 6
4 SDIV  R2, R0, R1    R2 ← R0/R1
5 STR   M[2], R2      M[2] ← R2
6 STOP
```

Codes d'instructions (1)



Codes d'instructions (2)



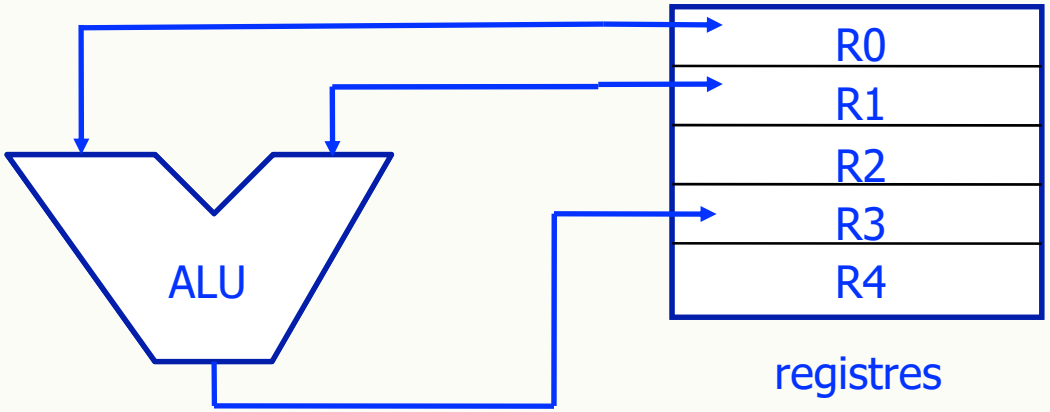


Division : codes hexadécimaux

Mnémoniques			Codes hexadécimaux
1	LDR	R0,M[0]	0000
2	LDR	R1,M[1]	0101
3	BEQ	zéro,6	2006
4	SDIV	R2,R0,R1	3201
5	STR	M[2],R2	1202
6	STOP		4000

Traitement

- Le traitement des données est effectué à l'intérieur du processeur par l'ALU, qui peut faire des opérations arithmétiques et logiques de base entre deux opérandes (addition, soustraction, et, ou, décalage, etc)
- Les deux opérandes de l'ALU et son résultat sont stockés dans les registres internes du processeur



- L'accès au registres est plus rapide qu'à la mémoire. Mais les registres sont peu nombreux (par exemple, 32 dans le cas de l'architecture MIPS)
- Comme les données se trouvent initialement dans la mémoire, il faut une instruction pour les ramener dans les registres: c'est l'instruction LOAD
- L'instruction LOAD utilise deux paramètres: l'adresse de la mémoire que l'on veut lire et le numéro du registre
- Exemple: si la donnée toto se trouve à la position 20 de la mémoire et nous voulons la stocker dans le registre 4 du processeur, il faut exécuter l'instruction

LOAD R4, 20

- Exemple: si nous voulons additionner les variables `toto` et `titi`, qui se trouvent dans les positions (adresses) 20 et 21 de la mémoire, il faut d'abord les ramener dans les registres internes du processeur
- Les instructions:

```
LOAD R4, 20
LOAD R5, 21
```

ramènent `toto` et `titi` dans les registres 4 et 5 du processeur, respectivement

- Une fois dans les registres internes, le processeur peut donner l'ordre à l'ALU d'effectuer l'addition: c'est l'instruction ADD
- L'instruction ADD utilise trois paramètres: les deux registres source et le registre destination, où le résultat de l'addition sera sauvegardé
- Par exemple, l'instruction

  ADD R0, R4, R5

  réalise l'addition de R4 avec R5 et stocke le résultat dans R0

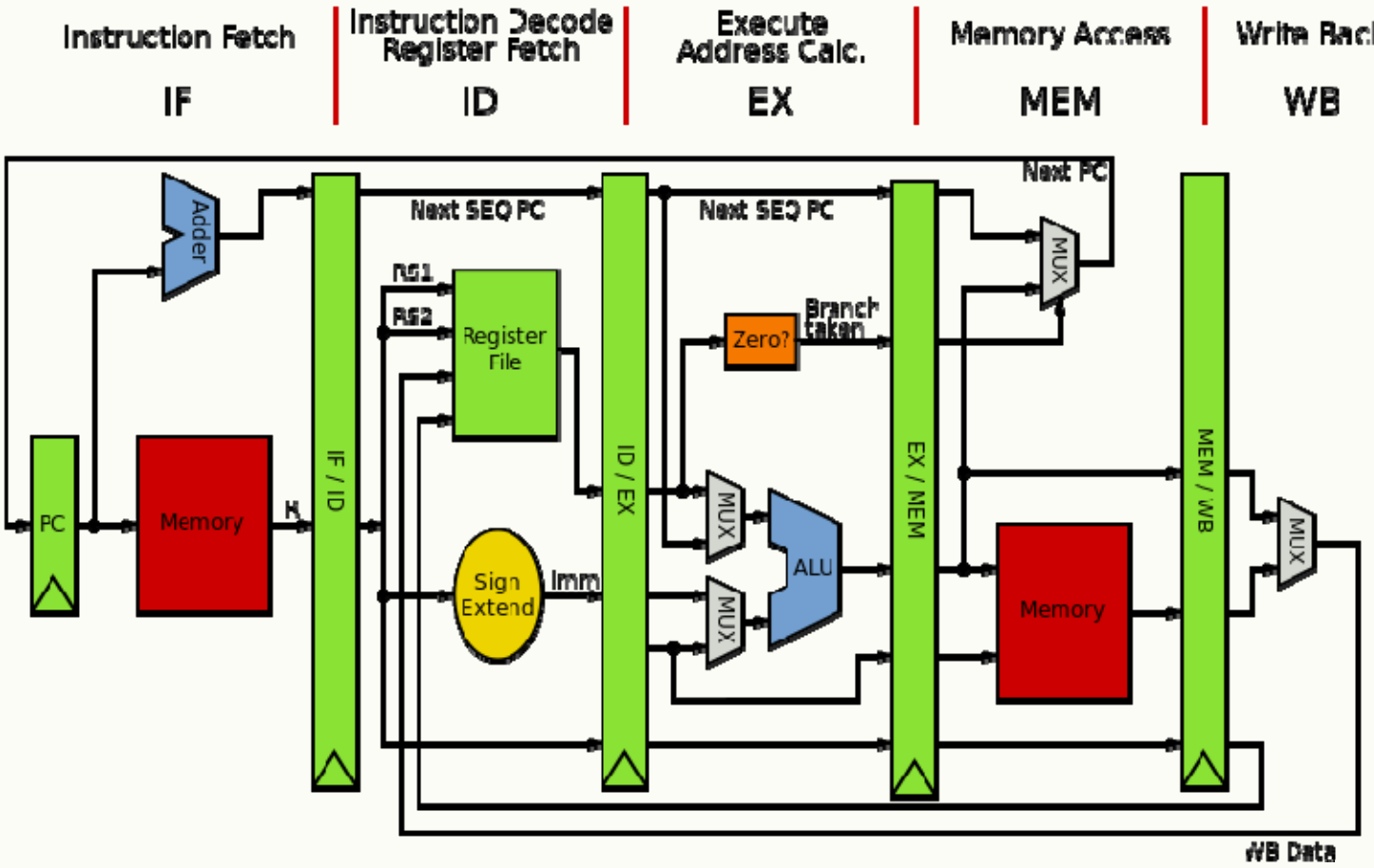
- Si l'opération voulue était

  ```
  toto = toto + titi
  ```

  il faut écrire le contenu du registre `R0` (le résultat de l'addition) dans la position 20 de la mémoire
- L'instruction **`STORE`** modifie une position de mémoire avec le contenu d'un registre
- Notre programme complet est maintenant:

```
LOAD  R4, 20
LOAD  R5, 21
ADD   R0, R4, R5
STORE 20, R0
```

• Exemple d'une architecture de processeur MIPS



Exécution d'une instruction

- Un processeur effectue sans arrêt une boucle composée de trois phases:
  - recherche (*fetch*) de l'instruction: l'adresse en mémoire de l'instruction à exécuter est stockée en permanence dans un registre du processeur, appelé PC (*Program Counter*). L'instruction pointée par le PC est cherchée dans la mémoire et stockée dans un autre registre du processeur: le IR (*Instruction Register*)
  - décodage de l'instruction (*decode*): chaque instruction est identifiée, grâce à un code (*opcode*). En fonction de ce code, le processeur choisit la tâche à exécuter, c'est-à-dire la séquence de micro-instructions à exécuter
  - exécution (*execute*) de l'instruction: à la fin de cette phase, on retourne à la première phase

Traitement d'une instruction

Le traitement s'effectue en plusieurs phases :

1. FETCH (*recherche de l'instruction)*: l'adresse de l'instruction à exécuter est fournie à la mémoire d'instruction. La mémoire retourne le code de l'instruction

2. DECODE *(décodage de l'instruction)*: chaque instruction est identifiée, grâce à un code (*opcode*). En fonction de ce code, le processeur choisit la tâche à exécuter

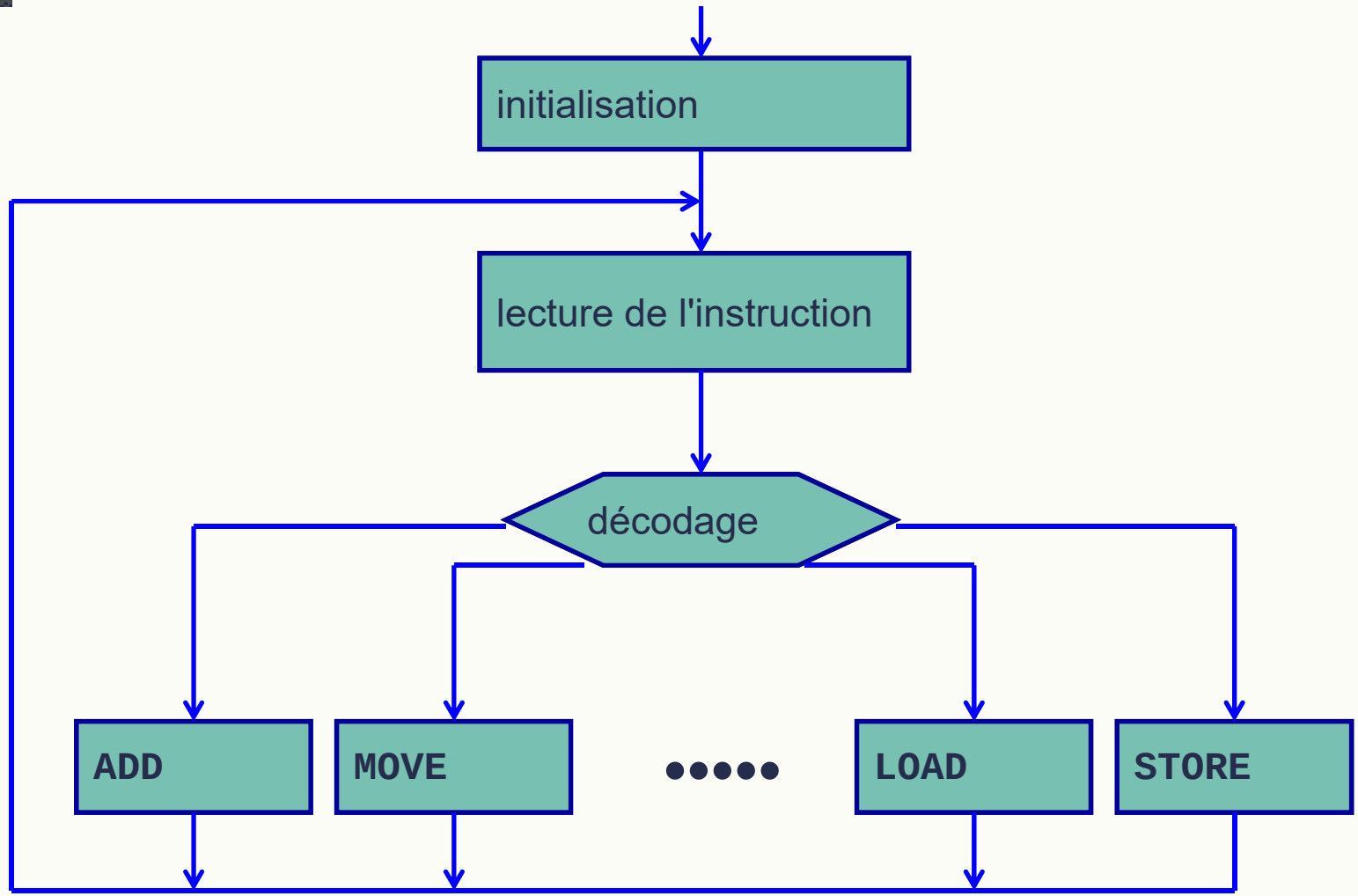
3. EXECUTE (*exécution de l'instruction*) : une opération arithmétique ou logique est effectuée

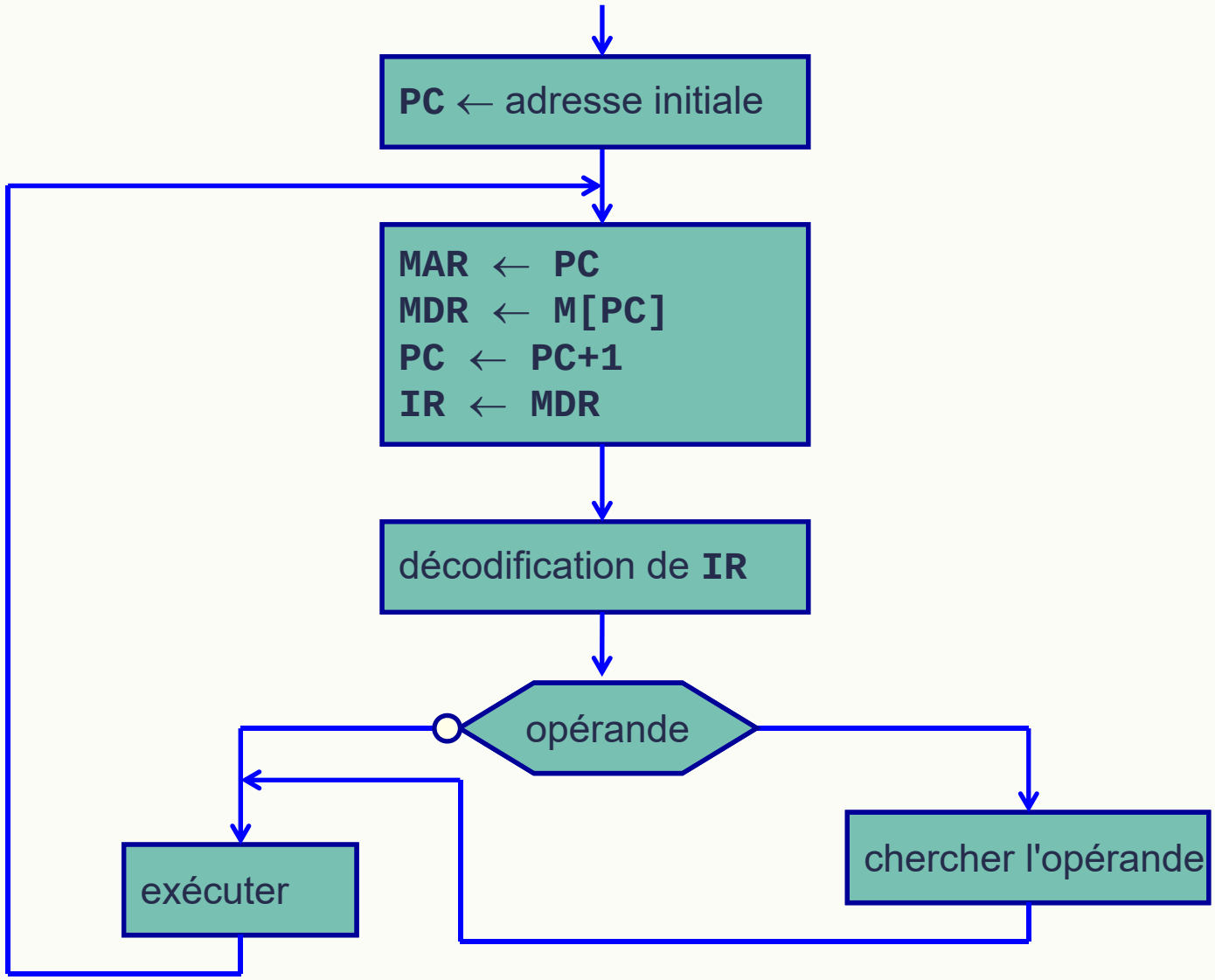
4. MEMORY (*accès mémoire*) : une donnée est lue ou écrite dans la mémoire

5. WRITE BACK : le résultat d'une opération ou la valeur lue en mémoire sont écrits dans un registre

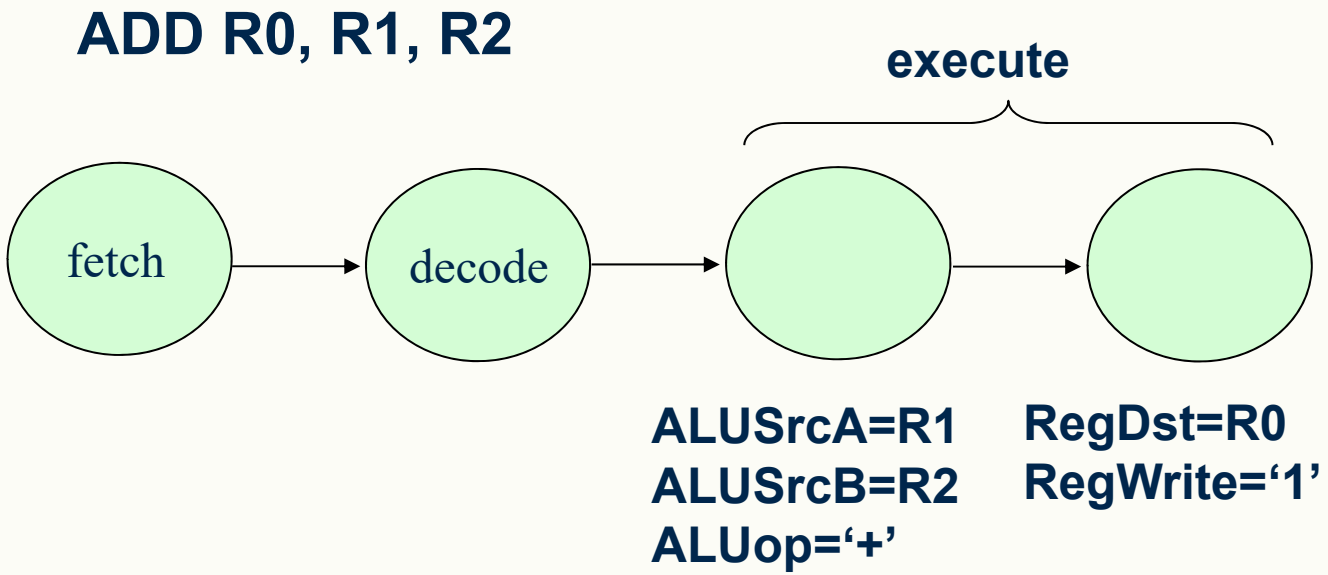
L'unité de contrôle d'un processeur

- L'unité de contrôle doit gérer l'exécution des opérations de base du processeur (c'est à dire du jeu d'instructions).
- Chaque instruction du processeur est exécutée en plusieurs phases, prenant plusieurs cycles d'horloge.
- Chaque instruction est d'abord cherchée dans la mémoire de l'ordinateur (fetch), après elle est décodée (decode) et enfin exécutée (execute).
- La phase execute, peut prendre plusieurs cycles d'horloge (e.g., la multiplication séquentielle).


initialisation
lecture de l'instruction
décodage
ADD
MOVE
LOAD
STORE


PC ← adresse initiale
MAR ← PC
MDR ← M[PC]
PC ← PC+1
IR ← MDR
décodification de IR
opérande
exécuter
chercher l'opérande

Exemple d'exécution d'une instruction



Exercice

- Supposez que vous voulez multiplier deux variables a et b, stockées dans les positions de mémoire `M[20]` et `M[21]`, respectivement, pour affecter cette valeur à la variable `toto`, stockée à la position de mémoire `M[40]`
- C'est-à-dire, vous voulez effectuer l'opération:

```
toto = a*b
```

ou:

```
M[40] = M[20]*M[21]
```

- Le processeur possède 8 registres (R0...R7). Le registre R0 contient toujours la valeur 0
- Les instructions du langage machine du processeur sont:
  - LOAD Rd, M[adr] — Rd ← M[adr]
  - STORE M[adr], Rs — M[adr] ← Rs
  - ADD Rd, Rs1, Rs2 — Rd ← Rs1 + Rs2
  - SUB Rd, Rs1, Rs2 — Rd ← Rs1 – Rs2
  - DEC R — R ← R - 1
  - JUMP zero, adr — si zero alors sauter à adr
- Avant d'écrire le programme, nous devons trouver un algorithme réalisant la tâche voulue

- Un algorithme possible serait: